Spatial best linear unbiased prediction: A computational mathematics approach for high dimensional massive datasets

Julio Enrique Castrillón-Candás[1*]

[1*]Department of Mathematics and Statistics, Boston University, 665 Commonwealth Avenue, Boston, 02215, MA, USA.

Corresponding author(s). E-mail(s): jcandas@bu.edu;

Abstract

With the advent of massive data sets much of the computational science and engineering community has moved toward data-intensive approaches in regression and classification. However, these present significant challenges due to increasing size, complexity and dimensionality of the problems. In particular, covariance matrices in many cases are numerically unstable and linear algebra shows that often such matrices cannot be inverted accurately on a finite precision computer. A common ad hoc approach to stabilizing a matrix is application of a so-called nugget. However, this can change the model and introduce error to the original solution. *It is well known from numerical analysis that ill-conditioned matrices cannot be accurately inverted.* In this paper we develop a multilevel computational method that scales well with the number of observations and dimensions. A multilevel basis is constructed adapted to a kD-tree partitioning of the observations. Numerically unstable covariance matrices with large condition numbers can be transformed into well conditioned multilevel ones without compromising accuracy. Moreover, it is shown that the multilevel prediction *exactly* solves the Best Linear Unbiased Predictor (BLUP) and Generalized Least Squares (GLS) model, but is numerically stable. The multilevel method is tested on numerically unstable problems of up to 25 dimensions. Numerical results show speedups of up to 42,050 times for solving the BLUP problem, but with the same accuracy as the traditional iterative approach. For very ill-conditioned cases the speedup is infinite. In addition, decay estimates of the multilevel covariance matrices are derived based on high dimensional interpolation techniques from the field of numerical analysis.

This work lies at the intersection of statistics, uncertainty quantification, high performance computing and computational applied mathematics.

Keywords: Hierarchical Basis, Best Linear Unbiased Prediction, High Performance Computing, Uncertainty Quantification

MSC Classification: 65C99 , 65F99 , 65F25 , 65F35 , 65-04 , 60G15 , 60G25 , 62-08 , 62H99

1 Introduction

Massive data sets arise from many fields, including, but not limited to commerce, astrophysical sky-surveys, environmental data, medical data, and tsunami warning systems. With the advent of massive datasets, much of the computational science and engineering community has moved toward data-intensive approaches in regression and classification. However, these present significant challenges due to increasing size, complexity, and dimensionality of the problems.

Best Linear Unbiased Prediction (BLUP), or sometimes referred also the best linear unbiased predictor is a well known technique in earth and environmental sciences [1, 2]. It was originally developed by Henderson [3, 4] in the context of biosciences and biostatistics. It is also popular in Longitudinal analysis. This field is very significant to gerontology as well as the biomedical, and behavioral and social sciences [5, 6]. Stein's book, referenced as [7], provides valuable insights into the topic of BLUPs.

Since solving for the BLUP requires inverting the covariance matrix, this in general requires $\mathcal{O}(N^3)$ computational steps and $\mathcal{O}(N^2)$ memory [1]. For massive datasets this quickly becomes intractable since: (I) The covariance matrix becomes too large, and (II) for spatial covariance functions, the problem is further compounded by ill-conditioning of the covariance matrix. *It is known from linear algebra that ill-conditioned matrices cannot be accurately inverted with accuracy on finite precision computers [8] and thus are difficult, if not impossible, to solve numerically.*

A common technique to correct the ill-conditioned covariance matrix $\mathbf{C}$ is to add a scaled identity matrix $\mathbf{I}$, e.g. $\mathbf{C} + \sigma\mathbf{I}$, where $\sigma > 0$. The term $\sigma\mathbf{I}$ is a called a *nugget*. However, inverting the matrix $\mathbf{C} + \sigma\mathbf{I}$ is not equivalent to inverting $\mathbf{C}$. The solution of the BLUP will be incorrect. Thus a tradeoff between accuracy and numerical stability is commonly accepted since if a matrix is ill-conditioned, it cannot be accurately inverted.

Many techniques for inverting $\mathbf{C}$ concentrate on the first problem (I). They rely on sparsification and/or identifying low rank approximations. In the context of estimating the covariance function, many methods have been developed using skeletonization factorizations [9], low-rank [10] and Hierarchical Matrices (HM) [11–13] approaches. These methods are very promising. In particular, for the HM approaches they have been shown to be near optimal. They work well for low dimensions. However, as the dimensions increases the computational burden explodes with each dimension. However, they are still subject to ill-conditioning and usually a nugget is added to change the model to make it more numerically stable, but does not solve the original problem. Thus these approaches are limited to covariance matrices that are well conditioned. Moreover, the model of the data is assumed to have zero trend and a non-zero nugget. For many practical cases this will not be valid. Note that in [14, 15] the authors developed an approach for constructing sparse positive definite kernel matrices with improved numerical stability.

In the approach developed in [16] it is shown that there exists a more stable form of the solution of the BLUP and the Generalized Least Square (GLS) that solves these problems exactly. This approach is based on the work on multilevel discrete basis developed in [17] for the radial basis function interpolation problem for scattered data. A similar basis has been developed in [18]. These approaches are based on the idea of using wavelets to compress integral operators [19].

Although an ill-conditioned covariance matrix leads to accuracy problems, this can be avoided by constructing an alternative multilevel covariance matrix $\mathbf{C_W}$ that is used in the stable form. It is also shown how the covariance matrix can be sparsified so that the covariance function can be estimated from the data using a Maximum Likelihood Estimate method. Error estimates for the decay of the covariance function are derived using derivative information of the covariance function. However, this approach is limited to 2 or 3 dimensions. The computational cost scales combinatorially fast with the spatial dimension, thus making it impractical for high dimensional problems.

In this paper the approach from [16] is extended using binary trees, which are well suited for high dimensional problems. The multilevel basis used in [16] and originally proposed in [17] is extended to the high dimensional setting. Ill-conditioned covariance matrices are transformed to numerically stable multilevel covariance matrices without compromising accuracy. In addition, a new distance criterion is developed to build sparse multilevel covariance matrices. Furthermore, sharper decay estimates of the coefficients of the multivariate covariance matrix are derived based on analytic extensions that are well suited for high dimensional problems.

In the research presented in [16], the authors establish the decay rates of covariance matrix entries for multilevel matrices in $\mathbb{R}^d$ using Taylor's theorem. While this method is effective for lower-dimensional problems, its practicality diminishes as the dimensionality, denoted as d, increases. This is due to the combinatorial growth in the number of required derivatives in Taylor's theorem, alongside the expansion of the derivative domain concerning the dimension d. In cases involving high-dimensional scenarios, the computation of constants related to the derivatives in the results of [16] becomes increasingly challenging. In contrast, the complex analytic approach offers the advantage of uniformly bounding these constants, which depend on the region of analytic extension. As a result, in the field of uncertainty quantification for stochastic

Partial Differential Equations featuring high-dimensional random parameters, complex analyticity is favored [20–24]. This approach is adopted in this paper.

The MLE estimation equations are transformed into a multilevel form based on the numerically stable multilevel covariance matrix. In practice a sparse version of the multilevel covariance matrix is used. A distance dependent method is used to build to a sparse version. Sharp decay estimates (sub-exponential) of the multilevel covariance matrices are derived using complex analytic extensions of the covariance function instead of Taylor series expansions, which are infeasible for relatively large dimensional problems. The numerical results show that the estimation is solved to good accuracy for a large number of observations.

The BLUP prediction step is remapped into an equivalent multilevel formulation that is numerically stable. It is shown that the solution to the multilevel prediction form *exactly* solves the BLUP problem. To my knowledge, this is a feature that is unique to the multilevel approach. If the covariance matrix $\mathbf{C}$ is ill-conditioned, then it is not possible to solve the problem accurately on a computer with a fixed machine precision. However, the BLUP solution arises from a constrained optimization problem. By taking advantage of this fact, the multilevel approach side steps the inversion of the covariance matrix and directly searches for the solution in a constrained space giving rise to a significantly more stable multilevel covariance matrix. Moreover, by using an iterative approach only one indirect matrix inversion of the multilevel covariance $\mathbf{C_W}$ matrix is required. This is in contrast to classical BLUP, including the Generalized Least Squares (GLS) prediction, that at least p indirect matrix inversions are required with an iterative approach, where p is the number of columns of the design matrix (See Remark 1 and 7). Numerical results show speedups of up to 42,050 for solving the BLUP problem to at least the same accuracy. This approach has been also applied for imputation of medical records [25].

In Section 2 the problem formulation is introduced. In section 3 it is shown how to construct the multilevel basis based on kd-trees. In section 4 the construction of the multilevel covariance matrix is discussed. In section 5 the multilevel estimator and predictor are formulated and numerical computational issues are discussed in section 6. In section 7 a mathematical analysis of the decay of the entries of the multilevel covariance matrix is developed. This section can also be skipped for the less mathematically inclined reader. In section 8 the multilevel BLUP method is tested on numerically unstable problems of up to 25 dimensions. Furthermore, a direct accuracy comparison is done with the traditional BLUP formulae. Highly ill-conditioned BLUP problems are solved to high accuracy. In the appendices all of the proofs are described in detail and in Appendix A a the multivariate polynomial interpolation based on complex analytic extensions is discussed. These results are used for to derive the decay of the entries of the multilevel covariance matrix.

2 Problem setup

Consider the following model for a Gaussian random field Z:

$$Z(\mathbf{x}) = \mathbf{k}(\mathbf{x})^{\mathrm{T}}\boldsymbol{\beta} + \varepsilon(\mathbf{x}), \qquad \mathbf{x} \in \mathbb{R}^d, \tag{1}$$

where d is the number of spatial dimensions, $\mathbf{k} : \mathbb{R}^d \to \mathbb{R}^p$ is a functional vector of the spatial location $\mathbf{x}$, $\boldsymbol{\beta} \in \mathbb{R}^p$ is an unknown vector of coefficients, and ε is a stationary mean zero Gaussian random field with parametric covariance function $\phi(\mathbf{x}, \mathbf{x}';\ \boldsymbol{\theta}) = \mathrm{cov}\{\varepsilon(\mathbf{x}), \varepsilon(\mathbf{x}')\}$ with an unknown vector of positive parameters $\boldsymbol{\theta} \in \mathbb{R}^w$, where w is the number of parameters.

Suppose that we obtain N observations and stack them in the data vector $\mathbf{Z} = (Z(\mathbf{x}_1), \ldots, Z(\mathbf{x}_N))^{\mathrm{T}}$ from locations $\mathbb{S} := \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where the elements in $\mathbb{S}$ are restricted such that the design matrix defined below, $\mathbf{X}$, has full column rank. Furthermore, without loss of generality all the locations in $\mathbb{S}$ are contained in the unit hypercube $\Gamma^d \equiv [-1, 1]^d$. Let $\mathbf{C}(\boldsymbol{\theta}) = \mathrm{cov}(\mathbf{Z}, \mathbf{Z}^{\mathrm{T}}) \in \mathbb{R}^{N\times N}$ be the covariance matrix of $\mathbf{Z}$ and assume it is positive definite for all $\boldsymbol{\theta} \in \mathbb{R}^w$. Define $\mathbf{X} = \big(\mathbf{k}(\mathbf{x}_1) \ldots \mathbf{k}(\mathbf{x}_N)\big)^{\mathrm{T}} \in \mathbb{R}^{N\times p}$ and assume it is of full rank p. Since the model (1) is a Gaussian random field, then from the samples of $\mathbb{S}$ the following vectorial model is obtained

$$\mathbf{Z} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{2}$$

where $\boldsymbol{\varepsilon}$ is a Gaussian random vector, $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}(\boldsymbol{\theta}))$ and $p < N$. The aim now is to: i) *Estimate* the unknown vectors $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$; and ii) *Predict* $Z(\mathbf{x}_0)$, where $\mathbf{x}_0$ is a new spatial location. These two tasks are particularly computationally challenging when the sample size N and number of dimensions d are large.

There is a very large literature on Gaussian process regression that deal with this problem. Please see [16] for a brief literature review. The unknown vectors $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ are estimated with the log-likelihood function $\ell(\boldsymbol{\beta}, \boldsymbol{\theta}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log\det\{\mathbf{C}(\boldsymbol{\theta})\} - \frac{1}{2}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})$. To reduce the dimensionality of the optimization problem, $\boldsymbol{\beta}$ is replaced with GLS estimate:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = \{\mathbf{X}^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}\mathbf{X}\}^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}\mathbf{Z}. \tag{3}$$

In general this is not a good choice, since replacing with the Maximum Likelihood Estimator (MLE) of $\boldsymbol{\theta}$ is prone to be biased [16].

For the prediction part, consider the BLUP $\hat{Z}(\mathbf{x}_0) = \lambda_0 + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{Z}$ where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_N)^{\mathrm{T}}$. The unbiased constraint implies $\lambda_0 = 0$ and $\mathbf{X}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{k}(\mathbf{x}_0)$. The minimization of the mean squared prediction error $\mathrm{E}[\{Z(\mathbf{x}_0) - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{Z}\}^2]$ under the unbiased constraint $\mathbf{X}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{k}(\mathbf{x}_0)$ yields

$$\hat{Z}(\mathbf{x}_0) = \mathbf{k}(\mathbf{x}_0)^{\mathrm{T}}\hat{\boldsymbol{\beta}} + \mathbf{c}(\boldsymbol{\theta})^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}(\mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}}), \tag{4}$$

where $\mathbf{c}(\boldsymbol{\theta}) = \mathrm{cov}\{\mathbf{Z}, Z(\mathbf{x}_0)\} \in \mathbb{R}^N$ and $\hat{\boldsymbol{\beta}}$ is defined in (3).

Remark 1 Notice that solving for $\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})$ requires computing $\mathbf{C}(\boldsymbol{\theta})^{-1}\mathbf{X}$. Since $\mathbf{X} \in \mathbb{R}^{N\times p}$ by using an iterative approach it would require p indirect inversions of the matrix $\mathbf{C}(\boldsymbol{\theta})^{-1}$.

Now, let $\alpha := (\alpha_1, \ldots, \alpha_d) \in \mathbb{Z}^d$, $|\alpha| := \alpha_1 + \cdots + \alpha_d$, $\mathbf{x} := [x_1, \ldots, x_d]$. For any $w \in \mathbb{N}_+$ (where $\mathbb{N}_+ := \mathbb{N} \cup \{0\}$) let $\mathcal{Q}^d_w$ be the set of Total Degree (TD) monomials $\{x_1^{\alpha_1} \ldots x_d^{\alpha_d} \mid |\alpha| \leq w\}$. The typical choice for the matrix $\mathbf{X}$ is to build it from the monomials of $\mathcal{Q}^d_w$ with cardinality $p(d, w) := \binom{d+w}{w}$.

The challenge is that the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ in many practical cases is ill-conditioned, leading to slow and inaccurate estimates of $\boldsymbol{\theta}$. Following the approach in [16] the data vector $\mathbf{Z}$ is transformed into decoupled multilevel description of the model (1). This multilevel representation leads to significant computational benefits, including numerical stability, when computing the multilevel predictor $\hat{Z}(\mathbf{x}_0)$ in (4) for large sample size N and high dimensions d. Note, that in this paper we shall refer to the *single level* approach to solving the estimation and prediction steps directly to the data $\mathbf{Z}$ and covariance matrix $\mathbf{C}(\boldsymbol{\theta})$.

3 Multilevel approach

The general approach of this paper and multilevel basis construction are now presented. We mostly follow the exposition laid out in [16]. The proof of Proposition 1 is repeated, but clarified with more details.

Let $\mathcal{P}^p(\mathbb{S})$ be the span of the columns of the design matrix $\mathbf{X}$. Suppose that there exists the orthogonal projections $\mathbf{L} : \mathbb{R}^N \to \mathcal{P}^p(\mathbb{S})$ and $\mathbf{W} : \mathbb{R}^N \to \mathcal{P}^p(\mathbb{S})^\perp$, where $\mathcal{P}^p(\mathbb{S})^\perp$ is the orthogonal complement of $\mathcal{P}^p(\mathbb{S})$. The operator $\begin{bmatrix} \mathbf{W} \\ \mathbf{L} \end{bmatrix}$ is assumed to be unitary.

The first step is to filter out the effect of the trend by projecting the observations onto the orthogonal subspace. Let $\mathbf{Z_W} := \mathbf{WZ}$, thus from equation (2) it follows that $\mathbf{Z_W} = \mathbf{W}(\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}) = \mathbf{W}\boldsymbol{\varepsilon}$. Notice that the trend component $\mathbf{X}\boldsymbol{\beta}$ is removed from the data $\mathbf{Z}$. The new log-likelihood function for $\mathbf{Z_W}$ becomes

$$\ell_{\mathbf{W}}(\boldsymbol{\theta}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log\det\{\mathbf{C_W}(\boldsymbol{\theta})\} - \frac{1}{2}\mathbf{Z}_{\mathbf{W}}^{\mathrm{T}}\mathbf{C_W}(\boldsymbol{\theta})^{-1}\mathbf{Z_W}, \qquad (5)$$

where $\mathbf{C_W}(\boldsymbol{\theta}) := \mathbf{WC}(\boldsymbol{\theta})\mathbf{W}^{\mathrm{T}}$ and $\mathbf{Z_W} \sim \mathcal{N}_{N-p}(\mathbf{0}, \mathbf{C_W}(\boldsymbol{\theta}))$. A consequence of the filtering is that we obtain an unbiased estimator [16]. The decoupling of the likelihood function is not the only advantage of using $\mathbf{C_W}(\boldsymbol{\theta})$. The following theorem also shows that $\mathbf{C_W}(\boldsymbol{\theta})$ is more numerically stable than $\mathbf{C}(\boldsymbol{\theta})$.

Proposition 1 *Let $\kappa(A) \to \mathbb{R}$ be the condition number of the matrix $A \in \mathbb{R}^{N\times N}$ then $\kappa(\mathbf{C_W}(\boldsymbol{\theta})) \leq \kappa(\mathbf{C}(\boldsymbol{\theta}))$.*

Proposition 1 states that the condition number of $\mathbf{C_W}(\boldsymbol{\theta})$ is less or equal to the condition number of $\mathbf{C}(\boldsymbol{\theta})$. Thus computing the inverse of $\mathbf{C_W}(\boldsymbol{\theta})$ (using a direct or iterative method) will generally be more stable. In practice, computing the inverse of $\mathbf{C_W}(\boldsymbol{\theta})$ can be significantly more stable than $\mathbf{C}(\boldsymbol{\theta})$ depending on the choice of $\mathcal{Q}^d_w$. This has many significant implications as it will now be possible to solve numerically unstable problems.

There are other advantages to the structure of the matrix $\mathbf{C_W}(\boldsymbol{\theta})$. In section 7 it is shown that for a good choice of the $\mathcal{P}^p(\mathbb{S})$ the entries of $\mathbf{C_W}(\boldsymbol{\theta})$ decay rapidly, and most of the entries can be safely eliminated. A level dependent criterion approach is shown in Section 4 that indicates which entries are computed and which ones are not. With this approach a sparse covariance matrix $\tilde{\mathbf{C}}_\mathbf{W}$ can be constructed such that it is close to $\mathbf{C_W}$ in a matrix norm sense, even if the observations are highly correlated with distance.

3.1 Binary multilevel basis

In this section the construction of Multilevel Basis (MB) is shown. The approach followed in this section is a based on the MB construction in [17]. The MB can then be used to: (i) form the multilevel likelihood (5); (ii) sparsify the covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$; and (iii) improve the numerical stability of the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ in it's multilevel form. But first, let us establish notations and definitions:

i) For any index $i, j \in \mathbb{N}_0$, $1 \leq i \leq N$, $1 \leq j \leq N$, let $\mathbf{e}_i[j] = \delta[i-j]$, where $\delta[\cdot]$ is the discrete Kronecker delta function.

ii) Let $\phi(\mathbf{x}, \mathbf{y};\ \boldsymbol{\theta}) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be the covariance function and assumed to be a positive definite. Let $\mathbf{C}(\boldsymbol{\theta})$ be the covariance matrix that is formed from all the interactions between the observation locations $\mathbb{S}$ i.e. $\mathbf{C}(\boldsymbol{\theta}) := \{\phi(\mathbf{x}_i, \mathbf{y}_j;\ \boldsymbol{\theta})\}$, where $i, j, = 1, \dots, N$. We shall assume that the covariance function can be restricted to the following form: There exists a function $\varphi : [0, \infty) \to \mathbb{R}$ such that $\phi(\mathbf{x}, \mathbf{y};\ \boldsymbol{\theta}) = \phi(r;\ \boldsymbol{\theta}) := \varphi(r)$, where $r(\boldsymbol{\theta}) = ((\mathbf{x}-\mathbf{y})^T \operatorname{diag}(\boldsymbol{\theta})(\mathbf{x}-\mathbf{y}))^{\frac{1}{2}}$, $\boldsymbol{\theta} = [\theta_1, \dots, \theta_d] \in \mathbb{R}^d_+$, and $\operatorname{diag}(\boldsymbol{\theta}) \in \mathbb{R}^{d \times d}$ is a diagonal matrix with the vector $\boldsymbol{\theta}$ on the diagonal.

Definition 1 Denote the Matérn covariance function:

$$\phi(r;\ \boldsymbol{\theta}) = \frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\sqrt{2\nu}\frac{r}{\rho}\right)^{\nu} K_\nu \left(\sqrt{2\nu}\frac{r}{\rho}\right),$$

where with a slight abuse of notation Γ is the gamma function, $r \in \mathbb{R}_+$, $0 < \nu$, $0 < \rho < \infty$, and K_ν is the modified Bessel function of the second kind. It is understood from context when Γ is the gamma function.

Remark 2 The Matérn covariance function is a good choice for the random field model. The parameter ρ controls the length correlation and the parameter ν changes the shape. For example, if $\nu = 1/2 + n$, where $n \in \mathbb{N}_+$, then (see [26]) $\phi(r;\ \rho) = \exp\left(-\frac{\sqrt{2\nu}r}{\rho}\right) \frac{\Gamma(n+1)}{\Gamma(2n+1)} \sum_{k=1}^{n} \frac{(n+1)!}{k!(n-k)!} \left(\frac{\sqrt{8v}r}{\rho}\right)^{n-k}$ and $\nu \to \infty \Rightarrow \phi(r;\ \boldsymbol{\theta}) \to \exp\Big(-$

$\left.\frac{r^2}{2\rho^2}\right)$. Note that even for a moderate number of derivatives the number of terms will grow exponentially fast leading to a very complex expression. This motivates the study of complex analytical extensions of the covariance function. See Section 7 for more details.

The first step is to decompose the observation locations of $\mathbb{S}$ in the hypercube domain Γ^d into a multilevel domain decomposition. A good choice is based on the a kD-tree decomposition of the space $\mathbb{R}^d$ [27]. Other choices include Projection (RP) trees. Kd-trees are usually applied for searching algorithms such as range and nearest neighbor. Kd-tree is a particular case of a binary tree, which are also used as decisions trees, sorting and classification, among many other applications.

We refer the reader to [28] for the construction of the kd-tree. However, usually a kd-tree has at most one location point for each of the leaves. Instead, the leaf is set to a maximum of p observations for the version that is used in this paper. First start with the root zero node and corresponding to cell B_0^0 at level 0 that contains all the observation nodes in $\mathbb{S}$. Now, split these nodes into two children cells B_1^1 and B_2^1 at level 1 according to the following rule: i) Choose a unit vector v in $\mathbb{R}^d$ along the axis of $\mathbb{R}^d$. This choice is the direction that leads to the maximum variance of the data in the cell along the direction of v. ii) Project all the nodes $\mathbf{x} \in \mathbb{S}$ in the cell onto the unit vector v. iii) Split the cell with respect to the median of the projections. For each non empty cell B_l^k with $\mathbb{S}_T$ points this procedure is repeated until the full binary tree is built. This rule corresponds to Algorithm 1. We now described the construction of the kd-tree by using this rule.

Algorithm 1 Rule function for cell-split

```
procedure RULE($\mathbb{S}_T$)
    $vrs \leftarrow$ variance of each coordinate direction of the data $\mathbb{S}_T$
    $v \leftarrow$ direction of maximal entry of $vrs$
    $proj\mathbb{S} \leftarrow$ projection of $\mathbb{S}_T$ on $v$
    threshold $\leftarrow$ median of $proj\mathbb{S}_T$
    return threshold, $v$, Rule($\mathbf{x}$) $\leftarrow \mathbf{x} \cdot v \leq$ threshold
end procedure
```

Algorithm 2 initializes the tree by setting the node number and depth of the tree to zero. All of the original observations points $\mathbb{S}$ belong at node zero of the tree. The MakeTree function from Algorithm 3 is then executed to start the tree construction.

Algorithm 2 InitialTree function

```
procedure INITIALTREE($\mathbb{S}$,$n_0$)
    node $\leftarrow$ 0, depth $\leftarrow$ 0
    Tree $\leftarrow$ MakeTree($\mathbb{S}$, node, depth + 1, $n_0$)
end procedure
```

Algorithm 3 splits the observation into binary cells at each level of the tree depth. Given the input observation locations in $\mathbb{S}_T$ they are split into two cells: Left and right according to the Rule. The tree is constructed by calling the MakeTree function recursively, both left and right. Note that the MakeTree function will construct all the left cells first until a leaf is reached. At this point the recursion is unwrapped one step and then the right cell is constructed. This is repeated many times over until all of the leafs are reached and the final tree is produced.

Algorithm 3 MakeTree function

```
procedure MakeTree($\mathbb{S}_T$, node, depth, $n_0$)
    Tree.node ← node, Tree.depth ← depth - 1, node ← node + 1
    if $|\mathbb{S}_T| < n_0$ then return                                    ▷ Leaf of the tree
    end if
    (Rule, threshold, $v$) ← ChooseRule($\mathbb{S}$)
    (Tree.LeftTree, node) ← MakeTree($\mathbf{x} \in \mathbb{S}_T$: Rule($\mathbf{x}$) = True, node, depth + 1, $n_0$)
    (Tree.RightTree, node) ← MakeTree($\mathbf{x} \in \mathbb{S}_T$: Rule($\mathbf{x}$) = false, node, depth + 1, $n_0$)
    Tree.threshold = threshold, Tree.$v$ = $v$
    return Tree
end procedure
```

A binary tree is produced, which is of the form B_0^0, B_1^1, B_2^1, B_3^2, B_4^2, B_5^2, B_6^2, $\dots$, where t is the maximal depth (level) of the tree. Note that each non zero cell B_k^l will correspond to a particular node and depth number. In Figure 1 an example illustration of the kd-tree is shown with a maximal set of locations at the leaves set to four.

Now, let $\mathcal{B}$ be the set of all the cells in the tree and $\mathcal{B}^n$ be the set of all the cells at level $0 \le n \le t$. In addition, for each cell a unique node number, current tree depth, threshold level and projection vector are also assigned to the node. In the Matlab code, this will be useful for searching the tree. Algorithms 1, 2, and 3 describe in more detail the construction of the kD-tree.

Using the binary tree a multilevel basis for $\mathbb{R}^N$ is constructed. Suppose there is a one-to-one mapping between the set of unit vectors $\mathcal{E} := \{\mathbf{e}_1, \dots, \mathbf{e}_N\}$, which is denoted as leaf unit vectors, and the set of locations $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, i.e. $\mathbf{x}_n \longleftrightarrow \mathbf{e}_n$ for all $n = 1, \dots, N$. It is clear that the span of the vectors $\{\mathbf{e}_1, \dots, \mathbf{e}_N\}$ is $\mathbb{R}^N$. The next step is to construct a new basis of $\mathbb{R}^N$ that is multilevel and orthonormal.

1. Start at the maximum level of the random projection tree, i.e. $q = t$.
2. For each leaf cell $B_k^q \in \mathcal{B}^q$ assume without loss of generality that there are s observations nodes $\mathbb{S}_k^q := \{\mathbf{x}_1, \dots, \mathbf{x}_s\}$ with associated vectors $C_k^q := \{\mathbf{e}_1, \dots, \mathbf{e}_s\}$. Denote $\mathcal{C}_k^q$ as the span of the vectors in C_k^q.
   (a) Let $\boldsymbol{\phi}_j^{q,k} := \sum_{\mathbf{e}_i \in C_k^q} c_{i,j}^{q,k} \mathbf{e}_i$, $j = 1, \dots, a$; $\boldsymbol{\psi}_j^{q,k} := \sum_{\mathbf{e}_i \in C_k^q} d_{i,j}^{q,k} \mathbf{e}_i$, $j = a + 1, \dots, s$, where $c_{i,j}^{q,k}$, $d_{i,j}^{q,k} \in \mathbb{R}$ and for some $a \in \mathbb{N}^+$. Note that a is unknown up to this point, but will be computed from the data. It is desired that the new discrete MB vector

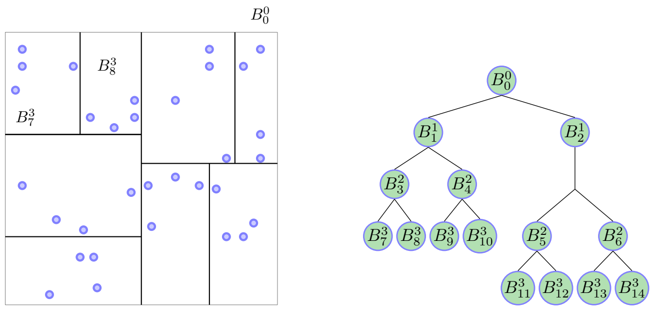

Fig. 1 Multilevel domain decomposition of the observations locations example. All of the initial points in $\mathbb{S}$ are split using Algorithms 1, 2, and 3 until at most 4 points are left in each of the cells corresponding to the leaves of the binary tree.

$\boldsymbol{\psi}_j^{q,k}$ be orthogonal to $\mathcal{P}^p(\mathbb{S})$, i.e., for all $g \in \mathcal{P}^p(\mathbb{S})$:

$$\sum_{i=1}^{n} g[i]\boldsymbol{\psi}_j^{q,k}[i] = 0 \tag{6}$$

(b) Form the matrix $\mathcal{M}^{q,k} := \mathbf{X}^{\mathrm{T}}\mathbf{V}^{q,k}$, where $\mathcal{M}^{q,k} \in \mathbb{R}^{p\times s}$, $\mathbf{V}^{q,k} \in \mathbb{R}^{N\times s}$, and $\mathbf{V}^{q,k} := [\mathbf{e}_1, \dots, \mathbf{e}_i, \dots, \mathbf{e}_s]$ for all $\mathbf{e}_i \in C_k^q$. Now, suppose that the matrix $\mathcal{M}^{q,k}$ has rank a and then perform the Singular Value Decomposition (SVD). Denote by $\mathbf{UDV}$ the SVD of $\mathcal{M}^{q,k}$, where $\mathbf{U} \in \mathbb{R}^{p\times p}$, $\mathbf{D} \in \mathbb{R}^{p\times s}$, and $\mathbf{V} \in \mathbb{R}^{s\times s}$.

Remark 3 Note that in practice we only keep track of the non-zero elements of the vectors $\mathbf{e}_1, \dots, \mathbf{e}_s$. Thus the computational cost is reduced significantly. This is taken into account in the complexity analysis in Lemma 2 and 3

(c) Following the same argument as in [16] but adapted to the kd-tree decomposition equation (6) is satisfied with the following choice

$$\left[\begin{array}{ccc|ccc} c_{1,1}^{q,k} & \dots & c_{1,a}^{q,k} & d_{1,a+1}^{q,k} & \dots & d_{1,s}^{q,k} \\ c_{2,1}^{q,k} & \dots & c_{2,a}^{q,k} & d_{2,a+1}^{q,k} & \dots & d_{2,s}^{q,k} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{s,1}^{q,k} & \dots & c_{s,a}^{q,k} & d_{a+1,s}^{q,k} & \dots & d_{s,s}^{q,k} \end{array}\right] := \mathbf{V}.$$

For this choice the coefficient a is equal to the number of non-zero singular values. Thus the columns $a+1, \dots, s$ form an orthonormal basis of the nullspace $N_0(\mathcal{M}^{q,k})$. Similarly, the columns $1, \dots, a$ form an orthonormal basis of $\mathbb{R}^s \backslash N_0(\mathcal{M}^{q,k})$. Since the vectors in C_k^q are orthonormal then $\boldsymbol{\phi}_1^{q,k}, \dots, \boldsymbol{\phi}_a^{q,k}$, $\boldsymbol{\psi}_{a+1}^{q,k}, \dots,$ $\boldsymbol{\psi}_s^{q,k}$ form an orthonormal basis of $\mathcal{C}_k^q$. Moreover $\boldsymbol{\psi}_{a+1}^{q,k}, \dots, \boldsymbol{\psi}_s^{q,k}$ satisfy equation (6), i.e., are orthogonal to $\mathcal{P}^p(\mathbb{S})$ and are locally adapted to the locations contained in the cell B_k^q.

(d) Denote by $D_k^{q,k}$ the collection of all the vectors $\boldsymbol{\psi}_{a+1}^{q,k}, \ldots, \boldsymbol{\psi}_s^{q,k}$. Notice that the vectors $\boldsymbol{\phi}_1^{q,k}, \ldots, \boldsymbol{\phi}_a^{q,k}$, which are denoted with a slight abuse of notation as the scaling vectors, are *not* orthogonal to $\mathcal{P}^p(\mathbb{S})$. They need to be further processed.

(e) Let $\mathcal{D}^q$ be the union of the vectors in D_k^q for all the cells $B_k^q \in \mathcal{B}_k^q$. Denote by $W_q(\mathbb{S})$ as the span of all the vectors in $\mathcal{D}^q$.

3. For any two sibling cells denote $B_{\texttt{left}}^q$ and $B_{\texttt{right}}^q$ at level q denote $C_{\tilde{k}}^{q-1}$ as the collection of the scaling functions from both cells, for some index $\tilde{k}$.
4. Let $q := q - 1$. If $B_k^q \in \mathcal{B}^q$ is a leaf cell then repeat steps (b) to (d). However, if $B_k^q \in \mathcal{B}^q$ is not a leaf cell, then repeat steps (b) to (d), but replace the leaf unit vectors with the scaling vectors contained in C_k^q with $C_{\tilde{k}}^{q-1}$.
5. When $q = -1$ is reached stop.

When the algorithm stops a series of orthogonal subspaces $V_0(\mathbb{S}), W_0(\mathbb{S}), \ldots, W_t(\mathbb{S})$ (and their corresponding basis vectors) are obtained. These subspaces are orthogonal to $V_0(\mathbb{S}) := span\{\phi_1^0, \ldots, \phi_p^0\}$. Note that the orthonormal basis vectors of $V_0(\mathbb{S})$ also span the space $\mathcal{P}^p(\mathbb{S})$.

Remark 4 Following Lemma 2 in [17] it can be shown that $\mathbb{R}^N = \mathcal{P}^p(\mathbb{S}) \oplus W_0(\mathbb{S}) \oplus W_1(\mathbb{S}) \oplus \cdots \oplus W_t(\mathbb{S})$, Also, it can then be shown that at most $\mathcal{O}(Nt)$ computational steps are needed to construct the multilevel basis of $\mathbb{R}^N$.

From the basis vectors of the subspaces $\mathcal{P}^p(\mathbb{S})^\perp = \cup_{i=0}^t W_i(\mathbb{S})$ an orthogonal projection matrix $\mathbf{W} : \mathbb{R}^N \to (\mathcal{P}^p(\mathbb{S}))^\perp$ can be built. The dimensions of $\mathbf{W}$ is $(N-p) \times N$ since the total number of orthonormal vectors that span $\mathcal{P}^p(\mathbb{S})$ is p. Conversely, the total number of orthonormal vectors that span $\mathcal{P}^p(\mathbb{S})^\perp$ is $N - p$. Let $\mathbf{L}$ be a matrix where each row is an orthonormal basis vector of $\mathcal{P}^p(\mathbb{S})$. For $i = 0, \ldots, t$ let $\mathbf{W}_i$ be a matrix where each row is a basis vector of the space $W_i(\mathbb{S})$. The matrix $\mathbf{W} \in \mathbb{R}^{(N-p)\times N}$ can now be formed, where $\mathbf{W} := \left[\mathbf{W}_t^{\mathrm{T}}, \ldots, \mathbf{W}_0^{\mathrm{T}}\right]^{\mathrm{T}}$. Following a similar approach to Lemma 2.11 in [17] it can be shown that: i) The matrix $\mathbf{P} := \begin{bmatrix} \mathbf{W} \\ \mathbf{L} \end{bmatrix}$ is orthonormal, i.e., $\mathbf{P}\mathbf{P}^{\mathrm{T}} = \mathbf{I}$. ii) Any vector $\mathbf{v} \in \mathbb{R}^n$ can be written as $\mathbf{v} = \mathbf{L}^{\mathrm{T}}\mathbf{v}_L + \mathbf{W}^{\mathrm{T}}\mathbf{v}_{\mathbf{W}}$ where $\mathbf{v}_L \in \mathbb{R}^p$ and $\mathbf{v}_{\mathbf{W}} \in \mathbb{R}^{N-p}$ are unique. The following useful lemmas are proved:

Lemma 2 *Assuming that $n_0 < 2p$, for any level $q = 0, \ldots, t$ there is at most $p2^q$ multilevel basis vectors.*

Lemma 3 *Assuming that $n_0 < 2p$ for any level $q = 0, \ldots, t$ any multilevel vector $\boldsymbol{\psi}_m^q$ associated with a cell $B_k^q \in \mathcal{B}^q$ has at most $2^{t-q+1}p$ non zero entries.*

From Lemma 2 and 3 it can be shown that the matrix $\mathbf{W}$ contains at most $\mathcal{O}(Nt)$ non-zero entries and $\mathbf{L}$ contains at most $\mathcal{O}(Np)$ non-zero entries. Thus for any vector $\mathbf{v} \in \mathbb{R}^n$ the matrix vector products $\mathbf{Wv}$ and $\mathbf{Lv}$ are respectively calculated with at most $\mathcal{O}(Nt)$ and $\mathcal{O}(Np)$ computational steps.

4 Multilevel covariance matrix

The multilevel covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$ and sparse version $\tilde{\mathbf{C}}_\mathbf{W}(\boldsymbol{\theta})$ can be now constructed. Recall from the discussion in Section 3 that $\mathbf{C_W}(\boldsymbol{\theta}) := \mathbf{WC}(\boldsymbol{\theta})\mathbf{W}^{\mathrm{T}}$. From the multilevel basis construct in Section 3.1 the following operator is built: $\mathbf{W} := \left[\mathbf{W}_t^{\mathrm{T}}, \ldots, \mathbf{W}_0^{\mathrm{T}}\right]^{\mathrm{T}}$. Thus the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ is transformed into $\mathbf{C_W}(\boldsymbol{\theta})$, where each of the blocks $\mathbf{C}_\mathbf{W}^{i,j}(\boldsymbol{\theta}) = \mathbf{W}_i\mathbf{C}(\boldsymbol{\theta})\mathbf{W}_j^{\mathrm{T}}$ are formed from all the interactions of the MB vectors between levels i and j, for all $i, j = 0, \ldots, t$. The structure of $\mathbf{C_W}(\boldsymbol{\theta})$ is shown in Figure 2(a). Thus for any $\boldsymbol{\psi}_{\tilde{l}}^i$ and $\boldsymbol{\psi}_{\tilde{k}}^j$ vectors there is a unique entry of $\mathbf{C}_\mathbf{W}^{i,j}$ of the form $(\boldsymbol{\psi}_{\tilde{k}}^i)^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}_{\tilde{l}}^j$. In Section 7 we show that far field entries of $\mathbf{C_W}(\boldsymbol{\theta})$, i.e. $(\boldsymbol{\psi}_{\tilde{k}}^i)^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}_{\tilde{l}}^j$, decay sub-exponentially with respect to $p(d, w)$ if there exists an analytic extension of the covariance function on a well defined domain in $\mathbf{C}^d$. Thus it is not necessary to compute all the entries. We introduce a distance criterion approach to produce a sparse matrix $\tilde{\mathbf{C}}_\mathbf{W}(\boldsymbol{\theta})$.

4.1 Sparsification of multilevel covariance matrix

A sparse version of the covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$ can be built by using a level and distance dependent strategy: i) Given a cell B_k^i at level $i \geq 0$ identify the corresponding tree node value Tree.node and the tree depth Tree.depth. Note that the Tree.depth and the MB level q are the same for $q = 0, \ldots, t$. ii) Let $\mathbb{K} \subset \mathbb{S}$ be all the observations nodes contained in the cell B_k^i. iii) Let $\tau_{i,j} \geq 0$ be the distance parameter given by the user corresponding to the level i, j from the block $\mathbf{C}_\mathbf{W}^{i,j}(\boldsymbol{\theta})$. iv) Let the Targetdepth be equal to the desired level of the tree.

The objective now is to find all the cells at the Targetdepth that overlap a hyper rectangle which is extended from B_k^i. For all observations $\mathbf{x} \in B_l^i$ along each dimension $k = 1, \ldots, d$ let $x_k^{min} := \min_{x_k \in B_m^i} x_k$ and $x_k^{max} := \max_{x_k \in B_m^i} x_k$. Any cell that intersects the interval $[x_k^{min} - \tau_{i,j}, x^{max} + \tau_{i,j}]$ is included. This is done by searching the tree from the root node. At each traversed node check that all the nodes $\mathbf{x} \in \mathbb{K}$ satisfy the following rule: If $\mathbf{x} \cdot \text{Tree}.v + \tau_{i,j} \leq \text{Tree.threshold}$ then search down the left tree. If $\mathbf{x} \cdot \text{Tree}.v - \tau_{i,j} > \text{Tree.threshold}$. the search down the right tree. Otherwise search both trees. The full search algorithm is described in Algorithms 4, 5, and 6.

In Figure 2 (b) & (c) an example for searching local neighborhood cells of randomly placed observations in $\mathbb{R}^2$ is shown. The orange nodes correspond to the source cell. By choosing a suitable value for $\tau_{i,j}$ the blue nodes in the immediate cell neighborhood are found by using Algorithms 4, 5, and 6. The sparse matrix blocks $\mathbf{C}_\mathbf{W}^{i,j}(\boldsymbol{\theta})$ can be built from all the cells that are obtained

Algorithm 4 SearchTree function

```
procedure SearchTree(Tree, $\mathbb{K}$, Targetdepth, $\tau_{i,j}$)
    Targetnodes $\leftarrow \emptyset$, Targetnodes $\leftarrow$ LocalSearchTree(Tree, $\mathbb{K}$, Targetdepth, $\tau_{i,j}$, Targetnodes)
    return Targetnodes
end procedure
```

Algorithm 5 LocalSearchTree function

```
procedure LocalSearchTree(Tree, $\mathbb{K}$, Targetdepth, $\tau_{i,j}$, Targetnodes)
    if Targetdepth = Tree.depth then
        return Targetnodes $\leftarrow$ Targetnodes $\cup$ Tree.node
    end if
    if Targetdepth = Leaf then return
    end if
    LeftRule = ChooseLeftRule($\mathbb{K}$, Tree, $\tau_{i,j}$), RightRule = ChooseRightRule($\mathbb{K}$, Tree, $\tau_{i,j}$)
    Targetnodes $\leftarrow$ LocalSearchTree(Tree.LeftTree, $\mathbb{K}$, Targetdepth, $\tau_{i,j}$, Targetnodes)
    Targetnodes $\leftarrow$ LocalSearchTree(Tree.RightTree, $\mathbb{K}$, Targetdepth, $\tau_{i,j}$, Targetnodes)
    return Targetnodes
end procedure
```

Algorithm 6 ChooseLeftRule and ChooseRightRule rule functions

```
procedure ChooseLeftRule($\mathbb{K}$,Tree,$\tau_{i,j}$)
    return Rule($\mathbf{x}$) := $\mathbf{x} \cdot$ Tree.$v + \tau_{i,j} \leq$ Tree.threshold
end procedure
procedure ChooseRightRule($\mathbb{K}$,Tree,$\tau_{i,j}$)
    return Rule($\mathbf{x}$) := $\mathbf{x} \cdot$ Tree.$v + \tau_{i,j} >$ Tree.threshold
end procedure
```

from SearchTree function of Algorithm 6. Compute all the entries of $\mathbf{C}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$ that correspond to the interactions between any two cells $B^i_k \in \mathcal{B}^i$ and $B^j_l \in \mathcal{B}^j$. In Algorithm 7) the construction of the sparse matrix $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$ is shown.

Remark 5 Since the matrix $\tilde{\mathbf{C}}_{\mathbf{W}}(\boldsymbol{\theta})$ is symmetric it is only necessary to compute the blocks $\mathbf{C}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$ for $i = 1, \ldots, t$ and $j = i, \ldots t$.

4.2 Computational cost of the multilevel matrix blocks of $\tilde{\mathbf{C}}_{\mathbf{W}}$

The cost of computing the multilevel blocks $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}$ will in general be $\mathcal{O}(N^2)$. However, for the special case that $d = 2$ and $d = 3$ it is possible to use a fast summation method such as the Kernel Independent Fast Multipole Method (KIFMM) by [29] to compute the blocks more efficiently. To my knowledge, there exists no equivalent fast summation method in higher dimensions that works satisfactorily. This KIFMM algorithm is flexible and efficient for computing the matrix vector products $\mathbf{C}(\boldsymbol{\theta})\mathbf{x}$ for a large class of kernel functions, including the Matérn covariance function. Given $\tilde{N}$ sources and $\tilde{M}$ targets, experimental results show a computational cost of about $\mathcal{O}(\tilde{N} + \tilde{M})$, $\alpha \approx 1$ with good accuracy (ε_{FMM} between 10^{-6} to 10^{-8}) with a slight degrade in the accuracy with increased source nodes.

Algorithm 7 Construction of sparse matrix $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$

```
procedure CONSTRUCTION(Tree, $i$, $j$, $\tau_{i,j}$, $\mathcal{B}^i$, $\mathcal{B}^j$, $\mathcal{D}^i$, $\mathcal{D}^i$, $\mathbf{C}(\boldsymbol{\theta})$)
    Targetnodes $\leftarrow \emptyset$
    for $B^i_m \in \mathcal{B}^i$ do
        $\mathbb{K} \leftarrow B^i_m$
        for $B^j_q gets$ SearchTree(Tree, $\mathbb{K}$, Targetdepth $(i)$, $\tau_{i,j}$, Targetnodes) do
            for $\psi^i_k \in D^i$ do
                for $\psi^j_l \in D^j$ do
                    Compute $(\boldsymbol{\psi}^i_k)^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}^j_l$ in $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$
                end for
            end for
        end for
    end for
end procedure
```

Assumption 1 Let $\mathbf{A}(\boldsymbol{\theta}) \in \mathbb{R}^{\tilde{M}\times\tilde{N}}$ be a kernel matrix formed from $\tilde{N}$ source observation nodes and $\tilde{M}$ target nodes in the space $\mathbb{R}^d$. Suppose that there exists a fast summation method that computes the matrix-vector products $\mathbf{A}(\boldsymbol{\theta})\mathbf{x}$ with $\varepsilon_{FMM} > 0$ accuracy in $\mathcal{O}((\tilde{N}+\tilde{M})^{\alpha})$ computations, for some $\alpha \geq 1$ and any $\mathbf{x} \in \mathbb{R}^d$.

For the kD-tree it is not possible to determine a-priori the sparsity of the blocks $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$. However, for a given a value $\tau_{i,j} \geq 0$ by running Algorithm 4 on every cell $B^i_k \in \mathcal{B}^i$, at level i, with the Targetdepth corresponding for level j it is possible to determine the computational cost of constructing the sparse blocks $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$ under the following assumption. Suppose that maximum number of cells $B^j_k \in \mathcal{B}^j$ given by Algorithm 4 is bounded by some $\gamma^{i,j} \in \mathbb{N}_+$.

Proposition 4 *The cost of computing each block* $\tilde{\mathbf{C}}^{i,j}_{\mathbf{W}}(\boldsymbol{\theta})$ *for* $i, j = 1, \ldots, t$ *by using a fast summation method with* $1 \leq \alpha \leq 2$ *is bounded by* $\mathcal{O}(\gamma_{i,j} p 2^i (2^{t-j+1}p + 2^{t-i+1}p)^{\alpha} + 2p2^t)$.

5 Multilevel estimator and predictor

The multilevel decomposition, kd-tree and basis can be exploited in such a way to significantly reduce the computational burden and to further increase the numerical stability of the estimation and prediction steps. This is an extension of the multilevel estimator and predictor formulated in [16] to binary trees in higher dimensions. The former is based on Oct-tree decompositions, thus making it unsuitable for higher dimensional problems.

5.1 Estimator

The multilevel likelihood function, $l_{\mathbf{W}}(\theta)$ (see equation (5)), has the clear advantage of being decoupled from the vector $\boldsymbol{\beta}$. Furthermore, the multilevel covariance matrix $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$ will be more numerically stable than $\mathbf{C}(\boldsymbol{\theta})$ thus making it easier to invert and to compute the determinant. However, it is

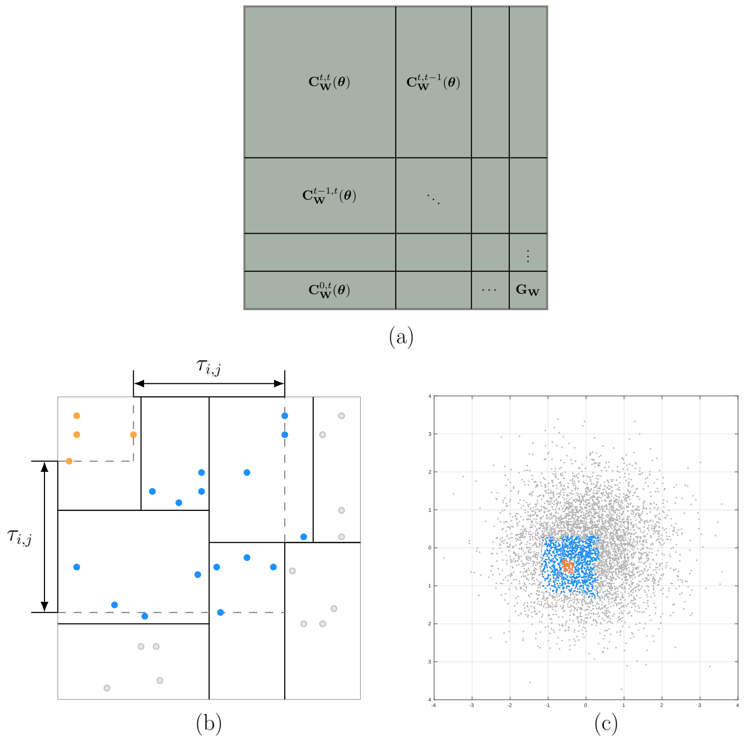
$\mathbf{C}_{\mathbf{W}}^{t,t}(\boldsymbol{\theta})$
$\mathbf{C}_{\mathbf{W}}^{t,t-1}(\boldsymbol{\theta})$
$\mathbf{C}_{\mathbf{W}}^{t-1,t}(\boldsymbol{\theta})$
$\mathbf{C}_{\mathbf{W}}^{0,t}(\boldsymbol{\theta})$
$\mathbf{G}_{\mathbf{W}}$
(a)
$\tau_{i,j}$
$\tau_{i,j}$
(b)
(c)

Fig. 2 Multilevel matrix and Neighborhood identification from source cell on a random kD-tree decomposition of observation locations in $\mathbb{R}^2$. (a) Multilevel covariance matrix where $\mathbf{G}_{\mathbf{W}} := \mathbf{C}_{\mathbf{W}}^{0,0}(\boldsymbol{\theta})$. (b) Cartoon example of axis wise distance criterion $\tau_{i,j}$ using Algorithms 4, 5, and 6. The orange observations knots correspond to the source cell. The blue knots correspond to all the target nodes. The gray knots are not included in the list of target nodes. (c) Example of local neighborhood contained in the axis wise distance $\tau_{i,j}$. The orange nodes are contained in the source cell. The blue nodes are are contained in the local neighborhood cells. The grey dots are all the observations that are not part of the source or local neighborhood cells.

not necessary to perform the MLE estimation on the full covariance matrix $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$, instead construct a series of multilevel likelihood functions $\tilde{\ell}^n_{\mathbf{W}}(\boldsymbol{\theta})$, for $n = 0, \dots t-1$, by applying the partial transform $[\mathbf{W}_t^{\mathrm{T}}, \dots, \mathbf{W}_n^{\mathrm{T}}]^{\mathrm{T}}$ to the data $\mathbf{Z}$. The following likelihood functions are obtained: For $n = 0, \dots, t-1$

$$\tilde{\ell}^n_{\mathbf{W}}(\boldsymbol{\theta}) = -\frac{\tilde{N}}{2}\log(2\pi) - \frac{1}{2}\log\det\{\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})\} - \frac{1}{2}(\mathbf{Z}^n_{\mathbf{W}})^{\mathrm{T}}\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})^{-1}\mathbf{Z}^n_{\mathbf{W}}, \quad (7)$$

where $\mathbf{Z}^n_{\mathbf{W}} := [\mathbf{W}_t^{\mathrm{T}}, \dots, \mathbf{W}_n^{\mathrm{T}}]^{\mathrm{T}}\mathbf{Z}$, $\tilde{N}$ is the length of $\mathbf{Z}^n_{\mathbf{W}}$, $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$ is the $\tilde{N} \times \tilde{N}$ upper-left sub-matrix of $\tilde{\mathbf{C}}_{\mathbf{W}}(\boldsymbol{\theta})$ and $\mathbf{C}^n_{\mathbf{W}}(\boldsymbol{\theta})$ is the $\tilde{N} \times \tilde{N}$ upper-left sub-matrix of $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$. For the case that $n = t$ then

$$\tilde{\ell}^t_{\mathbf{W}}(\boldsymbol{\theta}) = -\frac{\tilde{N}}{2}\log(2\pi) - \frac{1}{2}\log\det\{\tilde{\mathbf{C}}^t_{\mathbf{W}}(\boldsymbol{\theta})\} - \frac{1}{2}(\mathbf{Z}^t_{\mathbf{W}})^{\mathrm{T}}\tilde{\mathbf{C}}^t_{\mathbf{W}}(\boldsymbol{\theta})^{-1}\mathbf{Z}^t_{\mathbf{W}}, \quad (8)$$

where $\mathbf{Z}^t_{\mathbf{W}} := \mathbf{W}_t\mathbf{Z}$, $\tilde{N}$ is the length of $\mathbf{Z}^t_{\mathbf{W}}$, $\tilde{\mathbf{C}}^t_{\mathbf{W}}(\boldsymbol{\theta})$ is the $\tilde{N} \times \tilde{N}$ upper-left sub-matrix of $\tilde{\mathbf{C}}_{\mathbf{W}}(\boldsymbol{\theta})$ and $\mathbf{C}^t_{\mathbf{W}}(\boldsymbol{\theta})$ is the $\tilde{N} \times \tilde{N}$ upper-left sub-matrix of $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$.

A consequence of this approach is that for $n = 0, \ldots, t$ the matrices $\mathbf{C}^n_{\mathbf{W}}(\boldsymbol{\theta})$ are increasingly more stable, thus easier to solve computationally, as shown in the following theorem.

Proposition 5 *Let $\kappa(A) \to \mathbb{R}$ be the condition number of the matrix $A \in \mathbb{R}^{N\times N}$ then $\kappa(\mathbf{C}^t_{\mathbf{W}}(\boldsymbol{\theta})) \leq \kappa(\mathbf{C}^{t-1}_{\mathbf{W}}(\boldsymbol{\theta})) \leq \cdots \leq \kappa(\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})) \leq \kappa(\mathbf{C}(\boldsymbol{\theta}))$.*

Remark 6 If $\mathbf{C}(\boldsymbol{\theta})$ is symmetric positive definite then for $n = 0, \ldots, t$ the matrices $\mathbf{C}^n_{\mathbf{W}}(\boldsymbol{\theta})$ are symmetric positive definite. The proof is immediate. Furthermore, for $n = 1, \ldots, t$, if the matrix $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$ is close to $\mathbf{C}^n_{\mathbf{W}}(\boldsymbol{\theta})$, in some matrix norm sense, then the condition number of $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$ will be close to $\mathbf{C}^n_{\mathbf{W}}(\boldsymbol{\theta})$. Full error bounds will be derived in a future publication.

5.2 Predictor

In this section, we demonstrate how to construct a multilevel BLUP with a well-conditioned multilevel covariance matrix. Furthermore, the multilevel predictor is exact, implying that the solutions of the multilevel predictor and the BLUP equations (3) and (4) are identical. The key insight is to recognize that the BLUP arises from a constrained optimization problem (see Section 2). By seeking the solution within the constrained space, it becomes possible to formulate a set of equations that are numerically more stable. The multilevel approach bypasses the need to invert the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$, thereby avoiding the challenges posed by ill-conditioned matrices. It's important to note that ill-conditioned matrices offer no guarantees of numerical accuracy, as previously discussed in [8].

Consider the following system of equations

$$\begin{pmatrix} \mathbf{C}(\boldsymbol{\theta}) & \mathbf{X} \\ \mathbf{X}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{\gamma}} \\ \hat{\boldsymbol{\beta}} \end{pmatrix} = \begin{pmatrix} \mathbf{Z} \\ \mathbf{0} \end{pmatrix}. \tag{9}$$

From the argument given in [30] it is not hard to show that the solution of this problem leads to equation (3) and $\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta}) = \mathbf{C}^{-1}(\boldsymbol{\theta})(\mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}))$. The BLUP can be evaluated as

$$\hat{Z}(\mathbf{x}_0) = \mathbf{k}(\mathbf{x}_0)^{\mathrm{T}}\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}) + \mathbf{c}(\boldsymbol{\theta})^{\mathrm{T}}\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta}) \tag{10}$$

and the Mean Squared Error (MSE) at the target point $\mathbf{x}_0$ is given by $1 + \tilde{\mathbf{u}}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}\mathbf{X})^{-1}\tilde{\mathbf{u}} - \mathbf{c}(\boldsymbol{\theta})^{\mathrm{T}}\mathbf{C}^{-1}(\boldsymbol{\theta})\mathbf{c}(\boldsymbol{\theta})$ where $\tilde{\mathbf{u}}^{\mathrm{T}} := (\mathbf{X}\mathbf{C}^{-1}(\boldsymbol{\theta})\mathbf{c}(\boldsymbol{\theta}) - \mathbf{k}(\mathbf{x}_0))$.

From (9) it is observed that $\mathbf{X}^{\mathrm{T}}\hat{\boldsymbol{\gamma}}(\boldsymbol{\theta}) = \mathbf{0}$. This implies that $\hat{\boldsymbol{\gamma}} \in \mathbb{R}^n \backslash \mathcal{P}^p(\mathbb{S})$ i.e. the solution for $\hat{\boldsymbol{\gamma}}$ lives in a lower dimensional space, and can be written as

$\hat{\gamma} = \mathbf{W}^{\mathrm{T}}\boldsymbol{\gamma}_{\mathbf{W}}$ for some $\boldsymbol{\gamma}_{\mathbf{W}} \in \mathbb{R}^{N-p}$. From equation (9), rewrite $\mathbf{C}(\boldsymbol{\theta})\hat{\gamma} + \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{Z}$ as

$$\mathbf{C}(\boldsymbol{\theta})\mathbf{W}^{\mathrm{T}}\boldsymbol{\gamma}_{\mathbf{W}} + \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{Z}. \tag{11}$$

Now apply the matrix $\mathbf{W}$ to equation (11) and obtain $\mathbf{W}\{\mathbf{C}(\boldsymbol{\theta})\mathbf{W}^{\mathrm{T}}\boldsymbol{\gamma}_{\mathbf{W}} + \mathbf{X}\hat{\boldsymbol{\beta}}\} = \mathbf{W}\mathbf{Z}$. Since the columns of $\mathbf{X}$ belong to $\mathcal{P}^p(\mathbb{S})$ then $\mathbf{W}\mathbf{X} = \mathbf{0}$ and therefore

$$\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}_{\mathbf{W}} = \mathbf{Z}_{\mathbf{W}}. \tag{12}$$

Solving these set of equations leads to the unique solution $\hat{\boldsymbol{\gamma}}_W$ and the vector $\hat{\gamma}$ can be obtained by applying the inverse transform $\mathbf{W}^{\mathrm{T}}$ i.e. $\hat{\gamma} = \mathbf{W}^{\mathrm{T}}\boldsymbol{\gamma}_{\mathbf{W}}$. From (9) the GLS $\hat{\boldsymbol{\beta}}$ can now be computed as

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}(\mathbf{Z} - \mathbf{C}(\boldsymbol{\theta})\hat{\gamma}). \tag{13}$$

Thus $\hat{\gamma}(\boldsymbol{\theta})$ are $\boldsymbol{\beta}(\boldsymbol{\theta}))$ obtained by solving the multilevel equations (12) and (13), which also solves the system of equations (9). Thus the BLUP is solved exactly.

Remark 7 Solving for $\hat{Z}(\mathbf{x}_0)$, $\hat{\gamma}(\boldsymbol{\theta})$, $\boldsymbol{\beta}(\boldsymbol{\theta})$ only requires the indirect inversion of the covariance matrix $\mathbf{C}_{\mathbf{W}}$ in equation (13). This is in contrast to the classical BLUP method (See Remark 1).

Remark 8 Notice that to solve the GLS estimate $\hat{\boldsymbol{\beta}}$ it is not necessary to compute the full GLS of equation (3), but a least squares is all that is required. This is in contrast to the GLS estimate of equation (3) where if an iterative method is used the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ has to be inverted for each of the columns of $\mathbf{X}$ i.e. p times.

Remark 9 A simple preconditioner $\mathbf{P}_{\mathbf{W}}$ can be formed from the diagonal entries of the matrix $\mathbf{C}_{\mathbf{W}}$ i.e. $\mathbf{P}_{\mathbf{W}} = \mathit{diag}(\mathbf{C}_{\mathbf{W}})$ leading to the following system of equations $\mathbf{P}_{\mathbf{W}}^{-1}\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}_{\mathbf{W}} = \mathbf{P}_{\mathbf{W}}^{-1}\mathbf{Z}_{\mathbf{W}}$. Note that in some cases $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$ will have very small condition numbers. For this case we can set $\mathbf{P}_{\mathbf{W}} := I$, i.e. no preconditioner.

Theorem 6 *If the covariance function $\phi : \Gamma_d \times \Gamma_d \to \mathbb{R}$ is positive definite, then the matrix $\mathbf{P}_{\mathbf{W}}(\boldsymbol{\theta})$ is always symmetric positive definite.*

6 Numerical computation of multilevel estimator and predictor

6.1 Estimator: Computation of $\log \det\{\tilde{\mathbf{C}}^n_{\mathbf{W}}\}$ and $(\mathbf{Z}^n_{\mathbf{W}})^{\mathrm{T}}(\tilde{\mathbf{C}}^n_{\mathbf{W}})^{-1}\mathbf{Z}^n_{\mathbf{W}}$

An approach to computing the determinant of $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$ is to apply a sparse Cholesky factorization technique such that $\mathbf{G}\mathbf{G}^{\mathrm{T}} = \tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$, where $\mathbf{G}$ is a lower triangular matrix. Notice that the eigenvalues of $\mathbf{G}$ are located on the diagonal. This leads to $\log\det\{\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})\} = 2\sum_{i=1}^{\tilde{N}} \log \mathbf{G}_{ii}$.

The direct application of the sparse Cholesky algorithm can lead to significant fill-in of the factorization matrix $\mathbf{G}$. To alleviate this problem it is typical to use matrix reordering techniques. In particular, the fill-in are reduced by using the sparse Cholesky factorization *chol* from the Suite Sparse 4.2.1 package ([31–35]) coupled with Nested Dissection (NESDIS) function package. In practice, this approach leads to a significant reduction of fill-in. A theoretical worse case complexity bounded exists for $d = 2, 3$ dimensions (see [16]).

There are two choices for the computation of $(\mathbf{Z}^n_{\mathbf{W}})^{\mathrm{T}}\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})^{-1}$: i) a Cholesky factorization of $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$, or ii) a Preconditioned Conjugate Gradient (PCG). The PCG choice requires significantly less memory and allows more control of the error. However, the sparse Cholesky factorization of $\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})$ has already been used to compute the determinant. Thus we can use the same factors to compute $(\tilde{\mathbf{Z}}^n_{\mathbf{W}})^{\mathrm{T}}\tilde{\mathbf{C}}^n_{\mathbf{W}}(\boldsymbol{\theta})^{-1}\tilde{\mathbf{Z}}^n_{\mathbf{W}}$. The PCG avenue will be explored further in Section 6.2.

6.2 Predictor computation

For the predictor stage a different approach is used. Instead of inverting the sparse matrix $\tilde{\mathbf{C}}_{\mathbf{W}}(\boldsymbol{\theta})$ a Preconditioned Conjugate Gradient (PCG) method is employed to compute $\hat{\boldsymbol{\gamma}}_{\mathbf{W}} = \mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})^{-1}\mathbf{Z}_{\mathbf{W}}$.

Recall that $\mathbf{C}_{\mathbf{W}} = \mathbf{W}\mathbf{C}(\boldsymbol{\theta})\mathbf{W}^{\mathrm{T}}$, $\hat{\boldsymbol{\gamma}}_{\mathbf{W}} = \mathbf{W}\hat{\boldsymbol{\gamma}}$ and $\mathbf{Z}_{\mathbf{W}} = \mathbf{W}\mathbf{Z}$. Thus the matrix vector products $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}^n_{\mathbf{W}}$ in the PCG iteration are computed within three steps: $\boldsymbol{\gamma}^n_{\mathbf{W}} \xrightarrow[(1)]{\mathbf{W}^{\mathrm{T}}\boldsymbol{\gamma}^n_{\mathbf{W}}} \mathbf{a}_n \xrightarrow[(2)]{\mathbf{C}(\boldsymbol{\theta})\mathbf{a}_n} \mathbf{b}_n \xrightarrow[(3)]{\mathbf{W}\mathbf{b}_n} \mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}^n_{\mathbf{W}}$, where $\boldsymbol{\gamma}^0_{\mathbf{W}}$ is the initial guess and $\boldsymbol{\gamma}^n_{\mathbf{W}}$ is the n^{th} iteration of the PCG. (1) Transformation from multilevel representation to single level. This is done in at most $\mathcal{O}(Nt)$ steps. (2) Perform matrix vector product using a summation method. For $d = 2, 3$ a KIFMM is used to compute the matrix vector products with $\alpha \approx 1$. For $d > 3$ to my knowledge there is no reliable fast summation method. (3) Convert back to multilevel representation.

The matrix-vector products $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}^n_{\mathbf{W}}$, where $\boldsymbol{\gamma}^n_{\mathbf{W}} \in \mathbb{R}^{N-p}$, are computed in $\mathcal{O}(N^\alpha + 2Nt)$ computational steps to a fixed accuracy $\varepsilon_{FMM} > 0$. Note that $\alpha \geq 1$ is dependent on the efficiency of the fast summation method. The total computational cost is $\mathcal{O}(kN^\alpha + 2Nt)$, where k is the number of iterations needed to solve $\mathbf{P}_{\mathbf{W}}^{-1}\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\bar{\gamma}_{\mathbf{W}}(\boldsymbol{\theta}) = \mathbf{P}_{\mathbf{W}}^{-1}\bar{\mathbf{Z}}_{\mathbf{W}}$ to a predetermined accuracy $\varepsilon_{PCG} > 0$.

Remark 10 The introduction of a preconditioner can degrade the accuracy for computing $\hat{\gamma}_{\mathbf{W}} = \mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})^{-1}\mathbf{Z}_{\mathbf{W}}$ with the PCG method. The residual accuracy ε_{PCG} of the PCG iteration has to be set such that the residual of the *unpreconditioned* system $\|\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\gamma_{\mathbf{W}}(\boldsymbol{\theta}) - \mathbf{Z}_{\mathbf{W}}\|_{l^2} < \varepsilon$ for a user given tolerance $\varepsilon > 0$.

Now compute $\hat{\gamma} = \mathbf{W}^{\mathrm{T}}\hat{\gamma}_{\mathbf{W}}$ and $\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}(\mathbf{Z} - \mathbf{C}(\boldsymbol{\theta})\hat{\gamma})$ in at most $\mathcal{O}(N^{\alpha} + Np + p^3)$ computational steps. The matrix vector product $\mathbf{c}(\boldsymbol{\theta})^{\mathrm{T}}\hat{\gamma}(\boldsymbol{\theta})$ is computed in $\mathcal{O}(N)$ steps. Finally, the total cost for computing the estimate $\hat{\mathbf{Z}}(\mathbf{x}_0)$ from (10) is $\mathcal{O}(p^3 + (k+1)N^{\alpha} + 2Nt)$.

7 Multilevel covariance matrix decay

We derive decay estimates of the multilevel covariance matrix. This section is somewhat technical and can be skipped on a first read of the paper. It can be shown that most of the coefficients are small and thus it is not necessary to compute all of them. The final objective is to build a posteriori error estimates for $\mathbf{x}_{\mathbf{W}} = \mathbf{C}_{\mathbf{W}}^{n}(\boldsymbol{\theta})^{-1}\mathbf{Z}_{\mathbf{W}}^{n}$ and $\log\det(\mathbf{C}_{\mathbf{W}}^{n}(\boldsymbol{\theta}))$ that are needed for solving the multilevel estimator MLE. However, the full analysis is extensive and will be completed in a future publication. As a first step we show the decay of the multilevel covariance matrix. Note that this is not trivial and uses the results derived in the supplement. We recommend to first read the appendix since part of the notation used in this section is defined there. However, some of the notation and definitions will be included in this section so as to make it more self contained.

In the paper [16] the authors derive the decay rates of the entries of the covariance matrix for multilevel matrices in $\mathbb{R}^d$ based on Taylor's theorem. This approach is well suited for a small number of dimensions d. However, as d increases the number of derivatives in the Taylor's theorem increases combinatorially. Furthermore, the dimension of the domain of these derivatives increases with respect to d. For large dimensional problems it becomes increasingly difficult to compute the constants that depend on the derivatives in the bounds derived in [16]. In contrast, by using the complex analytic approach the constants can be uniformly bounded and depend on the region of the analytic extension. This is the reason that in the field of uncertainty quantification for stochastic Partial Differential Equations with high dimensional random parameters complex analyticity is used instead [20–24]. We follow this approach.

The decay of the coefficients of the matrix $\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})$ will depend directly on the choice of the multivariate index set $\mathcal{Q}_{w}^{d}$ and the complex analytic regularity extension of the covariance function. In general, the Matérn covariance function will be analytic except for a derivative discontinuity at the origin. However, with the application of the distance criterion $\tau_{i,j} > 0$ a minimal distance can be guaranteed and the origin can be avoided all together. In the following theorem, without loss of generality, it is assumed that the covariance function ϕ is defined on the domain $\Gamma^d \times \Gamma^d$ for on any two cells $B^i_m \in \mathcal{B}^i$ and $B^j_q \in \mathcal{B}^j$. This is achieved by using a pullback that we shall explain shortly. Furthermore, we restrict our attention to any two cells B^i_m and B^j_q that do not overlap. This will guarantee that the center of the covariance function is avoided and the existence of complex analytic extension as shown in Theorem 8.

Suppose that $\sigma > 0$ and denote by

$$\mathcal{E}_\sigma := \Big\{ z \in \mathbb{C}, \sigma \geq \delta \geq 0 : \operatorname{Re} z = \frac{e^\delta + e^{-\delta}}{2} cos(\theta), \operatorname{Im} z = \frac{e^\delta - e^{-\delta}}{2} sin(\theta), \\ \theta \in [0, 2\pi) \Big\}$$

as the region bounded by a Bernstein ellipse. Thus $\mathcal{E}_\sigma$ is an extension into the complex plane from the domain $\Gamma \equiv [-1, 1]$ (see Figure A1). Let $\mathcal{E}_{\sigma,n} \subset \mathbb{C}^d$ a complex region bounded by a Bernstein ellipse such that the restriction on Γ_d is along the n^{th} dimension and form the polyellipse $\mathcal{E}^d_\sigma := \prod_{n=1}^d \mathcal{E}_{\sigma,n}$.

Theorem 7 *Suppose that* $0 < \delta < 1$, $\hat{\sigma} := \sigma(1-\delta)$, *and* $\phi(\boldsymbol{\alpha}, \boldsymbol{\gamma};\ \boldsymbol{\theta}) \in C^0(\Gamma^d \times \Gamma^d;\ \mathbb{R})$, *where* $\boldsymbol{\alpha}, \boldsymbol{\gamma} \in \Gamma^d$, *can be analytically extended on* $\mathcal{E}^d_\sigma \times \mathcal{E}^d_\sigma$ *and is bounded by* $\tilde{M}(\phi)$. *Let* $\mathcal{P}^p(\mathbb{S})^\perp$ *be the subspace in* $\mathbb{R}^N$ *generated by the index set* $\mathcal{Q}^d_w$ *for some* $w \in \mathbb{N}_+$. *For* $i, j = 0, \ldots, t$ *consider any multilevel vector* $\boldsymbol{\psi}^i_m \in \mathcal{P}^p(\mathbb{S})^\perp$, *with* n_m *non-zero entries, from the cell* $B^i_m \in \mathcal{B}^i$ *and any multilevel vector* $\boldsymbol{\psi}^j_q \in \mathcal{P}^p(\mathbb{S})^\perp$, *with* n_q *non-zero entries, from the cell* $B^j_q \in \mathcal{B}^j$. *If* B^i_m *and* B^j_q *do not overlap, and* $p(d, w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$ *then* $|\sum_{k=1}^N \sum_{h=1}^N \phi(\mathbf{x}_k, \mathbf{y}_h;\ \boldsymbol{\theta}) \boldsymbol{\psi}^i_m[h] \boldsymbol{\psi}^j_q[k]|$ *is less or equal to*

$$\sqrt{n_m n_q} \left(\frac{C(\tilde{M}, \sigma)^d e^{d - \sigma(1-\delta)+1} \hat{\sigma} d}{(\sigma\delta)^d} \right)^2 \left(\frac{e^{\hat{\sigma}}}{1 - e^{-\hat{\sigma}}} \right)^{2d} \exp\left(-\frac{2d}{e} \hat{\sigma} p^{\frac{1}{d}} \right) p^{2(\frac{d-1}{d})}.$$

In Figure 3 a plot of the validity of the bound given by Theorem 7 is shown. For example, for $w = 20$ Theorem 7 will be valid for problems of up to $d = 60$ dimensions. For a small number of dimensions d the bounds derived in [16] are sufficient. However, as the number of dimensions increases it is preferable to use complex analyticity and the bound from Theorem 7.

Remark 11 Recall that the restriction $p(d, w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$ is not strict and can be relaxed such that sub-exponential convergence is still obtained. See Remark 15.

Remark 12 Note that the bounded given by Theorem 7 does not give an explicit bound with respect to the distance criterion $\tau_{i,j}$. However, the size of the polyellipse $\mathcal{E}^d$ and magnitude of $\tilde{M}(\phi)$ on $\mathcal{E}^d$ will depend on $\tau_{i,J}$. In Theorem 8 this dependence is shown explicitly.

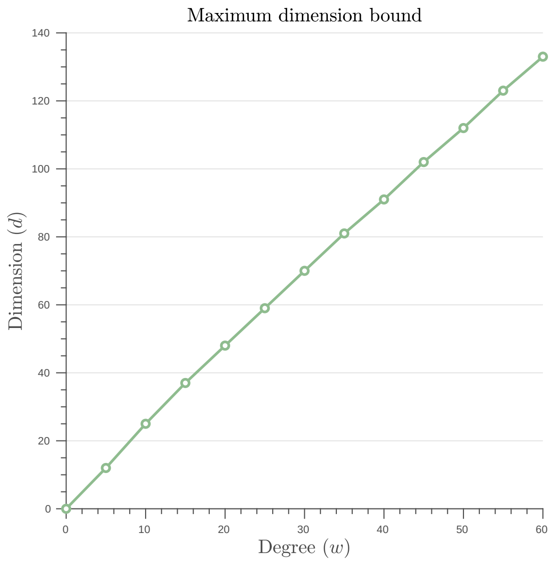

Fig. 3 Validity of Theorem 7 maximum dimensionality with respect to the degree w given by the bound $p(d, w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$.

Remark 13 The decay of the coefficients of $\mathbf{C}_{\mathbf{W}}^{i,j}$ is sub-exponential with respect to p. Even for a moderate magnitude for $\hat{\sigma} > 0$, $p > 0$ and $d \geq 1$ the entries of the multilevel matrix $\mathbf{C}_{\mathbf{W}}^{i,j}$ that do not correspond to the cells given by the distance criterion parameter $\tau_{i,j} \geq 0$ will be close to zero.

Theorem 7 provides a mechanism to control the decay of the coefficients of the multilevel covariance matrix $\mathbf{C}_{\mathbf{W}}$. To apply Theorem 7 we need to show that there exists a complex analytic extension of the Matérn covariance function

$$\phi(r;\ \boldsymbol{\theta}) := \frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\sqrt{2\nu} r(\boldsymbol{\theta})\right)^{\nu} K_{\nu}\left(\sqrt{2\nu} r(\boldsymbol{\theta})\right)$$

and a uniform bound $\tilde{M}(\phi) \leq \infty$ on a subdomain in $\mathbb{C}^d \times \mathbb{C}^d$, where $r(\boldsymbol{\theta}) = ((\mathbf{x}-\mathbf{y})^T \operatorname{diag}(\boldsymbol{\theta})(\mathbf{x}-\mathbf{y}))^{\frac{1}{2}}$, $\boldsymbol{\theta} = [\theta_1, \dots, \theta_d] \in \mathbb{R}^d_+$ are positive constants, and $\operatorname{diag}(\boldsymbol{\theta}) \in \mathbb{R}^{d \times d}$ is a diagonal matrix with the vector $\boldsymbol{\theta}$ on the diagonal for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

Due to the low regularity at the center of the covariance function $\phi(r;\ \boldsymbol{\theta})$ a complex analytic extension will not exist. However, if we avoid the center then such extension is possible. Consider any multilevel vector $\boldsymbol{\psi}^i_m \in \mathcal{P}^p(\mathbb{S})^{\perp}$, with n_m non-zero entries, from the cell $B^i_m \in \mathcal{B}^i$ and any multilevel vector $\boldsymbol{\psi}^j_q \in \mathcal{P}^p(\mathbb{S})^{\perp}$, with n_q non-zero entries, from the cell $B^j_q \in \mathcal{B}^j$. By placing the restriction that $\tau_{i,j} > 0$ and the cells B^i_m and B^j_q do not intersect then the covariance function on each of the locations in B^i_m will not cross the cell in B^j_q. Thus the low regularity center is avoided.

Our first step is to construct a pullback of the covariance function defined on the region covered by the cells B^i_m and B^j_q onto the region $\Gamma^d \equiv [-1,1]^d$. The rational behind this is that the interpolation theory in Appendix A and in [36] is defined on the domain Γ^d, thus we need to re-scale the covariance function on each of the cells B^i_m and B^j_q.

For $k = 1,\dots,d$ let $x^{min}_k := \min_{x^*_k \in B^i_m} x^*_k$, $x^{max}_k := \max_{x^*_k \in B^i_m} x^*_k$, $y^{min}_k := \min_{y^*_k \in B^i_m} y^*_k$, $y^{max}_k := \max_{y^*_k \in B^i_m} y^*_k$ and $\alpha_k, \gamma_k \in [-1,1]$. Define the region $\mathcal{X}^i_m := [x^{min}_1, x^{max}_1] \times \cdots \times [x^{min}_d, x^{max}_d]$ and $\mathcal{Y}^j_q := [y^{min}_1, y^{max}_1] \times \cdots \times [y^{min}_d, y^{max}_d]$.

The next step is to redefine $\phi(\mathbf{x},\mathbf{y};\ \boldsymbol{\theta}) : \mathcal{X}^i_m \times \mathcal{Y}^i_q \to \mathbb{R}$ as $\phi(\boldsymbol{\alpha},\boldsymbol{\gamma};\ \boldsymbol{\theta}) : \Gamma^d \times \Gamma^d \to \mathbb{R}$ through a pullback. For $k = 1,\dots,d$, let $x_k = \left(\frac{\alpha_k+1}{2}\right) a_k + b_k$ and $y_k = \left(\frac{\gamma_k+1}{2}\right) c_k + d_k$, where $a_k = x^{max}_k - x^{min}_k$, $b_k = x^{min}_k$, $c_k = y^{max}_k - y^{min}_k$ and $d_k = y^{min}_k$. Thus, all of the locations $\mathbb{S}$ in B^i_m and B^j_q will be contained in the hyperectangles $\mathcal{X}^i_m$ and $\mathcal{Y}^j_q$ respectively Furthermore, we have also obtained the pullback from $\mathcal{X}^i_m$ and $\mathcal{Y}^j_q$ onto Γ^d.

We now set certain parameters that insures the existence of the analytic extension from the dimensions of the hyperectangles $\mathcal{X}^i_m$ and $\mathcal{Y}^j_q$. The Matérn function consists of polynomial and Bessel components. The polynomial is an entire function, thus we do not have to worry about it. However, the function $K_\nu(\vartheta)$ and $\vartheta^{\frac{1}{2}}$ are analytic for all $\vartheta \in \mathbb{C}$ except at the branch cut $(-\infty, 0]$. Thus it is sufficient to check the analytic extension of $r(\boldsymbol{\theta}) = \left(\sum_{k=1}^d \theta_k (x_k - y_k)^2\right)^{\frac{1}{2}}$. The choice of these parameters will allow us to construct a complex analytic extension that avoids the branch cut at $(-\infty, 0]$. Please read the proof of Theorem 8 for more details. Now, for all $k = 1,\dots,d$ pick $\delta_k > 0$ be such that $\delta_k \leq \frac{\sqrt{32\,\tau^2_{i,j} + 8\,\tau_{i,j} + 1} - 1 - 4\,\tau_{i,j}}{4\,\tau_{i,j}}$. Let $\boldsymbol{\delta} := [\delta_1,\dots,\delta_k]$ and

$$\xi(\boldsymbol{\theta}, \boldsymbol{\delta}, \tau_{i,j}) := \tan^{-1}\left(\frac{\sum_{k=1}^d 2\tau_{i,j}\delta_k + 4\tau^2_{i,j}\delta^2_k}{\sum_{k=1}^d \theta_k \left(\tau^2_{i,j}(1 - 4\delta^2_k) - \frac{\tau_{i,j}\delta_k}{2}\right)}\right).$$

For $k = 1,\dots,d$ let $\sigma^\alpha_k := \cosh^{-1}\left(1 + \frac{\tau_{i,j}\delta_k}{a_k}\right)$ and $\sigma^\gamma_k := \cosh^{-1}\left(1 + \frac{\tau_{i,j}\delta_k}{c_k}\right)$. From these parameters we can form the polyellipses $\mathcal{E}^d_\alpha := \prod_{k=1}^d \mathcal{E}_{\sigma^\alpha_k}$ and $\mathcal{E}^d_\gamma := \prod_{k=1}^d \mathcal{E}_{\sigma^\gamma_k}$ Note that each of the Bernstein ellipses will contain the closed interval $[-1,1]$, thus they are an extension of Γ into the complex plane. From the choice of these parameters it is shown that there exists an analytic extension $r(\boldsymbol{\theta})$ onto $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$. The following result gives the existence of a complex extension of the Matérn kernel onto the polyellipses $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$.

Theorem 8 *For any two cells B^i_m and B^j_q that do not overlap with the associated distance criterion parameter $\tau_{i,j} > 0$ let $\phi(\boldsymbol{\alpha},\boldsymbol{\gamma};\ \boldsymbol{\theta}) : \Gamma^d \times \Gamma^d \to \mathbb{R}$ be the pullback of the Matérn covariance function $\phi(\mathbf{x},\mathbf{y};\ \boldsymbol{\theta}) : \mathcal{X}^i_m \times \mathcal{Y}^j_q \to \mathbb{R}$. Then there exists an* *analytic extension of* $\phi(\boldsymbol{\alpha}, \boldsymbol{\gamma};\ \boldsymbol{\theta}) : \Gamma^d \times \Gamma^d \to \mathbb{R}$ *onto the polyellipse* $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$ *and* $|\phi(\cdot,\cdot;\ \boldsymbol{\theta})| \leq \frac{\left(2\nu \sum_{k=1}^{d} \theta_k \mathcal{R}(\delta_k, \tau_{i,j})\right)^{\frac{\nu}{2}} |K_\nu(\Xi(\boldsymbol{\theta},\boldsymbol{\delta},\tau_{i,j}))|}{\Xi(\boldsymbol{\theta},\boldsymbol{\delta},\tau_{i,j})^\nu}$ *on* $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$*, where*

$$\Xi(\boldsymbol{\theta}, \boldsymbol{\delta}, \tau_{i,j}) := |K_\nu\Big(\sqrt{\frac{\nu}{2}} \cos(\xi(\boldsymbol{\theta}, \boldsymbol{\delta}, \tau_{i,j})/2) \sum_{k=1}^{d} \theta_k \Big(\tau_{i,j}^2(1-4\delta_k^2) - \frac{\tau_{i,j}\delta_k}{2}\Big)\Big)|$$

and $\mathcal{R}(\delta_k, \tau_{i,j}) := 1 + \frac{9}{2}\tau_{i,j}\delta_k + 5\tau_{i,j}^2\delta_k^2$.

We can now apply Theorem 8 to Theorem 7 to obtain a bound on the coefficients of $\mathbf{C_W}(\boldsymbol{\theta})$.

8 Numerical results

The performance of the multilevel solver for estimation and prediction formed from random datasets is tested. The results show that the computational burden is significantly reduced while retaining good accuracy. In particular, it is possible to now solve ill-conditioned problems efficiently. The experimental setup is described in Section B.

8.1 Condition numbers and sparsity of the covariance multilevel matrix

For many practical cases the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ becomes increasingly ill-conditioned for the Matérn covariance function as ρ, ν and the number of observations are increased. This leads to instability of the numerical solver. It is now shown how effective Theorem 1 becomes in practice. In Figure 4 the condition number of the multilevel covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$ is plotted with respect to the cardinality $p(w,d)$ of $\mathcal{Q}^d_w$ for different w levels. The multilevel covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$ is built from the random cube $\mathbf{C}^d_4$ or n-sphere $\mathbf{S}^d_4$ observations. The covariance function is set to Matérn with $\nu = 1$ and $\rho = 1, 10$. As the plots confirm the covariance matrix condition number significantly improves with increasing level w. This is in contrast with the large condition numbers of the original covariance matrix $\mathbf{C}(\boldsymbol{\theta})$. This is consistent with Theorem 1.

In Table 1 sparsity and construction wall clock times of the sparse matrices $\tilde{\mathbf{C}}^i_{\mathbf{W}}(\boldsymbol{\theta})$, $i = t, t-1, \ldots$, for various values of i are shown. The polynomial space of the index set $\mathcal{Q}^d_w$ is restricted to TD on a n-Sphere with $d = 10$ dimensions. The domain decomposition is formed with a kD-tree. The level of the index set is set to $w = 7$, which corresponds $p = 1001$. The covariance function is Matérn with $\nu = 3/4$, $\rho = 3/4$. The distance criterion for each (i,j) multilevel covariance matrix block is set to $\tau_{i,j} := 2^{(t-i)/2}2^{(t-j)/2}\tau$, for $i = 1, \ldots, t$ and $j = 1 \ldots, t$, where $\tau = 3 \times 10^{-6}$.

The first observation to notice is that all the sparse matrices $\tilde{\mathbf{C}}^i_{\mathbf{W}}(\boldsymbol{\theta})$, $i = t, t-1, \ldots$ *are very well conditioned, thus numerically stable.* This is in contrast to the original covariance matrices that are in general poorly conditioned. The

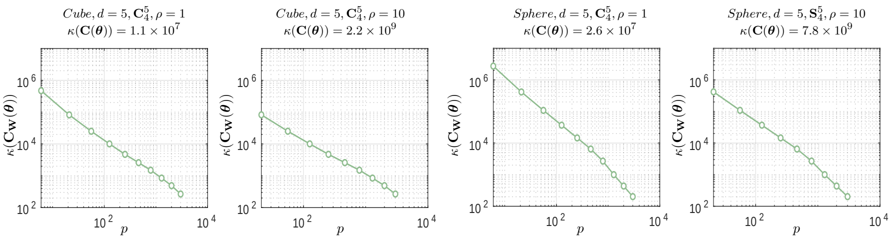

Fig. 4 Condition number of the multilevel covariance matrix $\mathbf{C_W}(\boldsymbol{\theta})$ with respect to the size p of the Total Degree (TD) polynomial space. The number of observations corresponds to 16,000 nodes generated on a hypercube or n-sphere of dimension $d = 5$. The covariance function is chosen to be Matérn with $\nu = 1$ and $\rho = 1, 10$. The condition number of the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ is placed on the top of each subplot for comparison. The MB is constructed from a kD-tree. As expected, as p increases with w the condition number of $\mathbf{C_W}(\boldsymbol{\theta})$ decreases significantly. This is consistent with Theorem 1.

sparsity of $\tilde{\mathbf{C}}^i_{\mathbf{W}}(\boldsymbol{\theta})$ and the Cholesky factor $\mathbf{G}$ are shown in columns 7 and 9. The construction time t_{con} of the $\tilde{\mathbf{C}}^i_{\mathbf{W}}(\boldsymbol{\theta})$ is shown in column 9. In column 5 t_{ML} is the time required to build the multilevel basis. We observe that for large matrices the sparse matrix $\tilde{\mathbf{C}}^i_{\mathbf{W}}(\boldsymbol{\theta})$ are built efficiently. It is noted that the sparse matrices in Table 1 are built with a direct summation method due to the dimensionality.

Table 1 Sparsity test on the matrices $\tilde{\mathbf{C}}^i_{\mathbf{W}}$, $i = t, t-1, \ldots$. The polynomial space of the index set $\mathcal{Q}^d_w$ is restricted to TD on a n-Sphere with $d = 10$ dimensions. The domain decomposition is formed from a kD-tree. The level of the index set is $w = 7$, which corresponds $p = 1001$. The kernel function is Matérn with $\nu = 3/4$, $\rho = 3/4$ and $\tau := 3 \times 10^{-6}$. The first column is the number of random n-Sphere nodes. The second is the maximum level of the kD tree and i is the level of the sparse matrix $\tilde{\mathbf{C}}^i_{\mathbf{W}}$. The fourth column is the condition number of $\tilde{\mathbf{C}}^i_{\mathbf{W}}$, which is excellent. The fifth column is the size of the matrix $\tilde{\mathbf{C}}^i_{\mathbf{W}}$. The seventh column, t_{ML}, is the total time for the construction of the multilevel basis. The eighth column is the sparsity of $\tilde{\mathbf{C}}^i_{\mathbf{W}}$. The ninth column, t_{con} is the total time for the construction of the matrix $\tilde{\mathbf{C}}^i_{\mathbf{W}}$. The tenth column is the sparsity of the Cholesky factor $\mathbf{G}$ (with nested dissection reordering) of the sparse matrix $\tilde{\mathbf{C}}^i_{\mathbf{W}}$. The last column is the total time to compute the Cholesky factor $\mathbf{G}$.

N	t	i	$\kappa(\tilde{\mathbf{C}}^i_{\mathbf{W}})$	Size	t_{ML}	nz	t_{con}	$nz(\mathbf{G})$	t_{sol}
32,000	4	4	5	15,984	46	6.3%	11	3.1%	1
32,000	4	3	8	23,992	46	10.4%	30	5.2%	3
32,000	4	2	13	27,996	46	15.6%	82	7.8%	7
32,000	4	1	19	29,998	46	20.1%	190	10.4%	16
32,000	4	0	23	30,999	46	25.7%	310	13.0%	17
64,000	5	5	6	31,968	104	3.5%	21	1.8%	3
64,000	5	4	11	47,984	105	6.3%	90	3.1%	12
64,000	5	3	18	55,992	106	9.6%	270	5.0%	18
64,000	5	2	121	59,996	121	13.4%	624	6.7%	34
128,000	6	6	8	63,936	237	4.0 %	120	2.1 %	15
128,000	6	5	17	95,968	237	5.5 %	378	6.7 %	140

8.2 Prediction numerical accuracy under ill-conditioning

In this section the numerical accuracy of the ML BLUP solver is tested for a series of highly ill-conditioned matrices and compared with the traditional formulae. Due to the ill-conditioning of the covariance matrices we will see in general that the traditional formulae cannot solve the BLUP with accuracy. Recall the relationship between accuracy and ill-conditioning [8]. This problem is circumvented by using the multilevel approach. For the covariance function the Gaussian kernel $\phi(r,\theta) = \exp(-\frac{r^2}{2\theta^2})$ is used, where θ controls the width of the kernel. This kernel is notorious for leading to covariance matrices with large condition numbers. The observation locations $\{\mathbf{x}_1,\dots,\mathbf{x}_N\}$ are randomly sampled on a unit disk (2D) with $N = 200$. The multilevel basis is constructed with degree $w = 8$, i.e. $p = 45$. Let $\mathbf{x}_k := [x^k_1, x^k_2, \dots x^k_N]$, then observations for $n = 1,\dots N$ are formed as $Z_n = 1 + \sin(\alpha x^n_1)\sin(\alpha x^n_2)$, where $\alpha \in \mathbb{R}$. The BLUP target nodes are be computed on $N_T = 200$ randomly sampled locations $\{\mathbf{g}_1,\dots \mathbf{g}_{N_T}\}$ on the unit disk. The BLUP is computed for each of the target points as $\hat{Z}(\mathbf{g}_k) = \mathbf{k}(\mathbf{g}_k)^{\mathrm{T}}\hat{\boldsymbol{\beta}} + \mathbf{c}(\boldsymbol{\theta})^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}(\mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}})$, where $\mathbf{c}(\boldsymbol{\theta}) = \mathrm{cov}\{\mathbf{Z}, Z(\mathbf{g}_k)\} \in \mathbb{R}^n$ and $k = 1,\dots N_T$.

Suppose the BLUP is computed using the traditional formulae i.e. $\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = (\mathbf{X}^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}\ \mathbf{X})^{-1}\ \mathbf{X}^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})^{-1}\mathbf{Z}$ and $\hat{\gamma}(\boldsymbol{\theta}) = \mathbf{C}^{-1}(\boldsymbol{\theta})(\mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}))$. Let $\hat{\boldsymbol{\beta}}_D$ be the GLS coefficients and $\hat{\mathbf{Z}}_D \in \mathbb{R}^{N_T}$ the BLUP at the target nodes using a traditional double precision computer. Conversely $\hat{\boldsymbol{\beta}}_{52}$ and $\hat{\mathbf{Z}}_{52}$ are computed using the symbolic toolbox of MATLAB with 52 digits accuracy. Note that the symbolic toolbox direct inversion methods are very slow and cannot be used for large matrices. Now, the BLUP is computed using the multilevel method. Let $\hat{\boldsymbol{\beta}}_{\mathbf{W}}$ be the GLS coefficients and $\hat{\mathbf{Z}}^{\mathbf{W}}_D$ be the BLUP at the target nodes with a double precision computer. The relative errors are computed with respect to the 52 digit solution as $\boldsymbol{\epsilon}_{\hat{\boldsymbol{\beta}}_{\mathbf{W}}} := \frac{\|\hat{\boldsymbol{\beta}}_{\mathbf{W}} - \hat{\boldsymbol{\beta}}_{52}\|}{\|\hat{\boldsymbol{\beta}}_{52}\|}$, $\boldsymbol{\epsilon}_{\hat{\mathbf{Z}}^{\mathbf{W}}_D} := \|\hat{\mathbf{Z}}^{\mathbf{W}}_D - \hat{\mathbf{Z}}_{52}\|\|\hat{\mathbf{Z}}_{52}\|^{-1}$, $\boldsymbol{\epsilon}_{\hat{\boldsymbol{\beta}}_D} := \|\hat{\boldsymbol{\beta}}_D - \hat{\boldsymbol{\beta}}_{52}\|\|\hat{\boldsymbol{\beta}}_{52}\|^{-1}$, and $\boldsymbol{\epsilon}_{\hat{\mathbf{Z}}_D} := \|\hat{\mathbf{Z}}_D - \hat{\mathbf{Z}}_{52}\|\|\hat{\mathbf{Z}}_{52}\|^{-1}$.

In Table 2 the accuracy results for the traditional formulae and the multilevel BLUP are tabulated. Observe that the original covariance matrices are very ill-conditioned and is reflected with high errors in the computation of the BLUP. This is in contrast to the multilevel approach, which leads to much higher accuracies and lower condition numbers of the covariance matrices. Notice that some round off errors still affect the overall accuracy. However, the multilevel method is significantly more robust that the traditional BLUP formulae.

8.3 Estimation

In this section estimation results are presented for the Matérn covariance matrix on high dimensional n-Sphere random locations by solving multilevel log-likelihood $\hat{\boldsymbol{\theta}} := \mathrm{argmax}_{\boldsymbol{\theta}}\, \tilde{\ell}^{i}_{\mathbf{W}}(\mathbf{Z}^{i,k,d}_W;\ \boldsymbol{\theta})$, where $\mathbf{Z}^{i,k,d}_W := [\mathbf{W}^{\mathrm{T}}_t, \dots, \mathbf{W}^{\mathrm{T}}_i]^{\mathrm{T}}\mathbf{Z}^d_k$, for $i = t, t-1, \dots$. The observation data $\mathbf{Z}^d_k$ is built from the n-Sphere $\mathbf{S}^d_k$ for $d = 3, 10$, $k = 6$ ($N = 64{,}000$) and $k = 7$ ($N = 128{,}000$). The covariance

α	θ	$\kappa(\mathbf{C})$	$\kappa(\mathbf{C_W})$	$\epsilon_{\hat{\boldsymbol{\beta}}_\mathbf{W}}$	$\epsilon_{\hat{\mathbf{Z}}_D^\mathbf{W}}$	$\epsilon_{\hat{\boldsymbol{\beta}}_D}$	$\epsilon_{\hat{\mathbf{Z}}_D}$
1	10	1.62×10^{19}	9.55×10^{3}	1.15×10^{-4}	1.96×10^{-6}	1.38×10^{2}	3.00×10^{2}
0.5	10	1.62×10^{19}	9.55×10^{3}	1.67×10^{-9}	1.91×10^{-9}	2.71×10^{0}	4.39×10^{0}
2	10	1.62×10^{19}	9.55×10^{2}	2.77×10^{-2}	1.60×10^{-3}	1.41×10^{3}	1.66×10^{4}
2	3	7.92×10^{19}	9.18×10^{2}	2.35×10^{0}	5.79×10^{-4}	1.37×10^{3}	1.47×10^{0}
0.5	3	7.92×10^{19}	9.18×10^{2}	2.81×10^{-4}	7.48×10^{-10}	5.23×10^{0}	6.30×10^{-5}

Table 2 BLUP relative accuracy errors. For different parameters of the test, the condition numbers of the multilevel approach improve significantly over the traditional BLUP formulae. The multilevel approach leads to high accuracy results. This is in contrast to the traditional BLUP formulae, which struggle to achieve reasonable accuracy.

function is Matérn for several values of ν and ρ. To test the performance of the multilevel estimator, $M = 100$ realizations are generated

The optimization problem of the log-likelihood function (7) (and (8)) is solved using a fmincon iteration search for the estimates $\hat{\nu}$ and $\hat{\rho}$ from the optimization toolbox in MATLAB [37]. The tolerance level is set to 10^{-6}. In Table 3 the mean and standard deviation of the Matérn covariance parameter estimates $\hat{\nu}$ and $\hat{\rho}$ are presented. The mean estimate $\mathbb{E}_M[\hat{\nu}]$ refers to the mean of M estimates $\hat{\nu}$ for the M realizations of the stochastic model. Similarly, $std_M[\hat{\nu}]$ refers to the standard deviation of the M realizations. For case (a) $(d = 3)$ the error mean and std is $\approx 1\%$. For case (b) $(d = 10)$ the error of the mean increase to $\approx 10\%$. In general, as i is reduced from t there is a tendency of a drop in the standard deviation $std_M[\hat{\nu}]$ of the estimator $\hat{\nu}$. However, there is also a tendency for the accuracy of the mean to degrade somewhat, except for (a) $N = 128,000$, $i = 12 \to 11$.

Table 3 Estimation of parameters $\hat{\nu}$ and $\hat{\rho}$ with: Total Degree polynomial index set $\mathcal{Q}^d_w$, kD tree, and n-Sphere with for $d = 3$ and $d = 10$. The observation data $\mathbf{Z}^d_k$ are formed from the Matérn covariance function. The number of realizations of the Gaussian random field model is set to $M = 100$. Several cases are tested and are given by the individual tables (a) and (b). The first to fourth columns are self-explanatory. The fifth column is the mean error of $\hat{\nu}$ with M realization. The sixth column is the mean error of $\hat{\rho}$. The last two columns are the standard deviation of M realizations of the parameters $\hat{\nu}$ and $\hat{\rho}$.

(a) TD, kD tree, n-Sphere, $d = 3$, $M = 100$, $\nu = 3/4$, $\rho = 1/6$, $\tau = 5 \times 10^{-2}$

N	w	t	i	$\mathbb{E}_M[\hat{\nu} - \nu]$	$\mathbb{E}_M[\hat{\rho} - \rho]$	$std_M[\hat{\nu}]$	$std_M[\hat{\rho}]$
64000	3	11	11	-1.92e-04	4.52e-04	1.36e-02	8.17e-03
64000	3	11	10	1.17e-03	-5.90e-04	7.08e-03	4.04e-03
128000	3	12	12	-2.51e-03	1.81e-03	8.54e-03	6.11e-03
128000	3	12	11	-6.90e-04	5.02e-04	4.17e-03	2.84e-03

(b) TD, kD tree, n-Sphere, $d = 10$, $M = 100$, $\nu = 3/4$, $\rho = 3/4$, $\tau = 1 \times 10^{-5}$

N	w	t	i	$\mathbb{E}_M[\hat{\nu} - \nu]$	$\mathbb{E}_M[\hat{\rho} - \rho]$	$std_M[\hat{\nu}]$	$std_M[\hat{\rho}]$
64000	4	5	5	8.70e-03	-1.12e-02	1.55e-02	1.85e-02
64000	4	5	4	-9.31e-02	8.02e-02	1.67e-02	1.97e-02
128000	4	6	6	-6.36e-03	5.51e-03	2.10e-02	1.72e-02
128000	4	6	5	-7.18e-02	6.27e-02	1.32e-02	1.46e-02

9 Prediction

In this section the computational performance of the multilevel solver is analyzed. Given a fixed Matérn parameters (ν, ρ) the BLUP vectors $\hat{\gamma}$ and $\hat{\beta}$ are computed. This involves solving the system of equations $\mathbf{P}_{\mathbf{W}}^{-1}\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\gamma_{\mathbf{W}} = \mathbf{P}_{\mathbf{W}}^{-1}\mathbf{Z}_{\mathbf{W}}$ and $\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\ \mathbf{X}^{\mathrm{T}}(\mathbf{Z} - \mathbf{C}\hat{\gamma})$. Results for computing $\hat{\gamma}$ and $\hat{\boldsymbol{\beta}}$ for the hypercube data set with $d = 3$ dimensions, kD tree, and the Total Degree index set $\mathcal{Q}_w^d$ are shown in Table 4. The Matérn covariance coefficients $\boldsymbol{\theta} = (\nu, \rho)$ are set to (3/4,1). The relative error of the residual of PCG method for the unpreconditioned system is set to $\varepsilon = 10^{-3}$. The KIFMM is set to high accuracy.

For computing the matrix vector products of the PCG iterations, the computational break even point of the KIFMM solver is reached for $N \approx 2,500$ compared to using the direct approach (with CPU and GPU). The increase in computational complexity is linear with respect to N. Thus all the matrix vector products for the PCG iterations are calculated using the KIFMM. The preconditioner $\mathbf{P}_{\mathbf{W}}$ is built using a combination of the GPU and CPU. This leads to a quadratic increase in computational cost with respect to the number of observations N. However, due to the high efficiency of the implementation and $p = 120$, the break even point for the use of the KIFMM solver is not reached, even for $N = 512,000$ observation points. From Table 4 observe that condition number of the covariance matrix $\mathbf{C}$ is much larger compared to $\mathbf{C}_{\mathbf{W}}$. This is already a good indication that solving the prediction problem will be more efficient using the multilevel approach.

The number of iterations needed to reach the same accuracy for both approaches are significantly better with the multilevel approach i.e. ≈ 70 times less iterations. However, the computation of $\boldsymbol{\beta}$ with the single level method requires solving $p = 121$ matrix inversions of $\mathbf{C}$. This is in contrast with a single matrix inversion of $\mathbf{C}_{\mathbf{W}}$ with the multilevel method. In practice, we did not solve all p matrix inversions for the single level approach, but measure the time required to compute a single matrix inversion and multiplied it 121 to obtain the estimated time complexity. For $N = 64,000$ observations we observe efficiencies of $\approx 7,000$ compared to the single level iterative approach. The multilevel approach is now tested on $d = 20, 25$ dimensional problems. Due to the high dimensionality of these problems, a fast summation approach is not an option. The matrix-vector products of each iteration are computed with the direct approach using the GPU and CPU. In Table 5(a) the numerical results for computing $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ for $d = 20$ and $\theta = (5/4, 10)$. Compared to the single level iterative approach the multilevel method is approximately 42,000 faster for $N = 64,000$ observations. Similar results are obtained shown in Table 5(b). for $d = 25$ and $\theta = (5/4, 10)$.

10 Conclusions

In this paper a multilevel method is developed that scales well with high dimensions for solving the spatial BLUP. A multilevel basis is constructed from a

Table 4 Numerical results for computing $\mathbf{P}_{\mathbf{W}}^{-1}\mathbf{C}_{\mathbf{W}}(\boldsymbol{\theta})\boldsymbol{\gamma}_{\mathbf{W}} = \mathbf{P}_{\mathbf{W}}^{-1}\mathbf{Z}_{\mathbf{W}}$ and $\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}(\mathbf{Z} - \mathbf{C}\hat{\boldsymbol{\gamma}})$ for the hypercube data set with $d = 3$ and the Total Degree index set Q_w^d. The Matérn covariance coefficients $\boldsymbol{\theta} = (\nu, \rho)$ are set to (3/4,1). The relative error of the residual of PCG method for the *unpreconditioned system* is set to $\varepsilon = 10^{-3}$. The KIFMM is set to high accuracy. (a) The second column of is the condition number of the covariance matrix $\mathbf{C}$, up to $N = 64,000$ observations, and is compared with the third column which corresponds to the condition number of $\mathbf{C}_W$. The third column is the number of CG iterations needed for convergence for 10^{-3} residual accuracy. The fourth column is the number of iterations need to achieve the residual error 10^{-3} for the unpreconditioned system with the preconditioner $\mathbf{P}_{\mathbf{W}}$. (b) The sixth column corresponds to the wall clock times in seconds for the preconditioner computation. The PCG iteration wall clock timings for $\mathbf{C}_{\mathbf{W}}$, by using a KIFMM, are given in the seventh column. The eighth column is the total time to compute $\boldsymbol{\gamma}_{\mathbf{W}}$, $\boldsymbol{\beta}$ and the multilevel basis construction. The eighth column is the computational efficiency for computing $\boldsymbol{\gamma}_{\mathbf{W}}$ vs $\mathbf{C}^{-1}\mathbf{Z}$ to same residual accuracy with respect to the number of iterations. The last column is the estimated efficiency of computing $\hat{\boldsymbol{\gamma}}$ and $\hat{\boldsymbol{\beta}}$ with the multilevel BLUP compared to the single level approach, equation (10), to approximately the same accuracy using a CG iteration with the KIFMM. We observe the significant speed ups ($\approx 7,000$ for $N = 64,000$) for calculating the BLUP by using the multilevel approach.

$\boldsymbol{\theta} = (3/4, 1)$, $d = 3$, $w = 7$ ($p = 120$)

N	$\kappa(\mathbf{C})$	$\kappa(\mathbf{C}_{\mathbf{W}})$	itr($\mathbf{C}$)	itr($\mathbf{C}_{\mathbf{W}}$)	$\mathbf{P}_{\mathbf{W}}$ (s)	Itr (s)	Total (s)	Eff$_{\boldsymbol{\gamma}}$	Eff$_{\boldsymbol{\gamma},\boldsymbol{\beta}}$
8,000	3.2×10^7	1.8×10^4	1,985	52	4	29	38	38	3,600
16,000	1.1×10^8	6.0×10^4	3,511	67	13	98	118	52	5,000
32,000	5.6×10^8	3.1×10^5	8,259	116	45	260	317	71	7,250
64,000	1.8×10^9	9.5×10^5	12,680	165	178	798	997	76	7,380
128,000	-	-	-	308	713	3,934	4,687	-	-
256,000	-	-	-	292	2,837	5,745	8,663	-	-
512,000	-	-	-	484	11,392	20,637	32,202	-	-

Table 5 Computing BLUP for the n-sphere data set with $d = 20$ and $d = 25$ dimensions, TD index set, and Matérn covariance function without pre-conditioner. The residual accuracy is set to $\varepsilon = 10^{-3}$. Since the dimension is greater than 3, the matrix vector products are computed directly with the GPU and CPU. The description of the columns of tables (a) and (b) are the same as for Table 4. In addition, column 6 corresponds to the wall clock time for computing the multilevel basis. (a) Computational times for solving the prediction for $d = 20$ and $\theta = (5/4, 10)$. The growth in computational cost is slightly faster than quadratic due to the lack of fast summation method in higher dimensions. However, compared to the single level iterative approach it is approximately 42,000 faster for $N = 64,000$ observations. (b) BLUP prediction for $d = 25$ and $\theta = (5/4, 10)$. The growth in computational cost is similar. The efficiency of this method is about 2,840 times faster.

(a) $\boldsymbol{\theta} = (\nu, \rho) = (5/4, 10)$, $d = 20$, $w = 3$ ($p = 1771$), No precond., Direct

N	$\kappa(\mathbf{C})$	$\kappa(\mathbf{C}_{\mathbf{W}})$	itr($\mathbf{C}$)	itr($\mathbf{C}_{\mathbf{W}}$)	MB(s)	Itr(s)	Total(s)	Eff$_{\boldsymbol{\gamma},\boldsymbol{\beta}}$
16,000	5×10^7	7	238	10	52	97	153	26,700
32,000	1×10^8	11	324	13	121	500	628	35,160
64,000	2×10^8	17	444	17	284	2,600	2,898	42,050
128,000	-	-	-	22	628	13,494	14,153	-

(b) $\boldsymbol{\theta} = (\nu, \rho) = (3/4, 10)$, $d = 25$, $w = 2$ ($p = 351$), No precond., Direct

N	$\kappa(\mathbf{C})$	$\kappa(\mathbf{C}_{\mathbf{W}})$	itr($\mathbf{C}$)	itr($\mathbf{C}_{\mathbf{W}}$)	MB(s)	Itr(s)	Total(s)	Eff$_{\boldsymbol{\gamma},\boldsymbol{\beta}}$
16,000	2×10^6	7	86	12	5	116	122	2,400
32,000	4×10^6	12	109	15	13	582	599	2,490
64,000	9×10^6	21	147	18	30	2,788	2,821	2,840
128,000	-	-	-	25	79	15,557	15,641	-
256,000	-	-	-	33	157	83,163	83,337	-

kD-tree and for the choice of Total Degree polynomial basis $\mathcal{Q}^d_w$. The approach described in the paper has the following characteristics and advantages:

i) The multilevel method is numerically stable. Hard estimation and prediction of large dimensional problems are now feasible. ii) The method is efficiently implemented by using a combination of MATLAB, c++ software packages and dynamic libraries. iii) Sub-exponential decay of multilevel covariance matrix $\mathbf{C_W}$ is proven based on complex analytic extensions. iv) Numerical results of up to 25 dimensional problems. These problems are difficult to solve with traditional methods due to the large condition numbers, but feasible with the multilevel method. v) The multilevel prediction approach is proven to be *exact*, in the sense that single level and multilevel prediction formulations are shown to be equivalent. vi) The efficiency of this approach will be further improved as high dimensional fast summation methods are developed. vii) An A-posteriori scheme and estimates for constructing the sparse covariance matrix $\tilde{\mathbf{C}}$ will be developed in a future paper. This will be possible with the error bounds for the entries of $\mathbf{C}$ derived in this paper since all the constants can be estimated.

Acknowledgments. I appreciate the help and advice from George Biros and Lexing Ying for setting up the KIFMM packages. In addition, I am grateful to the intense and very illuminating discussions with Michael Stein and Mihai Anitesc. I also appreciate the support that King Abdullah University of Science and Technology has provided to this project.

Appendix A Polynomial Interpolation

In this section we provide some background on polynomial interpolation in high dimensions. This will be critical to estimate the decay rates of the entries of the multilevel covariance matrix for high dimensional problems.

The decay of the coefficients will directly depend on the analytic properties of the covariance function. The traditional error estimates of polynomial interpolation are based on multi-variate m^{th} order derivatives. However, for many cases, such as the Matérn covariance function, the derivatives are too complex or expensive to manipulate for even a moderate number of dimensions. This motivates the study of polynomial numerical approximations based on complex analytic extensions, which are much better suited for high dimensions. Much of the discussion that follows has it roots in the field of uncertainty quantification and high dimensional interpolation [20, 22, 36] for partial differential equations.

Consider the problem of approximating a function $v : \Gamma^d \rightarrow \mathbb{R}$ on the domain Γ^d. Without loss of generality let $\Gamma \equiv [-1, 1]$ and $\Gamma^d \equiv \prod_{n=1}^{d} \Gamma$. Suppose that $\mathcal{G} \subset \Gamma^d$, then define the following spaces

$$L^q(\mathcal{G}) := \{v(\mathbf{y}) \,:\, \int_{\mathcal{G}} v(\mathbf{y})^q \mathrm{d}\mathbf{y} < \infty\} \text{ and } L^\infty(\mathcal{G}) := \{v(\mathbf{y}) \,:\, \sup_{\mathbf{y}\in\mathcal{G}} |v(\mathbf{y})| < \infty\}.$$

Suppose that $\mathcal{P}_q(\Gamma) := \text{span}\{y^k,\, k = 0, \dots, q\}$ i.e. the space of polynomials of degree at most q. Let $\mathcal{I}^m : C^0(\Gamma) \to \mathcal{P}_{m-1}(\Gamma)$ be the univariate Lagrange interpolant $\mathcal{I}_m(v(\mathbf{y})) := \sum_{k=1}^{m} v(y^{(k)}) l_{m,k}(y^{(k)})$, where $y^{(1)}, \dots, y^{(m)}$ is a set of distinct knots on Γ and $\{l_{n,k}\}_{k=0}^{m}$ is a Lagrange basis of the space $\mathcal{P}_{m-1}(\Gamma)$. The variable $m \in \mathbb{N}_0$ corresponds to the order of approximation of the Lagrange interpolant. However, for the case of the zero order interpolation $m = 0$ corresponds to $\mathcal{I}_0 = 0$.

Remark 14 For high dimensional interpolation the particular set of points $y^{(1)}, \dots,$ $y^{(m)}$ that we will use is the Clenshaw-Curtis abscissas. This is further discussed in this section. However, for now, we assume that the points are only distinct.

For $m \geq 1$ let $\Delta_m := \mathcal{I}_m - \mathcal{I}_{m-1}$. From the difference operator Δ_m we can readily observe that $\mathcal{I}_m = \sum_{k=1}^{m} \Delta_k$, which is reminiscent of multi resolution wavelet decompositions. The idea is to represent multivariate approximation as a summation of the difference operators.

Consider the multi-index tupple $\mathbf{m} = (m_1, \dots, m_d)$, where $\mathbf{m} \in \mathbb{N}_0^d$, and form the tensor product operator $\mathcal{S}_{w,d} : \Gamma \to \mathbb{R}$ as

$$\mathcal{S}_{w,d}[v(\mathbf{y})] := \sum_{\mathbf{m} \in \mathbb{N}^d : \sum_{i=1}^{d} m_i - 1 \leq w} \bigotimes_{n=1}^{d} \Delta^n_{m_n}(v(\mathbf{y})). \tag{A1}$$

Note that by $\Delta^n_{m_n}(v(\mathbf{y}))$ we mean that the difference operator Δ_{m_n} is applied along the n^{th} dimension in Γ.

Let $C^0(\Gamma^d;\, \mathbb{R}) := \{v : \Gamma^d \to \mathbb{R}$ is continuous on Γ^d and $\max_{\mathbf{y} \in \Gamma^d} |v(\mathbf{y})| < \infty\}$. From Proposition 1 in [38] it is shown that for any $v \in C^0(\Gamma^d;\, \mathbb{R})$, we have $\mathcal{S}_{w,d}[v] \in \mathcal{Q}^d_w$. Moreover, $\mathcal{S}_{w,d}[v] = v$, for all $v \in \mathcal{Q}^d_w$. The key observation to take away is that the operator $\mathcal{S}_{w,d}[v]$ is *exact* in the space of polynomials $\mathcal{Q}^d_w$. This will be useful in connecting the Lagrange interpolant with Chebyshev polynomials.

Let $T_k : \Gamma \to \mathbb{R}$, $k = 0, 1, \dots,$ be a Chebyshev polynomial over Γ, which are defined recursively as follows: $T_0(y) = 1$, $T_1(y) = y$, $\dots$, $T_{k+1}(y) = 2yT_k(y) - T_{k-1}(y), \dots$, where $y \in \Gamma$. Chebyshev polynomials are well suited for the approximation of functions with analytic extensions on a complex region bounded by a Bernstein ellipse. They bypassing the need of using derivative information and sharp bounds on the error are readily available. Suppose that $\sigma > 0$ and denote by

$$\mathcal{E}_\sigma := \Big\{ z \in \mathbb{C}, \sigma \geq \delta \geq 0 : \operatorname{Re} z = \frac{e^\delta + e^{-\delta}}{2} cos(\theta), \operatorname{Im} z = \frac{e^\delta - e^{-\delta}}{2} sin(\theta),$$
$$\theta \in [0, 2\pi) \Big\}$$

as the region bounded by a Bernstein ellipse (see Figure A1). The following theorem is based on complex analytic extensions on $\mathcal{E}_\sigma$ and provides a control for the Chebyshev polynomial approximation.

Theorem 9 *Suppose that for $u : \Gamma \to \mathbb{R}$ there exists an analytic extension on $\mathcal{E}_\sigma$. If* $\mathsf{u} \leq M < \infty$ *on $\mathcal{E}_\sigma$ then there exists a sequence of coefficients $|\alpha_k| \leq M/e^{k\sigma}$ such that $u \equiv \alpha_0 + 2\sum_{k=1}^{\infty} \alpha_k T_k$ on $\mathcal{E}_\sigma$. Moreover, if $y \in \Gamma$ then $|q(y) - \alpha_0 - 2\sum_{k=1}^{n} \alpha_k T_k(y)| \leq \frac{2M}{e^\sigma - 1} e^{-n\sigma}$.*

Proof See Theorem 2.25 in [39] □

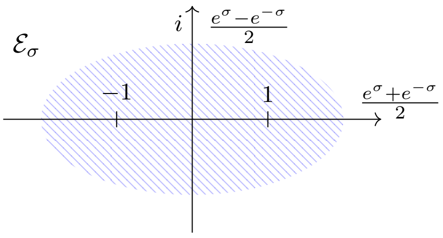
$\mathcal{E}_\sigma$
i
$\frac{e^\sigma - e^{-\sigma}}{2}$
-1
1
$\frac{e^\sigma + e^{-\sigma}}{2}$

Fig. A1 Complex region bounded by the Bernstein ellipse.

We can now connect the error due to the Lagrange interpolation with Chebyshev expansions. It is known that if $u \in C(\Gamma, \mathbb{R})$ then $\|(I - \mathcal{I}_m)u\|_{L^\infty(\Gamma)} \leq (1 + \Lambda_m) \min_{h \in \mathcal{P}_{m-1}} \|u - h\|_{L^\infty(\Gamma)}$, where Λ_m is the Lebesgue constant (See Lemma 7 in [40]). Note that $I : C^d(\Xi;\ \mathbb{R}) \to C^d(\Xi;\ \mathbb{R})$ refers to the identity operator and the domain Ξ is taken from context. For the previous case $\Xi = \Gamma$. Bounds on Λ_m are known in the context of the location of the knots $y^{(1)}, \ldots, y^{(m)} \in \Gamma$. In this article we restrict our attention to Clenshaw-Curtis abscissas $y^{(j)} = -\cos\left(\frac{\pi(j-1)}{m-1}\right)$, $j = 1, \ldots, m$ and Λ_m is bounded by $2\pi^{-1}(\log(m-1) + 1) \leq 2m - 1$ (see [40]). Since the interpolation operator $\mathcal{I}_m$ is exact on $\mathcal{P}_{m-1}$, then if $u : \Gamma \to \mathbb{R}$ has an analytic extension in $\mathcal{E}_\sigma$ we have from Theorem 9 (following a similar approach as in [40]) that $\|(I - \mathcal{I}_m)u\|_{L^\infty(\Gamma_n)} \leq (1 + \Lambda_m)\frac{2M}{e^\sigma - 1} e^{-\sigma(m-1)} \leq 2C(M, \sigma) m e^{-\sigma(m-1)}$, where $C(M, \sigma_n) := \frac{2M}{(e^\sigma - 1)}$. We then conclude that for all $k = 1, \ldots, m$

$$\begin{aligned} \|\Delta_k(u)\|_{L^\infty(\Gamma)} &= \|\mathcal{I}^m(u) - \mathcal{I}^{m-1}(u)\|_{L^\infty(\Gamma)} \leq \|(I - \mathcal{I}_m)u\|_{L^\infty(\Gamma)} \\ &+ \|(I - \mathcal{I}_{m-1})u\|_{L^\infty(\Gamma)} \leq e^{2\sigma} C(M, \sigma) m e^{-\sigma m}. \end{aligned} \tag{A2}$$

Let $\mathcal{E}_{\sigma,n} \subset \mathbb{C}^d$ a complex region bounded by a Bernstein ellipse such that the restriction on Γ_d is along the n^{th} dimension and form the polyellipse $\mathcal{E}_\sigma^d := \prod_{n=1}^{d} \mathcal{E}_{\sigma,n}$. Suppose that $v : \mathcal{E}_\sigma^d \to \mathbb{C}$ is analytic on $\mathcal{E}_\sigma^d$ and let $\tilde{M}(v) := \max_{\mathbf{z} \in \mathcal{E}_\sigma^d} |v(\mathbf{z})|$.

Note we refer to $\mathcal{I}_m^n$ as the Lagrange operator of order m along the n^{th} dimension and similarly $\mathcal{P}_{m-1}^n$ is the space of the span of univariate polynomials up to degree $m-1$ along the n^{th} dimension. Form the tensor product $\mathbf{I}_m^d := \mathcal{I}_m^1 \times \cdots \times \mathcal{I}_m^d$, thus $\mathbf{I} : C(\Gamma, \mathbb{R}) \to \mathbb{P}$ where $\mathbb{P} := \mathcal{P}_{m-1}^1 \times \cdots \times \mathcal{P}_{m-1}^d$. From Theorem 2.27 in [39] we can conclude that for a finite dimension d, as $m \to \infty$ then $\mathbf{I}_m^d[v] \to v$.

Applying equation (A2) to equation (A1) we have that

$$\begin{aligned}
\|(I - \mathcal{S}_{w,d})v(\mathbf{y})\|_{L^\infty(\Gamma^d)} &\leq \left\| \sum_{\mathbf{m}\in\mathbb{N}^d : \sum_{i=1}^d m_i - 1 > w} \bigotimes_{n=1}^d \Delta_{m_n}^n(v(\mathbf{y})) \right\|_{L^\infty(\Gamma^d)} \\
&\leq \sum_{\mathbf{m}\in\mathbb{N}^d : \sum_{i=1}^d m_i - 1 > w} \bigotimes_{n=1}^d \|\Delta_{m_n}^n(v(\mathbf{y}))\|_{L^\infty(\Gamma^d)} \\
&\leq \sum_{\mathbf{m}\in\mathbb{N}^d : \sum_{i=1}^d m_i - 1 > w} e^{2d} C(M,\sigma)^d \left(\prod_{n=1}^d m_n \right) \exp\left(-\sum_{n=1}^d \sigma m_n \right).
\end{aligned} \tag{A3}$$

By applying Theorem 2.10 and Corollary 2.11 in [36] if $w \geq d$ and $p(d,w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$, where $\kappa(d) := \sqrt[d]{d!} > d/e$ (Sterling approximation), then for any $\hat{\sigma} \in \mathbb{R}_+$

$$\begin{aligned}
\sum_{\mathbf{k}\in\mathbb{N}_0^d : \sum_{i=1}^d k_i > w} \exp\left(-\sum_{n=1}^d \hat{\sigma} k_n \right) &\leq \sum_{\mathbf{k}\in\mathbb{N}_0^d : \hat{\sigma} \sum_{i=1}^d k_i \geq w\hat{\sigma}} \exp\left(-\sum_{n=1}^d \hat{\sigma} k_n \right) \\
&\leq \hat{\sigma} d e \left(\frac{e^{\hat{\sigma}}}{1 - e^{-\hat{\sigma}}} \right)^d \exp\left(-\frac{d}{e} \hat{\sigma} p^{\frac{1}{d}} \right) p^{\frac{d-1}{d}}.
\end{aligned} \tag{A4}$$

where $\mathbf{k} \in \mathbb{N}_0^d$ and $\mathbf{k} := (k_1, \ldots, k_d)$.

Following the same approach as in [36] observe that for $0 < \delta < 1$ we can obtain a bounded constant $c_{n,\delta} \leq (e\sigma\delta)^{-1}$ such that $m_n \exp(-\sigma m_n) \leq (e\sigma\delta)^{-1}$ $\exp(-\sigma m_n(1-\delta))$. Set $\hat{\sigma} := \sigma(1-\delta)$ and by combining equations (A3) and (A4) we have proven the following result.

Lemma 10 *Suppose that* $0 < \delta < 1$, $\hat{\sigma} := \sigma(1-\delta)$, *and* $p(d,w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$ *then* $\|(I - \mathcal{S}_{w,d})v(\mathbf{y})\|_{L^\infty(\Gamma^d)} \leq \frac{C(\tilde{M},\sigma)^d e^{d-\sigma(1-\delta)+1} \hat{\sigma} d}{(\sigma\delta)^d} \left(\frac{e^{\hat{\sigma}}}{1-e^{-\hat{\sigma}}}\right)^d \exp\left(-\frac{d}{e}\hat{\sigma} p^{\frac{1}{d}}\right) p^{\frac{d-1}{d}}$.

Remark 15 The restriction $p(d,w) \geq \left(\frac{2d}{\kappa(d)}\right)^d$ is not strict and can be relaxed such that sub-exponential convergence is still obtained. We refer the reader to the bound of the Gamma function in Lemma 2.5 ([36]) and it's application in the proofs of Theorem 2.10 and Corollary 2.11.

Appendix B Experimental setup

1. **Matlab, C/C++ and MKL:** The binary tree, multilevel basis construction, formation of the sparse matrix $\tilde{\mathbf{C}}_{\mathbf{W}}$, estimation and prediction components are written and executed on Matlab [37]. However, the computational bottlenecks are executed by C/C++ software packages, Intel MKL [41], and the highly optimized BLAS and LAPACK packages contained in MATLAB. The C/C++ interfaces to matlab are constructed as dynamic shared libraries.
2. **Direct and fast summation:** The matlab code estimates the computational cost between the direct and fast summation methods and chooses the most efficient approach. For the direct method a combination of Graphic Processing Unit (GPU) and MKL intel libraries are used. For the fast summation method the KIFMM ($d = 3$) c++ code is used. The KIFMM is modified to include a Hermite interpolant approximation of the Matérn covariance function, which is implemented with the intel MKL package [41] (see [16] for details).
3. **Dynamic shared libraries:** These are produced with the GNU gcc/g++ packages. These libraries implement the Hermite interpolant with the intel MKL package (about 10 times faster than Matlab Matérn interpolant) and link the MATLAB code to the KIFMM.
4. **Cholesky and determinant computation:** The Suite Sparse 4.2.1 package ([31–35]) is used for the determinant computation of the sparse matrix $\tilde{\mathbf{C}}_{\mathbf{W}}(\boldsymbol{\theta})$.

The code is tested on a single CPU (4 core Intel i7-3770 CPU @ 3.40GHz.), one Nvidia 970 GTX GPU, with Linux Ubuntu 18.04 and 32 GB memory. In addition, the Boston University Shared Computing Cluster was used to generate test data. To test the effectiveness of the Multilevel solver the following data sets are generated:

1. **Random n-sphere data set:** The set of nested random observation $\mathbf{S}^d_0 \subset \cdots \subset \mathbf{S}^d_9$ vary from 1,000, 2000, 4000 to 256,000 knots generated on the n-sphere $\mathbf{S}_{d-1} := \{\mathbf{x} \in \mathbb{R}^d \ : \ \|\mathbf{x}\|_2 = 1\}$.
2. **Random hypercube data set:** The set of random observation locations $\mathbf{C}^d_0, \dots, \mathbf{C}^d_{10}$ vary from 1,000, 2000, 4,000 to 512,000 knots generated on the hypercube $[-1, 1]^d$ for d dimensions. The observations locations are also nested, i.e. $\mathbf{C}^d_0 \subset \cdots \subset \mathbf{C}^d_{10}$.
3. **Normal test data set** The set of observations values $\mathbf{Z}^d_0$, $\mathbf{Z}^d_1$, $\dots \mathbf{Z}^d_5$ are formed from the Gaussian random field model (1) for 1,000, 2,000, $\dots$ $256,000$ observation locations. The data set $\mathbf{Z}^d_n$ is generated from the set of nodes $\mathbf{S}^d_n$, with the covariance parameters (ν, ρ) and the corresponding set of monomials $\mathcal{Q}^d_w$. The Boston University Shared Computing Cluster was used to generate the normal test data.

Remark 16 All the timings for the numerical tests are given in wall clock times i.e. the actual time that is needed to solve a problem. This is to distinguish it from CPU time, which can be significantly smaller and may not accurately reflect the real-world time taken for solving a problem.

Appendix C Proofs

Proof of proposition 1

The proof is immediate.

Proof of lemma 2

Starting at the finest level t, for each cell $B^t_k \in \mathcal{B}^t$ there is at most p multilevel vectors. Since there is at most 2^t cells then there is at most $2^t p$ multilevel vectors.

Now, for each pair of left and right (siblings) cells at level t the parent cell at level $t-1$ will have at most $2p$ scaling functions. Thus at most p multilevel vectors and p scaling vectors are obtained that are to be used for the next level. Now, the rest of the cells at level t are leafs and will have at most p multilevel vectors and p scaling vectors that are to be used for the next level. Since there is at most 2^{t-1} cells at level $t-1$, there is at most $2^{t-1}p$ multilevel vectors. Now, follow an inductive argument until $q=0$ and the proof is done.

Proof of lemma 3

For any leaf cell at the bottom of the tree (level t) there is at most $2p$ observations. cell has at most $4p$ observations, thus the associated multilevel vectors has $4p$ non zero entries. By induction at any level l the number of nonzero entries is at most $2^{t-q+1}p$. Now for any leaf cell at any other level $l < t$ the number of nonzero entries is at most $2p$. Following an inductive argument the result is obtained.

Proof of proposition 4

Let us look at the cost of computing all the interactions between any two cells $B^i_k \in \mathcal{B}^i$ and $B^j_l \in \mathcal{B}^j$. Without loss of generality assume that $i \leq j$. For the cell B^l_k there is at most p multilevel vectors and from Lemma 3 $2^{t-i+1}p$ non zero entries. Similarly for B^j_l. All the interactions $(\boldsymbol{\psi}^i_{\tilde{k}})^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}^j_{\tilde{l}}$ now have to be computed, where $\boldsymbol{\psi}^i_{\tilde{k}} \in B^i_k$ and $\boldsymbol{\psi}^j_{\tilde{l}} \in B^j_l$.

The term $\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}^j_{\tilde{l}}$ is computed using a FMM with $2^{t-j+1}p$ sources and $2^{t-i+1}p$ targets at a cost of $\mathcal{O}(\ (2^{t-j+1}p + 2^{t-i+1}\tilde{p})^{\alpha})$. Since there is at most p multilevel vectors in B^i_k and B^j_l then the cost for computing all the interactions $(\boldsymbol{\psi}^i_{\tilde{k}})^{\mathrm{T}}\mathbf{C}(\boldsymbol{\theta})\boldsymbol{\psi}^j_{\tilde{l}}$ is $\mathcal{O}(p(2^{t-j+1}p + 2^{t-i+1}p)^{\alpha} + 2^{t-i+1}p)$.

Now, at any level i there is at most 2^i cells, thus the result follows.

Proof of proposition 5

A simple extension of the proof in Proposition 1.

Proof of theorem 6

Immediate.

Proof of theorem 7

We first have that

$$\sum_{k=1}^{N}\sum_{l=1}^{N}\phi(\mathbf{x}_k,\mathbf{y}_l;\ \boldsymbol{\theta})\boldsymbol{\psi}^i_m[k]\boldsymbol{\psi}^j_q[l] = \sum_{k=1}^{N}\sum_{l=1}^{N}\lim_{g\to\infty}(\mathbf{I}^d_g\otimes\mathbf{I}^d_g)[\phi(\mathbf{x}_k,\mathbf{y}_l;\ \boldsymbol{\theta})]\boldsymbol{\psi}^i_m[k]\boldsymbol{\psi}^j_q[l]$$
$$= \sum_{k=1}^{N}\sum_{l=1}^{N}(I_d-\mathcal{S}_{w,d})\otimes(I_d-\mathcal{S}_{w,d})[\phi(\mathbf{x}_k,\mathbf{y}_l;\ \boldsymbol{\theta})]\boldsymbol{\psi}^i_m[k]\boldsymbol{\psi}^j_q[l].$$

The last equality follows from $\boldsymbol{\psi}^i_m,\boldsymbol{\psi}^j_q\in\mathcal{P}^p(\mathbb{S})^\perp$. We now have that

$$\sum_{k=1}^{N}\sum_{l=1}^{N}\|(I_d-\mathcal{S}_{w,d})\otimes(I_d-\mathcal{S}_{w,d})[\phi(\mathbf{x}_k,\mathbf{y}_l;\ \boldsymbol{\theta})]\|_{L^\infty_\rho(\Gamma^d)}|\boldsymbol{\psi}^i_m[k]||\boldsymbol{\psi}^j_q[l]|$$
$$\leq \|(I_d-\mathcal{S}_{w,d}))[\phi(\mathbf{x}_k,\mathbf{y}_l;\ \boldsymbol{\theta})]\|^2_{L^\infty_\rho(\Gamma^d)}\sum_{k=1}^{N}\sum_{l=1}^{N}|\boldsymbol{\psi}^i_m[k]||\boldsymbol{\psi}^j_q[l]|.$$

Since $\boldsymbol{\psi}^i_m$ and $\boldsymbol{\psi}^j_q$ are orthonormal then $\sum_{k=1}^N\sum_{l=1}^N|\boldsymbol{\psi}^i_m[k]||\boldsymbol{\psi}^j_q[l]| \leq \sqrt{n_m n_q}\|\boldsymbol{\psi}^i_m[k]\|_{l^2}\|\boldsymbol{\psi}^j_q[l]\|_{l^2} = \sqrt{n_m n_q}$. From Lemma 10 the result follows.

Proof of theorem 8

The polynomial function is an entire function. However, the function $K_\nu(\vartheta)$ and $\vartheta^{\frac{1}{2}}$ are analytic for all $\vartheta\in\mathbb{C}$ except at the branch cut $(-\infty,0]$. Thus it is sufficient to check the analytic extension of $r(\boldsymbol{\theta}) = \left(\sum_{k=1}^d\theta_k(x_k-y_k)^2\right)^{\frac{1}{2}}$. Let $z\in\mathbb{C}$ be the complex extension of $r\in\mathbb{R}$. More precisely, $z=\left(\sum_{k=1}^d\theta_k z_k^2\right)^{\frac{1}{2}}$, where $z_k\in\mathbb{C}$ is the complex extension of (x_k-y_k).

Let $\vartheta=\sum_{k=1}^d\theta_k z_k^2$, then by taking the appropriate branch $\operatorname{Re} z = r_\vartheta \cos(\theta_\vartheta/2)$, where $r^2_\vartheta = (\operatorname{Re}\vartheta)^2+(\operatorname{Im}\vartheta)^2$ and $\theta_\vartheta=\tan^{-1}\frac{\operatorname{Im}\vartheta}{\operatorname{Re}\vartheta}$. Due to the branch cut at $(-\infty,0]$ we impose the restriction that $\operatorname{Re}\vartheta>0$ as x_k and y_k are extended in the complex plane. Consider any two cells $B^i_m\in\mathcal{B}^i$ and $B^j_q\in\mathcal{B}^j$, at levels i and j with the associated distance criterion constant $\tau_{i,j}>0$. From Algorithms 4, 5, 6 7, for any observations $\mathbf{x}^*\in B^i_m$ and $\mathbf{y}^*\in B^j_q$ we have that $(x^*_k-y^*_k)^2\geq\tau^2_{i,j}$ for $k=1,\dots,d$. For the rest of the discussion it is assumed that complex extension is respect to each component $k=1,\dots,d$ unless otherwise specified.

Extend $\alpha_k\to\alpha_k+v_k$ and $\gamma_k\to\gamma_k+w_k$ where $v_k:=v^R_k+iv^I_k$, $w_k:=w^R_k+iw^I_k$, and $v^R_k,v^I_k,w^R_k,w^I_k\in\mathbb{R}$. Let $\tilde{x}_k$ be the extension of x_k in the complex plane and similarly for $\tilde{y}_k$. It follows that $\tilde{x}^R_k:=\operatorname{Re}\tilde{x}_k=\frac{1}{2}(\alpha_k+1+v^R_k)a_k+b_k$, $\tilde{x}^I_k=\operatorname{Im}\tilde{x}_k=\frac{v^I_k}{2}a_k$, $y^R_k:=\operatorname{Re}\tilde{y}_k=\frac{1}{2}(\gamma_k+1+w^R_k)c_k+d_k$, and $y^I_k:=\operatorname{Im}\tilde{y}_k=$

$\frac{w_k^I}{2} c_k$. After some manipulation

$$\begin{aligned} \operatorname{Re} z_k^2 = (\tilde{x}_k^R - \tilde{y}_k^R)^2 - (\tilde{x}_k^I - \tilde{y}_k^I)^2 = (x_k - y_k)^2 + \frac{1}{4}(v_k^R a_k - w_k^R c_k) \\ + (x_k - y_k)(v_k^R a_k - w^R c_k) - \frac{1}{4}(a_k v_k^I - c_k w_k^I)^2. \end{aligned} \tag{C5}$$

Recall that $(x_k - y_k)^2 \geq \tau_{i,j}^2$ and suppose that there is a positive constant $\delta_k > 0$ such that

$$\delta_k \leq \frac{\sqrt{32\,\tau_{i,j}^2 + 8\,\tau_{i,j} + 1} - 1 - 4\,\tau_{i,j}}{4\,\tau_{i,j}}. \tag{C6}$$

Assume that $|v_k^R| \leq \tau_{i,j}\delta_k/a_k$, $|v_k^I| \leq \tau_{i,j}\delta_k/a_k$, $|w_k^R| \leq \tau_{i,j}\delta_k/c_k$, and $|w_k^I| \leq \tau_{i,j}\delta_k/c_k$. From equations (C5) and (C6) it follows that

$$\operatorname{Re} z_k^2 \geq \tau_{i,j}^2(1 - 4\delta_k^2) - \frac{\tau_{i,j}\delta_k}{2} > 0. \tag{C7}$$

Furthermore,

$$\begin{aligned} |\operatorname{Re} z_k^2| &\leq (\max\{|y_k^{max} - x_k^{min}|, |x_k^{max} - y_k^{min}|\})^2 \\ &\quad + \frac{1}{2}\tau_{i,j}\delta_k + \max\{|y_k^{max} - x_k^{min}|, |x_k^{max} - y_k^{min}|\} 2\tau_{i.j}\delta_k + \tau_{i,j}^2\delta_k^2 \\ &\leq 1 + \frac{5}{2}\tau_{i,j}\delta_k + \tau_{i,j}^2\delta_k^2. \end{aligned} \tag{C8}$$

Similarly,

$$|\operatorname{Im} z_k^2| = |2(\tilde{x}_k^R - \tilde{y}_k^R)(x_k^I - y_k^I)| \leq 2\tau_{i.j}\delta_k + 4\tau_{i,j}^2\delta_k^2. \tag{C9}$$

We now show how α_k and γ_k can be extended into the Bernstein ellipses $\mathcal{E}_{\sigma_k^\alpha}$ and $\mathcal{E}_{\sigma_k^\gamma}$, for some $\sigma_k^\alpha > 0$ and $\sigma_k^\gamma > 0$ such that $\operatorname{Re} z_k^2 > 0$. Recall that $|v_k^R| \leq \tau_{i,j}\delta_k/a_k$, $|v_k^I| \leq \tau_{i,j}\delta_k/a_k$, $|w_k^R| \leq \tau_{i,j}\delta_k/c_k$, and $|w_k^I| \leq \tau_{i,j}\delta_k/c_k$. These restrictions form a region in $\mathbb{C} \times \mathbb{C}$ and a Bernstein ellipse is embedded (See Figure C2). This is done by solving the following equation: $\frac{e^{\sigma_k^\alpha} + e^{-\sigma_k^\alpha}}{2} = 1 + \frac{\tau_{i,j}\delta_k}{a_k}$. The unique solution is $\sigma_k^\alpha = \cosh^{-1}\left(1 + \frac{\tau_{i,j}\delta_k}{a_k}\right)$ with $\sigma_k^\alpha > 0$. Following a similar argument we have that $\sigma_k^\gamma = \cosh^{-1}\left(1 + \frac{\tau_{i,j}\delta_k}{c_k}\right)$ with $\sigma_k^\gamma > 0$. Let $\mathcal{E}_\alpha^d := \prod_{k=1}^d \mathcal{E}_{\sigma_k^\alpha}$ and $\mathcal{E}_\gamma^d := \prod_{k=1}^d \mathcal{E}_{\sigma_k^\gamma}$. It follows that

$$\operatorname{Re} \vartheta \geq \sum_{k=1}^d \theta_k \operatorname{Re} z_k^2 \geq \sum_{k=1}^d \theta_k \left(\tau_{i,j}^2(1 - 4\delta_k^2) - \frac{\tau_{i,j}\delta_k}{2}\right) > 0. \tag{C10}$$

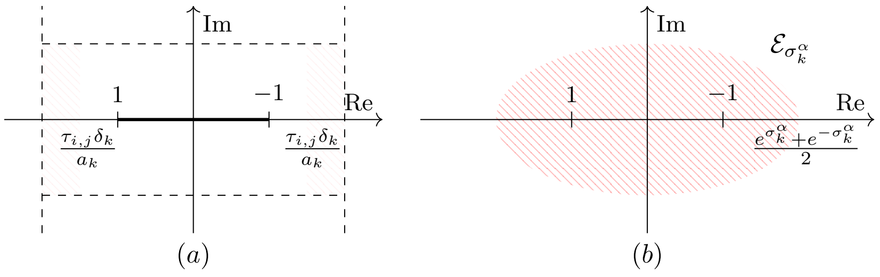
Im
1
−1
Re
$\frac{\tau_{i,j}\delta_k}{a_k}$
$\frac{\tau_{i,j}\delta_k}{a_k}$
(a)
Im
$\mathcal{E}_{\sigma_k^\alpha}$
1
−1
Re
$\frac{e^{\sigma_k^\alpha}+e^{-\sigma_k^\alpha}}{2}$
(b)

Fig. C2 (a) Region of Complex extension of α_k. (b) Embedding of Bernstein ellipse $\mathcal{E}_{\sigma_k^\alpha}$.

Thus there exist an analytic extension of $\phi(r;\ \boldsymbol{\theta}) : \Gamma^d \times \Gamma^d \to \mathbb{R}$ on $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$.

The next step is to bound the absolute value of the Matérn covariance function $|\phi(z;\ \boldsymbol{\theta})|$ in $\mathcal{E}^d_\alpha \times \mathcal{E}^d_\gamma$. If $\nu > \frac{1}{2}$ and $\operatorname{Re} z > 0$ then the modified Bessel function of the second kind satisfies the following identity: $K_\nu(\sqrt{2\nu}z) = \frac{\sqrt{\pi}(\sqrt{2\nu}z)^\nu}{2^\nu\Gamma(\nu+\frac{1}{2})}\int_1^\infty (t^2-1)^{\nu-\frac{1}{2}} \exp(-\sqrt{2\nu}zt)\,\mathrm{d}t$. It is not hard to show that for $\nu > \frac{1}{2}$ and $\operatorname{Re} z > 0$, we have that $|K_\nu(\sqrt{2\nu}z)| \le \frac{|\sqrt{2\nu}z|^\nu}{(\operatorname{Re}\sqrt{2\nu}z)^\nu} K_\nu(\sqrt{2\nu}\operatorname{Re} z)$. Note that $r_\vartheta \ge \operatorname{Re}\vartheta > 0$. From equation (C9) we have that $\operatorname{Im}\vartheta = \sum_{k=1}^d \theta_k \operatorname{Im} z_k^2 \le \sum_{k=1}^d 2\tau\delta_k + 4\tau^2\delta_k^2$. From equation (C10) $|\theta_\vartheta| \le \xi(\boldsymbol{\theta}, \boldsymbol{\delta}, \tau_{i,j})$.

Since $K_\nu(\cdot)$ is strictly completely monotonic [42] then

$$|K_\nu(\sqrt{2\nu}\operatorname{Re} z)| = |K_\nu\ (\sqrt{2\nu}r_\vartheta\cos(\theta_\vartheta/2))| \le |K_\nu\Big(\sqrt{\frac{\nu}{2}}\cos(\xi(\boldsymbol{\theta},\boldsymbol{\delta},\tau)/2) \sum_{k=1}^d \theta_k\Big(\tau_{i,j}^2(1-4\delta_k^2) - \frac{\tau_{i,j}\delta_k}{2}\Big)\Big)|. \tag{C11}$$

From equations (C8) (C9)

$$|z_k|^2 \le |\operatorname{Re} z_k^2| + |\operatorname{Im} z_k^2| \le \mathcal{R}(\delta_k, \tau_{i,j}) := 1 + \frac{9}{2}\tau_{i,j}\delta_k + 5\tau_{i,j}^2\delta_k^2$$

and therefore

$$|z| \le |\sum_{k=1}^d \theta_k z_k^2|^{\frac{1}{2}} \le \left(\sum_{k=1}^d \theta_k|z_k|^2\right)^{\frac{1}{2}} \le \left(\sum_{k=1}^d \theta_k\mathcal{R}(\delta_k,\tau_{i,j})\right)^{\frac{1}{2}}. \tag{C12}$$

By combining equations (C9), (C10), (C11), and (C12), we have now proven the Theorem.

Declarations

- This material is based upon work supported by the National Science Foundation under Grant No. 1736392 and No. 2319011.
- The authors declared that they have no conflict of interest.

References

[1] Cressie, N.A., Johannesson, G.: Spatial prediction for massive datasets. Faculty of Engineering and Information Sciences (2006). Papers: Part A. 5976

[2] Cressie, N.: Statistics for Spatial Data. John Wiley & Sons, Incorporated, New York, UNITED STATES (1993)

[3] Henderson, C.R.: Estimation of genetic parameters. Ann. Math. Stat. **21**(2), 309–310 (1950)

[4] Henderson, C.R.: Best linear unbiased estimation and prediction under a selection model. Biometrics **31**(2), 423–447 (1975)

[5] Liu, X.: Methods and applications of longitudinal data analysis. Academic Press, Oxford (2016)

[6] Liu, X.-Q., Rong, J.-Y., Liu, X.-Y.: Best linear unbiased prediction for linear combinations in general mixed linear models. Journal of Multivariate Analysis **99**(8), 1503–1517 (2008)

[7] Stein, M.L.: Interpolation of Spatial Data : Some Theory for Kriging. Springer series in statistics. Springer, New York (1999)

[8] Jabbari, F.: Chapter 3: Eigenvalues, Singular Values and Pseudo Inverse. University of California Irvine, Irvine, California. http://gram.eng.uci.edu/~fjabbari/me270b/me270b.html

[9] Minden, V., Damle, A., Ho, K.L., Ying, L.: Fast spatial gaussian process maximum likelihood estimation via skeletonization factorizations. ArXiv (arXiv:1603.08057v3) (2016)

[10] Nowak, W., Litvinenko, A.: Kriging and spatial design accelerated by orders of magnitude: Combining low-rank covariance approximations with fft-techniques. Mathematical Geosciences **45**(4), 411–435 (2013)

[11] Khoromskij, B.N., Litvinenko, A., Matthies, H.G.: Application of hierarchical matrices for computing the karhunen–loève expansion. Computing **84**(1-2), 49–67 (2009)

[12] Litvinenko, A., Sun, Y., Genton, M.G., Keyes, D.E.: Likelihood approximation with hierarchical matrices for large spatial datasets. Computational Statistics & Data Analysis **137**, 115–132 (2019)

[13] Geoga, C.J., Anitescu, M., Stein, M.L.: Scalable gaussian process computations using hierarchical matrices. Journal of Computational and Graphical Statistics **29**(2), 227–237 (2020)

[14] Schäfer, F., Sullivan, T.J., Owhadi, H.: Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity. arXiv 1706.02205 (2020)

[15] Schäfer, F., Katzfuss, M., Owhadi, H.: Sparse cholesky factorization by kullback–leibler minimization. SIAM Journal on Scientific Computing **43**(3), 2019–2046 (2021)

[16] Castrillón-Candás, J.E., Genton, M.G., Yokota, R.: Multi-level restricted maximum likelihood covariance estimation and kriging for large non-gridded spatial datasets. Spatial Statistics **18 Part A**, 105–124 (2016). Spatial Statistics Avignon: Emerging Patterns

[17] Castrillón-Candás, J.E., Li, J., Eijkhout, V.: A discrete adapted hierarchical basis solver for radial basis function interpolation. BIT Numerical Mathematics **53**(1), 57–86 (2013)

[18] Harbrecht, H., Multerer, M.: Samplets: Construction and scattered data compression. Journal of Computational Physics **471**, 111616 (2022)

[19] Beylkin, G., Coifman, R., Rokhlin, V.: Fast wavelet transforms and numerical algorithms I. Communications on Pure and Applied Mathematics **44**(2), 141–183 (1991)

[20] Nobile, F., Tempone, R., Webster, C.: A sparse grid stochastic collocation method for partial differential equations with random input data. SIAM Journal on Numerical Analysis **46**(5), 2309–2345 (2008)

[21] Nobile, F., Tempone, R., Webster, C.: An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. SIAM Journal on Numerical Analysis **46**(5), 2411–2442 (2008)

[22] Castrillon-Candas, J.E., Nobile, F., Tempone, R.: Analytic regularity and collocation approximation for pdes with random domain deformations. Computers and Mathematics with applications **71**(6), 1173–1197 (2016)

[23] Castrillón-Candás, J.E., Xu, J.: A stochastic collocation approach for parabolic PDEs with random domain deformations. Computers & Mathematics with Applications **93**, 32–49 (2021)

[24] Castrillón-Candás, J.E., Nobile, F., Tempone, R.F.: A hybrid collocation-perturbation approach for PDEs with random domains. Advances in Computational Mathematics **47**(3), 40 (2021)

[25] Li, W., Wang, X., Sun, Y., Milanovic, S., Kon, M., Castrillón-Candás, J.E.: Multilevel stochastic optimization for imputation in massive medical data records. IEEE Transaction on Big Data (2023). To Appear

[26] Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables. Applied mathematics series. Dover Publications, Mineola, New York, USA (1964)

[27] Dasgupta, S., Freund, Y.: Random projection trees and low dimensional manifolds. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing. STOC '08, pp. 537–546. ACM, New York, NY, USA (2008)

[28] Berg, M.d., Cheong, O., Kreveld, M.v., Overmars, M.: Computational Geometry: Algorithms and Applications, 3rd ed. edn. Springer, Santa Clara, CA, USA (2008)

[29] Ying, L., Biros, G., Zorin, D.: A kernel-independent adaptive fast multipole method in two and three dimensions. Journal of Computational Physics **196**(2), 591–626 (2004)

[30] Nielsen, H.B., Lophaven, S.N., Søndergaard, J.: DACE - A Matlab Kriging Toolbox. Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby (2002)

[31] Chen, Y., Davis, T.A., Hager, W.W., Rajamanickam, S.: Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. ACM Trans. Math. Softw. **35**(3), 22–12214 (2008)

[32] Davis, T.A., Hager, W.W.: Dynamic supernodes in sparse cholesky update/downdate and triangular solves. ACM Trans. Math. Softw. **35**(4), 27–12723 (2009)

[33] Davis, T., Hager, W.: Row modifications of a sparse cholesky factorization. SIAM Journal on Matrix Analysis and Applications **26**(3), 621–639 (2005)

[34] Davis, T., Hager, W.: Multiple-rank modifications of a sparse cholesky factorization. SIAM Journal on Matrix Analysis and Applications **22**(4), 997–1013 (2001)

[35] Davis, T., Hager, W.: Modifying a sparse cholesky factorization. SIAM

Journal on Matrix Analysis and Applications **20**(3), 606–627 (1999)

[36] Griebel, M., Oettershagen, J.: On tensor product approximation of analytic functions. Journal of Approximation Theory **207**, 348–379 (2016)

[37] MATLAB: R2019a. The MathWorks Inc., Natick, Massachusetts (2019)

[38] Bäck, J., Nobile, F., Tamellini, L., Tempone, R.: Stochastic Spectral Galerkin and Collocation Methods for PDEs with Random Coefficients: A Numerical Comparison. In: Hesthaven, J.S., Rønquist, E.M. (eds.) Spectral and High Order Methods for Partial Differential Equations. Lecture Notes in Computational Science and Engineering, vol. 76, pp. 43–62. Springer, Heidelberg, Germany (2011)

[39] Khoromskij, B.N.: Tensor Numerical Methods in Scientific Computing. De Gruyter, Inc., Berlin/Boston, UNITED STATES (2018)

[40] Babuska, I., Nobile, F., Tempone, R.: A stochastic collocation method for elliptic partial differential equations with random input data. SIAM Review **52**(2), 317–355 (2010)

[41] Intel Math Kernel Library: Intel, (2018). http://software.intel.com/en-us/articles/intel-mkl/

[42] Baricz, A., Ponnusamy, S., Vuorinen, M.: Functional inequalities for modified bessel functions. Expositiones Mathematicae **29**(4), 399–414 (2011)