Improving performances of MCMC for Nearest Neighbor Gaussian Process models with full data augmentation

Sébastien Coube-Sisqueille, Benoît Liquet

Version of Record: https://www.sciencedirect.com/science/article/pii/S0167947321002024
Manuscript_7f1ba3b60ebd69b6b0896dff3ab83099

Improving performances of MCMC for Nearest Neighbor Gaussian Process models with full data augmentation

Sébastien Coube-Sisqueille[1a], Benoît Liquet[2a,b]

[a] *Laboratoire de Mathématiques et de leurs Applications, Université de Pau et des Pays de l'Adour, UMR CNRS 5142, E2S-UPPA, Pau, France*
[b] *Department of Mathematics and Statistics, Macquarie University, Sydney*

Funding was provided by the Energy Environment Solutions (E2S-UPPA) consortium and the BIGCEES project from E2S-UPPA ("Big model and Big data in Computational Ecology and Environmental Sciences")

Abstract

Even though Nearest Neighbor Gaussian Processes (NNGP) alleviate MCMC implementation of Bayesian space-time models considerably, they do not solve the convergence problems caused by high model dimension. Frugal alternatives such as response or collapsed algorithms are one answer. An alternative approach is to keep full data augmentation, but to try and make it more efficient. Two strategies are presented.

The first is to pay particular attention to the seemingly trivial fixed effects of the model. Empirical exploration shows that re-centering the latent field on the intercept critically improves chain behavior. Theoretical elements support those observations. Besides the intercept, other fixed effects may have trouble mixing. This problem is addressed by interweaving, a simple method that requires no tuning, while remaining affordable thanks to the sparsity of NNGPs.

The second accelerates sampling of the random field using Chromatic samplers. This method boils long sequential simulation down to group-parallelized or group-vectorized sampling. The attractive possibility for parallelizing NNGP density can therefore be carried over to field sampling.

A R implementation of the two methods for Gaussian fields is freely avail-

[1]sebastien.coube@univ-pau.fr
[2]benoit.liquet@univ-pau.fr

able[1], an extensive vignette is provided. The presented implementation is run on two synthetic toy examples, along with the state of the art package spNNGP. Finally, the methods are applied to a real data set of lead contamination on the mainland of the United States of America.

Keywords: Nearest Neighbor Gaussian Process, Space-time models, Chromatic Sampler, Interweaving

1. Introduction

Many social or natural phenomena happen at the scale of a territory and must be observed at various sites and possibly times. The rise of modern GPS and Geographic Information Systems made large and high-quality point-referenced data sets increasingly available. Assume that, in a collection of sites $\mathcal{S}$ of the space or space-time domain $\mathcal{D}$, we have measurements $z(\cdot)$ with some kind of space or space-time coherence. This coherence can be accounted for by introducing a spatially-indexed process $w(\cdot)$ that has a well-defined joint distribution on any finite subset of the domain. We consider a Gaussian model where the observations $z(\cdot)$ have been disrupted by a Gaussian noise ϵ of standard deviation τ. Many models also add linear regression on covariates $X(\cdot)$, giving the following classical model formulation

$$z(s) = \beta_0 + X(s)\beta^T + w(s) + \epsilon(s), s \in \mathcal{S}. \tag{1}$$

In order to keep notations shorter, for any collection of spatial locations $\mathcal{P} \subset \mathcal{S}$, we denote the vector $\{w(s) : s \in \mathcal{P}\}$ as $w(\mathcal{P})$. Gaussian processes (GP) make an elegant prior distribution for $w(\cdot)$ for continuous data, see [1]. The GP prior distribution of $w(\mathcal{S})$ is $\mathcal{N}(\mu, \Sigma)$. The mean parameter of $w(\cdot)$ is usually fixed to $\mu = 0$ to avoid identification problems with the linear regression intercept β_0. The covariance matrix is computed using a positive definite function $k(\cdot)$ with covariance parameters θ, such as Matérn's covariance and its exponential and squared-exponential special cases. It can then be written as $\Sigma(\mathcal{S}, \theta)$, and its entries are $\Sigma(\mathcal{S}, \theta)_{i,j} = k(s_i, s_j, \theta)$. We denote $f(\cdot|\mu, \Sigma)$ the GP density, and we abbreviate it as $f(\cdot|\mu, \theta)$. The covariance parameters can have modeler-specified hyperpriors developed in [2, 3].

The weakness of GPs is that computing the GP prior density of $w(\mathcal{S})$ involves the determinant and inverse of $\Sigma(\mathcal{S}, \theta)$, incurring a computational

[1] https://github.com/SebastienCoube/Improving_NNGP_full_augmentation

cost that is cubic in the size of $\mathcal{S}$. Vecchia's approximation to Gaussian likelihoods has received increased attention in recent years, with the theoretical developments of [4, 5, 3, 6] and software presented in [7, 8]. The Nearest Neighbor Gaussian Process (NNGP) is a special case of Vecchia's approximation that provides a surrogate of the inverse Cholesky factor of Σ and uses it to approximate GP prior density. It starts by finding an ordering for the n locations of $\mathcal{S}$ which we will denote $(s_1, \ldots, s_n)$. The ordering may have an impact on the quality of the approximation, and is discussed in [3, 5]. The joint latent density of $w(s_1, \ldots, s_n)$ is then written under the recursive conditional form

$$f(w(s_1, \ldots, s_n)|\mu, \theta) = f(w(s_1)|\mu, \theta) \times \Pi_{i=2}^{n} f(w(s_i)|w(s_1, \ldots, s_{i-1}), \mu, \theta).$$

Since $f(w(s_1, \ldots, s_n)|\mu, \theta))$ is a Multi-Variate Normal (MVN) distribution function, the conditional density $f(w(s_i)|w(s_1, \ldots, s_{i-1}), \mu, \theta), i \in 2, \ldots, n$ is a Normal as well. A NNGP is obtained by replacing the vector $w(s_1, \ldots, s_{i-1})$ that conditions $w(s_i)$ by a much smaller parent subset denoted $w(pa(s_i))$ for each conditional density. The NNGP approximation to the GP prior joint density of $w(\cdot)$ is defined as

$$\tilde{f}(w(s_1, \ldots, s_n)|\mu, \theta) = f(w(s_1)|\mu, \theta) \times \Pi_{i=2}^{n} f(w(s_i)|w(pa(s_i)), \mu, \theta). \quad (2)$$

This very general principle can be applied to any kind of well-defined multivariate density. However, as far as we know, MVN density approximation is the only application. This may be explained by the fact that non-Gaussian data can be handled with GP modeling thanks to link functions. Moreover, a NNGP defines a MVN density and it is possible to compute the sparse Cholesky factor of the precision matrix explicitly and easily. The choice of the parents is critical, but no universal criterion exists. A popular choice is to choose the parent locations $pa(s_i)$ as s_i's nearest neighbors among $(s_1, \ldots, s_{i-1})$, explaining the denomination "Nearest Neighbors Gaussian Process" given in [3]. However, [3, 9] argue that mixing close and faraway observations can improve the approximation. This approximation is cheap and easily parallelisable. The latent density (2) can be split into small jobs and dispatched to a cluster of calculators ([3]). Its cost is linear in the number of observations, under the condition that the size of each parent set is bounded. More advanced strategies exist, such as grouping, proposed by Guinness in [5].

Although NNGPs work around the bottleneck of GP likelihood computation, they do not solve the problem of slow MCMC convergence. In [3], the

Gibbs sampler loops over θ, $w(\mathcal{S})$ and β, μ is fixed to 0. The latent field $w(\mathcal{S})$ is updated sequentially or by blocks. This sampler suffers from slow mixing, in particular when n increases. Other strategies have been proposed by Finley and al. [6] that avoid sampling the field in order to reduce the dimension of the model. Yet another method [6, 10] is to use convenient conjugate distributions for models where the range of $w(\cdot)$ and the variance ratio of $w(\cdot)$ and $\epsilon(\cdot)$ are fixed, and select the fixed parameters by cross-validation. Our approach is nevertheless to improve implementations of NNGP models where the latent field is explicitly sampled. Our first reason is that there may be situations where some of the methods presented in [6] perform poorly while full data augmentation works well. For example, the *collapsed NNGP* of [6] enjoys low dimensionality and nonetheless allows the latent field to be retrieved, but demands Cholesky factorization of large sparse matrices, which may not be feasible depending on n and the dimension of $\mathcal{D}$. The *Response NNGP* of [6] retrieves the covariance parameters θ but not the latent field $w(\mathcal{S})$. Our second reason is that efficient Gibbs sampler architectures can improve mixing sharply. A NNGP defines a Markov Random Field, allowing use of the blocking methods of [11]. The sparse Cholesky factor in a NNGP makes it possible to use the Ancillary-Sufficient Interweaving Strategy (ASIS) presented in [12]. The third reason is that full latent field sampling is all terrain, and can address many data models or be plugged into complex, non-stationary models like [13], while collapsed MCMC or conjugate models are much pickier.

Here is an outline of the article. Section 2 focuses on the seemingly trivial fixed effects of the hierarchical model. In 2.1 we propose a mild but efficient centering of the latent field on the least squares regression intercept. In 2.2, we extend centering to other fixed effects, and use interweaving from [12] to propose a robust, tuning-less application. Section 3 targets the simulation of the random field. In 3.1, we propose to use the chromatic samplers developed by Gonzalez and al. in [14] in order to carry the attractive parallelizability of NNGP density over to field sampling. In 3.2, we analyze the sensitivity of NNGP graph coloring and we benchmark coloring algorithms. We apply our methods in section 4. We present our implementation (available at `https://github.com/SebastienCoube/Improving_NNGP_full_augmentation`) in 4.1. We test our implementation along with the state of the art package spNNGP presented in [8] on synthetic toy examples in 4.2. In 4.3, we present an application on lead contamination on the mainland of the United States of America. The article ends by a discussion in Section 5.

2. Latent field centering

2.1. Centering the latent field on the intercept

The mean parameter μ of the prior density for the latent field $w(\cdot)$ is usually set to 0 in order to avoid identification problems with the intercept β_0. We call this formulation standard, since it is found in state of the art papers such as [3, 6]. We name samples of the standard formulation $w_s(\cdot)$. Our proposal is to replace $w_s(\mathcal{S})$ by a centered $w_c(\mathcal{S}) = w_s(\mathcal{S}) + \beta_0$ in the Gibbs Sampler. This substitution is a non-degenerate linear transform that keeps the model valid, while keeping the possibility of transforming the samples back to standard parametrization if needed. The centered parametrization can also be seen as a slightly different model, with (1) becoming

$$z(s) = X(s)\beta^T + w_c(s) + \epsilon(s), s \in \mathcal{S} \tag{3}$$

and the prior density of $w_c(\mathcal{S})$ becoming

$$\tilde{f}(w_c(\mathcal{S})|\mu = \beta_0, \theta).$$

These changes impact the full conditional distributions. Table 1 summarizes the changes in a Gibbs sampler for a Gaussian model found in [3]. We denote $f(\cdot|\cdot,\cdot)$ the normal density function, and $\tilde{Q}$ the latent field's precision matrix defined by the NNGP. We abbreviate the interest variables $X(\mathcal{S})$ as X. We denote the vector made of n times 1 as $\mathbf{1}$. The matrix obtained by adding $\mathbf{1}$ to the left side of X is named $[\mathbf{1}|X]$. We did not feature prior distributions on the high-level parameters like θ, τ or β: their full conditionals would not be affected, since centering changes only the NNGP prior and the observed data likelihood.

A point that may confuse a careful reader is the presence of an Ordinary Least Square (OLS) distribution in rows β and (β_0, β) of table 1, and later, in step 2 of algorithm 1: why is there an OLS if there is some spatial auto-correlation? The point is that this auto-correlation is captured by $w(\mathcal{S})$, and that conditionally on $w(\mathcal{S})$ the observations are independent. Remember that in the standard case the likelihood of the Gaussian observations is

$$f(z|\beta_0, \beta, w_s, \theta, \tau) \propto \mathcal{N}(w_s + \beta_0 + X\beta^T, \tau^2 I_n).$$

This likelihood is then plugged into the full conditional of β, explaining the OLS:

$$f(\beta_0, \beta|z, w_s, \theta, \tau) \propto \underbrace{f(z|\beta_0, \beta, w_s, \theta, \tau)}_{\text{independent Gaussian}} \underbrace{f(w_s|\beta_0, \beta, \theta, \tau)}_{\text{constant w.r.t } \beta_0, \beta} \underbrace{f(\beta_0, \beta, \theta, \tau)}_{\text{prior}}.$$

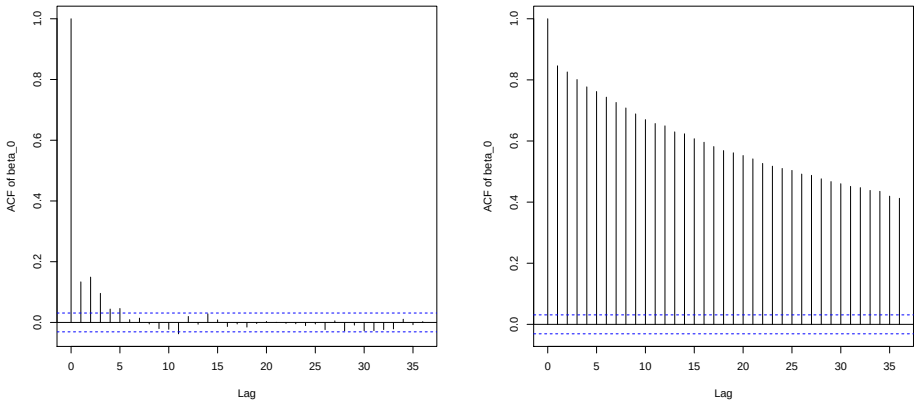
Table 1: Changes in the full conditional distributions

Variable	Standard	Centered
β_0		$f(\beta_0, (\mathbf{1}^T\tilde{Q}\mathbf{1})^{-1}(\mathbf{1}^T\tilde{Q}w_c), (\mathbf{1}^T\tilde{Q}\mathbf{1})^{-1})$
β		$f(\beta, (X^TX)^{-1}(X^T(z-w_c))), \tau^2(X^TX)^{-1})$
(β_0, β)	$f(\beta, ([\mathbf{1}\|X]^T[\mathbf{1}\|X])^{-1}([\mathbf{1}\|X]^T(z-w_s))),$ $\tau^2([\mathbf{1}\|X]^T[\mathbf{1}\|X])^{-1})$	
θ	$\tilde{f}(w_s(\mathcal{S})\|0, \theta)$	$\tilde{f}(w_c(\mathcal{S})\|\beta_0, \theta)$
τ	$\Pi_{s\in\mathcal{S}} f(z(s)\|w_s(s) + \beta_0 + X(s)\beta^T, \tau)$	$\Pi_{s\in\mathcal{S}} f(z(s)\|w_c(s) + X\beta^T, \tau)$
$w(x)$	$\tilde{f}(w_s(x)\|w_s(\mathcal{S}\backslash x), 0, \theta)$ $f(z(x)\|w_s(x) + \beta_0 + X(x)\beta^T, \tau)$	$\tilde{f}(w_c(x)\|w_c(\mathcal{S}\backslash x), \beta_0, \theta)$ $f(z(x)\|w_c(x) + X(x)\beta^T, \tau)$

The reasoning is similar at row β for the centered model, the only difference being that β_0 is excluded. Note that this OLS step is perfectly affordable whatever the prior distribution of the latent field. On the other hand, the centered update of β_0 implies multiplications with $\tilde{Q}$, and its efficiency is based on the sparsity of this matrix.

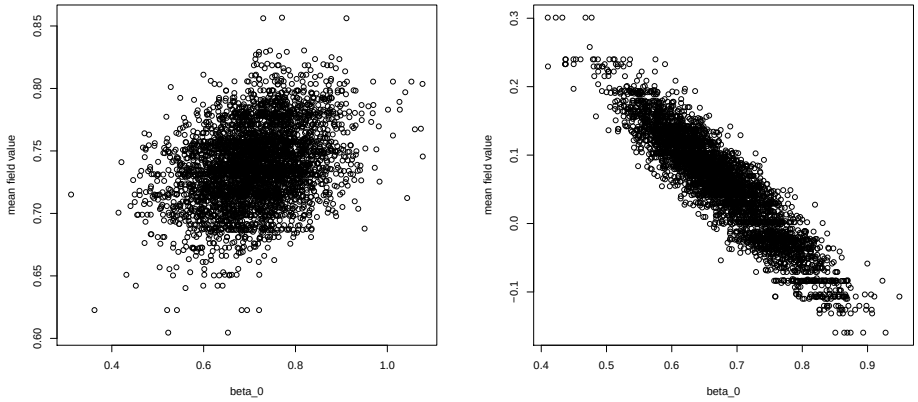
Even if the modification is minor, the improvement in the mixing of the intercept is clear. We simulated a small toy example with 1000 observations and ran the two Gibbs samplers. The autocorrelation plots (Figure 1) are clearly in favor of the centered formulation. In this toy example the standard model mixes after a few hundred iterations, but this is not the case for larger models. We observed empirically that there is much more correlation between $w(\mathcal{S})$ and β_0 in the standard implementation. Plotting $\frac{1}{n}\Sigma w(\mathcal{S})$ against β_0 (Figure 2) displays a clear ridge in the case of the standard model (Figure 2b). This means that the whole latent field has to shift upwards and downwards for the intercept to explore its posterior distribution, causing a slow exploration. Ridge-like densities are a well known plague of Gibbs samplers, and linear recombination is one of the tools to get rid of it, see [15].

The behavior of the toy example arises from the fact that the fraction of β_0^t that is carried over in w^{t+1} and β_0^{t+1} changes following the model. Take a simpler spatial model where only an intercept and the Gaussian latent field are estimated, while the Gaussian noise variance τ^2 and NNGP precision $\tilde{Q}$ are known. The intercept coefficient has an improper constant prior. Assume that the latent field is sampled in one step (which is usually not the case unless the data size is very small).
Denote the diagonalization $\tilde{Q} = V^T\lambda V$, V being a square matrix of eigen-



(a) Centered model (b) Standard model

Figure 1: The ACF of β_0 drops much faster in the centered model than in the standard model



(a) Centered model (b) Standard model

Figure 2: When plotting $\frac{1}{n}\Sigma w(\mathcal{S})$ against β_0, the standard model exhibits a ridge-shaped point cloud

vectors and λ being a diagonal matrix of eigenvalues. The eigenvalues are positive since $\tilde{Q} = \tilde{R}^T\tilde{R}$. Note α_i, $i = 1, \dots, n$ the coordinates of the vector $\mathbf{1} = (1, \dots, 1)$ in the orthonormal basis V. The following results are proved in Appendix A.
The first result is that a fraction of β_0 is carried over in the empirical mean of the latent field. Note $\rho = \Sigma_{i=1}^n \alpha_i^2 (\tau^2\lambda + I_n)_{i,i}^{-1}/n$. Then,

$$\overline{w_s^{t+1}} = \mu_s - \rho\beta_0^t + \epsilon_s^{t+1} \quad \text{and} \quad \overline{w_c^{t+1}} = \mu_c + (1-\rho)\beta_0^t + \epsilon_c^{t+1},$$

μ_s and μ_c being fixed, ϵ_s and ϵ_c being stochastic innovations, and t being the index of the iteration. Moreover, we have

$$0 \leq \rho \leq 1.$$

Thanks to this result, we can see that if a high fraction of β_0 is carried over in the mean of the standard latent field, then a low fraction of β_0 will be carried over in the mean of the centered latent field, and conversely. This is what can be seen clearly in figure 2.

The next question is why ρ is closer to 1 than to 0. This point is difficult to clarify because there is no analytic expression for terms where $\tilde{Q}$ is involved. For example, the range parameters are updated through a Metropolis step in [3] because a full conditional draw is challenging. However, we can start from $\rho = \Sigma_{i=1}^n \alpha_i^2 (\tau^2\lambda + I_n)_{i,i}^{-1}/n$ and make a deduction: if the sum is high, then $\lambda_{i,i}$ is small when α_i is big; and conversely $\lambda_{i,i}$ is big when α_i is small. We can re-formulate : $\lambda_{i,i}^{-1}$ is big when α_i is big. Now, remark that V and λ^{-1} respectively are the spatial basis and coefficients of the Karhunen-Loève decomposition of the NNGP prior. This means that α_i is high for the first components of the decomposition, where λ^{-1} is the highest. In other terms, $\mathbf{1}$ "resonates" with the first spatial basis functions of the Karhunen-Loève decomposition. This conclusion is consistent with the fact that $\tilde{Q}$ parametrizes a spatially coherent latent field, inducing that a few spatial basis functions are enough to describe most of w. For example, in the Predictive Process model of [16], w is approximated by a degenerate process with a low-rank covariance matrix.

Now, let's focus on the expressions of $[\beta_0^{t+1}|\beta_0^t]$. Like the mean of the latent field, they can be expressed with a fixed part, a geometric carry-over, and an innovation. In the standard model, a fraction ρ of β_0^t is carried over. We already discussed this quantity. As for the centered model, the fraction

of β_0^t which is conserved in β^{t+1} is

$$\Sigma_{i=1}^n \left((\alpha_i^2\lambda_{i,i})(\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1)\right)/\Sigma_{i=1}^n\alpha_i^2\lambda_{i,i}.$$

Once again, the geometric term is between 0 and 1, since $0 \leq (\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1) \leq 1$. Like before, suppose that α is big when λ is small. Then, when α_i is the largest, $(\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1)$ will be much smaller than 1, resulting in $\Sigma_{i=1}^n\left((\alpha_i^2\lambda_{i,i})(\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1)\right)$ being much smaller than $\Sigma_{i=1}^n\alpha_i^2\lambda_{i,i}$. Therefore, we can expect a small proportion of β_0^t to remain in β_0^{t+1}.

2.2. Adaptation to other fixed effects

Field centering can be extended to other fixed effects. In most cases it is unnecessary because centering and scaling $X(\mathcal{S})$ is enough to considerably improve chain behavior. What is more, centering the latent field on other linear effects usually worsens the MCMC behavior of the associated regression coefficients. However, this general rule is not always working. In particular, when some covariates have some spatial coherence, their regression coefficients may have trouble mixing. In this case, it is often useful to try and center $w(\cdot)$ not only on the intercept, but also on the troublesome covariates' fixed effect. However, doing preliminary runs and picking manually which fixed effects the field needs to be centered on would be tedious.

Interweaving, introduced by Yu and Meng [12], combines the advantages of the two strategies and removes the need to choose. The method takes advantage of the discordance between two parametrizations to construct the following step :

$$[Y_1|\theta^t] \to [Y_2|Y_1] \to [\theta^{t+1}|Y_2],$$

Y_1 and Y_2 being two data augmentations and θ the parameter. Usually, it is complicated to sample $[Y_2|Y_1]$ directly. Drawing an intermediary $\theta^{t+0.5}$ gives

$$[Y_1|\theta^t] \to [\theta^{t+0.5}|Y_1] \to [Y_2|\theta^{t+0.5}, Y_1] \to [\theta^{t+1}|Y_2].$$

It is possible that $[Y_2|\theta, Y_1]$ is a deterministic transformation, giving a degenerate joint distribution. Note that interweaving is not alternating: an alternating scheme would be $[Y_1|\theta^t] \to [\theta^{t+0.5}|Y_1] \to [Y_2|\theta^{t+0.5}] \to [\theta^{t+1}|Y_2]$. The strategy is usually very efficient if the two parametrizations are an ancillary-sufficient couple, giving an Ancillary-Sufficient Interweaving Strategy (ASIS), and can even be efficient when neither of the two augmentations performs well when implemented separately.

Algorithm 1 presents the steps to update the regression coefficients with interweaving. The two parameterizations are w, which is un-centered, and v, which is centered on all the fixed effects. For the sake of simplicity, we suppose that there is only one measurement of X per spatial location and we use an improper constant joint hyperprior on (β_0, β). The parameters that depend on the state in the Gibbs sampler are indexed by t. If the observations were not Gaussian, step 4 would be left unchanged, while step 2 would be adapted just like in any generalized NNGP model [3].

There are two limitations to this approach. The first is the case where several measurements of the interest variable $z(\cdot)$ and the regressors $X(\cdot)$ are done at the same spatial location. The model must be extended as

$$z(s,i) = X(s,i)\beta^T + w(s) + \epsilon(s,i), s \in \mathcal{S}, 1 \leq i \leq m(s),$$

$m(s) \geq 1$ being the number of observations in the site s. In this setting, some variables vary within one spatial location, while others do not. For example, the presence of asbestos in buildings may be considered as a location-wise regressor, while smoking is an observation-wise regressor. If the regressors vary within one location, it is impossible to center the field on the corresponding fixed effects. This would mean that the normal random variable $w(s)$ has several mean parameters at the same time. However, it is still possible to restrict interweaving to the regression coefficients associated to the location-wise variables. Our implementation allows us to specify which regressors are associated with spatial location and which are associated with individual measurements. As a NNGP is a point-measurement model, regressors obtained through gridded and areal data are immediately eligible for this method.

The second limitation is the computational cost. With an improper constant prior, the centered regression coefficients follow a MVN distribution whose mean and variance need to be computed at each update of θ. The sparsity induced by Vecchia's approximation is critical for the feasibility of the method, because it ensures that matrix multiplications involving $\tilde{Q}$ are affordable. Using a sparse matrix formulation for X could further alleviate this operation if X has dummy variables or null measurements.

Algorithm 1 Regression coefficient updating with interweaving

1: **input** $\tilde{Q}^t, w^t, X, \beta^t, \beta_0^t, \tau^t$
2: **simulate** $\beta_0^{t+.5}, \beta^{t+.5}$ following $\mathcal{N}(([\mathbf{1}|X]^T[\mathbf{1}|X])^{-1}([\mathbf{1}|X]^T z), \tau^2([\mathbf{1}|X]^T[\mathbf{1}|X])^{-1})$
3: $v = w^t + X(\beta^{t+.5})^T$
4: **simulate** $\beta_0^{t+1}, \beta^{t+1}$ following $\mathcal{N}(([\mathbf{1}|X]^T\tilde{Q}^t[\mathbf{1}|X])^{-1}([\mathbf{1}|X]^T\tilde{Q}^t v), ([\mathbf{1}|X]^T\tilde{Q}^t[\mathbf{1}|X])^{-1})$
5: $w^{t+0.5} = v - X(\beta^{t+1})^T$
6: **return** $\beta_0^{t+1}\beta^{t+1}, w^{t+0.5}$

3. Chromatic sampler for Nearest Neighbor Gaussian Process

3.1. Chromatic samplers and how to apply them to NNGP

In a Gibbs sampler, the parameters of a model are updated sequentially. If a set of variables happens to be mutually independent conditionally on the other variables of the model, and are updated consecutively by the Gibbs sampler, their sampling can be parallelized. Let's consider a Gibbs sampler or a Metropolis-Within-Gibbs aiming to sample from a joint multivariate distribution $f(x_1, \ldots, x_n)$.

$$\begin{aligned} x_1^{t+1} &\sim f(x_1|x_2^t, \ldots, x_n^t) \\ &\ldots \\ x_i^{t+1} &\sim f(x_i|x_1^{t+1}, \ldots, x_{i-1}^{t+1}, x_{i+1}^t, \ldots, x_n^t) \\ &\ldots \\ x_n^{t+1} &\sim f(x_n|x_1^{t+1}, \ldots, x_{n-1}^{t+1}). \end{aligned}$$

Let's introduce $p \leq n$ vectors $X_1, \ldots, X_p$ so that $(x_1, \ldots, x_n) = (X_1, \ldots, X_p)$, and suppose that $\forall X \in X_1, \ldots, X_p$, either X has only one element or the elements of X are conditionally independent given the other variables. The Gibbs sampler can then be re-written

$$\begin{aligned} x_i^{t+1} \in X_1 &\sim f(x_i|X_2^t, \ldots, X_p^t) \\ &\ldots \\ x_i^{t+1} \in X_i &\sim f(x_i|X_1^{t+1}, \ldots, X_{i-1}^{t+1}, X_{i+1}^t, \ldots, X_p^t) \\ &\ldots \\ x_i^{t+1} \in X_p &\sim f(x_i|X_1^{t+1}, \ldots, X_{p-1}^{t+1}). \end{aligned}$$

Since all elements from X_i are simulated from independent densities, it is possible to parallelize their sampling.

A NNGP is defined on a Directed Acyclic Graph (DAG) by Datta and al. [3], see [4] for discussion about other Vecchia's approximations. Then,

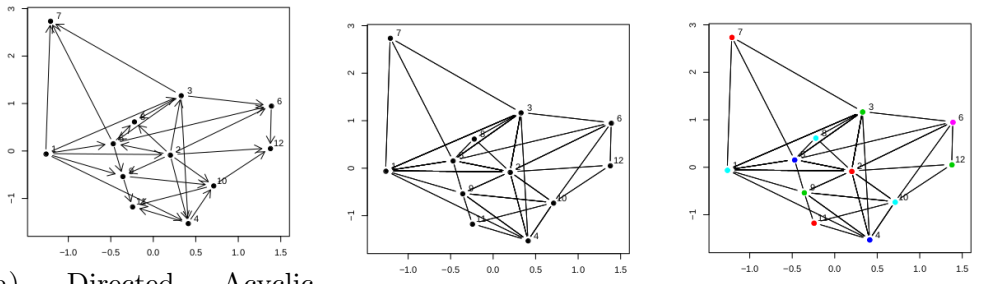
using the argument of recursive kernel factorization given in [17], it has the Markov properties on the moral graph obtained by un-directing the edges and "marrying" the parents in the DAG (Figure 3). Graph vertex coloring associates one color with each node of a graph while forbidding that two connected nodes have the same color, just like coloring a map while forbidding two countries that share a border having the same color. Using the Global Markov property inductively, it is possible to guarantee mutual conditional independence for the variables or the blocks that correspond to vertices sharing the same color. Chromatic sampling can be applied straightforwardly to the Gibbs sampler presented in [3]. It also allows normalizing constants to be computed, and can be combined with the covariance parameter blocking proposed by Knorr-Held and Rue in [11].

Chromatic samplers can be applied to blocked sampling as well. This method consists in updating the latent field in various locations at once. Chromatic sampling is a special case of blocked sampling, because in general there is no conditional independence within one block. Precisely, sampling the latent field jointly in a region of the domain reduces the negative impact of spatial auto-correlation on the behavior of MCMC chains. Blocked sampling may be applied to the latent field alone [3] or improve both covariance parameters and field sampling in [11]. Even though there is no conditional independence within one block, there is some conditional independence between the blocks, as long as there is no edge between any pair of their respective vertices, allowing for chromatic sampling. The matrix $B^T AB$ indicates the connections between the blocks, A being the adjacency matrix of the NNGP latent field's Markov graph, and B a vertex-block indicator matrix ($B_{i,j} = 1$ if vertex i belongs to block j).

3.2. Coloring of NNGP moral graphs: sensitivity analysis and benchmark of the algorithms

Coloring the moral graph $\mathcal{G}^m$ is a critical step in chromatic sampling and determines the attractiveness of the method with respect to the "vanilla" versions of the algorithms (one-site sequential sampling or blocked sampling with several blocks). We focus on two variables to summarize the efficiency of chromatic sampling:

- The number of colors: the smaller this number, the fewer the number of steps in the chromatic sampler.



(a) Directed Acyclic Graph (b) Moral Graph (c) Colored Moral Graph

Figure 3: Moralization and coloring of a DAG

- The time needed for coloring, which must be small with respect to the running time of the MCMC chains.

This section has two objectives. The first is to test the sensitivity of those two interest variables to the properties of $\mathcal{G}^m$ and the coloring algorithm using variance-based sensitivity analysis. The second objective is to benchmark various coloring algorithms and find a rule to choose the algorithm.
We test various factors that may change the structure of $\mathcal{G}^m$:

- Size n.
- Number of parents in the DAG m.
- Spatial domain dimension d.
- Ordering of the points.

We also test 3 coloring algorithms, given in detail in Appendix B.1:

- Naive greedy coloring: coloring each vertex with the smallest available color.
- Degree greedy coloring: reordering the vertices following their number of neighbors, and applying naive greedy coloring.
- DSATUR heuristic: coloring the node that has the highest number of distinct colors among its neighbors (*Degree of SATURation*), and breaking ties using the number of neighbors.

The full results of the experiments are given in Appendix B.2, and the sensitivity analyses are summarized in table 2.

3.2.1. Pilot experiment

Design

The objective is to perform preliminary sensitivity analysis and benchmark on small graphs. We test the three coloring algorithms and graphs with the following attributes, with each case being replicated 10 times:

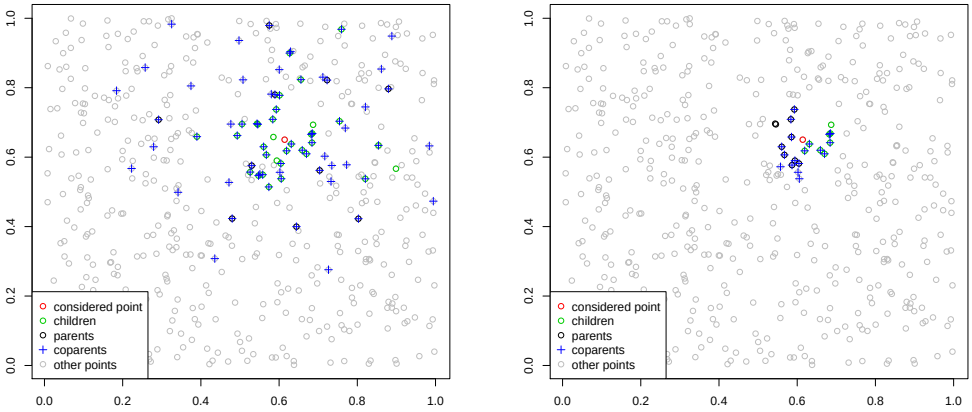
- Graph size $n = 500, 1000, 2000$.
- Number of parents $m = 5, 10, 20$.
- Dimension $d = 2, 3$.
- Ordering following the first coordinate from [3], at random, or using MaxMin heuristic from [5].

Sensitivity

The color count is overwhelmingly driven by the number of parents, the ordering, and interactions between them. The role of the parents is not surprising: the larger the parent sets, the more edges in the graph, the more colors are needed. As for the ordering, it does not change the density but rather the distribution of the edges, which may explain why the number of colors is much smaller in the coordinate ordering. In a graph obtained through coordinate ordering and the Nearest Neighbor heuristic, a vertex tends to be connected with its immediate spatial surroundings. Its parents in the DAG will be its predecessors along the coordinate used for ordering, its children will be its successors, and its co-parents will mostly be a mix of the closest parents and children. In a graph obtained thanks to random or max-min ordering, the connections can be much longer, in particular for points coming early in the ordering. This results in some vertices being connected to many other vertices, leading to a denser graph. This point is illustrated in figures 4 and 5.

The number of colors is robust with respect to the graph size because n and its interactions have very low percentages in table 2. Since the numbers of colors are small with respect to n (45 colors at the most), this makes chromatic sampling a good candidate for large data sets.

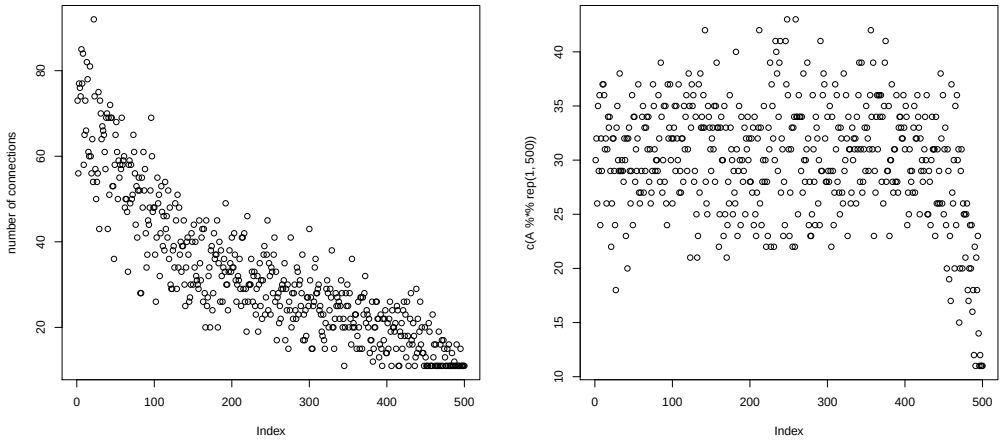
This point is counter-intuitive because a bigger graph is more complex than a smaller graph and should therefore be more difficult to color. The Markovian nature of NNGPs may be a lead to explain this point, because several sets of vertices that are not linked by a direct edge can be colored



(a) Max-min ordering
(the considered point is the 30^{th} of 500)

(b) Coordinate ordering

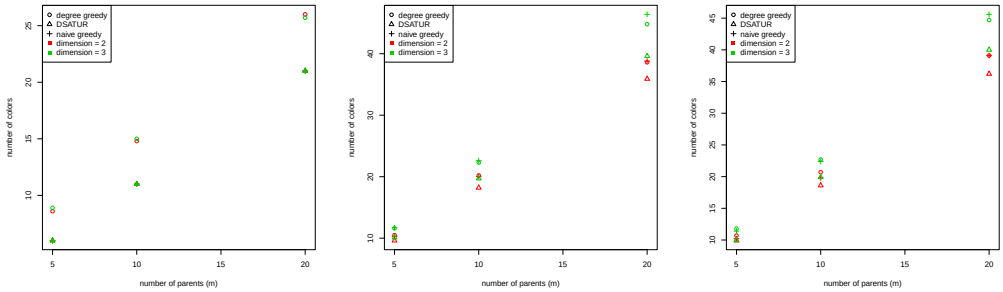
Figure 4: Connections of the same point, with two different orderings.



(a) Max-min ordering

(b) Coordinate ordering

Figure 5: Number of connections of a point given its place in the ordering (same graphs as in figure 4)



(a) Coordinate ordering (b) Max-min ordering (c) Naive

Figure 6: Impact of the spatial domain dimension and the coloring algorithm on the mean number of colors, for graphs of size $n = 2000$.

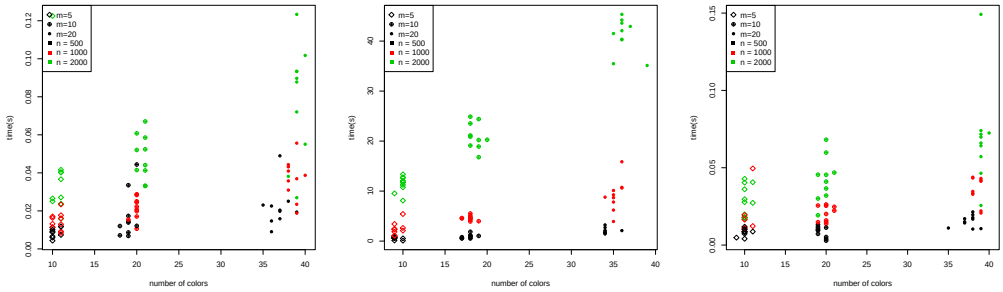
independently. While new vertices are added to the graph, older vertices can be forgotten and their colors can be re-used.

The dimension of the spatial domain d plays almost no role in the sensitivity analysis, and the choice of the coloring algorithm has a very marginal effect on the color count. Closer examination of the means reveals nonetheless that their effect is not nonexistent, but rather dwarfed by the prominent role of the ordering of the vertices and the number of parents. For graphs obtained with max-min or random ordering (6b and 6c), the number of colors increases if $d = 3$.

The running time is affected by n, as expected. However, it is mostly explained by the coloring algorithm and its interactions with n and m. In Figure 7, we see the results of the experiment when the ordering of the spatial points is random and $d = 2$. The number of parents m defines well-separated vertical clouds of points, showing a clear, positive impact on the number of colors. It also increases the running time: the clouds of points on the right are stretched higher along the ordinates axis. The graph size n affects the running time positively. The other cases with different ordering and dimension all show this clear, chromatography-like profile.

Benchmark

In order to see if one coloring algorithm has the better of the others, we compare the average number of colors for each case of the experiment in table B.5. Regardless of the ordering, m, and n, the number of colors favors DSATUR systematically but slightly over the two simpler algorithms. In the case of coordinate ordering, naive greedy coloring reaches the performances



(a) Degree (b) DSATUR (c) Naive

Figure 7: Repartition of the number of colors and running time with random ordering and d=2.

of DSATUR. While the two simple methods are very economical, the running time becomes high in DSATUR when the graph size augments (Figure 7b). We conclude that regardless of the structure of the graph, DSATUR must be chosen for smaller graphs. The two other methods must be chosen for larger graphs because DSATUR will become prohibitively expensive.

3.2.2. Coloring for large graphs

Design

The objective is to test the sensitivity of the two interest variables and to benchmark coloring algorithms when the graphs are bigger. The experiment is the same as before, with two differences:

- Only naive greedy and degree greedy coloring algorithms are tested
- The graph size $n = 50000, 100000, 200000$

Each case is replicated 10 times.

Sensitivity

For the number of colors, the results are the same as before (table 2). It is mostly determined by the ordering and the number of parents. The robustness of the number of colors with respect to n is confirmed. The running time is affected mostly by n, but the ordering and m also play a role.

Benchmark

In table B.6, we can see that naive coloring systematically has a lower mean number of colors than degree coloring. It is also slightly faster due to the

Table 2: Sensitivity analysis*

	Pilot		Large		Blocked	
	colors	time	colors	time	colors	time
ordering	15.4	3.4	10.6	8.7	40.7	1.7
algo	0.8	24.7	0.5	1.3	1.0	8.6
d	0.6	0.0	1.1	0.7	0.4	0.0
m	76.4	2.8	80.6	26.0	17.2	1.0
n/n blocks	0.2	11.8	0.1	40.7	15.2	16.2
ordering:algo	0.3	6.8	0.1	0.3	0.3	3.3
ordering:d	0.3	0.0	0.5	0.4	0.2	0.0
ordering:m	5.3	0.8	5.2	5.4	7.9	0.5
ordering:n	0.0	3.3	0.0	3.0	6.4	6.1
algo:d	0.0	0.1	0.0	0.0	0.0	0.1
algo:m	0.2	5.5	0.2	0.6	0.1	1.9
algo:n	0.0	23.3	0.0	0.7	0.9	31.3
d:m	0.2	0.0	0.4	0.5	0.0	0.0
d:n	0.0	0.0	0.0	0.2	0.1	0.1
m:n	0.0	2.4	0.0	8.0	6.3	3.5
total	99.7	84.8	99.5	96.6	96.6	74.6

* Read: "In the pilot experiment, the ordering of the spatial points explained 15.4 percents of the variance of the number of colors"

fact that the vertices are not sorted. Anyway, the running times are short in both cases and are never bigger than 15 seconds. We conclude that naive greedy coloring is the better option for large data sets.

3.2.3. Coloring blocked graphs

Design

The objective is to carry out sensitivity analysis and benchmark to graphs that correspond to spatial blocks used for a block-update of the latent field. Spatial clusters of vertices are found using a K-means algorithm on $n = 10000$ spatial locations, and coloring is applied to the Markov graph between the blocks. The orderings, numbers of parents, and dimensions remain the same as in the previous experiments. The parameters that change are:

- The graph size $n_{blocks} = 10, 20, 50, 100, 500$.

- All three algorithms (DSATUR, naive greedy, and degree greedy) are tested

Each case is replicated 10 times.

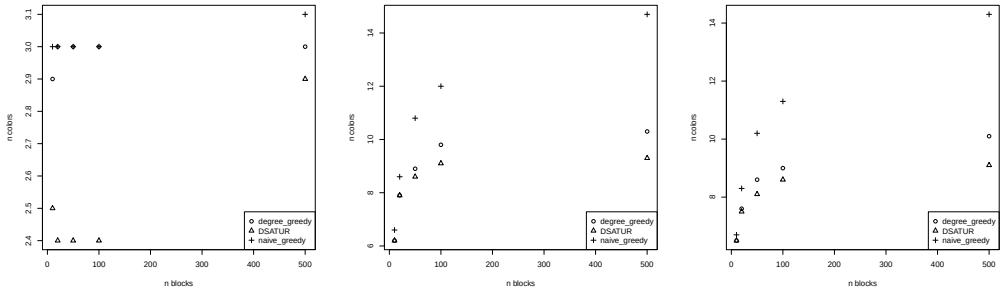
Sensitivity

The sensitivity of the number of colors (table 2) differs from the previous experiments. Even though m still matters, it is the ordering that becomes the most important variable.

This loss of importance of m can be explained by the fact that one edge is enough to connect two blocks, and once two blocks are connected adding new edges between them is redundant. On the other hand, the disposition of the edges in the space, which is induced by the ordering, keeps all its importance. Short edges induced by a coordinate ordering (4b) will connect adjacent spatial blocks, while the long edges induced by the max-min heuristic (4a) will connect distant regions. The important interaction between m and the ordering is well explained by this hypothesis. In table B.7 we see that m barely plays any role for coordinate ordering, while it keeps having an important impact for the other two orderings. Indeed, when $\mathcal{S}$ is ordered following a coordinate, adding more short connections between contiguous spatial blocks does not change anything: those blocks are already connected. For max-min and random ordering, though, adding long edges may link distant regions that were not connected yet. After m and the ordering, the number of blocks is the third most important variable. As expected, the more blocks there are in the graph, the more colors are needed. However, we remark that Max-Min and Random orderings perform poorly for graphs with few blocks, and actually need almost one color per block. Once the graphs get bigger, the number of colors stabilizes. Therefore, the observed sensitivity with respect to the number of blocks is mostly induced by the bad coloring of graphs with few blocks. The point can be visualized in figure 8 for $m = 5$ and $d = 2$.

Benchmark

Incontestably, DSATUR has the smallest number of colors, as seen in figure 8 and table B.7. Interestingly, degree greedy coloring has the second smallest number of colors. If we assume that the number of blocks will always be smaller than 1000, we can discard the running time from our criteria and say that DSATUR is the best option for blocked graphs. However, in the cases with random and Max-Min orderings and low numbers of blocks, chromatic sampling does not greatly reduce the number of steps with respect to vanilla



(a) Coordinate ordering (b) Max-min ordering (c) Naive

Figure 8: Number of colors following the number of blocks, for spatial domain dimension $d = 2$ and number of parents $m = 5$.

block sampling.

4. Implementation, testing and application

4.1. About our implementation

We tested our implementation with the state of the art package spNNGP presented in [8], that uses the Gibbs sampler architecture given in [3]. spNNGP uses Rcpp [18] and parallelizes the computation of NNGP density. In order to monitor convergence using the diagnostics from [19, 20], various chains need to be run one after the other. Our implementation is available at https://github.com/SebastienCoube/Improving_NNGP_full_augmentation. The code is done in R (see [21]), with the ASIS Gibbs sampler architecture of [12]. Chromatic sampling is implemented for individual locations. We used the package GpGp ([7]) for Vecchia's approximation factor computation. Our implementation runs several chains in parallel thanks to the package parallel (see [22]), but GpGp does not implement parallel Vecchia's approximation factor computation within each chain like spNNGP. We emphasized the ease of use, with real-time Gelman-Rubin diagnostics and chain plotting, greedy MCMC tuning in the first hundred iterations, and the possibility to start, stop, and run again easily. For some data sets, our implementation has an advantage over spNNGP because multiple measurements at the same spatial site are allowed. However, unlike spNNGP, we have only implemented a Gaussian model so far.

4.2. Toy examples

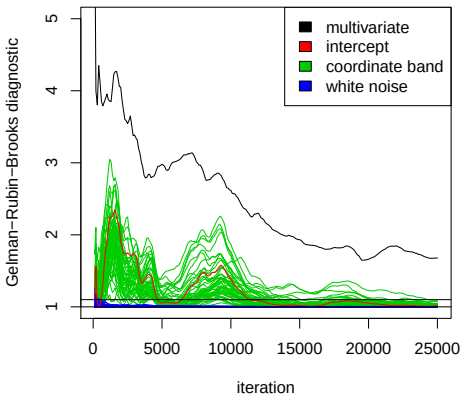
We present two toy examples in order to test our implementation, with the latent field NNGP implementation of spNNGP as a reference. For both implementations, 5 nearest neighbors were used for NNGP. The toy examples are Gaussian. We compare the MCMC behavior using the number of iterations and the time needed before the chains have mixed following the Gelman-Rubin-Brooks $\hat{R}$. We also compare the estimated covariance parameters with the values that were used to simulate the toy example. The covariance parameters are reported individually, and in the second toy example we report the Mean Square Error (MSE) of the fitted fixed effects with respect to their true value. Eventually, we compare the quality of the denoising using the MSE of the denoised field predicted by the model with respect to the simulated latent field. The first toy example is a simple Gaussian field simulated as follows.

1. Simulate spatial locations $\mathcal{S} \sim \mathcal{U}([0, 50] \times [0, 50])$
2. Simulate latent field $w(\mathcal{S}) \sim \mathcal{N}(0, \Sigma(\mathcal{S})), \Sigma(\mathcal{S})_{i,j} = exp(-0.5\|s_i, s_j\|)$
3. Simulate observed variable $z(\mathcal{S}) = w(\mathcal{S}) + \epsilon(\mathcal{S}), \epsilon(\mathcal{S}) \sim \mathcal{N}(0, 5I_n)$

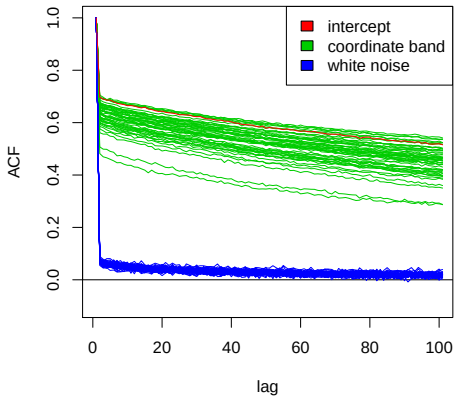
The second toy example intends to highlight the positive effect of our architecture when covariates have some spatial coherence. We integrate covariates that are areal indicators, and others that are white noise.

1. Simulate spatial locations $\mathcal{S} \sim \mathcal{U}([0, 50] \times [0, 50])$ and note $\mathcal{S}_1$ the first coordinates of the locations
2. Simulate latent field $w(\mathcal{S}) \sim \mathcal{N}(0, \Sigma(\mathcal{S})), \Sigma(\mathcal{S})_{i,j} = exp(-0.5\|s_i - s_j\|)$
3. Simulate regressors $X = [X_1|X_2]$ with $X_1 = [\mathbb{1}_{1 \leq \mathcal{S}_1 < 2}|\mathbb{1}_{2 \leq \mathcal{S}_1 < 3} \dots |\mathbb{1}_{49 \leq \mathcal{S}_1 \leq 50}]$ and X_2 a matrix of side $n \times 49$ with coefficients drawn following independent $\mathcal{N}(0, 1)$
4. Simulate regression coefficients $\beta \sim \mathcal{N}(0, I_{98})$
5. Simulate observed variable $z(\mathcal{S}) = w(\mathcal{S}) + X\beta^T + \epsilon(\mathcal{S}), \epsilon(\mathcal{S}) \sim \mathcal{N}(0, 5I_n)$

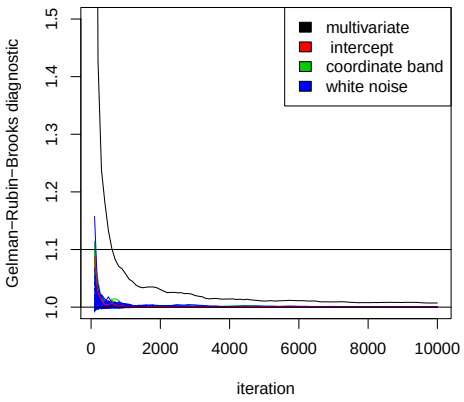
The results of the runs on the toy examples are presented in table 3. The estimates are close to the target and there is no clear gap between the methods.



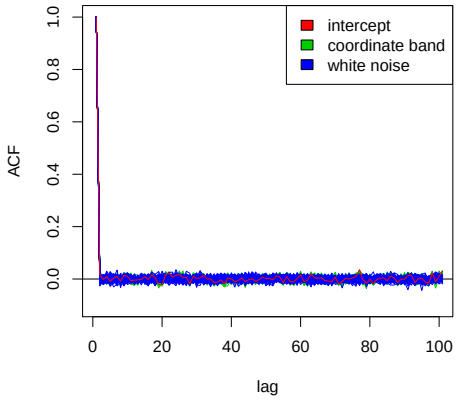
(a) $\hat{R}$ with spNNGP



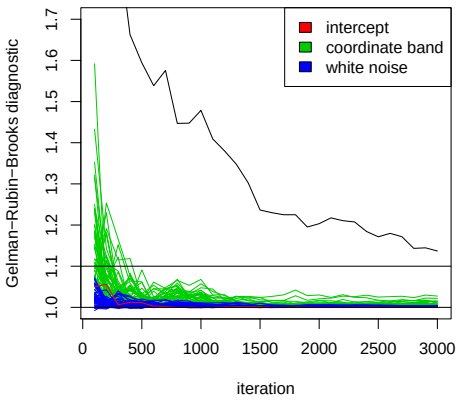
(b) Autocorrelations with spNNGP



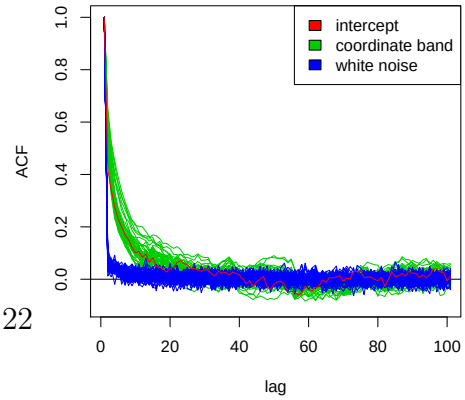
(c) $\hat{R}$ with response spNNGP



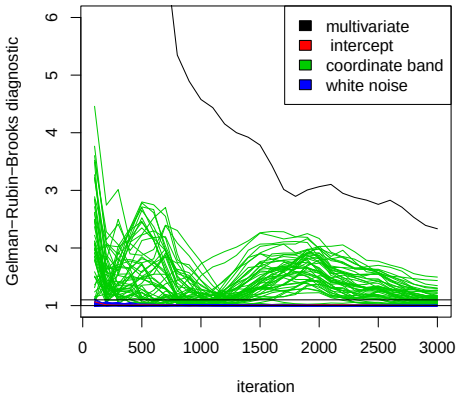
(d) Autocorrelations with response spNNGP



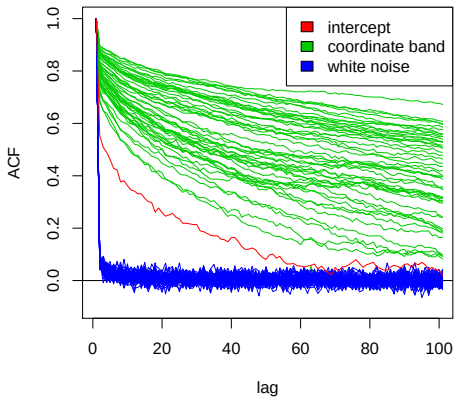
(e) $\hat{R}$ of our implementation



(f) Autocorrelations of our implementation



(g) $\hat{R}$ of our implementation without interweaving



(h) Autocorrelations of our implementation without interweaving

Figure 9: Behavior of the regression coefficients with spNNGP and with our implementation

Table 3: Summary of the toy example runs

(a) Summary of the first toy example

method	n iter.	time (min)	MSE	latent var.	noise var.	range
spNNGP	15000	28	0.40	1.07	4.99	1.10
Our code	3000	28	0.38	1.08	5.00	1.01
spNNGP res.	8000	13		1.06	5.00	0.99
true values				1.00	5.00	1.00

(b) Summary of the second toy example

method	n iter.	time (min)	MSE	$\beta-$MSE	latent var.	noise var.	range
spNNGP	25000	74	0.45	0.053	0.91	5.08	0.90
Our code	3000	36	0.42	0.057	0.98	5.06	1.13
spNNGP res.	10000	50		0.047	0.88	5.10	0.72
true values					1.00	5.00	1.00

Due to the fast-mixing AS-IS architecture from [12] and [23], our implementation needed far fewer iterations than the latent model of spNNGP (Even taking into account the fact that one AS-IS iteration needs two covariance parameters updates): our model takes thousands of iterations to converge, while spNNGP needs tens of thousands. The response model, in spite of its frugality, needed a few thousands iterations to converge, like our implementation. The running times end up being of the same order, due to the efficient multi-process implementation of spNNGP which offsets the number of iterations.

Let's now focus on the behavior of the regression coefficients in the second toy example (Figure 9). The best model regarding the mixing of the regression coefficients is incontestably the response model (9c, 9d). However, the covariance parameters needed more time to mix than the regression coefficients, explaining why 10000 iterations were needed. Moreover, the response model cannot retrieve the latent field, explaining why its MSE could not be computed. Except for the response model, we can see that the coherent regression coefficients of X_1, in green in 9, mix more slowly than the fuzzy coefficients of X_2, in blue. Nonetheless, for our implementation, the $\hat{R}$ diagnostics dropped to 1 in a few hundred iterations (figure 9e), against the tenths of thousands needed for spNNGP (figure 9a). For our implementation, the autocorrelations dropped to 0 after a few dozen iterations (figure 9f). The auto-correlations of spNNGP for the regression coefficients of X_1 were still between 0.4 and 0.6 after 100 iterations (figure 9b), while the autocorrelation for the coefficients of X_2 remained stuck slightly above 0. It is then clear that the chains behave much better in our implementation than in spNNGP. Moreover, the good behavior of our implementation could not be reproduced if we did not indicate that interweaving could be used, see figures 9g, 9h.

4.3. Application to lead contamination analysis

We used our implementation to study a heavy metal contamination data set proposed by Hengl in [24][3]. The dataset gathers measurements made by the United States Geological Survey of [25] and several covariates, including geophysical and environmental information about the sampling site, and potential contamination sources nearby. We added the predominant subsoil rock type given by the USGS study presented in [26][4]. We scaled

[3]https://spatial-analyst.net/book/NGS8HMC
[4]https://mrdata.usgs.gov/geology/state/

the quantitative regressors. After removing missing data, there were 64274 observations. We assumed the model

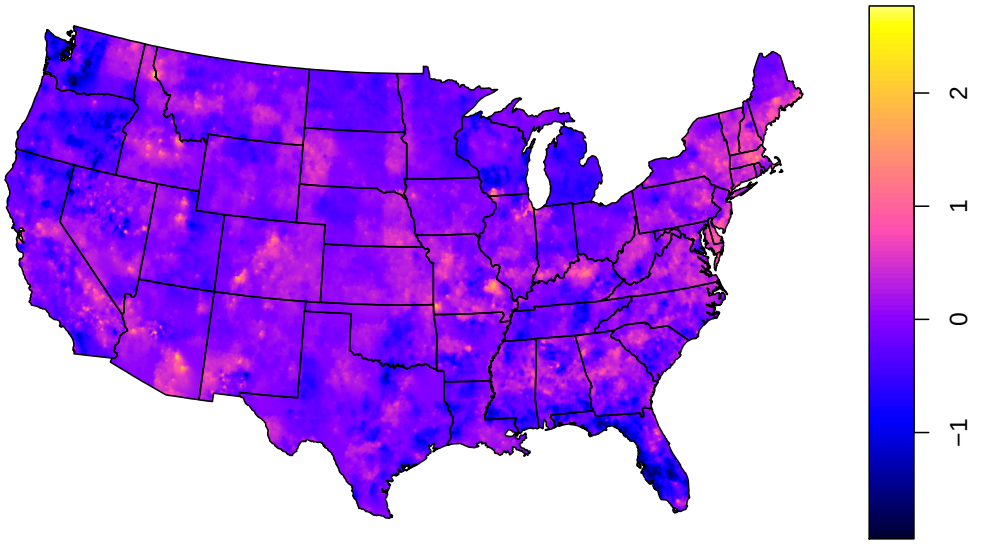
$$log(z(s)) = w(s) + X(s)\beta^T + \epsilon(s),$$

s being the sampling location, $X(\cdot)$ being the aforementioned covariates, $w(\cdot)$ being a latent Gaussian field with exponential covariance on the sphere, and ϵ being a white noise. The model converged in 4000 iterations, and 1 hour and 38 minutes were needed.

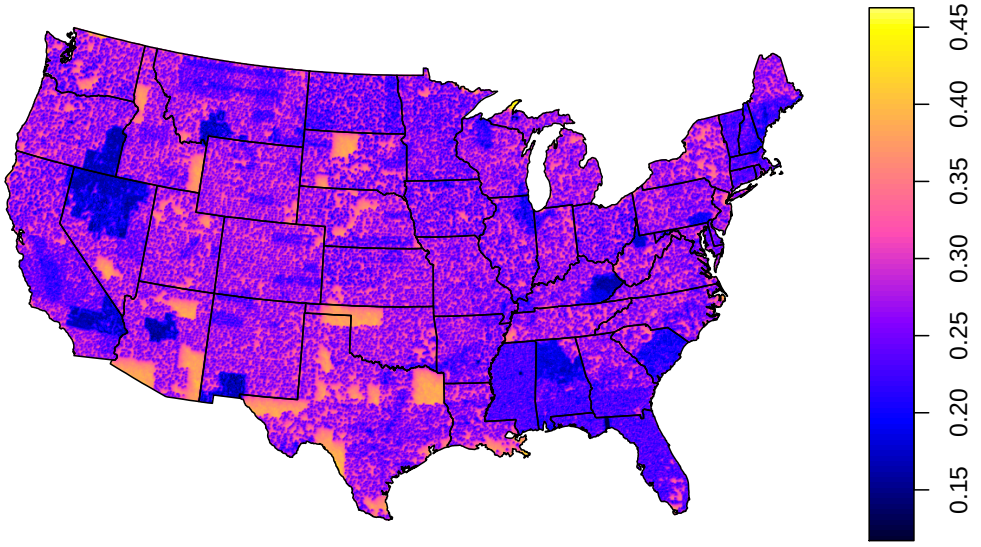
We tried to analyze the real data set with spNNGP in order to compare the results and the running time. Surprisingly, spNNGP had a pathological behavior in spite of its good performances on simulated data. The scale parameter kept straying towards values several orders of magnitude above the variance of the observed field, even with starting points corresponding to our estimates. This behavior was observed with both latent and response models, and various orderings of the locations.

We present our implementation's estimates of the covariance parameters and some of the fixed effects in table 4. We left out some regressors, such as the geological classification, indications about nearby mineral observations, and the geophysical characteristics of the sampling site. The variances of the latent field and the noise have equivalent orders ($\sigma^2 = 0.20, \tau^2 = 0.18$). The spatial range is 30 Km. With a rule of the thumb, this means that the correlation drops to 10% of the scale for locations separated by 60 Km. The regressors behave as expected: the urbanization level and contamination indicators have a positive, certain effect on lead concentration. However, the values of the regression coefficients remain modest with respect to the scale of the latent field.

We also provide predictions of the latent field on a 5-Km grid on the territory of the USA mainland. Predictions at un-observed locations are done using the MCMC samples of the covariance parameters θ and $w(\mathcal{S})$, see for example [6]. We report the predicted latent mean and standard deviation in figures 10a and 10b. The standard deviation map must be put in relation with the sampling sites map (Figure 11). The patches with high standard deviation correspond to zones with no measurements, while territories with dense sampling, such as Florida, will have low predicted standard deviation.



(a) Predicted latent mean



(b) Predicted latent standard deviation (closely related to sampling density)

Figure 10: Latent field predictions

Table 4: Summary of the covariance parameters and a subset of the fixed effects

	mean	qtile 0.025	median	qtile 0.975	st dev
Scale	0.198	0.188	0.198	0.209	0.0053
Noise variance	0.178	0.175	0.178	0.181	0.0017
Range (Km)	35.6	33.3	35.5	38.3	1.2700
(Intercept)	2.83	2.79	2.83	2.87	0.019
Air pollution dsty	0.0403	0.0195	0.0404	0.0605	0.0106
Mineral operations dsty	0.0180	0.0044	0.0182	0.0312	0.0069
Industrial toxic reject dsty	0.0641	0.0433	0.0639	0.0852	0.0107
Carbon biomass dsty	-0.0543	-0.0673	-0.0541	-0.0412	0.0066
Population dsty	0.1360	0.1100	0.1360	0.1640	0.0138
Night light	0.0384	0.0303	0.0384	0.0466	0.0042
Roads dsty	0.0193	0.0139	0.0193	0.0248	0.0028

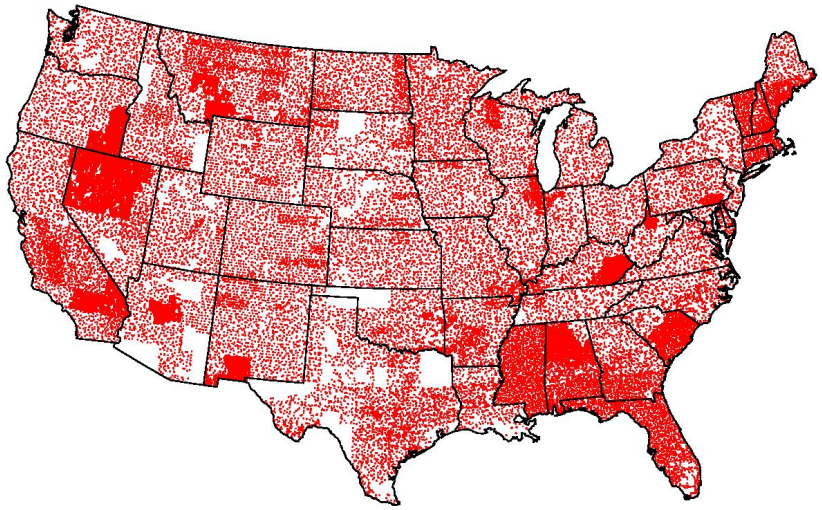


Figure 11: Sampling sites

5. Discussion

We presented two ways to improve the behavior of NNGPs with full data augmentation, that can be applied simply to previous implementations. What's more, while we assumed a Gaussian data model throughout the article, the two methods we proposed can easily be applied to other models.
While our article focused on a basic NNGP model, our field centering may have applications in complex models. Space-varying regression coefficients are an extension to GP models ([3, 16]). If we consider the latent field $w(\cdot)$ as a space-varying intercept, it seems natural to try to center a spatially variable parameter on the corresponding fixed effect. The extension to other fixed effects we presented could prove valuable in the case in which the regressor with a spatially variable β is correlated with other variables from X.
Another possible extension could be a GP defined as the sum of two or more GPs. It could have an interest in various applications, such as: modelling seasonality in a space-time process, modelling a process with short-range and long-range interactions, defining one non-separable space-time process as a sum of two separable processes. The equivalent of standard parametrization would be $z(\cdot) = \beta_0 + w_1(\cdot) + w_2(\cdot) + \epsilon$, $w_1(\cdot)$ and $w_2(\cdot)$ being GPs of mean 0. One could try out a Russian doll centering: $z(\cdot) = v_1(\cdot) + \epsilon$ where $v_1(\cdot)$ has mean $v_2(\cdot)$, and $v_2(\cdot)$ has mean β_0. In this case, in addition to applying the extension to other fixed effects we presented, it might be necessary to interweave the standard and the "Russian doll" parametrizations.
Beyond the improvements of chromatic sampling in the NNGP algorithm, exploration of the moralized graph could be an interesting approach to study Vecchia's approximation and evaluate heuristics concerning ordering and picking parents. For example, Guinness [5] has explored how various ordering and grouping strategies affected the Kullback-Leibler divergence of Vecchia's approximation with respect to the full GP density. Those strategies have a graphical translation. Grouping takes an existing graph and adds new edges, making it closer to the full GP graph (i.e. the saturated DAG and moralized graph). Ordering modifies the structure of the graph and the length of the edges, just like the mixing of observations explored in [9]. For example, a coordinate or a middle-out ordering with Nearest Neighbor heuristic will make a graph where each vertex is connected to its closest neighbors, while we could use a classical concept of Geography and say that a random or a max-min ordering will generate graphs not unlike a Christallerian system. Focusing on the neighbor-picking heuristics provides a close-up shot of what

is going on and has a direct algorithmic translation, but some descriptive statistics about the moralized graphs could give a more general view.

[1] A. E. Gelfand, P. Diggle, P. Guttorp, M. Fuentes, Handbook of Spatial Statistics (Chapman & Hall CRC Handbooks of Modern Statistical Methods), Chapman & Hall CRC Handbooks of Modern Statistical Methods, Taylor and Francis, 2010.
URL http://gen.lib.rus.ec/book/index.php?md5=96CFC718B8554AB6448050DB14685E28 2

[2] G.-A. Fuglstad, D. Simpson, F. Lindgren, H. Rue, Interpretable priors for hyperparameters for gaussian random fields, arXiv preprint arXiv:1503.00256 (2015). 2

[3] A. Datta, S. Banerjee, A. O. Finley, A. E. Gelfand, Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets, Journal of the American Statistical Association 111 (514) (2016) 800–812. 2, 3, 5, 8, 10, 11, 12, 14, 20, 28

[4] M. Katzfuss, J. Guinness, A general framework for Vecchia approximations of Gaussian processes, arXiv e-prints (2017) arXiv:1708.06302arXiv:1708.06302. 3, 11

[5] Guinness, Permutation and grouping methods for sharpening gaussian process approximations, Technometrics 60 (4) (2018) 415–429. arXiv:https://doi.org/10.1080/00401706.2018.1437476, doi:10.1080/00401706.2018.1437476.
URL https://doi.org/10.1080/00401706.2018.1437476 3, 14, 28

[6] A. O. Finley, A. Datta, B. D. Cook, D. C. Morton, H. E. Andersen, S. Banerjee, Efficient algorithms for bayesian nearest neighbor gaussian processes, Journal of Computational and Graphical Statistics 28 (2) (2019) 401–414. 3, 4, 5, 25

[7] K. Guinness, GpGp: Fast Gaussian Process Computation Using Vecchia's Approximation (2018).
URL https://CRAN.R-project.org/package=GpGp 3, 20

[8] A. Finley, A. Datta, S. Banerjee, spnngp: spatial regression models for large datasets using nearest neighbor gaussian processes, R package version 0.1 1 (2017). 3, 4, 20

[9] M. L. Stein, Z. Chi, L. J. Welty, Approximating likelihoods for large spatial data sets, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (2) (2004) 275–296. 3, 28

[10] L. Zhang, A. Datta, S. Banerjee, Practical bayesian modeling and inference for massive spatial data sets on modest computing environments, Statistical Analysis and Data Mining: The ASA Data Science Journal 12 (3) (2019) 197–209. 4

[11] L. Knorr-Held, H. Rue, On block updating in markov random field models for disease mapping, Scandinavian Journal of Statistics 29 (4) (2002) 597–614. 4, 12

[12] Y. Yu, X.-L. Meng, To center or not to center: That is not the question—an ancillarity–sufficiency interweaving strategy (asis) for boosting mcmc efficiency, Journal of Computational and Graphical Statistics 20 (3) (2011) 531–570. 4, 9, 20, 24

[13] M. Heinonen, H. Mannerström, J. Rousu, S. Kaski, H. Lähdesmäki, Non-stationary gaussian process regression with hamiltonian monte carlo, in: Artificial Intelligence and Statistics, 2016, pp. 732–740. 4

[14] J. E. Gonzalez, Y. Low, A. Gretton, C. Guestrin, Parallel gibbs sampling: From colored fields to thin junction trees, 2011. 4

[15] A. E. Gelfand, S. K. Sahu, B. P. Carlin, Efficient parametrisations for normal linear mixed models, Biometrika 82 (3) (1995) 479–488. 6

[16] S. Banerjee, A. E. Gelfand, A. O. Finley, H. Sang, Gaussian predictive process models for large spatial data sets, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70 (4) (2008) 825–848. doi:10.1111/j.1467-9868.2008.00663.x.
URL http://doi.wiley.com/10.1111/j.1467-9868.2008.00663.x
8, 28

[17] S. L. Lauritzen, Graphical models, Oxford Statistical Science Series, OUP, 1996.

URL http://gen.lib.rus.ec/book/index.php?md5=7ECA79CDE5FF909E7E0E7FC8A02D8A80 12

[18] D. Eddelbuettel, R. François, J. Allaire, K. Ushey, Q. Kou, N. Russel, J. Chambers, D. Bates, Rcpp: Seamless r and c++ integration, Journal of Statistical Software 40 (8) (2011) 1–18. 20

[19] A. Gelman, D. B. Rubin, Inference from iterative simulation using multiple sequences, Statistical science 7 (4) (1992) 457–472. 20

[20] S. P. Brooks, A. Gelman, General methods for monitoring convergence of iterative simulations, Journal of computational and graphical statistics 7 (4) (1998) 434–455. 20

[21] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2018). URL https://www.R-project.org/ 20

[22] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2018). URL https://www.R-project.org/ 20

[23] M. Filippone, M. Zhong, M. Girolami, A comparative evaluation of stochastic-based inference methods for gaussian process models, Machine Learning 93 (1) (2013) 93–114. 24

[24] T. Hengl, A practical guide to geostatistical mapping, Hengl Amsterdam, 2009. 24

[25] J. N. Grossman, et al., The National Geochemical Survey-database and documentation (2004). 24

[26] J. D. Horton, The state geologic map compilation (sgmc) geodatabase of the conterminous united states (2017). 24

[27] H. Rue, L. Held, Gaussian Markov random fields: theory and applications, 1st Edition, Monographs on statistics and applied probability 104, Chapman & Hall/CRC, 2005.
URL http://gen.lib.rus.ec/book/index.php?md5=800474D84710FBE4C4F989801371938A 32

Appendix A. Stochastic form of the intercept-field model

We obtain the full conditionals of w_c and w_s using the conditional expectation and variance formulas using precision matrices of [27], the joint precision of both (w_s, z) or (w_c, z) being $\begin{bmatrix} \tilde{Q}+\tau^2 I_n & -\tau^2 I_n \\ -\tau^2 I_n & \tau^2 I_n \end{bmatrix}$. The expectation of w_s is 0, the expectation of w_c is β_0, and the expectation of z is always β_0.

Distributions with the standard model

The full conditional distributions of β_0 and w_s are:

$$[\beta_0|w_s] \sim \mathcal{N}(\overline{z-w_s}, \tau^2/n), [w_s|\beta_0] \sim \mathcal{N}(-(\tilde{Q}+I_n/\tau^2)^{-1}(-I_n/\tau^2)(z-\beta_0), (\tilde{Q}+I_n/\tau^2)^{-1}).$$

Note $\mathbf{1}$ the vector of length n and filled with ones. From the second full conditional, we have a formula for the mean of w_s, which is obtained with $\overline{w_s} = \mathbf{1}^t w_s/n$. It has 3 terms: one is fixed, the second is a geometric carry-over of β_0, and the third is stochastic:

$$[\overline{w_s}|\beta_0] \sim \underbrace{\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}(z)/n}_{\text{fixed}} - \underbrace{(\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}\mathbf{1}/n)(\beta_0)}_{\text{carry-over}} + \underbrace{\mathcal{N}((0, \mathbf{1}^T(\tilde{Q}+I_n/\tau^2)^{-1}\mathbf{1}/n^2)}_{\text{innovation}}.$$

Injecting the full conditional of $\bar{w}_s$ into β_0's, we identify an expression with 3 terms like before:

$$\begin{aligned}[\beta_0^{t+1}|\beta_0^t] \sim\ & \underbrace{\overline{z} - \mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}(z)/n}_{\text{fixed}} + \underbrace{(\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}\mathbf{1}/n)\beta_0^t}_{\text{carry-over}} \\ & + \underbrace{\mathcal{N}(0, \mathbf{1}^T(\tilde{Q}+I_n/\tau^2)^{-1}\mathbf{1}/n^2 + \tau^2/n)}_{\text{innovation}}.\end{aligned}$$

Distributions with the centered model

The full conditional of β_0 and w_c are:

$$[\beta_0|w_c] \sim \mathcal{N}(\mathbf{1}^T\tilde{Q}w_c/\mathbf{1}^T\tilde{Q}\mathbf{1}, 1/\mathbf{1}^T\tilde{Q}\mathbf{1}), [w_c|\beta_0] \sim \mathcal{N}(\beta_0+(\tilde{Q}+I_n/\tau^2)^{-1}(z-\beta_0)/\tau^2, (\tilde{Q}+I_n/\tau^2)^{-1})$$

The mean of w_c behaves like the mean of w_s except for the term that depends on β_0:

$$\begin{aligned}[\overline{w_c}|\beta_0] \sim\ & \underbrace{\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}(z)/n}_{\text{fixed}} - \underbrace{(1-(\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}\mathbf{1}/n))\beta_0}_{\text{carry-over}} \\ & + \underbrace{\mathcal{N}(0, \mathbf{1}^T(\tilde{Q}+I_n/\tau^2)^{-1}\mathbf{1}/n^2)}_{\text{innovation}}.\end{aligned} \tag{A.1}$$

Injecting the full conditional of w_c into the full conditional of β_0, we have

$$[\beta_0^{t+1}|\beta_0^t] \sim \underbrace{\mathbf{1}^T\tilde{Q}(\tilde{Q}+I_n/\tau^2)^{-1}z/\tau^2\mathbf{1}^T\tilde{Q}\mathbf{1}}_{\text{fixed}} + \underbrace{\mathbf{1}^T\tilde{Q}(I_n-(\tau^2\tilde{Q}+I_n)^{-1})\mathbf{1}\beta_0/\mathbf{1}^T\tilde{Q}\mathbf{1}}_{\text{carry-over}}$$
$$+\underbrace{\mathcal{N}(0,\mathbf{1}^T\tilde{Q}(\tilde{Q}+I_n/\tau^2)^{-1}\tilde{Q}\mathbf{1}/(\mathbf{1}^T\tilde{Q}\mathbf{1})^2+1/\mathbf{1}^T\tilde{Q}\mathbf{1})}_{\text{innovation}}.$$

Passing to the SVD

Let's compare first the expressions of $[\overline{w_s}|\beta_0]$ and $[\overline{w_c}|\beta_0]$. Denote the diagonalization $\tilde{Q}=V^T\lambda V$, V being a square matrix of eigenvectors and λ being a diagonal matrix of eigenvalues. The eigenvalues are positive since $\tilde{Q}=\tilde{R}^T\tilde{R}$. Using the fact that adding I_n adds 1 to every eigenvalue without affecting the eigenvectors,

$$(\tau^2\tilde{Q}+I_n)^{-1}/n = V^T(\tau^2\lambda+I_n)^{-1}V/n.$$

Let $\alpha=(\alpha_1,\ldots,\alpha_n)$ be the coordinates of $\mathbf{1}$ in the orthonormal basis defined by V.

$$\mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}\mathbf{1}/n = \Sigma_{i=1}^n\alpha_i^2(\tau^2\lambda+I_n)^{-1}_{i,i}/n.$$

Using that $\tilde{Q}$ is positive-definite on the left and that $\Sigma_1^n\alpha_i =< \mathbf{1},\mathbf{1} >= n$ on the right, we have

$$0 \leq \mathbf{1}^T(\tau^2\tilde{Q}+I_n)^{-1}\mathbf{1}/n \leq 1.$$

As for the centered model, we re-write:

$$\tilde{Q}(I_n-(\tau^2\tilde{Q}+I_n)^{-1}) = V^T(\lambda(I_n-(\tau^2\lambda+I_n)^{-1}))V = V^T(\tau^2\lambda^2(\tau^2\lambda+I_n)^{-1})V.$$

Once this is done, we can express the fraction of β_0^t which is conserved in β^{t+1} in the centered model as

$$\mathbf{1}^T\tilde{Q}(I_n-(\tau^2\tilde{Q}+I_n)^{-1})\mathbf{1}/\mathbf{1}^T\tilde{Q}\mathbf{1} = \Sigma_{i=1}^n\left((\alpha_i^2\lambda_{i,i})(\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1)\right)/\Sigma_{i=1}^n\alpha_i^2\lambda_{i,i}.$$

Like before, thanks to the fact that the eigenvalues of $\tilde{Q}$ are positive,

$$0 \leq (\tau^2\lambda_{i,i})/(\tau^2\lambda_{i,i}+1) \leq 1.$$

Appendix B. Coloring

Appendix B.1. Details about the coloring algorithms

Algorithm 2 Naive greedy coloring

input A ▷ Input adjacency matrix
$(c_1, \ldots, c_n) = (0, \ldots, 0)$ ▷ Initialize colors
for $c_i \in (c_1, \ldots, c_n)$ **do** ▷ Coloration loop
$c_i = min((1, \ldots, n) \backslash (c_J))$ with $J = \{J / A_{i,j} = 1\}$ ▷ Using smallest available color
end for
return $(c_1, \ldots, c_n)$

Algorithm 3 Degree greedy coloring

input A ▷ Input adjacency matrix
$(c_1, \ldots, c_n) = (0, \ldots, 0)$ ▷ Initialize colors
find $(nd_1, \ldots, nd_n) = (1, \ldots, 1) \cdot A$ ▷ Compute connection degrees of nodes
find $(o(1), \ldots, o(n))$, a permutation of $1, \ldots, n$ such that $i < j \Rightarrow nd_{o(i)} \leq nd_{o(j)}$ ▷ order nodes by decreasing connection degree
for $c_i \in (c_{o(1)}, \ldots, c_{o(n)})$ **do** ▷ Coloration loop
$c_i = min((1, \ldots, n) \backslash (c_J))$ with $J = \{J / A_{i,j} = 1\}$ ▷ Using smallest available color
end for
return $(c_1, \ldots, c_n)$

Algorithm 4 DSATUR

input A ▷ Input adjacency matrix
$(c_1, \ldots, c_n) = (0, \ldots, 0)$ ▷ Initialize colors
$(sd_1, \ldots, sd_n) = (0, \ldots, 0)$ ▷ Initialize saturation degrees
$(nd_1, \ldots, nd_n) = (1, \ldots, 1) \cdot A$ ▷ Compute connection degrees of nodes
while $0 \in (c_1, \ldots, c_n)$ **do** ▷ Coloration loop
    $j = \{i / c_i == 0\}$
    $j = \{i \in j / sd_i == max_{i \in j}(sd_i)\}$ ▷ Saturation degree selection rule
    if $\#j > 1$ **then**
        $j = \{i \in j / nd_i == max_{i \in j}(nd_i)\}$ ▷ Node degree tiebreaking rule
    end if
    if $\#j > 1$ **then**
        $j = min(j)$ ▷ lexicographical tiebreaking rule
    end if
    $c_j = min((1, \ldots, n) \backslash (c_{i/A_{i,j}=1}))$ ▷ Using smallest available color
    $sd_{i/A_{i,j}=1} = sd_{i/A_{i,j}=1} + 1$ ▷ Updating saturation degrees
end while
return $(c_1, \ldots, c_n)$

Appendix B.2. Results of coloring experiments

Table B.5: Case-by-case mean number of colors in the pilot experiment.

			Coordinate ordering			Max-min ordering			Random ordering		
m	n	d	degree	dsatur	naive	degree	dsatur	naive	degree	dsatur	naive
5	500	2	7.4	6.0	6.0	9.8	9.1	10.1	10.3	9.2	10.0
		3	7.9	6.0	6.0	10.8	9.6	11.0	10.6	10.0	11.1
	1000	2	8.2	6.0	6.0	10.1	9.4	10.3	10.6	9.3	10.2
		3	8.0	6.0	6.0	11.8	10.0	11.1	11.3	10.1	11.1
	2000	2	8.6	6.0	6.0	10.5	9.6	10.3	10.7	9.9	10.2
		3	8.9	6.0	6.0	11.6	10.1	11.7	11.8	10.0	11.4
10	500	2	12.9	11.0	11.0	18.7	17.4	18.9	19.0	17.8	19.4
		3	13.0	11.0	11.0	20.9	19.1	21.2	21.0	18.9	21.3
	1000	2	13.7	11.0	11.0	19.2	17.8	19.5	19.9	17.9	20.0
		3	13.6	11.0	11.0	21.3	19.4	22.3	21.6	19.4	22.0
	2000	2	14.8	11.0	11.0	20.2	18.2	20.0	20.7	18.6	19.8
		3	15.0	11.0	11.0	22.3	19.7	22.6	22.7	19.9	22.4
20	500	2	23.0	21.0	21.0	36.5	33.9	37.4	36.8	34.2	37.5
		3	23.3	21.0	21.0	40.8	37.1	44.1	40.5	37.5	43.4
	1000	2	24.9	21.0	21.0	37.6	35.0	38.2	38.6	35.2	38.5
		3	23.9	21.0	21.0	42.9	38.4	45.3	43.1	38.8	44.7
	2000	2	26.0	21.0	21.0	38.6	35.9	38.8	39.1	36.2	39.1
		3	25.7	21.0	21.0	44.8	39.6	46.4	44.7	40.0	45.6

Table B.6: Case-by-case mean number of colors for large graphs.

			Coordinate ordering		Max-min ordering		Random ordering	
m	n	d	degree	naive	degree	naive	degree	naive
5	50000	2	10.0	8.8	11.3	11.0	12.3	11.1
		3	10.0	8.7	13.1	12.6	13.1	12.3
	100000	2	10.1	9.0	11.7	11.0	12.1	11.0
		3	10.0	9.1	13.1	13.0	13.2	12.3
	200000	2	10.1	9.3	11.9	11.2	12.1	11.4
		3	10.3	9.7	13.3	13.0	13.5	12.8
10	50000	2	17.1	13.8	21.5	21.0	22.7	20.8
		3	17.0	14.0	24.5	24.1	25.5	23.7
	100000	2	17.0	15.6	22.0	21.2	22.9	21.0
		3	17.0	15.5	24.9	24.2	25.6	23.9
	200000	2	17.9	16.7	22.2	21.3	22.9	21.1
		3	18.0	16.8	25.2	24.2	26.4	24.3
20	50000	2	31.4	21.1	41.4	40.1	43.0	40.4
		3	31.2	21.6	49.7	48.2	50.2	47.8
	100000	2	30.8	25.1	41.9	40.7	44.3	40.8
		3	31.0	24.6	50.0	48.5	50.8	47.6
	200000	2	30.3	28.6	41.7	40.7	44.3	40.8
		3	30.2	28.7	50.5	48.9	51.6	48.2

Table B.7: Case-by-case mean number of colors for blocked graphs.

			Coordinate ordering			Max-min ordering			Random ordering		
m	blocks	d	degree	dsatur	naive	degree	dsatur	naive	degree	dsatur	naive
5	10	2	2.9	2.5	3.0	6.2	6.2	6.6	6.5	6.5	6.7
		3	2.7	2.3	2.7	7.5	7.5	7.5	7.3	7.3	7.5
	20	2	3.0	2.4	3.0	7.9	7.9	8.6	7.6	7.5	8.3
		3	3.0	2.6	3.0	9.0	8.5	9.6	8.6	8.3	9.5
	50	2	3.0	2.4	3.0	8.9	8.6	10.8	8.6	8.1	10.2
		3	3.0	2.4	3.0	10.9	10.2	12.7	10.3	9.4	12.0
	100	2	3.0	2.4	3.0	9.8	9.1	12.0	9.0	8.6	11.3
		3	3.0	2.2	3.0	11.8	11.1	14.0	11.0	10.2	13.0
	500	2	3.0	2.9	3.1	10.3	9.3	14.7	10.1	9.1	14.3
		3	3.0	3.0	3.0	13.2	11.7	17.3	12.4	11.1	16.1
10	10	2	2.9	2.5	3.0	8.1	8.1	8.2	8.6	8.6	8.6
		3	2.7	2.3	2.7	9.3	9.3	9.3	9.3	9.3	9.3
	20	2	3.0	2.4	3.0	11.4	11.4	12.2	11.9	11.9	12.5
		3	3.0	2.6	3.0	13.5	13.4	13.8	12.5	12.3	13.0
	50	2	3.0	2.4	3.0	14.8	14.2	17.1	14.0	13.6	16.0
		3	3.0	2.4	3.0	17.5	16.6	19.0	16.2	15.5	18.2
	100	2	3.0	2.4	3.0	16.7	16.0	20.6	15.7	15.0	19.2
		3	3.0	2.2	3.0	19.8	18.5	22.9	18.0	17.1	21.2
	500	2	3.6	3.1	4.1	19.6	18.2	26.5	18.6	17.0	24.5
		3	3.8	3.4	4.3	22.9	21.0	30.9	20.9	18.8	28.0
20	10	2	2.9	2.5	3.0	9.8	9.8	9.8	9.9	9.9	9.9
		3	2.7	2.3	2.7	10.0	10.0	10.0	10.0	10.0	10.0
	20	2	3.0	2.4	3.0	16.2	16.2	16.7	16.3	16.3	16.5
		3	3.0	2.6	3.0	17.7	17.7	17.7	17.1	17.1	17.2
	50	2	3.0	2.4	3.0	25.1	24.5	27.0	23.0	22.9	25.4
		3	3.0	2.4	3.0	27.9	27.8	30.4	24.8	24.3	27.4
	100	2	3.0	2.4	3.0	30.4	29.1	34.6	26.7	25.7	31.0
		3	3.0	2.2	3.0	33.7	32.4	38.2	29.8	28.9	33.9
	500	2	5.2	4.7	5.6	37.2	35.2	47.6	34.0	31.2	42.6
		3	5.2	4.8	5.8	42.0	39.0	55.0	38.9	35.1	49.1