CrossMark

$\mathcal{H}$-matrix Accelerated Second Moment Analysis for Potentials with Rough Correlation

J. Dölz · H. Harbrecht · M. Peters

Abstract We consider the efficient solution of partial differential equations for strongly elliptic operators with constant coefficients and stochastic Dirichlet data by the boundary integral equation method. The computation of the solution's two-point correlation is well understood if the two-point correlation of the Dirichlet data is known and sufficiently smooth. Unfortunately, the problem becomes much more involved in case of roughly correlated data. We will show that the concept of the $\mathcal{H}$-matrix arithmetic provides a powerful tool to cope with this problem. By employing a parametric surface representation, we end up with an $\mathcal{H}$-matrix arithmetic based on balanced cluster trees. This considerably simplifies the implementation and improves the performance of the $\mathcal{H}$-matrix arithmetic. Numerical experiments are provided to validate and quantify the presented methods and algorithms.

Keywords $\mathcal{H}$-matrix arithmetic · Uncertainty quantification · Non-local operators

1 Introduction

Modelling and simulating boundary value problems with stochastic input parameters is of great importance for applications in engineering and science. A principal approach to such problems is the Monte Carlo approach, see e.g. [30] and the references therein. However, it is very costly to generate a large number of appropriate samples and to solve a determin-

J. Dölz · H. Harbrecht · M. Peters (✉)
Mathematisches Institut, Universität Basel, Rheinsprung 21, 4051 Basel, Switzerland
e-mail: michael.peters@unibas.ch

J. Dölz
e-mail: juergen.doelz@unibas.ch

H. Harbrecht
e-mail: helmut.harbrecht@unibas.ch

istic boundary value problem on each sample. Thus, we aim here at a direct, deterministic computation of the stochastic solution.

Deterministic approaches to solve stochastic partial differential equations have been proposed in several papers. For instance, stochastic loadings have been considered in [29,35], stochastic coefficients in [1,2,4–6,23,26,28], and stochastic domains in [21,37].

In this article, we present a new approach for the second moment analysis of stochastic, strongly elliptic potential problems with rough correlation functions, i.e. the eigenvalues of the correlation kernel provide a rather slow algebraic decay. Concretely, given a domain $D \subset \mathbb{R}^d$ and a separable, complete probability space $(\Omega, \Sigma, \mathbb{P})$, we consider the Dirichlet problem

$$\left.\begin{array}{ll} \mathcal{L}u(\omega, \mathbf{x}) = 0 & \text{for } \mathbf{x} \in D \\ u(\omega, \mathbf{x}) = f(\omega, \mathbf{x}) & \text{for } \mathbf{x} \in \Gamma := \partial D \end{array}\right\} \quad \mathbb{P}\text{-almost surely} \qquad (1)$$

with stochastic Dirichlet data $f(\omega, \mathbf{x})$. Here, the linear differential operator $\mathcal{L}$ is given by

$$(\mathcal{L}u)(\mathbf{x}) := -\sum_{i,j=1}^{d} a_{i,j}(\mathbf{x}) \frac{\partial^2 u}{\partial x_i \partial x_j}(\mathbf{x}) + \sum_{i=1}^{d} b_i(\mathbf{x}) \frac{\partial u}{\partial x_i}(\mathbf{x}) + c(\mathbf{x})u(\mathbf{x}) \qquad (2)$$

with $a_{i,j}, b_i, c \in C^\infty(D)$. We suppose $\mathcal{L}$ to be strongly elliptic, i.e. $\mathbf{y}^\mathsf{T}\mathbf{A}(\mathbf{x})\mathbf{y} \geq \underline{c}\mathbf{y}^\mathsf{T}\mathbf{y} > 0$ for all $\mathbf{y} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ and almost every $\mathbf{x} \in D$, where $\mathbf{A}(\mathbf{x}) := [a_{i,j}(\mathbf{x})]_{i,j}$. Under these conditions, the Dirichlet problem (1) is known to be uniquely solvable for appropriate Dirichlet data f, see e.g. [7].

We will later restrict ourselves to the case of constant coefficients in (2). This is a necessary restriction which guarantees, using the Malgrange–Ehrenpreis theorem, the existence of a fundamental solution. The fundamental solution is then required to reformulate the problem in terms of boundary integral equations.

For example, the Dirichlet problem for the Laplace- or the Helmholtz equation can be represented with a differential operator of the form (2) with constant coefficients.

We can compute the solution's mean

$$\mathbb{E}_u(\mathbf{x}) := \int_\Omega u(\omega, \mathbf{x}) \, d\mathbb{P}(\omega)$$

and also its two-point correlation

$$\text{Cor}_u(\mathbf{x}, \mathbf{y}) := \int_\Omega u(\omega, \mathbf{x})u(\omega, \mathbf{y}) \, d\mathbb{P}(\omega)$$

if the respective mean $\mathbb{E}_f$ and correlation Cor_f of the Dirichlet data are known. Namely, the mean $\mathbb{E}_u$ satisfies

$$\mathcal{L}\mathbb{E}_u = 0 \text{ in } D \quad \text{and} \quad \mathbb{E}_u = \mathbb{E}_f \text{ on } \Gamma \qquad (3)$$

due to the linearity of the expectation and the differential operator $\mathcal{L}$. By additionally taking into account the multi-linearity of the tensor product, one readily verifies by tensorizing (1) that

$$\begin{aligned} (\mathcal{L} \otimes \mathcal{L}) \, \text{Cor}_u &= 0 & \text{in } D \times D, \\ (\mathcal{L} \otimes \text{Id})u &= 0 & \text{on } D \times \Gamma, \\ (\text{Id} \otimes \mathcal{L})u &= 0 & \text{on } \Gamma \times D, \\ \text{Cor}_u &= \text{Cor}_f & \text{on } \Gamma \times \Gamma. \end{aligned} \qquad (4)$$

The numerical solution of problems similar to (4) have already been the topic of several articles. They all have in common that they are in some sense based on a sparse tensor product discretization of the solution. For example, the computation of the second moment, i.e. Cor_u, has been considered for elliptic diffusion problems with stochastic loadings in [35] by means of a sparse tensor product finite element method. A sparse tensor product wavelet boundary element method has been used in [21] to compute the solution's second moment for elliptic potential problems on random domains. In [17,20], the computation of the second moment was done by multilevel finite element frames. Recently, this concept has been simplified by using the combination technique, cf. [19].

We consider here another approach for the solution of (4) which does not employ a hierarchic decomposition of the ansatz and test spaces. Namely, we follow here the idea of [21] and reformulate (4) as a boundary integral equation. Then, we employ an $\mathcal{H}$-matrix discretization and solve the problem by means of the $\mathcal{H}$-matrix arithmetic. This is justified since rough two-point correlation kernels behave quite similar to kernels of boundary integral operators.

The general concept of $\mathcal{H}$-matrices and the corresponding arithmetic have at first been introduced in [13,15]. $\mathcal{H}$-matrices are feasible for the data-sparse representation of (block-) matrices which can be approximated block-wise with low-rank. Our particular realization is based on a parametric representation of the surface Γ by four-sided patches. Such parametric surface representations can be obtained directly from computer aided design (CAD). They are recently studied in the context of isogeometric analysis [18,22] and offer the advantage that they lead to balanced cluster trees. We develop a fast $\mathcal{H}$-matrix arithmetic tailored to such cluster trees which only induces a slight restriction to the generality of the $\mathcal{H}$-matrix concept. Nevertheless, the gain in performance and the much easier implementation justify this minor drawback. Especially, the $\mathcal{H}$-matrix based discretization of non-local operators on parametric surfaces has recently been studied in [18].

The rest of this article is organized as follows. In Sect. 2, we provide the theoretical background for the further considerations. Especially, we introduce the parametric surface representations which is referred to and reformulate problem (1) in terms of a boundary integral equation. Section 3 is then devoted to the Galerkin discretization of the obtained boundary integral equation. In Sect. 4, we explain in brief why sparse grids are not feasible to solve problems with rough correlation kernels. Furthermore, we introduce here the class of Matérn kernels. Section 5 is concerned with the $\mathcal{H}$-matrix arithmetic in case of balanced cluster trees. Some improvements of the conventional $\mathcal{H}$-matrix arithmetic are pointed out. In Sect. 6, we present the algorithm for the iterative solution of the linear system of equations derived in Sect. 2. This algorithm is based on the approximate computation of the inverse to the stiffness matrix which arises from the single-layer operator combined with an iterative refinement of the solution. Section 7 is dedicated to numerical experiments which validate and quantify the presented methods and algorithms. Finally, in Sect. 8, we summarize the theoretical and numerical results presented in this article and draw a conclusion.

In the following, in order to avoid the repeated use of generic but unspecified constants, by $C \lesssim D$ we mean that C can be bounded by a multiple of D, independently of parameters which C and D may depend on. Obviously, $C \gtrsim D$ is defined as $D \lesssim C$, and $C \sim D$ as $C \lesssim D$ and $C \gtrsim D$.

2 Preliminaries

2.1 Boundary Integral Formulation

In the following, we restrict ourselves to the most important case $d = 3$. Therefore, let $D \subset \mathbb{R}^3$ denote a domain with piecewise smooth and globally Lipschitz continuous surface $\Gamma := \partial D$. The coefficients of the differential operator $\mathcal{L}$, cf. (2), are assumed to be constant, i.e. $\mathbf{A} \in \mathbb{R}^{3 \times 3}$, $\mathbf{b} \in \mathbb{R}^3$ and $c \in \mathbb{R}$. Then, the fundamental solution for $\mathcal{L}$ is given by

$$\Phi(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi \sqrt{\det \mathbf{A}}} \frac{\exp\left(\mathbf{b}^{\mathsf{T}}(\mathbf{x} - \mathbf{y}) - \lambda \|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}\right)}{\|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}},$$

where $\|\mathbf{x}\|_{\mathbf{A}} := \sqrt{\mathbf{x}^{\mathsf{T}} \mathbf{A} \mathbf{x}}$ and $\lambda = \sqrt{\theta}$ if $\theta \geq 0$ or else $\lambda = -i\sqrt{|\theta|}$ with $\theta := c + \|\mathbf{b}\|_{\mathbf{A}}^2$, cf. [33]. Notice that the fundamental solution is symmetric if $\mathbf{b} = \mathbf{0}$. This will be employed later on in our numerical experiments.

Given a density $\rho \in H^{-1/2}(\Gamma)$, the single-layer potential

$$\tilde{\mathcal{S}} : H^{-1/2}(\Gamma) \to H^1(D), \quad (\tilde{\mathcal{S}}\rho)(\mathbf{x}) := \int_{\Gamma} \Phi(\mathbf{x}, \mathbf{y}) \rho(\mathbf{y}) \, d\sigma_{\mathbf{y}}$$

satisfies

$$\mathcal{L}(\tilde{\mathcal{S}}\rho)(\mathbf{x}) = 0 \quad \text{for } \mathbf{x} \in \mathbb{R}^3 \setminus \Gamma,$$

cf. [3]. Thus, for given boundary data $f \in H^{1/2}(\Gamma)$ a solution to the corresponding Dirichlet problem is obtained by solving the boundary integral equation

$$(\mathcal{S}\rho)(\mathbf{x}) := \gamma_0^{\text{int}}(\tilde{\mathcal{S}}\rho)(\mathbf{x}) = f(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Gamma. \tag{5}$$

Here, $\gamma_0^{\text{int}} : H^1(D) \to H^{1/2}(\Gamma)$ denotes the (interior) trace operator.

Remark 1 We consider here the indirect formulation by the single-layer potential since it provides a higher order of approximation to the solution of (1). Of course, also an ansatz by the double-layer potential would be possible, cf. [3,33].

For the solution of (1), it is reasonable to assume that the Dirichlet data f are contained in some Bochner space. More precisely, we have $f \in L^2_{\mathbb{P}}\left(\Omega, H^{1/2}(\Gamma)\right) \cong L^2_{\mathbb{P}}(\Omega) \otimes H^{1/2}(\Gamma)$. Thus, the solution $\rho \in L^2_{\mathbb{P}}(\Omega) \otimes H^{-1/2}(\Gamma)$ to

$$(\text{Id} \otimes \mathcal{S})\rho(\omega, \mathbf{x}) = \int_{\Gamma} \Phi(\mathbf{x}, \mathbf{y}) \rho(\omega, \mathbf{y}) \, d\sigma_{\mathbf{y}} = f(\omega, \mathbf{x}) \tag{6}$$

satisfies

$$\mathcal{L}(\text{Id} \otimes \tilde{\mathcal{S}})\rho(\omega, \mathbf{x}) = 0.$$

Together with Fubini's theorem, we arrive at

$$\int_{\Omega} (\text{Id} \otimes \mathcal{S})\rho(\omega, \mathbf{x}) \, d\mathbb{P}(\omega) = (\mathcal{S}\mathbb{E}_{\rho})(\mathbf{x}) = \mathbb{E}_f(\mathbf{x}) \tag{7}$$

and in complete analogy at $\mathcal{L}(\tilde{\mathcal{S}}\mathbb{E}_{\rho})(\mathbf{x}) = 0$. This means that, having determined the density $\mathbb{E}_{\rho}$ from the boundary integral equation (7), the solution $\mathbb{E}_u$ to (3) is given via $\tilde{\mathcal{S}}\mathbb{E}_{\rho}$.

By tensorizing (6) and integration with respect to the stochastic variable, we obtain

$$\int_{\Omega} (\text{Id} \otimes \mathcal{S})\rho(\omega, \mathbf{x})(\text{Id} \otimes \mathcal{S})\rho(\omega, \mathbf{y}) \, d\mathbb{P}(\omega) = (\mathcal{S} \otimes \mathcal{S}) \text{Cor}_{\rho}(\mathbf{x}, \mathbf{y}) = \text{Cor}_f(\mathbf{x}, \mathbf{y}). \tag{8}$$

Furthermore, it holds

$$(\mathcal{L} \otimes \mathcal{L})(\tilde{S} \otimes \tilde{S}) \operatorname{Cor}_\rho = 0 \quad \text{in } D \times D,$$
$$(\mathcal{L} \otimes \gamma_0^{\text{int}})(\tilde{S} \otimes \tilde{S}) \operatorname{Cor}_\rho = 0 \quad \text{on } D \times \Gamma,$$
$$(\gamma_0^{\text{int}} \otimes \mathcal{L})(\tilde{S} \otimes \tilde{S}) \operatorname{Cor}_\rho = 0 \quad \text{on } \Gamma \times D.$$

Therefore, we conclude that $\operatorname{Cor}_u = (\tilde{S} \otimes \tilde{S}) \operatorname{Cor}_\rho$ is the solution to (4).

Obviously, a solution to (7) is easily obtained by means of standard boundary element techniques. Thus, we shall focus on the solution of (8) which is much more involved. To that end, we assume that Cor_f is *asymptotically smooth*, i.e. Cor_f satisfies the following definition.

Definition 1 Let $k \colon \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$. The function k is called *asymptotically smooth* if for some constants $r_k > 0$ and $q \in \mathbb{R}$ holds

$$\left| \partial_{\mathbf{x}}^{\boldsymbol{\alpha}} \partial_{\mathbf{y}}^{\boldsymbol{\beta}} k(\mathbf{x}, \mathbf{y}) \right| \lesssim \frac{(|\boldsymbol{\alpha}| + |\boldsymbol{\beta}|)!}{r_k^{|\boldsymbol{\alpha}| + |\boldsymbol{\beta}|}} \|\mathbf{x} - \mathbf{y}\|_2^{-2 - 2q - |\boldsymbol{\alpha}| - |\boldsymbol{\beta}|} \tag{9}$$

independently of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

Note that the term $2q$ in this definition usually reflects the order of the integral operator under consideration, e.g. we have $2q = -1$ in the case of the single-layer operator $\mathcal{S}$ and $2q = 0$ for general *Hilbert-Schmidt operators*

$$\mathcal{A} \colon L^2(\Gamma) \to L^2(\Gamma), \quad (\mathcal{A}u)(\mathbf{x}) := \int_\Gamma k(\mathbf{x}, \mathbf{y}) u(\mathbf{y}) \, d\sigma_{\mathbf{y}}. \tag{10}$$

A main feature of asymptotically smooth functions is that they exhibit a data-sparse representation by means of an $\mathcal{H}$-matrix, cf. [15].

2.2 Parametric Surface Representation

To introduce $\mathcal{H}$-matrices, we have at first to provide a hierarchical subdivision of the surface Γ. Since Γ is assumed to be piecewise smooth and globally Lipschitz continuous, it is representable by the union of several smooth *patches*, i.e.

$$\Gamma = \bigcup_{i=1}^M \Gamma_i,$$

where the intersection $\Gamma_i \cap \Gamma_{i'}$ consists at most of a common vertex or a common edge for $i \neq i'$. Each patch Γ_i is supposed to be the image of the reference domain $\square := [0, 1]^2$ under a smooth diffeomorphism $\boldsymbol{\gamma}_i$, i.e.

$$\boldsymbol{\gamma}_i \colon \square \to \Gamma_i \quad \text{with} \quad \Gamma_i = \boldsymbol{\gamma}_i(\square) \quad \text{for } i = 1, 2, \ldots, M.$$

The regularity of the mesh is ensured by a *matching condition*: for each $\mathbf{x} = \boldsymbol{\gamma}_i(\mathbf{s})$ on a common edge of Γ_i and $\Gamma_{i'}$ exists a bijective, affine mapping $\boldsymbol{\Xi} \colon \square \to \square$ with $\boldsymbol{\gamma}_i(\mathbf{s}) = (\boldsymbol{\gamma}_{i'} \circ \boldsymbol{\Xi})(\mathbf{s})$.

A mesh $\mathcal{Q}_j$ on level j for Γ is induced by dyadic subdivisions of depth j of the unit square $\square$ into 4^j congruent squares, each of which is lifted to Γ by the associated parameterization $\boldsymbol{\gamma}_i$. This construction results in a quad-tree structured sequence $\mathcal{Q}_0 \subset \mathcal{Q}_1 \subset \ldots \subset \mathcal{Q}_J$ of meshes consisting of $N_j = 4^j M$ *elements* on level j. We refer to the particular elements by $\Gamma_{i,j,k}$ where i is the index of the applied parameterization $\boldsymbol{\gamma}_i$, j is the level of the element
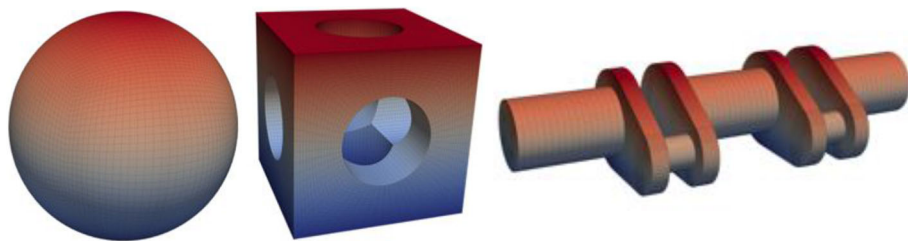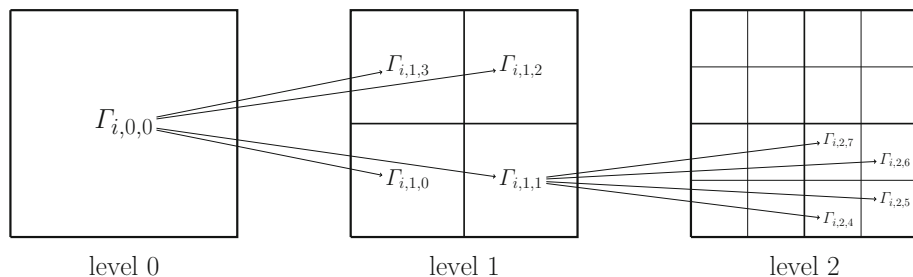
Fig. 1 Different parametric surfaces

Fig. 2 Visualization of the element tree

and k is the index of the element in hierarchical order. A visualization of the resulting quadrangulations of parametric surfaces can be found in Fig. 1.

It is convenient to refer to $\Gamma_{i,j,k}$ also as a *cluster*. In this case we think of $\Gamma_{i,j,k}$ as the union $\{\Gamma_{i,J,k'}: \Gamma_{i,J,k'} \subset \Gamma_{i,j,k}\}$, i.e. the set of all tree leafs appended to $\Gamma_{i,j,k}$ or its sons. Furthermore, we call the collection of all clusters up to the discretization level J the *cluster tree* $\mathcal{T}$. A scheme for the subdivisions of the patch Γ_i up to level 2 is shown in Fig. 2. In the following, it will also be handy to set $\boldsymbol{\lambda} := (i, j, k)$ with $|\boldsymbol{\lambda}| = j$. With regard to the tree structure of $\mathcal{T}$, we define also $\mathrm{dad}(\boldsymbol{\lambda}) := (i, j - 1, \lfloor k/4 \rfloor)$ and $\mathrm{sons}(\boldsymbol{\lambda}) := \{(i, j + 1, 4k + \ell): \ell = 0, \dots, 3\}$.

Note that the current setup refers to the framework of [18] where black-box boundary element methods for the efficient solution of boundary integral equations on *parametric surfaces* have been considered. As it turns out, the assembly and the arithmetics of $\mathcal{H}$-matrices are remarkably sped up due to the special structure of the geometry which results in balanced trees. Nevertheless, the special choice of the surface representation does not impose a restriction to the applicability of the presented algorithms. In fact, one could also introduce any other clustering of the surface which results in balanced trees.

2.3 Block-Cluster Tree

For the discretization of a Hilbert-Schmidt operator with asymptotical smooth kernel, it seems natural to introduce a subdivision of the preimage space $\Gamma \times \Gamma$ which separates smooth and non-smooth areas of the kernel function. Therefore, we choose a special sub-tree of the level-wise Cartesian product $\mathcal{T} \times \mathcal{T} := \{\Gamma_{\boldsymbol{\lambda}} \times \Gamma_{\boldsymbol{\lambda}'}: \Gamma_{\boldsymbol{\lambda}}, \Gamma_{\boldsymbol{\lambda}'} \in \mathcal{T}, |\boldsymbol{\lambda}| = |\boldsymbol{\lambda}'|\}$ with respect to the following definition.

Definition 2 (i) The clusters $\Gamma_{\boldsymbol{\lambda}}$ and $\Gamma_{\boldsymbol{\lambda}'}$ with $|\boldsymbol{\lambda}| = |\boldsymbol{\lambda}'|$ are called *admissible* if

$$\max \left\{ \mathrm{diam}(\Gamma_{\boldsymbol{\lambda}}), \mathrm{diam}(\Gamma_{\boldsymbol{\lambda}'}) \right\} \leq \eta \, \mathrm{dist} \, (\Gamma_{\boldsymbol{\lambda}}, \Gamma_{\boldsymbol{\lambda}'}) \tag{11}$$

holds for some fixed $\eta \in (0, 1)$.

(ii) The largest collection of admissible blocks $\Gamma_{\lambda} \times \Gamma_{\lambda'}$ such that $\Gamma_{\text{dad}(\lambda)} \times \Gamma_{\text{dad}(\lambda')}$ is not admissible forms the *far-field* $\mathcal{F} \subset \mathcal{T} \times \mathcal{T}$ of the operator. The remaining non-admissible blocks correspond to the *near-field* $\mathcal{N}$ of the operator.

(iii) Finally, we denote by $\mathcal{B} := \mathcal{F} \cup \mathcal{N}$ the *block-cluster tree*.

The block-cluster tree can be constructed in accordance with Algorithm 1.

Algorithm 1 Construction of the block-cluster tree $\mathcal{B}$

---

procedure BUILDBLOCKCLUSTERTREE(cluster $\Gamma_{\lambda}, \Gamma_{\lambda'}$)
    if $(\Gamma_{\lambda}, \Gamma_{\lambda'})$ is admissible **then**
        sons$(\Gamma_{\lambda} \times \Gamma_{\lambda'}) := \emptyset$
    else
        sons$(\Gamma_{\lambda} \times \Gamma_{\lambda'}) := \{\Gamma_{\mu} \times \Gamma_{\mu'} : \mu \in \text{sons}(\lambda), \mu' \in \text{sons}(\lambda')\}$
        for $\mu \in \text{sons}(\lambda), \mu' \in \text{sons}(\lambda')$ **do**
            BUILDBLOCKCLUSTERTREE($\Gamma_{\mu}, \Gamma_{\mu'}$)
        end for
    end if
end procedure

Remark 2 In practical applications, Algorithm 1 terminates if the cardinality $\#\Gamma_{\lambda} = 4^{J-|\lambda|}$ of Γ_{λ} falls below a certain threshold $k \in \mathbb{N}$.

Now, with the definition of the block-cluster tree at hand, we are able to introduce $\mathcal{H}$-matrices.

Definition 3 Let $\mathcal{B}$ be a block-cluster tree and $k \in \mathbb{N}$. We define the set $\mathcal{R}(n, k)$ of *rk-matrices* by

$$\mathcal{R}(n, k) := \{\mathbf{M} \in \mathbb{R}^{n \times n} : \text{rank}(\mathbf{M}) \le k\}.$$

Then, the set of $\mathcal{H}$-matrices is defined according to

$$\mathcal{H}(\mathcal{B}, k) := \left\{\mathbf{M} \in \mathbb{R}^{N_J \times N_J} : \mathbf{M}|_{\Gamma_{\lambda} \times \Gamma_{\lambda'}} \in \mathcal{R}(4^{J-|\lambda|}, k) \text{ for all } \Gamma_{\lambda} \times \Gamma_{\lambda'} \in \mathcal{F}\right\},$$

where we assume that $\#\Gamma_{\lambda} \le k$ for all non-admissible blocks $\Gamma_{\lambda} \times \Gamma_{\lambda'} \in \mathcal{N}$.

3 Galerkin Discretization

Given $s, t \in \mathbb{R}$, we define the Sobolev spaces $H_{\text{mix}}^{s,t}(\Gamma \times \Gamma)$ of dominant mixed derivatives on $\Gamma \times \Gamma$ by

$$H_{\text{mix}}^{s,t}(\Gamma \times \Gamma) := H^s(\Gamma) \otimes H^t(\Gamma).$$

Then, the variational formulation of the boundary integral equation (8) is given as follows:

Find $\text{Cor}_{\rho} \in H_{\text{mix}}^{-1/2,-1/2}(\Gamma \times \Gamma)$ such that

$$\big((\mathcal{S} \otimes \mathcal{S}) \text{Cor}_{\rho}, v\big)_{L^2(\Gamma \times \Gamma)} = (\text{Cor}_f, v)_{L^2(\Gamma \times \Gamma)} \text{ for all } v \in H_{\text{mix}}^{-1/2,-1/2}(\Gamma \times \Gamma). \tag{12}$$

For the Galerkin discretization of (12), we fix $J \in \mathbb{N}$ and introduce the space

$$\hat{V}_J := \big\{\hat{\varphi} : \square \to \mathbb{R} : \hat{\varphi}|_{\square_{J,k}} \text{ is a polynomial of order } d\big\} \subset L^2(\square),$$

where we define $\square_{j,k} := \boldsymbol{\gamma}_i^{-1}(\Gamma_{i,j,k})$. Then, the univariate ansatz space V_J on level J is given by

$$V_J := \left\{ \hat{\varphi} \circ \boldsymbol{\gamma}_i^{-1} : \hat{\varphi} \in \hat{V}_J, \ i = 1, \ldots, M \right\} \subset H^{-1/2}(\Gamma).$$

For different values of J, the spaces V_J are nested, i.e. $V_0 \subset V_1 \subset \cdots \subset V_J$. The Sobolev smoothness t of V_J depends on the global smoothness of the functions $\varphi \in V_J$. Especially, for transported piecewise constant functions ($d = 1$), we have $t < 1/2$, which is sufficient for the Galerkin discretization of (8). Note that, due to the parametric surface representation, the Galerkin discretization may be performed on the reference domain $\square$ with respect to the space $\hat{V}_J$. For the details we refer to [18].

By replacing the energy space $H_{\text{mix}}^{-1/2,-1/2}(\Gamma \times \Gamma)$ in the variational formulation (12) by the finite dimensional ansatz space $V_J \otimes V_J \subset H_{\text{mix}}^{-1/2,-1/2}(\Gamma \times \Gamma)$, we arrive at the Galerkin formulation for the boundary integral equation (8):

> Find $\text{Cor}_{\rho,J} \in V_J \otimes V_J$ such that
> $$\left((\mathcal{S} \otimes \mathcal{S})\text{Cor}_{\rho,J}, v\right)_{L^2(\Gamma \times \Gamma)} = (\text{Cor}_f, v)_{L^2(\Gamma \times \Gamma)} \quad \text{for all} \quad v \in V_J \otimes V_J. \tag{13}$$

We choose a basis $\{\varphi_\ell \otimes \varphi_{\ell'}\}_{\ell,\ell'}$ and represent $\text{Cor}_{\rho,J}$ by its basis expansion

$$\text{Cor}_{\rho,J} = \sum_{\ell,\ell'=1}^{N_J} c_{\rho,\ell,\ell'}(\varphi_\ell \otimes \varphi_{\ell'}).$$

Then, setting $\mathbf{C}_\rho := [c_{\rho,\ell,\ell'}]_{\ell,\ell'}$, we end up with the linear system of equations

$$(\mathbf{S} \otimes \mathbf{S})\,\text{vec}(\mathbf{C}_\rho) = \text{vec}(\mathbf{C}_f), \tag{14}$$

where $\mathbf{C}_f := \left[(\text{Cor}_f, \varphi_\ell \otimes \varphi_{\ell'})_{L^2(\Gamma \times \Gamma)}\right]_{\ell,\ell'}$ is the discretized two-point correlation of the Dirichlet data f and $\mathbf{S} := \left[(\mathcal{S}\varphi_{\ell'}, \varphi_\ell)_{L^2(\Gamma)}\right]_{\ell,\ell'}$ is the system matrix of the single-layer operator. In (14), the tensor product has, as usual in connection with matrices, to be understood as the Kronecker product. Furthermore, for a matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n] \in \mathbb{R}^{m \times n}$, the operation $\text{vec}(\mathbf{A})$ is defined as

$$\text{vec}([\mathbf{a}_1, \ldots, \mathbf{a}_n]) := \begin{bmatrix} \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} \in \mathbb{R}^{mn}.$$

For matrices $\mathbf{A} \in \mathbb{R}^{k \times n}$, $\mathbf{B} \in \mathbb{R}^{\ell \times m}$ and $\mathbf{X} \in \mathbb{R}^{m \times n}$, there holds the relation

$$(\mathbf{A} \otimes \mathbf{B})\,\text{vec}(\mathbf{X}) = \text{vec}(\mathbf{B}\mathbf{X}\mathbf{A}^\mathsf{T}).$$

Hence, we may rewrite (14) as

$$\mathbf{S}\mathbf{C}_\rho\mathbf{S}^\mathsf{T} = \mathbf{C}_f. \tag{15}$$

Since Φ and Cor_f are asymptotical smooth kernel functions, the related stiffness matrices $\mathbf{S}$ and $\mathbf{C}_f$ are compressible by means of $\mathcal{H}$-matrices. How this approximation can be achieved in case of parametric surfaces is explained in detail in [18]. By representing the unknown quantity $\mathbf{C}_\rho$ of interest also in the $\mathcal{H}$-matrix format, with respect to a common block-cluster tree, we can employ the $\mathcal{H}$-matrix arithmetic to solve the equation (15).

4 On the Sparse Tensor Product Approximation

With the Galerkin discretization (14) and even a multilevel hierarchy at hand, the question arises, why one should not follow the approach of the preceding articles [17,20] and use a sparse tensor product discretization, or equivalently the combination technique, cf. [11,19], for the solution of (8). As it turns out, a discretization with $\mathcal{H}$-matrices, which can be thought of as a compressed full tensor approximation, yields a rate of convergence which is twice as high as that of the sparse tensor product approximation. With the following considerations we want to elaborate on this point. Therefore, it is convenient to fix some notation.

We denote the energy space related with (8) by $\mathfrak{H} := H_{\mathrm{mix}}^{-1/2,-1/2}(\Gamma \times \Gamma)$ and its dual by $\mathfrak{H}' = H_{\mathrm{mix}}^{1/2,1/2}(\Gamma \times \Gamma)$. Then, the additional isotropic smoothness of a function relative to the energy space is measured by the spaces

$$\mathfrak{H}^s := \big\{ v \in \mathfrak{H} \colon \|\partial_{\mathbf{x}}^{\boldsymbol{\alpha}} \partial_{\mathbf{y}}^{\boldsymbol{\beta}} v\|_{\mathfrak{H}} < \infty, \ |\boldsymbol{\alpha}| + |\boldsymbol{\beta}| \le s \big\}.$$

To the best of our knowledge, these classes of Sobolev spaces have at first been considered in [12]. In order to measure the error of the sparse tensor product approximation, we also need the anisotropic version of these spaces, that is

$$\mathfrak{H}_{\mathrm{mix}}^s := \big\{ v \in \mathfrak{H} \colon \|\partial_{\mathbf{x}}^{\boldsymbol{\alpha}} \partial_{\mathbf{y}}^{\boldsymbol{\beta}} v\|_{\mathfrak{H}} < \infty, \ |\boldsymbol{\alpha}|, |\boldsymbol{\beta}| \le s \big\}.$$

Note that it holds $\mathfrak{H}_{\mathrm{mix}}^s = H_{\mathrm{mix}}^{s-1/2,s-1/2}(\Gamma \times \Gamma)$.

Provided that $\mathrm{Cor}_\rho \in \mathfrak{H}_{\mathrm{mix}}^s$ or $\mathrm{Cor}_\rho \in \mathfrak{H}^s$ respectively, we end up with the well known error estimates for Galerkin approximation $\widehat{\mathrm{Cor}}_{\rho,J}$ in the sparse tensor product space $\widehat{V_J \otimes V_J}$ and for Galerkin approximation $\mathrm{Cor}_{\rho,J}$ in the full tensor product space $V_J \otimes V_J$, respectively. Here, the sparse tensor product space is defined by

$$\widehat{V_J \otimes V_J} := \bigcup_{j=0}^{J} V_j \otimes V_{J-j}.$$

From this representation it becomes clear that $V_{J/2} \otimes V_{J/2}$ is the finest (isotropic) full tensor product space which is contained in $\widehat{V_J \otimes V_J}$.

According to [10,12], we have

$$\|\widehat{\mathrm{Cor}}_{\rho,J} - \mathrm{Cor}_\rho\|_{\mathfrak{H}} \lesssim 2^{-Js} \sqrt{J} \| \mathrm{Cor}_\rho \|_{\mathfrak{H}_{\mathrm{mix}}^s} \text{ for } s \le d + 1/2$$

in the case of the sparse tensor product approximation and

$$\|\mathrm{Cor}_{\rho,J} - \mathrm{Cor}_\rho\|_{\mathfrak{H}} \lesssim 2^{-Js} \| \mathrm{Cor}_\rho \|_{\mathfrak{H}^s} \text{ for } s \le d + 1/2$$

in the case of the full tensor product approximation. Thus, given that Cor_ρ is sufficiently smooth, both methods yield essentially the same order of convergence. Unfortunately, the available smoothness is limited in our case.

In the framework of stochastic fields it is quite common to assume that Cor_f is *isotropic*, i.e. $\mathrm{Cor}_f(\mathbf{x}, \mathbf{y})$ depends only on the distance $r = \|\mathbf{x} - \mathbf{y}\|_2$, cf. [31]. Examples for correlation functions of this kind are given by the Matérn class of kernels, i.e.

$$\mathrm{Cor}_f(r) = k_\nu(r) := \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}r}{\ell} \right)^\nu K_\nu \left(\frac{\sqrt{2\nu}r}{\ell} \right) \tag{16}$$
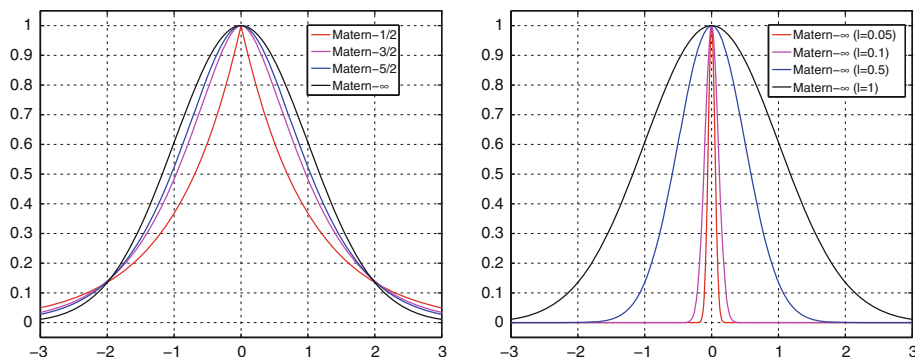
Fig. 3 Matérn kernels for different values of the smoothness parameter ν (*left*) and for different correlation lengths ℓ (*right*)

with $\ell, \nu \in (0, \infty)$. Here, K_ν denotes the *modified Bessel function of the second kind*. For half integer values of ν, i.e. $\nu = p + 1/2$ for $p \in \mathbb{N}$, the expression simplifies to

$$k_{p+1/2}(r) = \exp\left(\frac{-\sqrt{2\nu}r}{\ell}\right) \frac{p!}{(2p)!} \sum_{i=0}^{p} \frac{(p+i)!}{i!(p-i)!} \left(\frac{\sqrt{8\nu}r}{\ell}\right)^{p-i}.$$

In accordance with [31], we obtain in the limit case $\nu \to \infty$ the Gaussian kernel:

$$k_\infty(r) = \exp\left(\frac{-r^2}{2\ell^2}\right).$$

The smoothness of the kernel k_ν is controlled by the *smoothness parameter* ν and the *correlation length* ℓ. A visualization of these kernels for varying ν is given in the left plot of Fig. 3 and for varying values of ℓ in the right plot of Fig. 3. Although the correlation length ℓ does not influence the Sobolev smoothness of the kernel, it has a large impact on the preasymptotic behaviour and may cause severe numerical difficulties.

Let us assume that the two-point correlation Cor_f is contained in some isotropic Sobolev space $H^s(\Gamma \times \Gamma)$. For example, for the exponential two-point correlation, i.e. $\nu = 1/2$ in (16), we find $s = 2 - \varepsilon$ for any $\varepsilon > 0$ since the kernel is continuous with a kink on the two-dimensional diagonal. Whereas, for the Matérn kernel with $\nu = 3/2$, we find $s = 4 - \varepsilon$ for any $\varepsilon > 0$ since it is two orders smoother than the exponential kernel.

The highest order Sobolev space of dominant mixed derivatives which contains the Sobolev space $H^s(\Gamma \times \Gamma)$ is $H_{\mathrm{mix}}^{s/2,s/2}(\Gamma \times \Gamma) = \mathfrak{H}_{\mathrm{mix}}^{(s+1)/2}$. In particular, to ensure $\mathrm{Cor}_f \in \mathfrak{H}' = H_{\mathrm{mix}}^{1/2,1/2}(\Gamma \times \Gamma)$, we have to ensure that $s \geq 1$. In view of the shift property of the single-layer operator, cf. [3,33], we arrive at the regularity statement $\mathrm{Cor}_\rho \in \mathfrak{H}_{\mathrm{mix}}^{(s-1)/2}$.

For the error estimation in case of the full tensor product approximation, we make use of the following result.

Lemma 1 *Let $s \geq 1$. For $\mathrm{Cor}_f \in H^s(\Gamma \times \Gamma)$ it holds $\mathrm{Cor}_\rho \in \mathfrak{H}^{s-1}$.*

Proof The two-point correlation Cor_f satisfies

$$\mathrm{Cor}_f \in H^s(\Gamma \times \Gamma) = \bigcap_{s \geq t \geq 0} H_{\mathrm{mix}}^{s-t,t}(\Gamma \times \Gamma) \subset \mathfrak{H}'.$$

The shift properties of the single-layer operator imply thus

$$\text{Cor}_\rho \in \bigcap_{s \geq t \geq 0} H_{\text{mix}}^{s-t-1,t-1}(\Gamma \times \Gamma) \subset \mathfrak{H}.$$

From this, we derive $\|\partial_{\mathbf{x}}^{\boldsymbol{\alpha}} \partial_{\mathbf{y}}^{\boldsymbol{\beta}} \text{Cor}_\rho\|_{\mathfrak{H}} < \infty$ for all $|\boldsymbol{\alpha}| + |\boldsymbol{\beta}| \leq s - 1$ which proves the assertion. □

We conclude that, for small values of s, i.e. in case of rough correlation kernels, the rate of convergence in the full tensor product $V_J \otimes V_J$ is up to twice as high as the rate of convergence in the sparse tensor product $\widehat{V_J \otimes V_J}$. Nevertheless, by using the $\mathcal{H}$-matrix approach proposed in the subsequent section, the cost for the approximation is essentially linear for both approaches. We refer the reader to e.g. [18,35] for the computational complexity of the sparse tensor product approach.

5 $\mathcal{H}$-matrix Arithmetic

5.1 Preliminary Considerations

We assume that $\mathbf{C}_\rho$ can be approximated as an $\mathcal{H}$-matrix with respect to some block-cluster tree for Γ. Thus, there exists a common block-cluster tree $\mathcal{B}$ such that $\mathbf{C}_\rho \in \mathcal{H}(\mathcal{B}, k_1)$, $\mathbf{C}_f \in \mathcal{H}(\mathcal{B}, k_2)$ and $\mathbf{S} \in \mathcal{H}(\mathcal{B}, k_3)$. Therefore, we have $\mathbf{C}_\rho, \mathbf{C}_f, \mathbf{S} \in \mathcal{H}(\mathcal{B}, \max\{k_1, k_2, k_3\})$.

In the following, we show how to add and how to multiply $\mathcal{H}$-matrices with respect to a common block-cluster tree $\mathcal{B}$. We especially point out the simplifications which are possible with regard to the balanced cluster tree we have. More general results may be found in [14].

Let $\mathcal{B}$ denote a block-cluster tree on level $J \in \mathbb{N}$ and $\mathbf{H} \in \mathcal{H}(\mathcal{B}, k)$ an $\mathcal{H}$-matrix defined on $\mathcal{B}$. On level 0, $\mathcal{B}$ has M^2 children that correspond to M^2 quadratic submatrices $\mathbf{H}_{\ell,\ell'} \in \mathbb{R}^{4^J \times 4^J}$. Hence, we can consider the matrix $\mathbf{H}$ as an $M \times M$ block matrix, that is

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{1,1} & \dots & \mathbf{H}_{1,M} \\ \vdots & & \vdots \\ \mathbf{H}_{M,1} & \dots & \mathbf{H}_{M,M} \end{bmatrix}. \tag{17}$$

Here, each submatrix $\mathbf{H}_{\ell,\ell'}$ corresponds to a block-cluster $\Gamma_\lambda \times \Gamma_{\lambda'} \in \mathcal{B}$. If $\Gamma_\lambda \times \Gamma_{\lambda'}$ is a leaf of $\mathcal{B}$, then $\mathbf{H}_{\ell,\ell'}$ is either an rk-matrix or a full-matrix, respectively. Otherwise, $\Gamma_\lambda \times \Gamma_{\lambda'}$ has exactly 4^2 children. In this case, $\mathbf{H}_{\ell,\ell'}$ is again a block-matrix with quadratic matrix blocks $\mathbf{H}'_{\ell,\ell'} \in \mathbb{R}^{4^{J-1} \times 4^{J-1}}$.

On the one hand, in order to compute the sum $\mathbf{H}_1 + \mathbf{H}_2$ of two $\mathcal{H}$-matrices $\mathbf{H}_1, \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$, we exclusively have to explain the following elementary sums:

+	$\mathcal{H}$-matrix	rk-matrix	full-matrix
$\mathcal{H}$-matrix	recursively		
rk-matrix		approximately	
full-matrix			exactly

On the other hand, for the product $\mathbf{H}_1 * \mathbf{H}_2$ of two $\mathcal{H}$-matrices $\mathbf{H}_1, \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$, besides elementary products, we also have to introduce compound sums:

*	$\mathcal{H}$-matrix	rk-matrix	full-matrix
$\mathcal{H}$-matrix	recursively	exactly	
rk-matrix	exactly	exactly	exactly
full-matrix		exactly	exactly

+=	$\mathcal{H}$-matrix	rk-matrix	full-matrix
$\mathcal{H}$-matrix	recursively	recursively	
rk-matrix	approximately	approximately	approximately
full-matrix		exactly	exactly

In the table for the compound operations, i.e. +=, the operands in the rows of the table coincide with the target format of the respective operation. In all the preceding tables, *recursively* means that we use a recursive algorithm to compute the operation. The term *exactly* means that we can perform this operation without a truncation error. The term *approximately* indicates that we need to truncate the operation's result in order to guarantee $\mathbf{H}_1 + \mathbf{H}_2$, $\mathbf{H}_1 * \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$. Hence, the subsequent subsection is devoted to appropriate truncation operators.

5.2 Truncation Operators

We shall develop truncation operators to approximate a given matrix by an rk-matrix. Especially, we have here in mind the following situations from the preceding table for compound operations,

$$\mathbf{R}_1 += \mathbf{R}_2, \quad \mathbf{H} += \mathbf{R}, \quad \mathbf{R} += \mathbf{H}, \quad \mathbf{R} += \mathbf{F},$$

for $\mathbf{R}, \mathbf{R}_1, \mathbf{R}_2 \in \mathcal{R}(n, k)$, $\mathbf{H} \in \mathcal{H}(\mathcal{B}', k)$ and $\mathbf{F} \in \mathbb{R}^{n \times n}$. These operations can formally be introduced by means of a truncation operator.

Definition 4 Let $\mathbf{M} \in \mathbb{R}^{n \times n}$. We define the truncation operator $\mathfrak{T}$ to rk-matrices,

$$\mathfrak{T} \colon \mathbb{R}^{n \times n} \to \mathcal{R}(n, k), \quad \mathbf{M} \mapsto \mathbf{R},$$

to be the best approximation of $\mathbf{M}$ in $\mathcal{R}(n, k)$ with respect to a given norm. We call $\mathbf{R} = \mathfrak{T}(\mathbf{M})$ the truncation of $\mathbf{M}$ to rank k.

Remark 3 We may extend $\mathfrak{T}$ to $\mathcal{H}(\mathcal{B}, k)$ by the block-wise application of the truncation operator $\mathfrak{T} \colon \mathbb{R}^{n \times n} \to \mathcal{R}(n, k)$ with respect to the admissible blocks of a block-cluster tree $\mathcal{B}$.

With the truncation operator $\mathfrak{T}$ at hand, we may define

$$(\mathbf{R}_1 += \mathbf{R}_2) := \mathfrak{T}(\mathbf{R}_1 + \mathbf{R}_2) \in \mathcal{R}(n, k), \qquad (\mathbf{H} += \mathbf{R}) := \mathfrak{T}(\mathbf{H} + \mathbf{R}) \in \mathcal{H}(\mathcal{B}, k),$$
$$(\mathbf{R} += \mathbf{F}) := \mathfrak{T}(\mathbf{R} + \mathbf{F}) \in \mathcal{R}(n, k), \qquad (\mathbf{R} += \mathbf{H}) := \mathfrak{T}(\mathbf{R} + \mathbf{H}) \in \mathcal{R}(n, k).$$

For the specific realization, let the truncation operator $\mathfrak{T}$ be defined with respect to the Frobenius norm or the spectral norm. Then, the best approximation in $\mathcal{R}(n, k)$ is provided by the truncated *singular value decomposition*.

Definition 5 The singular value decomposition of a matrix $\mathbf{R} \in \mathcal{R}(n, k)$ is a decomposition of the form

$$\mathbf{R} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\mathsf{T}$$

where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n \times \tilde{k}}$ are orthogonal matrices, i.e. $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{V}^\mathsf{T}\mathbf{V} = \mathbf{I} \in \mathbb{R}^{\tilde{k} \times \tilde{k}}$, and $\mathbf{\Sigma} \in \mathbb{R}^{\tilde{k} \times \tilde{k}}$ is a diagonal matrix whose diagonal entries

$$\mathbf{\Sigma}_{1,1} \geq \cdots \geq \mathbf{\Sigma}_{\tilde{k}, \tilde{k}} > 0$$

are called *singular values*. Here, $\tilde{k} \leq k$ denotes the actual rank of $\mathbf{R}$.

Remark 4 This definition of the singular value decomposition is in contrast to the standard definition, see e.g. [9], where the matrices $\mathbf{U}$ and $\mathbf{V}$ contain a complete basis for the image space and the preimage space, respectively.

The truncation of the singular value decomposition provides the best approximation of a matrix in $\mathcal{R}(n, k)$.

Lemma 2 *Let $k_1 \leq \tilde{k} \leq k_2$. Then, for a matrix $\mathbf{R} \in \mathcal{R}(n, k_2)$ with actual rank $\tilde{k}$, the best approximation of $\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}$ in $\mathcal{R}(n, k_1)$ with respect to the Frobenius norm and the spectral norm is given by $\tilde{\mathbf{R}} = \mathbf{U}\tilde{\Sigma}V^\mathsf{T}$ where $\tilde{\Sigma} = \mathrm{diag}(\mathbf{\Sigma}_{1,1}, \ldots, \mathbf{\Sigma}_{k_1,k_1}, 0, \ldots, 0)$. In particular, it holds*

$$\|\mathbf{R} - \tilde{\mathbf{R}}\|_F = \sqrt{\sum_{i=k_1+1}^{k_2} \mathbf{\Sigma}_{ii}^2} \quad and \quad \|\mathbf{R} - \tilde{\mathbf{R}}\|_2 = \mathbf{\Sigma}_{k_1+1,k_1+1}.$$

Furthermore, $\tilde{\mathbf{R}}$ is the best approximation to $\mathbf{R}$ in the sense that

$$\|\mathbf{R} - \tilde{\mathbf{R}}\|_{F/2} = \min_{\mathbf{R}' \in \mathcal{R}(n,k_1)} \|\mathbf{R} - \mathbf{R}'\|_{F/2}.$$

The proof of this lemma is a straightforward consequence of the orthogonal invariance of the Frobenius norm and the spectral norm.

5.2.1 The Situation $\mathbf{R}_1$+=$\mathbf{R}_2$

In the following, it is convenient to assume that all matrices are in factorized form. This means, for $\mathbf{R}_1, \mathbf{R}_2 \in \mathcal{R}(n, k)$, we have $\mathbf{R}_i = \mathbf{A}_i\mathbf{B}_i^\mathsf{T}$ with $\mathbf{A}_i, \mathbf{B}_i \in \mathbb{R}^{n \times k}$. Then, without truncation, it holds $\mathbf{R}_1 + \mathbf{R}_2 = [\mathbf{A}_1\ \mathbf{A}_2][\mathbf{B}_1\ \mathbf{B}_2]^\mathsf{T} \in \mathcal{R}(n, 2k)$. The action of the truncation operator $\mathfrak{T}$ on $\mathbf{R}_1 + \mathbf{R}_2$ can now efficiently be computed with the *reduced singular value decomposition* (rSVD). Numerically, this is performed by computing the rSVD of $\mathbf{R}_1 + \mathbf{R}_2$ with Algorithm 2, cf. [14], and then truncating to the k-th dominant singular values.

Algorithm 2 Calculation of the rSVD for an rk-matrix

$\mathbf{Q}_\mathbf{A}\mathbf{R}_\mathbf{A} = $ QR-decomposition of $\mathbf{A}$, $\mathbf{Q}_\mathbf{A} \in \mathbb{R}^{n \times k}$, $\mathbf{R}_\mathbf{A} \in \mathbb{R}^{k \times k}$
$\mathbf{Q}_\mathbf{B}\mathbf{R}_\mathbf{B} = $ QR-decomposition of $\mathbf{B}$, $\mathbf{Q}_\mathbf{B} \in \mathbb{R}^{n \times k}$, $\mathbf{R}_\mathbf{B} \in \mathbb{R}^{k \times k}$
$\tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^\mathsf{T} = $ SVD$(\mathbf{R}_\mathbf{A}\mathbf{R}_\mathbf{B}^\mathsf{T})$
$\mathbf{U} = \mathbf{Q}_\mathbf{A}\tilde{\mathbf{U}}$
$\mathbf{V} = \mathbf{Q}_\mathbf{B}\tilde{\mathbf{V}}$

5.2.2 The Situation $\mathbf{H}$+=$\mathbf{R}$

With the definition of $\mathbf{R}_1$+=$\mathbf{R}_2$, we can easily explain $\mathbf{H}$+=$\mathbf{R}$ in a recursive manner. Let $\mathbf{H} \in \mathcal{H}(\mathcal{B}', k) \subset \mathbb{R}^{4^{J-j} \times 4^{J-j}}$ and $\mathbf{R} \in \mathcal{R}(4^{J-j}, k)$. We can exactly represent $\mathbf{R}$ with respect to the block-cluster structure of $\mathcal{B}'$ by recursively breaking up the rk-matrix structure. In the first step, we have

$$\mathbf{R} = \begin{array}{|c|}\hline \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} = (\mathbf{R}_{\ell,\ell'})_{\ell,\ell'}.$$

Now, proceeding this subdivision with respect to the structure of $\mathcal{B}'$ yields a situation where we only have to add either two *rk*-matrices or a full-matrix and an *rk*-matrix. The first is done with respect to the definition of $\mathbf{R}_1\mathrel{+}=\mathbf{R}_2$, the latter can be exactly performed.

Remark 5 For numerical issues, the subdivision of an *rk*-matrix with respect to the block-cluster tree $\mathcal{B}'$ can be realized by index shifts. Therefore, no additional calculations or storage are necessary here.

5.2.3 The Situation **R+=H**

In contrast to the idea of the *hierarchical approximation*, cf. [14], which is a successive approximation of the matrix $\mathbf{H} \in \mathcal{H}(\mathcal{B}', k) \subset \mathbb{R}^{4^{J-j} \times 4^{J-j}}$ by *rk*-matrices, we make another approach here: We exploit the fact that we can multiply the matrix $\mathbf{H}$ to a vector $\mathbf{v} \in \mathbb{R}^{4^{J-j}}$ with a complexity of $\mathcal{O}(2k(J-j)4^{J-j})$, cf. [14,18]. Thus, it seems reasonable to directly compute the truncated singular value decomposition of $\mathbf{H}$ up to rank k by means of a sparse eigensolver which only requires matrix-vector multiplications as for example ARPACK, cf. [25]. Then, we are again in the situation $\mathbf{R}_1\mathrel{+}=\mathbf{R}_2$. For practical issues in the product of $\mathcal{H}$-matrices, we rather have to consider the case $\mathbf{R}\mathrel{+}=\mathbf{H}_1\mathbf{H}_2$. This situation can analogously be treated by the *indirect singular value decomposition*.

5.3 The Indirect Singular Value Decomposition (iSVD)

As already pointed out, the complexity of a matrix-vector multiplication of an $\mathcal{H}$-matrix is $\mathcal{O}(2k(J-j)4^{J-j})$ for $\mathbf{H} \in \mathcal{H}(\mathcal{B}', k) \subset \mathbb{R}^{4^{J-j} \times 4^{J-j}}$ and $\mathbf{v} \in \mathbb{R}^{4^{J-j}}$. Therefore, the complexity of computing the product $(\mathbf{H}_1 \cdots \mathbf{H}_m)\mathbf{v}$ is obviously of order $\mathcal{O}(2k(J-j)4^{J-j}m)$.

In the first step, we show how the iSVD can be applied to the (exact) product of two $\mathcal{H}$-matrices in $\mathcal{H}(\mathcal{B}', k)$, i.e. $\mathbf{H}_1\mathbf{H}_2$, to compute the k largest singular values and singular vectors of this product. To that end let

$$\mathbf{W} := \begin{bmatrix} \mathbf{0} & \mathbf{H}_1\mathbf{H}_2 \\ \mathbf{H}_2^\mathsf{T}\mathbf{H}_1^\mathsf{T} & \mathbf{0} \end{bmatrix}$$

be the *Jordan-Wielandt matrix* with respect to $\mathbf{H}_1\mathbf{H}_2$. The positive eigenvalues of this matrix coincide with the singular values of $\mathbf{H}_1\mathbf{H}_2$, cf. [8,24]. The complexity of applying $\mathbf{W}$ to a vector $\mathbf{x} \in \mathbb{R}^{2 \cdot 4^{J-j}}$ is of order $\mathcal{O}(8k(J-j)4^{J-j})$. Thus, the computation of the k largest singular values of $\mathbf{W}$ can be performed within a complexity of $\mathcal{O}(8k(J-j)4^{J-j}\mathtt{ncv}^2)$, where $\mathtt{ncv}$ corresponds to the size of the Krylov subspace used for the eigenvalue approximation, cf. [25].

Remark 6 The iSVD can directly compute the best approximation of an *rk*-matrix $\mathbf{R}_{\ell,\ell'}$ resulting from a block inner-product of a row and a column in the product of two $\mathcal{H}$-matrices $\mathbf{H}_1, \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$ structured like (17):

$$\mathbf{R}_{\ell,\ell'} = \mathfrak{T}\left(\sum_{i=1}^{p} (\mathbf{H}_1)_{\ell,i}(\mathbf{H}_2)_{i,\ell'} \right).$$

Here, we have either $p = M$ for level 0 or $p = 4$ for any other level. Then, the complexity reads $\mathcal{O}(8k(J - j)4^{J-j}p \cdot \text{ncv}^2)$.

5.4 $\mathcal{H}$-matrix Multiplication

Applying recursively the procedure from Remark 6 yields the actual best approximation of the product $\mathbf{H}_1\mathbf{H}_2$ in $\mathcal{H}(\mathcal{B}, k)$. This realization of the product $\mathbf{H}_1*\mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$ is provided by Algorithm 3. Note that the calling sequence for Algorithm 3 is initiated with $p' = 1$.

Algorithm 3 Compute $\mathbf{H}_3 = \sum_{i=1}^{p'} \mathbf{H}_1^{(i)}\mathbf{H}_2^{(i)}$

---

function $\mathbf{H}_3 = \text{BestMult}\mathcal{H}(\{\mathbf{H}_1^{(i)}\}_{i=1}^{p'}, \{\mathbf{H}_2^{(i)}\}_{i=1}^{p'})$
   if $\mathbf{H}_3 \in \mathcal{H}(\mathcal{B}', k)$ **then**
      for $\ell, \ell' = 1, \ldots, p$ **do**
         $\mathcal{L} = \cup_{i=1}^{p'}\{(\mathbf{H}_1^{(i)})_{\ell,1}, \ldots, (\mathbf{H}_1^{(i)})_{\ell,p}\}$
         $\mathcal{C} = \cup_{i=1}^{p'}\{(\mathbf{H}_2^{(i)})_{1,\ell'}, \ldots, (\mathbf{H}_2^{(i)})_{p,\ell'}\}$
         $(\mathbf{H}_3)_{\ell,\ell'} = \text{BestMult}\mathcal{H}(\mathcal{L}, \mathcal{C})$
      end for
   else
      Compute $\mathbf{H}_3 = \mathfrak{T}\left(\sum_{i=1}^{p'} \mathbf{H}_1^{(i)}\mathbf{H}_2^{(i)}\right)$ with iSVD or as full matrix
   end if
end function

Unfortunately, although Algorithm 3 provides the best approximation of the $\mathcal{H}$-matrix product in $\mathcal{H}(\mathcal{B}, k)$, the numerical computation time is rather bad. We have reason to believe that this is caused by the slow convergence of the eigensolver in case of a clustering of the eigenvalues. Therefore, for practical purposes, we rather refer to the following Algorithm 4 to realize the $\mathcal{H}$-matrix product.

Algorithm 4 $\mathcal{H}$-matrix multiplication

---

function $\mathbf{H}_3 = \text{Mult}\mathcal{H}(\mathbf{H}_1, \mathbf{H}_2)$
   if $\mathbf{H}_3 \in \mathcal{H}(\mathcal{B}', k)$ **then**
      for $\ell, \ell' = 1, \ldots, p$ **do**
         for $i = 1, \ldots, p$ **do**
            $(\mathbf{H}_3)_{\ell,\ell'}$ += $\text{Mult}\mathcal{H}((\mathbf{H}_1)_{\ell,i}, (\mathbf{H}_2)_{i,\ell'})$
         end for
      end for
   else
      $\mathbf{H}_3$ += $\mathbf{H}_1\mathbf{H}_2$
   end if
end function

Note that the operation '+=' is overloaded here and depends one the type of the operands as introduced in Sects. 5.2 and 5.3.

Remark 7 The implementation of the $\mathcal{H}$-matrix multiplication in Algorithm 4 implicitly employs the *fast truncation* as proposed in [14].

5.5 $\mathcal{H}$-matrix Addition

The realization of the $\mathcal{H}$-matrix addition is based on the assumption that all operands are $\mathcal{H}$-matrices with respect to a common block-cluster tree, therefore let $\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3 \in \mathcal{H}(\mathcal{B}, k)$. Then, with respect to the truncation operators defined in Sect. 5.2, the $\mathcal{H}$-matrix addition can be implemented along the lines of Algorithm 5.

Algorithm 5 $\mathcal{H}$-matrix addition

$\quad$ **function** $\mathbf{H}_3 = \text{ADD}\mathcal{H}(\mathbf{H}_1, \mathbf{H}_2)$
$\qquad$ **if** $\mathbf{H}_3 \in \mathcal{H}(\mathcal{B}', k)$ **then**
$\qquad\quad$ **for** $\ell, \ell' = 1, \ldots, p$ **do**
$\qquad\qquad (\mathbf{H}_3)_{\ell,\ell'} = \text{ADD}\mathcal{H}((\mathbf{H}_1)_{\ell,\ell'}, (\mathbf{H}_2)_{\ell,\ell'})$
$\qquad\quad$ **end for**
$\qquad$ **else**
$\qquad\quad \mathbf{H}_3 = \mathfrak{T}(\mathbf{H}_1 + \mathbf{H}_2)$
$\qquad$ **end if**
$\quad$ **end function**

5.6 $\mathcal{H}$-matrix Inversion

The inversion of an $\mathcal{H}$-matrix can be done approximately by a recursive block Gaussian elimination. Algorithm 6, cf. [14], computes the approximate inverse $\hat{\mathbf{H}}^{-1} \in \mathcal{H}(\mathcal{B}, k)$ of $\mathbf{H} \in \mathcal{H}(\mathcal{B}, k)$ where the original matrix $\mathbf{H}$ is overwritten during the computation.

Algorithm 6 $\mathcal{H}$-matrix inversion

$\quad$ **function** $\hat{\mathbf{H}}^{-1} = \text{INVERT}\mathcal{H}(\mathbf{H})$
$\qquad$ **if** $\mathbf{H} \in \mathcal{H}(\mathcal{B}', k)$ **then**
$\qquad\quad$ **for** $\ell = 1, \ldots, p$ **do** $\qquad\qquad\qquad\qquad\qquad\qquad$ ▷ Eliminate lower blocks
$\qquad\qquad (\hat{\mathbf{H}}^{-1})_{\ell,\ell} = \text{INVERT}\mathcal{H}(\mathbf{H}_{\ell,\ell})$
$\qquad\qquad \{(\hat{\mathbf{H}}^{-1})_{\ell,\ell'}\}_{\ell'=1}^{\ell-1} = \{(\hat{\mathbf{H}}^{-1})_{\ell,\ell} * (\hat{\mathbf{H}}^{-1})_{\ell,\ell'}\}_{\ell'=1}^{\ell-1}$
$\qquad\qquad \{\mathbf{H}_{\ell,\ell'}\}_{\ell'=\ell+1}^{p} = \{(\hat{\mathbf{H}}^{-1})_{\ell,\ell} * \mathbf{H}_{\ell,\ell'}\}_{\ell'=\ell+1}^{p}$
$\qquad\qquad$ **for** $\ell' = \ell + 1, \ldots, p$ **do**
$\qquad\qquad\quad \{(\hat{\mathbf{H}}^{-1})_{\ell',i}\}_{i=1}^{\ell} \mathrel{+}= \{-\mathbf{H}_{\ell',\ell} * (\hat{\mathbf{H}}^{-1})_{\ell,i}\}_{i=1}^{\ell}$
$\qquad\qquad\quad \{\mathbf{H}_{\ell',i}\}_{i=\ell+1}^{p} \mathrel{+}= \{-\mathbf{H}_{\ell',\ell} * \mathbf{H}_{\ell,i}\}_{i=\ell+1}^{p}$
$\qquad\qquad$ **end for**
$\qquad\quad$ **end for**
$\qquad\quad$ **for** $\ell = p, \ldots, 1$ **do** $\qquad\qquad\qquad\qquad\qquad\qquad$ ▷ Eliminate upper blocks
$\qquad\qquad$ **for** $\ell' = \ell - 1, \ldots, p$ **do**
$\qquad\qquad\quad \{(\hat{\mathbf{H}}^{-1})_{\ell',i)}\}_{i=1}^{p} \mathrel{+}= \{-\mathbf{H}_{\ell',\ell} * (\hat{\mathbf{H}}^{-1})_{\ell,i}\}_{i=1}^{p}$
$\qquad\qquad$ **end for**
$\qquad\quad$ **end for**
$\qquad$ **else**
$\qquad\quad \hat{\mathbf{H}}^{-1} = \mathbf{H}^{-1}$
$\qquad$ **end if**
$\quad$ **end function**

For the case of symmetric positive definite matrices one can also employ a moderate modification of the block Gaussian elimination to obtain an inverse Cholesky decomposition $\hat{\mathbf{H}}^{-1} = \mathbf{L} * \mathbf{L}^{\mathsf{T}}$.

6 Iterative Solution

For two $\mathcal{H}$-matrices $\mathbf{H}_1, \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$, we have defined the addition $\mathbf{H}_1 + \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$ and the product $\mathbf{H}_1 * \mathbf{H}_2 \in \mathcal{H}(\mathcal{B}, k)$ as well as the $\mathcal{H}$-matrix inversion in the previous section.

Now, we will explain how these operations can be used to implement an iterative solver for the linear system of equations (15), given that all matrices are represented in the $\mathcal{H}$-matrix format.

With an approximate inverse of the $\mathcal{H}$-matrix at hand, we may consider an iterative solver based on the *iterative refinement* method, cf. [9,27,36].

Let $\hat{\mathbf{S}}^{-1} \in \mathcal{H}(\mathcal{B}, k)$ now be an approximate inverse to $\mathbf{S}$. Starting with the initial guess $\mathbf{C}_\rho^{(0)} = \hat{\mathbf{S}}^{-1} \mathbf{C}_f \hat{\mathbf{S}}^{-\mathsf{T}}$, the solution to (15) can then be approximated via the iteration

$$\boldsymbol{\Theta}^{(i)} = \mathbf{C}_f - \mathbf{S} \mathbf{C}_\rho^{(i)} \mathbf{S}^{\mathsf{T}}, \quad \mathbf{C}_\rho^{(i+1)} = \mathbf{C}_\rho^{(i)} + \hat{\mathbf{S}}^{-1} \boldsymbol{\Theta}^{(i)} \hat{\mathbf{S}}^{-\mathsf{T}}, \quad i = 0, 1, \dots$$

This method is known as iterative refinement and has originally been introduced in [36] for the improvement of solutions to linear systems of equations based on the LU-factorization, see e.g. [9].

Although we use here an approximate inverse based on the block Gaussian elimination rather than an LU-factorization, the idea stays the same: The residual $\boldsymbol{\Theta}^{(i)}$ is computed with a higher precision than the correction $\hat{\mathbf{S}}^{-1} \boldsymbol{\Theta}^{(i)} \hat{\mathbf{S}}^{-\mathsf{T}}$. This yields an improved approximation to the solution in each step. Note that this algorithm also coincides with an undamped (preconditioned) Richardson iteration, see e.g. [32].

Finally, we want to remark that we have also considered a *Newton-Schulz iteration*, cf. [16, 34], in order to compute an approximate inverse for a given $\mathcal{H}$-matrix. In this context, it has turned out that the inversion with aid of the block Gaussian elimination is more reliable and much faster.

7 Numerical Results

All computations of the following examples have been carried out on a computing server with 2 Intel(R) Xeon(R) E5-2670 CPUs with a clock rate of 2.60 GHz and a main memory of 256 GB. Each of the CPUs provides 8 physical cores, thus with Hyper-Threading enabled, we may access 32 cores in total. In order to control the ranks of the far-field, we have previously introduced the bound $k_{\max}$. Therefore, to achieve a preferably data-sparse representation of the $\mathcal{H}$-matrices evolving in the computations, we set the threshold for the block size in the near-field to the smallest power of four greater or equal to $k_{\max}$.

7.1 Tests for the $\mathcal{H}$-matrix Arithmetic

After having adapted the $\mathcal{H}$-matrix arithmetic to the regime of parametric surfaces, see e.g. [18], as explained in Sect. 5, the question arises how the presented algorithms perform in practical applications. To this end, we provide in this subsection sample computations for the $\mathcal{H}$-matrix arithmetic on two different geometries given as parametric surfaces. On the one hand, we consider the unit sphere $\mathbb{S}^2$ parameterized by six patches and, on the other hand, a pierced cube with circular holes on each face, which we will refer to as "toy" geometry, represented by 48 patches. A visualization of both geometries can be found in Fig. 1.

On each of the geometries, we assembled two $\mathcal{H}$-matrices $\mathbf{S}, \mathbf{T} \in \mathcal{H}(\mathcal{B}, k_{\max})$. Namely, $\mathbf{S}$ is the discrete and compressed single-layer operator for the Laplace equation, cf. (5), and $\mathbf{T}$ is the discrete and compressed integral operator with kernel

$$\Phi(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|_2).$$

This kernel is also known as exponential kernel and corresponds to the Matérn kernel with $\nu = 1/2$ and $\ell = 1$. In order to obtain meaningful results for the computational times, we have run the respective computations only on a single core with the rank limited to $k_{\max} = 16$ on the sphere and to $k_{\max} = 25$ on the "toy" geometry. In addition to the presented operators $+$ and $*$ for non-symmetric matrices we also employ their symmetric versions $\hat{+}$ and $\hat{*}$. The latter consider only the lower triangular part of the respective operands and compute only

Table 1 Computational times for each particular $\mathcal{H}$-matrix operation on the unit sphere (*top*) and the "toy" geometry (*bottom*)

Unit sphere

J	N_J	Time $\mathbf{S}\hat{+}\mathbf{T}$ (s)	Time $\mathbf{S}+\mathbf{T}$ (s)	Time $\mathbf{S}\hat{*}\mathbf{T}$ (s)	Time $\mathbf{S}*\mathbf{T}$ (s)	Time $\mathbf{L}$ (s)	Time $\hat{\mathbf{S}}^{-1}$ (s)
1	24	<1	<1	<1	<1	<1	<1
2	96	<1	<1	<1	<1	<1	<1
3	384	<1	<1	<1	1.26403	1.10483	1.78293
4	1,536	<1	<1	11.9248	24.0851	16.6238	27.7649
5	6,144	<1	<1	93.2262	193.603	119.235	210.002
6	24,576	3.03885	6.06252	521.222	1,100.32	652.681	1,187.8
7	98,304	19.3425	38.5616	2,771.35	5,940.23	3,397.62	6,150.32
8	393,216	114.049	229.159	14,235.1	30,512.8	17,690.1	31,309.4

"Toy" geometry

J	N_J	Time $\mathbf{S}\hat{+}\mathbf{T}$ (s)	Time $\mathbf{S}+\mathbf{T}$ (s)	Time $\mathbf{S}\hat{*}\mathbf{T}$ (s)	Time $\mathbf{S}*\mathbf{T}$ (s)	Time $\mathbf{L}$ (s)	Time $\hat{\mathbf{S}}^{-1}$ (s)
1	192	<1	<1	<1	<1	<1	<1
2	768	<1	<1	<1	<1	<1	<1
3	3,072	<1	<1	35.7135	72.2416	63.8146	104.71
4	12,288	<1	<1	295.689	609.632	557.688	927.998
5	49,152	3.64886	7.27567	1,601.64	3,342.97	2,648.21	4,642.44
6	196,608	26.0626	51.8281	8,024.69	16,929.3	12,612.5	22,213.7
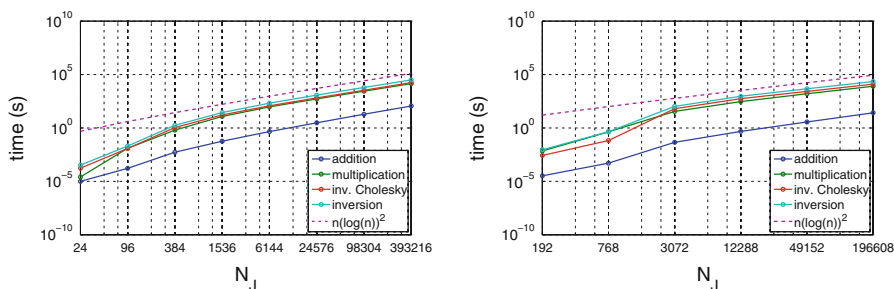
Fig. 4 Asymptotic behavior of the computation times on the unit sphere (*left*) and on the "toy" geometry (*right*)

the lower triangular part of the result. The computational times consumed for the operations $\mathbf{S}+\mathbf{T}$, $\mathbf{S}\hat{+}\mathbf{T}$, $\mathbf{S}*\mathbf{T}$, $\mathbf{S}\hat{*}\mathbf{T}$ and the computation of the inverse Cholesky decomposition $\mathbf{L}$ of $\mathbf{S}$ are tabulated in the upper part of Table 1 in case of the unit sphere and in the lower part of Table 1 in case of the "toy" geometry. We also show the computational times to compute the approximate inverse $\mathbf{L}*\mathbf{L}^\mathsf{T} = \hat{\mathbf{S}}^{-1}$ which consist of the computation time for $\mathbf{L}$ and the multiplication of two triangular matrices. Additionally, Fig. 4 shows the asymptotic behavior of the computational times. For sake of simplicity, we only show the times for the symmetric addition and multiplication. It seems that we obtain the rate of $N_J(\log N_J)^2$, which is in concordance with the complexity estimates proven in [14] and all constants which appear in these estimates set to 1.
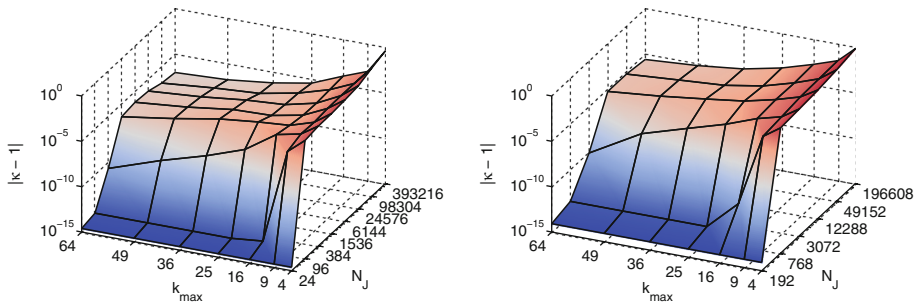
Fig. 5 Error for the condition of the product $\mathbf{S\hat{S}}^{-1}$ on the unit sphere (*left*) and on the "toy" geometry (*right*) for different ranks and levels
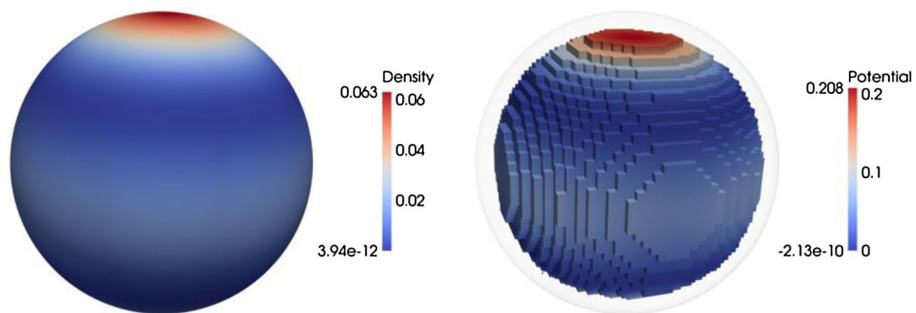

Fig. 6 Visualization of the trace of the density (*left*) and the trace of the potential (*right*) on the unit sphere for $N_J = 393{,}216$

We consider the condition number κ of $\mathbf{S\hat{S}}^{-1}$ which we compute with ARPACK, cf. [25], as measure for the quality of the approximate inverse $\mathbf{S}$. Especially, we do actually not need to compute the product $\mathbf{S\hat{S}}^{-1}$ explicitly, but rather provide the application of $\mathbf{S\hat{S}}^{-1}$ to a vector. Figure 5 visualizes the error related to the condition number, i.e. $|\kappa - 1|$, in dependence on the maximal rank $k_{\max}$ and the particular level for the unit sphere and the "toy" geometry. The visualization indicates that, even on the higher levels for an appropriately chosen rank $k_{\max}$, the approximation of the inverse yields an error which is about 10^{-5}. Qualitatively, we observe on each particular level the expected decay of the error when the rank increases.

7.2 Tests for the Iterative Solver

Now, with the $\mathcal{H}$-matrix arithmetic at hand, we want to show how it actually performs in solving the linear system of equations (15). At first, we provide two simple numerical examples in order to show that our solver indeed works and provides convergence. Afterwards, we consider a realistic example.

Wherever possible, we employ symmetric versions of the algorithms, especially we invert symmetric positive definite $\mathcal{H}$-matrices using the inverse block Cholesky decomposition, which is faster than the block Gaussian elimination.

On the unit sphere, we consider a tensorized spherical harmonic as right hand side, i.e. $\mathrm{Cor}_f = Y_2^0 \otimes Y_2^0$ with $Y_2^0(\mathbf{x}) = \sqrt{5/(16\pi)}(3x_3^2 - 1)$. This right hand side is obviously a single dyad and thus of rank 1. Especially, since Y_2^0 is a harmonic function, we can compute the related approximation error. To this end, we compute the error in the trace of the
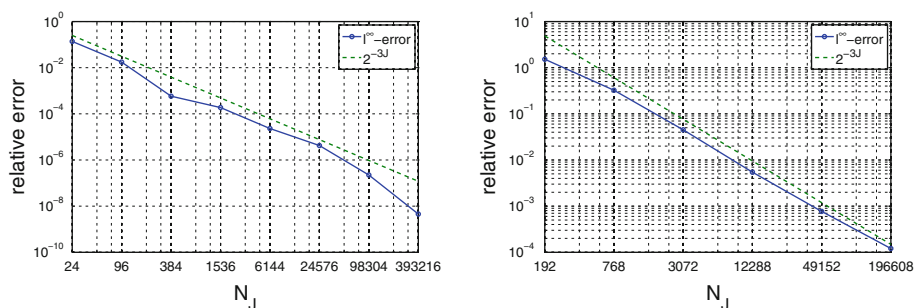
Fig. 7 Errors for the potentials' traces on the sphere (*left*) and on the "toy" geometry (*right*)
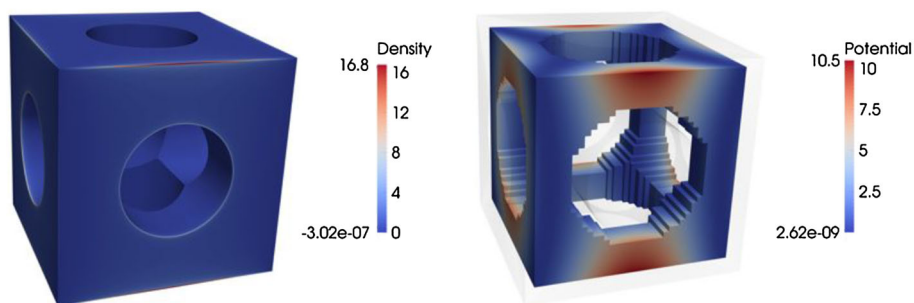

Fig. 8 Visualization of the trace of the density (*left*) and the trace of the potential (*right*) on the "toy" geometry for $N_J = 196,608$

tensor product single-layer potential, i.e. $(\mathcal{S} \otimes \mathcal{S})|_{\mathbf{x}=\mathbf{y}}$. As stopping criterion for the iterative solution, we require the relative error of the residuals' Frobenius norm to be smaller than $\varepsilon = 10^{-5}$. For the levels $J = 1, \ldots, 6$, we have chosen $k_{\max} = 16$. For the levels $J = 7, 8$, we had to increase $k_{\max}$ to $k_{\max} = 25$ in order to achieve convergence. The error is measured in 1,793 evaluation points which are uniformly distributed within the unit sphere. The left plot in Fig. 7 indicates that we obtain indeed the expected cubic convergence. Furthermore, we have tabulated the related errors in the upper part of Table 2. A visualization of the trace of the density and the trace of the potential on the unit sphere for $N_J = 393,216$ is found in Fig. 6.

Likewise, we consider a tensor product of a harmonic polynomial, as right hand side on the "toy" geometry, namely $\mathrm{Cor}_f = p \otimes p$ with $p(\mathbf{x}) = 4x_1^2 - 3x_2^2 - x_3^2$. The computations are performed for the levels $J = 1, \ldots, 6$ where we have always chosen $k_{\max} = 25$. For the "toy" geometry, the error is measured in 1,208 evaluation points which are uniformly distributed within the geometry. The plot on the right of Fig. 7 shows that we obtain cubic convergence for this example, too. The related errors are given in the lower part of Table 2. A visualization of the trace of the density and the trace of the potential on the "toy" geometry for $N_J = 196,608$ is found in Fig. 8.

Notice that, for both examples, we had to perform at most two steps of the iterative refinement to achieve the error bound of $\varepsilon = 10^{-5}$.

Table 2 Errors for the potentials' traces on the sphere (*top*) and on the "toy" geometry (*bottom*)

Unit sphere

J	1	2	3	4	5	6	7	8
N_J	24	96	384	1,536	6,144	24,576	98,304	393,216
l^∞-error	0.137424	0.0174729	0.000581418	0.000186857	$2.31215 \cdot 10^{-5}$	$4.22958 \cdot 10^{-6}$	$2.23217 \cdot 10^{-7}$	$4.53721 \cdot 10^{-9}$

"Toy" geometry

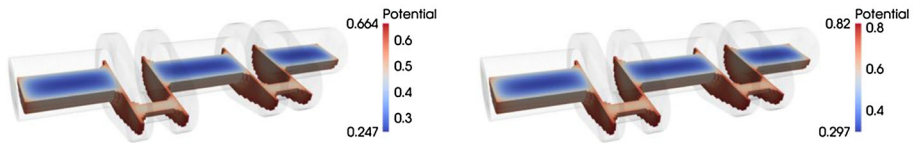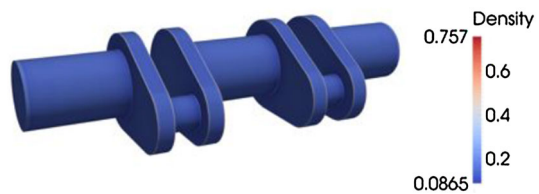| J	1	2	3	4	5	6
N_J	192	768	3,072	12,288	49,152	196,608
l^∞-error	1.52454	0.324403	0.0450429	0.00544676	0.000769455	0.000120884

Fig. 9 Cross sections of the potentials' traces on the crankshaft geometry for the Matérn kernel with $\nu = 1/2$ (*left*) and $\nu = 3/2$ (*right*)

Fig. 10 Trace of the density on the crankshaft geometry for the Matérn kernel with $\nu = 3/2$ and $N_J = 145,408$

7.3 Stochastic Application

Having shown that the numerical method works, we shall now apply the method to an actual stochastic problem. We consider the Laplace equation on a crankshaft as geometry which is parameterized by 142 patches, see also Fig. 1. For the discretization, we choose $J = 5$ which corresponds to $N_J = 145,408$ degrees of freedom. The maximum rank is set to $k_{\max} = 81$.

In the subsequent computations, we apply the Matérn kernels with $\nu = 1/2$, i.e. $\mathrm{Cor}_f(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|_2)$, and $\nu = 3/2$, i.e. $\mathrm{Cor}_f(\mathbf{x}, \mathbf{y}) = (1 + \sqrt{3}\|\mathbf{x} - \mathbf{y}\|_2) \exp(-\sqrt{3}\|\mathbf{x} - \mathbf{y}\|_2)$, as correlation kernels for the right hand side. These kernels are not of finite rank anymore, but provide asymptotical smoothness and are thus compressible by means of $\mathcal{H}$-matrices. In particular, according to Sect. 4, the related two-point correlations Cor_ρ provide regularity in terms of $H^{s,s}_{\mathrm{mix}}(\Gamma \times \Gamma)$ with $s = 0 - \varepsilon$ if $\nu = 1/2$ and with $s = 1 - \varepsilon$ if $\nu = 3/2$, respectively. As a consequence, a sparse tensor product approximation in case of $\nu = 1/2$ would suffer from the lack of regularity while a sparse tensor product approximation in case of $\nu = 3/2$ would nearly converge with the optimal rate.

For sake of lower computational times, we employ in this example a parallel version of the $\mathcal{H}$-matrix multiplication and inversion on at most 32 cores. Again, we employ symmetric versions of the algorithm wherever it is possible. This results in a computational time of about 8000 s for the approximate inversion of the discretized single-layer operator $\mathbf{S}$. The related error in the condition number of $\mathbf{S}\hat{\mathbf{S}}^{-1}$ is $4.95 \cdot 10^{-6}$. Applying this inverse yields an error of $6.13 \cdot 10^{-6}$ in one iteration in the iterative refinement in case of the Matérn kernel with $\nu = 1/2$ and an error of $5.74 \cdot 10^{-6}$ in case of the Matérn kernel with $\nu = 3/2$. The computation times were about 24,000 and 23,000 s, respectively. A visualization of the potentials' traces is found in Fig. 9. Finally, the density's trace in case of the Matérn kernel with $\nu = 3/2$ is found in Fig. 10.

8 Conclusion

In this article, we have proposed a new numerical framework to compute the solution's two-point correlation of a linear partial differential equation with Dirichlet boundary data. Instead of considering a sparse tensor product approximation, which would suffer from

the lack of smoothness, we employ a full tensor product discretization that is shown to be feasible by means of $\mathcal{H}$-matrices. Our theoretical considerations imply that this approach provides, in the case of rough correlations, twice of the rate of convergence compared to the sparse tensor product approach. Nonetheless, the $\mathcal{H}$-matrix approximation is still scaling log-linear in the number of boundary elements. Using the $\mathcal{H}$-matrix framework for parametric surfaces, we end up with special structured $\mathcal{H}$-matrices. The combination of this structure with a sparse eigensolver results in a very efficient $\mathcal{H}$-matrix arithmetic. The numerical examples validate the predicted cost complexity for the $\mathcal{H}$-matrix arithmetic. Furthermore, the numerical examples confirm that the presented iterative solver is well suited to numerically solve the correlation equation. Finally, results are given for roughly correlated stochastic problems with kernels from the Matérn class on a nontrivial geometry.

References

1. Babuška, I., Nobile, F., Tempone, R.: A stochastic collocation method for elliptic partial differential equations with random input data. SIAM J. Numer. Anal. **45**(3), 1005–1034 (2007)
2. Babuška, I., Tempone, R., Zouraris, G.: Galerkin finite element approximations of stochastic elliptic partial differential equations. SIAM J. Numer. Anal. **42**(2), 800–825 (2004)
3. Costabel, M.: Boundary integral operators on Lipschitz domains. Elementary results. SIAM J. Math. Anal. **19**(3), 613–626 (1988)
4. Deb, M., Babuška, I., Oden, J.: Solution of stochastic partial differential equations using Galerkin finite element techniques. Comput. Methods Appl. Mech. Eng. **190**(48), 6359–6372 (2001)
5. Frauenfelder, P., Schwab, C., Todor, R.: Finite elements for elliptic problems with stochastic coefficients. Comput. Methods Appl. Mech. Eng. **194**(2–5), 205–228 (2005)
6. Ghanem, R., Spanos, P.: Stochastic Finite Elements. A Spectral Approach. Springer, New York (1991)
7. Gilbarg, D., Trudinger, N.: Elliptic Partial Differential Equations of Second Order. Springer, Berlin-Heidelberg (1977)
8. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. J. Soc. Indust. Appl. Math. Ser. B Numer. Anal. **2**(2), 205–224 (1965)
9. Golub, G., Van Loan, C.: Matrix Computations, 4th edn. Johns Hopkins University Press, Baltimore (2012)
10. Griebel, M., Harbrecht, H.: On the construction of sparse tensor product spaces. Math. Comput. **82**(282), 975–994 (2013)
11. Griebel, M., Harbrecht, H.: On the convergence of the combination technique. Preprint No. 2013–07, Mathematisches Institut, Universität Basel (2013)
12. Griebel, M., Knapek, S.: Optimized tensor-product approximation spaces. Constr. Approx. **16**(4), 525–540 (2000)
13. Hackbusch, W.: A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: introduction to $\mathcal{H}$-matrices. Computing **62**(2), 89–108 (1999)
14. Hackbusch, W.: Hierarchische Matrizen: Algorithmen und Analysis. Springer, Berlin-Heidelberg (2009)
15. Hackbusch, W., Khoromskij, B.: A sparse $\mathcal{H}$-matrix arithmetic. General complexity estimates. J. Comput. Appl. Math. **125**(1–2), 479–501 (2000)
16. Hackbusch, W., Khoromskij, B., Tyrtyshnikov, E.: Approximate iterations for structured matrices. Numer. Math. **109**(3), 365–383 (2008)
17. Harbrecht, H.: A finite element method for elliptic problems with stochastic input data. Appl. Numer. Math. **60**(3), 227–244 (2010)
18. Harbrecht, H., Peters, M.: Comparison of fast boundary element methods on parametric surfaces. Comput. Methods Appl. Mech. Eng. **261–262**, 39–55 (2013)
19. Harbrecht, H., Peters, M., Siebenmorgen, M.: Combination technique based k-th moment analysis of elliptic problems with random diffusion. J. Comput. Phys. **252**, 128–141 (2013)
20. Harbrecht, H., Schneider, R., Schwab, C.: Multilevel frames for sparse tensor product spaces. Numer. Math. **110**(2), 199–220 (2008)
21. Harbrecht, H., Schneider, R., Schwab, C.: Sparse second moment analysis for elliptic problems in stochastic domains. Numer. Math. **109**(3), 385–414 (2008)
22. Hughes, T., Cottrell, J., Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput. Methods Appl. Mech. Eng. **194**(39–41), 4135–4195 (2005)

23. Khoromskij, B.N., Schwab, C.: Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs. SIAM J. Sci. Comput. **33**(1), 364–385 (2011)
24. Lanczos, C.: Linear Differential Operators. D. Van Nostrand Co., Ltd., London (1961)
25. Lehoucq, R., Sorensen, D., Yang, C.: Arpack User's Guide: Solution of Large-Scale Eigenvalue Problems With Implicityly Restorted Arnoldi Methods (Software, Environments, Tools). SIAM, Philadelphia (1998)
26. Matthies, H., Keese, A.: Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations. Comput. Methods Appl. Mech. Eng. **194**(12–16), 1295–1331 (2005)
27. Moler, C.: Iterative refinement in floating point. J. ACM **14**(2), 316–321 (1967)
28. Nobile, F., Tempone, R., Webster, C.: An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. SIAM J. Numer. Anal. **46**(5), 2411–2442 (2008)
29. von Petersdorff, T., Schwab, C.: Sparse finite element methods for operator equations with stochastic data. Appl. Math. **51**(2), 145–180 (2006)
30. Protter, P.: Stochastic Integration and Differential Equations. A New Approach, 3rd edn. Springer, Berlin (1995)
31. Rasmussen, C., Williams, C.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, Cambridge (2005)
32. Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
33. Sauter, S., Schwab, C.: Boundary Element Methods. Springer, Berlin-Heidelberg (2010)
34. Schulz, G.: Iterative Berechung der reziproken Matrix. ZAMM Z. Angew. Math. Mech. **13**(1), 57–59 (1933)
35. Schwab, C., Todor, R.: Sparse finite elements for elliptic problems with stochastic loading. Numer. Math. **95**(4), 707–734 (2003)
36. Wilkinson, J.: Rounding Errors in Algebraic Processes. Prentice-Hall, Englewood Cliffs (1963)
37. Xiu, D., Tartakovsky, D.: Numerical methods for differential equations in random domains. SIAM J. Sci. Comput. **28**(3), 1167–1185 (2006)