Efficient simulation of Gaussian Markov random fields by Chebyshev polynomial approximation

Mike Pereira, Nicolas Desassis

Version of Record: https://www.sciencedirect.com/science/article/pii/S2211675318302537
Manuscript_2aeb9396574eabec0bd6bbe144bccc96

Efficient simulation of Gaussian Markov random fields by Chebyshev polynomial approximation

Mike PEREIRA[a,b,*], Nicolas DESASSIS[a]

[a] *Geostatistics team, Geosciences, MINES ParisTech, PSL University, Fontainebleau, France*
[b] *Estimages, Paris, France*

Abstract

This paper presents an algorithm to simulate Gaussian random vectors whose precision matrix can be expressed as a polynomial of a sparse matrix. This situation arises in particular when simulating Gaussian Markov random fields obtained by the finite elements discretization of the solutions of some stochastic partial derivative equations. The proposed algorithm uses a Chebyshev polynomial approximation to compute simulated vectors with a linear complexity. This method is asymptotically exact as the approximation order grows. Criteria based on tests of the statistical properties of the produced vectors are derived to determine minimal orders of approximation.

Keywords— S imulation, GMRF, SPDE, Finite element method, Chebyshev approximation, Krylov subspaces

1. Introduction

Gaussian random fields (GRF) are widely used to model spatially correlated data in environmental and earth sciences (Chilès and Delfiner, 2012; Lantuéjoul, 2013; Wackernagel, 2013). The stochastic simulation of such fields (also called geostatistical simulation) is a common process in risk analysis (Chilès and Delfiner, 2012). Indeed, each simulation is seen as an alternate but plausible version of the reality. Spatial uncertainty can then be assessed in problems where the variables of interest are partially observed through comparisons over a set of simulations. There are two main classes of simulation algorithms. Exact algorithms aim at reproducing exactly the statistical properties of a targeted model. They include methods based on the factorization of covariance matrices (Davis, 1987b) or on the spectral properties of random fields (Pardo-Igúzquiza

*Corresponding author
Email addresses: mike.pereira@mines-paristech.fr (Mike PEREIRA), nicolas.desassis@mines-paristech.fr (Nicolas DESASSIS)

and Chica-Olmo, 1993; Dietrich and Newsam, 1993). These methods are suitable for small-sized problems or for compactly-supported covariance functions (Kaufman et al., 2008; Bevilacqua et al., 2019) which lead to sparse covariance matrices (Furrer et al., 2010). However, for general covariance functions or when the size of the problem is very large, they are replaced by approximate algorithms that generate simulations from a nearly multi-Gaussian spatial distribution or with approximated covariance properties. Examples of such algorithms include the turning bands (Matheron, 1973; Emery and Lantuéjoul, 2006; Emery et al., 2016) and the continuous spectral methods (Shinozuka and Jan, 1972), and also the sequential Gaussian simulation algorithm (Deutsch and Journel, 1998). Note also the circular embedding method that leverages the regularity of the simulation grids (Schlather et al., 2015; Wood and Chan, 1994) and which will be discussed in Section 2.2.

Continuous Markov random fields are particularly suited models for geostatistical simulations thanks to the computational efficiency they provide. Precisely, the sparsity of the precision matrices of their discretization allows fast computations of samples (and likelihood) (Rue and Held, 2005). When stationary and isotropic, these random fields have a spectral density, that is the Fourier transform of the covariance function, of the form $f(\omega) = 1/\mathrm{P}(\|\omega\|^2)$ where P is a real strictly positive polynomial on $\mathbb{R}_+$ (Rozanov, 1977). Equivalently, they can be seen as solutions of the stochastic partial derivative equation (SPDE) defined as (Rozanov, 1977; Lang and Potthoff, 2011; Simpson et al., 2012):

$$\mathrm{P}(-\Delta)^{1/2} Z = \mathcal{W} \tag{1}$$

where $\mathcal{W}$ is a Gaussian white noise and $\mathrm{P}(-\Delta)^{1/2}$ is the differential operator defined as:

$$\mathrm{P}(-\Delta)^{1/2}[.] = \mathscr{F}^{-1}\left[w \mapsto \sqrt{\mathrm{P}(\|\omega\|^2)}\mathscr{F}[.](\omega)\right]$$

where $\mathscr{F}$ denotes the Fourier transform operator.

For instance, following the results from Whittle (1954), Lindgren et al. (2011) consider stationary solutions of the SPDE:

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

with $\kappa > 0$, $\tau > 0$ and α an integer greater than half the dimension of the space, to characterize GRFs with Matérn covariance (or Matérn fields). They even use this result to extend isotropic Matérn fields to manifolds, and to non-stationary and even oscillating formulations (Lindgren et al., 2011).

SPDE (1) can be numerically solved using the finite element method. In that case, it is solved on a triangulated domain, and a finite element representation of the solution is built as:

$$Z(x) = \sum_i z_i \psi_i(x)$$

for finite and deterministic basis functions $\{\psi_i\}$ and Gaussian weights $\{z_i\}$. Simulating a solution is then equivalent to simply simulate the Gaussian weights $\{z_i\}$. In particular, the precision matrix of these weights can be specified using

weak formulations of the SPDE, and has the form:

$$\boldsymbol{Q} = \boldsymbol{D}\mathrm{P}(\boldsymbol{S})\boldsymbol{D} \tag{3}$$

where $\boldsymbol{D}$ is a diagonal matrix with strictly positive entries and $\boldsymbol{S}$ is a real, symmetric and positive semi-definite matrix. In particular, when piecewise linear basis functions are considered, $\boldsymbol{S}$ is a very *sparse* matrix, whose non-zero entries correspond to adjacent nodes in the triangulation.

Given that the precision matrix is known, the simulation of solutions is generally performed by matrix factorisation methods involving the Cholesky decomposition of $\boldsymbol{Q}$. Even if the sparsity inherited from the Markovian properties of the field reduces the complexity of an otherwise too expensive factorisation (Davis, 2006), computation and storage problems still arise for large simulation domains or when the dimension of the space increases (Simpson et al., 2008).

This article introduces instead a computationally efficient algorithm to simulate any Gaussian random vectors whose precision matrix can be expressed as (3). This algorithm is based on the construction of a polynomial approximation of a factorisation of $\boldsymbol{Q}$. It then relies on matrix-vector products between (a matrix as sparse as) $\boldsymbol{S}$ and vectors. It can produce simulations of vectors with a linear complexity, proportional to the number of non-zero entries of $\boldsymbol{S}$. This approach can be seen as an adaptation of the simulation algorithm first proposed by Davis (1987a) and then developed by Dietrich and Newsam (1995), and based on the polynomial approximation of a square-root of the covariance matrix.

The simulation algorithm presented in this article is equivalent to a filtering technique used in Graph signal processing (GSP) (Hammond et al., 2011). GSP is an emerging field focusing on developing tools to process complex data that are embedded on a graph, i.e. a structure composed of a set of objects, called vertices, and pairwise relationships between them, the edges (Bondy and Murty, 1976). Such data arise naturally in applications such as social, energy, transportation and neural networks. They are modelled as variables indexed by the vertices of the graph, named graph signals. Generalizations of classical signal processing notions and tools, such as the Fourier transform, filtering and translation operators are then used to study these signals (Shuman et al., 2013).

The outline of the article is as follows. In Section 2, methods for the simulation of Gaussian random vectors with known precision matrix are reviewed. In Section 3, the main idea behind the proposed algorithm is introduced and attention is devoted to the polynomial approximation it is based on. In Section 4, the overall workflow of the algorithm is presented, and its complexity and induced error are calculated. Then the framework of statistical tests is used to assess whether the vectors produced by the algorithm respect their targeted distribution, and criteria on the minimal order of approximation are deduced. Moreover, the link between our algorithm and the Krylov subspaces approach is exposed, and a comparison with a more standard method to generate samples of GMRF using the same approach is presented. Finally, in Section 5, examples of application of the algorithm are presented, highlighting the great adaptability of the algorithm for the simulation of Matérn fields and their generalizations.

2. Simulation of Gaussian random vectors

The aim is to simulate a zero-mean Gaussian random vector (GRV) whose precision matrix $\boldsymbol{Q}$ is given by:

$$\boldsymbol{Q} = \boldsymbol{D}\mathrm{P}(\boldsymbol{S})\boldsymbol{D} = \boldsymbol{D}\left(\sum_{l=0}^{L} b_l \boldsymbol{S}^l\right)\boldsymbol{D} \tag{4}$$

where $\mathrm{P} : x \mapsto \sum_{l=0}^{L} b_l x^l$ is a strictly positive polynomial function on $\mathbb{R}_+$; $\boldsymbol{S}$ is a real, sparse, symmetric and positive semi-definite matrix; and $\boldsymbol{D}$ is an invertible diagonal matrix.

2.1. Simulation by matrix factorisation

A non-conditional simulation of a zero-mean GRV $\boldsymbol{z}$ with known precision matrix $\boldsymbol{Q}$ can be obtained through:

$$\boldsymbol{z} = \boldsymbol{L}\boldsymbol{\varepsilon} \tag{5}$$

where $\boldsymbol{\varepsilon}$ is a vector with independent zero-mean, unit variance and normally distributed random components and $\boldsymbol{L}$ is a matrix such that (Gentle, 2009):

$$\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{Q}^{-1} \tag{6}$$

The most widely used candidate for such a matrix $\boldsymbol{L}$ is the Cholesky decomposition of $\boldsymbol{Q}^{-1}$ (Horn and Johnson, 1990). However, in the considered setting, only the precision matrix $\boldsymbol{Q}$ is known and not its inverse. Therefore, the simulation process can be performed in two steps:

Workflow: Simulation of a random vector using Cholesky decomposition

Require: A precision matrix $\boldsymbol{Q}$. A vector of independent standard Gaussian values $\boldsymbol{\varepsilon}$.

Output: A simulated vector $\boldsymbol{z}$ with precision matrix $\boldsymbol{Q}$.

1. Compute $\boldsymbol{Q}_{\mathrm{chol}}$ the Cholesky decomposition of the precision matrix $\boldsymbol{Q}$.
2. Compute the simulated vector $\boldsymbol{z}$ as the solution of the following linear system:

$$\boldsymbol{Q}_{\mathrm{chol}}^T \boldsymbol{z} = \boldsymbol{\varepsilon}$$

Two performance issues arise from this workflow. First, the computation of the Cholesky decomposition of $\boldsymbol{Q}$ is intractable for large problems or when the matrix is not sparse enough (Simpson et al., 2008). Then, once computed, this decomposition must be stored, and is used to solve a linear system. Both these tasks grow more expansive as the size or the filling of $\boldsymbol{Q}_{\mathrm{chol}}$ increases. The idea behind the algorithm presented in this article is to find another candidate for $\boldsymbol{L}$ that would take advantage of the fact that the precision matrix has the form (4).

2.2. Simulation by eigendecomposition

The matrix $\boldsymbol{S}$ being real and symmetric, it is diagonalizable with non-negative eigenvalues $\lambda_1, \ldots, \lambda_n$ and eigenvectors that form an orthonormal basis of $\mathbb{R}^n$ (with n the size of the matrix $\boldsymbol{S}$). Therefore there exists a matrix $\boldsymbol{V}$ satisfying $\boldsymbol{V}^{-1} = \boldsymbol{V}^T$ and:

$$\boldsymbol{S} = \boldsymbol{V} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} \boldsymbol{V}^{-1}$$

It can be shown that for any real polynomial P, $\mathrm{P}(\boldsymbol{S}) := \sum_{l=0}^{L} b_l \boldsymbol{S}^l$ is also a real symmetric matrix, and is diagonalizable in the same eigenbasis as $\boldsymbol{S}$. In particular, the eigenvalues of $\mathrm{P}(\boldsymbol{S})$ are $\mathrm{P}(\lambda_1), \ldots, \mathrm{P}(\lambda_n)$.

Let's then denote $\mathrm{P}(\boldsymbol{S})^{-1/2}$ the matrix defined for strictly positive polynomials P by:

$$\mathrm{P}(\boldsymbol{S})^{-1/2} := \boldsymbol{V} \begin{pmatrix} 1/\sqrt{\mathrm{P}(\lambda_1)} & & \\ & \ddots & \\ & & 1/\sqrt{\mathrm{P}(\lambda_n)} \end{pmatrix} \boldsymbol{V}^{-1} \tag{7}$$

Given that this matrix is symmetric, $\boldsymbol{L} = \boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1/2}$ satisfies (6). So using (5), a field $\boldsymbol{z}$ with precision matrix $\boldsymbol{Q}$ can be generated through:

$$\boldsymbol{z} = \boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1/2}\boldsymbol{\varepsilon} \tag{8}$$

A direct way to compute the matrix $\mathrm{P}(\boldsymbol{S})^{-1/2}$ is through (7) which supposes to diagonalize the matrix $\boldsymbol{S}$, and store its eigenvalues $\lambda_1, \ldots, \lambda_n$ and eigenvectors $\boldsymbol{V}$. The cost associated to this approach is generally prohibitive as it requires $\mathcal{O}(n^3)$ operations and a storage size of $\mathcal{O}(n^2)$.

However, a particular case when this is feasible is worth noticing. Rue and Held (2005) show that the precision matrix of a stationary Gaussian Markov random field defined on a torus (i.e. a regular lattice with cyclic boundary conditions) is block-circulant, with circulant blocks (Wood and Chan, 1994). They deduce that the eigenvalues and the eigenvectors of the precision matrix can be computed using the discrete Fourier Transform (DFT), and therefore without requiring a matrix diagonalization. They then sample from their Gaussian Markov random field using (8), where, following the notations of this section, $P(X) = X$, $\boldsymbol{D}$ is the identity matrix, and so $\boldsymbol{S} = \boldsymbol{Q}$. They just replace the product between the $\boldsymbol{V}$ (resp. $\boldsymbol{V}^{-1}$) and a vector by the DFT (resp. inverse DFT) of this vector.

Davis (1987a), and later Dietrich and Newsam (1995), apply this polynomial approximation trick to the case where the covariance matrix $\boldsymbol{\Sigma}$ of the random field is known. They propose to first approximate the square-root function over an interval containing the eigenvalues of the covariance matrix by a polynomial R. Then a simulation is generated by computing the product $\mathrm{R}(\boldsymbol{\Sigma})\boldsymbol{\varepsilon}$ for a vector of independent standard Gaussian values $\boldsymbol{\varepsilon}$. The computational and storage costs of their approach therefore relies on how easy it is to store $\boldsymbol{\Sigma}$ and to compute matrix-vector products involving $\boldsymbol{\Sigma}$. They present the particular case where simulations of a stationary field on a regular grid are performed, thus

yielding block Toeplitz covariance matrices. The storage cost of such matrices can be minimized using the repetitive structure of Toeplitz matrices, and products with a vector can be performed using the FFT algorithm. In the general case of simulating non-stationary fields on irregular domains, their approach becomes intractable as $\boldsymbol{\Sigma}$ becomes a full matrix with no evident structure. To circumvent these limitations, the algorithm presented hereafter uses the particular expression of the precision matrix given in (3).

3. Polynomial approximation

In the general case, the eigendecomposition of $\boldsymbol{S}$ is inevitable if (8) is used to simulate the GRV. To avoid this expensive operation, the idea is rather to compute a matrix polynomial approximation $\mathrm{P}_{\text{-}1/2}^{(K)}(\boldsymbol{S}) := \sum_{k=0}^{K} \alpha_k \boldsymbol{S}^k$ of $\mathrm{P}(\boldsymbol{S})^{-1/2}$ (of degree K). Indeed, both matrices can be decomposed in the same eigenbasis $\boldsymbol{V}$ as:

$$\mathrm{P}_{\text{-}1/2}^{(K)}(\boldsymbol{S}) = \boldsymbol{V} \begin{pmatrix} \mathrm{P}_{\text{-}1/2}^{(K)}(\lambda_1) & & \\ & \ddots & \\ & & \mathrm{P}_{\text{-}1/2}^{(K)}(\lambda_n) \end{pmatrix} \boldsymbol{V}^T \quad \text{and} \quad \mathrm{P}(\boldsymbol{S})^{-1/2} := \boldsymbol{V} \begin{pmatrix} 1/\sqrt{\mathrm{P}(\lambda_1)} & & \\ & \ddots & \\ & & 1/\sqrt{\mathrm{P}(\lambda_n)} \end{pmatrix} \boldsymbol{V}^T$$

Consequently, to approximate $\mathrm{P}(\boldsymbol{S})^{-1/2}$ by $\mathrm{P}_{\text{-}1/2}^{(K)}(\boldsymbol{S})$, the polynomial $\mathrm{P}_{\text{-}1/2}^{(K)}$ must satisfy:

$$\forall i \in [\![1, n]\!], \quad \mathrm{P}_{\text{-}1/2}^{(K)}(\lambda_i) \approx 1/\sqrt{\mathrm{P}(\lambda_i)} \tag{9}$$

In that case, using (8), a field $\boldsymbol{z}$ with precision matrix approximately equal to $\boldsymbol{Q}$ can be simulated via the formula:

$$\boldsymbol{z} = \boldsymbol{D}^{-1} \mathrm{P}_{\text{-}1/2}^{(K)}(\boldsymbol{S}) \boldsymbol{\varepsilon} = \boldsymbol{D}^{-1} \sum_{k=0}^{K} \alpha_k \boldsymbol{S}^k \boldsymbol{\varepsilon} \tag{10}$$

Once the $\{\alpha_k\}$ are known, computing the simulated field using (10) can be done using an iterative algorithm that only requires matrix-vector products involving the sparse matrix $\boldsymbol{S}$.

To define the expression of a polynomial satisfying (9), it is sufficient to solve the following problem: *given an interval $[a, b]$ containing all the eigenvalues of $\boldsymbol{S}$ and a degree of approximation K, find a polynomial $\mathrm{P}_{\text{-}1/2}^{(K)}$ of degree K that approximates the (continuous) function $x \mapsto 1/\sqrt{\mathrm{P}(x)}$ over $[a, b]$*. Such an interval $[a, b]$ can be obtained *without having to diagonalize $\boldsymbol{S}$*. Examples of such intervals are provided in Appendix Appendix A.2. Using these results and the fact that $\boldsymbol{S}$ is positive semi-definite, the following interval is considered in the applications presented in this paper:

$$[a, b] = \Big[0, \max_{i \in [\![1,n]\!]} \sum_{j \in [\![1,n]\!]} |S_{ij}|\Big] \tag{11}$$

The approximation of the function $1/\sqrt{\mathrm{P}}$ over the interval $[a, b]$ is carried out using Chebyshev polynomials (Mason and Handscomb, 2002; Press et al., 2007).

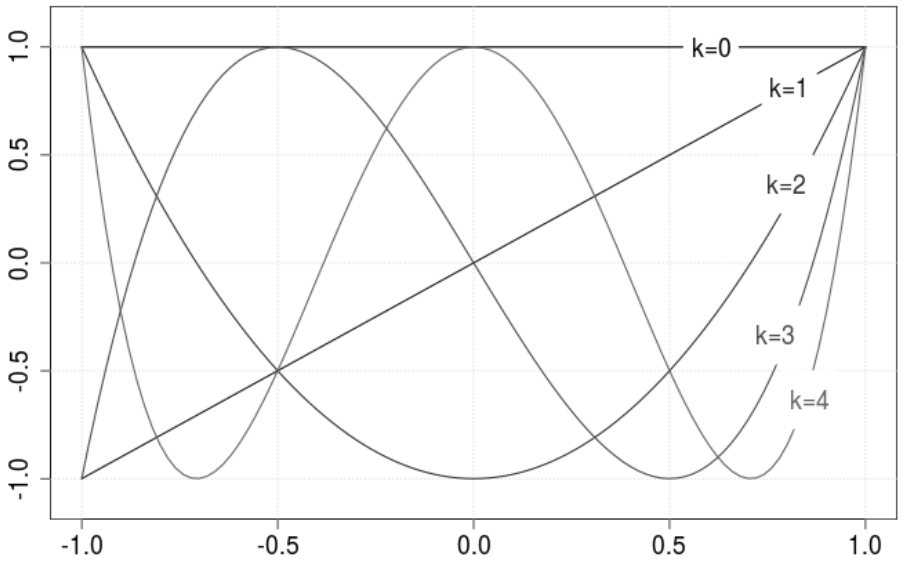


Fig. 1 : First 5 Chebyshev polynomials over $[-1, 1]$.

This family $(T_k)_{k\in\mathbb{N}}$ of polynomials is the sequence of polynomials defined over $[-1, 1]$ by:

$$\forall\theta \in \mathbb{R}, \quad T_k(\cos\theta) = \cos(k\theta)$$

or equivalently via the recurrence relation:

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x) \quad (k \geq 1)$$

A graphical representation of these polynomials is provided in Figure 1. Notice that they can be generalized to arbitrary intervals $[a, b] \subset \mathbb{R}$ via a simple change of variable:

$$y \in [a, b] \mapsto x = \frac{2y - b - a}{b - a} \in [-1, 1] \tag{12}$$

The choice of Chebyshev polynomials has several perks. For the sake of simplicity, they are listed in the case where a function f defined on $[-1, 1]$ has to be approximated.

- *Convergence* (Mason and Handscomb, 2002): If f is Lipschitz-continuous over $[-1, 1]$ (which is the case in the applications presented in this article), its Chebyshev series, defined as:

$$\forall x \in [-1, 1], \quad S(x) = \frac{1}{2}c_0T_0(x) + \sum_{k=1}^{\infty} c_kT_k(x) \tag{13}$$

  where $\forall k \in \mathbb{N}$:

$$c_k = \frac{2}{\pi}\int_{-1}^{1} f(x)T_k(x)\frac{1}{\sqrt{1 - x^2}}dx \tag{14}$$

is uniformly convergent on $[-1, 1]$. A similar result can be obtained for continuous functions when the Cesàro sums of their Chebyshev series are considered instead.

- *Near minimax property* (Mason and Handscomb, 2002; Press et al., 2007): Suppose that f is continuous. The minimax polynomial of degree K of f is the polynomial p_K^* defined as:

$$p_K^* = \underset{p \in \mathscr{P}_K}{\operatorname{argmin}} \|f - p\|_\infty$$

where $\mathscr{P}_K$ is the set of all polynomials of degree $\leq K$ and $\|.\|_\infty$ denotes the uniform norm (over $[-1, 1]$). It is generally very difficult to compute. However, the truncated Chebyshev series of degree K, denoted S_K, is a good approximation of p_K^* in the sense that $\|f - S_K\|_\infty$ is close to $\|f - p_K^*\|_\infty$.

- *Fast computation* (Press et al., 2007): A change of variable in (14) gives:

$$c_k = \frac{2}{\pi} \int_0^\pi f(\cos\theta) \cos(k\theta) d\theta \approx \sum_{j=0}^{J} f(\cos(j\frac{\pi}{J})) \cos(kj\frac{\pi}{J}) \qquad (15)$$

This last sum is the expression of the real part of the discrete Fourier Transform of the vector $\big(f(1), \ldots, f(\cos(j\frac{\pi}{J})), \ldots, f(-1)\big)^T$ for discretization order $J \in \mathbb{N}^*$. Hence the coefficients of the Chebyshev series of a function can be numerically computed using the Fast Fourier Transform algorithm, known for its speed and accuracy (Brigham, 1988).

4. Simulation algorithm

In this section, the workflow of the simulation algorithm is presented, then its complexity and the induced error are derived. Finally, criteria on the choice of the approximation order are given.

4.1. Presentation of the algorithm

Workflow: Simulation of a random vector using Chebyshev approximation

Require: A positive polynomial P, a real symmetric positive semi-definite $n \times n$ matrix $\boldsymbol{S}$ and an invertible diagonal matrix $\boldsymbol{D}$ of size n. An order of approximation $K \in \mathbb{N}$. A vector of n independent standard Gaussian components $\boldsymbol{\varepsilon}$.

Output: A vector $\boldsymbol{z}$ with precision matrix (approximately equal to) $\boldsymbol{Q} = \boldsymbol{D}\mathrm{P}(\boldsymbol{S})\boldsymbol{D}$

1. Find an interval $[a, b]$ containing all the eigenvalues of $\boldsymbol{S}$ (for instance (11)).
2. Compute a polynomial approximation $\mathrm{P}^{(K)}_{-1/2}$ of the function $x \mapsto 1/\sqrt{\mathrm{P}(x)}$ over $[a, b]$, by truncating its shifted Chebyshev series at order K:

$$\mathrm{P}^{(K)}_{-1/2}(x) = \frac{1}{2}c_0 T_0^{[a,b]}(x) + \sum_{k=1}^{K} c_k T_k^{[a,b]}(x), \quad T_k^{[a,b]}(x) := T_k\left(\frac{2}{b-a}x - \frac{b+a}{b-a}\right)$$

   The coefficients $(c_k)_{k\in[\![0,K]\!]}$ are computed by Fast Fourier Transform (using the changes of variables (12) in (15)).
3. Compute the product $\boldsymbol{u} = \mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})\boldsymbol{\varepsilon}$ using the recurrence relation satisfied by the Chebyshev polynomials.

$$\alpha := \frac{2}{b-a}; \quad \beta := \frac{b+a}{b-a}; \qquad k = 0;$$
$$\boldsymbol{u}^{(-2)} = \boldsymbol{\varepsilon}; \quad \boldsymbol{u} = \frac{1}{2}c_0\boldsymbol{u}^{(-2)}; \qquad k \leftarrow k+1;$$
$$\boldsymbol{u}^{(-1)} = \alpha\boldsymbol{S}\boldsymbol{\varepsilon} - \beta\boldsymbol{\varepsilon}; \quad \boldsymbol{u} \leftarrow \boldsymbol{u} + c_1\boldsymbol{u}^{(-1)}; \qquad k \leftarrow k+1;$$
$$\textbf{While}(k \le K)\{$$
$$\quad \boldsymbol{u}^{(0)} = \alpha\boldsymbol{S}\boldsymbol{u}^{(-1)} - \beta\boldsymbol{u}^{(-1)} - \boldsymbol{u}^{(-2)}; \quad \boldsymbol{u} \leftarrow \boldsymbol{u} + c_k\boldsymbol{u}^{(0)};$$
$$\quad \boldsymbol{u}^{(-2)} \leftarrow \boldsymbol{u}^{(-1)}; \quad \boldsymbol{u}^{(-1)} \leftarrow \boldsymbol{u}^{(0)}; \qquad k \leftarrow k+1;$$
$$\}$$
$$\textbf{Return } \boldsymbol{u}$$

4. The simulated field is given by: $\boldsymbol{z} = \boldsymbol{D}^{-1}\boldsymbol{u}$

4.2. Complexity of the algorithm

The complexity of the simulation algorithm can be explicitly calculated. Denote n_{nz} the number of non-zero entries of $\boldsymbol{S}$ and m_{nz} the mean number of non-zero entries of a row of $\boldsymbol{S}$: $n_{nz} = m_{nz} \times n$.
Denote K the order of the Chebyshev approximation. The cost associated with each step (ignoring additions and multiplications by non-stored zeros) is described as follows:

- Step 1 requires $\mathcal{O}(n_{nz})$ operations using (11) to compute the interval $[a, b]$.
- Step 2 requires to compute the Fast Fourier Transform of a vector of length K. The cost of this operation is $\mathcal{O}(K \log K)$.
- Step 3 requires to:
  - compute K products of matrix $\boldsymbol{S}$ and vectors $\rightarrow \mathcal{O}(Kn_{nz})$ operations
  - $2(K-1)+1$ subtractions of vectors, $3(K-1)+1$ multiplications of a vector by a scalar value and $(K-1)$ additions of vectors $\rightarrow (K-1)n$ operations.
- Step 4 requires n operations (product of a diagonal matrix and a vector).

Therefore, the overall cost of the simulation algorithm is $\mathcal{O}(Kn_{nz}) = \mathcal{O}(Km_{nz}n)$ operations.
And regarding the storage needs, aside from $\boldsymbol{S}$, $\boldsymbol{D}$ and $\boldsymbol{\varepsilon}$ which are assumed to be known (and therefore stored), the algorithm only needs to store 4 additional vectors of size n ($\boldsymbol{u}$, $\boldsymbol{u}^{(0)}$, $\boldsymbol{u}^{(-1)}$ and $\boldsymbol{u}^{(-2)}$).

4.3. Quantification of the numerical approximation error

The main idea of the simulation algorithm presented in this article is to replace the matrix $\boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1/2}$ in relation (8) by an efficient polynomial approximation, namely the matrix $\boldsymbol{D}^{-1}\mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})$. The numerical approximation error ϵ_{mat} between these matrices can be measured by :

$$\epsilon_{\mathrm{mat}} := \|\boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1/2} - \boldsymbol{D}^{-1}\mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})\|_\infty$$

where $\|.\|_\infty$ denotes the matrix max norm, defined by $\|\boldsymbol{A}\|_\infty := \max_{i,j} |A_{ij}|$. $\boldsymbol{D}$ being a diagonal matrix,

$$\epsilon_{\mathrm{mat}} \leq \|\boldsymbol{D}^{-1}\|_\infty \|\mathrm{P}(\boldsymbol{S})^{-1/2} - \mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})\|_\infty \leq \|\boldsymbol{D}^{-1}\|_\infty \|\mathrm{P}(\boldsymbol{S})^{-1/2} - \mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})\|_2$$

where $\|.\|_2$ denotes the Froebenius norm, defined by $\|\boldsymbol{A}\|_2 := \sqrt{\mathrm{Trace}(\boldsymbol{A}\boldsymbol{A}^T)}$. Therefore,

$$\epsilon_{\mathrm{mat}} \leq \|\boldsymbol{D}^{-1}\|_\infty \sum_{i=1}^{n} \left(\frac{1}{\sqrt{\mathrm{P}(\lambda_i)}} - \mathrm{P}^{(K)}_{-1/2}(\lambda_i) \right)^2 \leq n \|\boldsymbol{D}^{-1}\|_\infty \max_{x \in [a,b]} \left(\frac{1}{\sqrt{\mathrm{P}(x)}} - \mathrm{P}^{(K)}_{-1/2}(x) \right)^2 \tag{16}$$

Hence, the approximation error on the matrices is upper-bounded by the overall error that arises from the polynomial approximation of $x \mapsto 1/\sqrt{\mathrm{P}(x)}$ by its Chebyshev series. This last error can be made arbitrary small by truncating the polynomial series at a growing order. Therefore, the simulation algorithm is asymptotically exact given that asymptotically, the matrices $\boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1/2}$ and $\boldsymbol{D}^{-1}\mathrm{P}^{(K)}_{-1/2}(\boldsymbol{S})$ coincide.

4.4. Determination of the approximation order

A practical question still needs to be answered : how to choose the order of the approximation polynomial? A criterion could be based on relation (16), by imposing an order high enough so that the approximation error ϵ_{mat} of the matrices is below a given tolerance. But then, this tolerance would also need to be chosen. Given that the goal is actually to generate a random vector with the right covariance properties, a criterion based on the statistical properties of the random vectors produced by the proposed simulation algorithm, and not a numerical approximation error, seems more appropriate. In this section, such a criterion, based on the theory of statistical tests, is proposed.

4.4.1. Assessment of simulation validity through statistical tests

The aim is to simulate a Gaussian vector $\boldsymbol{z}$ with covariance matrix:

$$\boldsymbol{\Sigma} = \boldsymbol{Q}^{-1} = \boldsymbol{D}^{-1}\mathrm{P}(\boldsymbol{S})^{-1}\boldsymbol{D}^{-1}$$

But instead, the proposed simulation algorithm actually generates a Gaussian vector $\boldsymbol{z}_s$ with covariance matrix:

$$\boldsymbol{\Sigma}_s = \boldsymbol{D}^{-1}\mathrm{P}_{\text{approx}}(\boldsymbol{S})^2\boldsymbol{D}^{-1}$$

Where $\mathrm{P}_{\text{approx}}$ is a polynomial approximating the function $x \mapsto 1/\sqrt{\mathrm{P}(x)}$ over an interval containing all the eigenvalues of $\boldsymbol{S}$.

Consider then a sample of N independent zero-mean Gaussian vectors $\left(\boldsymbol{z}_s^{(1)}, \ldots, \boldsymbol{z}_s^{(N)}\right)$ with covariance matrix $\boldsymbol{\Sigma}_s$. Let's consider the following null hypothesis test:

$$H_0 : \left(\boldsymbol{z}_s^{(1)}, \ldots, \boldsymbol{z}_s^{(N)}\right) \text{ is a sample of zero-mean Gaussian vectors with covariance matrix } \boldsymbol{\Sigma}$$

Obviously, the condition on the mean is satisfied by construction of this sample. Besides, by definition (Tong, 2012), a random vector $\boldsymbol{z}$ is a Gaussian vector with covariance matrix $\boldsymbol{\Sigma}$ if and only if, for any $\boldsymbol{v} \in \mathbb{R}^n$, $\boldsymbol{v}^T\boldsymbol{z}$ is a Gaussian variable with variance $\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}$. Therefore, hypothesis H_0 won't be rejected if $\forall \boldsymbol{v} \in \mathbb{R}^n$, the hypothesis H_0^v defined by:

$$H_0^v : \left(\boldsymbol{v}^T\boldsymbol{z}_s^{(1)}, \ldots, \boldsymbol{v}^T\boldsymbol{z}_s^{(N)}\right) \text{ is a sample of zero-mean Gaussian variables with variance } \boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}^T$$

is not rejected.

Two-sided chi-square tests for the variance (Snedecor and Cochran, 1989) are considered. The results of these tests can actually be anticipated given that by definition, the sample $\left(\boldsymbol{v}^T\boldsymbol{z}_s^{(1)}, \ldots, \boldsymbol{v}^T\boldsymbol{z}_s^{(N)}\right)$ has a known distribution: it is Gaussian with variance $\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}$. In particular, a criterion on the quality of the polynomial approximation such that *for any* $\boldsymbol{v} \in \mathbb{R}^n$ the probability of rejecting hypothesis H_0^v can be controlled is derived.

Proposition 4.1. *Let $[a,b]$ be an interval containing all the eigenvalues of $\boldsymbol{S}$. Let ϵ_{pol} denote the polynomial approximation error defined by:*

$$\epsilon_{\text{pol}} := \max_{\lambda\in[a,b]} \left| \frac{1/\mathrm{P}(\lambda) - \mathrm{P}_{\text{approx}}(\lambda)^2}{\mathrm{P}_{\text{approx}}(\lambda)^2} \right| \tag{17}$$

Then $\forall\gamma > 0$, there exists $\epsilon_{N,\gamma} > 0$ such that:

$$\epsilon_{\text{pol}} \leq \epsilon_{N,\gamma} \Rightarrow \forall \boldsymbol{v} \in \mathbb{R}^n, \quad R_\alpha(\boldsymbol{v}) \leq (1+\gamma)\alpha \tag{18}$$

where $R_\alpha(\boldsymbol{v})$ is the probability of rejecting hypothesis H_0^v in a chi-square test for the variance with significance α (involving N samples).

Proof. See Appendix Appendix B. □

Therefore, if (18) is satisfied, then, for any $\boldsymbol{v}$, hypothesis H_0^v is actually rejected (with significance α) with a probability less than $(1+\gamma)\alpha$. This probability would have been equal to α if the samples were generated using the right covariance matrix. Therefore, the parameter γ represents relative increase of the rejection probability due to the fact that the samples are generated using $\boldsymbol{\Sigma}_s$ instead of $\boldsymbol{\Sigma}$.

As detailed in Appendix Appendix B, the bound $\epsilon_{N,\gamma}$ on the polynomial approximation error ϵ_{pol} can be numerically computed with the sole specification of the charcteristics of the statitical test (a sample size N and a significance level α) and tolerated error in the variance γ. In particular, it depends neither on the polynomial of approximation nor on the approximated function itself. 1 and 2 give typical values of the tolerance $\epsilon_{N,\gamma}$ for various sample sizes N and thresholds γ. The significance is fixed at $\alpha = 0.05$ for Table 1 and $\alpha = 0.01$ for Table 2.

Notice now that in the case of our Chebyshev simulation algorithm, the polynomial $\mathrm{P}_{\text{approx}}$ is defined as the truncation of a Chebyshev series at an order K. This order can be determined by specifying the characteristics of the statistical test the user would want its simulations to pass, along with a tolerated error in variance, which in turn would yield a value of $\epsilon_{N,\gamma}$ and therefore set a bound for the polynomial approximation error ϵ_{pol}. The order of truncation K is then chosen so that $\epsilon_{\text{pol}} \leq \epsilon_{N,\gamma}$.

4.4.2. Efficiency improvement

The previous subsection provides a link between the order of the polynomial approximation of $1/\sqrt{P}$ and the validity of resulting simulations using the proposed algorithm. In practice, this order can be reduced in some cases.

Following the notations of section 4, let $(c_k)_{k\in\mathbb{N}}$ be the coefficients of the Chebyshev series of $1/\sqrt{P}$ over an interval $[a,b]$ containing all the eigenvalues of $\boldsymbol{S}$ and for $m \in \mathbb{N}^*$, let $\boldsymbol{z}^{(m)}$ be the vector defined by:

$$\boldsymbol{z}^{(m)} = \boldsymbol{D}^{-1}\left(\frac{1}{2}c_0 T_0^{[a,b]}(\boldsymbol{S})\boldsymbol{\varepsilon} + \sum_{k=1}^{m} c_k T_k^{[a,b]}(\boldsymbol{S})\boldsymbol{\varepsilon}\right)$$

where $\boldsymbol{\varepsilon}$ is a vector with independent standard Gaussian values. Then, for

γ	Sample size N					
	50	100	500	1000	5000	10000
0.1%	6.40e-04	6.20e-04	5.40e-04	4.80e-04	3.00e-04	2.40e-04
1%	5.44e-03	4.80e-03	3.04e-03	2.36e-03	1.20e-03	8.60e-04
5%	1.89e-02	1.51e-02	8.06e-03	5.94e-03	2.82e-03	2.02e-03
10%	3.00e-02	2.33e-02	1.18e-02	8.64e-03	4.02e-03	2.88e-03
20%	4.59e-02	3.48e-02	1.71e-02	1.24e-02	5.74e-03	4.08e-03
50%	7.66e-02	5.71e-02	2.75e-02	1.98e-02	9.08e-03	6.46e-03
100%	1.10e-01	8.12e-02	3.89e-02	2.80e-02	1.28e-02	9.10e-03

Table 1: Values of the precision threshold $\epsilon_{N,\gamma}$ for different values of sample size N and of degradation of the type I error γ. The significance of the test is $\alpha = 0.05$

γ	Sample size N					
	50	100	500	1000	5000	10000
0.1%	4.00e-04	4.00e-04	3.60e-04	3.20e-04	2.20e-04	1.80e-04
1%	3.56e-03	3.24e-03	2.20e-03	1.74e-03	9.20e-04	6.60e-04
5%	1.33e-02	1.09e-02	6.06e-03	4.52e-03	2.18e-03	1.56e-03
10%	2.16e-02	1.71e-02	9.00e-03	6.62e-03	3.12e-03	2.24e-03
20%	3.36e-02	2.59e-02	1.31e-02	9.54e-03	4.44e-03	3.18e-03
50%	5.67e-02	4.28e-02	2.10e-02	1.52e-02	7.00e-03	5.00e-03
100%	8.11e-02	6.07e-02	2.94e-02	2.12e-02	9.76e-03	6.96e-03

Table 2: Values of the precision threshold $\epsilon_{N,\gamma}$ for different values of sample size N and of degradation of the type I error γ. The significance of the test is $\alpha = 0.01$

$m, l \in \mathbb{N}^*$:

$$\|\boldsymbol{z}^{(m+l)} - \boldsymbol{z}^{(m)}\| = \left\| \boldsymbol{D}^{-1} \sum_{k=1}^{l} c_{m+k} T_k^{[a,b]}(\boldsymbol{S})\boldsymbol{\varepsilon} \right\| \leq \|\boldsymbol{D}^{-1}\|_\infty \sum_{k=1}^{l} |c_{m+k}| \left\| T_k^{[a,b]}(\boldsymbol{S})\boldsymbol{\varepsilon} \right\|$$

where $\|.\|$ denotes the Euclidean norm on $\mathbb{R}^n$.
Recall that, according to the properties of Rayleigh quotients (see Appendix Appendix A.1), and using the symmetry of $T_k^{[a,b]}(\boldsymbol{S})$:

$$\frac{\left\| T_k^{[a,b]}(\boldsymbol{S})\boldsymbol{\varepsilon} \right\|^2}{\|\boldsymbol{\varepsilon}\|^2} = \frac{\boldsymbol{\varepsilon}^T T_k^{[a,b]}(\boldsymbol{S})^2 \boldsymbol{\varepsilon}}{\boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon}} \leq \lambda_{\max}\left(T_k^{[a,b]}(\boldsymbol{S})^2 \right)$$

where $\lambda_{\max}(.)$ denotes the largest eigenvalue of a matrix. Notice then that, given that the Chebyshev polynomials are upper-bounded (in absolute value) by 1, $\lambda_{\max}\left(T_k^{[a,b]}(\boldsymbol{S})^2 \right) \leq 1$. Consequently,

$$\|\boldsymbol{z}^{(m+l)} - \boldsymbol{z}^{(m)}\| \leq \|\boldsymbol{D}^{-1}\|_\infty \sum_{k=1}^{l} |c_{m+k}| \, \|\boldsymbol{\varepsilon}\|$$

Hence, $\|\boldsymbol{D}^{-1}\|_\infty \|\boldsymbol{\varepsilon}\| \sum_{k=1}^{l} |c_{m+k}| \approx 0 \Rightarrow \boldsymbol{z}^{(m+l)} \approx \boldsymbol{z}^{(m)}$. This gives an additional criterion for the choice of the approximation order of the algorithm. After computing a value of order L using the criterion based on statistical tests, its value can be decreased to K as long as:

$$\sum_{k=K+1}^{L} |c_k| \leq \frac{\eta}{\|\boldsymbol{D}^{-1}\|_\infty \|\boldsymbol{\varepsilon}\|}$$

for a fixed tolerance η corresponding to the Euclidean distance between the vector computed using order K and the one computed using order L. In particular, if η is of the form $\eta = \sqrt{\epsilon n}$ (where n is the size of the simulated vectors) and $\epsilon > 0$, then in average, the square of the components of the vector $(\boldsymbol{z}^{(L)} - \boldsymbol{z}^{(K)})$ will be less than ϵ.

4.5. Relation to Krylov subspaces methods

Krylov subspaces provide a framework for the study of some of the most used iterative algorithms used to solve eigenvalue problems and linear systems involving a matrix $\boldsymbol{A}$. The idea behind such algorithms is to iteratively generate a sequence of approximate solutions of the problem while relying at each iteration on recurrence relations based on matrix-vector products involving $\boldsymbol{A}$. The approximate solution obtained at the m-th iteration step lies in the subspace $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{v})$ defined by:

$$\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{v}) = \text{span}\{\boldsymbol{v}, \boldsymbol{A}\boldsymbol{v}, \ldots, \boldsymbol{A}^{m-1}\boldsymbol{v}\} = \{\pi(\boldsymbol{A})\boldsymbol{v} : \pi \text{ polynomial of degree } < m\}$$

In this section, the relation between our simulation algorithm and Krylov subspaces is exposed, and a comparison with a more standard Krylov subspaces approach to generate samples from a GMRF with known precision matrix is presented.

Section 2.2 provides a direct way to generate samples from a precision matrix satisfying (3). Indeed, from equation (8), the vector $\boldsymbol{z} = \boldsymbol{D}^{-1}\boldsymbol{u}$ where:

$$\boldsymbol{u} = \mathrm{P}(\boldsymbol{S})^{-1/2}\boldsymbol{\varepsilon} \tag{19}$$

and $\boldsymbol{\varepsilon}$ is a vector of independent standard Gaussian variables, is a zero-mean GMRF with precision matrix $\boldsymbol{Q}$ given by (3). The algorithm presented in this paper actually consists in replacing $\boldsymbol{u}$ by an approximation $\boldsymbol{u}_C^{(K)}$ given by:

$$\boldsymbol{u}_C^{(K)} = \mathrm{P}_{-1/2}^{(K)}(\boldsymbol{S})\boldsymbol{\varepsilon} \tag{20}$$

where $\mathrm{P}_{-1/2}^{(K)}$ is a polynomial of degree K defined as the truncation at order K of the Chebyshev series of the function $x \mapsto 1/\sqrt{P(x)}$ on an interval containing the eigenvalues of $\boldsymbol{S}$. In particular, $\boldsymbol{u}_C^{(K)} \in \mathcal{K}_{K+1}(\boldsymbol{S}, \boldsymbol{\varepsilon})$ and our algorithm can be seen as an iterative algorithm on the truncation order K. This justifies the fact that it can be considered as a Krylov subspace approach.

A standard approach using Krylov subspaces to generate samples from a GMRF with known precision matrix uses the Lanczos algorithm to come up with an approximation of $\boldsymbol{u}$ (Simpson et al., 2008). Indeed, in exact arithmetic,

this algorithm can provide an orthonormal basis of $\mathcal{K}_{K+1}(\boldsymbol{S}, \boldsymbol{\varepsilon})$ (Golub and Van Loan, 1996). $\boldsymbol{u}$ can then be approximated by (Frommer and Simoncini, 2008; Simpson et al., 2008):

$$\boldsymbol{u}_L^{(K)} = \|\boldsymbol{\varepsilon}\| \boldsymbol{V}_{K+1} P(\boldsymbol{T}_{K+1})^{-1/2} \boldsymbol{e}_1 \tag{21}$$

where $\boldsymbol{e}_1 = (1\ 0\ \ldots\ 0)^T$, $\boldsymbol{T}_{K+1}$ is a tridiagonal (symmetric) matrix of size $K+1$ and $\boldsymbol{V}_{K+1}$ is a matrix containing the $K+1$ vectors of the orthonormal basis of $\mathcal{K}_{K+1}(\boldsymbol{S}, \boldsymbol{\varepsilon})$, both matrices being products of the Lanczos algorithm.

This cost associated with computing $\boldsymbol{u}_L^{(K)}$ can be decomposed as follows :

- running the Lanczos algorithm for K iterations : $\mathcal{O}(K m_{nz} n)$ operations, where m_{nz} is the mean number of non-zero values in a row of $\boldsymbol{S}$.
- computing (21), which involves the diagonalisation of $\boldsymbol{T}_{K+1}$ and a matrix-vector product with $\boldsymbol{V}_{K+1}$: $\mathcal{O}((K+1)^2 + nK)$ operations.

Computing $\boldsymbol{u}_L^{(K)}$ therefore comes at an overall cost of $\mathcal{O}(K m_{nz} n + K^2)$ operations. Regarding the storage needs of this process, the matrix $\boldsymbol{V}_{K+1}$ and the eigendecomposition of $\boldsymbol{T}_{K+1}$ need to be stored, which requires a storage need of $\mathcal{O}(Kn + K^2)$.

From Section 4.2, it is clear that our Chebyshev approximation algorithm requires less operations and storage space to generate an approximation of $\boldsymbol{u}$ from the same Krylov subspace. But on the other hand, at the same approximation order K, the quality of the approximation obtained using the Lanczos algorithm will be better than the one using our Chebyshev algorithm. Indeed, in the Lanczos case (still in exact arithmetic) this approximation error satisfies (Musco et al., 2017):

$$\|\boldsymbol{u} - \boldsymbol{u}_L^{(K)}\| \leq 2\|\boldsymbol{\varepsilon}\| \delta_K, \quad \delta_K = \min_{\substack{\pi \text{ polynomial} \\ \text{of degree } \leq K}} \max_{x \in [\lambda_{\min}, \lambda_{\max}]} |1/\sqrt{P(x)} - \pi(x)|$$

Where $\lambda_{\min}$ (resp. $\lambda_{\max}$) denotes the smallest (resp. largest) eigenvalue of $\boldsymbol{S}$. Thus it yields in the Lanczos case an error of order $\mathcal{O}(\delta_K)$. And in the Chebyshev case, it is given by:

$$\|\boldsymbol{u} - \boldsymbol{u}_C^{(K)}\| \leq \|\mathrm{P}^{-1/2}(\boldsymbol{S}) - \mathrm{P}_{\text{-}1/2}^{(K)}(\boldsymbol{S})\| \|\boldsymbol{\varepsilon}\| \leq \|\boldsymbol{\varepsilon}\| \max_{x \in [\lambda_{\min}, \lambda_{\max}]} |1/\sqrt{P(x)} - \mathrm{P}_{\text{-}1/2}^{(K)}(x)|$$

This last estimate can be bounded using δ_K and the Lebesgue constant λ_K, thus giving for the Chebyshev approximation an error of order $\mathcal{O}(\lambda_K \delta_K) = \mathcal{O}(\delta_K \log K)$ (Mason and Handscomb, 2002). The results of the comparison between the Lanczos algorithm and our Chebyshev algorithm are summed up in Table 3.

For small values of K the Lanczos algorithms is more adequate as it provides an approximation with a lower error. Its main flaw resides in the fact that, contrary to our Chebyshev algorithm, the storage needs grow linearly with the order of approximation. Hence for large problems (i.e. when n is large), a restriction on the order of approximation has to be set according to the storage space available to the user. In order to tackle this storage problem, some adjustments can be made to the original Lanczos algorithm (Aune et al., 2013). For instance,

	Lanczos	Chebyshev
Computational cost	$\mathcal{O}(Km_{nz}n + K^2)$	$\mathcal{O}(Km_{nz}n)$
Storage needs	$\mathcal{O}(Kn + K^2)$	$\mathcal{O}(n)$
Approximation error of $\boldsymbol{u} = \boldsymbol{D}\boldsymbol{z}$	$\mathcal{O}(\delta_K)$	$\mathcal{O}(\delta_K \log K)$

Table 3: Comparison between the Lanczos algorithm and our Chebyshev algorithm after K iterations, for the simulation of a sample from a GMRF $\boldsymbol{z}$ with precision matrix (3).

restarting procedures allow to work with a fixed number of stored basis vectors of the Krylov space. However, those methods result in a loss of approximation quality and push to use complex preconditioning techniques in order to improve the convergence speed of the algorithm , which in turn increases the overall computational cost (Simpson et al., 2008). Our Chebyshev algorithm doesn't share this storage flaw, allowing it to make up for its relative lack of precision by the possibility to work with much higher orders of approximation without the headache of finding the right variation of Lanczos algorithm [1] .

Another attractive feature of the Chebyshev algorithm is the statistical stopping criterion derived in Section 4.4.1. This criterion was established by using the fact that $\boldsymbol{u}_C^{(K)}$ could be written as $\boldsymbol{u}_C^{(K)} = \pi_K(\boldsymbol{S})\boldsymbol{\varepsilon}$ where the coefficients defining π_K are deterministic (which in our case means that they are not linked to $\boldsymbol{\varepsilon}$) and that therefore $\boldsymbol{u}_C^{(K)}$ is a Gaussian vector with known covariance. This is no longer the case when considering the Lanczos algorithm given that these same coefficients would effectively depend on the entries of $\boldsymbol{\varepsilon}$ given that this vector is used to compute the matrices $\boldsymbol{V}_{K+1}$ and $\boldsymbol{T}_{K+1}$ used to define $\boldsymbol{u}_L^{(K)}$. The only available stopping criteria for the Lanczos algorithm are therefore linked to the actual approximation error $\|\boldsymbol{u} - \boldsymbol{u}_L^{(K)}\|$ and not the statistical properties of the vector we wish to simulate. Moreover, given that in practice $\boldsymbol{u}$ is not available, the stopping criteria actually rely on the link between the Lanczos algorithm and the Conjugate Gradient algorithm, using the residuals of the latter as a bound on the approximation error (Aune et al., 2013).

5. Application

5.1. Simulation of stationary Matérn models

The Matérn model is a widely used covariance model in Geostatistics due to its great flexibility. For a lag distance $h \in \mathbb{R}_+$, its isotropic formulation is

[1] Note also that the comparison is carried out under the assumption of exact arithmetic. In floating points computations, a loss of orthogonality of $\boldsymbol{V}_{K+1}$ is observed as K grows, leading to larger approximation errors (Musco et al., 2017) and forcing the user to adapt is algorithm using workarounds such as re-orthogonalisation techniques or restart techniques (thus increasing the overall complexity of the algorithm).

(Chilès and Delfiner, 2012):

$$C(h) = \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)} \left(\frac{h}{\phi}\right)^{\nu} K_\nu \left(\frac{h}{\phi}\right)$$

where $\sigma^2 > 0$ is the marginal variance, $\phi > 0$ is a scaling parameter, $\nu > 0$ is a shape parameter and K_ν is the modified Bessel function of the second kind of order ν. The parameter ν can be seen as a "smoothness" parameter as the underlying process is $\lfloor \nu \rfloor$-time mean-square differentiable.

Lindgren et al. (2011) define Markovian approximations GRF with Matérn covariance as numerical solutions of SPDE (2) with parameters:

$$\kappa = 1/\phi, \quad \alpha = \nu + d/2 \in \mathbb{N}, \quad \tau = \sigma\kappa^\nu \sqrt{(4\pi)^{d/2}\Gamma(\nu + d/2)/\Gamma(\nu)}$$

using the finite element method with linear triangular elements. They show that the precision matrix of the weights of the finite element representation of the solution can be written as (3), with:

$$\boldsymbol{S} = (1/\kappa^2)\boldsymbol{C}^{-1/2}\boldsymbol{G}\boldsymbol{C}^{-1/2}, \quad \boldsymbol{D} = (\kappa^\alpha/\tau)\boldsymbol{C}^{1/2}, \quad \mathrm{P} : x \mapsto (1+x)^\alpha \tag{22}$$

where:

$$\boldsymbol{C} = [\langle \psi_i, \psi_j \rangle], \quad \boldsymbol{G} = [\langle \nabla\psi_i, \nabla\psi_j \rangle], \quad \boldsymbol{C}^{-1/2} = (\boldsymbol{C}^{1/2})^{-1} \tag{23}$$

and $\boldsymbol{C}^{1/2}$ is a square-root of $\boldsymbol{C}$. The matrices $\boldsymbol{C}$ and $\boldsymbol{G}$ are sparse. Following again the results from Lindgren et al. (2011), $\boldsymbol{C}$ is replaced by a diagonal matrix with entries $[\langle \psi_i, 1 \rangle]$, which yields a Markov approximation of the solution that has the same convergence rate as the full finite element method formulation.

In this subsection, simulations of the Matérn fields are generated using two methods (for comparison purposes): the Cholesky factorisation algorithm presented in Section 2.1, and the simulation algorithm proposed in this paper and presented in Section 4.

When applied to the simulation of random fields using the finite element method, the proposed algorithm can actually be interpreted as an image convolution algorithm. To see that, notice that when defined as in (22), the matrix $\boldsymbol{S}$ can be interpreted as what is referred to as a shift operator in the GSP theory (Shuman et al., 2013). Namely, when (linear) triangular elements are used, $\boldsymbol{S}$ is a sparse matrix whose entry M_{ij} is non-zero if the nodes i and j of the triangulation are adjacent or if they coincide. Therefore, the non-zeros entries of a polynomial of degree K of $\boldsymbol{S}$ correspond to nodes that are within a distance K from each other in the triangulation (i.e. there exist a chain of at most K adjacent nodes between them).

Now, according to the workflow provided in Section 4, the output vector $\boldsymbol{z}$ of the simulation algorithm can be written as $\boldsymbol{z} = \boldsymbol{D}^{-1}\mathrm{P}_{-1/2}(\boldsymbol{S})\boldsymbol{\varepsilon}$, where $\boldsymbol{D}$ is a diagonal matrix, $\boldsymbol{\varepsilon}$ is a vector indexed by the nodes of the triangulation and $\mathrm{P}_{-1/2}$ is a polynomial of degree K. So, the i-th entry of $\boldsymbol{z}$ is given by a linear combination of the entries of $\boldsymbol{\varepsilon}$ that correspond to nodes within a distance K of i, and weighted by the entries of i-th row of $\boldsymbol{D}^{-1}\mathrm{P}_{-1/2}(\boldsymbol{S})$. Hence, $\boldsymbol{z}$ can be seen as a convolution of the entries $\boldsymbol{\varepsilon}$. The bigger the degree of the polynomial, the bigger the size of the convolution kernel.

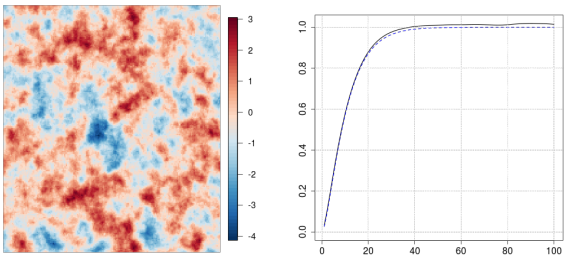


Fig. 2 : *(Left)* Simulations of a Matérn field on a 200x200 grid using Cholesky factorisation. *(Right)* Mean variograms over 50 simulations (solid line) and model (dotted line) : range=25, sill=1, smoothness=1.

5.1.1. Order of the polynomial approximation

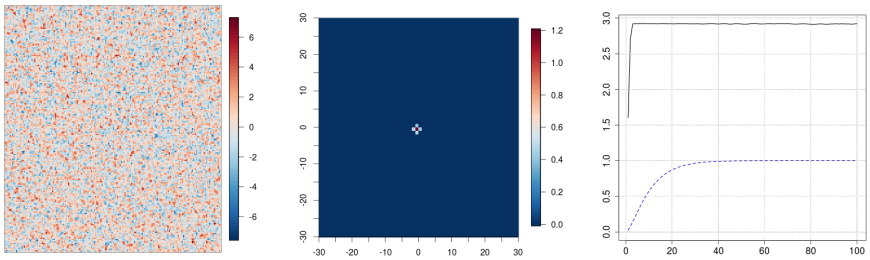
First, the effect of the order of the polynomial approximation on the resulting simulation is investigated. To do so, simulations of a Matérn field are generated on a 200x200 grid, with range 25, sill 1 and smoothness parameter 1, and with a growing order. In Figure 3, simulations obtained for degree values of 1, 5, 20 and 100 and the associated variogram (averaged over 50 simulations) are displayed. As a comparison, the same model simulated using the classical Cholesky factorisation algorithm is displayed in Figure 2.

As noticed in Figure 3, increasing the order of the polynomial tends to add smoothness and structure to the simulation. This is expected from a convolution algorithm as the size of the kernel, which is directly linked to the order of the polynomial, grows (center images in Figure 3). Moreover, there seems to be a point from which adding more polynomials doesn't change the simulation. This observation is just a consequence of the result presented in Subsection 4.4.2.

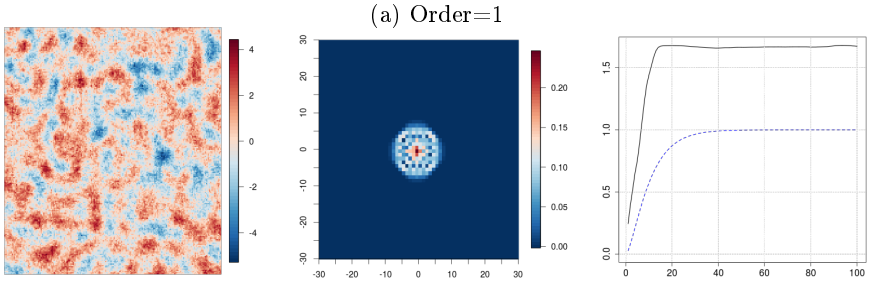
Remark also that the variogram of the simulations in Figure 3 tends to be respected as the order of the polynomial grows. This fact was predictable and is due to the fact that the proposed algorithm ensures that any linear combinations of the vectors generated by the algorithm have the right variance within a given tolerance (see subsection 4.4.1). Consequently, this will ensure that the variogram is respected given that its value at particular lag h is just the variance of the difference between two particular entries of the simulated vector that correspond to nodes of the triangulation separated by an Euclidean distance of h.

5.1.2. Influence of the model

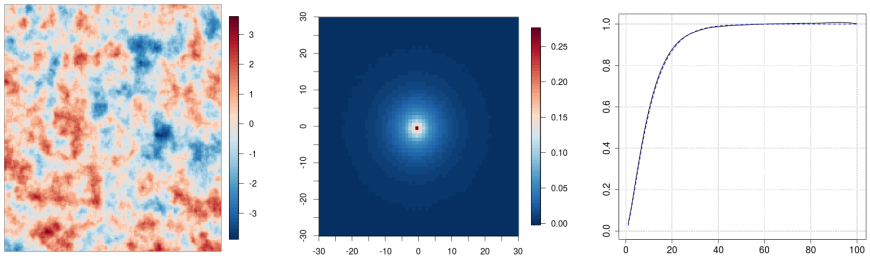
The influence of the covariance model parameters on the resulting approximations is now investigated. To do so, simulations of Matérn fields with different values of range and smoothness parameters are generated (cf. Figure 4). For each set of parameters, the order of approximation is set so that the probability of rejection on the statistical tests with significance $\alpha = 0.05$ is equal to $(1+10\%)\alpha$, which corresponds to a threshold on the approximation error (18) of 3.0e-02 (cf. Table 1). Following the result of subsection 4.4.2, the effective order



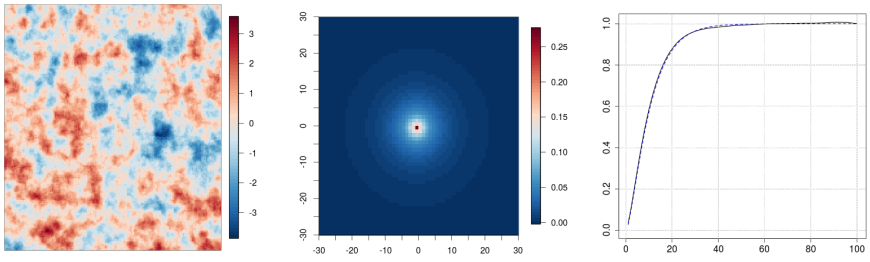
(a) Order=1



(b) Order=10

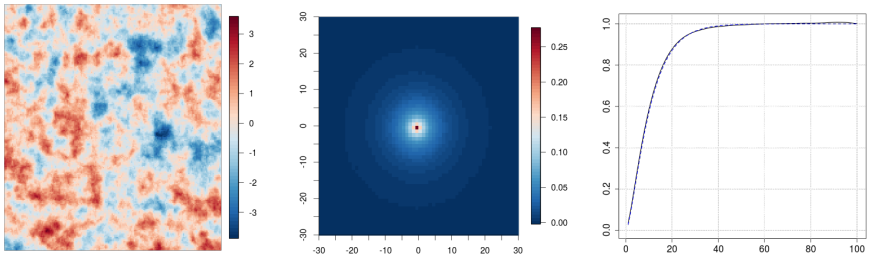


(c) Order=50

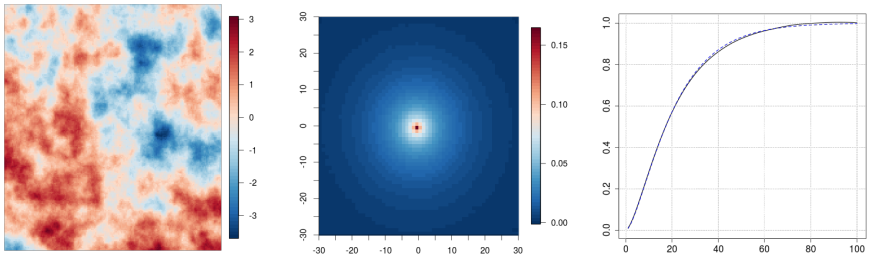


(d) Order=100

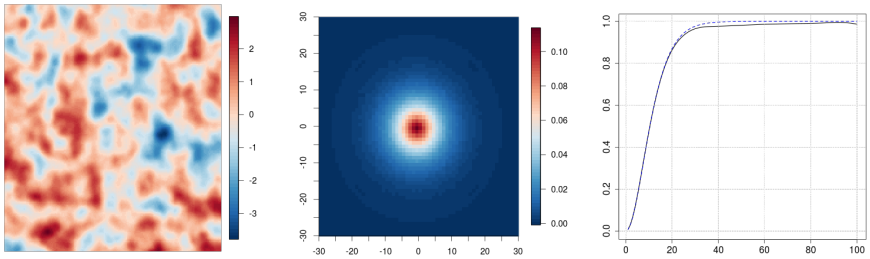
Fig. 3 : *(Left)* Simulations of a Matérn model with range 25, sill 1 and smoothness parameter 1 on a 200x200 grid using Chebyshev approximation with growing order. *(Center)* Convolution kernels associated with the simulation. *(Right)* Mean variograms over 50 simulations (solid line) and model (dotted line).



(a) Range = 25, Smoothness = 1 : Order = 76 (Effective order = 52)



(b) Range = 50, Smoothness = 1 : Order = 166 (Effective order = 102)



(c) Range = 25, Smoothness = 3 : Order = 84 (Effective order = 40)

Fig. 4 : *(Left)* Simulations using Chebyshev approximation of a Matérn field on a 200x200 grid with various model parameters. *(Center)* Convolution kernels associated with the simulation. *(Right)* Mean variograms over 50 simulations (solid line) and model (dotted line).

of approximation used to generate the simulation is reduced by considering a tolerance $\eta = \sqrt{10^{-4}n}$ where $n = 200^2$ is the size of the simulated vector.

The order of approximation used for each simulation is reported in Figure 4. It can be noticed that increasing the range results in significantly higher orders of approximation to achieve the same accuracy, whereas the effect of the smoothness parameters seems more limited. This is just another consequence of the "convolution" nature of the algorithm, as explained at the beginning of the section. The bigger the range is, the bigger the size of the kernel used to generate the simulation from a white noise image should be as bigger "spots" must be created, and therefore the bigger the order of approximation is. On the other hand, the smoothness parameter mainly affects the smoothness of the kernel, not its size.

5.2. Simulation of non-stationary fields

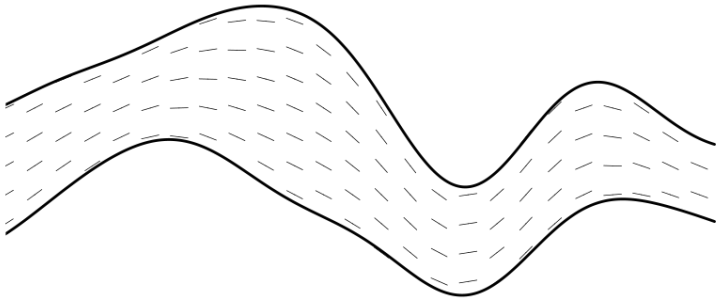
Following the modeling approach of Fuglstad et al. (2015), an expression for the precision matrix of a finite element solution of the SPDE (2) with locally varying coefficients is derived, resulting in a non-stationary field with local anisotropies. Specifically, let's consider a numerical solution of the SPDE:

$$\kappa^2 Z(x) - \operatorname{div}\left(H(x)\nabla Z(x)\right) = \tau \mathcal{W}(x)$$

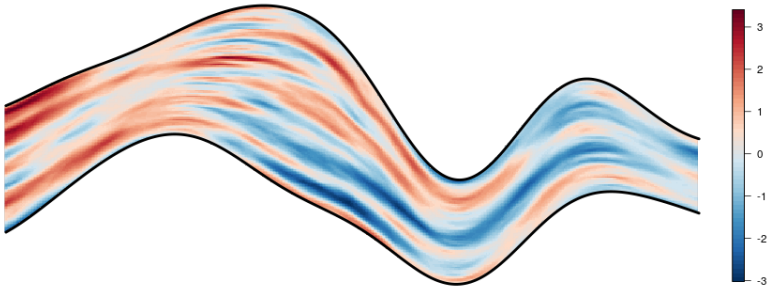
where $\kappa, \tau > 0$, $\mathcal{W}$ is a spatial Gaussian white noise and H is a field of positive definite matrices indexed by the space. Using weak formulations of this SPDE, it can be shown that the precision matrix of the weights has the same expression as in the stationary case presented in the previous subsection (with $\alpha = 2$) but where the matrix $\boldsymbol{G}$ is now defined by:

$$\boldsymbol{G} = [\langle \nabla\psi_i, H\nabla\psi_j \rangle]$$

Notice that the matrix $\boldsymbol{G}$ contains all the "anisotropy" parameters of the model in the sense that, similarly as in (Fuglstad et al., 2015), its elements are constructed by locally accounting for the varying coefficients of the SPDE in expressions (23).



(a) Anisotropy field : each line represents the local direction of the anisotropy. The anisotropy ratio is locally proportional to the thickness of the layer (maximum value : 1.5).



(b) Simulation.

Fig. 5 : Simulation using Chebyshev approximation of a non-stationary Matérn field with local anisotropy defined on a layer. Overall extension of the domain : 500x200. Model : range=150, sill=1.

Just like in Subsection 5.1, the Chebyshev approximation simulation algorithm to generate realizations of solutions of the SPDE. The results are displayed in Figure 5.

6. Conclusion

In this paper, an algorithm for simulating Gaussian random vectors whose precision matrix can be expressed as a polynomial P of a sparse matrix $\boldsymbol{S}$ is proposed. This case arises in particular when simulation is carried out by solving numerically some SPDEs using finite element method. In doing so, random fields with a Matérn covariance can be generated. But this approach can also be used to simulate non-stationary random fields arising when SPDEs with varying coefficients are considered (Fuglstad et al., 2015).

The proposed algorithm is based on a computationally efficient polynomial approximation of a square-root of the covariance matrix, using a Chebyshev polynomial approximation of the function $1/\sqrt{\mathrm{P}}$, and can generate random vectors in $\mathcal{O}(Kn_{nz})$ operations, where K is the order of approximation and n_{nz} is the number of non-zeros of $\boldsymbol{S}$. The error of approximation on $1/\sqrt{\mathrm{P}}$ was proved to be directly linked to the statistical properties of the random vectors generated by the algorithm, namely the variance of linear combinations of their components, providing a criterion on the level of approximation for the simulated vectors to satisfy the right properties within a given tolerance. Due to its low computational complexity and memory requirements, this algorithm can be used to generate large random vectors, arising for instance from the discretization of the solutions of SPDEs in 3 dimensions, even with large smoothness parameters. This approach is implemented in the R package RGeostats (Renard et al., 2018).

The proposed algorithm is a tool particularly well suited for the simulation of random fields represented as numerical solutions of SPDEs using finite element methods, and in particular Matérn fields and their extensions to manifolds, and oscillating and non-stationary covariances as presented in (Lindgren et al., 2011). Although only the case $\alpha \in \mathbb{N}$ (in SPDE (2)) has been examined, the generalization to non-integer values of α is quite straightforward, as the resulting fields can be approximated by a linear combinations of fields with $\alpha \in \mathbb{N}$ (Lindgren et al., 2011). The main limitation seems rather to come from the finite element method itself which for small values of $\nu = \alpha - d/2$ requires thinner and thinner meshes and hence, bigger and bigger matrix sizes.

Finally, the proposed algorithm could be applied to the simulation of more general random fields than Matérn fields. Indeed, the function $1/\sqrt{\mathrm{P}}$ approximated by the algorithm and used to generate the simulations, is proportional, for Matérn fields, to the actual square-root of the spectral density of the field. A way of generalizing the algorithm, which is under study and shows promising results, is to generate simulations of general Gaussian isotropic random fields using the same finite elements matrices as the one used here, but by replacing the approximation of the function $1/\sqrt{\mathrm{P}}$ by the approximation of the square-root of the spectral density of the targeted field.

Appendix A. Some elements of matrix analysis

Let $\boldsymbol{n} \in \mathbb{N}^*$. Let $\boldsymbol{M} = (M_{ij})_{i,j \in [\![1,n]\!]}$ be a real symmetric $n \times n$ matrix. $\boldsymbol{M}$ is diagonalizable in an orthonormal basis. Denote $\lambda_{\min} = \lambda_1 \leq \cdots \leq \lambda_n = \lambda_{\max}$ its eigenvalues, and $(\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)})$ the corresponding eigenvectors (forming the orthonormal basis).

Appendix A.1. Rayleigh quotient

Definition. *The* ***Rayleigh quotient*** $R(\boldsymbol{M}, \boldsymbol{x})$ *associated with* $\boldsymbol{M}$ *and* $\boldsymbol{x} \in (\mathbb{R}^n)^*$ *is the ratio:*

$$R(\boldsymbol{M}, \boldsymbol{x}) = \frac{\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} = \left(\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|} \right)^T \boldsymbol{M} \left(\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|} \right)$$

Proposition. $\forall \boldsymbol{x} \in (\mathbb{R}^n)^*, \quad \lambda_{\min} \leq R(\boldsymbol{M}, \boldsymbol{x}) \leq \lambda_{\max}$

Proof. Simply notice that $\boldsymbol{x}$ can be decomposed in the orthonormal basis $(\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)})$ as: $\boldsymbol{x} = \sum_{i=1}^n \alpha_i \boldsymbol{v}^{(i)}$ for some $(\alpha_1, \ldots, \alpha_n)^T \in (\mathbb{R}^n)^*$. Then,

$$R(\boldsymbol{M}, \boldsymbol{x}) = \frac{\sum_{i=1}^n \lambda_i \alpha_i^2}{\sum_{j=1}^n \alpha_j^2} = \sum_{i=1}^n \lambda_i \frac{\alpha_i^2}{\sum_{j=1}^n \alpha_j^2}$$

which is just a weighted sum of the eigenvalues with positive weights. □

Appendix A.2. Eigenvalue bounds

Proposition. $\forall i \in [\![1,n]\!], \quad |\lambda_i| \leq \sqrt{\mathit{Trace}\,(\boldsymbol{M}^2)}$.
All the eigenvalues of $\boldsymbol{M}$ *are therefore included in the interval* $[-\sqrt{\mathit{Trace}\,(\boldsymbol{M}^2)}, \sqrt{\mathit{Trace}\,(\boldsymbol{M}^2)}]$.

Proof. Notice that $\mathrm{Trace}\left(\boldsymbol{M}^2\right) = \sum_{j=1}^n \lambda_j^2$. Take $i \in [\![1,n]\!]$. If $\lambda_i = 0$, the statement is true. Otherwise, $\mathrm{Trace}\left(\boldsymbol{M}^2\right) = \lambda_i^2 \left(1 + \sum_{j \neq i} (\lambda_j/\lambda_i)^2\right) \geq \lambda_i^2$ and therefore the statement is also true. □

Theorem. Gerschgorin circle theorem (Gerschgorin, 1931).
Any eigenvalue λ *of* $\boldsymbol{M}$ *satisfies:*

$$\lambda \in \bigcup_{i \in [\![1,n]\!]} [M_{ii} - r_i, M_{ii} + r_i], \quad r_i = \sum_{j \neq i} |M_{ij}|$$

Proof. Take λ an eigenvalue of $\boldsymbol{M}$ and $\boldsymbol{v} = (v_1, \ldots, v_n)^T$ an associated eigenvector. Let i_0 be the index of the component of $\boldsymbol{v}$ with the largest magnitude: $\forall j, |v_j| \leq |v_{i_0}|$.
Then, given that $\boldsymbol{M}\boldsymbol{v} = \lambda \boldsymbol{v}$: $\lambda v_{i_0} = \sum_{j=1}^n M_{i_0 j} v_j = M_{i_0 i_0} v_{i_0} + \sum_{j \neq i_0} M_{i_0 j} v_j$, which gives:

$$(\lambda - M_{i_0 i_0}) v_{i_0} = \textstyle\sum_{j \neq i_0} M_{i_0 j} v_j$$

So:

$$|\lambda - M_{i_0 i_0}||v_{i_0}| = \left| \textstyle\sum_{j \neq i_0} M_{i_0 j} v_j \right| \leq \textstyle\sum_{j \neq i_0} |M_{i_0 j}||v_j|$$

And finally,

$$|\lambda - M_{i_0 i_0}| \leq \textstyle\sum_{j \neq i_0} |M_{i_0 j}| \frac{|v_j|}{|v_{i_0}|} \leq r_{i_0}$$

□

Appendix B. Proof of Proposition 4.1

Proof. Take $\boldsymbol{v} \in (\mathbb{R}^n)^*$. Let's perform a chi-squared test for the variance on hypothesis H_0^v. The statistic $t(\boldsymbol{v})$ of this test is:

$$t(\boldsymbol{v}) = (N-1)\frac{S^2(\boldsymbol{v})}{\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}}$$

where $S^2(\boldsymbol{v})$ is the (unbiased) sample variance defined as:

$$S^2(\boldsymbol{v}) = \frac{1}{N-1}\sum_{i=1}^{N}\left(\boldsymbol{v}^T\boldsymbol{z}_s^{(i)} - m(\boldsymbol{v})\right)^2, \quad m(\boldsymbol{v}) = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{v}^T\boldsymbol{z}_s^{(i)}$$

H_0^v will not be rejected with significance α if $t(\boldsymbol{v})$ satisfies :

$$\chi^2_{\frac{\alpha}{2},N-1} \leq t(\boldsymbol{v}) \leq \chi^2_{1-\frac{\alpha}{2},N-1}$$

where $\chi^2_{p,N-1}$ is the p-th quantile of the chi-squared distribution with $N-1$ degrees of freedom (denoted $\chi^2(N-1)$).

In particular, the probability $R_\alpha(\boldsymbol{v})$ that H_0^v is rejected with significance α is given by:

$$\begin{aligned} R_\alpha(\boldsymbol{v}) &= 1 - Prob\left(\chi^2_{\frac{\alpha}{2},N-1} \leq t(\boldsymbol{v}) \leq \chi^2_{1-\frac{\alpha}{2},N-1}\right) \\ &= 1 - Prob\left(\frac{\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}}\chi^2_{\frac{\alpha}{2},N-1} \leq t_s(\boldsymbol{v}) \leq \frac{\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}}\chi^2_{1-\frac{\alpha}{2},N-1}\right) \end{aligned}$$

where $t_s(\boldsymbol{v})$ is the statistic defined by:

$$t_s(\boldsymbol{v}) = \frac{\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}}t(\boldsymbol{v}) = (N-1)\frac{S^2}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}}$$

By definition, the sample $\left(\boldsymbol{v}^T\boldsymbol{z}_s^{(1)}, \ldots, \boldsymbol{v}^T\boldsymbol{z}_s^{(N)}\right)$ is Gaussian with variance $\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}$. Hence, $t_s(\boldsymbol{v})$ follows a $\chi^2(N-1)$ distribution. So, if $F_{\chi^2(N-1)}$ denotes the cumulative distribution function of the $\chi^2(N-1)$ distribution:

$$R_\alpha(\boldsymbol{v}) = R_\alpha(X) = 1 - \left[F_{\chi^2(N-1)}\left(\chi^2_{1-\frac{\alpha}{2},N-1}X\right) - F_{\chi^2(N-1)}\left(\chi^2_{\frac{\alpha}{2},N-1}X\right)\right], \quad X := \frac{\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}}$$

If the variance of the simulated sample were to be equal to the true variance (i.e. $X = 1$), the probability R_α of rejecting the test would be α, the type I error of the test. But here, this error depends on the ratio X. Due to the non-symmetry of the chi-squared distribution, it can even be smaller than the significance level α of the test. However, when the size of the sample increases, the χ^2 distribution tends to regain symmetry and the minimum of the function $R_\alpha(X)$ tends to be achieved at $X = 1$ for a value α.

Denote $\gamma_\alpha(X)$ the ratio:

$$\gamma_\alpha(X) = \frac{R_\alpha(X) - \alpha}{\alpha}$$

This ratio measures the degradation of the type I error of test, due to the fact that the simulated sample has variance $\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}$ instead of $\boldsymbol{v}^T\boldsymbol{\Sigma}\boldsymbol{v}$. A condition on the value of X so that $\gamma_\alpha(X) \leq \gamma$ for some fixed proportion $\gamma \geq 0$ can be

expressed for some $\epsilon_{N,\gamma} > 0$ as follows :

$$|X - 1| \leq \epsilon_{N,\gamma} \Rightarrow \gamma_\alpha(X) \leq \gamma \tag{B.1}$$

For fixed values of γ, α and N the value of $\epsilon_{N,\gamma}$ is determined numerically by finding the two roots of the equation $\gamma_\alpha(X) = \gamma$.

The condition on X in (B.1) can be written in terms of variances:

$$\left| \frac{\boldsymbol{v}^T(\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_s)\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{\Sigma}_s\boldsymbol{v}} \right| \leq \epsilon_{N,\gamma}$$

which in turn, by denoting $\boldsymbol{\Sigma}_s^{1/2} = \mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})\boldsymbol{D}^{-1}$, can be expressed as:

$$\left| \left(\frac{\boldsymbol{\Sigma}_s^{1/2}\boldsymbol{v}}{\|\boldsymbol{\Sigma}_s^{1/2}\boldsymbol{v}\|} \right)^T \mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^{-1}(\mathrm{P}(\boldsymbol{S})^{-1} - \mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^2)\mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^{-1} \left(\frac{\boldsymbol{\Sigma}_s^{1/2}\boldsymbol{v}}{\|\boldsymbol{\Sigma}_s^{1/2}\boldsymbol{v}\|} \right) \right| \leq \epsilon_{N,\gamma}$$

Noticing the Rayleigh quotient (cf. Appendix Appendix A.1) on the left side of the inequality, this relation can be satisfied *for any* $\boldsymbol{v} \in \mathbb{R}^n$ by simply imposing:

$$\lambda_{\text{max magn}} \left(\mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^{-1}(\mathrm{P}(\boldsymbol{S})^{-1} - \mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^2)\mathrm{P}_{\mathrm{approx}}(\boldsymbol{S})^{-1} \right) \leq \epsilon_{N,\gamma}$$

where $\lambda_{\text{max magn}}(.)$ denotes the eigenvalue with the greatest magnitude of a matrix. In turn, if $[a, b]$ denotes an interval containing all the eigenvalues of $\boldsymbol{S}$, this last condition can be satisfied by imposing :

$$\max_{\lambda \in [a,b]} \left| \frac{1/\mathrm{P}(\lambda) - \mathrm{P}_{\mathrm{approx}}(\lambda)^2}{\mathrm{P}_{\mathrm{approx}}(\lambda)^2} \right| \leq \epsilon_{N,\gamma}$$

□

References

Armstrong, M., Dowd, P., 2013. Geostatistical Simulations: Proceedings of the Geostatistical Simulation Workshop, Fontainebleau, France, 27–28 May 1993. volume 7. Springer Science & Business Media.

Aune, E., Eidsvik, J., Pokern, Y., 2013. Iterative numerical methods for sampling from high dimensional gaussian distributions. Statistics and Computing 23, 501–521. URL: https://doi.org/10.1007/s11222-012-9326-8, doi:10.1007/s11222-012-9326-8.

Bevilacqua, M., Faouzi, T., Furrer, R., Porcu, E., et al., 2019. Estimation and prediction using generalized wendland covariance functions under fixed domain asymptotics. The Annals of Statistics 47, 828–856.

Bondy, J.A., Murty, U.S.R., 1976. Graph theory with applications. Macmillan.

Brigham, E.O., 1988. The fast Fourier transform and its applications. volume 1. Prentice Hall.

Chilès, J.P., Delfiner, P., 2012. Geostatistics : Modeling Spatial uncertainty. 2nd Edition. Wiley Series In Probability and Statistics.

Davis, M.W., 1987a. Generating large stochastic simulations—the matrix polynomial approximation method. Mathematical Geology 19, 99–107. URL: https://doi.org/10.1007/BF00898190, doi:10.1007/BF00898190.

Davis, M.W., 1987b. Production of conditional simulations via the lu triangular decomposition of the covariance matrix. Mathematical geology 19, 91–98.

Davis, T.A., 2006. Direct Methods for Sparse Linear Systems. volume 2. SIAM.

Deutsch, C.V., Journel, A.G., 1998. GSLIB: Geostatistical Software Library and User's Guide. 2nd Edition. Oxford University Press.

Dietrich, C., Newsam, G., 1993. A fast and exact method for multidimensional gaussian stochastic simulations. Water Resources Research 29, 2861–2869.

Dietrich, C.R., Newsam, G.N., 1995. Efficient generation of conditional simulations by chebyshev matrix polynomial approximations to the symmetric square root of the covariance matrix. Mathematical Geology 27, 207–228. URL: https://doi.org/10.1007/BF02083211, doi:10.1007/BF02083211.

Emery, X., Arroyo, D., Porcu, E., 2016. An improved spectral turning-bands algorithm for simulating stationary vector gaussian random fields. Stochastic environmental research and risk assessment 30, 1863–1873.

Emery, X., Lantuéjoul, C., 2006. Tbsim: A computer program for conditional simulation of three-dimensional gaussian random fields via the turning bands method. Computers & Geosciences 32, 1615–1628.

Frommer, A., Simoncini, V., 2008. Matrix functions. Model order reduction: theory, research aspects and applications , 275–303.

Fuglstad, G.A., Lindgren, F., Simpson, D., Rue, H., 2015. Exploring a new class of non-stationary spatial gaussian random fields with varying local anisotropy. Statistica Sinica , 115–133.

Furrer, R., Sain, S.R., et al., 2010. spam: A sparse matrix r package with emphasis on mcmc methods for gaussian markov random fields. Journal of Statistical Software 36, 1–25.

Gentle, J.E., 2009. Computational Statistics. Springer New York, New York, NY. URL: https://doi.org/10.1007/978-0-387-98144-4_7, doi:10.1007/978-0-387-98144-4_7.

Gerschgorin, S., 1931. Uber die abgrenzung der eigenwerte einer matrix. lzv. Akad. Nauk. USSR. Otd. Fiz-Mat. Nauk 7, 749–754.

Golub, G.H., Van Loan, C.F., 1996. Matrix computations. 1996. Johns Hopkins University, Press, Baltimore, MD, USA , 374–426.

Hammond, D.K., Vandergheynst, P., Gribonval, R., 2011. Wavelets on graphs via spectral graph theory. Applied and Computational Harmonic Analysis 30, 129–150.

Horn, R.A., Johnson, C.R., 1990. Matrix analysis. Cambridge university press.

Kaufman, C.G., Schervish, M.J., Nychka, D.W., 2008. Covariance tapering for likelihood-based estimation in large spatial data sets. Journal of the American Statistical Association 103, 1545–1555.

Lang, A., Potthoff, J., 2011. Fast simulation of gaussian random fields. Monte Carlo Methods and Applications 17, 195–214.

Lantuéjoul, C., 2013. Geostatistical simulation: models and algorithms. Springer Science & Business Media.

Lindgren, F., Rue, H., Lindström, J., 2011. An explicit link between gaussian fields 670 and gaussian markov random fields: the spde approach (with discussion). JR 671 Stat Soc, Series B 73, 423–498.

Mason, J.C., Handscomb, D.C., 2002. Chebyshev polynomials. CRC Press.

Matheron, G., 1973. The intrinsic random functions and their applications. Advances in applied probability 5, 439–468.

Musco, C., Musco, C., Sidford, A., 2017. Stability of the Lanczos Method for Matrix Function Approximation. arXiv URL: https://arxiv.org/abs/1708.07788, arXiv:1708.07788.

Pardo-Igúzquiza, E., Chica-Olmo, M., 1993. The fourier integral method: An efficient spectral method for simulation of random fields. Mathematical Geology 25, 177–217. URL: https://doi.org/10.1007/BF00893272, doi:10.1007/BF00893272.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 2007. Numerical Recipes 3rd Edition: The Art of Scientific Computing. Cambridge University Press.

Renard, D., Bez, N., Desassis, N., Beucher, H., Ors, F., Freulon, X., 2018. RGeostats: The geostatistical package [11.2.2]. Ecole des Mines de Paris Free download from http://cg.ensmp.fr/rgeostats.

Rozanov, J.A., 1977. Markov random fields and stochastic partial differential equations. Mathematics of the USSR-Sbornik 32, 515. URL: http://stacks.iop.org/0025-5734/32/i=4/a=A08.

Rue, H., Held, L., 2005. Gaussian Markov random fields: theory and applications. CRC press.

Schlather, M., Malinowski, A., Menck, P.J., Oesting, M., Strokorb, K., et al., 2015. Analysis, simulation and prediction of multivariate random fields with package randomfields. Journal of Statistical Software 63, 1–25.

Shinozuka, M., Jan, C.M., 1972. Digital simulation of random processes and its applications. Journal of sound and vibration 25, 111–128.

Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P., 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Processing Magazine 30, 83–98.

Simpson, D., Lindgren, F., Rue, H., 2012. Think continuous: Markovian gaussian models in spatial statistics. Spatial Statistics 1, 16 – 29. URL: http://www.sciencedirect.com/science/article/pii/S2211675312000048, doi:https://doi.org/10.1016/j.spasta.2012.02.003.

Simpson, D.P., Turner, I.W., Pettitt, A.N., 2008. Fast sampling from a Gaussian Markov random field using Krylov subspace approaches. Technical Report.

Simpson, D.P., Turner, I.W., Strickland, C.M., Pettitt, A.N., 2013. Scalable iterative methods for sampling from massive gaussian random vectors. arXiv preprint arXiv:1312.1476 .

Snedecor, G.W., Cochran, W.G., 1989. Statistical methods. Eight Edition. Iowa University Press.

Tong, Y.L., 2012. The multivariate normal distribution. Springer Science & Business Media.

Wackernagel, H., 2013. Multivariate geostatistics: an introduction with applications. Springer Science & Business Media.

Whittle, P., 1954. On stationary processes in the plane. Biometrika , 434–449.

Wood, A.T., Chan, G., 1994. Simulation of stationary gaussian processes in [0, 1] d. Journal of computational and graphical statistics 3, 409–432.