UNIVERSITY OF CALIFORNIA
SANTA CRUZ

MULTIVARIATE NEAREST-NEIGHBORS GAUSSIAN PROCESSES WITH RANDOM COVARIANCE MATRICES

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

STATISTICAL SCIENCE

by

Isabelle Grenier

December 2021

The Dissertation of Isabelle Grenier
is approved:

Professor Bruno Sansó, Chair

Professor Zehang (Richard) Li

Professor Athanasios Kottas

Peter Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Multivariate Nearest-Neighbors Gaussian Processes with Random Covariance Matrices

by

Isabelle Grenier

Computational efficiency is at the forefront of many cutting edge spatial modeling techniques. Non-stationarity, on the other hand, has often been an unintentional feature of the approximations used in these spatial models. Deriving from the well known multivariate linear regression model, we propose a non-stationary and non-isotropic spatial model. In order to remain relevant with today's massive datasets challenges, we apply the concept of nearest-neighbors to our normal-inverse-Wishart framework. The model, called Nearest-Neighbor Gaussian Process with Random Covariance matrices (NN-RCM) is developed for both univariate and multivariate spatial settings and allow for specific characteristics such as duplicate observations and missing data. In a first dive into nearest-neighbor models such as Nearest-neighbor Gaussian Processes (NNGP), we initially look at divide-and-conquer implementations. Ultimately, we devise a technique to use NNGP models on blocks of data to then aggregate the posterior inference without loss of information. The models are illustrated in a case study of albedo assessments over CONUS from the Geostationary Operational Environmental Satellites (GOES) East and West. First, the divide-and-conquer algorithm is used to output multi-day successive predictions of albedo assessments in a sequential updating scheme. By applying the distributed model on each day of data, one can concatenate the posterior parameters of the blocks of interest to output evolving predictions. Finally, we apply the bivariate NN-RCM model using each satellite as a source of information. The objective is to merge the albedo assessments while also quantifying the discrepancy between the sources.

To Bibiane,

who would have loved to go school.

Acknowledgments

This work was done in collaboration with Dr. Jessica Matthews from the National Oceanic and Atmospheric Administration, with support from the National Science Foundation grants DMS-1513076 and DMS-1953168.

First and foremost, I would like to thank Dr. Bruno Sansó for his patience and his support throughout the entirety of my PhD degree. His wise and thoughtful advice about life will remain with me for the remaining of my career. I would like to show my gratitude to Dr. Athanasios Kottas and Dr. Zehang (Richard) Li for stepping in at a moment's notice and serve on my defense reading committee. I would like to acknowledge Dr. Amy Braverman for her input in my early research journey. Her enthusiasm and her work on distributed methods were the inspiration behind the first project. Finally, I would like to thank Dr. Maria Terres for her mentorship and her constant encouragement. Her tremendous support and advice have helped me see this through and continue to strive beyond this work.

The text of this dissertation includes reprints of the following previously published material:

Isabelle Grenier and Bruno Sansó. Distributed nearest-neighbor Gaussian processes. *Communications in Statistics - Simulation and Computation*, 0(0):1–13, 2021. doi: 10.1080/03610918.2021.1921798. URL `https://doi.org/10.1080/03610918.2021.1921798`.

The co-author listed in this publication directed and supervised the research which forms the basis for the dissertation.

Chapter 1

Introduction

The land surface albedo is the ratio between the upward and downward reflected solar radiation at the Earth's surface (NOAA, 2018). Quantifying the amount of light that hits the surface of the Earth without being reflected is essential for understanding climate change and its potential impact on human health. Missions such as the National Oceanic and Atmospheric Administration's (NOAA) Geostationary Operational Environmental Satellites (GOES) East and West have been designed to target specifically the forecasting of extreme weather events (e.g floods) and the monitoring of land process analysis (NOAA, 2016). In light of being a sensitive indicator of environmental changes, surface albedo has been classified an Essential Climate Variable (ECV) by the Global Climate Observing System (GCOS).

Figure 1.1 shows a few examples of the range of observed values for different land surfaces. For example, the reflectance on snow is close to 90% as opposed to the reflectance on dark roads which can be as low as 5 to 20%.

Surface albedo can be measured by a variety of instruments including ground-based stations and satellites. As we can expect, each platform has its advantages and disadvantages. Ground-based stations have fine temporal and spatial resolution but the available networks are very sparse. In terms of satellites, polar orbiting satellites have

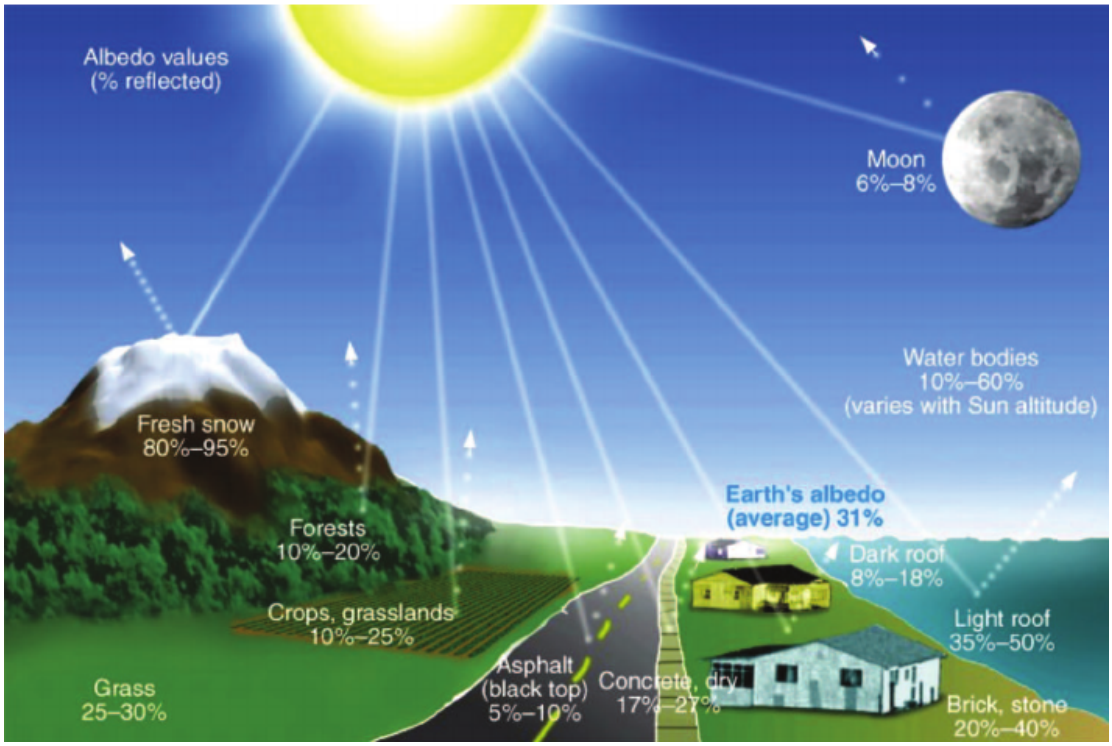


Figure 1.1: Reflectance over different surfaces. Courtesy of Univ. of Idaho, College of Science.

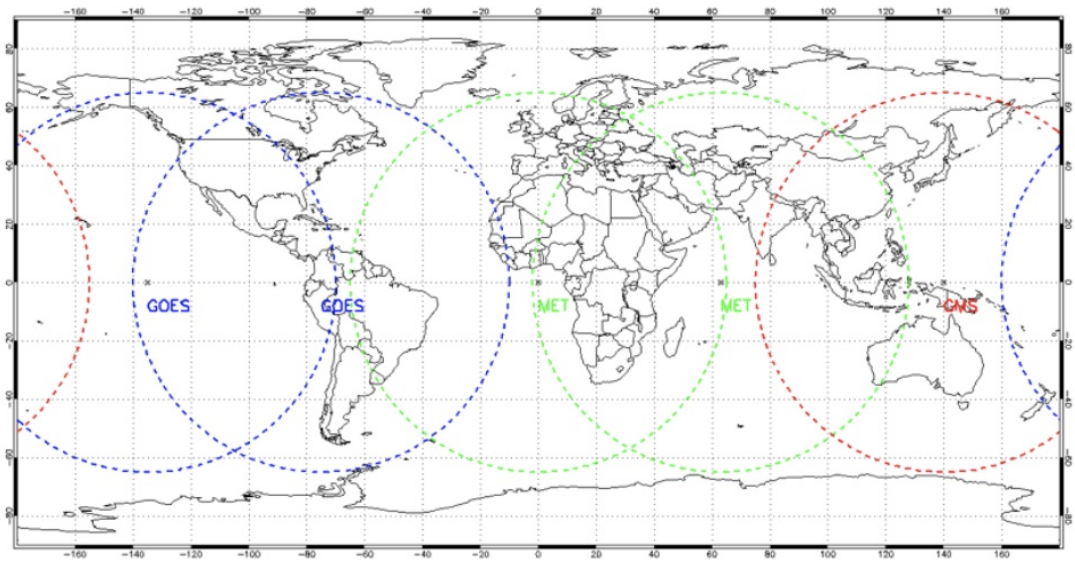


Figure 1.2: Illustration of the viewing angle limit of the five world satellites. Blue: GOES from NOAA. Green: Meteosat satellites from EUMETSAT. Red: Geostationary Meteorological Satellite (GMS) from JMA. Figure adapted from Govaerts et al. (2008).

global coverage but their biweekly temporal resolution is very coarse. On the other hand, geostationary satellites have excellent temporal resolution but they only provide coverage for a fixed window.

To resolve the shortcomings of geostationary satellites, a global project called SCOPE-CM ("Sustained and COordinated Processing of Environmental satellite data for Climate Monitoring") has been set in place to merge surface albedo from different agencies and their satellites. The project includes collaborators from the United States (NOAA), Europe (EUMETSAT), Japan (JMA), Switzerland (MeteoSwiss), and Korea (KMA). The circles in Figure 1.2 show the viewing angle limit for each of the five world geostationary satellite.

1.1 Large spatial data literature review

Ultimately, we faced two main challenges in the development of a methodology to reconcile the albedo assessments from the two GOES satellites for NOAA. The most important challenge presented by this project is the size of the data at hand which must be at the forefront of all modeling decisions. Although the project focuses on the years 1995-2017, the GOES archives include data starting in 1978. The raw imagery obtained from the satellites form millions of files for a total size of 270TB. That daily surface albedo extracted from the original imagery consists of daily files for 23 years for each satellite. In other words, we have approximately 17,000 files. Moreover, an additional 1,000 files are created along with the daily values. These files contain a 10-day assessment which is calculated three times per month. This assessment is selected based on the quality rating of each albedo observation for the 10-day period. That is, the albedo observation with the highest quality rating for those 10 days is retained as the best 10-day albedo assessment. In sum, we are dealing with 18,000 files, or 3TB of data.

The land surface albedo is therefore an example of a new class of problems which arise from dealing with massive spatial datasets. The increased availability of georeferenced data has produced a need to handle, analyze and make inferences and predictions for very large collections of data, posing a large computational burden on model-based inference of spatial fields. Geostatistical methods that deal with point-referenced data consider random fields that are indexed in space, usually 2D or 3D. Intuitively, proximity between observations should provide information for inference about unobserved values of the field. This is often formalized using model-based approaches for which there is a solid body of literature and software, see, for example, the books by Cressie (1993); Gelfand et al. (2010); Cressie and Wikle (2011); Banerjee et al. (2014). Modern geostatistical approaches provide flexible probabilistic models, coupled

with powerful learning methods, that are used to investigate challenging inferential questions related to geographically-referenced data.

Traditionally, model-based spatial models have relied on the Gaussian process (GP). GPs capture the dependence due to proximity through a covariance function. For a likelihood-based approach to GPs, the bottleneck lies in the computation of the determinant and the inverse of the covariance matrix induced by the locations of the available observations. Matrix inversion has order n^3 operations, which can be costly for datasets with large number of observations n. To illustrate the problem with some numbers, an application for observations at, say, 1,000,000 locations will produce a covariance matrix with 500,000,500,000 possibly different values. Such a large data structure would need to be stored, decomposed and operated with, possibly within an iterative procedure.

Modern applications can have datasets that are orders of magnitude larger than that. To tackle this problem most current methods take one of two approaches: exploit sparsity in the structure of the covariance matrix or reduce the dimensionality of the problem by seeking representations of GPs on lower dimensional subspaces. In both cases the goal is to speed up calculations, as well as reduce the size of the objects that need to be handled and stored in memory when performing computations. An example of the former is covariance tapering that consists of truncating the covariance function to zero for distant observations (Furrer et al., 2006; Kaufman et al., 2008; Du et al., 2009; Shaby and Ruppert, 2012). By using an appropriate tapered correlation kernel, the covariance matrix becomes very sparse while remaining positive definite. An example of the latter is the predictive Gaussian processes, as introduced by Banerjee et al. (2008a), which represents the GP using basis functions generated by the covariance function. This produces a reduction of the dimension of the matrix that needs to be inverted to perform inference to a fixed, small dimension, that depends on a pre-specified set of knots. Other extensions of predictive processes are explored in Finley et al. (2009)

and Guhaniyogi et al. (2011).

Another dimensionality reduction technique is the process convolution approach which operates on a reference grid and averages each location from a set of basis kernels for the neighbouring knots (Higdon, 1998; Lemos and Sansó, 2009). Fixed rank kriging takes advantage of a special type of covariance function which allows one to solve exactly for the best linear unbiased spatial prediction (Cressie and Johannesson, 2008; Katzfuss and Cressie, 2011). Approaching the problem differently, circulant embedding uses data augmentation and fast Fourier transforms to compute the likelihood of the complete dataset Stroud et al. (2017); Guinness and Fuentes (2016).

Finally, one different approach to dimension reduction is the use of subregions. These models seek to fit Gaussian processes on each subregion to then obtain a global model through a hierarchical structure. Variants on this idea include Bayesian Treed Gaussian Process models (Gramacy and Lee, 2008), multi-resolution models (Katzfuss, 2017) and multi-scale models (Kirsner and Sansó, 2020).The balance between the scalability of these methods through block-independence and the ability to share knowledge between regions is challenging. Moreover, the inference and predictions obtained from the models can be sensitive to the varying creation of the subregions.

A number of other techniques have been investigated. A few of them include the spectral domain approximation by Terres et al. (2015) and Gaussian Markov random fields by Lindgren et al. (2011). For further information about the state of the art model-based geostatistics methods suitable for large data sets see Banerjee (2017) and Heaton et al. (2019).

In this work, we focus on nearest-neighbor spatial processes such as nearest-neighbor GP (NNGP) Datta et al. (2016). These models are particularly intriguing, as it blends features of both the dimension reduction and the sparsity approaches, formalizing to a Gaussian process framework the popular likelihood approximation proposed in Vecchia (1988). Nearest-neighbor processes uses the conditional distribution structure

of a joint likelihood to build a directed network of neighbors. By making observations conditionally independent of non-neighboring locations, the new precision matrix has a very sparse structure. In that case, the number of operations needed to invert the precision matrix is limited by the number of neighboring locations allowed.

1.2 Non-stationary spatial model literature review

The second challenge that is tackled with the proposed models is that of the non-stationarity of the land surface albedo. A spatial process $w(s)$ is said to be second-order stationary when

$$E(w(s)) = E(w(s+h)) = \text{constant}$$

and

$$C(w(s), w(s+h)) = C(h),$$

where h is the distance between two locations and C denotes the covariance function of the process. As expected, this assumption is too rigid for the land surface albedo which has a different correlation based on the orientation of the locations.

Previous approaches have been developed to model anisotropy in spatial fields. Starting with spatial deformations where the observed geographic locations are mapped to a warped space where stationarity holds (Sampson and Guttorp, 1992). One specific example of spatial deformation is discussed in Schmidt and O'Hagan (2003) where the model seeks to learn the true covariance matrix Σ of the Gaussian process using the observed covariance matrix S. In Brown et al. (1994), non-stationarity occurs from the choice of using an inverse Wishart prior for the covariance matrix within the general multivariate normal framework. Using a different approach, Paciorek and Schervish (2006) create a new general class of non-stationary covariance functions.

Many of the approaches for large spatial fields discussed in section 1.1 result in non-stationary processes naturally. Partition models like the ones introduced

in Gramacy and Lee (2008) and Kim et al. (2005) construct non-stationary models by averaging over locally stationary processes. Further examples include Fuentes (2001) and Fuentes and Smith (2003) where the non-stationary process is a discrete or continuous weighted average of stationary models for subregions of the space. Similarly, basis function models such as low-rank approximations (Katzfuss, 2013) and multi-resolution models (Nychka et al., 2015; Katzfuss, 2017), while heavily focused on the computational effort, induce non-stationarity unintentionally by their definition. Finally, process convolutions also result in non-stationary processes by the way they are constructed. In Higdon (1998), the use of basis kernel which is ultimately for computational reasons results in an approximated Gaussian process which is non-stationary. Another example of non-stationarity being owed to discrete convolutions processes is found in Lemos and Sansó (2009).

In Chapter 3, we leverage the inverse Wishart prior in the multivariate normal framework to set our initial focus on developing a non-isotropic covariance structure. Then, we adapt the model to handle large spatial datasets by including a nearest-neighbors idea.

1.3 Roadmap

In Chapter 2, we introduce a distributed implementation for NNGPs which tackles multiple issues of large datasets that are often overlooked. Specifically, the distributed approach suggested offers a solution for datasets with restricted data access or data movement. It is often assumed in model-based inferences that one can handle a dataset at once. As we see with increasingly large problems, this is no longer a reliable assumption. We start by reviewing the current state of distributed spatial modeling methodologies. Then, we develop and discuss the equivalence between the distributed full conditionals and the original full conditionals of the NNGP model. Finally, we

illustrate the model using simulated datasets up to 1,000,000 datapoints. Note that we return to the distributed approach in Chapter 5 by developing a sequential technique for daily predictions updates.

In Chapter 3, we introduce a univariate and a multivariate non-isotropic hierarchical nearest-neighbor model. For both the univariate and the multivariate settings, we develop two versions of what we call the nearest-neighbors Gaussian processes with random covariance matrices (NN-RCM). The first version leverages the marginal posterior distribution of the model and allows to obtain posterior inference of the covariance parameters efficiently. The second version builds a hierarchical structure which includes fixed effects and an observational error. This second model relies on the parallelization of the model which has a structure reminiscent of multivariate linear regression models.

In Chapter 4, we discuss the journey that followed the development of the `R` package `NNRCM` to accompany the model. We first walk the user through reproducible examples of the univariate model along with a model comparison to the NNGP model. We also present the results obtained from applying the univariate NN-RCM model to a dataset of Mediterranean sea surface temperature. Through this application, we highlight the lack of support for duplicate observations in the existing methods. By suggesting the use of larger neighborhoods for datasets involving duplicates, we demonstrate the increased accuracy of the resulting predictions. We then detail the implementation of the bivariate NN-RCM model. Again through reproducible simulations we discuss the many questions regarding multivariate nearest-neighbor spatial processes that were unanswered by current methods. We first describe the creation of multivariate neighborhoods, and then touch on the issue of misaligned sources of information. For these examples, we compare our results to predictive processes as NNGPs do not offer a multivariate implementation of their model.

In Chapter 5, we apply the two methodologies developed to the land surface albedo dataset. As an initial exploration, we investigate the 10-day albedo average which

is created along with the daily albedo assessments. We present an alternative to the current averaging methodology by applying our divide-and-conquer NNGP approach to illustrate its potential in a sequential updating setting. Subsequently, we use the bivariate NN-RCM model to extract the common surface between the two satellites and quantify the discrepancy surface in the process.

Finally, in the last chapter, we explore the next steps for our model, including our initial thoughts for a distributed implementation of the NN-RCM models. We also summarize our findings in the surface albedo case study and discuss the next phase of the analysis.

Chapter 2

Distributed Nearest-Neighbor Gaussian Processes

While data storage systems has been improved to accommodate large flows of information, model-based approaches to analyse spatial observations rely on data access and ultimately on data movement. Yet, in most cases, statistical models take for granted that the data can be accessed and manipulated all at once. It is not uncommon for large data collections to be stored in multiple locations (distributed data). Similarly, data may be stored in one location but be too large for computations to be performed at once. In addition, it is often the case that not all data are available at the same time, as in the case of sequential data retrieval processes. Under these three possible scenarios, our primary objective is to adapt a powerful and flexible statistical method to perform spatial interpolation to keep up with distributed data structures. Distributed computing and parallel implementation are key to the next significant gain in efficiency of statistical inference.

In this chapter, we introduce a divide-and-conquer approach for nearest-neighbor Gaussian processes (NNGP). The gain in efficiency from the distributed approach is of particular interest as it does not sacrifice accuracy in the process. In Section 2.1, we

review the distributed approach for low-rank models suggested by Katzfuss and Hammerling (2014). In Section 2.2, we introduce NNGPs and detail our divide-and-conquer strategy. In Section 2.3, we illustrate the method by applying it to three synthetic datasets and testing the runtime. A final demonstration of the potential for a sequential updating framework will be presented in Chapter 5 by applying our method to the Geostationary Operational Environmental Satellites (GOES) East and West satellites for three days of July 2000.

2.1 Distributed computing for spatial models

Several distributed methods for spatial models have been suggested over the last few years. The idea to define independent data subregions is explored in Heaton et al. (2017). Other approaches have looked at aggregating posterior samples into a global posterior distribution. Meta-Kriging is defined in Guhaniyogi and Banerjee (2017) and Distributed Kriging (DISK) is introduced in Guhaniyogi et al. (2017). The latter combines posterior samples from a collection of MCMC samples by averaging over the empirical quantiles of each samples. While the strength of these frameworks is that they are agnostic to the choice of model, an important drawback is the assumption that the subsets are created at random and contain locations for each region of the spatial domain. In many applications, it is often the case that data is stored by region, violating the assumption. In those situations, an initial randomization step would be required to proceed with the technique which can be costly and inefficient. A more desirable approach will tackle the dataset as presented without incurring additional data movement costs. In the next section, we review the distributed implementation of low-rank models by Katzfuss and Hammerling (2014) which uses a similar approach to our suggestion for the distributed implementation of NNGPs. As opposed to other approaches, these methodologies are not approximations. The inference obtained is

exactly the same as under the direct implementation.

2.1.1 Parallel inference for low-rank models

While low-rank models already achieve a significant improvement in computational efficiency compared to the implementation of the full GP, Katzfuss and Hammerling (2014) took a step further by using parallelization to achieve an even greater speed gain. Their divide-and-conquer approach was developed assuming that a dataset can be separated into J servers. They focus on the model

$$y(s_{j,i}) = w(s_{j,i}) + \epsilon(s_{j,i}) \tag{2.1}$$

where $s_{j,i}$ is the location of observation i on server j for $i = 1, ..., n_j$, $j = 1, ..., J$, and $\epsilon(s_{j,i}) \sim N(0, v_\epsilon(s_{j,i}))$ is independent of y for a known function v_ϵ. In spatial low-rank models, the approximation of the true underlying process is based on a set of m basis functions B:

$$w(s_{j,i}) = B(s_{j,i})^{'}\boldsymbol{\eta} + \delta(s_{j,i}),$$

where $\delta \sim N(0, v_\delta(s_{j,i}))$ is spatially independent and independent of $\boldsymbol{\eta}$. Assuming the covariance parameters are fixed, and using a normal prior, $\boldsymbol{\eta} \sim N_m(\boldsymbol{\nu}_0, K_0)$, the posterior distribution of $\boldsymbol{\eta}$ is $N_m(\boldsymbol{\nu}_y, K_y)$, where

$$\begin{aligned} K_y^{-1} &= K_0^{-1} + R, \;\; R = B_{1:J}^{'} V_{1:J}^{-1} B_{1:J} \\ \boldsymbol{\nu}_y &= K_y(K_0^{-1}\boldsymbol{\nu}_0 + \boldsymbol{\Gamma}), \;\; \boldsymbol{\Gamma} = K_{1:J}^{'} K_{1:J}^{-1} \boldsymbol{y}_{1:J}, \end{aligned}$$

where $B_{1:J} = (B_1, ..., B_J)$ is a vector of matrices where each $B_j = (B(s_{j,1}), ..., B(s_{j,n_j}))$ and $V_{1:J} = blockdiag(V_1, ..., V_J)$ where each $V_j = diag(v_\delta(s_{j,1}) + v_\epsilon(s_{j,1}), ..., v_\delta(s_{j,n_j}) + v_\epsilon(s_{j,n_j}))$. Because of the block structure of $V_{1:J}$, the above calculations become a sum of quantities that can be computed independently on each server:

$$R = \sum_{j=1}^{J} B_j^{'} V_j^{-1} B_j \;, \;\; \boldsymbol{\Gamma} = \sum_{j=1}^{J} B_j^{'} V_j^{-1} \boldsymbol{y}_j.$$

The main algorithm consists of computing the posterior parameters on each server j, moving the results to a central node and adding them to obtain the posterior parameters for the joint model. This is a special case of the algorithm developed by Qian (2018) discussed in Section 2.2.2.

2.2 Distributed NNGP

Let $w(s) \sim GP(0, C_{\boldsymbol{\theta}})$ denote a zero-centered GP where s is any location in a space $\mathcal{D}$. The process relies on a valid covariance function $C_{\boldsymbol{\theta}}$, $\boldsymbol{\theta} = (\tau^2, \phi)$, which only depends on the distance between pairs of observations. Let $\mathcal{S} = s_1, ..., s_k$ be any set of reference locations, possibly involving a grid where the indexes $1, ..., k$ are tied to a specified ordering of the locations. Note that the choice of reference set is not limited to a grid and can be expanded to include as many locations as needed. Since this is a fixed subset of $\mathcal{D}$, $\boldsymbol{w}_{\mathcal{S}} \sim N_k(0, C_{\boldsymbol{\theta},\mathcal{S}})$ where $C_{\boldsymbol{\theta},\mathcal{S}}$ is the covariance matrix corresponding to the locations in $\mathcal{S}$, generated by the covariance function $C_{\boldsymbol{\theta}}$. The joint distribution of $\boldsymbol{w}_{\mathcal{S}}$ can be expressed using a chain of conditional distribution:

$$p(\boldsymbol{w}_{\mathcal{S}}) = p(w(s_1)) \prod_{i=2}^{k} p(w(s_i)|w(s_1), ..., w(s_{i-1})).$$

The Vecchia approximation as introduced in Vecchia (1988) suggests that for a large i, the conditional distribution above includes superfluous information. It is therefore appropriate to restrict the conditional distribution to an approximation of order m based on the Euclidean distance between the locations. This likelihood approximation method implies that the locations that are closest to s_i influence the value of $w(s_i)$ the most. Applying this to the full joint distribution, we obtain

$$\tilde{p}(\boldsymbol{w}_{\mathcal{S}}) = p(w(s_1)) \prod_{i=2}^{k} p(w(s_i)|\boldsymbol{w}_{N(s_i)}),$$

where $N(s_i) \subset \{s_1, s_2, ..., s_{i-1}\}$ is a neighborhood which includes the m closest locations to s_i. This sequential structure produced by the ordering of the reference locations

creates a directed acyclic graph which guarantees a proper joint density. Using the conditional normal distribution density, we obtain the following approximated joint distribution

$$\tilde{p}(\boldsymbol{w}_{\mathcal{S}}) = \prod_{i=1}^{k} N(w(s_i)|B_{s_i}\boldsymbol{w}_{N(s_i)}, F_{s_i}),$$

where

$$\begin{aligned} B_{s_i} &= C_{\boldsymbol{\theta},s_i,N(s_i)}C^{-1}_{\boldsymbol{\theta},N(s_i)} \\ F_{s_i} &= C_{\boldsymbol{\theta},s_i} - C_{\boldsymbol{\theta},s_i,N(s_i)}C^{-1}_{\boldsymbol{\theta},N(s_i)}C_{\boldsymbol{\theta},N(s_i),s_i}, \end{aligned}$$

where $C_{\boldsymbol{\theta},s_i,N(s_i)}$ is a vector where each entry is the covariance between s_i and its neighbors $N(s_i)$, $C_{\boldsymbol{\theta},N(s_i)}$ is a symmetric matrix with elements corresponding to the covariance of each pair of neighbors, and $C_{\boldsymbol{\theta},s_i}$ is the variance of s_i. In all cases, the covariances are fully specified by the covariance function $C_{\boldsymbol{\theta}}$. The resulting distribution for $\tilde{p}(\boldsymbol{w}_{\mathcal{S}})$ is a multivariate normal distribution with covariance matrix denoted $\tilde{C}_{\boldsymbol{\theta},\mathcal{S}}$.

Let u be any location in $\mathcal{D}$, and $N(u)$ be the set of m neighbors of u in $\mathcal{S}$. As detailed in Datta et al. (2016), given a parent spatial process and a fixed reference set $\mathcal{S}$, we can construct a new process over $\mathcal{D}$. In this case, the original process is $GP(0, C_{\boldsymbol{\theta}})$, therefore for a fixed set of locations $\mathcal{U} = u_1, ..., u_n$, the nearest-neighbors density of $\boldsymbol{w}_{\mathcal{U}}$ conditional on $\boldsymbol{w}_{\mathcal{S}}$ is

$$\begin{aligned} \tilde{p}(\boldsymbol{w}_{\mathcal{U}}|\boldsymbol{w}_{\mathcal{S}}) &= \prod_{i=1}^{n} p(w(u_i)|\boldsymbol{w}_{N(u_i)}) && (2.2) \\ &= \prod_{i=1}^{n} N(w(u_i)|B_{u_i}\boldsymbol{w}_{N(u_i)}, F_{u_i}), && (2.3) \end{aligned}$$

where

$$\begin{aligned} B_{u_i} &= C_{\boldsymbol{\theta},u_i,N(u_i)}C^{-1}_{\boldsymbol{\theta},N(u_i)} \\ F_{u_i} &= C_{\boldsymbol{\theta},u_i} - C_{\boldsymbol{\theta},u_i,N(u_i)}C^{-1}_{\boldsymbol{\theta},N(u_i)}C_{\boldsymbol{\theta},N(u_i),u_i} \end{aligned}$$

From this point, we can define a new covariance function $\tilde{C}^{*}_{\boldsymbol{\theta}}$. For any two

locations u_1 and u_2 in $\mathcal{D}$, we have

$$\tilde{C}^*_{\boldsymbol{\theta}}(u_1, u_2) = \begin{cases} \tilde{C}_{\boldsymbol{\theta},s_1,s_2}, & \text{if } u_1 = s_1,\ u_2 = s_2 \\ B_{u_1}\tilde{C}_{\boldsymbol{\theta},N(u_1),s_2}, & \text{if } u_1 \notin \mathcal{S},\ u_2 = s_2 \\ B_{u_1}\tilde{C}_{\boldsymbol{\theta},N(u_1),N(u_2)}B'_{u_2} + 1_{(u_1=u_2)}F_{u_1}, & \text{if } u_1, u_2 \notin \mathcal{S} \end{cases}$$

where $\tilde{C}_{\boldsymbol{\theta}}$ is the covariance matrix associated with the density of $\tilde{\boldsymbol{w}}_{\mathcal{S}}$. This completes the construction of the new spatial process which is denoted $NNGP(0, \tilde{C}^*_{\boldsymbol{\theta}})$.

2.2.1 Current implementation

The current implementation of the NNGP model takes advantage of a fully parallel algorithm. The initial spatial model is as follows,

$$y(s_i) = \boldsymbol{x}(s_i)\boldsymbol{\beta} + w(s_i) + \epsilon(s_i),$$

where s_i is the location of observation i for $i = 1, ..., n$ and $\boldsymbol{x}(s_i)$ is a vector of dimension p. As shown in Finley et al. (2018), after integrating out the random effects in the collapsed NNGP model, we are left with a multivariate Gaussian distribution with a sparse precision matrix. The computations with this matrix can then be achieved in parallel. A similar parallelization is obtained in their NNGP-response model which does not allow for the random effects to be recovered. The authors also suggest a parallel MCMC-free implementation which chooses the covariance parameters $\boldsymbol{\theta}$ based on the minimization of the predictive mean squared error.

All three implementations of the model have computational complexity that are linear in n. The order of computations also depend on the size of the neighborhood (m^3) and the size of the parameter space (p^3). In the spatial settings, we assume that m and p are sufficiently small and therefore have a minimal impact of the computational efficiency. The issue that arises in these three algorithms however is the cost of the communication at each step. For large datasets, handling the data in one location can

be undesirable or even impossible. Moreover, when data is incoming sequentially, this implementation requires the algorithm to be performed every time from scratch, rather than update the results with every new batch of observations. In the next section, we propose a new implementation of NNGPs which solves the communication issue by reducing the communication cost to the size of the reference set.

2.2.2 Distributed implementation of posterior inference

The core computational advantage of NNGPs resides in the sparsity of the resulting precision matrix. Such sparsity can be explicitly leveraged in a dimension reduction setting, as illustrated in Banerjee (2017), where the conditional distribution of $\boldsymbol{w}_{\mathcal{U}}$ on $\boldsymbol{w}_{\mathcal{S}}$ is rewritten as a linear model:

$$w(u_i) = \sum_{j=1}^{m} a_j(u_i) w(s_j) + \eta(u_i), \;\; \eta(u_i) \sim N(0, \delta^2(u_i)), \tag{2.4}$$

where $\boldsymbol{a}_{N(u_i)}(u_i) = (a_1(u_i), ..., a_m(u_i))$ and $\delta^2(u_i)$ are fully specified by the covariance function. We can compute the values of $\boldsymbol{a}_{N(u_i)}(u_i)$ and $\delta(u_i)$ efficienctly as:

$$\begin{aligned} \boldsymbol{a}_{N(u_i)}(u_i) &= C^{-1}_{\boldsymbol{\theta},N(u_i)} C_{\boldsymbol{\theta},N(u_i),u_i}, \\ \delta^2(u_i) &= C_{\boldsymbol{\theta},u_i} - C_{\boldsymbol{\theta},u_i,N(u_i)} \boldsymbol{a}_{N(u_i)}(u_i). \end{aligned}$$

The only inverse matrices to be computed have size m. For that reason, the computational complexity is capped at $\mathcal{O}(m^3)$. The remaining operations, the matrix-vector multiplication and the dot product are of order $\mathcal{O}(m^2)$ and $\mathcal{O}(m)$ respectively. Completing the previous calculations for all datapoints is therefore linear in n, the number of observed locations.

Rewriting Equation 2.4 for the joint model of $\mathcal{U}$ and including p covariates encoded in a matrix X, we have

$$\boldsymbol{y} = \begin{bmatrix} X & A \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{w}_{\mathcal{S}} \end{bmatrix} + \boldsymbol{\xi}, \;\; \boldsymbol{\xi} \sim N_n(0, D + \tau^2 I),$$

where A is the matrix formed by the vectors $\boldsymbol{a}_{N(u_i)}(u_i)$ where $\forall s_j \notin N(u_i),\ a_{s_j}(u_i) = 0$, and D is a diagonal matrix with elements $\delta^2(u_i)$. This linear representation allows us to use the split-and-merge idea presented in Qian (2018) to compute the posterior distribution of the parameters $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2. The proposed method uses the posterior parameters of the multivariate Gaussian model computed on separate blocks of data. With a careful aggregation of the information in each block, we can recover the posterior parameters for the full dataset without loss of information. More details about the process will follow in the next section.

2.2.2.1 Posterior inference for $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2

Our approach focuses on the implementation of the NNGP in three possible scenarios: The data are too large to be handled at once, thus we seek a divide and conquer strategy where the data are split into J groups, and the processing of each group is assigned to a different processor; The data are in a distributed storage system, and it is not practical or possible to concentrate it in one device, thus, we seek to process it locally and then blend the results; The data are obtained sequentially, and we seek a compatible sequential strategy to process it. The implementation of the NNGP model in Datta et al. (2016) suggests to use $\mathcal{U}$ as the reference set. However, this is not a feasible choice in the cases considered above. Thus, the implementation of a distributed approach relies on the use of a smaller reference set of dimension k, which is assumed to be known by all servers.

Assuming the prior for $\boldsymbol{\beta}$ is $N_p(\mathbf{0}, K_\beta^{-1})$, and the prior for σ^2 is $IG(a, b)$ the joint posterior distribution for $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2 follows a normal-inverse-gamma (NIG) model, $NIG_{p+k}(\boldsymbol{\mu}^*, V^*, a^*, b^*)$ where

$$
\begin{aligned}
\boldsymbol{\mu}^* &= V^* \left(\begin{bmatrix} X & A \end{bmatrix}' G^{-1} \boldsymbol{y} \right) \\
V^* &= \left(\begin{bmatrix} X & A \end{bmatrix}' G^{-1} \begin{bmatrix} X & A \end{bmatrix} + \tilde{K} \right)^{-1} \\
a^* &= a + \frac{n}{2} \\
b^* &= b + \frac{1}{2} \left(\boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\mu}'^* V^* \boldsymbol{\mu}^* \right),
\end{aligned}
$$

where $\sigma^2 G = D + \tau^2 I$, and $\tilde{K} = blockdiag(\tilde{C}^{-1}_{\boldsymbol{\theta},S}, K_{\boldsymbol{\beta}})$. Applying the divide-and-conquer construction from Qian (2018) with the assumption that we have J blocks, the parameters of the posterior distribution are equivalently computed by

$$
\begin{aligned}
\boldsymbol{\mu}^* &= V^* \left(\sum_{j=1}^{J} \begin{bmatrix} X_j & A_j \end{bmatrix}' G_j^{-1} \boldsymbol{y}_j \right) \\
V^* &= \left(\sum_{j=1}^{J} \begin{bmatrix} X_j & A_j \end{bmatrix}' G_j^{-1} \begin{bmatrix} X_j & A_j \end{bmatrix} + \tilde{K} \right)^{-1} \\
a^* &= a + \sum_{j=1}^{J} \frac{n_j}{2} \\
b^* &= b + \frac{1}{2} \boldsymbol{\mu}^{*'} \tilde{K} \boldsymbol{\mu}^* + \frac{1}{2} \sum_{j=1}^{J} (\boldsymbol{\mu}_j - \boldsymbol{\mu}^*)' (V_j^{-1}) (\boldsymbol{\mu}_j - \boldsymbol{\mu}^*),
\end{aligned}
$$

where

$$
\begin{aligned}
\boldsymbol{\mu}_j &= V_j \begin{bmatrix} X_j & A_j \end{bmatrix}' G_j^{-1} \boldsymbol{y}_j \\
V_j &= \left(\begin{bmatrix} X_j & A_j \end{bmatrix}' G_j^{-1} \begin{bmatrix} X_j & A_j \end{bmatrix} \right)^{-1},
\end{aligned}
$$

and A_j, X_j, G_j and $\boldsymbol{y}_j$ include the locations u_i corresponding to block j.

The complexity of the posterior calculations for each block are linear in n_j. To compute $\boldsymbol{\mu}_j$, the calculations involve matrix multiplication of order $\mathcal{O}(n_j(p+k))$. To compute the matrix V_j, the calculations are of order $\mathcal{O}(n_j(p+k)^2)$. Note that these complexities do not account for the sparsity of the matrices. By using the sparse

algebra available in the `Matrix` package developed by Bates and Maechler (2018), we substantially decrease the number of computations required to obtained the previous quantities. Looking at the aggregation of the posterior parameters into the global posterior inference, the calculations required for $\boldsymbol{\mu}^*$ and V^* have complexity $\mathcal{O}((p+k)^2)$ and $\mathcal{O}((p+k)^3)$ respectively. Finally, the calculations needed to compute the posterior parameters for σ^2 are constant for a^* and of order $\mathcal{O}((p+k)^3)$ for b^*. The gain is the ability to fully distribute the computations across the servers requiring them to communicate only the values of $\boldsymbol{\mu}_j$ and V_j. Since both objects are small in size, $p+k$ and $(p+k)\times(p+k)$ respectively, the communication cost is greatly reduced.

2.2.2.2 Posterior inference of $\boldsymbol{\theta}$

The posterior inference for the covariance function parameters $\boldsymbol{\theta}$ can be obtained under two different approaches. The first one consists of estimating $\boldsymbol{\theta}$ by maximizing its marginal posterior distribution. While this method is highly efficient as it does not require to sample the random effects, it only provides a point estimate for the parameters. A more complete approach uses block sampling within an MCMC. The drawbacks of this method is that it requires constant communication between the servers and the user node. More specifically, at each iteration, the new parameters must be communicated to the user node, and the covariance matrix must be fully computed for the reference set. While both methods are feasible, we favor the former to maintain the efficiency of the proposed distributed approach. As previously mentioned, the fast implementation of the NNGP model also uses an MCMC-free version with the optimization over a grid of values for $\boldsymbol{\theta}$.

For both approaches we require the marginal likelihood. This can be computed using the outputs from each server, and is given as

$$m(\boldsymbol{y}|\boldsymbol{\theta}) = \frac{p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{w}_{\mathcal{S}}, \sigma^2, \boldsymbol{\theta}) p(\boldsymbol{\beta}, \boldsymbol{w}_{\mathcal{S}}, \sigma^2|\boldsymbol{\theta})}{p(\boldsymbol{\beta}, \boldsymbol{w}_{\mathcal{S}}, \sigma^2|\boldsymbol{y}, \boldsymbol{\theta})}. \tag{2.5}$$

This expression holds true for any value of $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2. By fixing $\boldsymbol{\beta}, \boldsymbol{w}_{\mathcal{S}} = \mathbf{0}$ and $\sigma^2 = 1$ the sampling distribution simplifies to:

$$p(\boldsymbol{y}|\boldsymbol{\beta} = \mathbf{0}, \boldsymbol{w}_{\mathcal{S}} = \mathbf{0}, \sigma^2 = 1, \boldsymbol{\theta}) = (2\pi)^{-\frac{n}{2}} \left|G\right|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{y}^T G^{-1} \boldsymbol{y}\right)$$

Finally, since G is diagonal, we can further rewrite the expressions to represent the J servers

$$\begin{aligned} \boldsymbol{y}^T G^{-1} \boldsymbol{y} &= \sum_{i=1}^{n} \frac{y(s_i)^2}{\delta(s_i) + \tau^2} = \sum_{j=1}^{J} \sum_{i=1}^{n_j} \frac{y(s_{ji})^2}{\delta(s_{ji}) + \tau^2} \\ \left|G\right|^{-1/2} &= \prod_{j=1}^{J} \left|G_j\right|^{-1/2}. \end{aligned}$$

In terms of the joint prior and posterior distributions, we need to evaluate the terms at the chosen values for $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2. The evaluation can be done at the user node level using the aggregated posterior parameters obtained from fitting the model. Therefore, the marginal likelihood in Equation 2.5 becomes

$$\begin{aligned} m(\boldsymbol{y}|\boldsymbol{\theta}) &= \frac{p(\boldsymbol{y}|\boldsymbol{\beta} = \mathbf{0}, \boldsymbol{w}_{\mathcal{S}} = \mathbf{0}, \sigma^2 = 1, \boldsymbol{\theta}) p(\boldsymbol{\beta} = \mathbf{0}, \boldsymbol{w}_{\mathcal{S}} = \mathbf{0}, \sigma^2 = 1|\boldsymbol{\theta})}{p(\boldsymbol{\beta} = \mathbf{0}, \boldsymbol{w}_{\mathcal{S}} = \mathbf{0}, \sigma^2 = 1|\boldsymbol{y}, \boldsymbol{\theta})} \\ &= \frac{(2\pi)^{-\frac{n}{2}} \left|G\right|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{y}^T G^{-1} \boldsymbol{y}\right) |\tilde{K}|^{-1/2}}{|V^*|^{-1/2} \mathrm{e}^{-\frac{1}{2}\boldsymbol{\mu}^{*T} V^{*-1} \boldsymbol{\mu}^*}} \end{aligned}$$

Again the complexity of computations for the marginal likelihood with respect to n_j is linear. The computation of the prior and posterior distributions for the specified values of $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$ and σ^2 are of order $\mathcal{O}((p+k)^3)$ due to the presence of determinants and inverse matrices. The order of computations are therefore highly dependent on the choice of the reference set. If a small reference set is adequate to obtain accurate posterior inference on the model, then this is not an issue. On the other hand, if one requires a larger reference set, then taking advantage of the sparsity of V^* will be essential.

2.2.3 Posterior predictive distribution

Using the posterior inference for $\boldsymbol{\beta}$, $\boldsymbol{w}_{\mathcal{S}}$, and $\boldsymbol{\theta}$, we can obtain the posterior predictive distribution for any new location u^*. Denote $N(u^*)$ as the set of m neighbors of u^* in the reference set $\mathcal{S}$. After obtaining B samples of the random effects $\boldsymbol{w}_{\mathcal{S}}^{(b)}$, $b = 1, ..., B$ using the posterior inference of $\boldsymbol{\mu}^*$ and V^*, we can obtain a posterior predictive sample $w^{(b)}(u^*)$ from

$$w^{(b)}(u^*) = \sum_{j=1}^{m} a_j^{(b)}(u^*) w^{(b)}(s_j)$$

where the linear coefficients are computed from the covariance function as

$$\boldsymbol{a}_{N(u^*)}^{(b)}(u^*) = C_{\boldsymbol{\theta}^*, N(u^*)}^{-1} C_{\boldsymbol{\theta}^*, N(u^*), u^*},$$

where $\boldsymbol{\theta}^*$ is the posterior point estimate of $\boldsymbol{\theta}$ obtained from maximizing the marginal likelihood.

2.3 Simulation

As a first demonstration of the potential of our algorithm, we applied our distributed approach to a range of synthetic datasets. The objective is to illustrate the potential scalability of the methodology when varying the size of the dataset and the number of blocks used to perform model inference.

The three simulated datasets varied in size from 10,000 datapoints to 1 million datapoints. The observed locations bounded by a square area were randomly generated. The observations were then sampled based on a spatial process with a Matérn covariance function with smoothness, range and variance parameters equal to $\nu = 3/2$, $\phi = 1/3$ and $\sigma^2 = 1$ respectively. In addition, an observational error was added on the observations using a Gaussian distribution with variance $\tau^2 = 0.5$.

To test the runtime of our algorithm, we repeated the maximization process using an increasing number of blocks of data from 1 block (original dataset) to 1000

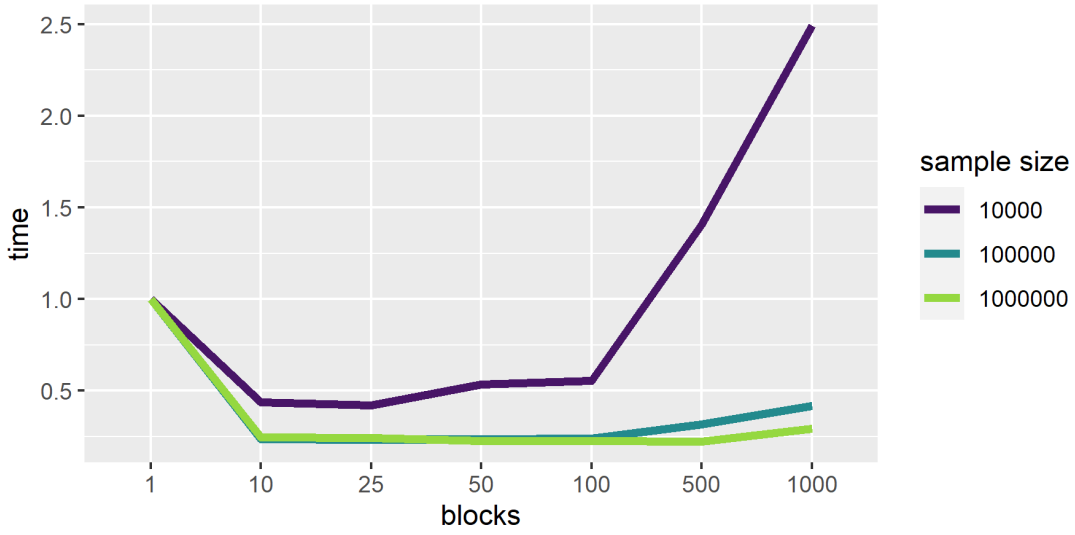


Figure 2.1: Runtime results for simulations with varying sample size and increasing number of blocks of data.

sample size	10,000	100,000	1,000,000
1 block	0.90	7.89	75.15
10 blocks	0.39	1.86	18.45
25 blocks	0.38	1.85	18.34
50 blocks	0.48	1.88	17.13
100 blocks	0.50	1.90	17.05
500 blocks	1.27	2.51	16.81
1000 blocks	2.25	3.30	22.32

Table 2.1: Runtime in minutes for each test run

blocks. For each scenario, the blocks are generated randomly. The computer used for this work was a Razer Blade 15 with Intel(R) Core(TM) i7-10750H Processor with 16.0GB of RAM. The six dual-cores give us a total of 12 cores to run the model. In Figure 2.1, we plot the runtime for each number of blocks as a percentage of the runtime with 1 block. See Table 2.1 for a summary of the runtimes in minutes. We notice that the runtime plateaus and then increases for large number of blocks. This implies that the gain in efficiency is optimal when using a moderate number of blocks. We also conclude that the gain in efficiency is more significant for larger datasets.

2.4 Conclusions and future work

As datasets continue to grow, the need to bring our statistical methods to the data rather than have the data come to us will become a priority. With many spatial applications, the cost of moving data to a central location has become a bottleneck in our ability to perform accurate statistical inference. In this chapter, we have laid out a technique to speed up the implementation of NNGP models by distributing the computations by servers. Not only can we improve the computational aspect by a factor proportional to the number of servers available to the user, we also prevent the need to manipulate the data all at once.

In that light, our method is a good alternative for sequential problems where data is collected over time. The algorithm allows the user to fit the model and update the posterior parameters as new data become available. This idea is presented in section 5.1 where we use daily albedo assessments to provide time evolving predictions over multiple days. Alternatively, instead of aggregating the posterior parameters for all blocks of data, we can gradually phase out the earlier time points with the use of a moving window. For example, in the case of the albedo assessments where we have daily updates, we could obtain spatial predictions each day by only aggregating the posterior inference from the two previous days. This would allow to see the evolution of the data without having to fit the model over the same day multiple times.

While the divide-and-conquer methodology presented focused on the univariate case, the NNGP model has been extended to the multivariate setting. Many applications rely on the joint modeling of multiple measurements and the implementation of a distributed approach would be largely beneficial.

Chapter 3

Nearest-Neighbor Gaussian Processes with Random Covariance Matrices

As previously stated, the increasing size of spatial datasets urges the development of efficient and scalable spatial models. Many methods explored rely on isotropic models, more specifically on iterations on Gaussian Processes. However, there are many cases where distance is not sufficient to determine the correlation between observations. We therefore suggest an extension to nearest-neighbor Gaussian process (NNGP) as a non-stationary and non-isotropic model.

In what follows, we develop both a univariate and a multivariate normal-inverse-Wishart (NIW) spatial model with an implementation that relies on a fully parallel algorithm. We first discuss the general framework of the model which is reminiscent of a multivariate linear regression model. Then we introduce a marginal model which does not allow for fixed effects and observational errors. The advantage of this condensed model is that inference can be obtained using optimization and still provide us with full posterior predictive uncertainty. The second approach is a more complete hierarchical model which includes fixed effects, spatial random effects and an observational error. Not only is this version more realistic but it also has the potential to

output the full posterior inference for all the parameters of interest.

3.1 Model framework

We define a nearest-neighbors Gaussian Process with random covariance matrix (NN-RCM) based on the NIW regression framework. Let $\mathcal{S} = s_1, ..., s_k$ be a set of locations for which we have spatially dependent q-variate observations $\boldsymbol{y}(\mathcal{S})$, the proposed model is:

$$\begin{aligned} \boldsymbol{y}(\mathcal{S})|\Sigma &\sim N_{kq}(0, \Sigma), \\ \Sigma &\sim IW(\alpha, (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta}}), \end{aligned} \tag{3.1}$$

where $\mathcal{C}_\theta$, when no missing values are present, is a $kq \times kq$ covariance matrix obtained from a valid cross-covariance function with parameters $\boldsymbol{\theta}$. A cross-covariance function K_θ takes two locations, s_i and s_j and returns a $q \times q$ covariance matrix where each element is the covariance between the respective components:

$$K_\theta(s_i, s_j) = \begin{pmatrix} cov(y_1(s_i), y_1(s_j)) & \cdots & cov(y_1(s_i), y_q(s_j)) \\ \vdots & \ddots & \vdots \\ cov(y_q(s_i), y_1(s_j)) & \cdots & cov(y_q(s_i), y_q(s_j)) \end{pmatrix}.$$

The covariance matrix C_θ is therefore constructed from using the covariance from the appropriate components for each location.

The model in Equation 3.1 is such that $E(\Sigma) = \mathcal{C}_\theta$. Note that as $\alpha \to \infty$, Σ converges to its mean, reverting the model back to a regular GP. This proposed model, while relying on a valid cross-covariance function apriori, offers many advantages in comparison to GPs. Primarily, the suggested approach allows for the expected covariance to be adjusted with respect to what was observed.

3.1.1 Vecchia's approximation

By introducing an ordering of the locations based on their index, we can write the joint multivariate normal model as the product of conditional normal distributions.

$$p(\boldsymbol{y}(\mathcal{S})) = \prod_{i=1}^{k} p(\boldsymbol{y}(s_i)|\boldsymbol{y}(s_1), ..., \boldsymbol{y}(s_{i-1}))$$

Given that $\boldsymbol{y}(\mathcal{S})$ is a zero-mean Gaussian process, we can express the recursive conditional representation of $\boldsymbol{y}(s_i)$ using a linear combination of $\boldsymbol{y}(s_1), ..., \boldsymbol{y}(s_{i-1})$. This in fact shows the decomposition of Σ into a lower triangular matrix L_s and a diagonal matrix Λ, both of size $kq \times kq$,

$$(I_{kq} - L_s)\boldsymbol{y}(\mathcal{S}) = e_s, \quad e_s \sim N_{kq}(0, \Lambda),$$

where I_{kq} is the identity matrix and $\Sigma = (I_{kq} - L_s)^{-1}\Lambda(I_{kq} - L_s)^{-T}$. To induce sparsity, Vecchia's approximation, which consists of reducing the size of the conditioning set to a small number m of variables, has become increasingly popular (Vecchia, 1988; Datta et al., 2016). Denote $N(s_i)$ as the m closest neighbors of s_i in $\{s_1, ..., s_{i-1}\} \in \mathcal{S}$ when $i \geq m$, and as the set $\{s_1, ..., s_{i-1}\}$ when $i < m$. The joint density can therefore be written as:

$$p(\boldsymbol{y}(\mathcal{S})) \approx \prod_{i=1}^{k} p(\boldsymbol{y}(s_i)|\boldsymbol{y}_{N(s_i)}),$$

which corresponds to setting the elements of the matrix L outside of $N(s)$ to zero.

3.1.2 Linearization of the NIW model

The distribution of $\boldsymbol{y}(s_i)$ given its m neighbors is available in closed form since the joint distribution of $\boldsymbol{y}(\mathcal{S})$ is a multivariate normal distribution. We first denote the joint covariance matrix between s_i and its m neighbors as the submatrix of Σ, $\Sigma_{\{s_i, N(s_i)\}}$:

$$\Sigma_{\{s_i, N(s_i)\}} = \begin{pmatrix} \Sigma_{s_i, N(s_i)} & \Sigma_{s_i, N(s_i)} \\ \Sigma_{N(s_i), s_i} & \Sigma_{N(s_i), N(s_i)} \end{pmatrix}.$$

Using this decomposition, we can write the conditional distribution of $y(s_i)$ as:

$$p(\boldsymbol{y}(s_i)|\boldsymbol{y}_{N(s_i)}) = N_q(\Gamma(s_i)\boldsymbol{y}_{N(s_i)}, \Phi(s_i)),$$

where

$$\Gamma(s_i) = \Sigma_{s_i,N(s_i)}\Sigma^{-1}_{N(s_i),N(s_i)},$$
$$\Phi(s_i) = \Sigma_{s_i|N(s_i)} = \Sigma_{s_i,s_i} - \Sigma_{s_i,N(s_i)}\Sigma^{-1}_{N(s_i),N(s_i)}\Sigma_{N(s_i),s_i}.$$

The form of $\Gamma(s_i)$ and $\Phi(s_i)$ come from the Bartlett decomposition which gives us the equivalence between $\Sigma_{\{s_i,N(s_i)\}}$ and

$$\Sigma_{\{s_i,N(s_i)\}} = \begin{pmatrix} \Phi(s_i) + \Gamma'(s_i)\Sigma_{N(s_i),N(s_i)}\Gamma(s_i) & \Gamma(s_i)'\Sigma_{N(s_i),N(s_i)} \\ \Sigma_{N(s_i),N(s_i)}\Gamma(s_i) & \Sigma_{N(s_i),N(s_i)} \end{pmatrix}$$

The properties and distributions of $\Gamma(s_i)$ and $\Phi(s_i)$ are then proven in a Lemma from Brown et al. (1994) and are also discussed in Dawid and Lauritzen (1993).

Lemma 3.1.1. *Suppose $\Sigma_{\{s_i,N(s_i)\}}$ is a matrix of size $(m+1)q \times (m+1)q$ and follows a Wishart distribution $IW(\delta, C)$. After the transformation to $\Sigma_{N(s_i),N(s_i)}, \Gamma(s_i)$ and $\Phi(s_i)$, the following hold:*

1. *$\Sigma_{N(s_i),N(s_i)}$ is independent of $\Gamma(s_i)$ and $\Phi(s_i)$*
2. *$\Sigma_{N(s_i),N(s_i)} \sim IW(\delta, C_{N(s_i),N(s_i)})$*
3. *$\Phi(s_i) \sim IW(\delta + m, C_{s_i|N(s_i)})$*
4. *$\Gamma(s_i)|\Phi(s_i) \sim N_{m,q}(C^{-1}_{N(s_i),N(s_i)}C_{N(s_i),s_i}, C^{-1}_{N(s_i),N(s_i)}, \Phi(s_i))$.*

Inspired by the approach from Dobra et al. (2004), we apply this result to our model in Equation 3.1 to get a model for $\boldsymbol{y}(s)$ that is compatible to the global model for $\boldsymbol{y}(\mathcal{S})$. The equivalence is summarized in Theorem 3.1.1. For clarity purposes, we first introduce the matrix Normal distribution notation in Definition 3.1.1.

Definition 3.1.1. *Let X be an $n \times p$ random matrix that follows a matrix normal distribution denoted $N_{n,p}(M, U, V)$. The three parameters defining the matrix normal probability density function include M, an $n \times p$ location matrix, U an $n \times n$ scale matrix and V, a second scale matrix with size $p \times p$. The probability density function of X has the form*

$$p(X|M,U,V) = \frac{\exp\left(-\frac{1}{2}\text{tr}(V^{-1}(X-M)'U^{-1}(X-M))\right)}{(2\pi)^{np/2}|V|^{n/2}|U|^{p/2}}.$$

Theorem 3.1.1. *Let $(\boldsymbol{y}_{\mathcal{S}}, \Sigma)$ be a kq-dimensional Normal-Inverse-Wishart hierarchical model, where $k \geq 2$ as follows;*

$$\begin{aligned} \boldsymbol{y}_{\mathcal{S}}|\Sigma &\sim N_{kq}(0, \Sigma) \\ \Sigma &\sim IWishart(\alpha, (\alpha - kq - 1)C_\theta). \end{aligned}$$

The model is such that $E(\Sigma) = C_\theta$, where C_θ is obtained from a valid cross-covariance function with parameters θ. The joint distribution of s_i and its m nearest-neighbors $N(s_i)$ can be written as

$$\boldsymbol{y}(s_i), \boldsymbol{y}_{N(s_i)}|\Sigma \sim N_{(m+1)q}(0, \Sigma_{\{s_i, N(s_i)\}}),$$

where $\Sigma_{\{s_i,N(s_i)\}}$ is an $(m+1)q \times (m+1)q$ submatrix of Σ which corresponds to the location of s_i and its m neighbors. The conditional distribution of $y(s_i)$ given its m nearest-neighbors $N(s_i)$ can be written as

$$\begin{aligned} \boldsymbol{y}(s_i)|\boldsymbol{y}_{N(s_i)}, \Sigma &\sim N_q(\Sigma_{s_i,N(s_i)}\Sigma^{-1}_{N(s_i),N(s_i)}\boldsymbol{y}_{N(s_i)}, \Sigma_{s_i,s_i} - \Sigma_{s_i,N(s_i)}\Sigma^{-1}_{N(s_i),N(s_i)}\Sigma_{N(s_i),s_i}) \\ \Sigma_{\{s_i,N(s_i)\}} &\sim IW(\alpha - kq + (m+1)q, (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},\{s_i,N(s_i)\}}. \end{aligned}$$

or equivalently as

$$\begin{aligned} \boldsymbol{y}(s_i)|\boldsymbol{y}_{N(s_i)}, \Gamma(s_i), \Phi(s_i) &= N_q(\Gamma'(s_i)\boldsymbol{y}_{N(s_i)}, \Phi(s_i)) \\ \Gamma(s_i)|\Phi(s_i) &\sim N_{mq,q}(\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s_i)}\mathcal{C}_{\boldsymbol{\theta},N(s_i),s_i}, \frac{1}{(\alpha - kq - 1)}\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s_i)}, \Phi(s_i)) \\ \Phi(s_i) &\sim IW\left(\alpha - kq + (m+1)q, (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},s_i|N(s_i)}\right), \end{aligned}$$

where $\mathcal{C}_{\boldsymbol{\theta},s_i|N(s_i)} = \mathcal{C}_{\boldsymbol{\theta},s_i} - \mathcal{C}_{\boldsymbol{\theta},s_i,N(s_i)}\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s_i)}\mathcal{C}_{\boldsymbol{\theta},N(s_i),s_i}$.

Hence, the covariance matrix likelihood has been transformed into an aggregate of normal linear regression models. This series of normal linear regressions intrinsically offers a methodology to parallelize the calculations needed for the posterior inference of the model. The strong computational efficiency potential will be discussed further in chapter 4 where we develop an `R` package to fit the presented model.

3.1.3 Extension to a stochastic process

Finally, we demonstrate that the model introduced is a valid stochastic process. Let $\mathcal{U} = u_1, ..., u_n$ be any location outside our set of observed values $\mathcal{S}$, and let $N(u_i)$ denote the neighborhood of u_i in $\mathcal{S}$. The conditional density

$$\tilde{p}(\boldsymbol{y}_{\mathcal{U}}|\boldsymbol{y}_{\mathcal{S}}, \Sigma_{\mathcal{U},\mathcal{S}}) \sim \prod_{i=1}^{n} p(\boldsymbol{y}(u_i)|\boldsymbol{y}_{N(u_i)}, \Sigma_{u_i,N(u_i)})$$

is proper because $N(u_i) \in \mathcal{S}$. Since $\tilde{p}(\boldsymbol{y}_{\mathcal{S}})$ is a multivariate Normal distribution, we have

$$\tilde{p}(\boldsymbol{y}_{\mathcal{U}}|\boldsymbol{y}_{\mathcal{S}}, \Sigma_{\mathcal{U},\mathcal{S}}) \sim \prod_{i=1}^{n} N(\Gamma(u_i)\boldsymbol{y}_{N(u_i)}, \Phi(u_i)).$$

To extend the model to a process, we generalize with the set $\mathcal{V} = v_1, ..., v_b$, which contains any location in space. Denote the non-overlapping portion with the observed values as $\mathcal{U} = \mathcal{V} \backslash \mathcal{S}$. The joint conditional density for the set $\mathcal{V}$ is

$$\tilde{p}(\boldsymbol{y}_{\mathcal{V}}|\Sigma_{\mathcal{V}}) = \int \tilde{p}(\boldsymbol{y}_{\mathcal{U}}|\boldsymbol{y}_{\mathcal{S}}, \Sigma_{\mathcal{U},\mathcal{S}})\tilde{p}(\boldsymbol{y}_{\mathcal{S}}|\Sigma_{\mathcal{S}}) \prod_{\mathcal{S}\backslash\mathcal{V}} d(\boldsymbol{y}_{\mathcal{S}}),$$

where we integrate out the locations in $\mathcal{S}$ that do not appear in $\mathcal{V}$. Since $\tilde{p}(\boldsymbol{y}_{\mathcal{U}}|\boldsymbol{y}_{\mathcal{S}}, \Sigma_{\mathcal{U},\mathcal{S}})$ and $\tilde{p}(\boldsymbol{y}_{\mathcal{S}}|\Sigma_{\mathcal{S}})$ are multivariate Normal distributions, the resulting distribution $\tilde{p}(\boldsymbol{y}_{\mathcal{V}}|\Sigma_{\mathcal{V}})$ is also a multivariate Normal. The covariance for two locations $v_1, v_2 \in \mathcal{V}$, conditional on $\Sigma_{\mathcal{V}}$, is,

$$\Sigma_{v_1,v_2} = \begin{cases} \Sigma_{\boldsymbol{s}_1,\boldsymbol{s}_2}, \text{ if } v_1 = \boldsymbol{s}_1,\ v_2 = \boldsymbol{s}_2 \\ \Gamma(v_1)\Sigma_{N(v_1),\boldsymbol{s}_2}, \text{ if } v_1 \notin \mathcal{S},\ v_2 = \boldsymbol{s}_2 \\ \Gamma(v_1)\Sigma_{N(v_1),N(v_2)}\Gamma'(v_2) + 1_{(v_1=v_2)}\Phi(v_1), \text{ if } v_1, v_2 \notin \mathcal{S}. \end{cases}$$

In summary, we have developed a multivariate nearest-neighbor non-stationary stochastic process with a tremendous potential for parallel computations. The NN-RCM model creates a dedicated non-isotropic framework that leverages the advantages of multivariate linear regression in terms of computational efficiency. As will be demonstrated in the next section, the possibility to marginalize the spatial random effects furthers even more the ability of obtaining fast posterior inference.

3.1.4 Special case: univariate observations

We briefly highlight the important case of univariate spatial observations. By setting $q = 1$ in the previous model definition, we obtain a generalization of the model in Dobra et al. (2004):

$$\begin{aligned} y(s_i) &= \boldsymbol{\gamma}'(s_i)\boldsymbol{y}_{N(s_i)} + \nu(s_i),\ \ \nu(s_i) \sim N(0, \phi(s_i)), \\ \boldsymbol{\gamma}(s_i)|\phi(s_i) &\sim N_m\left(C^{-1}_{\boldsymbol{\theta},N(s_i)}C_{\boldsymbol{\theta},N(s_i),s_i}, \frac{\phi(s_i)}{(\alpha-k-1)}C^{-1}_{\boldsymbol{\theta},N(s_i)}\right), \\ \phi(s_i) &\sim IG\left(\alpha-k+1+m, (\alpha-k-1)(C_{\boldsymbol{\theta},s_i} - C_{\boldsymbol{\theta},s_i,N(s_i)}C^{-1}_{\boldsymbol{\theta},N(s_i)}C_{\boldsymbol{\theta},N(s_i),s_i})\right). \end{aligned}$$

3.2 Marginal NN-RCM model

The first version of the model that we will consider focuses on creating a response model for predictions. By starting with the proposed model and integrating out the random effects Γ and Φ, we obtain a marginal model which is specified by two sets of parameters α and θ. The goal is to develop an efficient method to obtain the inference on α and θ and use it for prediction purposes. In the following section,

we discuss the steps to obtain the posterior marginal distribution and the suggested methodology to be employed to obtain predictions.

3.2.1 Posterior inference for α and θ

Obtaining the full posterior inference for θ, the parameters of the covariance function C_θ, requires the use of MCMC which can be costly and ineffective. We suggest instead to maximize the posterior distribution of θ to obtain the maximum aposteriori point estimates for θ. The optimization can be implemented using a fully parallel algorithm which makes the evaluation independent of the size of the dataset. For maximization purposes, we can evaluate the posterior distribution of θ upto a proportionality constant,

$$\pi(\theta|\boldsymbol{y}_{\mathcal{S}}) \propto \prod_{i=1}^{kq} m(y(s_i)|\boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta})\pi(\theta).$$

Lemma 3.2.1. *The marginal distribution of $y(s_i)$ conditional on its qm neighbors is*

$$m(y(s_i)|\boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta}) = \sqrt{\frac{1}{\pi}} \frac{\Gamma\left(\frac{\alpha-kq+mq+2}{2}\right)}{\Gamma\left(\frac{\alpha-kq+mq+1}{2}\right)} \frac{|V_{\theta,\{s_i,N(s_i)\}}|^{\frac{\alpha-kq+mq+1}{2}}}{|(V_\theta+S)_{\{s_i,N(s_i)\}}|^{\frac{\alpha-kq+mq+2}{2}}} \times \frac{|(V_\theta+S)_{N(s_i)}|^{\frac{\alpha-kq+mq+1}{2}}}{|V_{\theta,N(s_i)}|^{\frac{\alpha-kq+m}{2}}},$$

where $V_{\boldsymbol{\theta}} = (\alpha - kq - 1)C_{\boldsymbol{\theta}}$ and $S = \boldsymbol{y}\boldsymbol{y}'$.

Proof. Note that for clarity purposes, we will assume that we are handling univariate observations, that is, $q = 1$. Starting with the model in Equation 3.1, we can find the marginal distribution of $y(s_i)$ conditional on its m neighbors using the following equivalence,

$$m(y(s_i)|\boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta}) = \frac{m(y(s_i), \boldsymbol{y}_{N(s_i)}|\boldsymbol{\theta})}{m(\boldsymbol{y}_{N(s_i)}|\boldsymbol{\theta})}. \tag{3.2}$$

To find the joint marginal distribution of $y(s_i)$ and its m neighbors, we integrate out the covariance matrix Σ from the likelihood function. Recall that

$$\begin{aligned} y(s_i), \boldsymbol{y}_{N(s_i)} | \Sigma, \boldsymbol{\theta} &\sim N_{m+1}(0, \Sigma_{\{s_i, N(s_i)\}}), \\ \Sigma_{\{s_i, N(s_i)\}} | \boldsymbol{\theta} &\sim IWishart(\alpha - k + m + 1, V_{\boldsymbol{\theta}, \{s_i, N(s_i)\}}), \\ V_{\boldsymbol{\theta}, \{s_i, N(s_i)\}} &= (\alpha - k - 1) C_{\boldsymbol{\theta}, \{s_i, N(s_i)\}} \end{aligned}$$

where $C_{\theta, \{s_i, N(s_i)\}}$ is an $(m+1) \times (m+1)$ submatrix of C_θ which corresponds to the location of s_i and its m neighbors. Therefore, we find the marginal distribution of $y(s_i), \boldsymbol{y}_{N(s_i)}$ as,

$$\begin{aligned} m(y(s_i), \boldsymbol{y}_{N(s_i)} | \boldsymbol{\theta}) = \left(\frac{1}{2\pi}\right)^{\frac{m+1}{2}} & \frac{2^{-\frac{(\alpha-k+m+1)(m+1)}{2}}}{\Gamma_{m+1}\left(\frac{\alpha-k+m+1}{2}\right)} |V_{\theta, s_i}|^{\frac{\alpha-k+m+1}{2}} \\ & \times \int |\Sigma_{\{s_i, N(s_i)\}}|^{-\frac{\alpha-k+2m+4}{2}} \mathrm{e}^{\left(-\frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}_{\{s_i, N(s_i)\}} (V_\theta + S)_{\{s_i, N(s_i)\}}\right)\right)} d\Sigma_{\{s_i, N(s_i)\}}, \end{aligned}$$

where $S = \boldsymbol{y}\boldsymbol{y}'$. Simplifying the expression, we obtain

$$\begin{aligned} m(y(s_i), \boldsymbol{y}_{N(s_i)} | \boldsymbol{\theta}) &= \left(\frac{1}{2\pi}\right)^{\frac{m+1}{2}} \frac{2^{-\frac{(\alpha-k+m+1)(m+1)}{2}}}{\Gamma_{m+1}\left(\frac{\alpha-k+m+1}{2}\right)} \frac{\Gamma_{m+1}\left(\frac{\alpha-k+m+2}{2}\right)}{2^{-\frac{(\alpha-k+m+2)(m+1)}{2}}} \\ &\quad \times \frac{|V_{\theta, \{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+1}{2}}}{|(V_\theta + S)_{\{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+2}{2}}} \\ &= \left(\frac{1}{\pi}\right)^{\frac{m+1}{2}} \frac{\Gamma\left(\frac{\alpha-k+m+2}{2}\right)}{\Gamma\left(\frac{\alpha-k+1}{2}\right)} \frac{|V_{\theta, \{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+1}{2}}}{|(V_\theta + S)_{\{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+2}{2}}}. \end{aligned}$$

Similarly, the marginal distribution for the neighbors of $y(s_i)$ is found to be

$$m(\boldsymbol{y}_{N(s_i)} | \boldsymbol{\theta}) = \left(\frac{1}{\pi}\right)^{\frac{m}{2}} \frac{\Gamma\left(\frac{\alpha-k+m+1}{2}\right)}{\Gamma\left(\frac{\alpha-k+1}{2}\right)} \frac{|(V_{\theta, N(s_i)}|^{\frac{\alpha-k+m}{2}}}{|(V_\theta + S)_{N(s_i)}|^{\frac{\alpha-k+m+1}{2}}},$$

Using the two joint marginal distributions in Equation 3.2, we can find the marginal distribution of $y(s_i)$ given its m neighbors,

$$m(y(s_i) | \boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta}) = \sqrt{\frac{1}{\pi}} \frac{\Gamma\left(\frac{\alpha-k+m+2}{2}\right)}{\Gamma\left(\frac{\alpha-k+m+1}{2}\right)} \frac{|V_{\theta, \{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+1}{2}}}{|(V_\theta + S)_{\{s_i, N(s_i)\}}|^{\frac{\alpha-k+m+2}{2}}} \frac{|(V_\theta + S)_{N(s_i)}|^{\frac{\alpha-k+m+1}{2}}}{|V_{\theta, N(s_i)}|^{\frac{\alpha-k+m}{2}}}.$$

□

For the implementation of the optimization problem, we look at the summation of the log marginals. Moreover, since the terms $\Gamma\left(\frac{\alpha-kq+mq+2}{2}\right)$ and $\Gamma\left(\frac{\alpha-kq+mq+1}{2}\right)$ are very large, we can use the log-implementation of the functions or an approximation of the ratio. One possible approximation is as follows

$$\frac{\Gamma\left(\frac{\alpha-kq+mq+2}{2}\right)}{\Gamma\left(\frac{\alpha-kq+mq+1}{2}\right)} \approx \sqrt{\frac{\alpha-kq+mq+2}{2}-1}.$$

Under the assumption that the prior distribution on θ is proper, the posterior distribution is also guaranteed to be proper. In the next section, we discuss our suggestions for the choice of prior distributions on α and θ.

3.2.2 Prior distribution of α and θ

The parameters that constitutes θ depend on the choice of correlation function used for the model. In the univariate setting, the usual spatial correlation functions are categorized by families with the most common ones including the Matérn, the Gaussian and the Spherical families. In particular, the Matérn family has become the most popular choice of spatial covariance function due to its flexibility. The Matérn family depends on two parameters the range parameter ν and the smoothness parameter κ:

$$C_{\nu,\kappa}(d) = \frac{2^{\kappa-1}}{\Gamma(\kappa)}\left(\sqrt{2\kappa}\frac{d}{\nu}\right)^{\kappa} K_{\kappa}\left(\sqrt{2\kappa}\frac{d}{\nu}\right),$$

where Γ is the gamma function and K_κ is the modified Bessel function of the second kind. In addition, we also include a nugget ξ^2 to account for the initial lag at the zero distance. Finally, we complete θ with the partial sill σ^2 to obtain the final covariance matrix $C_\theta = \sigma^2(C_{\nu,\kappa} + \xi^2 I)$.

The difficulty in imposing valid priors for the range parameter ν in a GP setting is discussed in Berger et al. (2001) and Kazianka and Pilz (2012). While the focus is to use priors that guarantee to result in a proper posterior distribution, the maximization aspect of the marginal model requires additional restrictions. Without these limitations,

there is no guarantee that the maximum of the marginal will not occur with ν tending to zero or ∞. Following a similar argument as in Gu and Berger (2016), the following restriction on the prior specification of the range ν must be respected.

Lemma 3.2.2. *To guarantee that the maximized marginal distribution occurs at $\nu > 0$, the prior on ν must be such that $\pi(\nu) \to 0$ when $\nu \to 0$.*

Proof. When $\nu \to 0$, $C_\nu \to I$. Consequently, $C_\theta \to \sigma^2(1+\xi^2)I$ and $V_\theta \to (\alpha - k - 1)\sigma^2(1+\xi^2)I$. Thus when $\nu \to 0$,

$$\begin{aligned} m(y(s_i)|\boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta}) &= \sqrt{\frac{1}{\pi}} \frac{\Gamma\left(\frac{\alpha-k+m+2}{2}\right)}{\Gamma\left(\frac{\alpha-k+m+1}{2}\right)} \frac{|(\alpha-k-1)\sigma^2(1+\xi^2)I_{m+1}|^{\frac{\alpha-k+m+1}{2}}}{|((\alpha-k-1)\sigma^2(1+\xi^2)I+S)_{\{s_i,N(s_i)\}}|^{\frac{\alpha-k+m+2}{2}}} \\ &\quad \times \frac{|((\alpha-k-1)\sigma^2(1+\xi^2)I+S)_{N(s_i)}|^{\frac{\alpha-k+m+1}{2}}}{|(\alpha-k-1)\sigma^2(1+\xi^2)I_m|^{\frac{\alpha-k+m}{2}}} \\ &\to b, \end{aligned}$$

where b is a constant greater than 0. By utilizing a prior $\pi(\nu)$ such that $\pi(\nu) \to 0$ when $\nu \to 0$, we obtain that

$$m(y(s_i)|\boldsymbol{y}_{N(s_i)}, \boldsymbol{\theta})\pi(\nu) \to b\pi(\nu) \to 0$$

as $\nu \to 0$. Therefore, we ensure that the maximum of the posterior marginal will not achieved for a small value of ν. □

Adding to the limitation on the prior for the range parameter, we must also restrict the choice of prior for the partial sill σ^2 and the nugget ξ^2. To avoid excessively large values of σ^2 and ξ^2, we restrict the choice of priors to distributions that go to 0 when $\sigma^2, \xi^2 \to \infty$.

Following these guidelines, and in order to remain consistent with the hierarchical model presented in the next section, the recommended choice of prior distribution for σ^2 is the Inverse Gamma distribution. Similarly, the recommended prior on the proxy nugget ξ^2 is also an Inverse Gamma distribution. Finally, for the range parameter ν

we choose a Gamma distribution where the mean is an approximate of the smallest observed distance between two datapoints. To satisfy the rule outlined in Lemma 3.2.2, the shape a_ν must be greater than 1.

The discussion of the covariance parameter priors for the multivariate case is delayed to Chapter 4 where we detail four options that were investigated in the development of the `NNRCM` `R` package.

Finally, a natural choice for the prior distribution of the degrees of freedom α is the Pareto(x_m, p) distribution which has a truncated domain and has very thick tails. By setting the scale parameter x_m at the minimum possible value for the degrees of freedom, we ensure that the distribution is only valid on the proper interval. For the shape parameter p, a value of $p = 1$ is recommended as it results in a distribution with infinite variance.

3.2.3 Posterior predictive distribution

While the posterior marginal likelihood model only results in point estimates for our parameters, we can still recover the full posterior predictive distribution for any new location. Let s^* denote this new location with its set of m neighbors from the original data $\boldsymbol{y}(s_1), ..., \boldsymbol{y}(s_k)$. We wish to draw B samples from the posterior predictive distribution of s^* using our point estimates α^* and $\boldsymbol{\theta}^*$ obtained from the posterior marginal likelihood model. Looking back at the original model, we start by sampling $\Phi_b(s^*)$

$$\Phi_b(s^*) \sim IW\left(\alpha^* - kq + (1+m)q, (\alpha^* - kq - 1)C_{\boldsymbol{\theta}^*, s^*|N(s^*)}\right)$$

where $b = 1, ..., B$. Second, we can sample the matrix $\Gamma_b(s^*)$

$$\Gamma_b(s^*)|\Phi_b(s^*) \sim N_{mq,q}\left(C^{-1}_{\boldsymbol{\theta}^*, N(s^*)} C_{\boldsymbol{\theta}^*, N(s^*), s^*}, \frac{1}{(\alpha^* - kq - 1)} C^{-1}_{\boldsymbol{\theta}^*, N(s^*)}, \Phi_b(s^*)\right).$$

Finally, using these two set of posterior predictive samples and the neighboring observations, we can generate a set of predictions for s^*,

$$\boldsymbol{y}_b(s^*)|\Phi_b(s^*), \Gamma'(s^*) = \Gamma'(s^*)\boldsymbol{y}_{N(s^*)} + \epsilon(s^*), \;\; \epsilon(s^*) \sim N_q(0, \Phi_b(s^*)).$$

Consequently, the samples $\boldsymbol{y}_1(s^*), ..., \boldsymbol{y}_B(s^*)$ gives us the full posterior predictive distribution for $\boldsymbol{y}(s^*)$.

3.3 Hierarchical NN-RCM model with covariates

The marginal model allowed us to obtain posterior inference on the degrees of freedom and covariance parameters in a computationally efficient manner. By using an optimization scheme, we avoid costly MCMC procedures to obtain point estimates for the covariance parameters. The marginal model also enabled us to learn the full posterior predictive distribution of $y(s)$ without having to sample the random effects $w(s)$.

To elaborate on this, we designed a hierarchical model that includes fixed effects and an observational error $\epsilon(s)$. Let $y(s)$ be a q-variate observation, the model then becomes

$$\boldsymbol{y}(s) = X(s)\boldsymbol{\beta} + A\boldsymbol{w}(s) + \boldsymbol{\epsilon}(s),$$

where $X(s)$ are the covariates, $\boldsymbol{\beta}$ are the fixed effects, A is a known nonsingular matrix and $w(s)$ is the spatial field. The likelihood function and the priors are therefore as

follows:

$$
\begin{aligned}
\boldsymbol{y}(s)|\boldsymbol{\beta}, \boldsymbol{w}(s), \tau^2 &\sim N_q(X(s)\boldsymbol{\beta} + A\boldsymbol{w}(s), \boldsymbol{\tau}^2 I_q) \\
\boldsymbol{\beta} &\sim N_p(0, s_\beta^2 I) \\
\boldsymbol{w}(s)|\Gamma(s), \Phi(s) &\sim N_q(\Gamma(s)\boldsymbol{w}_{N(s)}, \Phi(s)) \\
\Gamma(s)|\Phi(s) &\sim N_{mq,q}(\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s)} C_{\boldsymbol{\theta},N(s),s}, \frac{1}{\alpha - kq - 1}\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s)}, \Phi(s)) \\
\Phi(s) &\sim IW(\alpha - kq + (1+m)q, (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},s|N(s)}), \\
\pi(\boldsymbol{\tau}^2) &\sim \prod_{i=1}^{q} IG(\tau_i^2 | a_\tau, b_\tau),
\end{aligned}
$$

where p is the dimension of the fixed effects. As previously mentioned, the structure, reminiscent of multivariate linear regression models, allows us to derive the full conditionals for each observed location s_i independently. Therefore, in the implementation of the MCMC algorithm, the spatial field $w(s_i)$ can be sampled in a fully parallel fashion. The full conditional posterior distribution for s_i, where $i = 1, ..., k$ are obtained using the Bayesian normal linear model posterior inference formulas,

$$
\boldsymbol{w}(s_i)|- \sim N_q(\boldsymbol{u}(s_i), S(s_i)),
$$

where

$$
\begin{aligned}
\boldsymbol{u}(s_i) &= S(s_i)\left(\Gamma'(s_i)\Phi^{-1}(s_i)\boldsymbol{w}_{N(s_i)} + A'\left(\boldsymbol{\tau}^2 I_q\right)^{-1}\boldsymbol{y}(s_i) + \sum_{u \in N^{-1}(s_i)} \Gamma'_{u,s_i}\Phi^{-1}(u) b_{u,s_i}\right) \\
S(s_i) &= \left(\Phi^{-1}(s_i) + A'\left(\boldsymbol{\tau}^2 I_q\right)^{-1} A + \sum_{u \in N^{-1}(s_i)} \Gamma'_{u,s_i}\Phi^{-1}(u)\Gamma_{u,s_i}\right)^{-1},
\end{aligned}
$$

where the set $u \in N^{-1}(s_i)$ is the group of locations which has s_i as a neighbor. The notation Γ_{u,s_i} indicates the rows of $\Gamma(u)$ that corresponds to the neighbors located at s_i and $b_{u,s_i} = \boldsymbol{w}(u) - \sum_{r \in N(u) \setminus s_i} \Gamma'(u,r)\boldsymbol{w}_r$.

The full conditionals for the spatial random effects $\Gamma(s_i)$ and $\Phi(s_i)$ can also be obtained for each location separately. Similar to the spatial field, we leverage the

Bayesian linear model posterior inference equations to obtained the closed-form full conditional distributions.

$$
\begin{aligned}
\Gamma(s_i)|- &\sim N_{mq,q}(U(s_i), V(s_i), \Phi(s_i)), \\
U(s_i) &= V(s_i)\left(\boldsymbol{w}_{N(s_i)}\boldsymbol{w}(s_i) + (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},N(s_i),s_i}\right) \\
V(s_i) &= \left(\boldsymbol{w}_{N(s_i)}\boldsymbol{w}'_{N(s_i)} + (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},N(s_i)}\right)^{-1} \\
\Phi(s_i)|- &\sim IW(\alpha - kq + 2q(1+m), (\alpha - kq - 1)\mathcal{C}_{\boldsymbol{\theta},s_i|N(s_i)} + Q(s_i) + R(s_i) \\
Q(s_i) &= (\boldsymbol{w}(s_i) - \Gamma'(s_i)\boldsymbol{w}_N(s_i))(\boldsymbol{w}(s_i) - \Gamma'(s_i)\boldsymbol{w}_N(s_i))' \\
R(s_i) &= (\alpha - kq - 1)(\Gamma(s_i) - \mathcal{C}^{-1}_{\boldsymbol{\theta},N(s_i)}\mathcal{C}_{\boldsymbol{\theta},N(s_i),s})'\mathcal{C}_{\boldsymbol{\theta},N(s_i)}(\Gamma(s_i) - \mathcal{C}^{-1}_{\boldsymbol{\theta},N(s_i)}\mathcal{C}_{\boldsymbol{\theta},N(s_i),s}).
\end{aligned}
$$

Finally, for each sampling iteration of the MCMC algorithm, we have global full conditionals for each element τ_q^2 of the observational error $\boldsymbol{\tau}^2$. The likelihood and the choice of prior for the observational errors in the hierarchical structure leads to conditional conjugacy. Therefore, the full conditional of τ_q^2 is also an inverse gamma distribution,

$$
\tau_q^2|- \sim IG\left(a_\tau + \frac{k}{2}, b_\tau + \sum_{s_i \in S}\frac{(w_q(s_i) - y_q(s_i))^2}{2}\right).
$$

3.3.1 Special case: univariate observations full conditionals

We again call attention to the special case of univariate spatial observations. Setting $q = 1$, the model simplifies to ,

$$
y(s) = \boldsymbol{x}(s)\boldsymbol{\beta} + w(s) + \epsilon(s),
$$

where

$$
\begin{aligned}
y(s)|\boldsymbol{\beta}, w(s), \tau^2 &\sim N(\boldsymbol{x}(s)\boldsymbol{\beta} + w(s), \tau^2) \\
\boldsymbol{\beta} &\sim N_p(0, s_\beta^2 I) \\
w(s)|\gamma(s), \phi(s), \sigma^2 &\sim N(\gamma(s)\boldsymbol{w}_{N(s)}, \sigma^2\phi(s)) \\
\gamma(s)|\phi(s) &\sim N_m(\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s)} C_{\boldsymbol{\theta},N(s),s}, \frac{\phi(s)}{\alpha - k - 1}\mathcal{C}^{-1}_{\boldsymbol{\theta},N(s)}) \\
\phi(s) &\sim IG(\alpha - k + (1+m), (\alpha - k - 1)\mathcal{C}_{\boldsymbol{\theta},s|N(s)}), \\
\pi(\sigma^2, \tau^2) &= IG(\tau^2|a_\tau, b_\tau) \times IG(\sigma^2|a_\sigma, b_\sigma),
\end{aligned}
$$

where p is the dimension of the fixed effects and m is the number of neighbors. We recognize the framework of a multivariate linear regression model in the prior structure of the spatial random effects $w(s)$. By using the posterior equations from the Normal-Normal model, we obtain a closed-form solution for the full conditional distribution of $w(s)$:

$$
w(s_i)|- \sim N(\mu(s_i), \sigma^2(s_i)),
$$

where

$$
\begin{aligned}
\mu(s_i) &= \left(\frac{1}{\sigma^2\phi(s_i)} + \frac{1}{\tau^2}\right)^{-1} \left(\frac{\boldsymbol{\gamma}'(s_i)\boldsymbol{w}_{N(s_i)}}{\sigma^2\phi(s_i)} + \frac{y(s_i)}{\tau^2} + \sum_{u \in N^{-1}(s_i)} \frac{\gamma_{u,s_i}}{\sigma^2\phi(u)} b_{u,s_i}\right) \\
\sigma^2(s_i) &= \left(\frac{1}{\sigma^2\phi(s_i)} + \frac{1}{\tau^2} + \sum_{u \in N^{-1}(s_i)} \frac{\gamma^2_{u,s_i}}{\sigma^2\phi(u)}\right)^{-1},
\end{aligned}
$$

where the notation γ_{u,s_i} indicates the element of $\gamma(u)$ that corresponds to the neighbors located at s_i and $b_{u,s_i} = w(u) - \sum_{r \in N(u) \setminus s_i} \gamma_{u,r} w(r)$.

To obtain the full conditional distributions of the random effects $\boldsymbol{\gamma}(s_i)$ and

$\phi(s_i)$, we use the normal equations from the linear models posterior inference.

$$\begin{aligned}
\boldsymbol{\gamma}(s_i)|- &\sim N_m(\boldsymbol{\mu}_{\boldsymbol{\gamma}(s_i)}, h^2_{\boldsymbol{\gamma}(s_i)}), \\
\boldsymbol{\mu}_{\boldsymbol{\gamma}(s_i)} &= \frac{h^2_{\boldsymbol{\gamma}(s_i)}}{\phi(s_i)}\left(\frac{\boldsymbol{w}_{N(s_i)}w(s_i)}{\sigma^2} + (\alpha-k-1)C_{\boldsymbol{\theta},N(s_i),s_i}\right) \\
h^2_{\boldsymbol{\gamma}(s_i)} &= \phi(s_i)\left(\frac{\boldsymbol{w}_{N(s_i)}\boldsymbol{w}'_{N(s_i)}}{\sigma^2} + (\alpha-k-1)C_{\boldsymbol{\theta},N(s_i)}\right)^{-1} \\
\phi(s_i)|- &\sim IG\left(\alpha-k+1+m+\frac{m+1}{2}, (\alpha-k-1)C_{\boldsymbol{\theta},s_i|N(s_i)} + \frac{q(s_i)}{2\sigma^2} + \frac{r(s_i)}{2}\right) \\
q(s_i) &= (w(s_i) - \boldsymbol{\gamma}'(s_i)\boldsymbol{w}_N(s_i))^2 \\
r(s_i) &= (\alpha-k-1)(\boldsymbol{\gamma}(s_i) - C^{-1}_{\boldsymbol{\theta},N(s_i)}C_{\boldsymbol{\theta},N(s_i),s})'C_{\boldsymbol{\theta},N(s_i)}(\boldsymbol{\gamma}(s_i) - C^{-1}_{\boldsymbol{\theta},N(s_i)}C_{\boldsymbol{\theta},N(s_i),s})
\end{aligned}$$

Finally, we are left with the posterior inference of the observational error τ^2 and the partial sill σ^2. In contrast to the multivariate model, the partial sill is factored out of the covariance matrix. In the previously defined model, we therefore have

$$\sigma^2 C_{\boldsymbol{\theta}} = \sigma^2(C_\nu + \xi^2 I).$$

This allows us to learn the full posterior distribution for the partial sill under the hierarchical NN-RCM model. The choice of prior for the observational error and partial sill both lead to closed-form full conditional distributions:

$$\begin{aligned}
\sigma^2|- &\sim IG\left(a_\sigma + \frac{k}{2}, b_\sigma + \sum_{s_i\in S}\frac{(w(s_i) - \boldsymbol{\gamma}'(s_i)\boldsymbol{w}_N(s_i))^2}{2\phi(s_i)}\right) \\
\tau^2|- &\sim IG\left(a_\tau + \frac{k}{2}, b_\tau + \sum_{s_i\in S}\frac{(w(s_i) - y(s_i))^2}{2}\right).
\end{aligned}$$

3.3.2 Posterior inference for α and θ

To sample α and θ, we leverage the marginal likelihood obtained for the marginal model. By following the same derivation, the marginal likelihood for $w(s)$ can be used in the Metropolis step. Therefore, by removing the random effects, the

likelihood involving α and θ is reduced to

$$m(w(s)|w_{N(s)},\theta,\alpha) \propto \frac{\Gamma\left(\frac{\alpha-k+m+2}{2}\right)}{\Gamma\left(\frac{\alpha-k+m+1}{2}\right)} \frac{|V_{\theta,\{s_i,N(s_i)\}}|^{\frac{\alpha-k+m+1}{2}}}{|(V_\theta+S)_{\{s_i,N(s_i)\}}|^{\frac{\alpha-k+m+2}{2}}} \frac{|(V_\theta+S)_{N(s_i)}|^{\frac{\alpha-k+m+1}{2}}}{|V_{\theta,N(s_i)}|^{\frac{\alpha-k+m}{2}}},$$

where $V_{\boldsymbol{\theta}} = (\alpha-k-1)C_{\boldsymbol{\theta}}$ and $S = \boldsymbol{y}\boldsymbol{y}'$.

It is worth noting that sampling α and θ results in a significant increase in computation time. A possible strategy to accelerate the inference based on the hierarchical model is to increase the intervals at which the parameters are sampled. Taken to the extreme, the parameters α and θ can be held fixed at the point estimates obtained from the marginal approach for the duration of the algorithm.

As previously explained for the marginal model, a natural prior selection for α is the Pareto distribution as it highlights the lower bound of the parameter. As for the range parameter ν, we are still faced with the limitations of the prior which should converge to 0 as ν approaches 0. For that reason, we use the Gamma distribution that is centered at the smallest distance between two observations. Note that even if one samples the two parameters, we highly suggest using the marginal model as a first step to obtain their starting values.

The last parameter of θ is the proxy nugget ξ^2. Unfortunately, after extensive testing, we have concluded that it is impossible to sample the proxy nugget along with the remaining covariance parameters and the observational error. Ultimately, the convergence of the MCMC algorithm was compromised by including too much flexibility in the variability of the model. Therefore, we suggest fixing ξ^2 to either a small value, or to use the point estimates obtained from the marginal model. This results in a stabilized MCMC chain and does not impact the uncertainty or the resulting predictions.

3.3.3 Posterior predictive distribution

Similar to the simplified model, using the posterior samples obtained from running an MCMC with the full conditionals detailed above, we can generate predictive

samples. Let s^* and the set $N(s^*)$ denote a new location and its qm neighbors. Assuming we have B posterior samples, we start by drawing $\Phi_b(s^*)$, where $b = 1, ..., B$ from

$$\Phi_b(s^*) \sim IW(\alpha_b - kq + (1+m)q, (\alpha_b - kq - 1)\mathcal{C}_{\boldsymbol{\theta}_b, s|N(s)}).$$

Next, we generate the matrix $\Gamma_b(s^*)$ from the posterior samples and the neighbors of s^*,

$$\Gamma_b(s^*)|\Phi_b(s) \sim N_{mq,q}(\mathcal{C}^{-1}_{\boldsymbol{\theta}_b, N(s^*)} C_{\boldsymbol{\theta}_b, N(s^*), s^*}, \frac{1}{\alpha_b - kq - 1}\mathcal{C}^{-1}_{\boldsymbol{\theta}_b, N(s^*)}, \Phi_b(s^*)).$$

Using the posterior predictive samples for $\Gamma(s^*)$ and $\Phi(s^*)$, we can sample the spatial random effects $w(s^*)$

$$\boldsymbol{w}_b(s^*)|\Gamma_b(s*), \Phi_b(s^*) \sim N_q(\Gamma_b(s^*)\boldsymbol{w}_{b, N(s^*)}, \Phi_b(s^*)).$$

Finally, we can use the original model to generate the predictions samples,

$$\boldsymbol{y}_b(s^*)|\boldsymbol{\beta}_b, \boldsymbol{w}(s^*), \tau_b^2 \sim N_q(X(s^*)\boldsymbol{\beta}_b + \boldsymbol{w}_b(s^*), \boldsymbol{\tau}_b^2 I_q).$$

In sum, we have recovered the full posterior predictive distribution of $y(s^*)$. These predictive samples can then be used for prediction purposes but also for risk assessment and for model validation.

3.4 Model properties illustration

The most important feature of the proposed model is its ability to capture anisotropy in a dataset. To understand how the non-stationarity of the model impacts the covariance, we compare the prior and posterior mean of Σ given some point estimates obtained for θ in the univariate case. Recall, the prior of Σ is

$$\Sigma \sim IW(\alpha, (\alpha - k - 1)C_\theta).$$

The posterior of Σ is available in closed-form as we have a Normal-Inverse-Wishart structure. By using the full conditionals, we obtain the following:

$$\begin{aligned}
\pi(\Sigma|\boldsymbol{y}) &\propto \pi(\Sigma)\pi(\boldsymbol{y}|\Sigma) \\
&= |\Sigma|^{-\frac{\alpha+k+1}{2}} \exp\left(-\frac{1}{2} tr(C_\theta \Sigma^{-1})\right) |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{y}'\Sigma^{-1}\boldsymbol{y}\right) \\
&= |\Sigma|^{-\frac{(\alpha+1)+k+1}{2}} \exp\left(-\frac{1}{2} tr(C_\theta \Sigma^{-1})\right) \exp\left(-\frac{1}{2}(tr(\boldsymbol{y}\boldsymbol{y}'\Sigma^{-1})\right) \\
&= |\Sigma|^{-\frac{(\alpha+1)+k+1}{2}} \exp\left(-\frac{1}{2} tr((C_\theta + \boldsymbol{y}\boldsymbol{y}')\Sigma^{-1})\right).
\end{aligned}$$

Therefore, we identify the kernel of the posterior of Σ as

$$\Sigma|\boldsymbol{y} \sim IW(\alpha+1, (\alpha-k-1)C_\theta + \boldsymbol{y}\boldsymbol{y}').$$

Recall that, by construct, the prior mean of Σ is C_θ. Moreover, from the posterior distribution of Σ, the posterior mean is

$$E(\Sigma|\boldsymbol{y}) = \frac{(\alpha-k-1)C_\theta + \boldsymbol{y}\boldsymbol{y}'}{\alpha-k}.$$

Therefore, while we have apriori an isotropic expected covariance, the aposteriori covariance is no longer stationary and incorporates information from the dataset.

To illustrate the benefits from the model and its features, we use a 1-D simulation which includes significant localized variability. Note that in the next chapter, we present a complete overview of the implementation of the NN-RCM model as an `R` package.

We start with a mean function which contains irregular patterns and local modes. Figure 3.1 shows the mean function which we wish to recover in red and the observed dataset which includes additional noise in black. We have $k = 1000$ observations which are evenly spaced over the interval $[0, 10]$.

Using the Matérn covariance family with smoothness 1 for the prior covariance, we optimize the marginal posterior distribution as shown in section 3.2.1 to recover

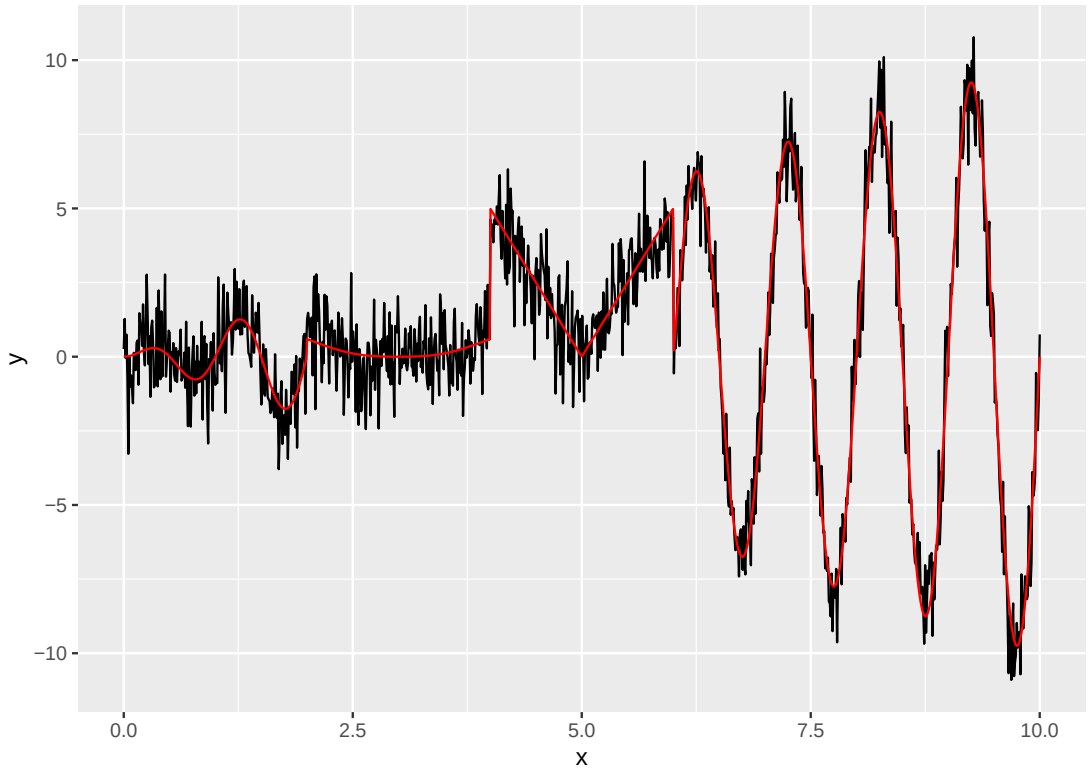


Figure 3.1: 1-D simulated dataset. The observations are in black and the underlying true surface is in red.

posterior estimates of θ. Table 3.1 summarizes the point estimates and the runtime needed to fit the model.

Using the posterior predictive inference methodology described in section 3.2.3, we can look at the posterior predictions obtained from using the marginal model. Figure 3.2 shows the mean function and the posterior predictive mean superimposed for a high resolution sequence. We see that the model is able to recover the truth through the optimization of the parameter θ.

Runtime	Degrees of Freedom	Range	Covariance	Nugget
	α	ν	σ^2	τ^2
3.13s	1100	6.14	0.14	0.16

Table 3.1: 1-D simulated dataset. Point estimates for the parameters α and θ.

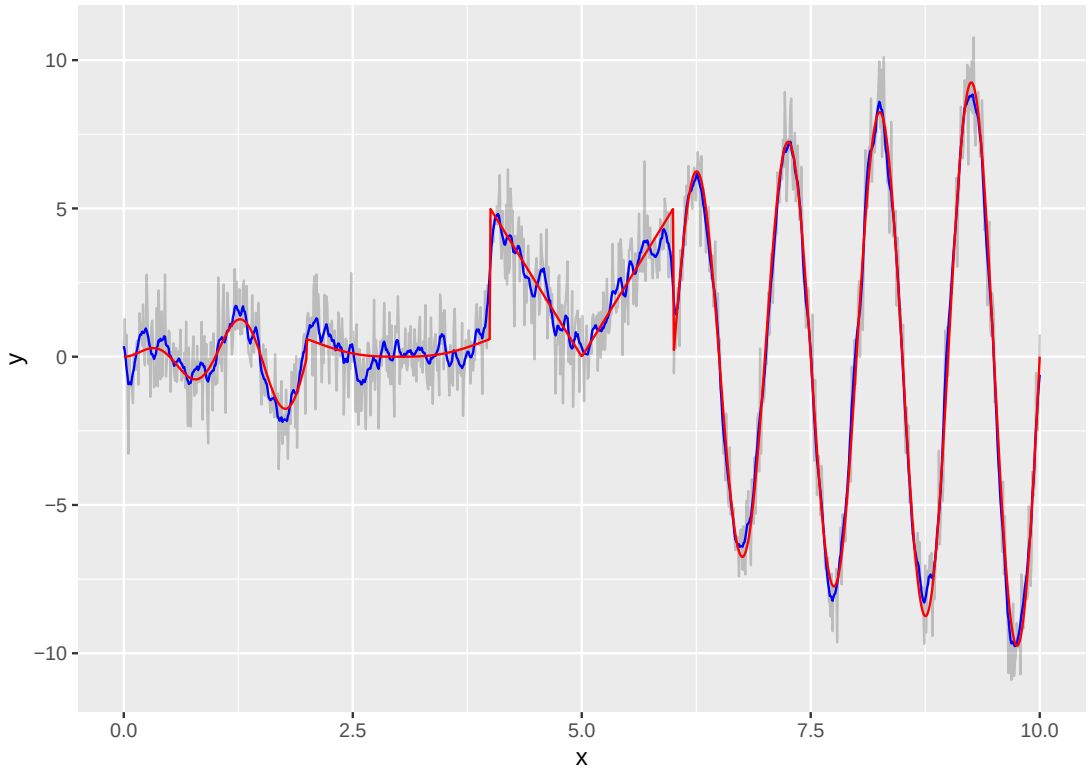
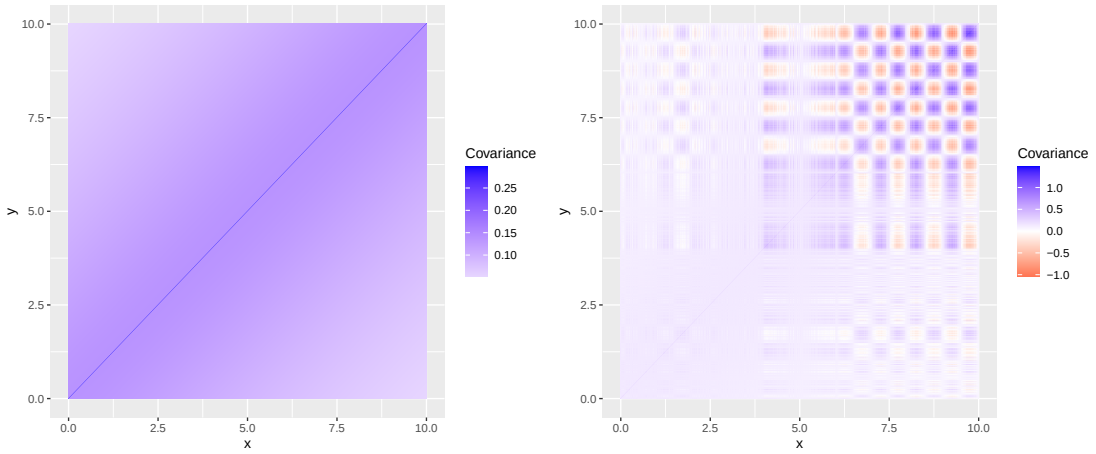


Figure 3.2: 1-D simulated dataset. The observations are in grey, the underlying true surface is in red and the predicted surface is in blue.

Finally, in Figure 3.3, we compare the prior mean and the posterior mean of Σ in our example. Since the locations are ordered from left to right and evenly spaced, the distance is constant for lines that are parallel to the diagonal. We can see that for the prior mean of Σ, shown on the left, the covariance is constant along those diagonal lines. This is a reflection of the apriori isotropy of the covariance matrix. However, for the posterior mean shown on the right, this is no longer the case. It is especially noticeable for the posterior covariance of the second half of the dataset for which the observations are generated from a sine wave. Therefore, we can conclude that the model has the particular ability to capture the local variability of the correlation structure of the dataset.



(a) Prior mean (b) Posterior mean

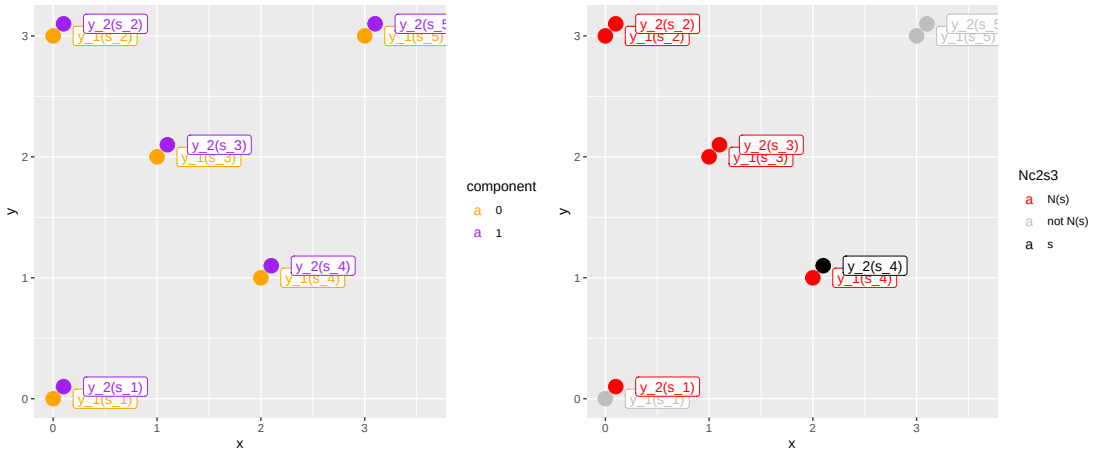
Figure 3.3: Comparison of the prior and posterior mean of the covariance parameter Σ.

3.5 Misalignment of observations

In many multivariate cases, we are faced with a misalignment of the observations. In other words, we frequently have to deal with incomplete or missing values at some locations. To date, to our knowledge, no existing nearest-neighbor models have been developed with the ability to tackle this problem. The methodology that we define in order to address this issue seeks to create neighborhoods for each source of observations separately. That is, the m neighbors from each q-variate need not come from a fixed set of m locations.

Figure 3.4 illustrates how the neighborhoods are created using a small example. Assuming that we have 5 observed locations with two components shown in orange and purple. The ordering is from left to right and the components are ordered sequentially. Therefore, to create a neighborhood of size 6 for the second component of the location s_4, which is denoted in black in the right plot of Figure 3.4, we would select the three closest previous observations from component 1 and component 2 separately. We denote these in red and together, they form the neighborhood of the observation. It is important

to highlight that the neighborhood does include the first component of the observed location.



(a) Observations (b) Neighborhood for $y_2(s_4)$

Figure 3.4: Creation of a bivariate neighborhoods using an even number of neighbors from each component.

Starting with the marginal model, we are therefore extending the initial ordering of the observations from section 3.1. Instead of ordering the observations by locations, we can extend the product to go over each component of each location.

$$p(\boldsymbol{y}(\mathcal{S})) = \prod_{i=1}^{k} \prod_{j=1}^{q_i} p(y_j(s_i)|y_1(s_1), ..., y_{j-1}(s_i))$$

where q_i indicates the number of observed variables at location s_i. Using this new ordering, we can define neighbors as we did previously based on the m closest locations. For each variate, we define a neighborhood of size m based on the previous available observations.

$$p(\boldsymbol{y}(\mathcal{S})) \approx \prod_{i=1}^{k} \prod_{j=1}^{q_i} p(y_j(s_i)|\boldsymbol{y}_{j,N(s_i)}).$$

We should note that this definition allows for a location to have itself as a neighbor, only for a previous variable.

The same methodology can be used in the hierarchical NN-RCM and the neighborhood creation for the spatial random effects $\boldsymbol{w}(s)$. Again, using the extended product over the locations and the variables, we obtain unique neighborhoods for each $w_j(s_i)$. Once these are obtained, the conditional model reverts to a product of univariate Normal-Inverse-Gamma distributions. The full conditional posterior distribution of $w_j(s_i)$ is therefore

$$w_j(s_i)|- \sim N(\mu(s_i), \sigma^2(s_i)),$$

where

$$\begin{aligned}
\mu(s_i) &= \sigma^2(s_i)\left(\frac{\boldsymbol{\gamma}'(s_i)\boldsymbol{w}_{N(s_i)}}{\phi(s_i)} + A'_{j\cdot}(\tau^2 I)^{-1}(\boldsymbol{y}(s_i) - X(s_i)\boldsymbol{\beta}) + \sum_{u\in N^{-1}(s_i)} \frac{\gamma_{j,u,s_i}}{\phi(u)} b_{u,s_i}\right) \\
\sigma^2(s_i) &= \left(\frac{1}{\sigma^2\phi(s_i)} + A'_{j\cdot}(\tau^2 I)^{-1}A_{j\cdot} + \sum_{u\in N^{-1}(s_i)} \frac{\gamma^2_{u,t,s_i}}{\phi(u)}\right)^{-1}.
\end{aligned}$$

where the notation γ_{u,t,s_i} indicates the element of vector $\gamma_(u)$ that corresponds to the t-variate of the neighbor located at s_i. The notation $b_{u,t,s_i} = w_t(u) - \sum_{r\in N(u)\setminus s_i} \gamma_{u,t^*,r} w_{t^*}(r)$ where t^* indicates all the variates that are neighbors at location r.

The impact of the missing values is seen in the posterior parameters. If only a subset of $\boldsymbol{y}(s_i)$ is available, then only the corresponding elements of A and $\boldsymbol{\tau}^2$ are maintained. Note that for some choices of A, the two terms $A'_{j\cdot}(\tau^2 I)^{-1}(\boldsymbol{y}(s_i) - X(s_i)\boldsymbol{\beta})$ and $A'_{j\cdot}(\tau^2 I)^{-1}A_{j\cdot}$ can disappear altogether. For example, if $A = I_q$, then the spatial random effects for any missing observation could not rely directly on any observations in its posterior. However, beyond the prior information, neighboring locations offer insight and therefore create a kriging effect for the locations with no information from $\boldsymbol{y}(s_i)$.

3.6 Conclusions and future work

We have demonstrated the pertinence of a non-isotropic nearest-neighbor hierarchical spatial model and discussed its implementation for both univariate and mul-

tivariate observations. In the next chapters, we present simulated examples and real life cases to illustrate how the model was implemented.

The implementation of the bivariate model generates an important discussion on the handling of missing data and neighborhoods in multivariate nearest-neighbor methods. The misalignment of multivariate observations is a key issue for many datasets that had not yet been addressed. Similarly, the question of neighborhoods in multivariate models can be answered with multiple difference options. As will be presented in the next chapter, our implementation for the bivariate model concatenates two neighborhoods each based on one source of data.

Further advancement for the NN-RCM model include a deep dive into distributed implementations. While the current development relies on parallel methods, divide-and-conquer algorithms present even stronger opportunities for computational efficiency. As featured in the previous chapter, distributed methods can not only reduce the computing time linearly, but also avoid redundancies when it comes to recurring data collection.

Chapter 4

NNRCM - R Package

The `NNRCM` package offers an implementation of the nearest-neighbors Gaussian process with random covariance matrix (NN-RCM) for both univariate and bivariate processes. The advantage of the model lies in its ability to capture anisotropy in a process. By conditioning apriori on a valid spatial covariance function, we allow added flexibility around the covariance matrix. By its definition, the model is fully parallelizable making it scalable to large datasets.

In this chapter, we demonstrate the use of the `NNRCM` package through reproducible examples and comparisons. First, we show two univariate simulations to illustrate how to use the marginal and hierarchical models. Second, we apply the univariate framework to a real dataset of Mediterranean sea surface temperatures. This dataset is a great example of a difficult and unique spatial problem. Not only does the data include a large number of duplicates, but the observed locations form a particular pattern that ultimately impacts the final predictions. Finally, we discuss the bivariate models through the use of two simulated datasets. For reproducibility purposes, all the examples have been ran on a personal computer with six core processing units, for a total of 12 threads. More specifically, the computer used for the applications was a Razer Blade 15 with Intel(R) Core(TM) i7-10750H Processor with 16.0GB of RAM.

4.1 Scoring methods

In order to compare the various univariate applications, we face our immediate competitor, the nearest-neighbor Gaussian process (NNGP) model and its implementation as the `spNNGP` package by Finley et al. (2020). Since the multivariate NNGP model has not been implemented, we compare the bivariate applications to predictive processes models available under the `spBayes` package (Finley et al., 2007, 2015). Using the posterior predictive samples obtained under each model, we are able to compare the results quantitatively. To do so, the `NNRCM` package includes a function called `NNRCM.scores` which computes the following four scoring methods:

- Predictive Mean Squared Error
- Continuous Rank Probability Score
- Posterior Predictive Loss Criterion
- 95% Predicted Interval Coverage.

While these four scoring methods can be defined for spatial vectors, it is challenging to obtain samples from the joint predictive distribution. Instead, we calculate the scores at each location s_i, $i = 1, ..., k$ of the testing subset and average the results and report the global scores.

The predictive mean squared error (PMSE) is an important tool to assess the predictive ability of a model. Not only is the PMSE a good indicator for predictive power, it also allows to keep overfitting problems under control.

Proposition 4.1.1. *Assuming that we have posterior predictive estimates $\hat{y}(s_i)$ for a set observations $y(s_i)$, where $i = 1, ..., k$, the Predictive Mean Squared Error (PMSE) is computed as*

$$PMSE = \frac{1}{k} \sum_{i=1}^{k} (y(s_i) - \hat{y}(s_i))^2$$

The Continuous Rank Probability Score (CRPS) is a second measure that assesses the forecast ability of a model (Gneiting and Raftery, 2007). Again, smaller values are desirable and indicate a better modeling of the probabilistic model.

Proposition 4.1.2. *Assuming that we have B posterior predictive samples for an observation $y(s)$, the Continuous Rank Probability Score (CRPS) is computed as*

$$\hat{crps}(F, y(s)) = \frac{1}{B}\sum_{i=1}^{B}|y^{(i)}(s) - y(s)| - \frac{1}{2B^2}\sum_{i=1}^{B}\sum_{j=1}^{B}|y^{(i)}(s) - y^{(j)}(s)|.$$

For a set of k observations $y(s_k)$, the global CRPS is computed as the average of their respective $\hat{crps}(F, y(s_k))$.

The Posterior predictive loss criterion (PPLC) is a metric that combines the measurement of predictive accuracy and uncertainty (Gelfand and Ghosh, 1998). The criterion seeks to penalize models that output high uncertainty predictions. Therefore, large prediction variances are less desirable than small prediction biases.

Proposition 4.1.3. *Assuming that we have posterior predictive samples of $y(s_i)$ denoted as $y(s_i)^{rep}$ where $i = 1, ..., k$, the Posterior Predictive Loss Criterion (PPLC) with a penalty term p is computed as*

$$D_p(k) = \sum_{i=1}^{k} var(y(s_i)^{rep}) + \frac{p}{p+1}\sum_{i=1}^{k}(y(s_i) - E(y(s_i)^{rep}))^2$$

Finally, the last scoring method included in the `NNRCM.scores` function is the 95% predicted interval coverage (Coverage). While this metric is difficult to compare across models, it is a good indicator of the uncertainty associated with the predicted values. A coverage of 100% can highlight the excessive variance of the predictions. On the other hand, a low coverage can point to a bias problem or overfitting issues.

Proposition 4.1.4. *Assuming that we have B posterior predictive samples for a set observations $y(s_i)$, where $i = 1, ..., k$, the 95% Predicted Interval Coverage (Coverage)*

is computed as

$$Coverage = \frac{1}{k}\sum_{i=1}^{k} \mathbb{1}_{\{y(s_i)\geq y^{(0.025k)}(s_i)\}} \mathbb{1}_{\{y(s_i)\leq y^{(0.975k)}(s_i)\}}$$

In the next section, we start with the univariate examples and a discussion of the package implementation. Note, we first use "set.seed(1)" so the user can easily reproduce the results.

4.2 Univariate spatial process

We first generate a dataset using a Gaussian Process with a Matérn covariance function with smoothness parameter $\kappa = 1/2$. We generate $n = 2500$ observations $y(s_i)$ in a 10×10 square using

$$Y(s) \sim GP(0, C_\theta),$$

where $\theta = (\sigma^2, \nu) = (1, 1)$ represents the partial sill and the range of the covariance function. For added difficulty, we also add white noise to the dataset according to a $N(0, 0.5^2)$ distribution. Finally, we split the generated observations equally into a training set and a testing set.

```
library(NNRCM)

# define the parameters
N1 <- 50
N2 <- 50
N <- N1 * N2
n <- N / 2
theta <- c(1, 1)
kappa <- 1/2

# define the locations
set.seed(1)
```

```
observed.locations <- expand.grid(seq(0, 10, length.out = N1),
                                  seq(0, 10, length.out = N2))
observed.distance <- fields::rdist(observed.locations)

# generate the observations
C <- exp(-observed.distance / theta[2])
y.truth <- mvrnorm(1, rep(0, N), C)

# add noise (optional)
y <- y.truth + rnorm(N, 0, 0.5)

# divide the observations into the training and testing sets
training.sample <- sample(1:N, n, replace = FALSE)
training.locations <- observed.locations[training.sample, ]
testing.locations <- observed.locations[-training.sample, ]
training.y <- y[training.sample]
testing.y <- y[-training.sample]
```

4.2.1 Marginal model

We first call the marginal posterior function to fit the marginal version of the model. We use the default settings for the number of neighbors (10) and the starting values $(n + 2, 0.5, 0.5, 0.5)$.

```
# inference from the marginal model on the training set
optim.results <-
  NNRCM.marginal.infer(Y = training.y,
                       observed.locations = training.locations,
                       smoothness = kappa)
```

The implementation of the marginal model relies on the `optim` function from the base `stats` package of the `R` software (R Core Team, 2021). Since the covariance parameters and the degrees of freedom are bounded values, the method used

is the `L-BFGS-B` option. This method uses the implementation of the limited memory Broyden–Fletcher–Goldfarb–Shanno algorithm with bounded constraints by Byrd et al. (1995). Therefore, the object returned from the marginal inference function `NNRCM.marginal.infer` is the output from the `optim` call.

The computation of the objective function relies on the `RcppArmadillo` package by Eddelbuettel and Sanderson (2014) and the `Rcpp` package by Eddelbuettel and François (2011); Eddelbuettel (2013); Eddelbuettel and Balamuta (2018). We compute the posterior marginal distribution using three consecutive parallel loops from the package `RcppParallel` (Allaire et al., 2020). The first two are `parallelFor` loops which compute the neighbor covariance and posterior covariance matrices at each observed location. These matrices are saved using `cubes` objects from the `RcppArmadillo`. These two arrays are then used in a final `parallelReduce` loop to compute the log marginal at each observed location and add up their value to obtain the global posterior marginal value.

It is also essential to give credit to the `spConjNNGP` function of the `spNNGP` package for the very fast neighbor structure retrieval process. This crucial step, which comes prior to the `optim` function was originally a burden on the runtime of our model. By using the conjugate NNGP implementation, we can obtain the neighborhood sets almost instantly to then proceed with the rest of the model.

```
optim.results
$par
[1] 2354.3895857    0.8592910    0.7475868    0.2909917

$value
[1] -1114.765

$counts
function gradient
      50       50
```

```
$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

Using the element `par` from the `optim` object, we can then call the function `NNRCM.marginal.predict`. This function allows us to obtain predictive samples for each location of interest. We again use the default settings for the number of neighbors. Computationally, the function is a shell calling a parallel routine over each new location. We first compute the neighbor covariance matrix and then sample the predictions from the posterior predictive model. Note that for all the posterior predictive sampling in the `NNRCM` package, the neighbor structure retrieval is done using the `nn2` function from the `RANN` package by Arya et al. (2019).

```
# posterior predictive sampling on the testing set
posterior.predictive.marginal <-
  NNRCM.marginal.predict(Y = training.y,
                         observed.locations = training.locations,
                         predicted.locations = testing.locations,
                         optim.pars = optim.results$par,
                         smoothness = kappa)
```

Using these posterior predictive samples, we call the function `NNRCM.scores` to compute the PMSE, CRPS, PPLC and Coverage values. In Section 4.2.3, we will revisit these values when comparing the model to its NNGP counterpart.

```
# scoring
NNRCM.scores(Y = y.truth[-training.sample],
             pp.samples = posterior.predictive.marginal,
             k = 1)
       PMSE         CRPS         PPLC     Coverage
  0.2627284    0.3004626 821.4719384    0.9992000
```

Finally, to compare the true surface and the predicted surface, we generate posterior predictive samples for every location of a fine grid spanning over the whole surface.

```
# posterior predictive sampling over high resolution surface
predicted.locations <-
  expand.grid(seq(-0.01, 10.01, length.out = 2 * N1),
              seq(-0.01, 10.01, length.out = 2 * N2))
predicted.surface <-
  NNRCM.marginal.predict(Y = training.y,
                         observed.locations = training.locations,
                         predicted.locations = predicted.locations,
                         optim.pars = optim.results$par,
                         smoothness = kappa)
```

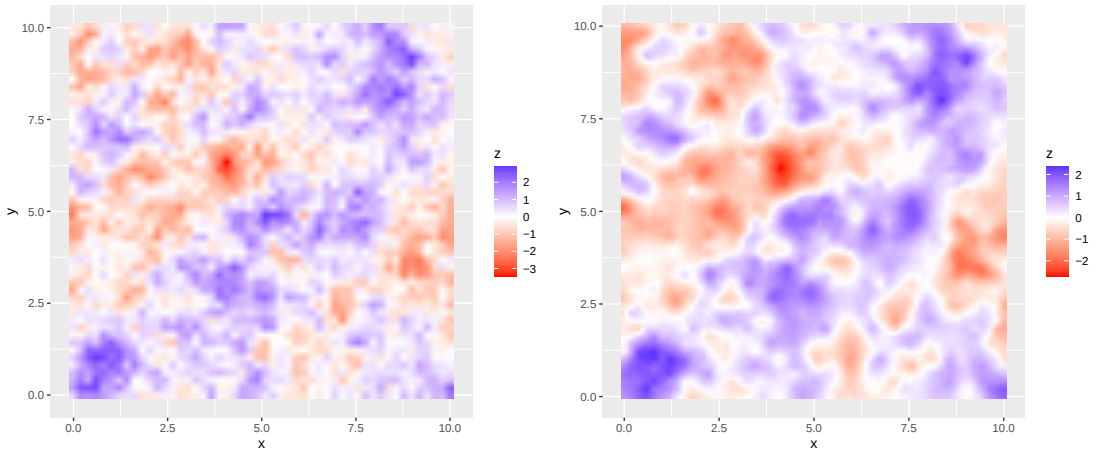
Using the `ggplot2` package, we can plot Figure 4.1 which shows the true surface (without noise) and the predicted surface side by side. The predicted surface is taken as the posterior mean computed independently at each predicted location. Running the example from the beginning to the end takes approximately 25 seconds. A more detailed breakdown of the runtime will follow in section 4.2.3 as we compare our efficiency to the `spNNGP` package.

```
# plot the true surface
true.df <- data.frame(x = observed.locations[, 1],
                      y = observed.locations[ ,2],
                      z = y.truth)

ggplot2::ggplot(true.df, aes(x = x, y = y, fill = z)) +
  geom_raster(interpolate = T) +
  scale_fill_gradient2(low = "red", mid = "white", high = "blue",
                       midpoint = 0, space = "Lab", na.value = "grey50",
                       guide = "colourbar", aesthetics = "fill")
# plot the high resolution predictions
predicted.df <- data.frame(x = predicted.locations[, 1],
```

```
                              y = predicted.locations[ ,2],
                              z = apply(predicted.surface, 1, mean))

ggplot2::ggplot(predicted.df, aes(x = x, y = y, fill = z)) +
  geom_raster(interpolate = T) +
  scale_fill_gradient2(low = "red", mid = "white", high = "blue",
                       midpoint = 0, space = "Lab", na.value = "grey50",
                       guide = "colourbar", aesthetics = "fill")
```



(a) True surface (b) Predicted surface

Figure 4.1: Comparison of true surface (no noise) and predicted surface for the univariate marginal NN-RCM model.

4.2.2 Hierarchical model

In order to model the observational error and to include fixed effects, we need to extend our analysis to the hierarchical framework. Since no covariates were used in the simulated dataset above, we only included an intercept in the training and testing design matrices.

```
# define the fixed effects
training.X <- rep(1, n)
```

```
testing.X <- rep(1, n)
```

The implementation of the hierarchical mode takes full advantage of the parallelizeable nature of the model. The Gibbs sampling step for the random effects, $w(s)$, $\phi(s)$ and $\gamma(s)$ is performed using a `parallelFor` loop over the observed locations. This significantly reduces the runtime as each location is sampled simultaneously. Next, we sample the covariance parameters and the degrees of freedom using a Metropolis step. As explained in section 3.3.2, we leverage the marginal likelihood instead of using the full conditionals of each parameters. Finally, each iteration ends with the sampling of the fixed effects based on the sampled spatial random effects $w(s)$.

As an added feature, the user has the option to choose the interval for the sampling of the degrees of freedom and the range parameter. Using an interval of 1 implies that the parameters will be samples at each MCMC iteration. At the other extreme, using an interval larger than the total number of samples will keep the parameters constant for the duration of the MCMC algorithm.

In this simulation, we do not specify the interval which ultimately keeps the parameters constant. We highly suggest this method since the marginal model provides good point estimates and this reduces the runtime considerably. As for the remaining function parameters, we again use the default settings for the number of neighbors (10) and the size of the posterior samples (1000).

```
# inference from the hierarchical model on the training set
training.samples <-
  NNRCM.hierarchical.infer(Y = training.y,
                           observed.locations = training.locations,
                           X = training.X,
                           starting.values = optim.results$par[1:3],
                           smoothness = kappa,
                           nugget = optim.results$par[4])
```

Using the posterior samples obtained, we can generate the posterior predictive samples

for the testing dataset. Similar to the marginal model, we also output the scores for the hierarchical model.

```
# posterior predictive sampling for the testing set
testing.samples <-
  NNRCM.hierarchical.predict(
    Y = training.y,
    X = testing.X,
    observed.locations = training.locations,
    predicted.locations = testing.locations,
    posterior.samples = training.samples,
    point.estimates = c(mean(training.samples$alpha),
                        1,
                        mean(training.samples$nu),
                        optim.results$par[4]),
    smoothness = kappa)

# scoring
NNRCM.scores(Y = y.truth[-training.sample],
             pp.samples = testing.samples,
             k = 1)
        PMSE          CRPS          PPLC     Coverage
   0.2749540     0.3160237 1041.2992470     1.0000000
```

Finally, we can complete the univariate simulation study by looking at the fine resolution predictions over the entire space. In Figure 4.2, we compare side by side the true surface, the predictions obtained from the marginal model and the hierarchical model. We notice that the two predicted surfaces are very similar.

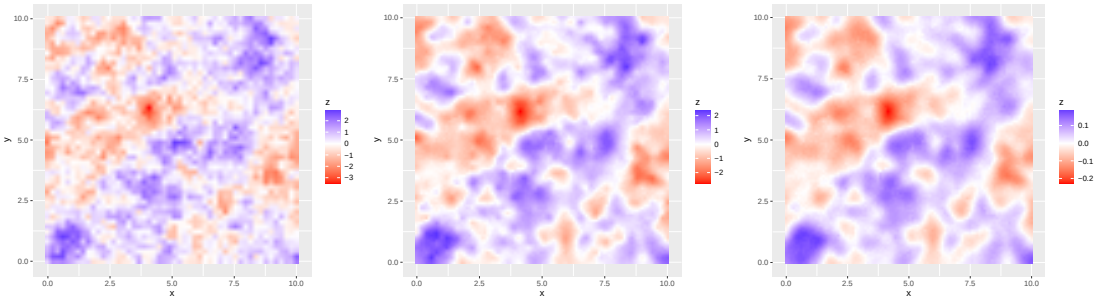
```
# posterior predictive sampling for the high resolution surface
predicted.X <- rep(1, 4 * N1 * N2)

predicted.surface.2 <-
  NNRCM.hierarchical.predict(
```

```
    Y = training.y,
    X = predicted.X,
    observed.locations = training.locations,
    predicted.locations = predicted.locations,
    posterior.samples = training.samples,
    point.estimates = c(mean(training.samples$alpha),
                        1,
                        mean(training.samples$nu),
                        optim.results$par[4]),
    smoothness = kappa)

# plot the high resolution predictions
predicted.df <- data.frame(x = predicted.locations[, 1],
                           y = predicted.locations[ ,2],
                           z = apply(predicted.surface.2, 1, mean))

ggplot2::ggplot(predicted.df, aes(x = x, y = y, fill = z)) +
  geom_raster(interpolate = T) +
  scale_fill_gradient2(low = "red", mid = "white", high = "blue",
                       midpoint = 0, space = "Lab", na.value = "grey50",
                       guide = "colourbar", aesthetics = "fill")
```



(a) True surface (b) Marginal predictions (c) Hierarchical predictions

Figure 4.2: Comparison of true surface (no noise) and predicted surfaces for the univariate NN-RCM models.

4.2.3 Comparison to `spNNGP`

To compare the results obtained using the NN-RCM model, we apply the NNGP model to the same simulations. Two versions of the NNGP models are available in the `spNNGP` package. The first function, named `spConjNNGP`, fixes the range and the nugget of the covariance structure to achieve fast posterior sampling of the fixed effects and the partial sill. In order to obtain point estimates for the fixed parameters, the suggestion is to use a variogram or to proceed with a cross-validation scheme over a grid of values. While the advantage of the maximization is that it is MCMC-free, the predictions obtained do not include any uncertainty. To obtain posterior predictive samples, we sample a Gaussian distribution centered at the predicted point estimate with the estimated variance as the scale.

The second implementation of the NNGP model is the MCMC version and is available under the function named `spNNGP`. This function has the option to output the posterior samples for all the parameters, including the random effects. To obtain the posterior predictive function, the object returned by the `spNNGP` function can be used in the `predict` function. In this case, the full posterior predictive distribution is recovered and can be used to compare the results to the NN-RCM model.

Figure 4.3 and 4.4 show the posterior predictive mean obtained under the `spNNGP` package and the `NNRCM` package. In Table 4.1 we summarize the scores obtained for the different models. From both the conjugate and the full MCMC figures, we notice heavier localized patterns of the NNGP predictions.

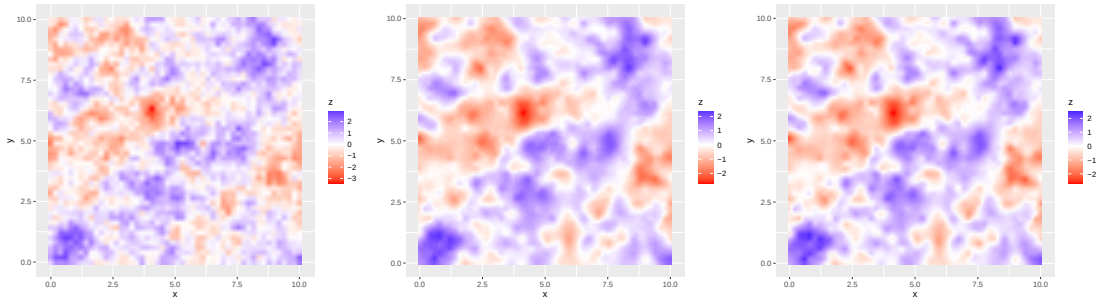
For the conjugate NNGP model, given a moderately sized grid of values (100×100) for the covariance point estimates, the runtime was 50 seconds. In comparison, the marginal NN-RCM model optimized the covariance parameters in under 5 seconds with no guidance on the starting values. It is however important to be more transparent in regards to the factors that can impact the runtime of the marginal NN-RCM function. In an attempt to reduce the global runtime of the `spConjNNGP` function, one could use

a coarse grid for the choice of range and nugget. On the other hand, without a proper first guess from the simulation, the `spConjNNGP` function can require multiple runs or make it difficult to use a small grid. Moreover, the simulation discussed involved only two parameters due to the exponential covariance function. In other cases, the number of parameters can be larger (e.g. Matérn) which makes it increasingly difficult to maintain a small grid of values. While the optimization grid supplied to the `spConjNNGP` directly impacts its runtime, the tuning of the marginal NN-RCM function is more covert. Ultimately, what dictates its runtime is the stopping rule of the optimization process. Using a very large convergence tolerance or limiting the number of iterations can decrease the runtime artificially. In our case, for the univariate marginal model, we rely on the default `optim` stopping rule which uses a convergence tolerance of $1e-08$ and a total number of iterations of 100. Finally, we note that the starting values can also have a small impact on the runtime of the marginal NN-RCM model. For the simulation, we made an honest attempt at keeping the starting values within a reasonable guess, similar to the grid of values given to the `spConjNNGP` function.

The inference of the full MCMC NNGP model took 3.45 seconds and the predictions on the testing set took 8.55 seconds. Similarly, the hierarchical NN-RCM model took 4.05 seconds for the posterior inference but sampled the predictions 4 times faster taking only 2.21 seconds. Note that the scale of the difference is maintained for larger prediction location sets. Obtaining the high resolution predictions used to plot the results took 70 seconds for the NNGP `predict` which is exactly 4 times more than the NN-RCM model at 17.5 seconds.

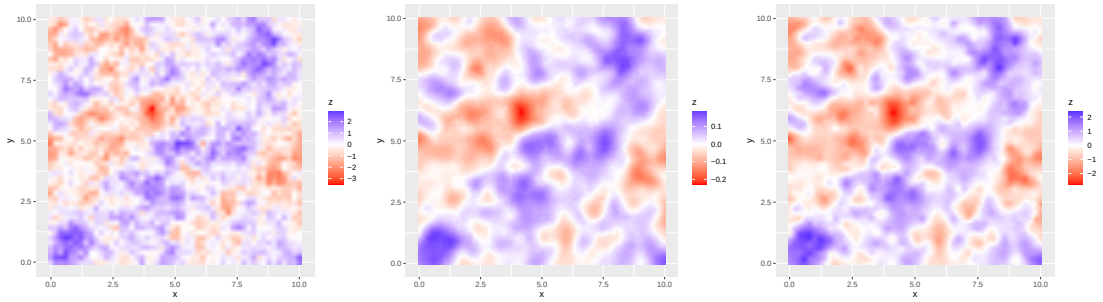
Model	PMSE	CRPS	PPLC	Coverage
marginal NN-RCM	0.2627	0.3005	821.47	0.9992
hierarchical NN-RCM	0.2749	0.3160	1041.29	1.0000
conjugate NNGP	0.2646	0.3021	841.71	0.9984
MCMC NNGP	0.2619	0.3028	872.04	0.9984

Table 4.1: Scores comparison for the univariate simulation under the four studied models



(a) True surface (b) Marginal NN-RCM (c) Conjugate NNGP

Figure 4.3: Comparison of true surface (no noise) and predicted surfaces for the marginal NN-RCM model and conjugate NNGP model.



(a) True surface (b) Hierarchical NN-RCM (c) MCMC NNGP

Figure 4.4: Comparison of true surface (no noise) and predicted surfaces for the hierarchical NN-RCM model and full MCMC NNGP model.

4.3 Mediterranean sea surface temperature

Stepping away from simulations, we explore the use of our model to analyse a Mediterranean sea surface temperature (SST) dataset from the month of December 2003. Those observations are collected from multiple devices including buoys and readings along ship routes. The dataset is available as part of the NN-RCM package using the `data` function.

```
1 data(sst)
```

Figure 4.5 shows the irregular spacing between the collected observations. We also im-

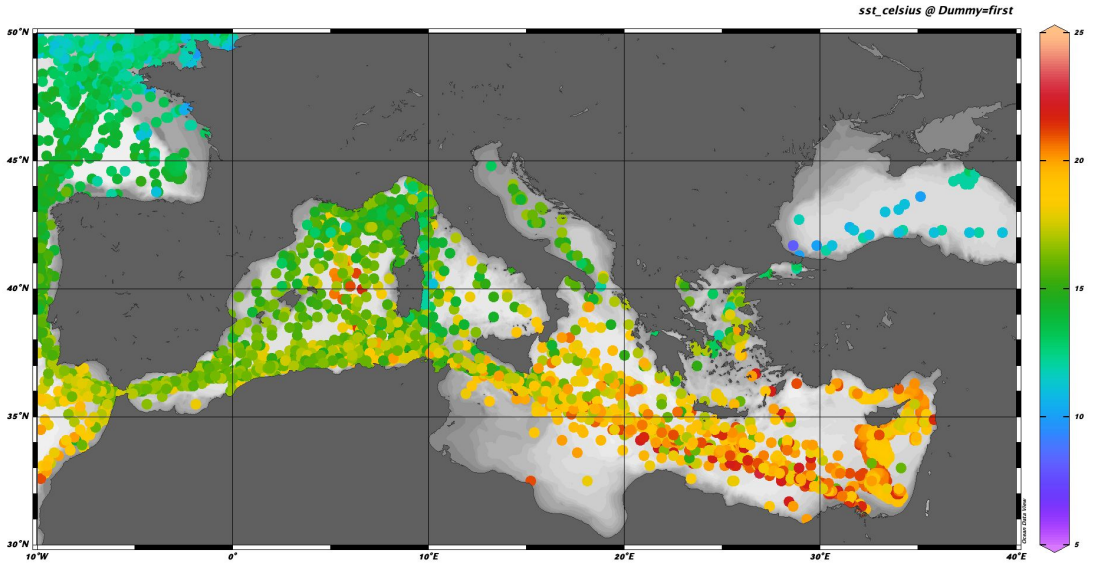


Figure 4.5: Mediterranean SST from the month of December 2003.

mediately notice that the dataset contains a large quantity of duplicates. The reason for this issue is twofold. First, some of the data were collected from free buoys over multiple days. Since the buoys tend not to move quickly through space, this results in duplicate observations. Second, the data were recorded using a very coarse resolution. The number of unique locations is 3,072 while the dataset contains over 12,000 observations. In the most extreme case, one location recorded 710 observations. We therefore have two choices to perform our analysis: average out duplicates, or adjust our model to handle duplicates. In what follows we present both with an emphasis on the novel method of handing duplicates in a nearest-neighbor spatial model.

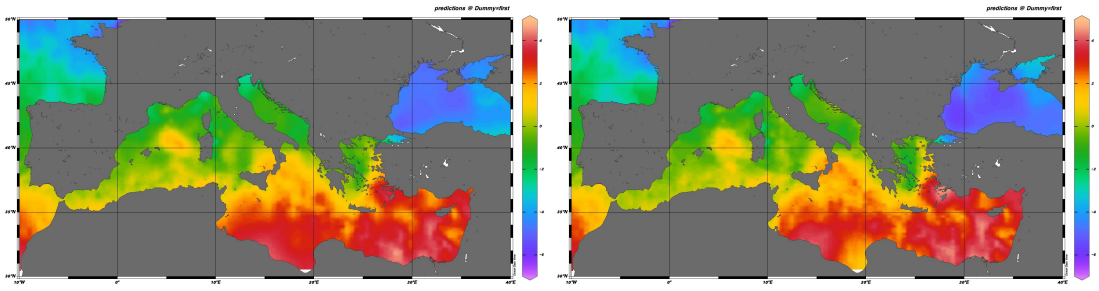
4.3.1 Averaged out duplicates

As a mean of comparison, we first output predictions using both the NN-RCM model and the `spNNGP` package. Since we did not have any external data to complement the model, the choice of fixed effects was limited to an intercept, the longitude and the latitude. The selection of the covariance function was done based on the relative

roughness of the observations. To reflect the high variation in the observations, we chose to use an exponential covariance function for both models. Finally, to compare the models, we split the dataset into two groups, the training set and the testing set. Without duplicates, the training set contained 2,572 observations and the testing subset had 500 locations. After fitting the models and obtaining posterior predictive samples for the 500 new locations, we computed the four scoring functions, PMSE, CRPS, PPLC and the Coverage. The scores in Table 4.2 show that the two models have very similar predictive samples when no duplicates are included. The NN-RCM does have a better PMSE and CRPS but has a slightly larger variability penalty in the PPLC. The similarity in the results is also visible in the predicted surfaces illustrated in Figure 4.6.

Model	PMSE	CRPS	PPLC	Coverage
Hierarchical NN-RCM	1.3497	0.6375	957.98	0.9180
Full MCMC NNGP	1.3601	0.6400	956.16	0.9280

Table 4.2: Scores comparison between the hierarchical NN-RCM model and the full MCMC NNGP model for the no duplicates SST dataset.



(a) MCMC NNGP (b) Hierarchical NN-RCM

Figure 4.6: Posterior predictive averages of the no duplicates dataset using the full MCMC NNGP model and the hierarchical NN-RCM model.

4.3.2 Allowing duplicates

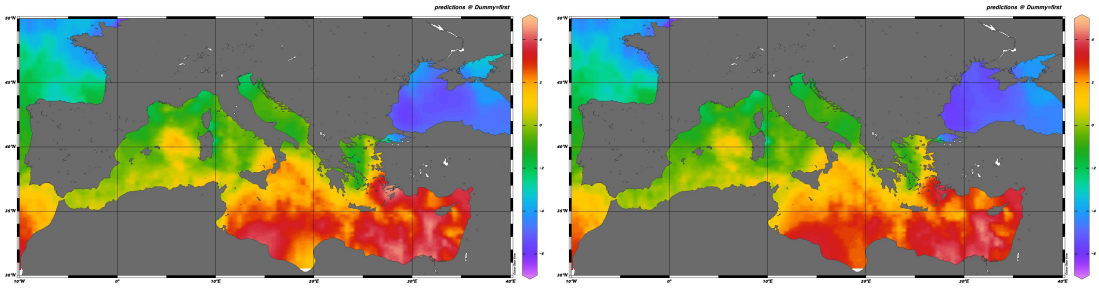
Duplicates present a serious challenge when it comes to nearest-neighbors method. In fact, the `spNNGP` package clearly specifies that duplicate locations cannot be handled by their method. In general, assuming that all locations have q duplicates, we suggest considering a neighborhood of size qm, where m is the desired number of external neighbors. Therefore handling a large number of duplicates can significantly impact the runtime of the algorithm.

In cases where only some locations have duplicates, the size of the neighborhood could be reduced but would still require to be at least as large as the highest count of duplicates. In this specific dataset, the location with the most duplicates has 710 observations. Accounting for them while allowing that location to have external neighbors defeats the purpose of using a nearest-neighbor approach. What we suggest is to reduce the number of duplicates to account for the range of the observations. For example, we reduced the duplicates to five summary statistics: the minimum, the maximum, the first and third quartile and the median.

Running the model again and comparing it against our initial results, we see a large improvement in the predictions. As seen in Table 4.3, the PMSE, CRPS and PPLC are all significantly smaller than under the no duplicates model. The coverage is also closer to 95% which is the desired level of probability we were reaching for. Figure 4.7 shows the difference in the predicted surfaces of the hierarchical NN-RCM model when including and excluding duplicates.

Model	Duplicates	PMSE	CRPS	PPLC	Coverage
hierarchical NN-RCM	No	1.3498	0.6376	957.99	0.9180
NNGP	No	1.3602	0.6400	956.16	0.9280
hierarchical NN-RCM	Yes	1.1075	0.5561	882.90	0.9700

Table 4.3: Scores comparison between the hierarchical NN-RCM and full MCMC models for the SST dataset with and without duplicates



(a) No duplicates (b) With duplicates

Figure 4.7: Posterior predictive averages using the hierarchical NN-RCM model for the SST dataset with and without duplicates.

4.4 Bivariate spatial process

The second simulation example seeks to demonstrate the ability of the package to analyze spatially dependent bivariate data. While the model has been developped for the general multivariate case, we limited the `NNRCM` package to handle bivariate data. This choice was made in light of our own interest for a particular bivariate dataset. Note that to extend the model further, additional attention to the validity of the cross-covariance function would be needed. Before we dive into the example, we must describe the four possible options for the cross-covariance functions that the user can choose to model their data.

4.4.1 Cross-covariance function options

There are many different approaches to the definition of a multivariate spatial covariance function. Genton and Kleiber (2015) give a thorough overview of the multiple options available for spatial modeling. In what follows, we review three of these methods in more detail and discuss their specific implementation in the `NNRCM` package.

The first choice is a separable model where the cross-covariance matrix is the result of the Kronecker product between a covariance matrix T and a spatial covariance

matrix generated from a common covariance function C. That is, the covariance between the variate i and j for two locations s_1 and s_2 can be written as

$$C_{ij}(s_1, s_2) = t_{ij} * C(s_1, s_2).$$

In the package, this option is available for the Matérn covariance function.

Another popular option, called the linear model of coregionalization, consists of combining univariate random fields using a linear representation. Assuming the linear combinations can be represented by the nonsingular matrix A, the cross-covariance is then defined as $A'C^*A$, where C^* is a block-matrix where each block represents the covariance matrix obtained from the respective univariate covariance function C_k. Any element of the cross-covariance matrix follows the equation below,

$$C_{ij}(s_1, s_2) = \sum_{k=1}^{q} C_k(s_1, s_2) a_{ik} a_{jk}.$$

In the package, the method of coregionalization is defined where A is a lower triangular matrix and the univariate covariance functions all follow Matérn kernels.

Finally, a multivariate extension of the Matérn covariance function has been developed. As shown in Gneiting et al. (2010), in order to use the most flexible definition of the Matérn cross-covariance function, a set of conditions must be met. The parsimonious option where the range parameter ν is kept fixed offers a largely simplified implementation of the function. In that case, the cross-covariance is calculated as

$$C_{ij}(s_1, s_2) = \rho_{ij}\sigma_i\sigma_j M(s_1, s_2|\frac{1}{2}(\kappa_i + \kappa_j), \nu),$$

where $M(\cdot|\cdot)$ is the univariate Matérn covariance function for two locations s_1 and s_2, $\rho_{11} = \rho_{22} = 1$ and κ_1 and κ_2 are the smoothness parameters. The only condition for the validity of this cross-covariance function, which is both sufficient and necessary is

$$|\rho_{12}| \leq \left(\frac{\Gamma(\kappa_1 + \frac{d}{2})}{\Gamma(\kappa_1}\right)^{1/2} \left(\frac{\Gamma(\kappa_2 + \frac{d}{2})}{\Gamma(\kappa_2}\right)^{1/2} \frac{\Gamma(\frac{1}{2}(\kappa_1 + \kappa_2)}{\Gamma(\frac{1}{2}(\kappa_1 + \kappa_2) + \frac{d}{2})},$$

where d is the dimension of the space. On the Euclidean plane, where $d = 2$, this condition reduces to

$$|\rho_{12}| \leq \frac{(\kappa_1 \kappa_2)^{1/2}}{\frac{1}{2}(\kappa_1 + \kappa_2)}.$$

A slightly more relaxed version is also proposed where the range parameters can vary as long as $\nu_1 + \nu_2 \geq 2\nu_{12}$. For our purposes, we consider the case where the cross range parameter is the average of the individual ranges, $\nu_1 + \nu_2 = 2\nu_{12}$.

4.4.2 Bivariate marginal model

The first bivariate example that we discuss consists of two surfaces generated using a cross-covariance Matérn function. For simplicity, we use the package `RandomFields` to assist us with the generation of the datasets (Schlather et al., 2021, 2015). The goal is both to demonstrate how the model was implemented but also to illustrate how the cross-covariance functions perform. Therefore, in order to challenge the different cross-covariance choices, we use range and smoothness parameters that vary greatly between the two surfaces. Moreover, we include a correlation of -0.5 between the two surfaces.

```
library(RandomFields)

# define the parameters
N1 <- 25
N2 <- 25
N <- N1 * N2
n <- 300
kappa.bv <- c(3, 0.5, 1)
range.bv <- c(2, 1, 0.5)
sill.bv <- c(1, 1, -0.5)

# define the locations
observed.locations.grid <- cbind(seq(0, 10, length.out = N1),
```

```
                                    seq(0, 10, length.out = N2))
observed.locations <- expand.grid(seq(0, 10, length.out = N1),
                                    seq(0, 10, length.out = N2))

# define the bivariate Mat{\'e}rn model using the RandomFields package
set.seed(1)
model.bv <- RMbiwm(nudiag = c(kappa.bv[1], kappa.bv[2]),
                   nured = kappa.bv[3],
                   s = range.bv,
                   cdiag = c(sill.bv[1], sill.bv[2]),
                   rhored = sill.bv[3],
                   notinvnu = TRUE)

# generate the observations
y.modeled <- RFsimulate(model.bv, x = observed.locations.grid, spConform=
    FALSE)
y.truth <- cbind(as.vector(y.modeled[,,1]), as.vector(y.modeled[,,2]))

# add noise (optional)
y <- y.truth + array(rnorm(N * 2, 0, 0.5), dim = c(N, 2))

# divide the observations into the training and testing sets
training.sample <- sample(1:N, n, replace = FALSE)
training.y.bv <- y[training.sample, ]
testing.y.bv <- y[-training.sample, ]
training.locations <- observed.locations[training.sample, ]
testing.locations <- observed.locations[-training.sample, ]
```

Similar to the univariate version, the bivariate marginal NN-RCM function uses `optim` to obtain the point estimates for the model. Once again, we rely on the `Rcpp`, `RcppArmadillo` and `RcppParallel` packages to compute the objective function of the optimization program. However, unlike the univariate function, we limit the number of iteration of the `optim` function to 20 to avoid excessive runtimes.

An added difficulty of the bivariate dataset is the creation of the neighborhoods sets. As we relied on existing functions for the univariate case, the same could not be done for the bivariate model. To account for the potential misalignment of the data, we needed to use a loop to create balanced neighborhoods for each observed location. In an attempt to speed-up the computations, we allow the user to forego the even neighborhood option. In that case, it is possible to use the `spConjNNGP` method to obtain the neighbor structure. The downside is that this no longer guarantees that each observation will benefit from having neighbors from both sources of data. For the simulation at hand, because of the small size of the dataset, we do not select this option and instead continue with the balanced neighbors choice.

The four cross-covariance function options are selected using the two flags `parsimonious` and `coregionalization`. By default, the model optimizes the parsimonious cross-covariance Matérn function. Therefore, we can obtain our first set of point estimates using the following,

```
# inference using the parsimonious Matern cross-covariance function
optim.results <-
  NNRCM.bv.marginal.infer(Y = training.y.bv,
                          observed.locations = training.locations,
                          smoothness = kappa.bv)
```

We can easily obtain the point estimates under the flexible range Matérn cross-covariance function and under the coregionalization model by using the appropriate tags. We keep the results separate using the abbreviation "ns", standing for not simplified, and "cr" standing for coregionalization.

```
# inference using the flexible Matern cross-covariance function
optim.results.ns <-
  NNRCM.bv.marginal.infer(Y = training.y.bv,
                          observed.locations = training.locations,
                          smoothness = kappa.bv,
```

```
                          parsimonious = FALSE)

# inference using coregionalization
optim.results.cr <-
  NNRCM.bv.marginal.infer(Y = training.y.bv,
                          observed.locations = training.locations,
                          smoothness = kappa.bv,
                          coregionalization = TRUE)
```

Finally, for testing purposes we also fit the separable model which assumes a constant range and smoothness parameter for the two surfaces and their cross-correlation. In order to fit this model, we use the default settings which implements the parsimonious model and provide a vector of repeated elements for the smoothness parameter. In this case, we decided to use the average of the true smoothness parameters as an acceptable compromise. We denote the results using the letters "sp" for separable.

```
# inference using a separable covariance function
optim.results.sp <-
  NNRCM.bv.marginal.infer(Y = training.y.bv,
                          observed.locations = training.locations,
                          smoothness = rep(mean(kappa.bv[1:2]), 3))
```

The next step is to output posterior predictive samples for the testing set. While we only show how to achieve this for the parsimonious Matérn cross-covariance function, the same process can be repeated for each set of point estimates. The user can modify the same flags and smoothness parameters according to what was used in the first step.

```
# posterior predictive sampling of the testing set
posterior.predictive.marginal <-
  NNRCM.bv.marginal.predict(Y = training.y.bv,
                            observed.locations = training.locations,
                            predicted.locations = testing.locations,
                            point.estimates = optim.results$par,
```

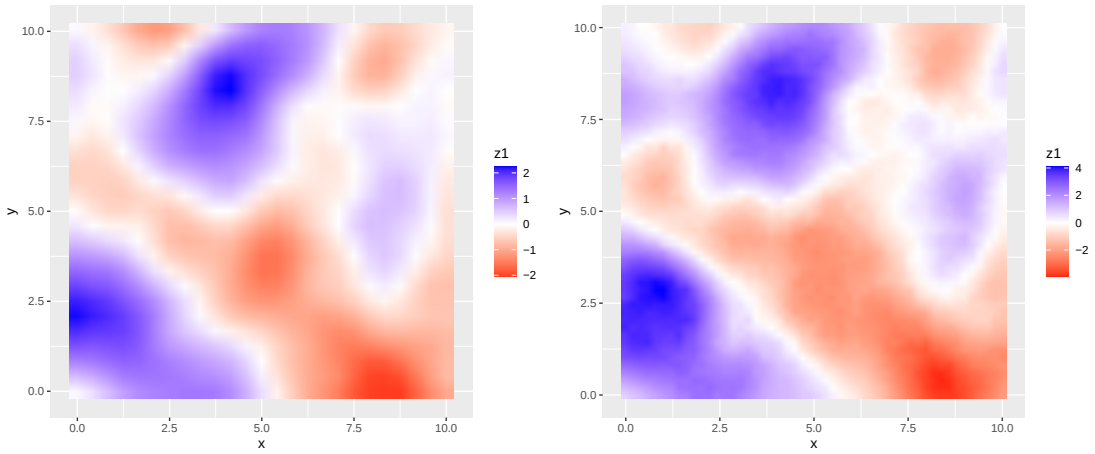
Model	Runtime	Component	PMSE	CRPS	PPLC	Coverage
Coregionalization	49.01s	Component 1	0.0503	0.1674	115.66	1.0000
		Component 2	0.6838	0.4658	418.68	0.9754
Parsimonious	6.98s	Component 1	0.0522	0.1680	116.42	1.0000
		Component 2	0.6884	0.4657	415.16	0.9754
Flexible	8.05s	Component 1	0.0507	0.1673	116.77	1.0000
		Component 2	0.6749	0.4617	417.37	0.9754
Separable	6.58s	Component 1	0.0643	0.1772	121.98	1.0000
		Component 2	0.7063	0.4706	419.81	0.9815

Table 4.4: Scores comparison between the four possible cross-covariance function choices for the bivariate marginal NN-RCM model. The scores are computed for each simulated surface individually.

```
7                                 smoothness = kappa.bv)
8
9  # scoring
10 NNRCM.scores(testing.y.bv[, 1], posterior.predictive.marginal[,,1])
11 NNRCM.scores(testing.y.bv[, 2], posterior.predictive.marginal[,,2])
```

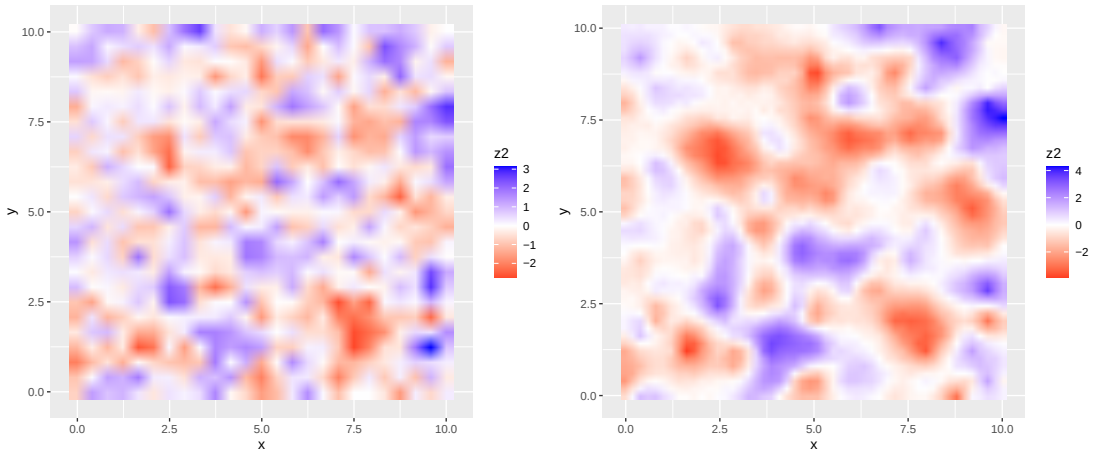
By using the same scoring function as we did under the univariate simulations, we can test the differences between the four choices of cross-covariance models. For each model, we compute the scores for each of the two surfaces of the bivariate dataset. If the global scores are desired, they can be obtained by averaging the values from each surface. From Table 4.4 we see that the separable model had a comparable runtime to the parsimonious model but did not perform as well. Additionally, even though the coregionalization model performed well, its optimization took three times longer.

To complete the example, we output high resolution predicted surfaces and compare them to the true surfaces. Figure 4.8 and 4.9 show the resulting predictions which seems to accurately depict the original datasets.



(a) True surface (b) Marginal predicted surface

Figure 4.8: Comparison of the first true surface (no noise) and predicted surface from the bivariate marginal NN-RCM model.



(a) True surface (b) Marginal predicted surface

Figure 4.9: Comparison of the second true surface (no noise) and predicted surface from the bivariate marginal NN-RCM model.

4.4.3 Bivariate hierarchical model

The final example shows the performance of the bivariate hierarchical NN-RCM model for a linear combination of the spatial random effects. Assume that we

observed $\boldsymbol{y}(s)$ which is obtained from the transformation

$$\boldsymbol{y}(s) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \boldsymbol{w}(s).$$

Our interest is to recover the two spatial surfaces $w_1(s)$ and $w_2(s)$. For this simulation, we start with the two surfaces used in the marginal example and transform them into our observed data.

```
# transform the dataset
A <- matrix(c(1,1,0,1), ncol =2)
transformed.training.y <- training.y.bv %*% t(A)
transformed.testing.y <- testing.y.bv %*% t(A)
```

Using the point estimates that we obtained from the marginal model as starting values, we fit the bivariate hierarchical model on the transformed dataset. We obtain 1000 posterior samples for the parameters in 4.01 seconds.

```
# inference from the bivariate hierarchical model on the training set
posterior.samples <-
  NNRCM.bv.hierarchical.infer(
    Y = transformed.training.y,
    observed.locations = training.locations,
    starting.values = list(alpha = optim.results$par[1],
                           sigma = optim.results$par[2:4],
                           nu = optim.results$par[5],
                           nugget = optim.results$par[7:8]),
    smoothness = kappa.bv,
    a.nu.update.interval = 1005,
    A = A)
```

In terms of the neighbor structure retrieval, the possible misalignment of the dataset does not have any impact. Since the neighbor structure is based on the spatial random effects, the missing data does not change the neighbors of each location. Therefore, we can once more leverage the `spConjNNGP` function with a slight adjust-

Model	Runtime	Component	PMSE	CRPS	PPLC	Coverage
Parsimonious	6.86s	Component 1	0.0522	0.1680	116.42	1.0000
		Component 2	0.6884	0.4657	415.16	0.9754
Hierarchical	4.00s	Component 1	0.0804	0.1915	135.62	1.0000
		Component 2	0.7089	0.4754	459.70	0.9785
`spMvLM`	2.35m	Component 1	0.0534	0.1737	124.03	1.0000
		Component 2	0.7742	0.4995	515.49	0.9815

Table 4.5: Scores comparison for the bivariate simulation under the bivariate NN-RCM models and the posterior predictive process model. The scores are computed for each surface individually.

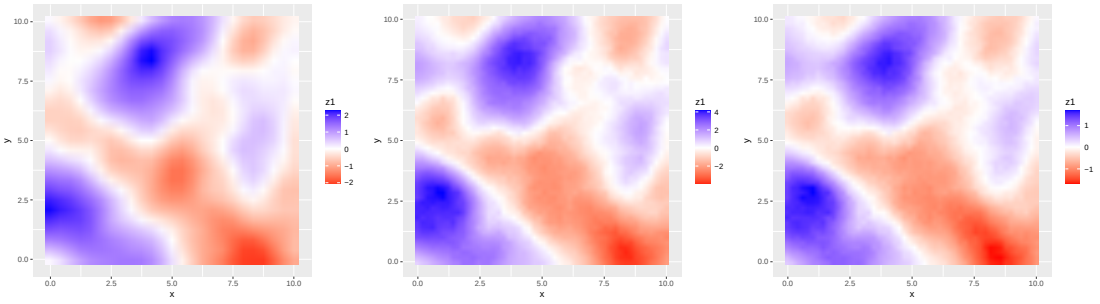
ment. While we must build different neighborhoods for each variable of each location, fortunately, the neighborhoods follow a strict logic. Let $w_1(s)$ and $w_2(s)$ be the random effects for a location s and assume that they join the global ordering of the locations in numerical order. The neighboring sets of $w_1(s)$ denoted by $N_{1,1}(s)$ and $N_{1,2}(s)$ must be the same since we have no missing data. For $w_2(s)$, the neighboring set $N_{2,2}(s)$ is the same as $N_{1,2}(s)$. The neighboring set $N_{2,1}(s)$ however must be modified: the last neighbor must be removed and $w_1(s)$ must be added. The remaining of the posterior inference of the bivariate hierarchical NN-RCM model is relatively similar to the univariate case. One notable difference is that for memory allocation reasons, we do not output the posterior samples of the random effects $\boldsymbol{\gamma}$ and . As we did for the marginal model, we use cross-validation to test the predictions obtained from the model. In Table 4.5 we compare the scores obtained under the parsimonious marginal model and the hierarchical model. The conclusion is that the hierarchical model performs very similarly in terms of prediction error and in terms of the variability of the posterior predictive samples.

```
# posterior predictive sampling for the testing set
leave.out.predictive.samples <-
  NNRCM.bv.hierarchical.predict(
    Y = transformed.training.y,
```

```
    kappa = kappa.bv,
    observed.locations = training.locations,
    predicted.locations = testing.locations,
    posterior.samples = posterior.samples,
    point.estimates = list(alpha = optim.results$par[1],
                           sigma = optim.results$par[2:4],
                           nu = optim.results$par[5],
                           nugget = optim.results$par[7:8]),
                           A = A)

# scoring
NNRCM.scores(testing.y.bv[, 1], leave.out.predictive.samples[,,1])
NNRCM.scores(testing.y.bv[, 2], leave.out.predictive.samples[,,2])
```

Finally, we output predictions for a high resolution grid and compare it to our previous results in Figure 4.10 and 4.11. We see that the model is able to recover the two transformed surfaces accurately.

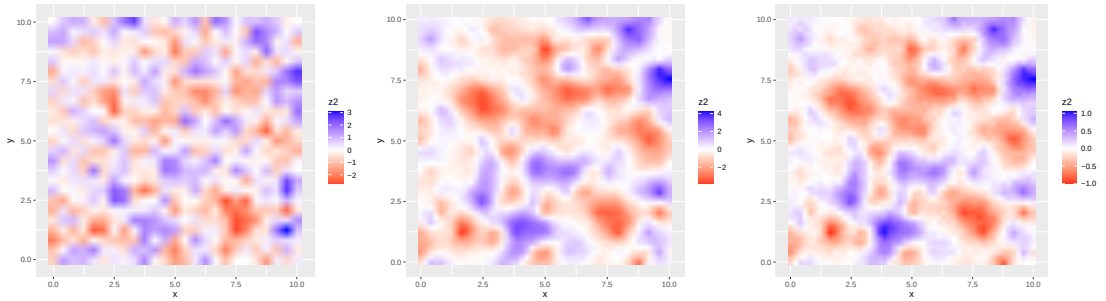


(a) True surface (b) Marginal NN-RCM (c) Hierarchical NN-RCM

Figure 4.10: Comparison of the first true surface (no noise) and predicted surfaces from the bivariate NN-RCM models.

4.4.4 Comparison to `spBayes`

Since the `spNNGP` package does not offer a multivariate implementation of the NNGP model, we resort to comparing our results to the `spBayes` package (Finley et al.,



(a) True surface (b) Marginal predictions (c) Hierarchical predictions

Figure 4.11: Comparison of the second true surface (no noise) and predicted surfaces from the bivariate NN-RCM models.

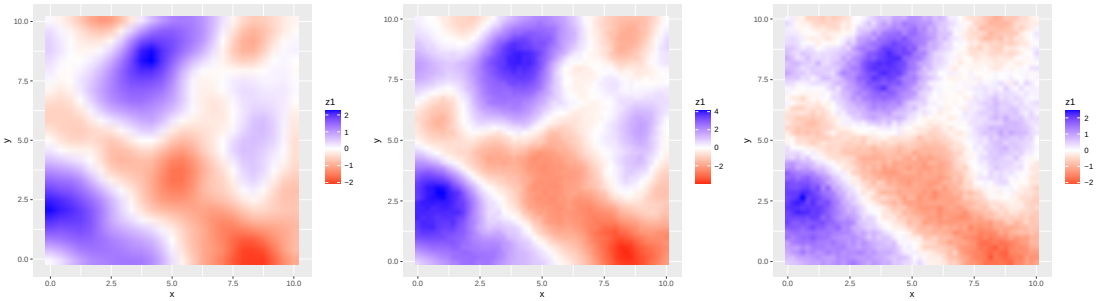
2015). Specifically, the `spMvLM` function implements the multivariate predictive process for a given set of knots over the observed surface (Banerjee et al., 2008b). Denote the set of knots by S^* and the realizations of $\boldsymbol{w}(s)$ over S^* by $\boldsymbol{w}^*$. The multivariate predictive process is then defined as

$$\tilde{\boldsymbol{w}}(s) = C_\theta(\boldsymbol{w}(s), \boldsymbol{w}^*) C_\theta(\boldsymbol{w}^*) \boldsymbol{w}^*,$$

where C_θ is any valid cross-covariance function with parameters θ. The process on $\tilde{\boldsymbol{w}}(s)$ is effectively an interpolator of the orignal GP, $w(s)$, with anchors such that $\tilde{\boldsymbol{w}}(s^*) = w(s^*)$ for all $s^* \in S^*$. Figure 4.12 and 4.13 show the true surfaces and the high resolution predictions from the marginal NN-RCM model and the predictive process side-by-side. In Table 4.5, we compare the posterior predictive scores obtained under the best bivariate marginal model to the results from the `spMvLM` function with a 10×10 grid of knots. We notice that the scores of the two models are similar for the first surface. The smoothness of the first surface presents an easy task for the predictive process which can often suffer from oversmoothing. For the second surface, the limitations of the knots is shown as the predictive process struggles to fully recover the true surface. The predictions are overly smooth which leads to a larger predictive mean squared error than the NN-RCM model. It is important to note that the predictive process required

heavy tuning and burn-in to obtain the presented output.

We must also highlight that the runtime to obtain the posterior inference was significantly longer under the `spBayes` package. To obtain the posterior inference, the `spMvLM` function took 2.35 minutes and the bivariate NN-RCM model took 6.86 seconds. The runtime to recover the predictions over the testing set was almost 300 times longer under the `spPredict` function of the `spBayes` package. In seconds, the runtime for the `NNRCM` package was 1.18 seconds and 320.50 seconds for `spBayes`. Extending this for the high resolution predictions, the runtime for the `spPredict` function was just over 1 hour compared to 7.94 seconds for the NN-RCM model.

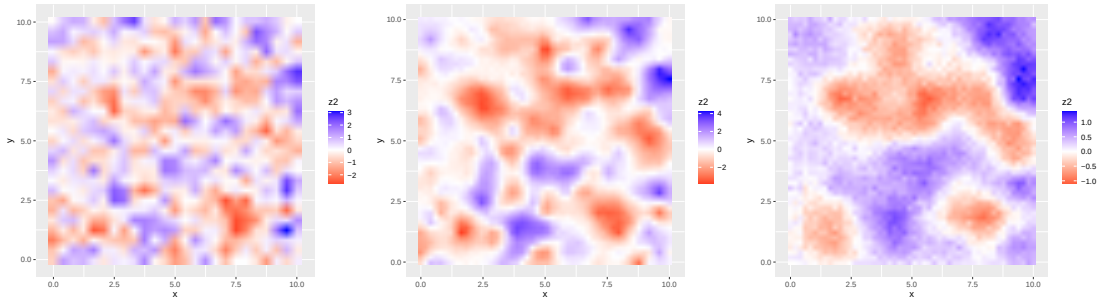


(a) True surface (b) Marginal NN-RCM (c) Predictive process

Figure 4.12: Comparison of the first true surface (no noise) and predicted surfaces from the bivariate marginal NN-RCM model and the posterior predictive process model.

4.5 Conclusions and future work

We have developed a flexible and efficient package for the univariate and bivariate NN-RCM models. The univariate model not only rivals with the `spNNGP` package in terms of accuracy and efficiency, it also provides the user with the unique ability to fit a spatial model for duplicated observations. This unique feature is especially useful in settings like the SST dataset where a large number of observations are available for each location. Reducing the data by taking the average at each location greatly reduces



(a) True surface (b) Marginal NN-RCM (c) Predictive Process

Figure 4.13: Comparison of the second true surface (no noise) and predicted surfaces from the bivariate marginal NN-RCM model and the posterior predictive process model.

the potential to understand the uncertainty of the data and the predictions.

In the multivariate settings, a spatial nearest-neighbor model had not yet been implemented in `R`. Our comparison with the established `spBayes` package showed the superior potential of the NN-RCM method for rough surfaces. More importantly, the implementation of the NN-RCM models allows for significantly faster inference even for small datasets. The difference in runtime for recovering predictive samples is also worth noting as the NN-RCM model was 300 times faster.

The development of the `NN-RCM` package is still in progress with a few key items being of particular importance. First, while the Matérn covariance function is largely favored in spatial settings, we wish to expand the covariance function options of the package. At a minimum, the package could benefit from offering the Spherical and Gaussian covariance functions. Next, the hierarchical bivariate model does not currently accept and model fixed effects. While we do not expect any issues with the posterior sampling and the full conditionals have been discussed in the previous chapter, for testing purposes, we have not yet included them in our implementation. One foreseeable hurdle is the impact on the runtime which is already sensitive to the dataset size.

Finally, one essential improvement going forward is the reduction of the mem-

ory footprint of the functions in the package. Currently, the memory allocation is heavily solicited to keep the distance matrices available at all times. Similar to the `spNNGP` package, the model could be improved by a more clever implementation which does not require keeping large arrays in memory. The size of the posterior samples array outputted by the hierarchical NN-RCM models also causes memory allocation issues. In the future, it could be beneficial to include the possibility to routinely save the posterior samples externally or devise an alternative method to save the samples using more efficient objects. In particular, this issue will be highlighted in the next chapter with our discussion of the Geostationary Operational Environmental Satellites (GOES) for albedo assessments.

Chapter 5

Surface Albedo Case Study

As mentioned in Chapter 1, one of the motivations for the development of the methods considered in this dissertation is to analyse the surface albedo assessments obtained from the National Oceanic and Atmospheric Administration's (NOAA) Geostationary Operational Environmental Satellites (GOES). The ultimate goal in this chapter is to reconcile the observations obtained from both satellites, GOES-East and GOES-West.

In the first section we discuss an alternative to the 10-day average currently being used. We apply the distributed nearest-neighbor Gaussian process (NNGP) implementation in order to develop a sequential updating for the surface albedo predictions. The current approach creates a product with a disconnect between time and space. What we suggest is a seamless update of the albedo predictions after each day of a 10-day period. This is demonstrated with a proof of concept for a three-day average.

In the second section we tackle the core of the satellite reconciliation project. We apply the bivariate spatial model developed in Chapter 3 which will allows us to merge the observations from the two satellites using a common surface $w(s)$ and a differential surface $d(s)$. We start our analysis with an area limited to a small portion of the Midwest to illustrate how the model performs in comparison to its univariate

counterpart. We then extend our analysis to a larger area by looking at the albedo assessments over the state of Texas. Finally, we present the final results of the bivariate model over the Continental United States (CONUS).

5.1 Sequential updating of surface albedo

We apply the divide-and-conquer approach to a subset of surface albedo measurements obtained from NOAA's GOES East and West. The GOES-East and GOES-West are two Earth satellites that assess the bi-hemispherical reflectance factor (BHRiso) daily. In this analysis, we wish to demonstrate the sequential potential of the divide-and-conquer algorithm. In our settings, the GP can be fitted once per day after which the data is no longer needed. This implies that at no point in time one needs to have the data for more than one day on one server. This is especially useful for the task of recording multi-day averages from massive datasets.

5.1.1 Sequential updating

To perform our experiment, we selected data from the GOES East and West satellites for three days in the month of July 2000. Albedo varies slowly in time and its retrieval can be affected by atmospheric conditions prevalent in a given day. Thus, it is reasonable to combine several days of data to obtain updated predictions of the albedo estimates. As presented in our results, we highlight the evolution of the predictions updates over the course of July 1st-3rd, 2000. The data is retrieved at a resolution of 4km by 4km. By restricting our analysis to CONUS, the result is a dataset with approximately 665 000 observations per day for a total of roughly 2 million observations over three days. As previously noted, the albedo product is captured by a percentage of the weight between upward and downward radiant influxes. In order to transform the range to be applicable for the GP model, we use a logit transformation on the data.

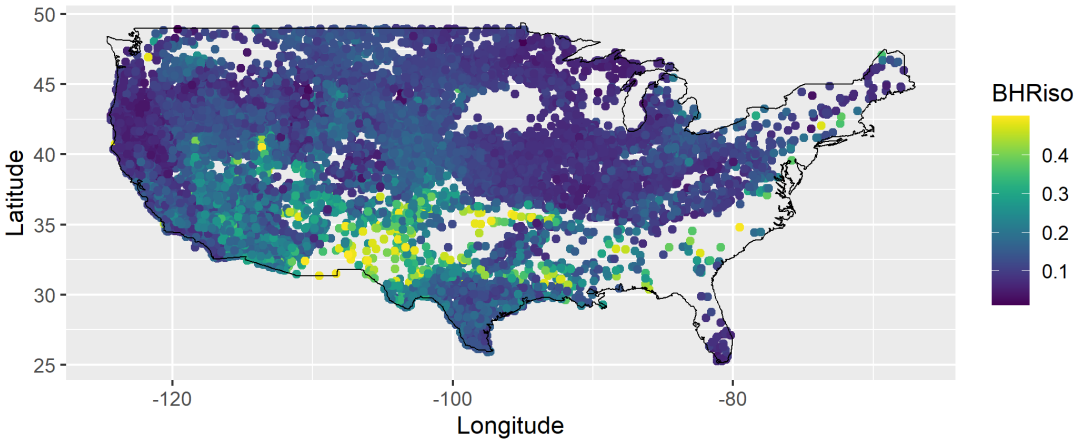


Figure 5.1: Subset of surface albedo observations from July 1, 2000.

Figure 5.1 shows a subset of size 10,000 for the July 1st, 2000 data. Points colored bright yellow are likely to correspond to areas with dense cloud covering.

As mentioned in Section 3.2, a reference set is needed to implement our proposed method. To create the set, we first generated an equidistant grid of size 50 by 50 based on a rectangle spanning over CONUS. By cropping the grid to the inland locations, the size of the final reference grid was reduced to 1,650 locations.

The covariance function used in the results presented was the Matérn with smoothing parameter $\nu = 3/2$. Different smoothness parameters were investigated but did not lead to significant differences in the resulting predictions. The point estimates for the nugget τ^2 and the covariance range parameter ϕ were obtained after each day by maximizing the marginal likelihood over a grid of possible values. The grid for τ^2 included values between 0.5 and 1.00, and the grid for ϕ ranged from 5 and 10. For all three days, the pair of parameters that maximized the marginal likelihood were the same at $\tau^2 = 0.55$ and $\phi = 6$.

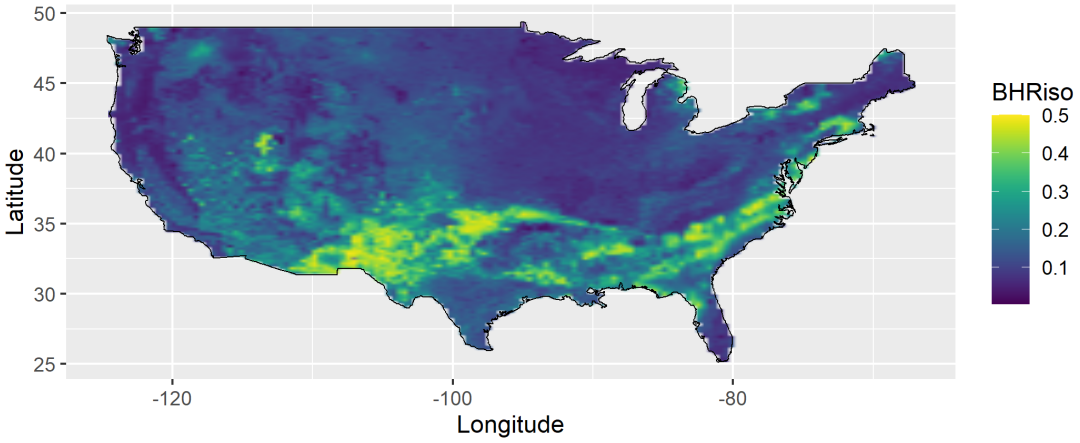


Figure 5.2: Posterior predictive average of surface albedo for July 1, 2000.

Using the point estimates obtained from the marginal optimization, we used the `R` NNGP package `spNNGP` and the function `spConjNNGP` to obtain posterior predictions after each day. The cumulative predictions over the three days are shown in Figures 5.2 to 5.4. For each day, we computed the posterior parameters by separating the new incoming data in 100 blocks. We then combined the new values to the current estimates of the posterior parameters. Once the new parameters are obtained, the data can be discarded as they are no longer needed for future updates.

5.1.2 Runtime comparison

Each day, obtaining the sufficient statistics to compute the marginal likelihood over a range of values for τ^2 and ϕ took approximately 24 minutes. Once these sufficient statistics are obtained, the runtime to combine them to maximize the marginal likelihood is negligible. The analysis was also performed on a Razer Blade 15 with Intel(R) Core(TM) i7-10750H Processor with 16.0GB of RAM. Obtaining predictions using the optimal parameters using `spConjNNGP` took less than one minute.

As a comparison we used the function `spConjNNGP` to perform model inference over the same data using the same grid for τ^2 and ϕ. The function `spConjNNGP` uses

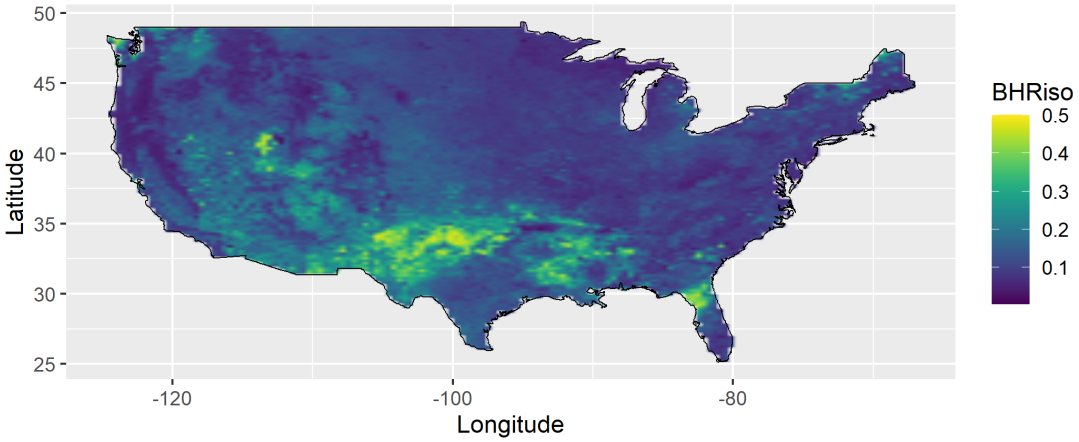


Figure 5.3: Posterior predictive average of surface albedo for July 1-2, 2000.

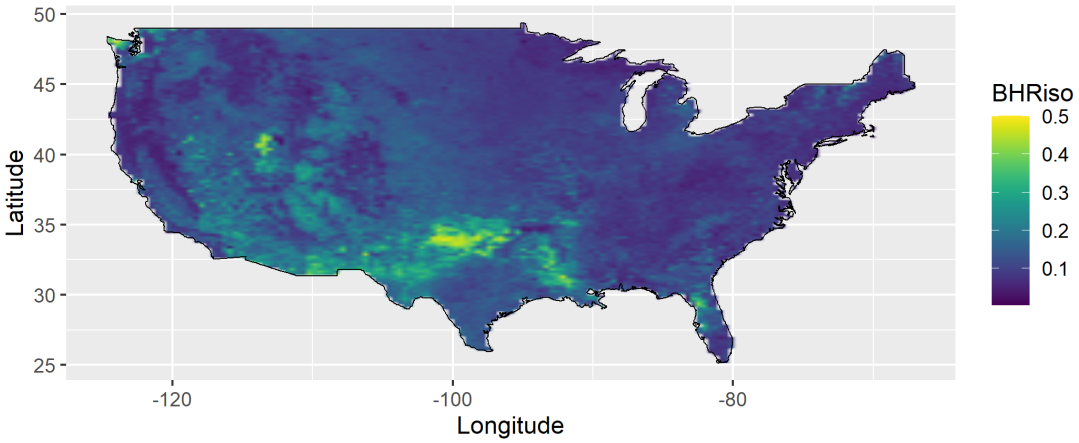


Figure 5.4: Posterior predictive average of surface albedo for July 1-3, 2000.

cross-validation to select the optimal point estimates for τ^2 and ϕ. This inference must then be repeated from scratch adding the additional data for each new day. The runtime therefore increases each day, in this case, from 7 minutes on the first day to around 22 minutes on the third day. Although our implementation of the sequential method is not fully optimized, we see its potential for sequential updating, as the runtime is constant over time as opposed to the observed linear increase of the function `spConjNNGP`.

We also compared our methodology to the MCMC implementation of NNGP using the function `spNNGP` of the `spNNGP` package. Over the course of the three days, the runtime increases from 20 minutes for the first day to 65 minutes needed to handle the data accumulated after the third day. It is worth noting that while this method is superior in terms of quantifying and propagating the uncertainty quantification of τ^2 and ϕ, we noticed poor mixing of the MCMC samples. In order to obtain better inference, we would need to rerun the model with updated starting values, priors and tuning parameters.

In summary, we see that the possibility of updating the sufficient statistics each day to obtain the marginal likelihood is beneficial for the runtime in the long-run. Each day, the sufficient statistics for the new dataset can be retained and combined with any subset of previous observations. For applications where cumulative predictions are needed each day, maintaining a constant runtime can be particularly beneficial.

5.2 Bivariate surface albedo (cropland)

The model that we seek to use to analyse the albedo observations relies on two spatial processes, a common surface and a differential surface. Let s be any location in our space of interest, we denote the two possible observations from the GOES-East and the GOES-West satellites as $y_E(s)$ and $y_W(s)$. We attempt to reconcile them to obtain a common surface albedo represented by the spatial process $w(s)$ and a discrepancy

quantification from the process $d(s)$. The model we use is therefore denoted as

$$\begin{pmatrix} f(y_E(s)) \\ f(y_W(s)) \end{pmatrix} = X(s)\boldsymbol{\beta} + \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} w(s) \\ d(s) \end{pmatrix} + \begin{pmatrix} \epsilon_1(s) \\ \epsilon_2(s) \end{pmatrix},$$

where f is any appropriate transformation to the real line, $X(s)$ and $\boldsymbol{\beta}$ represent the covariates and fixed effects to be included, and $\epsilon_i \sim N(0, \tau_i^2)$, for $i = 1, 2$. The covariates included in $X(s)$ are an intercept and the longitude and the latitude of the location s. The choice of having common fixed effects $\boldsymbol{\beta}$ for both sources of information reflects our assumption that the effects of the longitude and latitude on the albedo assessments are not impacted by the satellite retrieval process.

As an initial exploration, we focus our analysis on croplands from the Midwest where the satellites GOES-East and GOES-West contain around 18,000 and 6,000 observations respectively. To capture the predictive fit of the model, we first split the data into two equal subsets which we refer to as the training set and the testing set. We will first fit the model on the training set only to evaluate the predictive power. Then, we will fit the model once more on the full 24,000 datapoints to obtain high resolution posterior predictive surfaces.

To fit this model as efficiently as possible, we proceed with a multi-tier approach. First, we discuss the transformation f and the fixed effects. Second, we look at the univariate marginal nearest-neighbor Gaussian processes with rancom covariance matrices (NN-RCM) models used to obtain point estimates for the covariance parameters and the degrees of freedom. Since the bivariate marginal model does not output point estimates for linear combinations of the observations, we use the univariate marginal NN-RCM model in a two-step process. Finally, we use the bivariate hierarchical NN-RCM model to obtain the posterior inference on $w(s)$ and $d(s)$. Using these posterior samples, we obtain posterior predictive samples for both $w(s)$ and $d(s)$ which help us understand the uncertainty of our predictions and the difference between the two GOES satellites.

Covariate	Mean	St. dev.
Intercept	-2.4466	0.0820
Longitude	-0.0017	0.0005
Latitude	-0.0032	0.0015

Table 5.1: Posterior mean of the fixed effects of the stacked albedo linear model for the Midwest croplands.

5.2.1 Univariate marginal model tiered approach

As we previously explained in the sequential updating problem, the albedo observations are percentages and first need to be transformed. Similarly, we used a logit transformation to transform the percentages to the real line. We then use a univariate linear model on the stacked training observations to remove the average fixed effects. Table 5.1 summarizes the posterior mean and standard deviation for the fixed effects obtained using the `lm` function in `R`. From these, we can remove the predicted expected value from our transformed observations and continue with our spatial analysis.

Denote the transformed and centered observations as $y_E^*(s)$ and $y_W^*(s)$. The bivariate marginal NN-RCM model is fitted on two spatial processes that correspond to each source of information. Therefore, in this scenario, the point estimates that the bivariate marginal NN-RCM model obtains do not correspond to the covariance parameters for $w(s)$ and $d(s)$. Instead, we suggest using a two-step method to obtain an approximation of the those covariance parameters.

First, we fit the univariate marginal model using the observations from the first satellite, GOES-East only. The runtime for this first step was 22.15 seconds. Using the posterior inference obtained, we extract predictions $y_W^{E_(b)}(s)$, for $b = 1, ..., 1000$, for each the observed locations s of the second satellite, GOES-West. This second step took 1.43 seconds. Finally, we fit a second univariate marginal NN-RCM model on the difference between the observed value and the average predicted value for the second source:

$$y_W(s) - \sum_{b=1}^{1000} \frac{y_W^{E_(b)}(s)}{1000} \sim NN - RCM(0, C_\theta).$$

This last step took 10.28 seconds for an overall runtime of 34 seconds. Together, the two models supply us with approximates for $\xi_w^2, \xi_d^2, \sigma_w^2, \sigma_d^2, \nu_w, \nu_d, \alpha_w, \alpha_d$. We can therefore use the proxy nugget estimates and the partial sill estimates directly. For the range parameter, since the bivariate model uses a common range, we have multiple options. We decided to use the average of the two estimates, but another choice was to use the smallest of the two ranges. Finally, for the degrees of freedom, we can obtain the point estimate for α by adding the two estimates. We therefore have $\alpha = 2 * (\alpha_w + \alpha_d)$, where the estimate must be multiplied by 2 to account for the misalignment of the satellites. More precisely, the two satellites do not share the same assessment grid which results in a bivariate dataset with exactly 50% missing data.

This methodology does not provide us with an estimate for the cross-covariance term of the Matérn covariance function, ρ_{wd}. A good approximation of the parameter is obtained by computing the empirical correlation between the posterior predictive averages for the second satellite and the difference between the albedo assessments and the posterior predictive means. Carrying this calculation and adjusting for the smoothness parameters, we obtain an estimate of $\rho_{wd} = -0.0770$. See Table 5.2 for the complete list of point estimates obtained from the univariate marginal NN-RCM models approach.

5.2.2 Bivariate hierarchical model

The final step of our spatial inference analysis consists of fitting the bivariate hierarchical NN-RCM model to $y_E^*(s)$ and $y_W^*(s)$. The first few steps have allowed us to reduce the model to the following,

$$\begin{pmatrix} y_E^*(s) \\ y_W^*(s) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} w(s) \\ d(s) \end{pmatrix} + \begin{pmatrix} \epsilon_1(s) \\ \epsilon_2(s) \end{pmatrix},$$

Parameter	Notation	$w(s)$	$d(s)$	Bivariate estimate
Degrees of freedom	$\alpha_w, \alpha_d, \alpha$	8960	2727	79582
Range	ν_w, ν_d, ν	1.1047	1.5790	1.3419
Partial Sill	σ_w^2, σ_d^2	0.0361	0.0769	-
Correlation	ρ_{wd}	-	-	-0.0770
Nugget	ξ_w^2, ξ_d^2	1.0499	0.0962	-

Table 5.2: Point estimates for the covariance parameters obtained from the multi-tiered approach using the univariate marginal NN-RCM models for the Midwest croplands.

Model	Runtime	Satellite	PMSE	CRPS	PPLC	Coverage
Bivariate NN-RCM	1.76 mins	GOES-E	0.0096	0.0713	590.59	0.9992
		GOES-W	0.0250	0.0975	241.46	0.9993
Univariate NN-RCM	1.11 mins	GOES-E	0.0205	0.0799	358.06	0.9662
		GOES-W	0.0894	0.1984	196.95	0.6975

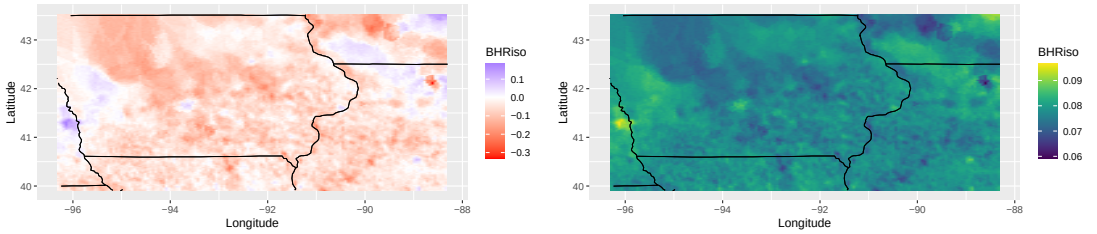
Table 5.3: Scores comparison for the hierarchical NN-RCM model and the univariate marginal NN-RCM model for the Midwest croplands. The scores for the GOES-East and GOES-West testing sets are reported individually.

where $\epsilon_i \sim N(0, \tau_i^2)$, for $i = 1, 2$. We obtain posterior samples for the spatial random effects using the bivariate hierarchical NN-RCM model with the point estimates as outlined in Table 5.2. Overall, the posterior inference took 1.76 minutes to complete. Using these posterior samples, we complete the final step of the cross-validation analysis by sampling the posterior predictive distribution of the testing set locations. With those, we can compute the four scoring values and compare them to using a univariate NN-RCM model on the stacked dataset. Table 5.3 summarizes the results and shows that the bivariate model does indeed outperform its univariate counterpart.

5.2.3 High resolution predicted surface

Repeating the same steps but for the full Midwest dataset, we again obtain posterior samples for the bivariate hierarchical NN-RCM model. Using these samples

we generate posterior predictive samples for a high resolution grid. Figure 5.5 shows the posterior mean for the spatial random effect $w(s)$ and the albedo predictions on the original scale.



(a) $w(s)$ (b) Albedo predictions

Figure 5.5: Spatial random effects $w(s)$ and albedo predictions from the bivariate hierarchical NN-RCM model for the Midwest croplands.

We are particularly interested in the spatial surface $d(s)$ to learn about the differences between the two satellites GOES-East and GOES-West. Figure 5.6 shows the posterior predictive mean for $d(s)$ and the posterior predictive difference between $y_E(s)$ and $y_W(s)$ on the original scale. That is, for a location s, we plot the average of the following quantity:

$$d_b^*(s) = f^{-1}(w_b(s) + d_b(s) + X(s)\boldsymbol{\beta}) - f^{-1}(w_b(s) + X(s)\boldsymbol{\beta}),$$

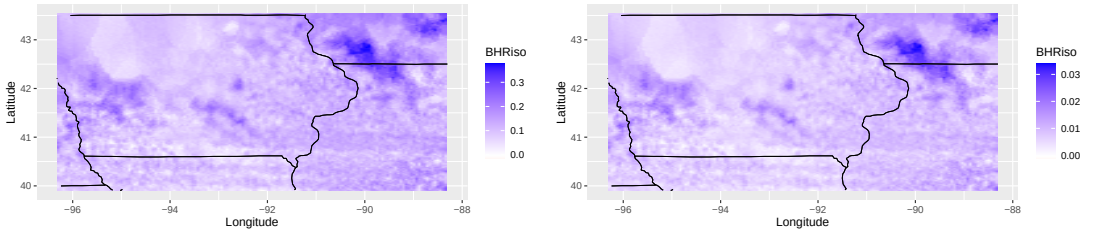
where $b = 1, ..., 1000$.

We immediately notice that the discrepancy between the two satellites presents irregular spatial patterns. A simple solution such as investigating the longitude/latitude or the satellite view angle is not reasonable in view of these results. One plausible avenue would be to look in more detail at the topography and the land cover of the area.

By looking at the 90% posterior predictive intervals at each location s, we note that only 6% of them are significant. Reducing the interval to contain 80% of the distribution, the significance increases to 20%. Therefore, we conclude that the discrepancy is highly localized which reinforces the idea that the model would benefit

from additional topographic information.

In the next sections, we look at larger spatial areas of CONUS. Taking into consideration the high variability of the field $d(s)$, we want to gauge the significance of discrepancy in a more global model.



(a) $d(s)$ (b) $y_E(s) - y_W(s)$

Figure 5.6: Spatial random effects $d(s)$ and posterior predictive mean of $y_E(s) - y_W(s)$ from the bivariate hierarchical NN-RCM model for the Midwest croplands.

5.3 Bivariate surface albedo (Texas)

As a second phase in our analysis, we investigate the differences between the bivariate and univariate NN-RCM models for a larger area. We look at the state of Texas which contains around 40,000 observations from each satellite. In this case, while the datasets are balanced, they both contain a large number of missing values for the northern portion of the state due to cloud coverage. We proceed again with the univariate multi-tier approach to obtain point estimates for the covariance parameters.

5.3.1 Univariate marginal model tiered approach

We first start by transforming the dataset which includes removing the fixed effects. Table 5.4 summarizes the posterior mean and standard deviation for the fixed effects obtained using the `lm` function in `R`. After removing the predicted expected value from our transformed observations, we continue with the univariate models.

Covariate	Mean	St. dev.
Intercept	-11.9600	0.0524
Longitude	-0.0817	0.0005
Latitude	0.0753	0.0006

Table 5.4: Posterior mean of the fixed effects of the stacked albedo linear model for the state of Texas

Parameter	Notation	$w(s)$	$d(s)$	Bivariate estimate
Degrees of freedom	$\alpha_w, \alpha_d, \alpha$	23801	15990	79582
Range	ν_w, ν_d, ν	0.6193	0.4123	0.5158
Partial Sill	σ_w^2, σ_d^2	0.4094	0.1475	-
Correlation	ρ_{wd}	-	-	-0.0428
Nugget	ξ_w^2, ξ_d^2	0.2577	0.6952	-

Table 5.5: Point estimates for the covariance parameters obtained from the multi-tiered approach using the univariate marginal NN-RCM models for the state of Texas.

The three steps of the univariate multi-tier approach were performed with an overall runtime of 75 seconds. The first step which consists of fitting the marginal univariate NN-RCM model on the observations from the GOES-East satellite took 40.21 seconds. Obtaining the posterior predictive samples for the GOES-West observed locations then took 8.46 seconds. Finally, the second univariate model which looks at the difference between the observed albedo assessment from the GOES-West satellite and the posterior predictive mean obtain from the first model ran in 26.03 seconds. In Table 5.5, we summarize the list of point estimates obtained from the univariate marginal models approach.

5.3.2 Bivariate hierarchical model

Using the point estimates obtained in the previous section we sample the bivariate hierarchical NN-RCM model in 31.17 minutes. Finally, we obtain the posterior

Model	Runtime	Satellite	PMSE	CRPS	PPLC	Coverage
Bivariate NN-RCM	31.2 mins	GOES-E	0.0837	0.1735	8914.36	0.9962
		GOES-W	0.1276	0.2290	12121.86	0.9991
Univariate NN-RCM	1.1 mins	GOES-E	0.0530	0.1207	1915.54	0.9600
		GOES-W	0.0613	0.1320	1358.17	0.9433

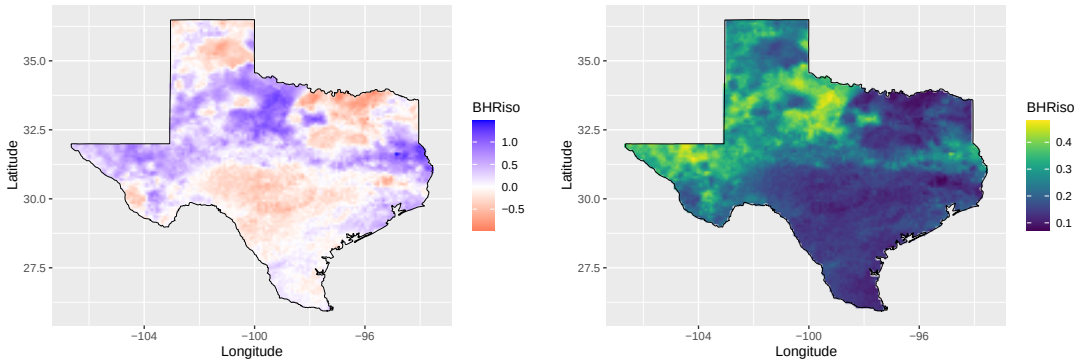
Table 5.6: Scores comparison for the hierarchical NN-RCM model and the univariate marginal NN-RCM model for the state of Texas. The scores for the GOES-East and GOES-West testing sets are reported individually.

predictive distribution samples for the testing locations and perform our cross-validation of the model. Table 5.6 summarizes the results and shows that for this larger area, the bivariate model does not perform better than the univariate marginal NN-RCM model. To understand why the model is not as successful as in the croplands case, we investigate the difference surface of the high resolution predictions in the next section.

5.3.3 High resolution predicted surface

Repeating the same steps but for the full Texas dataset, we again obtain posterior samples for the bivariate hierarchical NN-RCM model. Using these samples we generate posterior predictive samples for a high resolution grid. Figure 5.7 shows the posterior mean for the spatial random effect $w(s)$ and the albedo predictions on the original scale respectively.

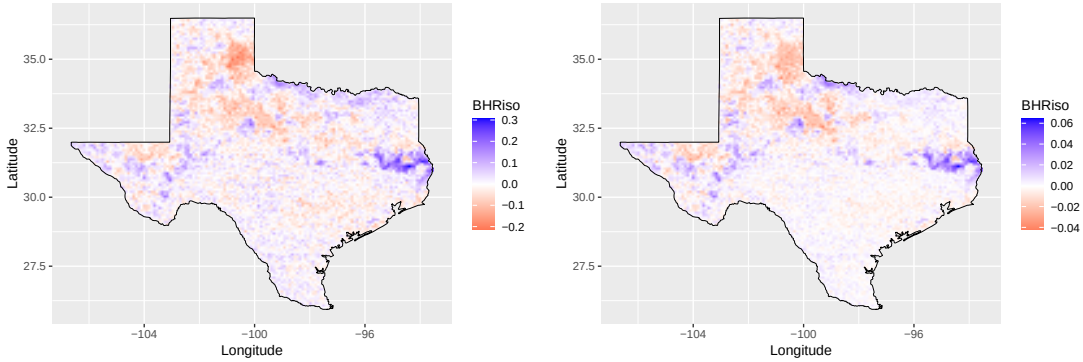
Once again we pay special attention to $d(s)$ to learn about the differences between the two satellites GOES-East and GOES-West. Figure 5.8 shows the posterior predictive mean for $d(s)$ and the posterior predictive difference between $y_E(s)$ and $y_W(s)$ on the original scale. We immediately notice that on the original scale, the difference surface is very close to 0. Looking at the 90% predictive intervals at each location s, we see that over 99% of them include 0. This is a strong indication that the difference between the satellites for the state of Texas is likely not significant. Other the hand, it may also further suggest that the global variance of the model overwhelms the local



(a) $w(s)$ (b) Albedo predictions

Figure 5.7: Spatial random effects $w(s)$ and albedo predictions from the bivariate hierarchical NN-RCM model for the state of Texas.

variability of $d(s)$. As we saw for the cropland area, the spatial layout of the discrepancy field indicate that larger differences occur in localized patterns. As concluded earlier, on a large scale level, the univariate model is sufficient to obtain joint predictions from both sources of information.



(a) $d(s)$ (b) $y_E(s) - y_W(s)$

Figure 5.8: Spatial random effects $d(s)$ and posterior predictive mean of $y_E(s) - y_W(s)$ from the bivariate hierarchical NN-RCM model for the state of Texas.

Covariate	Mean	St. dev.
Intercept	-1.1250	0.0066
Longitude	-0.0088	0.0001
Latitude	-0.0436	0.0001

Table 5.7: Posterior mean of the fixed effects of the stacked albedo linear model for CONUS

Parameter	Notation	$w(s)$	$d(s)$	Bivariate estimate
Degrees of freedom	$\alpha_w, \alpha_d, \alpha$	386233	278686	1329822
Range	ν_w, ν_d, ν	1.1402	3.2358	2.1880
Partial Sill	σ_w^2, σ_d^2	0.4522	0.0705	-
Correlation	ρ_{wd}	-	-	0.0705
Nugget	ξ_w^2, ξ_d^2	0.1879	0.7995	-

Table 5.8: Point estimates for the covariance parameters obtained from the multi-tiered approach using the univariate marginal NN-RCM models for CONUS.

5.4 Bivariate surface albedo (CONUS)

In this final section, we repeat the analysis one additional time for the full albedo dataset over CONUS. Table 5.7 summarizes the posterior mean and standard deviation for the fixed effects obtained using the `lm` function. Continuing with out two-step univariate approach, we fit the univariate marginal model using the observations from the first satellite, GOES-East only. The runtime for this first step was 12.24 minutes. Before fitting the second univariate marginal model, we obtained the predictions for the second satellite which took 4.68 minutes. Finally, the second univariate marginal model ran for 17.24 minutes for an overall runtime of 34.16 minutes.

Using the same approach as for the Midwest croplands and for Texas, we obtain the point estimates for the bivariate hierarchical NN-rCMc model which are summarized in 5.8. We notice that the cross-covariance parameter estimate $\rho_{wd} = -0.0705$ is similar to that of the two previous areas.

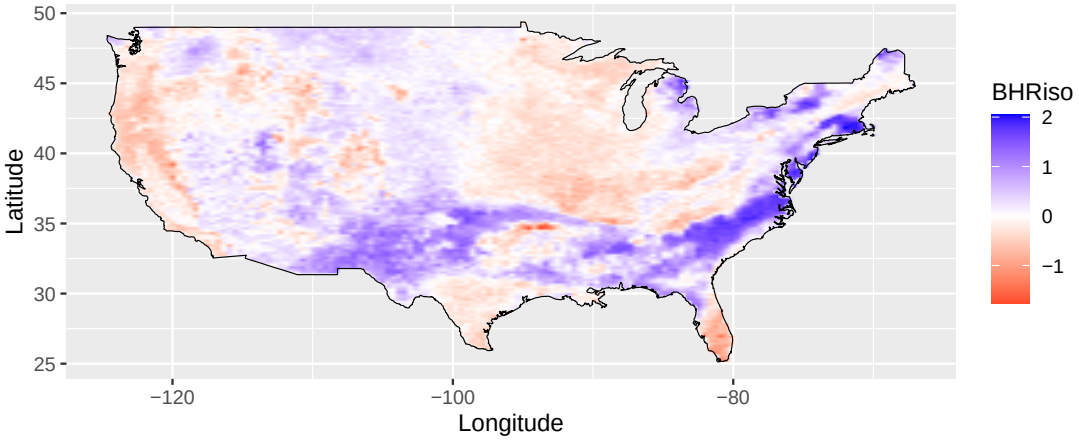


Figure 5.9: Predicted spatial random effects $w(s)$ from the bivariate hierarchical NN-RCM model for CONUS.

5.4.1 Bivariate hierarchical model

Unlike the smaller examples, we encountered a memory allocation limit on the number of posterior samples that can be obtained. Namely, we can only output 50 bivariate samples for the 631,441 observed locations. In total, the object has size over 500Mb. The runtime is also impacted by the limited CPU usage. Overall, the posterior inference took 20 hours to complete. Using these samples we generate posterior predictive samples for a high resolution grid. Figure 5.9 and 5.10 show the posterior mean for the spatial random effect $w(s)$ and the albedo predictions on the original scale respectively.

Figure 5.11 and Figure 5.12 show the posterior predictive mean for $d(s)$ and the posterior predictive difference between $y_E(s)$ and $y_W(s)$ on the original scale. In this scenario also, the 90% predictive intervals overlap 0 in 99.9% of the predicted locations. It is even more apparent in such a large area that the discrepancy is significant only at the local level. In this example, $d(s)$ shows very few spatial patterns even in areas like the Midwest where were saw a significant difference in an earlier example. One could expect that this is due to the lower resolution of the posterior predictive fields over

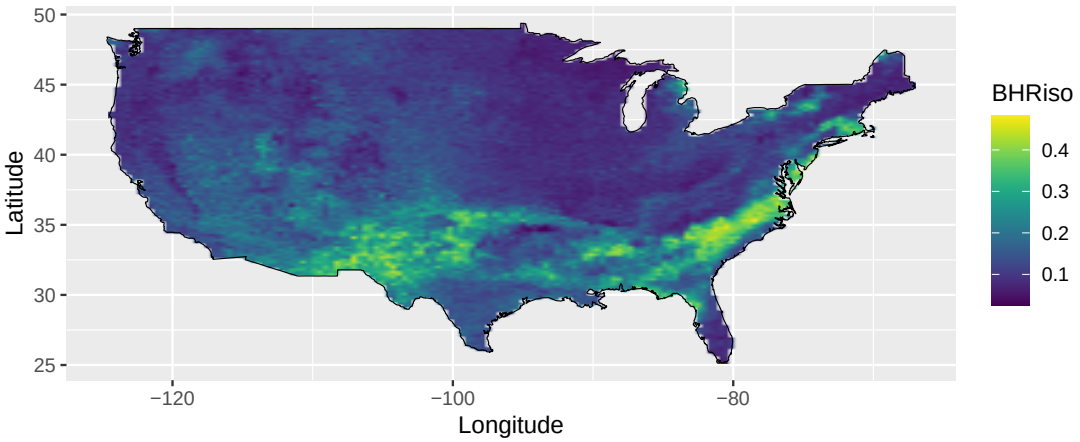


Figure 5.10: Albedo predictions from the bivariate hierarchical NN-RCM model for CONUS.

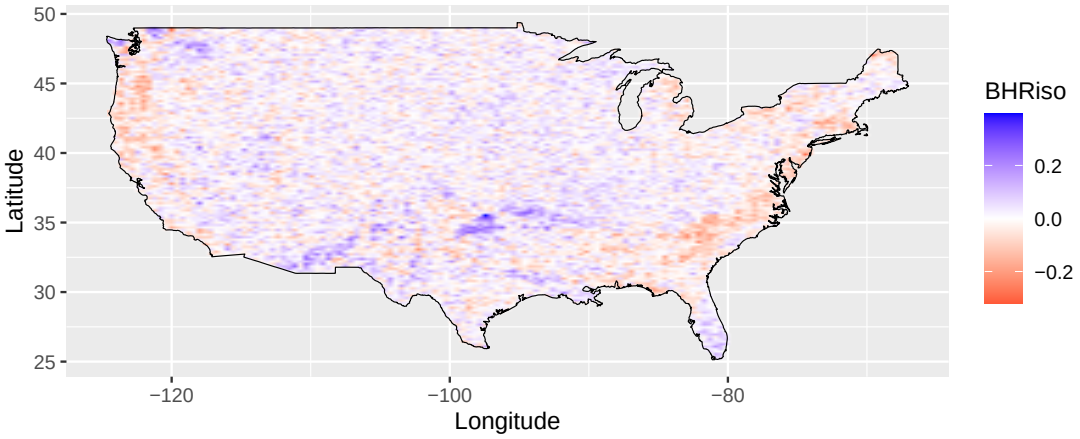


Figure 5.11: Predicted spatial random effects $d(s)$ from the bivariate hierarchical NN-RCM model for CONUS.

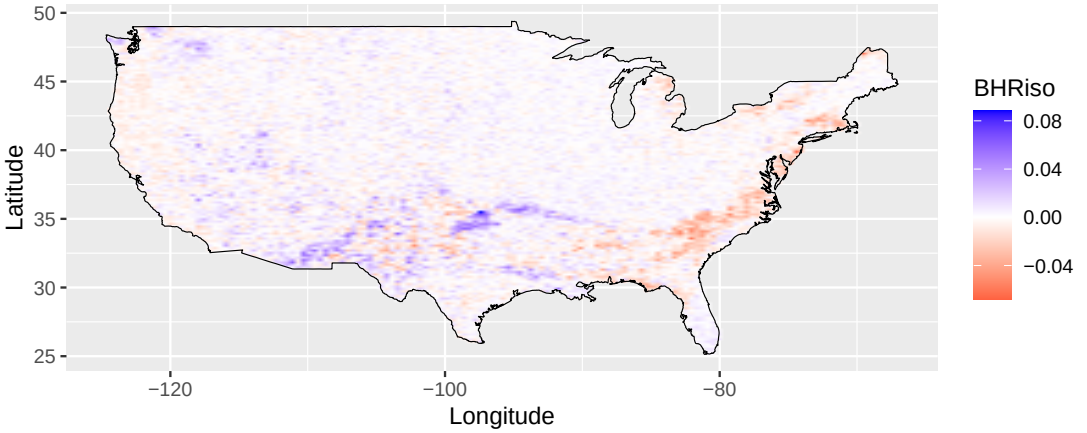
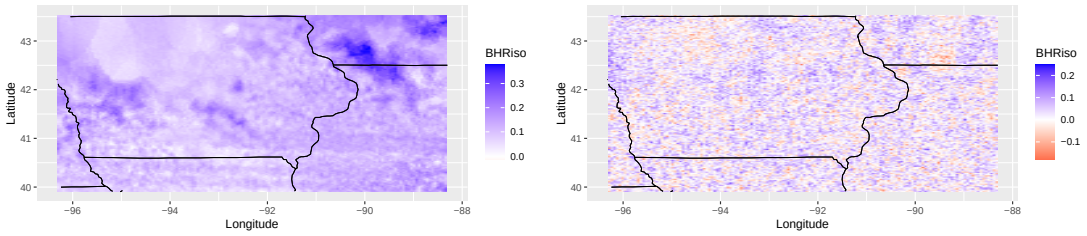


Figure 5.12: Posterior predictive mean of $y_E(s) - y_W(s)$ from the bivariate hierarchical NN-RCM model for CONUS.

the croplands. In Figure 5.13, we use the same predictive grid to output the posterior predictive mean over the Midwest area. We see that despite the higher resolution, the spatial field under the full CONUS model is still a rendition of white noise. Therefore, the final conclusion is that the model with the full dataset smooths the assessements of the smaller subsets and the global variance is too large to recover the regional variability.



(a) Midwest model (b) CONUS model

Figure 5.13: Spatial random effects $d(s)$ from the bivariate hierarchical NN-RCM model using the Midwest dataset vs using the full CONUS dataset.

5.5 Conclusions and future work

In this chapter, we have first successfully implemented a very substantive example of sequential updating for NNGP. We have developed a methodology that allows for multi-day averaging without running the model over the same dataset more than once.

Second, we have demonstrated the ability of the univariate and bivariate NN-RCM models to deal with increasingly large datasets. In addition to facing continental scale data, we have also tackled the challenge of misaligned datasets. As noted, the two satellites GOES East and GOES West do not share their retrieval grid. In other words, we overcame datasets with exactly 50% missing data. Finally, the linear structure of the bivariate hierarchical NN-RCM model allowed us to quantify the discrepancy between the two satellites. Using this knowledge, we were able to conclude that the differences in the albedo assessements fall on more intrinsic characteristics than the Zenith viewing angle of the satellite.

From our analyses of the various areas of CONUS, we concluded that for large scale predictions, stacking the observations from both satellites and using the univariate NN-RCM model is sufficient. The next step in this analysis will be to look at predictions for the subsequent days of data. Using the univariate model, we can also compare the daily predictions to the stacked predictions from multiple days. Since the `NNRCM` package can handle duplicates, stacking multiple days of data does not bring any issues to the model.

In terms of small scale analysis, for example with the croplands, the bivariate model does provide a good understanding of the spatial differences between the satellites. As shown in our cross-validation results, the bivariate model outputs a smaller predictive mean squared error and a better coverage of the true values. The bivariate model is therefore useful to gain an understanding of specific differences. While in this case we

were looking at croplands in the Midwest, the next step will be to look at other regions with interesting geographic characteristics. For example, we could look at urban regions, grasslands, shrublands, forests and deserts.

One of the next stage of our analysis will be to consider the extension to the full scale of the satellites. The viewing angle of GOES-East and GOES-West go well beyond CONUS to include all of the Americas. Moreover, it will be essential to look at an extended timeframe outside of the month of July to see how the discrepancy evolves over time.

Finally, while we were interested in the white sky albedo assessement (BHRiso), the GOES East and GOES West satellites also retrieve the directional Hemispherical Reflectance (DHR30) also called the black sky albedo. It would be not only possible to model DHR30 as we did for BHRiso, but it would also be interesting to design a bivariate model to understand the differences between the two ratings.

Chapter 6

Discussion

The importance of scalable methodologies in the realm of spatial statistics has been firmly established. Many of the existing approaches that tackle massive spatial data rely on approximations with result in non-stationary spatial models. Our main objective was to shift the priorities by developing a non-stationary and non-isotropic spatial model that is also computationally efficient. In this chapter, we summarize the achievements and the next steps of the development of the nearest-neighbor Gaussian process with random covariance matrix (NN-RCM) model. Before discussing the NN-RCM model, however, we first review the implementation and the future for the distributed nearest-neighbor Gaussian process (NNGP) methodology.

6.1 Distributed NNGP

While NNGP already benefit from a highly efficient implementation, we have highlighted specific situations where it may not be sufficient. For example, in the case of distributed datasets where one may not be able to access the data all at once, an alternative solution must be defined. We proposed a distributed approach which allows to recover the exact posterior inference by applying the model on blocks of data. By using a common reference set for all blocks, we are able to aggregate the posterior

parameters and obtain the global posterior distribution. Another frequent situation involves recurring data. In that case, it is desirable to devise a sequential updating plan for the model. The same distributed approach can be applied which results in a constant runtime over time. We have illustrated this concept in the albedo case study where we wish to obtain multi-day posterior predictive averages. By obtaining the posterior parameters for each day of data for a common reference set, we are able to concatenate the days of interest to obtain the global posterior distribution. We have shown that the runtime of competing methods will increase linearly with the size of the dataset as opposed to our method which keeps a constant runtime based on one day of data.

One of the next step for the distributed method will be to extend the procedure for multivariate NNGP. Joint modeling of spatial processes is commonly found in many applications and present an even bigger computational challenge. Having a divide-and-conquer method could be helpful especially in situations where the evolution of data over time is of interest.

6.2 NN-RCM

We introduced a non-isotropic extension to the NNGP which we called nearest-neighbor Gaussian process with random covariance matrices (NN-RCM). We have developed and implemented in the `R` package `NNRCM` both a univariate and a multivariate version of the model. Moreover, for both the univariate and multivariate settings, we established a marginal model and an extended hierarchical model.

By marginalizing the spatial random effects, we obtain the posterior marginal distribution of the model. The optimization of the posterior marginal distribution gives us point estimates for the covariance parameters very quickly. Using these point estimates, we can recover the posterior predictive distribution of the observations allowing

us to fully quantify the uncertainty of the predictions.

The extension to the hierarchical model allows to include fixed effects and an observational error to the structure. In this case, we use an MCMC algorithm to recover the full posterior distribution of all the parameters in the model. For a faster option, it is reasonable to fix the covariance parameters and the degrees of freedom using the previously discussed marginal model. In that case, the MCMC algorithm retrieves posterior samples for the spatial random effects, the fixed effects and the observational error while keeping the remaining parameters constant. Ultimately, this saves the algorithm from performing a Metropolis step.

Through the implementation of the NN-RCM model, we have dealt with multiple issues that are often overlooked. First, we addressed the issue of duplicate observations with a case study on the Mediterranean sea surface temperature (SST). The inclusion of a proxy nugget in the covariance structure of our model allows us to include duplicates without obtaining singular matrices. Allowing duplicates however must come with a thoughtful discussion on the size of the neighborhood. Ideally, the duplicates would be included in addition to a set number of unique neighbors. In the most extreme cases, where each observed location has q duplicates, the neighborhood size would be extended to mq, where m represents the number of neighboring locations. In the SST dataset, the number of duplicates was initially too large but we were able to reduce it by taking a variety of measures of variability. While the neighborhood were significantly extended, this resulted in a smaller predictive mean squared error and better continuous rank probability score and posterior predictive loss criterion when compared to the no duplicates NN-RCM and NNGP models.

A second challenge that we tackled was the presence of misaligned multivariate data. In the albedo case study, the missing data represented 50% of the data as the assessments from the two satellites were fully misaligned. Our implementation of both the bivariate marginal and hierarchical models support the case of missing observations.

For the marginal model, we create the neighborhoods and the likelihoods based on the available data and can obtain the posterior inference for the covariance parameters as such. In the hierarchical case, the spatial random effects $\boldsymbol{w}(s)$ can be be fully sampled even for missing observations. Therefore, the neighborhoods are created based on the full spatial random effects being available. The impact of the missing values are reflected in the likelihood of the spatial random effects. That is, the posterior distribution of the spatial random effects that are associated with a missing value relies only on its neighbors and its prior.

Through the scarcity of the nearest-neighbors and the linearization of the model, we achieve a scalable model which is fully parallelizable. One future goal is the possibility of implementing a distributed version of the presented model. While the current implementation does not allow for it directly, it would be possible to devise a sequential updating of the posterior parameters.

In the distributed NNGP model, we saved the marginal values for each cell of a grid of parameters. Then, by adding these tables for the days or blocks of interest, we could recover the global marginal values. Applying the optimization was then straightforward as selecting the combinations of parameters resulting in the highest global likelihood value. While a similar approach would be possible for the NN-RCM, it is not desirable as the dimension of the covariance parameters is larger. One would need thousands of combinations of the four covariance parameters in order to carry such an analysis.

A better alternative would be to use a sequence of starting values for the marginal optimization based on the previous days or blocks. For example, after observations are collected for one day, we can maximize the marginal and obtain point estimates for θ and α. On the following day, we can repeat the process by starting the optimization at the previous day's estimates.

There is one issue that we have not addressed which pertains to the creation

of the neighborhoods. In the case of the distributed NNGP, the neighbors were chosen from a common reference grid shared by each block of data. Therefore, the selection of the neighbors was consistent across each new day. In our case, we do not have a reference grid. To maintain the order of the observations, we would need to accumulate the observed data which defeats the purpose of the distributed implementation. Instead, we would explore a new notion of truncated neighborhoods for which each observations would only contain neighbors from the current day. That is, the ordering of the locations would effectively be restarted every day.

6.3 Albedo case study

In the final chapter, we presented a case study on the surface albedo assessments from NOAA's GOES East and West. The importance of surface albedo stems from its potential to display the rate of climate change. The first objective of our analysis was to explore multi-day posterior predictive averages. By using the previously discussed sequential updating NNGP model, we obtained evolving predictions over the course of three days. This method can then be extended to aggregate the posterior parameters of any days of interest. One use of this averaging, in conjunction with the quality rating of the assessments, would be to replace the 10-day product that is currently being produced. One of the next step in our analysis would be to create these averages and conduct a formal comparison with the existing statistic.

The second objective of the study was to quantify the discrepancy between GOES-East and GOES-West. While the two satellites seem similar on a global level, for smaller areas, the subtle differences were clearly visible. We used the bivariate hierarchical NN-RCM model to capture the spatial random effects as a common albedo surface $w(s)$ and a discrepancy surface denoted $d(s)$. Starting with a smaller area in the Midwest of the United States, we concluded that the difference between the two satellites

was not only significant but also dictated by a non-trivial spatial pattern. One possible idea would be to include additional fixed effects related to the topography of the land. We then moved on to larger areas, including the state of Texas and the whole CONUS area. Overall, these two examples included 80,000 and 630,000 datapoints respectively. The conclusions for both were that the global variability over large areas was too great to quantify the discrepancy. The uncertainty surrounding the discrepancy surface was too large to be significant. This tells us that, at this time, with the current additional information included in the model, combining the assessments from both satellites into a univariate model is sufficient when it comes to large scale predictions.

The next stage of our analysis will be to expand the model to the entirety of the data available from both satellites. This includes most of the Americas totalling over two millions observations each day. It is also of interest to understand if the time of year has an impact on the discrepancy between the satellites. While we focused on the month of July for CONUS, it will be interesting to see how surface albedo is impacted in the winter months. Finally, through our study of croplands in the Midwest, we were able to detect patterns in the differences between the satellites. It will be essential to study other small areas and identify what causes these spatial patterns and how to capture them with our bivariate NN-RCM model.

Bibliography

Joseph J. Allaire, Romain Francois, Kevin Ushey, Gregory Vandenbrouck, Marcus Geelnard, and Intel. *RcppParallel: Parallel Programming Tools for 'Rcpp'*, 2020. URL `https://CRAN.R-project.org/package=RcppParallel`. R package version 5.0.2.

Sunil Arya, David Mount, Samuel E. Kemp, and Gregory Jefferis. *RANN: Fast Nearest Neighbour Search (Wraps ANN Library) Using L2 Metric*, 2019. URL `https://CRAN.R-project.org/package=RANN`. R package version 2.6.1.

Sudipto Banerjee. High-dimensional Bayesian geostatistics. *Bayesian Analysis*, 12 (2):583–614, 2017. doi: 10.1214/17-BA1056R. URL `https://doi.org/10.1214/17-BA1056R`.

Sudipto Banerjee, Alan E. Gelfand, Andrew O. Finley, and Sang Huiyan. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):825–848, 2008a. doi: 10.1111/j.1467-9868.2008.00663.x. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2008.00663.x`.

Sudipto Banerjee, Alan E. Gelfand, Andrew O. Finley, and Huiyan Sang. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):825–848, 2008b. doi: https://doi.org/10.1111/j.1467-9868.2008.00663.x. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2008.00663.x`.

Sudipto Banerjee, Bradley P. Carlin, and Alan E. Gelfand. *Hierarchical Modeling and Analysis of Spatial Data*. Chapman and Hall, New York, second edition, 2014.

Douglas Bates and Martin Maechler. *Matrix: Sparse and Dense Matrix Classes and Methods*, 2018. URL `https://CRAN.R-project.org/package=Matrix`. R package version 1.2-15.

James O. Berger, Victor de Oliveira, and Bruno Sansó. Objective Bayesian analysis of spatially correlated data. *Journal of the American Statistical Association*, 96(456): 1361–1374, 2001. ISSN 01621459. URL `http://www.jstor.org/stable/3085905`.

Philip J. Brown, Nhu D. Le, and James V. Zidek. Multivariate spatial interpolation and exposure to air pollutants. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 22(4):489–509, 1994. ISSN 03195724. URL `http://www.jstor.org/stable/3315406`.

Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995. doi: 10.1137/0916069. URL `https://doi.org/10.1137/0916069`.

Noel A. C. Cressie. *Statistics for Spatial Data, Revised Edition*. John Wiley and Sons, New York, 1993.

Noel A. C. Cressie and Gardar Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):209–226, 2008. doi: 10.1111/j.1467-9868.2007.00633.x. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2007.00633.x`.

Noel A. C. Cressie and Cristopher K. Wikle. *Statistics for Spatio-Temporal Data*. Wiley, Hoboken, NJ, 2011.

Abhirup Datta, Sudipto Banerjee, Andrew O. Finley, and Alan E. Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016. doi: 10.1080/01621459.2015.1044091. URL https://doi.org/10.1080/01621459.2015.1044091.

Philip Dawid and Steffen L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21(3):1272–1317, 1993. ISSN 0090-5364. doi: 10.1214/aos/1176349260. Korrektion vedlagt, Annals of Statistics, 23:5, 1995.

Adrian Dobra, Chris Hans, Beatrix Jones, Joseph R. Nevins, Guang Yao, and Mike West. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196 – 212, 2004. ISSN 0047-259X. doi: https://doi.org/10.1016/j.jmva.2004.02.009. URL http://www.sciencedirect.com/science/article/pii/S0047259X04000259. Special Issue on Multivariate Methods in Genomic Data Analysis.

Juan Du, Hao Zhang, and V. S. Mandrekar. Fixed-domain asymptotic properties of tapered maximum likelihood estimators. *The Annals of Statistics*, 37(6A):3330 – 3361, 2009. doi: 10.1214/08-AOS676. URL https://doi.org/10.1214/08-AOS676.

Dirk Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer, New York, 2013. doi: 10.1007/978-1-4614-6868-4. ISBN 978-1-4614-6867-7.

Dirk Eddelbuettel and James Joseph Balamuta. Extending extitR with extitC++: A Brief Introduction to extitRcpp. *The American Statistician*, 72(1):28–36, 2018. doi: 10.1080/00031305.2017.1375990. URL https://doi.org/10.1080/00031305.2017.1375990.

Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ integration.

Journal of Statistical Software, 40(8):1–18, 2011. doi: 10.18637/jss.v040.i08. URL https://www.jstatsoft.org/v40/i08/.

Dirk Eddelbuettel and Conrad Sanderson. Rcpparmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics and Data Analysis*, 71: 1054–1063, March 2014. URL http://dx.doi.org/10.1016/j.csda.2013.02.005.

Andrew O. Finley, Sudipto Banerjee, and Bradley P. Carlin. spBayes: An R package for univariate and multivariate hierarchical point-referenced spatial models. *Journal of Statistical Software*, 19(4):1–24, 2007. URL https://www.jstatsoft.org/article/view/v019i04.

Andrew O. Finley, Sudipto Banerjee, Patrik Waldmann, and Tore Ericsson. Hierarchical spatial modeling of additive and dominance genetic variance for large spatial trial datasets. *Biometrics*, 65(2):441–451, 2009. ISSN 0006341X, 15410420. URL http://www.jstor.org/stable/25502305.

Andrew O. Finley, Sudipto Banerjee, and Alan E. Gelfand. spBayes for large univariate and multivariate point-referenced spatio-temporal data models. *Journal of Statistical Software*, 63(13):1–28, 2015. doi: 10.18637/jss.v063.i13. URL https://www.jstatsoft.org/index.php/jss/article/view/v063i13.

Andrew O. Finley, Abhirup Datta, Bruce Cook, Douglas Morton, Hans Andersen, and Sudipto Banerjee. Efficient algorithms for Bayesian nearest-neighbor Gaussian processes. *Journal of Computational and Graphical Statistics*, 28:1–37, 2018. doi: 10.1080/10618600.2018.1537924.

Andrew O. Finley, Abhirup Datta, and Sudipto Banerjee. *spNNGP: Spatial Regression Models for Large Datasets using Nearest Neighbor Gaussian Processes*, 2020. URL https://CRAN.R-project.org/package=spNNGP. R package version 0.1.4.

Montserrat Fuentes. A high frequency kriging approach for non-stationary environmental processes. *Environmetrics*, 12(5):469–483, 2001. doi: https://doi.org/10.1002/env.473. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/env.473.

Montserrat Fuentes and Richard Smith. A new class of nonstationary spatial models. *Journal of the American Statistical Association*, 02 2003.

Reinhard Furrer, Marc G. Genton, and Douglas Nychka. Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, 15(3):502–523, 2006. ISSN 10618600. URL http://www.jstor.org/stable/27594195.

Alan E. Gelfand and Sujit K. Ghosh. Model choice: A minimum posterior predictive loss approach. *Biometrika*, 85(1):1–11, 1998. ISSN 00063444. URL http://www.jstor.org/stable/2337305.

Alan E. Gelfand, Peter J. Diggle, Montserrat Fuentes, and Peter Guttorp. *Handbook of Spatial Statistics*. Chapman and Hall, Boca Raton, USA, 2010.

Marc G. Genton and William Kleiber. Cross-covariance functions for multivariate geostatistics. *Statistical Science*, 30(2):147 – 163, 2015. doi: 10.1214/14-STS487. URL https://doi.org/10.1214/14-STS487.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Tilmann Gneiting, William Kleiber, and Martin Schlather. Matérn cross-covariance functions for multivariate random fields. *Journal of the American Statistical Association*, 105(491):1167–1177, 2010. doi: 10.1198/jasa.2010.tm09420. URL https://doi.org/10.1198/jasa.2010.tm09420.

Yves Govaerts, Alessio Lattanzio, Malcom Taberner, and Bernard Pinty. Generating global surface albedo products from multiple geostationary satellites. *Remote Sensing of Environment - REMOTE SENS ENVIRON*, 112:2804–2816, 06 2008. doi: 10.1016/j.rse.2008.01.012.

Robert B. Gramacy and Herbert K. H. Lee. Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008. ISSN 01621459. URL `http://www.jstor.org/stable/27640148`.

Isabelle Grenier and Bruno Sansó. Distributed nearest-neighbor Gaussian processes. *Communications in Statistics - Simulation and Computation*, 0(0):1–13, 2021. doi: 10.1080/03610918.2021.1921798. URL `https://doi.org/10.1080/03610918.2021.1921798`.

Mengyang Gu and James O. Berger. Parallel partial Gaussian process emulation for computer models with massive output. *The Annals of Applied Statistics*, 10(3): 1317 – 1347, 2016. doi: 10.1214/16-AOAS934. URL `https://doi.org/10.1214/16-AOAS934`.

Rajarshi Guhaniyogi and Sudipto Banerjee. Meta-kriging: Scalable Bayesian modeling and inference for massive spatial datasets. *Technometrics*, 60(4):430–444, 2017. doi: 10.1080/00401706.2018.1437474. URL `https://doi.org/10.1080/00401706.2018.1437474`.

Rajarshi Guhaniyogi, Andrew O. Finley, Sudipto Banerjee, and Alan E. Gelfand. Adaptive Gaussian predictive process models for large spatial datasets. *Environmetrics*, 22:997–1007, 12 2011. doi: 10.1002/env.1131.

Rajarshi Guhaniyogi, Cheng Li, Terrance D. Savitsky, and Sanvesh Srivastava. A divide-and-conquer Bayesian approach to large-scale kriging. *ArXiv e-prints*, 2017.

Joseph Guinness and Montserrat Fuentes. Circulant embedding of approximate covariances for inference from Gaussian data on large lattices. *Journal of Computational and Graphical Statistics*, 26(3), 2016.

Matthew J. Heaton, William F. Christensen, and Maria A. Terres. Nonstationary Gaussian process models using spatial hierarchical clustering from finite differences. *Technometrics*, 59(1):93–101, 2017. doi: 10.1080/00401706.2015.1102763. URL https://doi.org/10.1080/00401706.2015.1102763.

Matthew J. Heaton, Abhirup Datta, Andrew O. Finley, Reinhard Furrer, Joseph Guinness, Rajarshi Guhaniyogi, Florian Gerber, Robert B. Gramacy, Dorit Hammerling, Matthias Katzfuss, Finn Lindgren, Douglas W. Nychka, Furong Sun, and Andrew Zammit-Mangion. A case study competition among methods for analyzing large spatial data. *Journal of Agricultural, Biological and Environmental Statistics*, 24(3):398–425, 2019. ISSN 1537-2693. doi: 10.1007/s13253-018-00348-w. URL https://doi.org/10.1007/s13253-018-00348-w.

David Higdon. A process-convolution approach to modelling temperatures in the North Atlantic Ocean. *Environmental and Ecological Statistics*, 5(2):173–190, 1998. ISSN 1573-3009. doi: 10.1023/A:1009666805688. URL https://doi.org/10.1023/A:1009666805688.

Matthias Katzfuss. Bayesian nonstationary spatial modeling for very large datasets. *Environmetrics*, 24(3):189–200, 2013. doi: https://doi.org/10.1002/env.2200. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/env.2200.

Matthias Katzfuss. A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, 112(517):201–214, Jan 2017. ISSN 1537-274X. doi: 10.1080/01621459.2015.1123632. URL http://dx.doi.org/10.1080/01621459.2015.1123632.

Matthias Katzfuss and Noel A. C. Cressie. Spatio-temporal smoothing and EM estimation for massive remote-sensing data sets. *Journal of Time Series Analysis*, 32: 430–446, 07 2011. doi: 10.1111/j.1467-9892.2011.00732.x.

Matthias Katzfuss and Dorit Hammerling. Parallel inference for massive distributed spatial data using low-rank models. *Statistics and Computing*, 27, 2014. doi: 10.1007/s11222-016-9627-4.

Cari G. Kaufman, Mark J. Schervish, and Douglas W. Nychka. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103(484):1545–1555, 2008. ISSN 01621459. URL `http://www.jstor.org/stable/27640203`.

Hannes Kazianka and Jürgen Pilz. Objective Bayesian analysis of spatial data with uncertain nugget and range parameters. *Canadian Journal of Statistics-revue Canadienne De Statistique*, 40:304–327, 2012.

Hyoung-Moon Kim, Bani K Mallick, and C. C Holmes. Analyzing nonstationary spatial data using piecewise Gaussian processes. *Journal of the American Statistical Association*, 100(470):653–668, 2005. doi: 10.1198/016214504000002014. URL `https://doi.org/10.1198/016214504000002014`.

Daniel Kirsner and Bruno Sansó. Multi-scale shotgun stochastic search for large spatial datasets. *Comput. Stat. Data Anal.*, 146:106931, 2020.

Ricardo T. Lemos and Bruno Sansó. A spatio-temporal model for mean, anomaly, and trend fields of North Atlantic sea surface temperature. *Journal of the American Statistical Association*, 104(485):5–18, 2009. doi: 10.1198/jasa.2009.0018. URL `https://doi.org/10.1198/jasa.2009.0018`.

Finn Lindgren, Håvard Rue, and Johan Lindström. An explicit link between Gaussian fields and Gaussian markov random fields: The stochastic partial differential

equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498, 2011. doi: https://doi.org/10.1111/j.1467-9868.2011.00777.x. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2011.00777.x.

NOAA. GOES-R Series Mission. *https://www.nesdis.noaa.gov/GOES-R-Mission*, 2016.

NOAA. Geostationary Satellite Server. *https://www.goes.noaa.gov/index.html*, 2018.

Douglas Nychka, Soutir Bandyopadhyay, Dorit Hammerling, Finn Lindgren, and Stephan Sain. A multiresolution Gaussian process model for the analysis of large spatial datasets. *Journal of Computational and Graphical Statistics*, 24(2):579–599, 2015. doi: 10.1080/10618600.2014.914946. URL https://doi.org/10.1080/10618600.2014.914946.

Christopher J. Paciorek and Mark J. Schervish. Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics*, 17(5):483–506, 2006. doi: https://doi.org/10.1002/env.785. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/env.785.

Hang Qian. Big data Bayesian linear regression and variable selection by normal-inverse-gamma summation. *Bayesian Analysis*, 13(4):1011–1035, 2018. doi: 10.1214/17-BA1083. URL https://doi.org/10.1214/17-BA1083.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL https://www.R-project.org/.

Paul D. Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417):108–119, 1992. ISSN 01621459. URL http://www.jstor.org/stable/2290458.

Martin Schlather, Alexander Malinowski, Peter J. Menck, Marco Oesting, and Kirstin Strokorb. Analysis, simulation and prediction of multivariate random fields with package RandomFields. *Journal of Statistical Software*, 63(8):1–25, 2015. URL https://www.jstatsoft.org/v63/i08/.

Martin Schlather, Alexander Malinowski, Marco Oesting, Daphne Boecker, Kirstin Strokorb, Sebastian Engelke, Johannes Martini, Felix Ballani, Olga Moreva, Jonas Auel, Peter J Menck, Sebastian Gross, Ulrike Ober, Paulo Ribeiro, Brian D Ripley, Richard Singleton, Ben Pfaff, and R Core Team. *RandomFields: Simulation and Analysis of Random Fields*, 2021. URL https://cran.r-project.org/package=RandomFields. R package version 3.3.10.

Alexandra M. Schmidt and Anthony O'Hagan. Bayesian inference for non-stationary spatial covariance structure via spatial deformations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(3):743–758, 2003. doi: https://doi.org/10.1111/1467-9868.00413. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00413.

Benjamin Shaby and David Ruppert. Tapered covariance: Bayesian estimation and asymptotics. *Journal of Computational and Graphical Statistics*, 21(2):433–452, 2012. doi: 10.1080/10618600.2012.680819. URL https://doi.org/10.1080/10618600.2012.680819.

Jonathan R. Stroud, Michael L. Stein, and Shaun Lysen. Bayesian and maximum likelihood estimation for Gaussian processes on an incomplete lattice. *Journal of Computational and Graphical Statistics*, 26(1):108–120, 2017. doi: 10.1080/10618600.2016.1152970. URL https://doi.org/10.1080/10618600.2016.1152970.

Maria A. Terres, Montserrat Fuentes, Dean L. Hesterberg, and Matthew L. Polizzotto.

Bayesian spectral modeling of microscale spatial distributions in a multivariate soil matrix. *ArXiv e-prints*, 2015.

Aldo V. Vecchia. Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):297–312, 1988. ISSN 00359246. URL `http://www.jstor.org/stable/2345768`.