How to initialize the Circulant Embedding method to speed up the generation of stationary Gaussian Random Fields?

Géraldine Pichot, Simon Legrand, Michel Kern, Nathanael Tepakbong-Tematio

Journal of Computational Mathematics
Vol. 00, Pages 000–000 (XXXX)

How to initialize the Circulant Embedding method to speed up the generation of stationary Gaussian Random Fields?

Géraldine Pichot [1]
Simon Legrand [2]
Michel Kern [3]
Nathanael Tepakbong-Tematio [4]

[1] Inria, 2 rue Simone Iff, 75589 Paris, France and Université Paris-Est, CERMICS (ENPC), 6 et 8 av. Blaise Pascal, 77455 Marne-la-Vallée Cedex 2, France
Email address: geraldine.pichot@inria.fr
[2] Inria, 2 rue Simone Iff, 75589 Paris, France
Email address: simon.legrand@inria.fr
[3] Inria, 2 rue Simone Iff, 75589 Paris, France and Université Paris-Est, CERMICS (ENPC), 6 et 8 av. Blaise Pascal, 77455 Marne-la-Vallée Cedex 2, France
Email address: michel.kern@inria.fr
[4] ISAE-SUPAERO, 10, avenue Édouard-Belin, BP 54032, 31055 Toulouse Cedex 4, France
Email address: nathanael.tepakbong-tematio@student.isae-supaero.fr.

Abstract. The Circulant Embedding Method (CEM) is a well known technique to generate stationary Gaussian Random Fields (GRF). The main idea is to embed the covariance matrix in a larger nested block circulant matrix, whose factorization can be rapidly computed thanks to the fast Fourier transform (FFT) algorithm. The CEM requires the extended matrix to be at least positive semidefinite which is proven to be the case if the enclosing domain is sufficiently large, as proven by Theorem 2.3 in [Graham *et al*, SIAM Journal on Numerical Analysis, 2018] for cubic domains. In this paper, we generalize this theorem to the case of rectangular parallelepipeds. Then we propose a new initialization stage of the CEM algorithm which allows to quickly jump to a domain size close to the one needed for the CEM algorithm to work. These domain size estimates are based on fitting functions. Examples of fitting functions are given for the Matérn family of covariances. These functions are inspired by our numerical simulations and by the theoretical work from [Graham *et al*, SIAM Journal on Numerical Analysis, 2018]. The parameters estimation of the fitting functions is done numerically. Several numerical tests are performed to show the efficiency of the proposed algorithms, for both isotropic and anisotropic Matérn covariances.

Keywords. stationary Gaussian random fields, circulant embedding method, Matérn covariances, fast Fourier transform.

2020 Mathematics Subject Classification. 60G60; 65C10; 65C05; 86A32.

1. Introduction

Stationary Gaussian Random Fields (GRF) are classically used to model physical properties in environment applications. For example, in hydrogeology, stationary GRF are used to model the permeability of a porous domain [8, 2]. Here we focus on the generation of stationary GRF on regular grids. If the grid has M points, the general algorithm to generate a realization of the stationary random vector $\mathbf{Y}$ of M normal variables, with zero mean and the $M \times M$ covariance matrix $\mathbf{R}$, is based on a factorization of the covariance matrix $\mathbf{R}$ as shown by Algorithm 1.

Using Algorithm 1, $\mathbf{Y}$ has $\mathbf{R}$ as covariance matrix:

$$\mathbb{E}[\mathbf{Y}\mathbf{Y}^T] = \mathbb{E}[(\mathbf{B}\boldsymbol{\theta})(\mathbf{B}\boldsymbol{\theta})^T] = \mathbf{B}\mathbb{E}[\boldsymbol{\theta}\boldsymbol{\theta}^T]\mathbf{B}^T = \mathbf{B}\mathbf{B}^T = \mathbf{R}.$$

Algorithm 1 General algorithm

1: Factorize $\mathbf{R} = \mathbf{B}\mathbf{B}^T$ with $\mathbf{B}$ of size $M \times M$;
2: Generate a vector $\boldsymbol{\theta}$ of size M of realizations of standard normal random variables, with zero mean and $\mathbb{E}[\boldsymbol{\theta}\boldsymbol{\theta}^T] = Id$,
3: One realization with the requested covariance matrix $\mathbf{R}$ is obtained by computing $\mathbf{Y} = \mathbf{B}\boldsymbol{\theta}$.

Algorithm 1 requires the factorization of the covariance matrix $\mathbf{R}$ which can be done according to different methods, a review of those methods can be found in [11]. The most direct method is based on the Cholesky factorization of matrix $\mathbf{R}$. However it requires $\mathrm{O}(M^3)$ operations to factorize a matrix of size M, which becomes prohibitive for large M. Among the other techniques, let us cite the Karhunen-Loève (K.-L.) expansion [13]. The K.-L. expansion is infinite, so in practice the expansion is truncated to obtain an approximation of a GRF. The number of terms to keep depends on the correlation length and on the input domain size: typically the smaller the correlation length with respect to the input domain size, the larger the number of terms to keep and the larger the computational costs. Another method based on H-matrices has recently been developed in [6]. It also has the advantage of handling non-stationary covariance functions. It gives approximations of random fields at given points, also possibly distributed on irregular grids. Another method is the Circulant Embedding Method (CEM) which produces random vectors with exactly the required correlation structure on regular grids [4, 5, 3, 9]. The principle of CEM is to embed the matrix $\mathbf{R}$ of size M in a bigger $s \times s$ nested block circulant matrix $\mathbf{R}^{\mathrm{ext}}$, whose factorization can be rapidly computed thanks to the FFT algorithm in $\mathrm{O}(s \log(s))$ operations. The matrix $\mathbf{R}$ is positive semidefinite and also a nested block symmetric Toeplitz matrix under an appropriate ordering of the indices. However the extension $\mathbf{R}^{\mathrm{ext}}$ might not be positive semidefinite unless the extended domain is large enough [9]. In [9], a uniform grid of points on the d-dimensional unit cube $[0, 1]^d$ (d is the dimension) is considered together with isotropic covariance functions.

In this work, we are interested in reducing the time needed to generate Gaussian random fields with the CEM on d-dimensional rectangular parallelepipeds of size $L_1 \times \cdots \times L_d$ with isotropic and also anisotropic covariance functions. We want to decrease the number of tests that have to be carried out before a semi-positive definite matrix $\mathbf{R}^{\mathrm{ext}}$ is obtained (see step 3 of Algorithm 2 on page 6). Towards this aim we propose a new initialization of the CEM algorithm to save computational resources. This new initialization is based on an offline stage to set up a function that estimates the size of the enlarged domain, close to the one that ensures that the extended matrix $\mathbf{R}^{\mathrm{ext}}$ is at least positive semidefinite. Once this function is obtained, it is used to compute a guess of the required enlarged domain. The main contributions of the paper are:

- an extension of Theorem 2.3 from [9] to the cases of d-dimensional rectangular parallelepipeds;
- a design of fitting functions for the specific cases of Matérn and Gaussian covariance functions;
- an extensive set of numerical experiments that validate the proposed procedure for the case of anisotropic covariance functions.

It is hoped that the resulting functions may be useful for researchers and practitioners who have to use the CEM method.

The outline of this paper is as follows: Section 2 presents the notations used in this paper, the Matérn family of covariances with smoothness parameter ν ($1/2 \leq \nu \leq \infty$) together with the classical CEM algorithm. Section 3 proves that, under quite general conditions on the covariance functions and

for d-dimensional rectangular parallelepipeds, the CEM algorithm always terminates. Section 4 gives the new initialization stage we propose to speed up the computations of the classical CEM algorithm and the procedure to get a function that gives a good estimate of the required size of the enlarged domain. Section 5 proposes to apply this method to the Matérn family of covariances. Section 6 confirms the quality of the estimates obtained with our new initialization on several isotropic and anisotropic Matérn covariance test cases.

All the simulations in this paper were carried out with the parallel C++17 `ParaCirce`[1] library that implements both the classical and the new CEM algorithms presented in this paper. `ParaCirce` performs the discrete Fourier transforms with the `FFTW3` library [7] and uses `Rngstream` [10] for the generation of pseudo-random numbers. `ParaCirce` handles 64-bit and 80-bit floating point precision with template arguments. `ParaCirce` parallelization is based on the MPI standard. The simulations are done in 80-bit floating point precision in order to avoid, as much as possible, the stagnation of the minimum eigenvalue as explained in Appendix A.

2. The CEM method

2.1. Notations

Consider a grid of size $L_1 \times \cdots \times L_d$ with given number of points $m_{0,i}+1$ in each direction i, $i = 1, \ldots, d$. The objective is to the generate a stationary random vector $\mathbf{Y}$ on this grid with

$$M = (m_{0,1} + 1) \times \cdots \times (m_{0,d} + 1)$$

points. The grid spacing along each direction i is constant and denoted by $h_{0,i}$. It is given by

$$h_{0,i} := \frac{L_i}{m_{0,i}}, \tag{2.1}$$

and the grid points are $\mathbf{x}_1; \mathbf{x}_2; ...; \mathbf{x}_M$.

We denote by diag the operator that, given a vector argument, creates a matrix with the vector values along the diagonal and $\mathbf{H}_0 = \mathrm{diag}(h_{0,1}, \ldots, h_{0,d})$ the diagonal space step matrix.

For the particular case of a cubic domain of size L, the given number of points in each direction is $m_0 + 1$ and the spacing h_0 is the same in each direction and given by:

$$h_0 := \frac{L}{m_0}. \tag{2.2}$$

Let us consider the discrete representation of $\mathbf{Y}$ as $[Y_i]_{i=1}^M$. The covariance matrix $\mathbf{R}$ is

$$\mathbf{R} = [\rho(\mathbf{x}_i - \mathbf{x}_j)]_{i;j=1}^M = \mathbb{E}[Y_i Y_j]_{i;j=1}^M$$

with $\rho : \mathbb{R}^d \to \mathbb{R}$ the covariance function.

2.2. The Matérn family of covariances

A common family of covariances is the Matérn family [12] with correlation lengths $\boldsymbol{\lambda} = (\lambda_i)_{i=1,\ldots,d}$. We denote by $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ the diagonal correlation length matrix. The standard anisotropic Matérn family of covariances functions in 2D ($d = 2$) and in 3D ($d = 3$) with $\boldsymbol{x} = (x_i)_{i=1,\ldots,d}$ writes

[1] https://gitlab.inria.fr/slegrand/paracirce

$$\rho_d(\mathbf{x}, \nu, \boldsymbol{\lambda}) = \kappa\left(\sqrt{\sum_{i=1}^{d}\left(\frac{x_i}{\lambda_i}\right)^2}, \nu\right) = \kappa\left(||\boldsymbol{\Lambda}^{-1}\mathbf{x}||_{\mathbf{2}}, \nu\right), \tag{2.3}$$

where

$$\kappa(r, \nu) = \frac{2^{1-\nu}}{\Gamma(\nu)}(\sqrt{2\nu}\, r)^{\nu} K_{\nu}(\sqrt{2\nu}\, r), \tag{2.4}$$

with Γ the gamma function and K_ν is the modified Bessel function of the second kind, λ_i is the correlation length in direction i, $i = 1, \ldots, d$ and $\nu > 0$ is a given smoothness parameter. The case $\nu = 1/2$ corresponds to the (non separable) exponential covariance with $\kappa(r) = \exp(-r)$ and $\nu = \infty$ to the Gaussian covariance with $\kappa(r) = \exp(-r^2/2)$.

In the isotropic case, $\lambda_i = \lambda, \forall i = 1, \ldots, d$ and the standard isotropic Matérn family of covariances becomes

$$\rho(\mathbf{x}, \nu, \lambda) = \kappa\left(\frac{||\mathbf{x}||_{\mathbf{2}}}{\lambda}, \nu\right). \tag{2.5}$$

Remark 2.1. If $\mathbf{Y}$ is a second order stationary Gaussian field with mean 0 and covariance function ρ, $\mu + \sigma\mathbf{Y}$ is a second order stationary Gaussian field with mean μ and covariance function $\sigma^2\rho$.

Examples of 2D fields with isotropic Matérn covariances are shown on Figure 1.

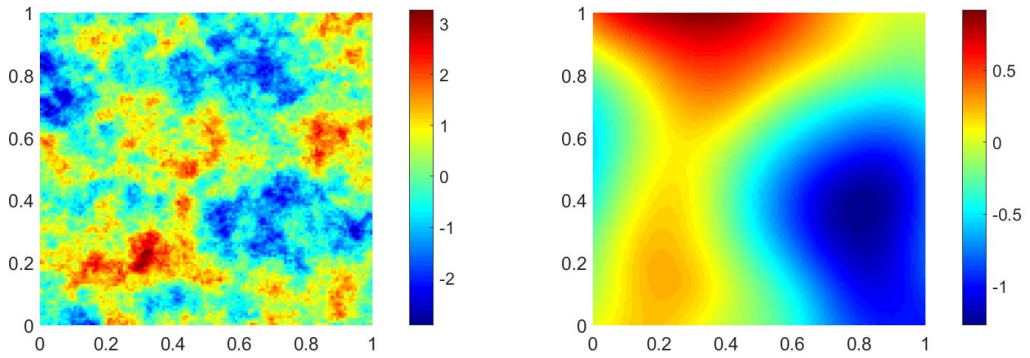

Figure 1. 2D case ($m_{0,1} = m_{0,2} = 128$, $L_1 = L_2 = 1$) - isotropic Matérn covariances: $\lambda = 0.125$ and $\nu = 0.5$ (left); $\lambda = 0.5$ and $\nu = 4$ (right).

Examples of 2D fields with anisotropic Matérn covariances are shown on Figure 2.
Examples of 2D fields with isotropic and anisotropic Gaussian covariances are shown on Figure 3.

2.3. **Classical simulation algorithm**

The principle of CEM is to embed the matrix $\mathbf{R}$ in a bigger $s \times s$ symmetric nested block circulant matrix $\mathbf{R}^{\text{ext}}$ with $s = (2m_1) \times \cdots \times (2m_d)$, so that its factorization can be rapidly computed with the FFT algorithm. The m_i, $i = 1, \ldots, d$ have to be determined so that $\mathbf{R}^{\text{ext}}$ is at least positive semidefinite. In the following, we explain how to estimate them in an efficient way. The circulant structure is obtained by a mirroring step of $\mathbf{R}$ as explained in details for example in [8, 9]. In practice, with the CEM, we never need to build the full nested block circulant matrix $\mathbf{R}^{\text{ext}}$, we only need its

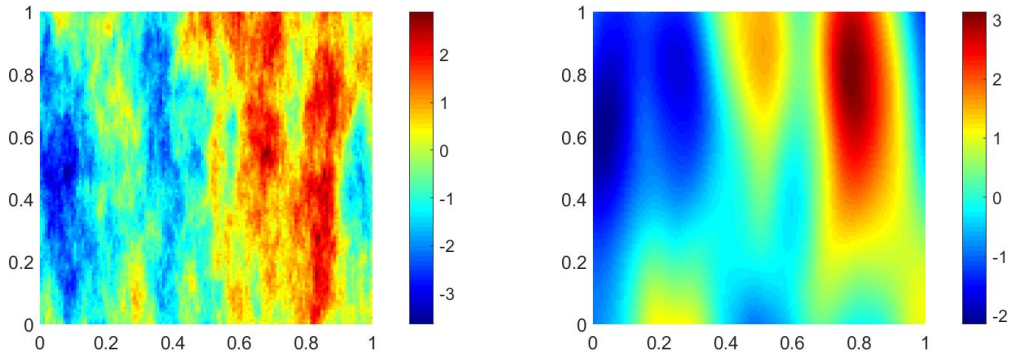

FIGURE 2. 2D case ($m_{0,1} = m_{0,2} = 128$, $L_1 = L_2 = 1$) - anisotropic Matérn covariances, $\lambda_1 = 0.125$, $\lambda_2 = 0.5$: $\nu = 0.5$ (left); $\nu = 4$ (right)

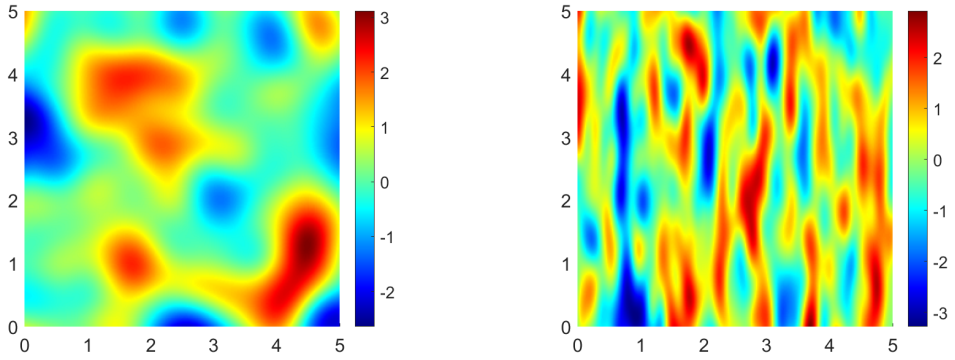

FIGURE 3. 2D case - Gaussian covariance ($m_{0,1} = m_{0,2} = 256$, $L_1 = L_2 = 5$) - isotropic: $\lambda = 0.5$ (left); anisotropic: $\lambda_1 = 0.125$, $\lambda_2 = 0.5$ (right).

first column, denoted by $\mathbf{r}$. If one is interested by building the full matrix $\mathbf{R}^{\text{ext}}$, details are given for example page 77 in [3] or in [9]. The sizes $m_i \geq m_{0,i}$, or equivalently the enlarged domain of size

$$\ell_i := m_i \times h_{0,i} \geq L_i, \quad i = 1, \ldots, d \quad (h_{0,i} \text{ is the grid spacing given by (2.1)})$$

must be chosen so as to guarantee that the matrix $\mathbf{R}^{\text{ext}}$ is at least positive semidefinite, otherwise the algorithm cannot work due to the presence of negative eigenvalues. Here, we call padding the extension between $m_{0,i}$ and m_i. We denote by $\mathbf{M} = \text{diag}(m_1, \ldots, m_d)$ the diagonal matrix containing m_i. As a padding strategy, we follow the approach from [8, 12] to extend the covariance with the values of $\rho(\mathbf{x}, \nu, \lambda)$. Other padding strategies could have been chosen, e.g. as proposed in the original paper [4] (for instance a filling with zeros), but these are not studied in this paper. Note that the padding may be different in each direction according to the different values of L_i, ν, λ_i and $m_{0,i}$, $i = 1, \ldots, d$.

In the case of a cubic domain of size L, the length of the extended domain to guarantee that the matrix $\mathbf{R}^{\text{ext}}$ is at least positive semidefinite is called $\ell \geq L$ and is such that

$$\ell := m \times h_0 \text{ with } m \geq m_0 \quad (h_0 \text{ is the grid spacing given by (2.2)})$$

with m to be determined.

Therefore the CEM method relies on two steps:

Step 1. the computation of the size of the extended domain ℓ_i, or equivalently m_i, $i = 1, \ldots, d$ (or ℓ and m for a cubic domain) to guarantee that $\mathbf{R}^{\text{ext}}$ is at least positive semidefinite;

Step 2. the sampling of instances of the random field with the discrete Fourier transform.

These two steps can be performed according to Algorithm 2, with a usual start with $m_{0,i} + 1$ points, $i = \{1, \ldots, d\}$ and Algorithm 3 (modified from [9] to consider rectangular parallelepipeds). In Algorithm 2, the number of iterations at Step 5 (that is the number of times m_i is incremented) depends on the initial value of $m_{0,i}$, L_i, λ_i, ν and d. Each iteration costs the computation of $\mathbf{r}$, the first column of $\mathbf{R}^{\text{ext}}$, and a d-dimensional FFT. The last stage of Algorithm 3 allows to recover two independent instances with the required correlation structure and the correct input size $(m_{0,1}+1) \times \cdots \times (m_{0,d}+1)$ from the two fields $\mathbf{w}_{re}$ and $\mathbf{w}_{im}$ of size $(2m_1) \times \cdots \times (2m_d)$ respectively.

Algorithm 2 Step 1. Compute the size of the extended domain to guarantee that $\mathbf{R}^{\text{ext}}$ is at least positive semidefinite.

Data: d, $m_{0,i}$, L_i and covariance function ρ with correlation length λ_i, $i = 1, \ldots, d$ and a given starting number of points $m_i^{\text{start}} + 1$, $i = \{1, \ldots, d\}$.
Result: Number of points m_i+1 ($i = \{1, \ldots, d\}$) to guarantee that $\mathbf{R}^{\text{ext}}$ is positive definite and the vector of eigenvalues $\mathbf{v}$.
1: Set $m_i = m_i^{\text{start}}$, $i = \{1, \ldots, d\}$.
2: Calculate $\mathbf{r}$, the first column of $\mathbf{R}^{\text{ext}}$.
3: Calculate $\mathbf{v}$, the vector of eigenvalues of $\mathbf{R}^{\text{ext}}$, by d-dimensional FFT on $\mathbf{r}$.
4: **if** smallest eigenvalue < 0 **then**
5: increment m_i and go to Step 2.

Algorithm 3 Step 2. Sample two independent instances $\mathbf{z}_1$ and $\mathbf{z}_2$ of the random field.

Data: d, $m_{0,i}$, and m_i and $\mathbf{v}$ obtained by Algorithm 2.
Result: Two instances $\mathbf{z}_1$ and $\mathbf{z}_2$ of the random field.
1: With $s = (2m_1) \times \cdots \times (2m_d)$, sample two s-dimensional normal random vector, called $\mathbf{y}_{re}$ and $\mathbf{y}_{im}$ to stand for the real and imaginary part of a complex random vector called $\mathbf{y}$: $\mathbf{y} = \mathbf{y}_{re} + i\mathbf{y}_{im}$
2: Update $\mathbf{y}_{re}$ and $\mathbf{y}_{im}$ by elementwise multiplication with $\sqrt{\mathbf{v}}$.
3: Set $\mathbf{w} = \mathbf{w}_{re} + i\mathbf{w}_{im}$ to be the d-dimensional iFFT of $\mathbf{y}$ with $\mathbf{w}_{re}$ and $\mathbf{w}_{im}$ the real and imaginary part of $\mathbf{w}$ respectively.
4: Obtain two independent instances $\mathbf{z}_1$ and $\mathbf{z}_2$ by extracting the appropriate $M = (m_{0,1}+1) \times \cdots \times (m_{0,d}+1)$ entries of $\mathbf{w}_{re}$ and $\mathbf{w}_{im}$ respectively.

Remark 2.2. Algorithm 3 yields two independent realizations of the random field [4]. Lemma 5 in [8] and Algorithm 2 in [9] explain how to sample only one field.

3. Termination of Algorithm 2

For the case of a d-dimensional unit cube $[0, 1]^d$, with $\mathbf{\Lambda} = \lambda I$ and $\mathbf{H}_0 = h_0 I$, termination of Algorithm 2 is proven in Theorem 2.3 in [9]. For general domain of size $L_1 \times \cdots \times L_d$ with given number of $m_{0,i} + 1$ points in each direction i, $i = 1, \ldots, d$, we generalize Theorem 2.3 in [9] as follows

Theorem 3.1 (Generalization of Theorem 2.3 in [9])**.** *Suppose that $\rho \in L^1(\mathbb{R}^d)$ is a real-valued, symmetric positive definite function with the additional reflectional symmetry*

$$\rho(\mathbf{x}) = \rho(\pm x_1, \ldots, \pm x_d) = \rho(|x_1|, \ldots, |x_d|) \quad \textit{for all } \mathbf{x} \in \mathbb{R}^d,$$

and suppose its Fourier transform $\hat{\rho} \in L^1(\mathbb{R}^d)$. Suppose also that for the given matrix $\mathbf{H}_0$ we have:

$$\sum_{k \in \mathbb{Z}^d} |\rho(\mathbf{H}_0 \mathbf{k})| < \infty.$$

Then Algorithm 2 will always terminate with finite values of $m_1, \ldots, m_d$ and the resulting matrix $\mathbf{R}^{ext}$ will be positive definite.

Let us define, for any integer $m_i \geq 1$, $i = 1, \ldots, d$,

$$\mathbb{Z}^d_{2M} := \{0, ..., 2m_1 - 1\} \times \cdots \times \{0, ..., 2m_d - 1\}, \quad \overline{\mathbb{Z}}^d_M := \{-m_1, ..., m_1 - 1\} \times \cdots \times \{-m_d, ..., m_d - 1\}.$$

To prove Theorem 3.1, we need the following Lemma:

Lemma 3.2 (Generalization of Lemma 2.4 in [9])**.** *Under the assumptions of Theorem 3.1, the eigenvalues Δ_k^{ext} of $\mathbf{R}^{ext}$ all satisfy the estimate*

$$\Delta_k^{ext} \geq \frac{1}{\prod_{i=1}^d h_{0,i}} \min_{\boldsymbol{\zeta} \in [-\frac{1}{2}, \frac{1}{2}]^d} \sum_{\mathbf{r} \in \mathbb{Z}^d} \hat{\rho}\left(\mathbf{H}_0^{-1}(\boldsymbol{\zeta} + \mathbf{r})\right) - \sum_{\mathbf{k}' \in \mathbb{Z}^d \setminus \overline{\mathbb{Z}}^d_M} |\rho(\mathbf{H}_0 \mathbf{k}')|. \tag{3.1}$$

Proof.
Recall that $\mathbf{M} = \mathrm{diag}(m_1, \ldots, m_d)$ denotes the diagonal matrix containing m_i.

The extended matrix $\mathbf{R}^{\mathrm{ext}}$ has eigenvalues $\Delta_{\mathbf{k}}^{\mathrm{ext}}$ and corresponding eigenvectors $\mathbf{V}_{\mathbf{k}}$, given, for $\mathbf{k} \in \mathbb{Z}^d_{2M}$, by the formulae

$$\Delta_{\mathbf{k}}^{\mathrm{ext}} = \sum_{\mathbf{k}' \in \overline{\mathbb{Z}}^d_M} \rho\left(\mathbf{H}_0 \mathbf{k}'\right) \exp\left(-2i\pi \frac{\mathbf{k}^T \mathbf{M}^{-1} \mathbf{k}'}{2}\right), \tag{3.2}$$

$$(\mathbf{V}_{\mathbf{k}})_{\boldsymbol{\kappa}} = \frac{1}{\sqrt{\prod_{i=1}^d (2m_i)}} \exp\left(2i\pi \frac{\mathbf{k}^T \mathbf{M}^{-1} \boldsymbol{\kappa}}{2}\right), \boldsymbol{\kappa} \in \mathbb{Z}^d_{2M}. \tag{3.3}$$

From (3.2) (as 2.13 in [9]):

$$\Delta_{\mathbf{k}}^{\mathrm{ext}} = \sum_{\mathbf{k}' \in \mathbb{Z}^d} \rho\left(\mathbf{H}_0 \mathbf{k}'\right) \exp\left(-2i\pi \frac{\mathbf{k}^T \mathbf{M}^{-1} \mathbf{k}'}{2}\right) - \sum_{\mathbf{k}' \in \mathbb{Z}^d \setminus \overline{\mathbb{Z}}^d_M} \rho\left(\mathbf{H}_0 \mathbf{k}'\right) \exp\left(-2i\pi \frac{\mathbf{k}^T \mathbf{M}^{-1} \mathbf{k}'}{2}\right) \tag{3.4}$$

With the same arguments as in Appendix A in [9], we obtain a similar lower bound on the first sum on the right-hand side of (3.4):

$$\sum_{\mathbf{k}' \in \mathbb{Z}^d} \rho\left(\mathbf{H}_0 \mathbf{k}'\right) \exp\left(-2i\pi \frac{\mathbf{k}^T \mathbf{M}^{-1} \mathbf{k}'}{2}\right) \geq \operatorname*{essinf}_{\boldsymbol{\zeta} \in \prod_{i=1}^d [-\frac{1}{2h_i}, \frac{1}{2h_i}]^d} \frac{1}{\prod_{i=1}^d h_i} \sum_{\mathbf{r} \in \mathbb{Z}^d} \hat{\rho}(\boldsymbol{\zeta} + \mathbf{H}_0^{-1} \mathbf{r}) \tag{3.5}$$

$$\geq \operatorname*{essinf}_{\boldsymbol{\zeta} \in \prod_{i=1}^d [-\frac{1}{2h_i}, \frac{1}{2h_i}]^d} \frac{1}{\prod_{i=1}^d h_i} \max_{\mathbf{r} \in \mathbb{Z}^d} \hat{\rho}(\boldsymbol{\zeta} + \mathbf{H}_0^{-1} \mathbf{r}) \tag{3.6}$$

$$\geq \max_{\mathbf{r} \in \mathbb{Z}^d} \min_{\boldsymbol{\zeta} \in [-\frac{1}{2}, \frac{1}{2}]^d} \frac{1}{\prod_{i=1}^d h_i} \hat{\rho}\left(\mathbf{H}_0^{-1}(\boldsymbol{\zeta} + \mathbf{r})\right). \tag{3.7}$$

The upper bound on the second sum of (3.4) is obtained in an obvious way.

■

Proof. [Proof of Theorem 3.1]
The proof of Theorem 3.1 is similar to the one given in [9] (page 1879). We use Lemma 3.2 and notice that the first term of the right hand side of (3.1) is independent of $m_1, \ldots, m_d$ while the second term is the tail of a converging series, so will be smaller than the first term provided $m_1, \ldots, m_d$ are chosen large enough.

■

According to Theorem 3.1, it is proven that, provided the domain is sufficiently enlarged, Algorithm 2 theoretically always terminates. This offers the possibility to choose a starting value of m_i larger than $m_{0,i}$ $(i = 1, \ldots, d)$, at Step 1.

Remark 3.3. In some of our numerical experiments, we observed stagnation of the smallest eigenvalue (see Appendix A). This is due to the limited precision caused by floating point arithmetic. Increasing floating point precision of the code might help in solving this numerical issue.

4. New initialization to speed up Algorithm 2

Now we propose a strategy to speed-up Algorithm 2 for any covariance function that satisfies the hypotheses of Theorem 3.1. A classical initialization Step 1 of Algorithm 2 is to set m_i to the given number of points: $m_i^{\text{start}} = m_{0,i}$, $i = \{1, \ldots, d\}$ [9]. If no padding is required (the smallest eigenvalue is positive), Algorithm 2 terminates instantly, but if the smallest eigenvalue is negative (hence a padding is required), m_i, $i = 1, \ldots, d$, must be incremented as suggested at Step 5 of Algorithm 2 until the smallest eigenvalue becomes positive. Without any *a priori* knowledge on how to increment m_i at Step 5, by default, a possible choice is to increment each $m_i, i = 1, \ldots, d$ by a constant, for instance by 1. Other choices are possible (that could be more costly), like doubling the number of points m_i on each direction i. Each time m_i, $i = 1, \ldots, d$ is incremented, the computation of a d-dimensional FFT on $\mathbf{r}$ is required, which can be costly if a large padding is needed.

4.1. New initialization stage

Instead of starting Algorithm 2 from $m_i^{\text{start}} = m_{0,i}$, $i = \{1, \ldots, d\}$ at Step 1, we propose to start from a first estimate: $m_i^{\text{start}} = m_{\mathcal{F},i}^{\text{est}}$, $i = \{1, \ldots, d\}$. The next subsection explains how to compute this estimate $m_{\mathcal{F},i}^{\text{est}}$. The subscript $\mathcal{F}$ stands for the name of fitting function used in the estimation.

If the estimates $m_{\mathcal{F},i}^{\text{est}}$, $i = \{1, \ldots, d\}$, are good enough, there will be no need to iterate, hence saving a lot of computational time and resources. If the estimates underestimate the padded domain (meaning the number of iterations is not zero), this is not an issue as Algorithm 2 suggests to further increase m_i, $i = 1, \ldots, d$, until the smallest eigenvalue is positive. Moreover, from Theorem 3.1, we know that Algorithm 2 terminates provided the hypotheses on the covariance function ρ are satisfied and the domain is sufficiently large, hence Algorithm 2 with this new initialization stage will always terminate as well, while potentially decreasing the number of FFT computations.

4.2. Estimation of $m_{\mathcal{F},i}^{\text{est}}$

To estimate $m_{\mathcal{F},i}^{\text{est}}$, $i = 1, \ldots, d$, we propose two offline steps:

(S1) Define a set of parameters in the isotropic case ($\lambda_i = \lambda$, $\forall i \in \{1, \ldots, d\}$) with a cubic domain ($L_i = L$, $m_{0,i} = m_0$, $h_{0,i} = h_0 = L/m_0$, $\forall i \in \{1, \ldots, d\}$). Using Algorithm 2 with $m_i^{\text{start}} = m_{0,i}$, $i = \{1, \ldots, d\}$ and an increment of 1 at Step 5 ($m_i \leftarrow m_i + 1$), **compute the minimum value of the domain size** denoted by $\ell^{\min}$ so that Algorithm 2 terminates.

(S2) **Fit the data from Step (S1) to a fitting function** $\mathcal{F}$ with a known form (inspired by numerical simulations and possibly theoretical work).

Once the fitting function is determined, it can be applied to estimate the minimum length $\ell_{\mathcal{F}}^{\text{est}}$ (and the associated number of points $m_{\mathcal{F}}^{\text{est}}+1$) for each new set of parameters. In the anisotropic cases, the function $\mathcal{F}$ is applied separately in each direction $i \in \{1,\dots,d\}$ to get $m_{\mathcal{F},i}^{\text{est}}$.

Remark 4.1. As $\mathbf{R}^{\text{ext}}$ has many very small eigenvalues and some of them might very close to zero but negative in finite precision arithmetic, we stop Algorithm 2 once all eigenvalues are above a given threshold τ (see Appendix A). Those eigenvalues that are less than this given threshold τ will be set to 0. In [9], τ is set to -10^{-13}.

Remark 4.2. At step (S1), as mentionned before, other choices for the increment at Step 5 of Algorithm 2 could have been made. Here we have chosen an increment of 1 in order to get the exact enlarged domain size $\ell^{\min}$ from which all eigenvalues are above the threshold τ.

Here are more details of this offline stage and how it is applied to compute $m_{\mathcal{F},i}^{\text{est}}$. For the sake of simplicity, we consider a set of simulations on a unit cubic domain so that we work with the dimensionless quantities: λ/L and h_0/L. Once all the simulations have been run, one obtains a table of the estimated dimensionless length $\ell^{\min}/L$ versus the set of input parameters (ν, λ/L, h_0/L). From this table, we introduce a fitting function $\mathcal{F}$, which fits $\ell^{\min}/L$ in a least-squares sense, to obtain an estimated length $\ell_{\mathcal{F}}^{\text{est}}$ and an associated number of points $m_{\mathcal{F}}^{\text{est}}+1$:

$$\ell_{\mathcal{F}}^{\text{est}} := \mathcal{F}(\nu, \lambda/L, h_0/L)L, \tag{4.1}$$

$$m_{\mathcal{F}}^{\text{est}} := \max(m_0, \lceil \ell_{\mathcal{F}}^{\text{est}}/h_0 \rceil), \tag{4.2}$$

where $\lceil x \rceil$ denotes the ceiling part of x.

Finally, to obtain an estimate of $\ell_{\mathcal{F},i}^{\text{est}}$ and the associated number of points $m_{\mathcal{F},i}^{\text{est}}+1$ of the enlarged domain in the general anisotropic and d-dimensional rectangular parallelepipeds case, we apply the fitting function $\mathcal{F}$ in each direction, that is, for every $i \in \{1,\dots,d\}$:

$$\ell_{\mathcal{F},i}^{\text{est}} := \mathcal{F}(\nu, \lambda_i/L_i, h_{0,i}/L_i)L_i, \tag{4.3}$$

$$m_{\mathcal{F},i}^{\text{est}} := \max(m_{0,i}, \lceil \ell_{\mathcal{F},i}^{\text{est}}/h_{0,i} \rceil). \tag{4.4}$$

The corresponding estimated length is $\ell_i^{\text{est}} := m_{\mathcal{F},i}^{\text{est}} \times h_{0,i}$ (it might slightly differ from $\ell_{\mathcal{F},i}^{\text{est}}$ due to the ceiling part taken to get an integer value for $m_{\mathcal{F},i}^{\text{est}}$) and the extended domain in the direction i is $\max(L_i, \ell_i^{\text{est}})$.

Now let us present some examples of fitting functions for the Matérn family of covariances with $1/2 \le \nu < \infty$ and for the Gaussian covariances ($\nu = \infty$).

5. Application to the Matérn family of covariances

We now apply the generic procedure of Section 4 to the specific case of the Matérn family of covariances, which includes the Gaussian case as a limit case with $\nu = \infty$. We consider separately the Matérn case with $1/2 \le \nu < \infty$ and the Gaussian case $\nu = \infty$. For the sake of clarity, the name of the fitting function $\mathcal{F}$ is replaced by $\mathcal{M}$ in the Matérn case with $1/2 \le \nu < \infty$ (and $m_{\mathcal{M},i}^{\text{est}}$ is the estimated number of points) and by $\mathcal{G}$ in the Gaussian case (and $m_{\mathcal{G},i}^{\text{est}}$ the estimated number of points). Our new

initialization requires to estimate the starting number of points $m^{\text{est}}_{\mathcal{M},i}$ and $m^{\text{est}}_{\mathcal{G},i}$, $i = 1, \ldots, d$ according to the steps described at subsection 4.2.

5.1. Matérn case ($1/2 \leq \nu < \infty$): estimation of $m^{\text{est}}_{\mathcal{M},i}$

To design the fitting function $\mathcal{M}$, we propose to combine the strategy proposed in 4.2 with the theoretical work from [9] (steps (S1) and (S2) of subsection 4.2).

Theorem 2.9 in [9] provides a lower bound for the minimum extended domain length for the isotropic Matérn family of covariances ($1/2 \leq \nu < \infty$) on a unit cube domain ($L_i = L = 1$, $m_{0,i} = m_0$, $h_{0,i} = h_0 = 1/m_0$, $\forall i$):

Theorem 5.1 (From [9], Theorem 2.9)**.** *According to* [9]*, for a domain* $[0;1]^d$*, consider the isotropic Matérn covariance family, with smoothness parameter* ν *satisfying* $\frac{1}{2} \leq \nu < \infty$ *and the correlation length* $\lambda \leq 1$*. Suppose* $h_0/\lambda \leq e^{-1}$*. Then there exists positive constants* c_1 *and* $c_2 \geq 2\sqrt{2}$ *which may depend on the dimension* d *but are independent of the other parameters* ℓ, h_0, λ, ν *such that* $\mathbf{R}^{ext}$ *is positive definite if:*

$$\ell/\lambda \geq c_1 + c_2 \nu^{0.5} \log(\max(\lambda/h_0, \nu^{0.5})) \tag{5.1}$$

The extension of Theorem 5.1 to $\nu \in (0; 1/2)$ remains an open question. In practical applications, $\nu \geq 1/2$ [9].

From Theorem 5.1, let us look for a fitting function

$$\mathcal{M}(\nu, u, v) := \mathcal{H}(\nu, u/v)\, u \tag{5.2}$$

with

$$\mathcal{H}(\nu, w) := c_1^{\text{est}} + c_2^{\text{est}}(\nu)\, \nu^{0.5}\, \log(\max(w, \nu^{0.5})), \tag{5.3}$$

with c_1^{est} and $c_2^{\text{est}}(\nu)$ to be determined with numerical simulations.

In the isotropic covariances case and for a general domain size L, we will consider the function $\mathcal{M}(\nu, \lambda/L, h_0/L)$ (see (4.1)). In the anisotropic case on more general domain of size L_i, $i = 1, \ldots, d$, we consider $\mathcal{M}(\nu, \lambda_i/L_i, h_{0,i}/L_i)$ (see (4.3)). According to the numerical simulations in the next subsection, the constants c_1^{est} and c_2^{est} do not depend on λ_i/L_i or $h_{0,i}/L_i$ but c_2^{est} depends on ν in the 3D case.

To estimate the values of c_1^{est} and $c_2^{\text{est}}(\nu)$ of the function $\mathcal{H}$ defined by (5.3), we propose set of numerical experiments with Algorithm 2, with a start $m_i^{\text{start}} = m_{0,i}$, $i \in \{1, \ldots, d\}$ and an increment of 1 at Step 5, to obtain the minimum domain size $\ell^{\min}/\lambda$ versus the quantities

$$\zeta := \nu^{0.5} \log(\max(\lambda/h_0, \nu^{0.5})). \tag{5.4}$$

We consider different ratio $\lambda/h_0 \in \{2, 3, 4, 8, 16, 32, 64\}$ for the 2D case and $\lambda/h_0 \in \{2, 3, 4, 8, 16\}$ for the 3D case. We also consider $\nu \in \{0.5, 1, 2, 4\}$. For each ratio λ/h_0 and for each value of ν, we compute the value of ζ defined by (5.4). Then for each value of ζ, the value of $\ell^{\min}$ is obtained with Algorithm 2, with a start $m_i^{\text{start}} = m_{0,i}$, $i \in \{1, \ldots, d\}$ and an increment of 1 at Step 5. For each set of simulations, the value of the threshold τ is given.

Remark 5.2. Notice the values of $\ell^{\min}$ estimated here depend on the value of τ. The larger the absolute value of τ, the smaller $\ell^{\min}$. It also depends on the floating point precision of the software. Here the 80-bit precision software `ParaCirce` is used.

Figure 4 displays the ratio $\ell^{\min}/\lambda$ versus ζ for different values of ν in the 2D case (left) and in the 3D case (right).

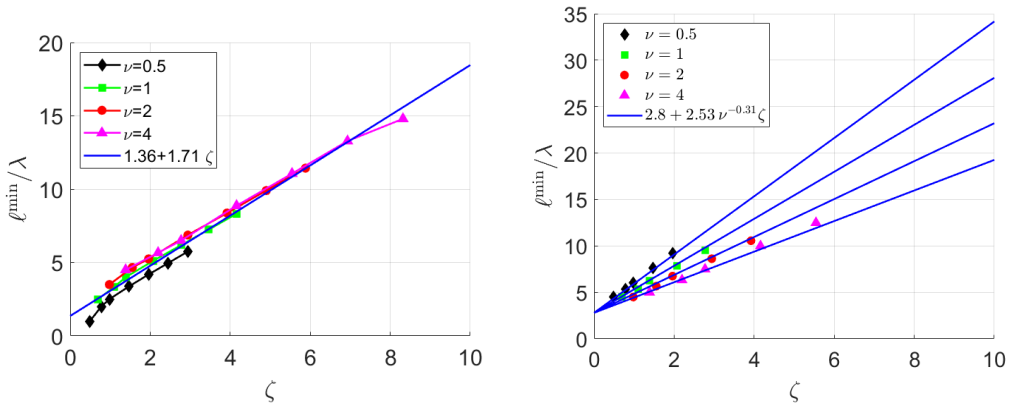

FIGURE 4. Isotropic Matérn covariances - $\ell^{\min}/\lambda$ versus $\zeta = \nu^{0.5}\log(\max(\lambda/h_0, \nu^{0.5}))$ ($\tau = -10^{-13}$) : (left) 2D; (right) 3D.

In 2D (left figure), the line that best fits all the data (in a least-squares sense) has the equation $c_1^{\text{est}} + c_2^{\text{est}}(\nu)\zeta$ with $c_1^{\text{est}} = 1.36$ and $c_2^{\text{est}}(\nu) = 1.71$ (blue line). Note that the estimated length in the case $\nu = 0.5$ are slightly overestimated. In 2D, c_2^{est} is a constant, independent of ν.

In 3D (right figure), contrary to the 2D case, here we observe more differences in the slopes of the curves with respect to ν. The slope depends on ν. A good fit of the slope is a power fit function: $c_2^{\text{est}} = 2.53\nu^{-0.31}$ (blue curves).

To sum up, our numerical experiments suggest the following parameters c_1^{est} and $c_2^{\text{est}}(\nu)$ of the fitting function $\mathcal{H}$ defined by (5.3):

- [`2D case:`] $c_1^{\text{est}} = 1.36$ and $c_2^{\text{est}} = 1.71$;
- [`3D case:`] $c_1^{\text{est}} = 2.80$ and $c_2^{\text{est}}(\nu) = 2.53\nu^{-0.31}$.

Remark 5.3. These estimates c_1^{est} and $c_2^{\text{est}}(\nu)$ could be improved by adding more simulations.

Now we have the fitting function $\mathcal{M}$ defined by (5.2) and (5.3) and the previous values c_1^{est} and $c_2^{\text{est}}(\nu)$, we are be able to compute the estimation $m_{\mathcal{M},i}^{\text{est}}$ thanks to (4.3) and (4.4).

5.2. Gaussian case ($\nu = \infty$): estimation of $m_{\mathcal{G},i}^{\text{est}}$

For the Gaussian case, Theorem 2.11 in [9] provides a lower bound on the required domain size but the bound is quite loose, hence it will not be used to design the fitting function $\mathcal{G}$.

We found numerically that $\ell^{\min}/\lambda$ varies linearly in λ/h_0. Hence we define the following function $\mathcal{G}$ as:

$$\mathcal{G}(u, v) := \mathcal{J}(u/v)\, u, \tag{5.5}$$

with

$$\mathcal{J}(w) := \alpha_1^{\text{est}} w + \alpha_2^{\text{est}} \tag{5.6}$$

where α_1^{est} and α_2^{est} are constants that will be numerically estimated in the following subsections.

We will consider the function $\mathcal{G}(\lambda/L, h_0/L)$ in the isotropic covariances case (see (4.1)) and $\mathcal{G}(\lambda_i/L_i, h_{0,i}/L_i)$ in the anisotropic case (see (4.3)).

To estimate the values of α_1^{est} and α_2^{est} in the definition (5.6) of $\mathcal{J}$, we propose a set of numerical experiments with Algorithm 2 (with an increment of 1 and a starting point $m_i^{\text{start}} = m_{0,i}$, $i = \{1, \dots d\}$) to obtain the minimum domain size $\ell^{\min}/\lambda$ for different ratio λ/h_0.

The computed values of $\ell^{\min}/\lambda$ obtained with Algorithm 2 (with an increment of 1 and a start at $m_i^{\text{start}} = m_{0,i}$), for different ratio λ/h_0 are given in Table 1. In 2D, τ is chosen equal to -10^{-13}.

λ/h_0	2	2.5	3	3.5	4	6	8	10	16	32	64
$\ell^{\min}/\lambda$	5.50	7.60	8.00	8.00	8.25	8.17	8.13	8.20	8.31	8.44	8.59

Table 1. Ratio $\ell^{\min}/\lambda$ versus λ/h_0 according to Algorithm 2 with an increment of 1 and a start at $m_i^{\text{start}} = m_{0,i}$ – 2D case : Gaussian covariance ($\tau = -10^{-13}$).

The computed values of $\ell^{\min}/\lambda$ obtained with Algorithm 2 (with an increment of 1 and a start at $m_i^{\text{start}} = m_{0,i}$) for different ratio λ/h_0 are given in Table 2. In 3D, τ is chosen equal to -5.10^{-13} to avoid, as much as possible, the stagnation of the minimum eigenvalue as depicted in paragraph A.

λ/h_0	2	2.5	3	3.5	4	6	8	10	16	32
$\ell^{\min}/\lambda$	5.50	7.60	8.33	8.29	8.25	8.33	8.38	8.40	8.56	8.78

Table 2. Ratio $\ell^{\min}/\lambda$ versus λ/h_0 according to Algorithm 2 with an increment of 1 and a start at $m_i^{\text{start}} = m_{0,i}$ – 3D case: Gaussian covariance ($\tau = -5.10^{-13}$).

Figure 5 displays the ratio $\ell^{\min}/\lambda$ versus λ/h_0 given in Table 1 (2D, left figure) and Table 2 (3D, right figure), together with the fitting functions.

Both in 2D and in 3D, we have identified a quadratic behavior of $\ell^{\min}/\lambda$ for small ratio λ/h_0 and a linear behavior for larger ratio λ/h_0. We propose to consider only a linear fitting (in a least-squares sense) of the data $\ell^{\min}/\lambda$ corresponding to $\lambda/h_0 \geq 3$ which gives the following constants:

- [`2D case:`] $\alpha_1^{\text{est}} = 8.69\,10^{-3}$, $\alpha_2^{\text{est}} = 8.09$,
- [`3D case:`] $\alpha_1^{\text{est}} = 1.76\,10^{-2}$, $\alpha_2^{\text{est}} = 8.23$.

Other choices could have been made, especially to propose a quadratic fitting for $\lambda/h_0 < 3$. But as the ratio λ/h_0 gives the number of points per correlation length, it is unlikely to be small in practical applications.

Now we have the fitting function $\mathcal{G}$ defined by (5.5) and (5.6) and the previous values α_1^{est} and α_2^{est}, we are be able to compute the estimation $m_{\mathcal{G},i}^{\text{est}}$ thanks to (4.3) and (4.4).

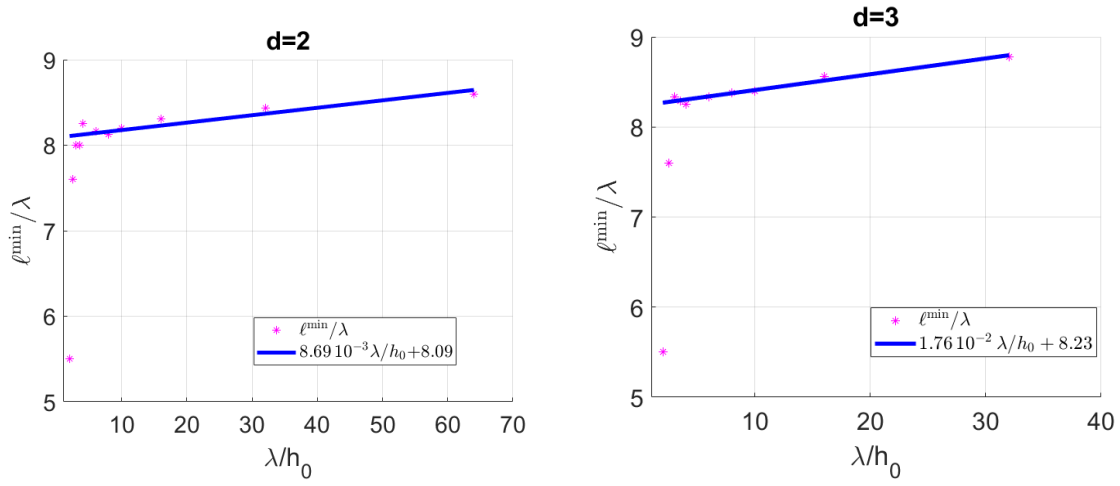

FIGURE 5. Minimum length $\ell^{\min}/\lambda$ (with $\ell^{\min}$ obtained with Algorithm 2) versus λ/h_0: (left) 2D case ($\tau = -10^{-13}$); (right) 3D case ($\tau = -5.10^{-13}$) – Gaussian covariance.

6. Numerical validation

In the following, we first consider the case of isotropic covariances on cubic domains and then the more general case of anisotropic covariances on d-dimensional rectangular parallelepipeds.

6.1. Validation criterion

The validation criterion is the number of iterations at Step 5 of Algorithm 2 needed to reach the required enlarged domain size. Situations of interest for our new initialization stage is when a padding is required (which means the initial domain is not large enough to guarantee the positivity of all eigenvalues).

6.2. Isotropic covariances on a cubic domain

In the isotropic case on a cubic domain of size L, a noticable advantage compared to the anisotropic case to analyze the efficiency of the proposed initialization, is that we can compute the exact minimum number of points $m^{\min}+1$ with $m^{\min} := \max(m_0, \ell^{\min}/h_0)$ using Algorithm 2 with an increment of 1 and a start at $m_i^{\text{start}} = m_{0,i}$, $i = \{1, \ldots, d\}$. Then we can evaluate the number of iterations $\#\text{It}_{\mathcal{F}}^{\text{step5}}$ needed at Step 5 with the new initialization technique defined as:

$$\#\text{It}_{\mathcal{F}}^{\text{step5}} := \max(m^{\min} - m_{\mathcal{F}}^{\text{est}}, 0)$$

with the fitting function $\mathcal{F}$. The fitting function $\mathcal{F}$ is denoted by $\mathcal{M}$ in the Matérn case with $1/2 \leq \nu < \infty$ and $\mathcal{G}$ in the Gaussian case. The value of $m_{\mathcal{M}}^{\text{est}}$ is computed according to (6.1) and the fitting function $\mathcal{M}$ defined by (5.2). The value of $m_{\mathcal{G}}^{\text{est}}$ is computed according to (6.2) and the fitting function $\mathcal{G}$ defined by (5.5). By comparison, the number of iterations at Step 5 performed with the classical initialization is:

$$\#\text{It}_{\text{classic}}^{\text{step5}} := m^{\min} - m_0.$$

If $\#\text{It}_{\mathcal{F}}^{\text{step5}} > 0$, $m^{\min}$ is underestimated and the difference gives the number of iterations required by Algorithm 2 to stop.

If $\#\mathrm{It}_{\mathcal{F}}^{\mathrm{step5}} = 0$, $m^{\min}$ is either estimated exactly or overestimated and Algorithm 2 stops with no iteration. In the case of an overestimation of the domain, Algorithm 2 does some extra-work (as the extended domain is overestimated) but we expect the percentage of overestimation

$$p_o := \max(0, \frac{m_{\mathcal{F}}^{\mathrm{est}} - m^{\min}}{m^{\min}})(\%)$$

to remain relatively small.

Our fittings are considered satisfactory when $\#\mathrm{It}_{\mathcal{F}}^{\mathrm{step5}}$ is zero or close to zero and p_o is small.

Remark 6.1. Each iteration at Step 3 of Algorithm 2 costs an FFT, which is likely to be costly (more costly than increasing p_o) for large domains or if the padding is large. A possibility to further reduce $\#\mathrm{It}_{\mathcal{F}}^{\mathrm{step5}}$, and then the number of FFTs, is to multiply $m_{\mathcal{F}}^{\mathrm{est}}$ by a given percentage to initialize Algorithm 2 with an overestimate of the domain. This will increase p_o but will likely decrease $\#\mathrm{It}_{\mathcal{F}}^{\mathrm{step5}}$.

6.2.1. *Matérn case, $1/2 \leq \nu < \infty$: quality of the estimates $m_{\mathcal{M}}^{est}$*

According to the definition of $\mathcal{M}$ (see (5.2)), the expression of $m_{\mathcal{M}}^{\mathrm{est}}$ (4.2) reduces to:

$$m_{\mathcal{M}}^{\mathrm{est}} = \max(m_0, \mathcal{H}(\nu, \lambda/h_0)\, \lambda/h_0). \tag{6.1}$$

The parameters are chosen to cover various ratio λ/h_0 used in subsection 5.1 and also some ratio λ/h_0 that were not used in the fitting (in 2D: $\lambda/h_0 = 24$ and $\lambda/h_0 = 128$; in 3D: $\lambda/h_0 = 10$ and $\lambda/h_0 = 24$). Table 3 displays, for (A) $d = 2$ and (B) $d = 3$, $m^{\min}$, $m_{\mathcal{M}}^{\mathrm{est}}$, $\#\mathrm{It}_{\mathcal{M}}^{\mathrm{step5}}$ and the percentage p_o.

ν	λ/h_0	$m^{\min}$	$m_{\mathcal{M}}^{\mathrm{est}}$	$\#\mathrm{It}_{\mathcal{M}}^{\mathrm{step5}}$	$p_o(\%)$
0.5	16	67	76	0	13.43%
	24	111	125	0	12.61%
	64	368	409	0	11.14%
	128	833	926	0	11.16%
1	16	99	98	1	0.00%
	24	163	164	0	0.61%
	64	533	543	0	1.88%
	128	1202	1237	0	2.91%
2	16	134	130	4	0.00%
	24	223	218	5	0.00%
	64	732	731	1	0.00%
	128	1656	1676	0	1.21%
4	16	177	174	3	0.00%
	24	297	294	3	0.00%
	64	947	998	0	5.39%
	128	1871*	2299	0	22.9%

(A) 2D

ν	λ/h_0	$m^{\min}$	$m_{\mathcal{M}}^{\mathrm{est}}$	$\#\mathrm{It}_{\mathcal{M}}^{\mathrm{step5}}$	$p_o(\%)$
0.5	4	24	24	0	0.00%
	10	82	80	2	0.00%
	16	147	144	3	0.00%
	24	243	237	6	0.00%
1	4	25	26	0	4.00%
	10	84	87	0	3.57%
	16	152	158	0	3.95%
	24	251	261	0	3.98%
2	4	27	28	0	3.70%
	10	93	95	0	2.15%
	16	169	173	0	2.37%
	24	281	288	0	2.49%
4	4	30	30	0	0.00%
	10	108	104	4	0.00%
	16	200	191	9	0.00%
	24	335	319	16	0.00%

(B) 3D

Table 3. Minimum number of points $m^{\min}$ versus $m_{\mathcal{M}}^{\mathrm{est}}$, number of iterations at Step 5 of Algorithm 2 $\#\mathrm{It}_{\mathcal{M}}^{\mathrm{step5}}$ and percentage of domain overestimation p_o – Matérn covariance (A) 2D with $\tau = -10^{-13}$ except for *: $\tau = -3.10^{-13}$ (see Appendix A) and (B) 3D.

The fittings are good in 2D and in 3D as shown by the small number of iterations $\#\mathrm{It}_{\mathcal{M}}^{\mathrm{step5}}$. In 2D, the domain size is overestimated for $\nu = 0.5$ and for the case $\nu = 4$ and $\lambda/h_0 = 128$ (as expected from our fitting, see the black diamonds and the pink triangles on Figure 4 (left)). For the case $\lambda/h_0 = 128$ and $\nu = 4$, we also are in a situation described in Appendix A where the threshold must be changed for Algorithm 2 to terminate and we are also in the situation where Remark 5.2 applies. The fitting is better in 3D as the parameters take the dependence on ν into account.

6.2.2. *Gaussian case: quality of the estimates $m_{\mathcal{G}}^{est}$*

According to the definition of $\mathcal{G}$ (see (5.5)), the expression of $m_{\mathcal{G}}^{\mathrm{est}}$ (4.2) reduces to:

$$m_{\mathcal{G}}^{\mathrm{est}} = \max(m_0, \mathcal{J}(\lambda/h_0)\,\lambda/h_0). \tag{6.2}$$

The parameters are chosen to cover the range of ratio λ/h_0 proposed in Table 1 and Table 2 and also some ratio λ/h_0 that were not used in the fitting (in 2D: $\lambda/h_0 = 24$ and $\lambda/h_0 = 128$; in 3D: $\lambda/h_0 = 24$ and $\lambda/h_0 = 64$). Table 4 displays, for (A) $d = 2$ and (B) $d = 3$, the minimum number of points $m^{\min}$ obtained with Algorithm 2 with an increment of 1 and a start at $m_i^{\mathrm{start}} = m_{0,i}$, the number of points $m_{\mathcal{G}}^{\mathrm{est}}$ computed according to (6.2) and the difference $m^{\min} - m_{\mathcal{G}}^{\mathrm{est}}$.

λ/h_0	$m^{\min}$	$m_{\mathcal{G}}^{\mathrm{est}}$	$\#\mathrm{It}_{\mathcal{G}}^{\mathrm{step5}}$	$p_o(\%)$
3	24	25	0	4.17%
4	33	33	0	0.00%
6	49	49	0	0.00%
8	65	66	0	1.54%
10	82	82	0	0.00%
16	133	132	1	0.00%
24	201	200	1	0.00%
32	270	268	2	0.00%
64	550	554	0	0.73%
128	1121	1178	0	5.08%

(A) 2D

λ/h_0	$m^{\min}$	$m_{\mathcal{G}}^{\mathrm{est}}$	$\#\mathrm{It}_{\mathcal{G}}^{\mathrm{step5}}$	$p_o(\%)$
3	25	25	0	0.00%
4	33	34	0	3.03%
6	50	51	0	2.00%
8	67	67	0	0.00%
10	84	85	0	1.19%
16	137	137	0	0.00%
24	204*	208	0	1.96%
32	281	282	0	0.36%
64	574**	600	0	4.53%

(B) 3D

TABLE 4. $m^{\min}$ versus $m_{\mathcal{G}}^{\mathrm{est}}$, number of iterations at Step 5 of Algorithm 2 $\#\mathrm{It}_{\mathcal{G}}^{\mathrm{step5}}$ and the percentage of domain overestimation p_o - Gaussian covariance (A) 2D ($\tau = -10^{-13}$) and (B) 3D ($\tau = -5.10^{-13}$ except for *: $\tau = -4.10^{-12}$ and for **: $\tau = -8.10^{-13}$, see Appendix A)

The fittings are good, both in 2D and in 3D: for each case, the number of points $m_{\mathcal{G}}^{\mathrm{est}}$ is very close to $m^{\min}$ whatever the ratio λ/h_0 and the percentage of overestimation of the domain p_o is low.

6.2.3. *Gain with the new initialization by comparison with the classical one*

The difference in number of iterations between a classical start at $m_0 + 1$ and an new start at $m_{\mathcal{F}}^{\mathrm{est}} + 1$ is given by the quantity:

$$\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}} := It_{\mathrm{classic}}^{\mathrm{step5}} - It_{\mathcal{F}}^{\mathrm{step5}}.$$

If $\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}}$ is zero, no iteration is saved (for cases with no padding for example). If $\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}}$ is greater than zero, it gives the number of forward FFT saved with our new initialization.

Let us take some examples. Table 5, 6 and 7 display the number of saved iterations $\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}}$ for different values of h_0/L and λ/L for the Matérn 2D, for the Matérn 3D and for the Gaussian covariances respectively.

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	0	7
1/8	0	0	3	19
1/16	0	0	11	51
1/32	0	0	35	127
1/64	0	9	95	304

(A) $\nu = 0.5$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	1	11
1/8	0	0	7	32
1/16	0	0	24	82
1/32	0	8	66	200
1/64	0	34	168	469

(B) $\nu = 1$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	3	15
1/8	0	0	11	44
1/16	0	3	36	114
1/32	0	20	98	280
1/64	0	66	248	667

(C) $\nu = 2$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	4	21
1/8	0	0	17	60
1/16	0	9	52	158
1/32	0	36	142	391
1/64	4	110	359	883

(D) $\nu = 4$

Table 5. Number of iterations saved $\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}}$ for different values of h_0/L and λ/L - 2D Matérn covariance ($\tau = -10^{-13}$):

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	5	20
1/8	0	1	16	52
1/16	0	8	44	128
1/32	0	28	112	304

(A) $\nu = 0.5$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	5	22
1/8	0	1	18	55
1/16	0	10	47	136
1/32	0	31	120	324

(B) $\nu = 1$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	5	23
1/8	0	1	19	61
1/16	0	11	53	153
1/32	0	37	137	368

(C) $\nu = 2$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	0	6	26
1/8	0	2	22	70
1/16	0	14	62	175
1/32	0	46	159	423*

(D) $\nu = 4$

Table 6. Number of iterations saved $\#\mathrm{It}_{\mathrm{saved}}^{\mathrm{step5}}$ for different values of h_0/L and λ/L - 3D Matérn covariance ($\tau = -10^{-13}$, except for *: $\tau = -1.7\,10^{-12}$, see Appendix A)

We observe that our new initialization helps in saving more iterations as the ratio λ/L increases. The same can be observed when the ratio h_0/L increases. For large ratio λ/L and h_0/L, the saving is

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	3	10	29
1/8	0	7	25	57
1/16	0	17	49	116
1/32	1	33	100	236
1/64	1	68	204	486

(A) $2D$

h_0/L \ λ/L	0.125	0.25	0.5	1
1/4	0	3	10	29
1/8	0	7	25	59
1/16	0	17	51	121
1/32	1	35	105	249
1/64	3	73	217	510*

(B) $3D$

TABLE 7. Number of iterations saved $\#\text{It}^{\text{step5}}_{\text{saved}}$ for different values of h_0/L and λ/L – Gaussian covariance (A) 2D ($\tau = -10^{-13}$) and (B) 3D ($\tau = -5.10^{-13}$ except for *: $\tau = -8.10^{-13}$, see Appendix A)

the most important. The number of saved iterations also increases with ν for the Matérn covariances and is greater in 3D than in 2D for the same parameters.

6.3. **Anisotropic covariance function on a general domain**

Let us now consider the more general case of anisotropic covariance function on d-dimensional rectangular parallelepipeds. We do not have access to a required length $m^{\min}$ anymore, as Algorithm 2 suggests to increase the domain equally in each direction, while it is possible to have some padding in only some directions.

As we do not have access to $m^{\min}$, we cannot estimate the percentage of overestimation of the domain p_o but we can compare the number of iterations $\#\text{It}^{\text{step5}}_{\mathcal{F}}$ and $\#\text{It}^{\text{step5}}_{classic}$ needed at Step 5 of Algorithm 2 to reach the required enlarged domain size, for the new and classical initialization steps respectively. Zero iteration means the domain size is large enough.

6.3.1. *Parameters*

In the previous section, we tested the same padding in all directions. Now we will compare the two initializations for test cases with a requested padding in one direction only. Table 8 gives the 2D parameters. Table 9 gives the 3D parameters. For example, let us choose direction 1 as the direction where a padding is required, with two values for the ratio λ_1/L, a moderate one ($\lambda_1/L = 0.5$) and a large one ($\lambda_1/L = 1$), and two discretization steps: $h_{0,1}/L = 1/8$ ($m_{0,1} = 8$) and $h_{0,1}/L = 1/32$. The other directions will have a small value (0.125) of the correlation length with respect to the domain size and a small discretization step (1/8). The name are formatted like $A_{\lambda_1/L;h_{0,1}/L}$ where λ_1/L and $h_{0,1}/L$ are replaced by their respective values.

name	λ_1/L	$h_{0,1}/L$	λ_2/L	$h_{0,2}/L$
$A^{2D}_{0.5;1/8}$	0.5	1/8	0.125	1/8
$A^{2D}_{0.5;1/32}$	0.5	1/32	0.125	1/8
$A^{2D}_{1;1/8}$	1	1/8	0.125	1/8
$A^{2D}_{1;1/32}$	1	1/32	0.125	1/8

TABLE 8. Anisotropic test cases - 2D parameters.

name	λ_1/L	$h_{0,1}/L$	λ_2/L	$h_{0,2}/L$	λ_3/L	$h_{0,3}/L$
$A^{2D}_{0.5;1/8}$	0.5	1/8	0.125	1/8	0.125	1/8
$A^{2D}_{0.5;1/32}$	0.5	1/32	0.125	1/8	0.125	1/8
$A^{2D}_{1;1/8}$	1	1/8	0.125	1/8	0.125	1/8
$A^{2D}_{1;1/32}$	1	1/32	0.125	1/8	0.125	1/8

Table 9. Anisotropic test cases - 3D parameters.

6.3.2. *Numerical results*

Table 10 and Table 11 give the number of iterations at Step 5 of Algorithm 2 for the Matérn family of covariances (for $\nu = 1$ and $\nu = 4$) and for the Gaussian covariances respectively, for the two possible initializations: $m_i^{\text{start}} = m_{0,i}$ and $m_i^{\text{start}} = m^{\text{est}}_{\mathcal{M},i}$ (respectively $m_i^{\text{start}} = m^{\text{est}}_{\mathcal{G},i}$), $i = \{1, \dots, d\}$, $d = 2$ or $d = 3$. We indicate, in parenthesis, the minimim $(m^{\min})_i$ obtained with the classical initialization as well as $(m^{\text{est}}_{\mathcal{M},i})_i$ and $(m^{\text{est}}_{\mathcal{G},i})_i$ obtained with the fitting functions $\mathcal{M}$ and $\mathcal{G}$ respectively.

name	$\#\text{It}^{\text{step5}}_{classic}$ $(m^{\min})_i$	$\#\text{It}^{\text{step5}}_{\mathcal{M}}$ $(m^{\text{est}}_{\mathcal{M},i})_i$
$A^{2D}_{0.5;1/8}$	5 (13,13)	0 (15,8)
$A^{2D}_{0.5;1/32}$	35 (67,43)	0 (98,8)
$A^{2D}_{1;1/8}$	21 (29,29)	0 (40,8)
$A^{2D}_{1;1/32}$	119 (151,127)	0 (234,8)

(A) 2D, $\nu = 1$

name	$\#\text{It}^{\text{step5}}_{classic}$ $(m^{\min})_i$	$\#\text{It}^{\text{step5}}_{\mathcal{M}}$ $(m^{\text{est}}_{\mathcal{M},i})_i$
$A^{2D}_{0.5;1/8}$	17 (25,25)	0 (25,8)
$A^{2D}_{0.5;1/32}$	133 (165,141)	0 (174,8)
$A^{2D}_{1;1/8}$	59 (67,67)	0 (68,8)
$A^{2D}_{1;1/32}$	359 (391,367)	0 (423,8)

(B) 2D, $\nu = 4$

name	$\#\text{It}^{\text{step5}}_{classic}$ $(m^{\min})_i$	$\#\text{It}^{\text{step5}}_{\mathcal{M}}$ $(m^{\text{est}}_{\mathcal{M},i})_i$
$A^{3D}_{0.5;1/8}$	11 (19,19,19)	0 (26,8,8)
$A^{3D}_{0.5;1/32}$	56 (88,64,64)	0 (158,8,8)
$A^{3D}_{1;1/8}$	32 (40,40,40)	0 (65,8,8)
$A^{3D}_{1;1/32}$	163 (195,171,171)	0 (371,8,8)

(C) 3D, $\nu = 1$

name	$\#\text{It}^{\text{step5}}_{classic}$ $(m^{\min})_i$	$\#\text{It}^{\text{step5}}_{\mathcal{M}}$ $(m^{\text{est}}_{\mathcal{M},i})_i$
$A^{3D}_{0.5;1/8}$	20 (28,28,28)	0 (30,8,8)
$A^{3D}_{0.5;1/32}$	144 (176,152,152)	0 (191,8,8)
$A^{3D}_{1;1/8}$	64 (72,72,72)	0 (78,8,8)
$A^{3D}_{1;1/32}$	381 (413,389,389)	0 (455,8,8)

(D) 3D, $\nu = 4$

Table 10. Number of iterations at Step 5 of Algorithm 2 for the classical initialization $\#\text{It}^{\text{step5}}_{classic}$ and for the new initialization $\#\text{It}^{\text{step5}}_{\mathcal{M}}$, enlarged domain size $(m^{\min})_i$ and estimated domain size $(m^{\text{est}}_{\mathcal{M},i})_i$ in parenthesis – Matérn covariance ($\tau = -10^{-13}$) (A) and (B) 2D case, (C) and (D) 3D case for $\nu = 1$ (left tables) and $\nu = 4$ (right tables).

No extra-iteration is needed for both Matérn and Gaussian covariances with our new initialization, hence saving important computational resources, compared with the classical initialization stage. Moreover the extended domain is much smaller with our new initialization.

7. Conclusion

We have proposed a new initialization step that computes fairly good guesses of the required extended domain size for the CEM algorithm to work. Theorem 3.1 guarantees that Algorithm 2 always terminates. Several test cases have been conducted to check the efficiency of these new algorithms in

name	$\#\mathrm{It}^{\mathrm{step5}}_{classic}\ (m^{\min})_i$	$\#\mathrm{It}^{\mathrm{step5}}_{\mathcal{G}}(m^{\mathrm{est}}_{\mathcal{G},i})_i$
$A^{2D}_{0.5;1/8}$	24 (32,32)	0 (33,9)
$A^{2D}_{0.5;1/32}$	95 (127,103)	0 (132,9)
$A^{2D}_{1;1/8}$	55 (63,63)	0 (66,9)
$A^{2D}_{1;1/32}$	225 (257,233)	0 (268,9)

(A) $2D$

name	$\#\mathrm{It}^{\mathrm{step5}}_{classic}\ (m^{\min})_i$	$\#\mathrm{It}^{\mathrm{step5}}_{\mathcal{G}}(m^{\mathrm{est}}_{\mathcal{G},i})_i$
$A^{3D}_{0.5;1/8}$	23 (31,31,31)	0 (34,9,9)
$A^{3D}_{0.5;1/32}$	94 (126,102,102)	0 (137,9,9)
$A^{3D}_{1;1/8}$	55 (63,63,63)	0 (67,9,9)
$A^{3D}_{1;1/32}$	222 (254,230,230)	0 (282,9,9)

(B) $3D$

TABLE 11. Number of iterations at Step 5 of Algorithm 2 for the classical initialization $\#\mathrm{It}^{\mathrm{step5}}_{classic}$ and for the new initialization $\#\mathrm{It}^{\mathrm{step5}}_{\mathcal{G}}$ and enlarged domain size $(m^{\min})_i$ and estimated domain size $(m^{\mathrm{est}}_{\mathcal{G},i})_i$ in parenthesis – Gaussian covariance (A) 2D ($\tau = -10^{-13}$) and (B) 3D ($\tau = -5.10^{-13}$).

terms of estimation of the enlarged domain size and number of iterations and the results are very good, hence saving a lot of computational resources by comparison with a classical initialization. This new initialization step is implemented in the `ParaCirce` library. Our current work is to study the performance of `ParaCirce`, especially its parallel efficiency, in order to provide all the necessary tools for large simulations of GRF with the CEM (typically in case of large ratio λ_i/L_i and small ratio $h_{0,i}/L_i$). In a recent paper [1], a better estimate for $\ell_{\min}$ is proven thanks to a smooth periodization of the Matérn family of covariance functions ($0 \leq \nu < \infty$): the estimate does not blow up with λ/h_0 and depends only on ν (see equation (1.11) in [1]). A nice future work would be to apply our fitting procedure to that case. Finally, another interesting work would be to propose a rule to automatically set the value of the threshold τ according to the input parameters and the floating point precision (in view of the stagnation of the minimum eigenvalue for some test cases as described in Appendix A).

Appendix A. Influence of the floating point precision on the computations

We have observed that finite precision effects may prevent the termination of Algorithm 2. Therefore, the numerical simulations of this paper have been performed using a 80-bit precision code. What happens is that, depending on the input parameters and the chosen numerical precision, the minimum eigenvalue might reach a plateau and potentially never reaches the given threshold τ. Using an extended numerical precision code shifts the plateau, that is why we prefer to use the 80-bit version of `ParaCirce` to avoid, as much as possible, this situation.

In the 2D example shown on Figure 6, the 80-bit precision is enough and Algorithm 2 stops for $\tau = -10^{-13}$.

For some other cases, for a given 80-bit version of `ParaCirce`, the threshold τ must be changed to a lower value for Algorithm 2 to stop. We have encountered this situation a few times in our numerical experiments, according to the threshold we have chosen. The simulations are the ones with the stars in the tables. For example, the case of the isotropic 3D Matérn covariance with $\nu = 4$, $\lambda/L = 1$ and $h_0/L = 1/32$ (Table 6) is shown on Figure 7. The case of the 3D isotropic Gaussian covariance with $\lambda/L = 1$ and $h_0/L = 1/64$ (Table 7) is shown on Figure 8.

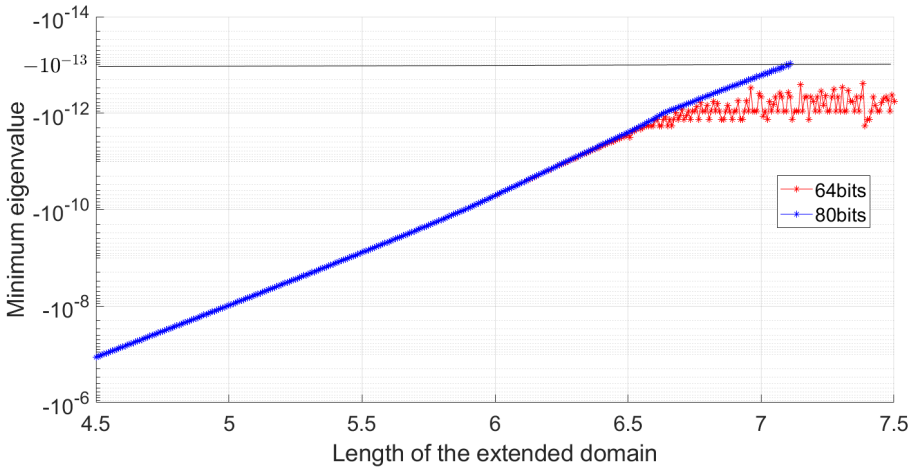

Figure 6. 2D case - isotropic Matérn covariance ($L = 1$, $\nu = 4, \lambda = 0.5, m_0 = 128$): Minimum eigenvalue versus the length of the extended domain

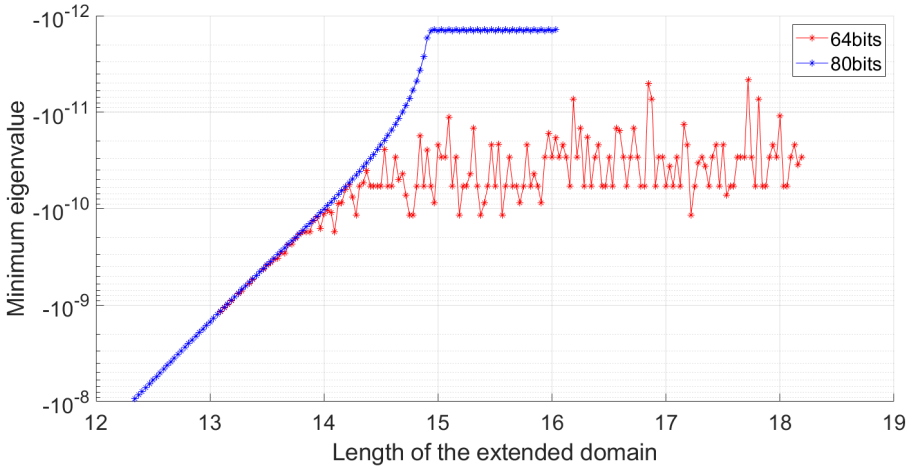

Figure 7. 3D case - isotropic Matérn covariance ($L = 1$, $\nu = 4, \lambda = 1, m_0 = 32$): Minimum eigenvalue versus the length of the extended domain

Acknowledgements

The authors warmly thank Jocelyne Erhel (Inria, France) and Ivan Graham (University of Bath, United Kingdom) for fruitful discussions related to the Circulant Embedding Method. The authors are grateful to the CLEPS infrastructure from the Inria of Paris for providing resources and support.

Bibliography

[1] M. Bachmayr, I. G. Graham, V. K. Nguyen, and R. Scheichl. Unified analysis of periodization-based sampling methods for Matérn covariances. *SIAM Journal on Numerical Analysis*, 58(5):2953–2980, 2020.

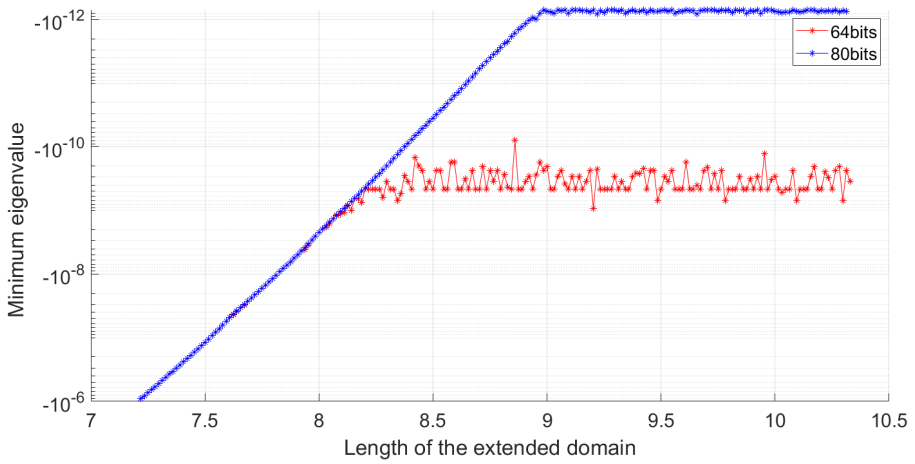

FIGURE 8. 3D case - isotropic Gaussian covariance ($L = 1$, $\lambda = 1, m_0 = 64$): Minimum eigenvalue versus the length of the extended domain

[2] A. Beaudoin and J.-R. de Dreuzy. Numerical assessment of 3-D macrodispersion in heterogeneous porous media. *Water Resources Research*, 49(5):2489–2496, 2013.

[3] X. H. Dang. *Identification de la variabilité spatiale des champs de contraintes dans les agrégats polycristallins et application à l'approche locale de la rupture*. Theses, Université Blaise Pascal - Clermont-Ferrand II, October 2012.

[4] C. R. Dietrich and G. N. Newsam. A fast and exact method for multidimensional Gaussian stochastic simulations. *Water Resources Research*, 29(8):2861–2869, 1993.

[5] C. R. Dietrich and G. N. Newsam. Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. *SIAM J. Sci. Comput.*, 18(4):1088–1107, 1997.

[6] M. Feischl, F.Y. Kuo, and I.H. Sloan. Fast random field generation with H-matrices. *Numer. Math.*, 140:639–676, 2018.

[7] M. Frigo and S. G. Johnson. The design and implementation of FFTW3. *Proc. IEEE 93*, 2:216–231, 2005.

[8] I. G. Graham, F. Y. Kuo, D. Nuyens, R. Scheichl, and I. H. Sloan. Quasi-Monte Carlo methods for elliptic PDEs with random coefficients and applications. *Journal of Computational Physics*, 230(10):3668–3694, 2011.

[9] I. G. Graham, F. Y. Kuo, D. Nuyens, R. Scheichl, and I. H. Sloan. Analysis of Circulant Embedding Methods for Sampling Stationary Random Fields. *SIAM Journal on Numerical Analysis*, 56(3):1871–1895, January 2018.

[10] P. L'Ecuyer, R. Simard, E. J. Chen, and W. D. Kelton. An objected-oriented random-number package with many long streams and substreams. *Operations Research*, 50(6):1073–1075, 2002.

[11] Y. Liu, J. Li, S. Sun, and B. Yu. Advances in Gaussian random field generation: a review. *Comput Geosci*, 23:1011–1047, 2019.

[12] G. J. Lord, C. E. Powell, and T. Shardlow. *An Introduction to Computational Stochastic PDEs*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2014.

[13] C. Schwab and R. A. Todor. Karhunen–Loève approximation of random fields by generalized fast multipole methods. *Journal of Computational Physics*, 217(1):100–122, 2006. Uncertainty Quantification in Simulation Science.