Institut für Numerische Simulation
Rheinische Friedrich-Wilhelms-Universität Bonn

Friedrich-Hirzebruch-Allee 7 • 53115 Bonn • Germany
phone +49 228 73-69828 • fax +49 228 73-69847
www.ins.uni-bonn.de

M. Griebel, H. Harbrecht, M. Multerer

Kernel interpolation on sparse grids

INS Preprint No. 2504

May 2025

KERNEL INTERPOLATION ON SPARSE GRIDS

MICHAEL GRIEBEL, HELMUT HARBRECHT, AND MICHAEL MULTERER

Abstract. We consider scattered data approximation on product regions of equal and different dimensionality. On each of these regions, we assume quasi-uniform but unstructured data sites and construct optimal sparse grids for scattered data interpolation on the product region. For this, we derive new improved error estimates for the respective kernel interpolation error by invoking duality arguments. An efficient algorithm to solve the underlying linear system of equations is proposed. The algorithm is based on the sparse grid combination technique, where a sparse direct solver is used for the elementary anisotropic tensor product kernel interpolation problems. The application of the sparse direct solver is facilitated by applying a samplet matrix compression to each univariate kernel matrix, resulting in an essentially sparse representation of the latter. In this way, we obtain a method that is able to deal with large problems up to billions of interpolation points, especially in case of reproducing kernels of nonlocal nature. Numerical results are presented to qualify and quantify the approach.

1. Introduction

Scattered data approximation using kernels is popular in many areas, ranging from approximation theory to statistics. The approach facilitates the estimation of missing values in a dataset or to make predictions for new data sites based on the available data. Scattered data approximation is particularly applied in imaging processing, surface reconstruction and machine learning, see for example [10, 32, 35, 36] and the references therein. However, the naive computation of the kernel approximate is known to suffer from the so-called *curse of dimensionality* when the data dimension increases.

Various concepts exist to overcome the curse of dimensionality to a certain extent. A prominent approach is offered by *sparse grids* or more general *sparse tensor product spaces*, where the dimensions only mildly enter in the cost estimates through a dimension-dependent power of a logarithmic factor, see [3, 34, 37] for example. In this article, we aim at the construction and implementation of suitable sparse grids for the approximation of tensor product kernels. Interpreting kernel approximation in the context of Gaussian process learning, see [30], the approach under consideration amounts to a multi-fidelity fusion model, see e.g. [11, 29], where the hierarchy of surrogate models is given by kernel approximates on a hierarchy of subspaces. A fundamental contribution to sparse grids for kernel approximation has recently been provided by [25, 26]. While the sparse grid construction therein relies on a multilevel approach invoking level dependent correlation lengths of the kernel function under consideration, we use here a kernel function of fixed correlation length to construct the sparse grid interpolant. Especially, we discuss the optimality of the underlying sparse tensor product spaces and provide improved error estimates based on results in [31, 33].

The starting point for our construction is a tensor product Hilbert space

$$\boldsymbol{\mathcal{H}} = \bigotimes_{i=1}^{m} \mathcal{H}^{(i)},$$

2020 *Mathematics Subject Classification.* Primary 41A46; Secondary 41A63, 46E35.

Key words and phrases. Reproducing kernel Hilbert space, sparse grid, fast kernel approximation.

formed by a finite collection of reproducing kernel Hilbert spaces $\mathcal{H}^{(i)}$ with reproducing kernels κ_i, $i = 1, \ldots, m$, defined on a collection of Lipschitz-smooth regions $\Omega_i \subset \mathbb{R}^{d_i}$ of general dimensions $d_i \in \mathbb{N}$. Associated to the tensor product reproducing kernel Hilbert space $\boldsymbol{\mathcal{H}}$, we consider the product kernel

$$\boldsymbol{\kappa}(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{m} \kappa_i(x_i, y_i).$$

The kernel is the reproducing kernel of the space $\boldsymbol{\mathcal{H}}$ and renders it itself a reproducing kernel Hilbert space defined on the product region $\boldsymbol{\Omega} = \bigtimes_{i=1}^{m} \Omega_i$.

For this setup, we construct an optimized sparse tensor product space to compute the kernel interpolant with respect to the underlying sparse grid. Employing results from [15, 16, 33], we are able to derive improved error estimates for the sparse grid approximation error and related complexity bounds. To this end, we assume for each of the regions Ω_i sets of quasi-uniform data sites. We propose a simple algorithm to coarsen these sets in order to construct the necessary multilevel hierarchy of approximation spaces for the sparse grid. The implementation of the sparse grid and the computation of the sparse grid interpolant is then based on the *sparse grid combination technique* as introduced in [17, 34]. This approach is known to successively compose the respective solution from the solutions to certain anisotropic standard tensor product interpolation problems, see [24, 25, 26]. We provide the details on the implementation of the sparse grid combination technique as well as the storage and solution of the tensor product subproblems. To solve the latter, we suggest the use of a direct solver that combines *samplet matrix compression* with a *sparse direct solver* as proposed in [20, 21]. This way, the approach becomes computationally feasible, especially in case of nonlocal reproducing kernels. We present extensive numerical studies to qualify and quantify the approach.

The rest of the article is structured as follows: In Section 2, we introduce reproducing kernel Hilbert spaces and their basic theory. Then, in Section 3, we define generalized sparse grids and discuss their optimality concerning their complexity. The numerical implementation and related algorithms are described in Section 4. In Section 5, we perform numerical experiments which validate the present theory. Finally, in Section 6, we draw some conclusions.

Throughout this article, to avoid the repeated use of unspecified generic constants, we write $A \lesssim B$ if A is bounded by a uniform constant times B, where the constant does not depend on any parameters which A and B might depend on. Similarly, we write $A \gtrsim B$ if and only if $B \lesssim A$. Finally, if $A \lesssim B$ and $B \lesssim A$, we write $A \sim B$.

2. Preliminaries

2.1. **Reproducing kernel Hilbert spaces.**

Let $\Omega \subset \mathbb{R}^d$, $d \in \mathbb{N}$, be a given Lipschitz-smooth region that can be bounded or unbounded. We start with the following definition:

Definition 2.1. A *reproducing kernel* for a Hilbert space $\mathcal{H}$ of functions $u \colon \Omega \to \mathbb{R}$ with inner product $(\cdot, \cdot)_{\mathcal{H}}$ is a function $\kappa \colon \Omega \times \Omega \to \mathbb{R}$ such that

(1) $\kappa(\cdot, y) \in \mathcal{H}$ for all $y \in \Omega$,
(2) $u(y) = \big(u, \kappa(\cdot, y)\big)_{\mathcal{H}}$ for all $u \in \mathcal{H}$ and all $y \in \Omega$.

A Hilbert space $\mathcal{H}$ with reproducing kernel $\kappa \colon \Omega \times \Omega \to \mathbb{R}$ is called *reproducing kernel Hilbert space* (RKHS).

A continuous kernel $\kappa : \Omega \times \Omega \to \mathbb{R}$ is called *positive semidefinite* on $\Omega \subset \mathbb{R}^d$ if

$$\sum_{i,j=1}^{N} \alpha_i \alpha_j \kappa(x_i, x_j) \geq 0 \tag{2.1}$$

holds for all all mutually distinct points $x_1, \ldots, x_N \in \Omega$ and all $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$, for any $N \in \mathbb{N}$. The kernel is even *positive definite* if the inequality in (2.1) is strict whenever at least one α_i is different from 0. A typical example of such kernels are the *Matérn kernels*, which are the reproducing kernels of the Sobolev spaces $H^s(\mathbb{R}^d)$ for $s > d/2$, see [27] for example.

Given a set $X = \{x_1, \ldots, x_N\}$ of N mutually distinct data sites, we introduce the *kernel translates* $\phi_j := \kappa(\cdot, x_j)$ for $j = 1, \ldots, N$. If the kernel κ is positive definite, these kernel translates span the N-dimensional subspace

$$\mathcal{H}_X := \operatorname{span}\{\phi_1, \ldots, \phi_N\} \subset \mathcal{H}.$$

The best approximation $f_X \in \mathcal{H}_X$ of a function $f \in \mathcal{H}$ with respect to $\mathcal{H}$ amounts to its $\mathcal{H}$-orthogonal projection onto $\mathcal{H}_X$. The latter can be obtained as the solution of the variational formulation

$$\text{find } f_X \in \mathcal{H}_X, \text{ such that } (f_X, v)_{\mathcal{H}} = (f, v)_{\mathcal{H}} \text{ for all } v \in \mathcal{H}_X.$$

In view of the reproducing property, i.e., the second property from Definition 2.1, the ansatz $f_X = \sum_{i=1}^N \alpha_i \phi_i$ leads to the linear system of equations

$$\boldsymbol{K}\boldsymbol{\alpha} = \boldsymbol{f}$$

with the kernel matrix

$$\boldsymbol{K} = \begin{bmatrix} \kappa(x_1, x_1) & \cdots & \kappa(x_1, x_N) \\ \vdots & \ddots & \vdots \\ \kappa(x_N, x_1) & \cdots & \kappa(x_N, x_N) \end{bmatrix}$$

and the right-hand side $\boldsymbol{f} = [f(x_1), \ldots, f(x_N)]^T$.

In particular, we observe that the resulting system of equations coincides with the one for the generalized Vandermonde matrix for the interpolation at the data sites in X, i.e.

$$u(x_j) = \sum_{i=1}^N \alpha_i \phi(x_j) \overset{!}{=} f(x_j) \quad \text{for } j = 1, \ldots, N.$$

This means that, within the RKHS framework, the best approximation $u \in \mathcal{H}_X$ of a function $f \in \mathcal{H}$ is given by the interpolant for the data sites X. This is also referred to as *kernel interpolation.* Since kernel interpolation works on arbitrarily unstructured sets of data sites, it is often used to approximate scattered data. Such scattered data can be found in computer graphics, but also in machine learning of high-dimensional data sets, see e.g., [10, 36].

2.2. **Error estimates.**

Having fixed the kernel of interest, we consider the problem of function approximation. We are interested in recovering an unknown function $f \in \mathcal{H}$, given only a finite data set

$$\{(x_1, f_1), \ldots, (x_N, f_N)\} \subset \Omega \times \mathbb{R}.$$

We collect the data sites in the set $X := \{x_1, \ldots, x_N\} \subset \Omega$. Associated to this set, we define two characteristic quantities, namely the *fill distance*

$$h_{X,\Omega} := \sup_{x \in \Omega} \min_{x_i \in X} \|x - x_i\|_2$$

and the *separation radius*

$$q_X := \min_{i \neq j} \|x_i - x_j\|_2.$$

For the theoretical results presented later, we require that the set of data sites is *quasi-uniform*, i.e., there is a constant $c_{\text{qu}} > 0$ such that $q_X \leq h_{X,\Omega} \leq c_{\text{qu}} q_X$. But note that the subsequent error estimates do not require quasi-uniformity of X. Quasi-uniformity is merely rather required to bound the complexity. A comparison of volumes yields the relation $h_{X,\Omega} \sim N^{-1/d}$ for a quasi-uniform set X, see, e.g., [36, Proposition 14.1].

If $\mathcal{H}$ is isomorphic to the Sobolev space $H^s(\Omega)$ with $s > d/2$, i.e., if there holds $\|f\|_{\mathcal{H}} \sim \|f\|_{H^s(\Omega)}$ for all $f \in \mathcal{H}$, then we have the following error estimate

$$\|f - f_X\|_{L^2(\Omega)} \lesssim h_{X,\Omega}^s \|f\|_{H^s(\Omega)}, \tag{2.2}$$

compare [36]. Moreover, by using [33, Theorem 1] and

$$(u, v)_{\mathcal{H}} \lesssim \|u\|_{L^2(\Omega)} \|v\|_{H^{2s}(\Omega)}$$

for all $u \in \mathcal{H}$ and $v \in H^{2s}(\Omega)$, we may double the rate of convergence with respect to $L^2(\Omega)$, when the data provide extra smoothness in terms of $f \in H^{2s}(\Omega)$. In this case, there holds

$$\|f - f_X\|_{L^2(\Omega)} \lesssim h_{X,\Omega}^{2s} \|f\|_{H^{2s}(\Omega)}. \tag{2.3}$$

Finally, we also establish an error estimate with respect to the energy space $\mathcal{H}$. By using the orthogonality $f - f_X \perp_{\mathcal{H}} \mathcal{H}_X$, we conclude

$$\begin{aligned} \|f - f_X\|_{\mathcal{H}}^2 &= (f - f_X, f)_{\mathcal{H}} \\ &\lesssim \|f - f_X\|_{L^2(\Omega)} \|f\|_{H^{2s}(\Omega)} \\ &\lesssim h_{X,\Omega}^{2s} \|f\|_{H^{2s}(\Omega)}^2, \end{aligned}$$

which implies the desired error estimate with respect to the energy space, i.e.,

$$\|f - f_X\|_{\mathcal{H}} \lesssim h_{X,\Omega}^{s} \|f\|_{H^{2s}(\Omega)}. \tag{2.4}$$

In view of (2.2), (2.3), and (2.4), we may employ standard interpolation arguments to summarize the above error estimates in accordance with

$$\|f - f_X\|_{H^t(\Omega)} \lesssim h_{X,\Omega}^{t'-t} \|f\|_{H^{t'}(\Omega)}, \quad 0 \le t \le s \le t' \le 2s. \tag{2.5}$$

2.3. **Multilevel sequences.** We consider a sequence of quasi-uniform sets of data sites

$$X_0 \subset X_1 \subset X_2 \subset \cdots \subset \Omega \tag{2.6}$$

such that $h_j := h_{X_j,\Omega} \sim 2^{-j}$ and, consequently, $|X_j| \sim 2^{jd}$. Associated to the sequence of sets of data sites, we obtain the multilevel hierarchy of finite dimensional approximation spaces

$$\mathcal{H}_0 \subset \mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H},$$

where we write $\mathcal{H}_j := \mathcal{H}_{X_j}$ for the sake of simplicity.

Let

$$P_j \colon \mathcal{H} \to \mathcal{H}_j \tag{2.7}$$

denote the $\mathcal{H}$-orthogonal projection onto $\mathcal{H}_j$ and define the *detail projection*

$$Q_j := P_j - P_{j-1}, \tag{2.8}$$

where we set $P_{-1} := 0$, i.e., $Q_0 = P_0$. The detail projections Q_j give rise to the $\mathcal{H}$-orthogonal decomposition

$$\mathcal{H}_J = \bigoplus_{j=0}^{J} \mathcal{W}_j, \quad \text{where } \mathcal{W}_j := Q_j(\mathcal{H}).$$

Especially, the error estimate (2.5) implies

$$\begin{aligned} \|Q_j f\|_{H^t(\Omega)} &\le \|f - P_j f\|_{H^t(\Omega)} + \|f - P_{j-1} f\|_{H^t(\Omega)} \\ &\lesssim h_j^{t'-t} \|f\|_{H^{t'}(\Omega)}, \end{aligned} \tag{2.9}$$

for all $0 \le t \le s \le t' \le 2s$ provided that $f \in H^{t'}(\Omega)$.

3. Multivariate setting

3.1. **Tensor product spaces.** We consider $m \in \mathbb{N}$ possibly distinct RKHS $\mathcal{H}^{(1)}, \dots, \mathcal{H}^{(m)}$ with reproducing kernels $\kappa_1(x_1, y_1), \dots, \kappa_m(x_m, y_m)$ and associated regions $\Omega_1 \in \mathbb{R}^{d_1}, \dots, \Omega_m \in \mathbb{R}^{d_m}$, respectively. We are interested in the efficient approximation of functions in the tensor product space

$$\boldsymbol{\mathcal{H}} := \bigotimes_{i=1}^{m} \mathcal{H}^{(i)}.$$

Of course, this is again an RKHS with reproducing kernel in product form

$$\boldsymbol{\kappa}(\boldsymbol{x}, \boldsymbol{y}) := \kappa_1(x_1, y_1) \cdots \kappa_m(x_m, y_m),$$

where $\boldsymbol{x} = (x_1, \dots, x_m), \boldsymbol{y} = (y_1, \dots, y_m) \in \boldsymbol{\Omega}$ with $\boldsymbol{\Omega} := \Omega_1 \times \cdots \times \Omega_m$ denoting the m-fold product region.

For each $i = 1, \dots, m$, we assume the existence of a nested sequence of sets of data sites, i.e.,

$$X_0^{(i)} \subset X_1^{(i)} \subset X_2^{(i)} \subset \cdots \subset \Omega_i,$$

such that $h_j^{(i)} := h_{X_j^{(i)}, \Omega_i} \sim 2^{-j}$. This yields associated multiscale hierarchies of finite dimensional approximation spaces

$$\mathcal{H}_0^{(i)} \subset \mathcal{H}_1^{(i)} \subset \mathcal{H}_2^{(i)} \subset \cdots \subset \mathcal{H}^{(i)}, \quad i = 1, \dots, m,$$

with $\mathcal{H}_j^{(i)} := \mathcal{H}_{X_j^{(i)}}^{(i)}$. Given a multi-index $\boldsymbol{j} = [j_1, \dots, j_m] \subset \mathbb{N}_0^m$, we can define the *tensor product grid*

$$\boldsymbol{X_j} := X_{j_1}^{(1)} \times \cdots \times X_{j_m}^{(m)} \subset \boldsymbol{\Omega}$$

with associated tensor product approximation space

$$\boldsymbol{\mathcal{H}_j} := \operatorname{span}\{\boldsymbol{\kappa}(\cdot, \boldsymbol{x}) : \boldsymbol{x} \in \boldsymbol{X_j}\} = \mathcal{H}_{j_1}^{(1)} \otimes \cdots \otimes \mathcal{H}_{j_m}^{(m)} \subset \boldsymbol{\mathcal{H}}.$$

Given a function $u \in \boldsymbol{\mathcal{H}}$, the kernel interpolant $u_{\boldsymbol{j}} \in \boldsymbol{\mathcal{H}_j}$ with respect to the tensor product grid $\boldsymbol{X_j}$ is retrieved by solving the linear system of equations

$$\boldsymbol{K_j \alpha_j} = \boldsymbol{f_j}. \tag{3.1}$$

Herein, the kernel matrix $\boldsymbol{K_j}$ is defined as the Kronecker product

$$\boldsymbol{K_j} := \boldsymbol{K}_{j_1}^{(1)} \otimes \cdots \otimes \boldsymbol{K}_{j_m}^{(m)}$$

of the univariate kernel matrices

$$\boldsymbol{K}_{j_i}^{(i)} = [\kappa(x_k, y_k)]_{x_k, y_k \in X_{j_i}^{(i)}},$$

while the right hand side is defined as $\boldsymbol{f_j} = [f(\boldsymbol{x_k})]_{\boldsymbol{x_k} \in \boldsymbol{X_j}}$. Since the kernel interpolant is the best approximation in each of the univariate subspaces, it is evident that $\boldsymbol{u_j}$ is the best approximation of $u \in \boldsymbol{\mathcal{H}}$ in the subspace $\boldsymbol{\mathcal{H}_j}$ with respect to the norm in $\boldsymbol{\mathcal{H}}$. Since the number of interpolation points

$$|\boldsymbol{X_j}| = \prod_{i=1}^{m} |X_{j_i}^{(i)}| = \prod_{i=1}^{m} 2^{j_i}$$

grows exponentially in m, the computation of $\boldsymbol{\alpha_j}$ suffers from the curse of dimension.

3.2. **Sparse tensor product spaces.** A way to mitigate the curse of dimension is to employ *sparse tensor product approximation.* To this end, we introduce the $\boldsymbol{\mathcal{H}_j}$-orthogonal detail projections, cf. (2.8),

$$\boldsymbol{Q_j}\colon \boldsymbol{\mathcal{H}} \to \boldsymbol{\mathcal{H}_j}, \quad \boldsymbol{Q_j} := Q_{j_1}^{(1)} \otimes \cdots \otimes Q_{j_m}^{(m)}, \quad \boldsymbol{j} \geq \boldsymbol{0}.$$

We assume that the univariate spaces $\mathcal{H}^{(i)}$ are equivalent to Sobolev spaces $H^{s_i}(\Omega_i)$ for all $i = 1, \ldots, m$ and for the vector of Sobolev indices $\boldsymbol{s} = [s_1, \ldots, s_m]^T$. Moreover, for $\mathbf{t} = [t_1, \ldots, t_m]^T \geq \mathbf{0}$, we introduce the tensor product Sobolev space

$$\boldsymbol{H^t}(\boldsymbol{\Omega}) := H^{t_1}(\Omega_1) \otimes \cdots \otimes H^{t_m}(\Omega_m).$$

This tensor product Sobolev space is frequently also called Sobolev space of functions with dominating mixed derivatives. Here and in the following, inequalities between vectors are to be understood componentwise.

In view of (2.9), we conclude by standard tensor product arguments the decay estimate

$$\|\boldsymbol{Q_j} f\|_{\boldsymbol{H^t}(\boldsymbol{\Omega})} \lesssim \boldsymbol{h_j^{t'-\mathbf{t}}} \|f\|_{\boldsymbol{H^{t'}}(\boldsymbol{\Omega})}, \quad \boldsymbol{0} \leq \boldsymbol{t} \leq \boldsymbol{s} \leq \boldsymbol{t'} \leq 2\boldsymbol{s}, \tag{3.2}$$

where

$$\boldsymbol{h_j^{t'-\mathbf{t}}} := \big(h_{j_1}^{(1)}\big)^{t_1'-t_1} \cdots \big(h_{j_m}^{(m)}\big)^{t_m'-t_m},$$

as usual for powers of vectors with matching dimensions.

Next, we define *sparse tensor product spaces.* To this end, we introduce a weight vector $\boldsymbol{0} < \boldsymbol{w} = [w_1, \ldots, w_m]^T$ such that $\|\boldsymbol{w}\|_\infty = 1$. The (weighted) sparse tensor product space is then defined by

$$\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}} = \bigoplus_{\boldsymbol{j}^T\boldsymbol{w} \leq J} \boldsymbol{\mathcal{W}_j}, \quad \text{where } \boldsymbol{\mathcal{W}_j} := \boldsymbol{Q_j}(\boldsymbol{\mathcal{H}}).$$

Corresponding to $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$, we define the *sparse grid projection*

$$\widehat{\boldsymbol{P}}_J^{\boldsymbol{w}}\colon \boldsymbol{\mathcal{H}} \to \widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}, \quad \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}} f = \sum_{\boldsymbol{j}^T\boldsymbol{w} \leq J} \big(Q_{j_1}^{(1)} \otimes \cdots \otimes Q_{j_m}^{(m)}\big) f,$$

which yields the sparse grid kernel interpolant $\widehat{u}_J^{\boldsymbol{w}} = \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}} f \in \widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ of a given function $f \in \boldsymbol{\mathcal{H}}$.

3.3. **Error estimates.** In [15, 16], the construction of generalized sparse tensor product spaces has been considered. Following the theory provided therein, we derive the following results:

Theorem 3.1 (Convergence)**.** *Let $\boldsymbol{0} \leq \boldsymbol{t} < \boldsymbol{s} < \boldsymbol{t'} \leq 2\boldsymbol{s}$ and $f \in \boldsymbol{H^{t'}}(\boldsymbol{\Omega})$. Then, there holds the error estimate*

$$\big\|\big(I - \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}}\big) f\big\|_{\boldsymbol{H^t}(\boldsymbol{\Omega})} \lesssim 2^{-J \min\{\frac{t_1'-t_1}{w_1}, \ldots, \frac{t_m'-t_m}{w_m}\}} J^{P-1} \|f\|_{\boldsymbol{H^{t'}}(\boldsymbol{\Omega})}. \tag{3.3}$$

Here, $P \in \mathbb{N}$ counts how often the minimum is attained in the exponent.

Proof. We have by the triangle inequality and by (3.2) that

$$\big\|\big(I - \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}}\big) f\big\|_{\boldsymbol{H^t}(\boldsymbol{\Omega})} \leq \sum_{\boldsymbol{j}^T\boldsymbol{w} > J} \|\boldsymbol{Q_j} f\|_{\boldsymbol{H^t}(\boldsymbol{\Omega})} \lesssim \sum_{\boldsymbol{j}^T\boldsymbol{w} > J} \boldsymbol{h_j^{t-t'}} \|f\|_{\boldsymbol{H^t}(\boldsymbol{\Omega})}.$$

Due to $h_{j_i}^{(i)} \sim 2^{-j}$ for all $i = 1, 2, \ldots, m$ and hence $\boldsymbol{h_j} \sim 2^{-|\boldsymbol{j}|}$, we can now follow line-by-line the proof of [16, Theorem 4.2] and obtain the desired estimate. □

Remark 3.2. Estimate (3.3) remains valid without the logarithmic factor J^{P-1} in the case $\boldsymbol{t} = \boldsymbol{t}' = \boldsymbol{s}$ due to the Galerkin orthogonality in accordance with

$$\|(I - \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}})f\|_{\boldsymbol{\mathcal{H}}}^2 = \sum_{\boldsymbol{j}^T\boldsymbol{w}>J} \|\boldsymbol{Q_j}f\|_{\boldsymbol{\mathcal{H}}}^2 \le \|f\|_{\boldsymbol{\mathcal{H}}}^2.$$

As a consequence, if $\boldsymbol{t} = \boldsymbol{s}$ and $\boldsymbol{s} < \boldsymbol{t}'$, the logarithmic factor in (3.3) is only $J^{(P-1)/2}$. Likewise, by applying the Aubin-Nitsche lemma, one concludes only the factor $J^{(P-1)/2}$ if $\boldsymbol{t} < \boldsymbol{s}$ and $\boldsymbol{s} = \boldsymbol{t}'$, which improves the result of [25, 26].

We shall next count the degrees of freedom, i.e., the dimension, of the sparse tensor product space $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$.

Theorem 3.3 (Complexity)**.** *For any $\boldsymbol{w} > \boldsymbol{0}$, the dimension of the sparse tensor product space $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ is proportional to $2^{J\max\{d_1/w_1,\ldots,d_m/w_m\}}J^{R-1}$, where $R \in \mathbb{N}$ counts how often the maximum is attained.*

Proof. In view of $\dim \mathcal{H}_{j_i}^{(i)} = 2^{j_i d_i}$ for all $i = 1, 2, \ldots, m$, the assertion follows by nearly verbatim rewriting the proof of [16, Theorem 4.1]. □

As shown in [16], the combination of Theorems 3.1 and 3.3 yields the following estimate on the cost-complexity of the approximation in the sparse tensor product space $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$:

Theorem 3.4 (Cost-complexity rate)**.** *Let $\boldsymbol{0} \le \boldsymbol{t} < \boldsymbol{s} < \boldsymbol{t}' \le 2\boldsymbol{s}$ and $f \in \boldsymbol{H}^{\boldsymbol{t}'}(\boldsymbol{\Omega})$. Furthermore, denote by $N := \dim \widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ the number of degrees of freedom in the sparse tensor product space $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ and set*

$$\beta := \frac{\min\{(t_1' - t_1)/w_1, \ldots, (t_m' - t_m)/w_m\}}{\max\{d_1/w_1, \ldots, d_m/w_m\}}.$$

Assume that the minimum in the enumerator is attained $P \in \mathbb{N}$ times and the maximum in the denominator is attained $R \in \mathbb{N}$ times. Then, the sparse grid kernel interpolant in $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ satisfies the error estimate

$$\|(I - \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}})f\|_{\boldsymbol{H}^{\boldsymbol{t}}(\boldsymbol{\Omega})} \lesssim N^{-\beta}(\log N)^{(P-1)+\beta(R-1)}\|f\|_{\boldsymbol{H}^{\boldsymbol{t}'}(\boldsymbol{\Omega})} \tag{3.4}$$

in terms of the degrees of freedom N.

It has been shown in [16, Lemma 5.1] that there holds

$$\beta \le \beta^\star := \min\left\{\frac{t_1' - t_1}{d_1}, \ldots, \frac{t_m' - t_m}{d_m}\right\}$$

for all $\boldsymbol{w} > \boldsymbol{0}$. Moreover, if the above minimum is attained for the index $1 \le \ell \le m$, then we achieve the maximum rate $\beta = \beta^\star$ in (3.4) for all $\boldsymbol{w} > \boldsymbol{0}$ such that

$$\frac{t_\ell' - t_\ell}{t_i' - t_i} \le \frac{w_\ell}{w_i} \le \frac{d_\ell}{d_i} \quad \text{for all } i = 1, 2, \ldots, m. \tag{3.5}$$

Natural choices of the parameter $\boldsymbol{w} > \boldsymbol{0}$ are:

(*i.*) To equilibrate the accuracy in the extremal univariate spaces $\mathcal{H}_{J/w_i}^{(i)}$, $i = 1, 2, \ldots, m$, we obtain the condition

$$2^{-J(t_1'-t_1)/w_1} = 2^{-J(t_2'-t_2)/w_2} = \cdots = 2^{-J(t_m'-t_m)/w_m}.$$

This means that we have to choose $\widetilde{w}_i := t_i' - t_i$ for all $i = 1, 2, \ldots, m$ and then rescale $\boldsymbol{w} := \widetilde{\boldsymbol{w}}/\|\widetilde{\boldsymbol{w}}\|_\infty$. This choice corresponds to the lower bound in (3.5).

(*ii.*) To equilibrate the number of degrees of freedom in the extremal univariate spaces $\mathcal{H}^{(i)}_{J/w_i}$, $i = 1, 2, \ldots, m$, we obtain the condition

$$2^{Jd_1/w_1} = 2^{Jd_2/w_2} = \cdots = 2^{Jd_m/w_m}.$$

This condition is satisfied if $\widetilde{w}_i := d_i$ for all $i = 1, 2, \ldots, m$ and then setting $\boldsymbol{w} := \widetilde{\boldsymbol{w}}/\|\widetilde{\boldsymbol{w}}\|_\infty$. This choice yields the upper bound in (3.5).

(*iii.*) Following the idea of an *equilibrated cost-benefit rate* (see [3]), we get the condition

$$2^{j_1(d_1+t_1'-t_1)} \cdot 2^{j_2(d_2+t_2'-t_2)} \cdots 2^{j_m(d_m+t_m'-t_m)} = 2^{J\cdot const.}$$

for all $\boldsymbol{j}^T\boldsymbol{w} = J$. For $const. = 1$, we find $\widetilde{w}_i = d_i + t_i' - t_i$ for all $i = 1, 2, \ldots, m$. By setting again $\boldsymbol{w} := \widetilde{\boldsymbol{w}}/\|\widetilde{\boldsymbol{w}}\|_\infty$ we derive a weight $\boldsymbol{w}$ which is between the lower and upper bound in (3.5) provided that these differ from each other.

We like to emphasize that the equilibration of the degrees of freedom is the only choice which gives always the highest rate $\beta^\star$ (except for polylogarithmic factors), independent of the kernel under consideration or the particular smoothness of the function to be approximated. We refer the reader to [16] for a more detailed discussion.

3.4. **Comparison of sampling rates.** We now want to put our result into perspective. In the regular sparse grid case on the unit m-cube $\boldsymbol{\Omega} = [0, 1]^m$ and a product kernel that belongs to an RKHS being equivalent to $H^s([0, 1])$, i.e.

$$n_1 = n_2 = \cdots = n_m = 1, \quad s_1 = s_2 = \cdots = s_m = s,$$

the upper and lower bound coincide and the only optimal weight is

$$w_1 = w_2 = \cdots = w_m = 1.$$

It is well known that the standard Smolyak construction without exploiting orthogonality gives

$$\|f - f_N^{\text{Smolyak}}\|_{L^2(\boldsymbol{\Omega})} \lesssim N^{-s}(\log N)^{(s+1)(m-1)}\|f\|_{\boldsymbol{H}^s(\boldsymbol{\Omega})},$$

see [34]. But we can now exploit the orthogonality with respect to the RKHS in the error estimate as outlined in Remark 3.2. Hence, (3.3) has only the logarithmic power $(P-1)/2$ instead of $P-1$ for $\mathbf{t} = \mathbf{0}$ and $\mathbf{t}' = \mathbf{s}$. Thus, since $P = R = m$ and $\beta = s$, the respective cost-complexity rate for a function $f \in \boldsymbol{H}^s(\boldsymbol{\Omega})$ is

$$\|(I - \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}})f\|_{L^2(\boldsymbol{\Omega})} \lesssim N^{-s}(\log N)^{(s+1/2)(m-1)}\|f\|_{\boldsymbol{H}^s(\boldsymbol{\Omega})}. \tag{3.6}$$

Note at this point that it is known from [7] that there exists a set of N points such that the best possible sampling rate would be given by

$$\|f - f_N^{\text{best}}\|_{L^2(\boldsymbol{\Omega})} \lesssim N^{-s}(\log N)^{s(m-1)}\|f\|_{\boldsymbol{H}^s(\boldsymbol{\Omega})}.$$

This approach is however not constructive and such optimal point sets are not yet computable. The currently best point sets which are constructable provide the rate

$$\|f - f_N^{\text{constructive}}\|_{L^2(\boldsymbol{\Omega})} \lesssim N^{-s}(\log N)^{s(m-1)+1/2}\|f\|_{\boldsymbol{H}^s(\boldsymbol{\Omega})}. \tag{3.7}$$

compare [2]. This rate can be seen from (1.8) in [2] and the linear widths for Sobolev spaces of bounded mixed derivatives $\mathbf{H^s}(\boldsymbol{\Omega})$ in [9, p. 46]. We should emphasize that such point sets have to be computed in an offline phase that has runtime $\mathcal{O}(N^3)$.

The cost-complexity rate (3.7) is the same as for our sparse grid point sets in (3.6) for the case $m = 2$ and is indeed better for $m > 2$ by an additive factor $(m-2)/2$ in the exponent of the logarithmic term. However, the huge practical advantage of sparse grid points over more general point sets is that the point distributions are structured which can be exploited to speed-up computations considerably. Moreover, the parallelization of the implementation based on the sparse grid combination technique is straightforward.

3.5. **Sparse grid combination technique.** Due to the Galerkin orthogonality, it is easy to see that the detail projections satisfy

$$(\boldsymbol{Q}_{\boldsymbol{j}} u, \boldsymbol{Q}_{\boldsymbol{j}'} v)_{\boldsymbol{\mathcal{H}}} = 0 \quad \text{for } \boldsymbol{j} \neq \boldsymbol{j}' \text{ and any } u, v \in \boldsymbol{\mathcal{H}}.$$

Therefore, the subspaces $\boldsymbol{\mathcal{W}}_{\boldsymbol{j}}$ and $\boldsymbol{\mathcal{W}}_{\boldsymbol{j}'}$ are $\boldsymbol{\mathcal{H}}$-perpendicular. Thus, since the kernel under consideration is of product type, the theory of [24] tells us that we can compute the kernel interpolant in the sparse tensor product space $\widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ by means of the combination technique.

With this in mind, we define the tensorized version of the orthogonal projections (2.7) given by

$$\boldsymbol{P}_{\boldsymbol{j}} \colon \boldsymbol{\mathcal{H}} \to \boldsymbol{\mathcal{H}}_{\boldsymbol{j}}, \quad \boldsymbol{P}_{\boldsymbol{j}} := P_{j_1}^{(1)} \otimes \cdots \otimes P_{j_m}^{(m)},$$

and note that there holds the identity

$$\boldsymbol{P}_{\boldsymbol{j}} = \sum_{\boldsymbol{\ell} \leq \boldsymbol{j}} \boldsymbol{Q}_{\boldsymbol{\ell}}.$$

Moreover, in accordance with [5, 6, 17, 19, 34], we introduce the (weighted) *combination technique index set*

$$\mathcal{J}_J^{\boldsymbol{w}} := \big\{ \boldsymbol{j} \in \mathbb{N}_0^m : J - |\boldsymbol{w}| < \boldsymbol{j}^T \boldsymbol{w} \leq J \big\}. \tag{3.8}$$

With these definitions set at hand, one has the identity

$$\widehat{\boldsymbol{P}}_J^{\boldsymbol{w}} = \sum_{\boldsymbol{j} \in \mathcal{J}_J^{\boldsymbol{w}}} c_{\boldsymbol{j}}^{\boldsymbol{w}} \boldsymbol{P}_{\boldsymbol{j}}, \quad \text{where } c_{\boldsymbol{j}}^{\boldsymbol{w}} := \sum_{\substack{\boldsymbol{j}' \in \{0,1\}^m \\ (\boldsymbol{j}+\boldsymbol{j}')^T \boldsymbol{w} \leq J}} (-1)^{|\boldsymbol{j}'|}. \tag{3.9}$$

Hence, the sought sparse grid kernel interpolant $\widehat{u}_J^{\boldsymbol{w}} = \widehat{\boldsymbol{P}}_J^{\boldsymbol{w}} f \in \widehat{\boldsymbol{\mathcal{H}}}_J^{\boldsymbol{w}}$ is composed by the tensor product kernel interpolants $u_{\boldsymbol{j}} = \boldsymbol{P}_{\boldsymbol{j}} f$ from different full tensor product spaces $\boldsymbol{\mathcal{H}}_{\boldsymbol{j}}$. Each of these tensor product kernel interpolants $u_{\boldsymbol{j}}$ can now be computed in accordance with Subsection 3.1.

4. Implementation

4.1. **Construction of nested point sets.** In this section, we comment on our implementation of the sparse grid kernel interpolation. We first describe how we generate the multilevel sequence (2.6) from a given set of quasi-uniform data sites $X \in \Omega$. Then, since each particular term in the sparse grid combination technique amounts to the solution of a dense linear system of equations which is of tensor product structure, we apply tensorization methods. Moreover, we use a fast method for nonlocal operators for each subproblem that is associated to direction i, where $i = 1, \ldots, m$. As we will demonstrate by numerical experiments, we altogether obtain a very efficient method to compute the sparse grid kernel interpolant.

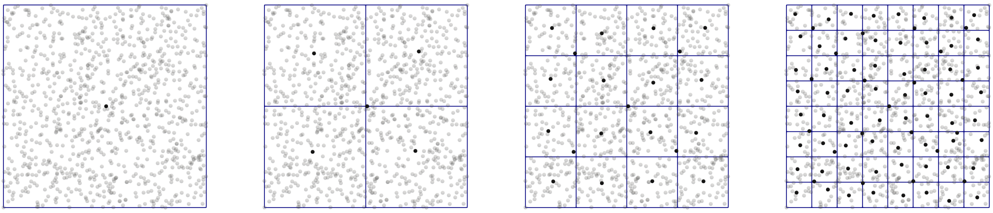

Figure 1. Visualization of the subsampling procedure starting from a set of 1000 uniformly chosen random points on $[0, 1]^2$.

To construct a multilevel sequence (2.6) for a given set of quasi-uniform data sites X, we first rescale the data sites to the unit hypercube $[0, 1]^d$ by an affine transform and then apply the following top-down algorithm: For each level $j = 0, 1, \ldots, J$, we subdivide $[0,1]^d$ equidistantly into 2^{jd} cuboids of edge length 2^{-j}. To determine the point set X_j, we start from X_{j-1} and add points that are not already contained in this set. To that end, from all points that are in a given cuboid, the point which is closest to the midpoint of the cuboid is chosen (in case of nonuniqueness, one randomly chooses one of the closest points). Thus, if each cuboid's intersection with the region Ω contains at least one point, a fill distance $h_{X_j,\Omega} \sim 2^{-j}$ is guaranteed. We remark that this is already achieved by taking any point within each cuboid. But choosing the point closest to the midpoint has the advantage of improving the separation radius. A visualization of the subsampling procedure for $j = 0, 1, 2, 3$, starting from a set of 1000 uniformly chosen random points on $[0,1]^2$, is given in Figure 1.

Algorithm 1 Uniform Subsample

```
function uniformSubsample($\mathcal{I}, X, J$)
    $\boldsymbol{b}_{\min} \leftarrow [\min_{\boldsymbol{x}\in X} x^{(o)}]_{o=1}^d$
    $\boldsymbol{b}_{\max} \leftarrow [\max_{\boldsymbol{x}\in X} x^{(o)}]_{o=1}^d$
    $\boldsymbol{a} \leftarrow \boldsymbol{b}_{\max} - \boldsymbol{b}_{\min}$
    $\mathcal{I}^c \leftarrow [1, \ldots, |X|] \setminus \mathcal{I}$
    $\mathcal{I}_{\text{new}} \leftarrow \emptyset$
    for all $p \in \mathcal{I}^c$ do
        $\boldsymbol{c} \leftarrow 2^J(\boldsymbol{x}_p - \boldsymbol{b}_{\min}) \oslash \boldsymbol{a}$
        $\boldsymbol{m} \leftarrow \big(2^{-J}(\boldsymbol{c} + 0.5) + \boldsymbol{b}_{\min}\big) \odot \boldsymbol{a}$
        if $p_{\boldsymbol{c}} \in \mathcal{I}_{\text{new}}$ then
            if $\|\boldsymbol{x}_{p_{\boldsymbol{c}}} - \boldsymbol{m}\| > \|\boldsymbol{x}_p - \boldsymbol{m}\|$ then
                $p_{\boldsymbol{c}} \leftarrow p$
        else
            $p_{\boldsymbol{c}} \leftarrow p$
            $\mathcal{I}_{\text{new}} \leftarrow \mathcal{I}_{\text{new}} \cup \{p_{\boldsymbol{c}}\}$
    return $\mathcal{I} \cup \mathcal{I}_{\text{new}}$
```

An implementation can be found in Algorithm 1. It updates a given index set $\mathcal{I}$ by selecting associated points as described above which are not already in $\mathcal{I}$. Starting from $\mathcal{I} = \emptyset$ and iterating then for $j = 0, \ldots, J$ results in the desired multilevel hierarchy. Note that "$\oslash$" denotes here the element-wise division of two vectors while "$\odot$" is their Hadamard product. The cost of the algorithm for each level j is linear in the cardinality of X.

4.2. **Computing the sparse grid kernel interpolant.** For the computation of the sparse grid kernel interpolant, we rely on the combination technique (3.8). For each multi-index $\boldsymbol{j} \in \mathcal{J}_J^{\boldsymbol{w}}$, we have to solve the tensor product linear system

$$\boldsymbol{K}_{\boldsymbol{j}} \boldsymbol{\alpha}_{\boldsymbol{j}} = \boldsymbol{f}_{\boldsymbol{j}} \quad \text{with} \quad \boldsymbol{K}_{\boldsymbol{j}} = \boldsymbol{K}_{j_1}^{(1)} \otimes \cdots \otimes \boldsymbol{K}_{j_m}^{(m)}, \tag{4.1}$$

compare (3.1). To exploit the tensor product structure of the linear system (4.1), we need a suitable representation of the quantities $\boldsymbol{\alpha}_{\boldsymbol{j}}$ and $\boldsymbol{f}_{\boldsymbol{j}}$ in the computer. Moreover, we need to be able to unfold the tensor linear system (4.1) to conventional linear systems $\boldsymbol{K}_{j_i}^{(i)} \boldsymbol{\alpha}_{j_i}^{(i)} = \boldsymbol{f}_{j_i}^{(i)}$ which belong to the directions $i = 1, \ldots, m$, and are to be solved successively. Finally, we need a backtransform of the resulting solutions $\boldsymbol{\alpha}_{j,i}^{(i)}$ to their associated tensor representation $\boldsymbol{\alpha}_{\boldsymbol{j}}^{(i)}$. Such tensor methods have become important tools in the recent years, see [18] for example, and can be applied in our context.

The class Tensor in Algorithm 2 provides an implementation of an elementwise serialization of a given tensor in main memory by means of the method toScalarIndex.

Algorithm 2 Class Tensor

```
function toScalarIndex($\boldsymbol{k}, \boldsymbol{n} \in \mathbb{N}_0^m$)
    $\boldsymbol{b} \leftarrow$ strides($\boldsymbol{n}$)
    $p \leftarrow 0$
    for $i = 1, \ldots, m$ do
        $p \leftarrow p + k_i b_i$
    return $p$

function toMultiIndex($p \in \mathbb{N}_0, \boldsymbol{n} \in \mathbb{N}_0^m$)
    $\boldsymbol{b} \leftarrow$ strides($\boldsymbol{n}$)
    $\boldsymbol{k} \leftarrow \boldsymbol{0}$
    for $i = 1, \ldots, m$ do
        $(k_i, p) \leftarrow (p/b_i, p \mod b_i)$
    return $\boldsymbol{k}$

function matricize($k \in \mathbb{N}_0, o, p \in \mathbb{N}, \boldsymbol{n} \in \mathbb{N}_0^m$)
    $\boldsymbol{b} \leftarrow$ strides($\boldsymbol{n}$)
    $z \leftarrow 0$
    $r \leftarrow p$
    for $i = 1, \ldots, k - 1$ do
        $s \leftarrow b_i / n_k$
        $(c, r) \leftarrow (\lfloor r/s \rfloor, r \mod s)$
        $z \leftarrow z + c b_i$
    $r \leftarrow r + o b_k$
    for $i = k + 1, \ldots, m$ do
        $(c, r) \leftarrow (\lfloor r/b_i \rfloor, r \mod b_i)$
        $z \leftarrow z + c b_i$
    return $z$

function strides($\boldsymbol{n}$)
    $\boldsymbol{b} = \boldsymbol{0}$
    for $i = 1, \ldots, m$ do
        $b_i \leftarrow \prod_{o=i+1}^{m} n_o$
    return $\boldsymbol{b}$
```

Given a multi-index $\boldsymbol{k} \in \{0, n_1\} \times \cdots \times \{0, n_m\}$, the function assigns a unique linear index $p = \text{toScalarIndex}(\boldsymbol{k}, \boldsymbol{n}) \in \{0, \ldots, \prod_{i=1}^{m} n_i\}$. This is achieved by a mixed radix representation, with the basis generated by the method strides. The corresponding inverse mapping from a scalar index to a multi-index is given by toMultiIndex, which amounts to the Euclidean division algorithm.

Now, mapping each entry of a tensor $\boldsymbol{\alpha} \in \mathbb{R}^{\boldsymbol{n}}$ by toScalarIndex yields the serialization

$$\text{Tensor}(\boldsymbol{n}).\text{serialize}(\boldsymbol{\alpha}) \in \mathbb{R}^{\prod_{i=1}^{m} n_i}.$$

To efficiently solve the linear system (4.1), we require all possible matricizations $\boldsymbol{M} \in \mathbb{R}^{n_i \times \prod_{o \neq i} n_o}$ of $\boldsymbol{\alpha_j}$. The elementwise matricization is again based on the Euclidean division algorithm and a possible implementation is found in matricize. With a slight abuse of notation, we refer to the entire matricization as

$$\boldsymbol{M} = \text{Tensor}(\boldsymbol{n}).\text{matricize}(\boldsymbol{\alpha}, i) \in \mathbb{R}^{n_i \times \prod_{o \neq i} n_o}.$$

The complete computation of the sparse grid kernel interpolant is presented in Algorithm 3.

4.3. **Fast solution of the linear system of equations.** It remains to provide an efficient solver for each of the kernel matrices $\boldsymbol{K}_{\boldsymbol{j}_i}^{(i)}$, $i = 1, \dots, m$ and each multi-index $\boldsymbol{j} \in \mathcal{J}_J^{\boldsymbol{w}}$, occurring in line 7 of Algorithm 3. For this, we compute a sparse approximation to $\boldsymbol{K}_{\boldsymbol{j}_i}^{(i)}$ by employing the samplet-based kernel matrix compression, see [20, 23], in combination with the sparse direct solver `CHOLMOD`, see [4]. We give a brief summary of the samplet matrix compression and refer the reader to [20] for details.

Samplets are a multiresolution basis of localized discrete signed measures with vanishing moments, which have a natural embedding into RKHS by means of the Riesz isometry. Let $\boldsymbol{K}_{\boldsymbol{j}_i}^{\Sigma,(i)}$ denote the kernel matrix $\boldsymbol{K}_{\boldsymbol{j}_i}^{(i)}$ in samplet coordinates and let $\eta > 0$ be a fixed parameter. Then, there holds,

$$\left\| \boldsymbol{K}_{\boldsymbol{j}_i}^{\Sigma,(i)} - \widetilde{\boldsymbol{K}}_{\boldsymbol{j}_i}^{\Sigma,(i)} \right\|_F \leq C(c\eta)^{-2(q+1)} \left\| \boldsymbol{K}_{\boldsymbol{j}_i}^{\Sigma,(i)} \right\|_F, \tag{4.2}$$

where $q+1$ is the number of vanishing moments, and $C, c > 0$ are constants. The matrix $\widetilde{\boldsymbol{K}}_{\boldsymbol{j}_i}^{\Sigma,(i)}$ is obtained from $\boldsymbol{K}_{\boldsymbol{j}_i}^{\Sigma,(i)}$ by setting all entries to zero, whose associated samplets have supports τ, τ' that satisfy

$$\operatorname{dist}(\tau, \tau') \geq \eta \max\{\operatorname{diam}(\tau), \operatorname{diam}(\tau')\}, \quad \eta > 0.$$

The compressed matrix has $\mathcal{O}(N \log N)$ remaining entries. The error estimate (4.2) is valid for *asymptotically smooth kernels*, especially for the Matérn class of kernels. For such kernels, the samplet compressed kernel matrices can be computed efficiently with loglinear cost-complexity by means of a multipole method, see [14]. In contrast to this early work, we follow [13] and use $\mathcal{H}^2$-matrices and interpolation of the kernel under consideration. We refer the reader to [22] for the description of the implementation of the particular multipole method we use. To further reduce the number of entries, an a-posteriori thresholding of small entries in $\widetilde{\boldsymbol{K}}_{\boldsymbol{j}_i}^{\Sigma,(i)}$ may be performed once the samplet compressed matrix has been assembled. Figure 2 illustrates the samplet compressed matrix $\widetilde{\boldsymbol{K}}_{\boldsymbol{j}_i}^{\Sigma,(i)}$, its nested dissection reordering (see [12] for details), and the resulting Cholesky factor in case of the exponential kernel on the unit square for 300 000 uniform random data sites.

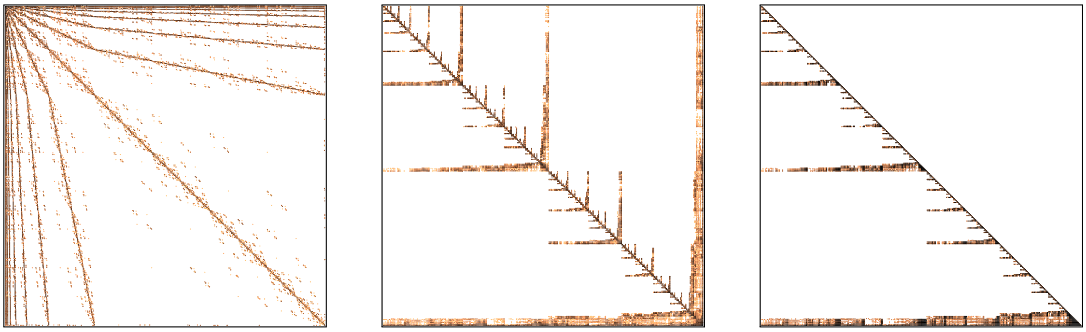

FIGURE 2. Sparsity patterns of the samplet compressed exponential kernel on the unit square (left) for 300 000 data sites, the nested dissection reordering (middle), and the Cholesky factor (right). Each dot represents a matrix block of size 300 × 300. The number of entries per block is color coded, where lighter blocks have less entries.

The sparse direct solver mitigates to some extent the computational cost for the numerical solution caused by the ill-conditioning of the kernel matrices for increasing numbers of points. Note here that the increasing condition number requires a corresponding increase in compression error accuracy to maintain a fixed overall consistency error. We refer to [1] for a detailed discussion on this matter.

Algorithm 3 Compute Sparse Grid Kernel Interpolant

```
function COMPUTE($[\boldsymbol{K}_{\boldsymbol{j}}]_{\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}}$, $[\boldsymbol{f}_{\boldsymbol{j}}]_{\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}}$)
    for all $\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}$ do
        $\boldsymbol{n}_{\boldsymbol{j}} \leftarrow \big[|X_{j_1}^{(1)}|,\ldots,|X_{j_m}^{(m)}|\big]^T$
        $\boldsymbol{\alpha}_{\boldsymbol{j}} \leftarrow$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).SERIALIZE($\boldsymbol{f}_{\boldsymbol{j}}$)
        for $i = 1,\ldots,m$ do
            $\boldsymbol{M} \leftarrow$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).MATRICIZE($\boldsymbol{\alpha}_{\boldsymbol{j}}, i$)
            $\boldsymbol{M} \leftarrow \big(\boldsymbol{K}_{j_i}^{(i)}\big)^{-1}\boldsymbol{M}$
            $\boldsymbol{\alpha}_{\boldsymbol{j}} \leftarrow$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).SERIALIZE($\boldsymbol{M}$)
    return $[\boldsymbol{\alpha}_{\boldsymbol{j}}]_{\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}}$
```

4.4. **Evaluation of the sparse grid kernel interpolant.** Given sets of evaluation points $X_{\text{eval}}^{(1)}\subset\Omega_1,\ldots,X_{\text{eval}}^{(m)}\subset\Omega_m$, the evaluation of the sparse grid kernel interpolant on the tensor product grid $\bigtimes_{i=1}^m X_{\text{eval}}^{(i)}$ is similar to the solution of the interpolation problems in the sparse grid combination technique. The linear solver just needs to be replaced by a matrix-vector multiplication with the kernel matrices $\boldsymbol{K}^{(i)}_{X_{\text{eval}}^{(i)},X_{j_i}^{(i)}}$. The evaluation of the sparse grid interpolant is summarized in Algorithm 4. The matrix-vector multiplication therein can either be performed directly, in case of a relative small number of points in $X_{j_i}^{(i)}$ or $X_{\text{eval}}^{(i)}$, or can be sped up by means of the fast multipole method.

Algorithm 4 Evaluate Sparse Grid Kernel Interpolant

```
procedure EVALUATE($[\boldsymbol{\alpha}_{\boldsymbol{j}}]_{\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}}$, $X_{\text{eval}}^{(1)},\ldots,X_{\text{eval}}^{(m)}$)
    $\boldsymbol{u} \leftarrow \boldsymbol{0}$
    for all $\boldsymbol{j}\in\mathcal{J}_J^{\boldsymbol{w}}$ do
        $\boldsymbol{n}_{\boldsymbol{j}} \leftarrow \big[|X_{j_1}^{(1)}|,\ldots,|X_{j_m}^{(m)}|\big]^T$
        for $i = 1,\ldots,m$ do
            $\boldsymbol{M} \leftarrow$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).MATRICIZE($\boldsymbol{\alpha}_{\boldsymbol{j}}, i$)
            $\boldsymbol{M} \leftarrow \boldsymbol{K}^{(i)}_{X_{\text{eval}}^{(i)},X_{j_i}^{(i)}}\boldsymbol{M}$
            $\boldsymbol{u}_{\boldsymbol{j}} \leftarrow$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).SERIALIZE($\boldsymbol{M}$)
        $\boldsymbol{u} \leftarrow \boldsymbol{u} + c_{\boldsymbol{j}}^{\boldsymbol{w}}$ TENSOR($\boldsymbol{n}_{\boldsymbol{j}}$).SERIALIZE($\boldsymbol{u}_{\boldsymbol{j}}$)
    return $\boldsymbol{u}$
```

5. Numerical results

5.1. **General setup.** In our numerical experiments, we employ the *Matérn kernels* or *Sobolev splines* $\kappa_\nu\colon\mathbb{R}^d\times\mathbb{R}^d\to\mathbb{R}$, which are dependent on the *smoothness parameter* $\nu > d/2$. They are defined by

$$\kappa_\nu(\mathbf{x},\mathbf{y}) := \frac{2^{1-\nu}}{\Gamma(\nu)} r^{\nu-\frac{d}{2}} K_{\nu-\frac{d}{2}}(r), \quad r := \frac{1}{\sigma}\|\mathbf{x}-\boldsymbol{y}\|_2, \tag{5.1}$$

where Γ is the Riemannian gamma function and K_β is the modified Bessel function of the second kind, see [27] for example. These kernels are known to be nonlocal and are hence not straightforward to deal with numerically since standard discretizations result in densely populated system matrices. Nonetheless, they are the reproducing kernels of the Sobolev spaces $H^{\nu+d/2}(\mathbb{R}^d)$ and hence of great importance in practice.

Throughout our experiments, we always interpolate the data generating process $f \equiv 1$. At first glance, this may appear like a very simple problem. However, for kernel interpolation it is nontrivial, since all RKHS under consideration do not include polynomials. On the other hand, the function $f \equiv 1$ is arbitrarily smooth and does not depend on the dimensionality, which makes it a perfect test case. As mentioned in the previous section, the compression of smoother kernels poses a particular challenge in terms of accuracy. In particular for $d = 1$, we employ samplets with $q + 1 = 9$ vanishing moments and set the parameter for the cut-off criterion to $\eta = 5$, compare [20]. In addition, an a-posteriori compression with threshold 10^{-15} relative to the Frobenius norm of the compressed kernel matrix is performed. For $d = 2, 3$, samplets with $q + 1 = 4$ vanishing moments and the parameter of the cut-off criterion set to $\eta = 2$ have been sufficient to maintain the overall consistency error. The threshold in the a-posteriori compression has been chosen as 10^{-6}, compare Section 4.3. The length scale parameter of the kernels is set to $\sigma = 2\sqrt{d}$ in our examples.

All computations have been carried out on a compute server with two AMD EPYC 7763 CPUs (64 cores each) with 2TB of main memory and using up to 16 `OpenMP` threads if not stated otherwise. The implementation of the samplet matrix compression as well as of the sparse grid combination technique are open source and available online at `https://github.com/muchip/fmca`.

5.2. **Tensor product of the unit interval.** We first consider the situation

$$\Omega_1 = \Omega_2 = \cdots = \Omega_m = [0, 1],$$

i.e., the unit hypercube $\bigtimes_{i=1}^m \Omega_i = [0, 1]^m$ and $d_1 = d_2 = \cdots = d_m = 1$. To this end, we use the tensor product kernel

$$\boldsymbol{\kappa} := \bigotimes_{i=1}^{m} \kappa,$$

where κ is the Matérn-17/16 kernel. The corresponding univariate RKHS is isomorphic to the Sobolev space $H^{25/16}(0, 1)$, that is, we have $s_1 = s_2 = \cdots = s_m = \frac{25}{16}$. Hence, the best possible univariate convergence rate with respect to $L^2(0, 1)$ is $25/8 = 3.125$ provided that the given data are smooth. Especially, the upper and lower bound in (3.5) coincide and we have to choose

$$w_1 = w_2 = \cdots = w_m = 1.$$

We use the equidistant grid points

$$X_j = \{2^{-(j+1)}k : k = 1, 2, \ldots, 2^{j+1} - 1\}, \quad j \geq 0,$$

in the univariate directions, such that our construction computes the kernel interpolant with respect to the traditional m-variate sparse grid (without points at the boundary).

We first provide a benchmark on the runtime of our implementation. Figure 3 shows the cumulative times for the setup of the direct solver in samplet coordinates, computation of the combination technique index set, the computation of the coefficients and the evaluation of the interpolant at the single point $[1/3, \ldots, 1/3]^T \in \mathbb{R}^m$ for $m = 3, 6, 9, 12, 18$ dimensions and $J = 1, 2, \ldots, 10$ levels. The combination technique index set (3.8) is computed up front using a single thread, as the computing time is negligible compared to the loops in line 2 of Algorithm 3 and in line 3 of Algorithm 4, respectively. We remark that both loops are trivial to parallelize. The reported times in this paragraph have been computed by using 64 `OpenMP` threads with dynamic load balancing.

The computation of all univariate direct solvers, which takes approximately 1.6 seconds for $J = 10$, is dominating the overall computation time until roughly $N = 10^5$ sparse grid points. For larger N, the cost for the computation of the coefficients of the sparse grid kernel interpolant and its evaluation become dominant. As can be seen in Figure 3, the computation times almost match the theoretical loglinear rate that is caused by the loglinear growth number of nonzero coefficients of the samplet compressed kernel matrices and the matrix factors used for the forward- and backward substitution, respectively.

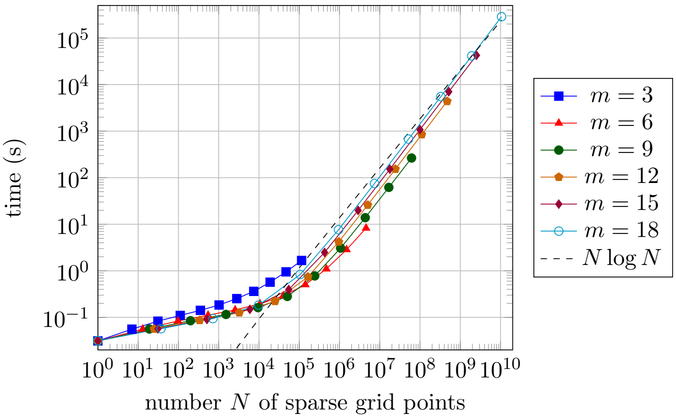

Figure 3. Computation times for the canonical sparse grid on the unit hypercube $(0,1)^m$ and $m = 3, 6, 9, 12, 15, 18$.

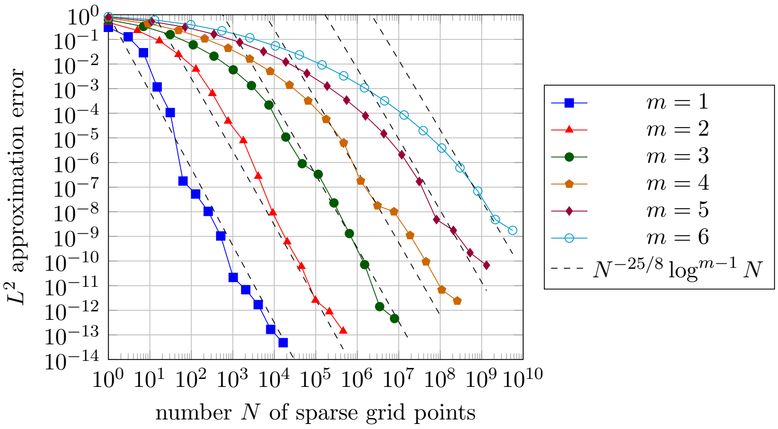

Figure 4. Convergence of the kernel interpolant on the canonical sparse grid in $(0,1)^m$.

In Figure 4, we show the convergence of the interpolant in the L^2-norm, exemplarily for $m = 1, \ldots, 6$. The L^2-norm of the error is approximated by using a tensorized four point Gauss-Legendre quadrature, which exhibits 4^m quadrature points and, hence, becomes very costly in higher dimensions. We indeed observe the theoretical convergence behaviour $N^{-\beta} \log^{m-1} N$ with $\beta = 25/8$ as predicted by Theorem 3.4 for $t_i' = 25/8$ and $t_i = 0$. Nonetheless, we also see that the constant in front of the approximation rate increases as the spatial dimension m increases, which is a well known observation for sparse grid constructions.

5.3. **Tensor product of unit hypercubes in $1+2+3$ dimensions.** Next, we consider kernel interpolation on the unit hypercube $[0,1]^6$ by splitting it into the product $\Omega_1 \times \Omega_2 \times \Omega_3$ with

$$\Omega_1 = [0,1], \quad \Omega_2 = [0,1]^2, \quad \Omega_3 = [0,1]^3.$$

The tensor product kernel, which we consider, is

$$\boldsymbol{\kappa} := \kappa_1 \otimes \kappa_2 \otimes \kappa_3,$$

where $\kappa_d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the Matérn-$\big(\frac{25}{16} - \frac{d}{2}\big)$ kernel, $d = 1, 2, 3$. Thus, each corresponding d-variate RKHS is isomorphic to the Sobolev space $H^{25/16}\big([0,1]^d\big)$. Hence, the highest convergence rate in a particular direction Ω_i is $25/8 = 3.125$.

We use a regular grid for each of the subregions, i.e., the interpolation points

$$X_j^{(1)} := \big\{2^{-(j+1)}k : k = 0, 1, \ldots, 2^{j+1}\big\}, \quad j \geq 0,$$

on Ω_1 are chosen equidistantly, while $X_j^{(2)} := \big(X_j^{(1)}\big)^2$ and $X_j^{(3)} := \big(X_j^{(1)}\big)^3$. Therefore, we have $\big|X_j^{(i)}\big| = (2^{j+1} + 1)^i$ points per level j for $i = 1, 2, 3$. In particular, there holds $h_{X,\Omega_i} = 2^{-(j+1)}\sqrt{i}$ and $q_X = 2^{-j}$ for $i = 1, 2, 3$ by construction.

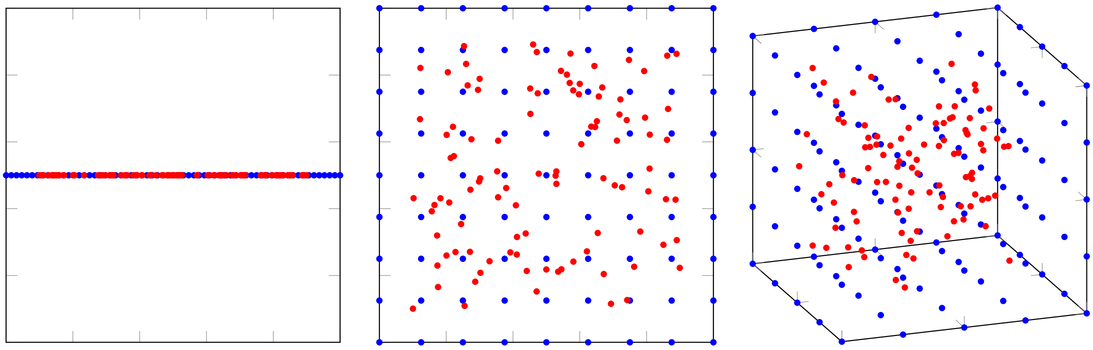

FIGURE 5. Sketch of the regular grid points (blue) and the evaluation points (red) on the unit interval, the unit square, and the unit cube.

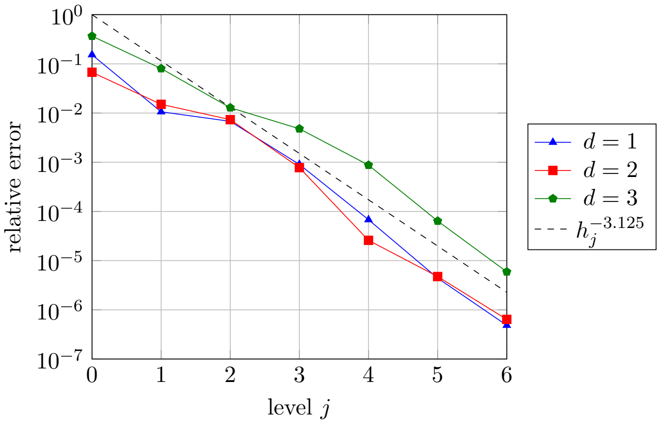

FIGURE 6. Convergence of the d-variate kernel approximation in case of the hypercube $[0,1]^d$ for $d = 1, 2, 3$.

After the kernel interpolant has been computed, it is evaluated at 100 uniform random points for each subregion Ω_i, located in a hypercube of distance 0.1 from the respective subregion's boundary. This ensures that singularity effects at the boundary are avoided which are caused by truncating the Matérn kernels to a finite region. We refer to Figure 5 for a visualization of the presented setup. Therein, the evaluation points are indicated in red.

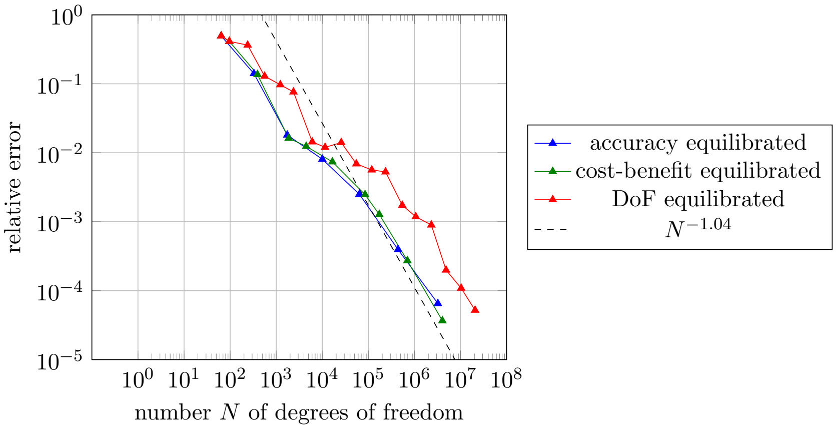

FIGURE 7. Convergence rates of the kernel approximant with respect to different sparse grids on $(0,1)\times(0,1)^2\times(0,1)^3$.

Next, we consider the d-variate approximation for $d=1,2,3$ to validate the appropriate choice of the number of vanishing moments of the samplets and the compression parameter η for the matrix compression, and thus for our solver for the different subproblem directions. The convergence of the approximant with respect to each particular subregion Ω_i is shown in Figure 6. Indeed, we observe the convergence rate $h_j^{-3.125}$ in all three case as predicted, so that we can be sure that the compression works correctly.

We next consider the kernel interpolation of the respective sparse grid. For the present setting, we can summarize the parameters as

$$d_1=1,\quad d_2=2,\quad d_3=3,\quad s_1=s_2=s_3=\frac{25}{16}. \tag{5.2}$$

Therefore, choosing the weights

$$w_1=w_2=w_3=1 \tag{5.3}$$

for the sparse grid construction equilibrates the accuracies in the particular directions, while choosing the weights

$$w_1=1/3,\quad w_2=2/3,\quad w_3=1 \tag{5.4}$$

equilibrates their degrees of freedom. For the equlibration of the cost-benefit-rate, we have to choose

$$w_1=33/49,\quad w_2=33/41,\quad w_3=1. \tag{5.5}$$

The resulting convergence rates with respect to the number N of the degrees of freedom are given in Figure 7. The expected rate of convergence is $N^{-\beta}$ with $\beta=25/24\approx 1.04$ up to polylogarithmic terms. Indeed, after some preasymptotic regime, we observe the predicted convergence rate of $N^{-1{,}04}$.

5.4. **Tensor product of general regions in $1+2+3$ dimensions.** In our final numerical experiment, we consider the tensor product of uniformly chosen random points on the unit interval $\Omega_1=[0,1]$, of uniformly chosen random points on the sphere $\Omega_2=\mathbb{S}^2$, and the nodal points of a tetrahedral mesh of a rabbit $\Omega_3\subset\mathbb{R}^3$ (involving three-dimensional points at the surface of the well-known Stanford bunny and in the interior of the bunny). We refer to Figure 8 for an illustration of this geometrical situation.

Table 1 lists the number of points per level for each of the geometries and all considered combinations. As can be seen, when proceeding from level j to $j+1$, the number of points approximately doubles on the interval. For the sphere, which is a two-dimensional

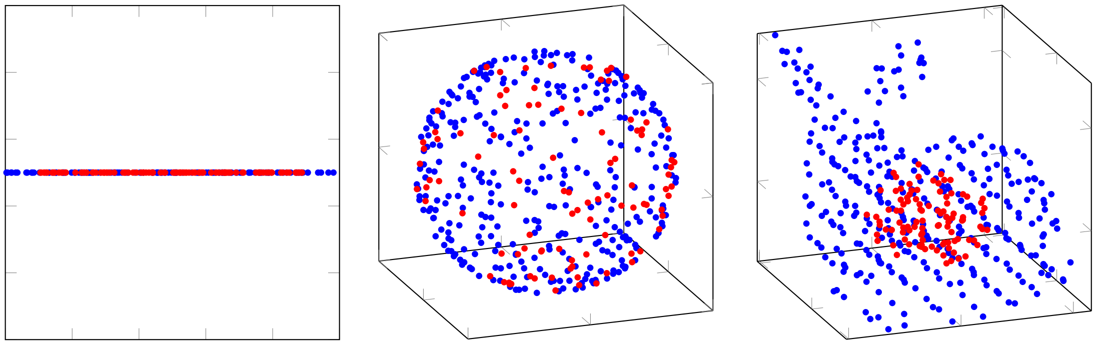

FIGURE 8. Sketch of the quasi-uniform points (blue) and evaluation points (red) on the unit interval, the unit sphere, and the Stanford bunny.

	Interval	Sphere	Rabbit
$j=0$	1	1	1
$j=1$	3	9	9
$j=2$	7	65	58
$j=3$	15	337	326
$j=4$	31	1497	1933
$j=5$	63	6246	12482
$j=6$	127	24952	88489
$j=7$	255	97224	—
$j=8$	511	—	—
$j=9$	1023	—	—
$j=10$	2047	—	—
$j=11$	4095	—	—
$j=12$	8191	—	—
$j=13$	16383	—	—
$j=14$	32767	—	—
$j=15$	65535	—	—
$j=16$	131071	—	—
$j=17$	262143	—	—

TABLE 1. Numbers N of points per level that enter the sparse grid construction for the interval ($d=1$), the sphere ($d=2$), and the rabbit ($d=3$).

manifold, i.e., $d_2=2$, we asymptotically observe the factor four. Moreover, the number of points of the rabbit grows with a factor about 6–8.

On the particular subregions Ω_i, we have unstructured, quasi-uniform data sites, which we coarsen by employing Algorithm 1 as given in Subsection 4.1. On the unit interval, we start from a point set with 4 319 030 points, a separation radius of $5.32\cdot10^{-14}$ and a fill distance of $2.62\cdot10^{-6}$, while on the sphere, we start from a point set with 2 879 320 points, a separation radius of $8.13\cdot10^{-7}$ and a fill distance of $5.09\cdot10^{-3}$, and finally on the rabbit, we start from a point set with 1 439 610 points, a separation radius of $5.25\cdot10^{-4}$ and a fill distance of $8.93\cdot10^{-3}$. It can be seen from Table 2 that the fill distance $h_{X_j,X}$, which we consider an approximation of $h_{X_j,\Omega}$, approximately halves with respect to the level in each particular example, as desired. On the other hand, the separation radius

	Interval		Sphere		Rabbit	
	q_{X_j}	h_{X_j,X_J}	q_{X_j}	h_{X_j,X_J}	q_{X_j}	h_{X_j,X_J}
$j=0$	—	$6.80\cdot 10^{-1}$	—	2.00	—	1.01
$j=1$	$6.25\cdot 10^{-2}$	$4.45\cdot 10^{-1}$	$4.03\cdot 10^{-1}$	1.24	$1.01\cdot 10^{-1}$	$5.82\cdot 10^{-1}$
$j=2$	$2.34\cdot 10^{-2}$	$2.03\cdot 10^{-1}$	$5.49\cdot 10^{-2}$	$5.66\cdot 10^{-1}$	$3.57\cdot 10^{-2}$	$3.01\cdot 10^{-1}$
$j=3$	$7.81\cdot 10^{-3}$	$1.41\cdot 10^{-1}$	$1.62\cdot 10^{-2}$	$2.87\cdot 10^{-1}$	$1.28\cdot 10^{-2}$	$1.76\cdot 10^{-1}$
$j=4$	$7.81\cdot 10^{-3}$	$1.33\cdot 10^{-1}$	$7.01\cdot 10^{-4}$	$1.62\cdot 10^{-1}$	$8.12\cdot 10^{-3}$	$1.06\cdot 10^{-1}$
$j=5$	$7.81\cdot 10^{-3}$	$4.68\cdot 10^{-2}$	$6.48\cdot 10^{-4}$	$8.22\cdot 10^{-2}$	$3.63\cdot 10^{-3}$	$4.69\cdot 10^{-2}$
$j=6$	$7.81\cdot 10^{-3}$	$7.81\cdot 10^{-3}$	$4.22\cdot 10^{-5}$	$2.92\cdot 10^{-2}$	$1.99\cdot 10^{-3}$	$1.83\cdot 10^{-2}$

TABLE 2. Separation radius and fill distance for the different geometries.

stays proportional to the fill distance. Therefore, we have a nested sequence of sets $X_j^{(i)}$ of data sites which satisfy $\big|X_j^{(i)}\big| \sim 2^{ji}$, $i=1,2,3$.

Moreover, after the sparse grid kernel interpolant is computed, it is evaluated at the product of randomly distributed points $X_{\text{eval}}^{(i)} \subset \Omega_i$. These are, in case of the interval and the rabbit, again chosen with a certain distance from the boundary to avoid boundary singularities of the approximants. Note that the sphere is a closed manifold, such that there are no singularities. We refer again to Figure 8 for a visualization. The convergence of the univariate solvers is shown in Figure 9. As can be seen, all of them achieve the expected convergence rate of $h_j^{-3.125}$.

The parameters for the construction of the sparse grid are the same as in the previous experiment, i.e., the parameters are given as in (5.2) for the underlying approximation spaces. Therefore, the accuracy-equilibrated sparse grid is given by the weights in (5.3), the degrees-of-freedom-equilibrated sparse grid is given by the weights in (5.4), and finally the cost-benefit-equilibrated sparse grid is given by the weights in (5.5). As can be inferred from Figure 10, the different settings produce essentially the same convergence rate, which indeed shows the $N^{-1.04}$ behavior as the number N of sparse grid points increases.

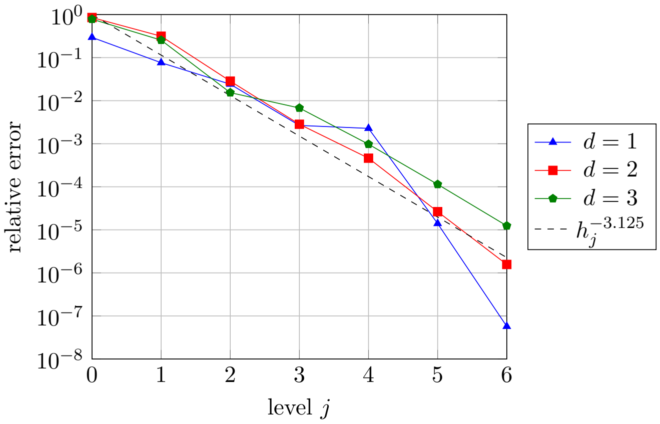

FIGURE 9. Convergence of the kernel interpolant on the interval ($d=1$), the sphere ($d=2$), and the rabbit ($d=3$).

6. CONCLUSION

In the present article, we have considered kernel interpolation on sparse grids in Sobolev spaces of dominating mixed derivatives. We have discussed the optimal construction of

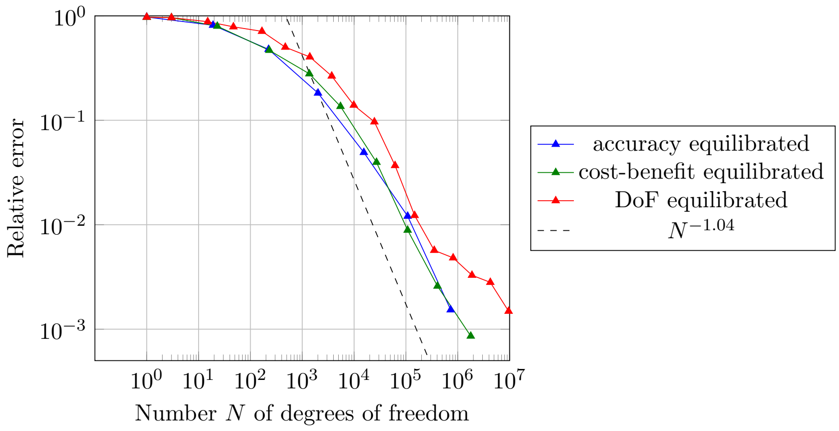

FIGURE 10. Convergence rates of the kernel approximant with respect to different sparse grids on the product of general subregions in $1+2+3$ dimensions.

the sparse grid in case of product regions of arbitrary dimension and of arbitrary smoothness with respect to the particular regions. Especially, we derived improved estimates on the approximation error, using duality arguments, provided that the function to be interpolated exhibits additional smoothness. For the numerical solution of the interpolation problem, we have proposed an efficient algorithm that combines the sparse grid combination technique with a fast direct solver for nonlocal operators on the subproblems. We presented the results of numerical experiments in up to 18 dimensions and with billions of degrees of freedoms in the sparse grid, which validate the presented theory. Wee emphasize that the problem size would have been restricted seriously without the application of an efficient method for dealing with the nonlocal kernel matrices,

We finally point out that the proposed sparse grid kernel interpolation is also applicable with straightforward modification when dimension weights are present. In this case, the logarithmic factors might be removed and even dimension-robustness can be achieved provided that the weights decay sufficiently fast. Such a situation is typically found in uncertainty quantification or machine learning, see [8, 28] for example.

Acknowledgments

Michael Griebel was supported by the *Hausdorff Center for Mathematics* (HCM) in Bonn, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2047/1 – 390685813 and by the CRC 1060 *The Mathematics of Emergent Effects* – 211504053 of the Deutsche Forschungsgemeinschaft. Helmut Harbrecht was funded in parts by the Swiss National Science Foundation (SNSF) through the Vietnamese-Swiss Joint Research Project IZVSZ2_229568. Michael Multerer was funded in parts by the Swiss Federal Office of Energy SFOE as part of the SWEET project SURE and by the SNSF starting grant "Multiresolution methods for unstructured data" (TMSGI2_211684). Finally, the authors thank Felix Bartel for fruitful discussions, especially with respect to Subsection 3.4.

References

[1] S. Avesani, R. Kempf, M. Multerer, and H. Wendland. Multiscale scattered data analysis in samplet coordinates. arXiv:2409.14791, 2024.

[2] F. Bartel, M. Schäfer, and T. Ullrich. Constructive subsampling of finite frames with applications in optimal function recovery. *Appl. Comput. Harmon. Anal.*, 65:209-248, 2023.

[3] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numer.*, 13:147–269, 2004.

[4] Y. Chen, T. Davis, W. Hager, and S. Rajamanickam, Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate, *ACM Trans. Math. Software*, 35(3):22:1–22:14, 2008.

[5] F.-J. Delvos. d-variate Boolean interpolation. *J. Approx. Theory*, 34(2):99–114, 1982.

[6] F.-J. Delvos and W. Schempp. *Boolean Methods in Interpolation and Approximation.* Longman Scientific & Technical, Harlow, 1989.

[7] M. Dolbeault, D. Krieg, and M. Ullrich. A sharp upper bound for sampling numbers in L^2. *Appl. Comput. Harmon. Anal.*, 63:113-134, 2023.

[8] D. Dũng , V. Nguyen, C. Schwab, and J. Zech. *Analyticity and Sparsity in Uncertainty Quantification for PDEs with Gaussian Random Field Inputs*, volume 2334 of *Lecture Notes in Mathematics*, Springer Nature Switzerland AG, 2023.

[9] D. Dũng, V. Temlyakov, and T. Ullrich. *Hyperbolic Cross Approximation.* Advanced Courses in Mathematics. CRM Barcelona. Birkhäuser/Springer Basel, 2018.

[10] G. Fasshauer. *Meshfree Approximation Methods with MATLAB.* World Scientific, River Edge, NJ, 2007.

[11] A. Feldstein, D. Lazzara, N. Princen, and K. Willcox. Multifidelity data fusion: Application to blended-wing-body multidisciplinary analysis under uncertainty. *AIAA J.*, 58(2):889–906, 2020.

[12] A. George. Nested dissection of a regular finite element mesh. *SIAM J. Numer. Anal.*, 10(2):345–363, 1973.

[13] K. Giebermann. Multilevel approximation of boundary integral operators. *Computing*, 67(3):183—207, 2001.

[14] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73(2):325–348, 1987.

[15] M. Griebel and H. Harbrecht. On the construction of sparse tensor product spaces. *Math. Comput.*, 82(282):975–994, 2013.

[16] M. Griebel and H. Harbrecht. A note on the construction of L-fold sparse tensor product spaces. *Constr. Approx.*, 38(2):235–251, 2013.

[17] M. Griebel, M. Schneider and C. Zenger. A combination technique for the solution of sparse grid problems. In P. de Groen and R. Beauwens, editors, *Iterative Methods in Linear Algebra*, pages 263–281. IMACS, Elsevier, North Holland, 1992.

[18] W. Hackbusch. *Tensor Spaces and Numerical Tensor Calculus.* 2nd ed., Springer, Berlin-Heidelberg, 2019.

[19] A.-L. Haji-Ali, H. Harbrecht, M. Peters, and M. Siebenmorgen. Novel results for the anisotropic sparse grid quadrature. *J. Complexity*, 47:62—85, 2018.

[20] H. Harbrecht and M. Multerer. Samplets: Construction and scattered data compression. J. Comput. Phys., 471:111616, 2022.

[21] H. Harbrecht and M. Multerer. Samplets: Wavelet concepts for scattered data. In R. DeVore and A. Kunoth, editors, *Multiscale, Nonlinear and Adaptive Approximation II*, pages 299–326. Springer, Berlin-Heidelberg, 2024.

[22] H. Harbrecht, M. Multerer, and J. Quizi. The dimension weighted fast multipole method for scattered data approximation. *J. Comput. Phys.*, 532:113956, 2025.

[23] H. Harbrecht, M. Multerer, O. Schenk, and C. Schwab. Multiresolution kernel matrix algebra. *Numer. Math.*, 156:1085–1114, 2024.

[24] H. Harbrecht, M. Peters, and M. Siebenmorgen. Combination technique based k-th moment analysis of elliptic problems with random diffusion. *J. Comput. Phys.*, 252:128–141, 2013.

[25] R. Kempf. *The Tensor Product Multilevel Method for High-dimensional Meshfree Approximation.* PhD thesis, Universität Bayreuth, 2023.

[26] R. Kempf and H. Wendland. High-dimensional approximation with kernel-based multilevel methods on sparse grids. *Numer. Math.*, 154:485–519, 2023.

[27] B. Matérn. *Spatial Variation*, volume 36 of *Lecture Notes in Statistics.* Springer, Berlin, second edition, 1986.

[28] J. Opschoor, C. Schwab, and J. Zech. Deep learning in high dimension: ReLU neural network expression for Bayesian PDE inversion De Gruyter, Berlin, 2022.

[29] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Rev.*, 60(3):550–591, 2018.

[30] C. Williams and C. Rasmussen. *Gaussian Processes for Machine Learning.* MIT Press, Cambridge, MA, 2006.

[31] R. Schaback. Superconvergence of kernel-based interpolation. *J. Approx. Theory*, 235:1–19, 2018.

[32] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, 2002.

[33] I. Sloan and V. Kaarnioja. Doubling the rate: Improved error bounds for orthogonal projection with application to numerical analysis. *BIT Numer. Math.*, 65:10, 2025.

[34] S. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Doklady Akademii Nauk SSSR*, 4:240–243, 1963.

[35] S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Academic Press, Waltham, MA, fourth edition, 2009.

[36] H. Wendland. *Scattered Data Approximation.* Cambridge University Press, Cambridge, 2004.

[37] C. Zenger. Sparse grids. In *Parallel algorithms for partial differential equations*, Proceedings of the 6th GAMM-Seminar, Kiel/Germany, 1990, Notes Numer. Fluid Mech. 31, pages 241–251, Vieweg, Braunschweig, 1991.

Michael Griebel, Institut für Numerische Simulation, Universität Bonn, Friedrich-Hirzebruch-Allee 7, 53115 Bonn, Germany and Fraunhofer Institute for Algorithms and Scientific Computing (SCAI), Schloss Birlinghoven, 53754 Sankt Augustin, Germany
Email address: griebel@ins.uni-bonn.de

Helmut Harbrecht, Departement Mathematik und Informatik, Universität Basel, Spiegelgasse 1, 4051 Basel, Switzerland
Email address: helmut.harbrecht@unibas.ch

Michael Multerer, Istituto Dalle Molle di studi sull'intelligenza artificiale, Università della Svizzera italiana, Via la Santa 1, 6962 Lugano, Switzerland
Email address: michael.multerer@usi.ch