Institut für Numerische Simulation
Rheinische Friedrich-Wilhelms-Universität Bonn
Endenicher Allee 19b • 53115 Bonn • Germany
phone +49 228 73-69828
www.ins.uni-bonn.de

B. Bohn, M. Griebel and J. Oettershagen

Optimally rotated coordinate systems for adaptive least-squares regression on sparse grids

INS Preprint No. 1812

October 2018

Optimally rotated coordinate systems for adaptive least-squares regression on sparse grids[*]

Bastian Bohn[†] Michael Griebel[‡†] Jens Oettershagen[†]

Abstract

For low-dimensional data sets with a large amount of data points, standard kernel methods are usually not feasible for regression anymore. Besides simple linear models or involved heuristic deep learning models, grid-based discretizations of larger (kernel) model classes lead to algorithms, which naturally scale linearly in the amount of data points. For moderate-dimensional or high-dimensional regression tasks, these grid-based discretizations suffer from the curse of dimensionality. Here, sparse grid methods have proven to circumvent this problem to a large extent. In this context, space- and dimension-adaptive sparse grids, which can detect and exploit a given low effective dimensionality of nominally high-dimensional data, are particularly successful. They nevertheless rely on an axis-aligned structure of the solution and exhibit issues for data with predominantly skewed and rotated coordinates.

In this paper we propose a preprocessing approach for these adaptive sparse grid algorithms that determines an optimized, problem-dependent coordinate system and, thus, reduces the effective dimensionality of a given data set in the ANOVA sense. We provide numerical examples on synthetic data as well as real-world data to show how an adaptive sparse grid least squares algorithm benefits from our preprocessing method.

Keywords: effective dimensionality, ANOVA decomposition, adaptive sparse grids, least-squares regression.

1 Introduction

In function regression, we determine f from an admissible set S which best approximates given data $(\boldsymbol{t}_i, x_i)_{i=1}^N \subset \mathbb{R}^d \times \mathbb{R}$, i.e. $f(\boldsymbol{t}_i) \approx x_i$ for $i = 1, \ldots, N$. While for deep neural network classes S a complete theoretical foundation is still missing, the famous representer theorem provides a direct way to compute f if S is a subset of a reproducing kernel Hilbert space $\mathcal{H}$ [11]. However, the cost complexity of the underlying algorithm usually scales at least quadratically in N.

An easy and straightforward way to achieve linear cost complexity with respect to N is to employ grid-based discretizations of localized functions from $\mathcal{H}$. However, standard tensor grids can only be used up to dimension $d = 3$ because of the exponential dependence of the underlying computational costs with respect to d. This effect resembles the well-known *curse of dimensionality.* Using a sparse grid discretization, which relies on the boundedness of mixed derivatives up to a fixed order, this exponential dependence is mitigated significantly while the discretization error is almost of the same order as for tensor grids [2].

A further reduction of the costs in sparse grid regression can be achieved when space- and dimension-adaptive variants are employed. Here, the algorithm adapts in an a posteriori way to the underlying structure of the data [10]. This is particularly helpful if f only depends on $k < d$ variables and is (nearly) constant along the remaining $d - k$ directions. In this case k is called the *effective dimension* of f. Note that this is directly related to the concept of an analysis of variance (ANOVA) decomposition in statistics. Furthermore, there is a direct connection between the ANOVA decomposition and certain sparse grid discretizations, see also [5].

Instead of using an Euclidean coordinate system, it is common to use problem-dependent coordinates which simplify the description of the underlying problem: Any (sufficiently differentiable) bijection $\psi : \mathbb{R}^d \to \mathbb{R}^d$ describes a coordinate transformation. Then, if $f : \mathbb{R}^d \to \mathbb{R}$ approximates the data $(\boldsymbol{t}_i, x_i)$, the function $\bar{f} = f \circ \psi : \mathbb{R}^d \to \mathbb{R}$ approximates the transformed dataset $(\psi^{-1}(\boldsymbol{t}_i), x_i)_{i=1}^N$. The question is which bijections ψ yield a small effective dimension of the transformed data set in the ANOVA sense, are still sufficiently cheap to represent, and allow for a more efficient approximation of $\bar{f}$ than the one using $\psi = \mathrm{id}$.

In this work, we concentrate on problems of effective (truncation) dimension $k < d$ in the ANOVA sense. More specifically, for a problem with effective dimension k, we will focus on transformations from the

[*] Supported by CRC 1060 - The Mathematics of Emergent Effects funded by the Deutsche Forschungsgemeinschaft.

[†] Institute for Numerical Simulation, University of Bonn, Endenicher Allee 19b, 53115 Bonn, Germany, bohn/griebel/oettersh@ins.uni-bonn.de

[‡] Fraunhofer Institute for Algorithms and Scientific Computing SCAI, Schloss Birlinghoven, 53754 Sankt Augustin, Germany

Stiefel manifold $V_k(\mathbb{R}^d) \subset \mathbb{R}^{d\times k}$, which consists of all orthogonal k-frames in $\mathbb{R}^d$. Then, our goal is to find a $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ such that the computational costs of learning the dataset $(\boldsymbol{Q}^T\boldsymbol{t}_i, x_i)$ are as small as possible. However, we do not want to solve the regression problem $(\boldsymbol{Q}^T\boldsymbol{t}_i, x_i)_{i=1}^N$ for each candidate $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ involved in the optimization process. Therefore, we make a crude approximation to the true function f by a homogeneous d-variate polynomial $p \in \mathcal{P}_m^{(d)}$ of total degree less than m, which has relatively few degrees of freedom and is invariant under orthogonal transformations. This means that we can determine p before searching for the optimal $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ by minimizing the effective dimension of the low degree polynomial $p\circ\boldsymbol{Q}$ with respect to $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$. It is important to note that a crude approximation p to f is sufficient in our case since we are only interested in the rough behavior of the lower-order ANOVA terms of $f \circ \boldsymbol{Q}$ and not in f itself.

Overall, we aim to efficiently determine $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ such that $f \circ \boldsymbol{Q}$ can be well approximated by an adaptive sparse grid. The latter benefits from the low effective dimensionality of the transformed problem as it automatically detects important ANOVA terms and refines only along relevant coordinates.

There are similarities to several other established dimensionality reduction and data transformation algorithms. For instance, a linear preprocessing technique to solve multivariate integration problems has been used in [7, 8]. Maximizing gradients of the transformed initial function is also the main idea behind the active subspace method [3]. An active subspace approach based on polynomial surrogates for data-driven tasks can be found in [4]. One of the main differences to our proposed method is that the authors directly minimize the least-squares regression error of a linearly transformed polynomial. This leads to a coupled optimization problem in the polynomial p and the linear transformation $\boldsymbol{Q}$. In our case, however, we exploit the fact that our sparse grid discretizations directly benefit from transformations $\boldsymbol{Q}$ such that $p\circ\boldsymbol{Q}$ has a small effective ANOVA dimension. Therefore, we will fix p a priorly and then search for a $\boldsymbol{Q}$ which minimizes the effective dimension of $p\circ\boldsymbol{Q}$. Subsequently, we will learn a sparse grid function to approximate $f \circ \boldsymbol{Q}$. In this way, we rely on the polynomial surrogate p only to determine $\boldsymbol{Q}$. This makes the optimization with respect to $\boldsymbol{Q}$ much easier and still allows for efficient sparse grid discretizations of the underlying regression problem.

The remainder of this article is organized as follows: In Section 2, we introduce the classical ANOVA-decomposition and the concept of effective dimensionality. In Section 3, we discuss the reduction of the effective dimensionality using coordinate systems from $V_k(\mathbb{R}^d)$. In Section 4 we apply this method to machine learning by reducing the effective dimensionality of data sets and learning the transformed set with a space- and dimension-adaptive sparse grid. Section 5 contains numerical results for our approach to validate the benefit of the coordinate transformation. In Section 6 we give some concluding remarks.

2 Effective dimensionality of functions

In this section we recall the classical analysis-of-variance (ANOVA) decomposition and the concept of effective dimensionality. To this end, let $\Omega \subseteq \mathbb{R}$ be a fixed domain. For all subsets $\mathbf{u} \subseteq \mathcal{D} := \{1, 2, \ldots, d\}$, we define the $|\mathbf{u}|$-dimensional product domains $\Omega^{|\mathbf{u}|} \subseteq \mathbb{R}^{|\mathbf{u}|}$. In the following, we write $\mathbf{u}^c$ to denote $\mathcal{D} \setminus \mathbf{u}$. Let $\mathrm{d}\mu(\boldsymbol{x}) = \prod_{j=1}^d \mathrm{d}\mu_j(x_j)$ be a d-dimensional product of probability measures μ_j on the Borel-algebra of Ω. The associated measures on $\Omega^{|\mathbf{u}|}$ are given by $\mathrm{d}\mu_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}}) := \prod_{j\in\mathbf{u}} \mathrm{d}\mu_j(x_j)$, where $\boldsymbol{x}_{\mathbf{u}}$ denotes the $|\mathbf{u}|$-dimensional vector which contains those components of $\boldsymbol{x}$ with indices in $\mathbf{u}$. Let $\mathcal{T}^{(d)} := L_2(\Omega^d, \mu)$ be endowed with the inner product

$$(f, g)_\mu := \int_{\Omega^d} f(\boldsymbol{x})g(\boldsymbol{x})\ \mathrm{d}\mu(\boldsymbol{x})$$

and its induced norm $\|f\|_{2,\mu}^2 = \int_{\Omega^d} f(\boldsymbol{x})^2\ \mathrm{d}\mu(\boldsymbol{x})$. For $\mathbf{u} \subset \mathcal{D}$, the spaces $\mathcal{T}^{\mathbf{u}} := L_2(\Omega^{\mathbf{u}}, \mu_{\mathbf{u}})$ will be treated as subspaces of $\mathcal{T}^{(d)}$ by viewing their elements as d-variate functions that only depend on the variables $j \in \mathbf{u}$, i.e.

$$\mathcal{T}^{\mathbf{u}} = \{f \in \mathcal{T}^{(d)} : f(\boldsymbol{x}_{\mathbf{u}}, \boldsymbol{y}_{\mathbf{u}^c}) = f(\boldsymbol{x}_{\mathbf{u}}, \tilde{\boldsymbol{y}}_{\mathbf{u}^c})\ \forall\ \boldsymbol{y}_{\mathbf{u}^c}, \tilde{\boldsymbol{y}}_{\mathbf{u}^c}\}$$

The basic idea behind the ANOVA decomposition is to define projections $\mathcal{T}^{(d)} \to \mathcal{T}^{\mathbf{u}}$ which will then be employed to decompose a d-variate function $f \in \mathcal{T}^{(d)}$ into a sum of low-dimensional functions, i.e.

$$f(\boldsymbol{x}) = f_\emptyset + \sum_{i=1}^d f_{\{i\}}(x_i) + \sum_{\substack{i,j=1,\ldots,d\\ i<j}} f_{\{i,j\}}(x_i, x_j) + \ldots + f_{\mathcal{D}}(x_1, \ldots, x_d)$$

This sum will be abbreviated by $f(\boldsymbol{x}) = \sum_{\mathbf{u}\subseteq\mathcal{D}} f_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}})$.

2.1 The ANOVA decomposition.

We begin by defining the orthogonal projectors $P_{\mathbf{u}} : \mathcal{T}^{(d)} \to \mathcal{T}^{\mathbf{u}}$ via

$$\begin{aligned} P_{\mathbf{u}}(f)(\boldsymbol{x}_{\mathbf{u}}) &:= \int_{\Omega^{\mathbf{u}^c}} f(\boldsymbol{x})\ \mathrm{d}\mu_{\mathbf{u}^c}(\boldsymbol{x}_{\mathbf{u}^c}) && \text{for } \mathbf{u} \subsetneq \mathcal{D} \\ P_{\mathbf{u}}(f)(\boldsymbol{x}) &:= f(\boldsymbol{x}) && \text{for } \mathbf{u} = \mathcal{D}. \end{aligned}$$

The projections are orthogonal in $\mathcal{T}^{(d)}$ and hence give the $\mathcal{T}^{(d)}$-optimal low-dimensional approximation to $f \in$

$\mathcal{T}^{(d)}$ by functions from $\mathcal{T}^{\mathbf{u}}$. Now, let

$$f_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}}) := P_{\mathbf{u}}(f)(\boldsymbol{x}_{\mathbf{u}}) - \sum_{\mathbf{v}\subsetneq\mathbf{u}} f_{\mathbf{v}}(\boldsymbol{x}_{\mathbf{v}}) \tag{2.1}$$

for all $\boldsymbol{x}_{\mathbf{u}} \in \Omega^{\mathbf{u}}$. Then it holds

$$f(\boldsymbol{x}) = \sum_{\mathbf{u}\subseteq\mathcal{D}} f_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}}) \text{ and } (f_{\mathbf{u}}, f_{\mathbf{v}})_{\mu} = 0 \text{ for } \mathbf{u} \neq \mathbf{v}.$$

The 2^d summands $f_{\mathbf{u}}, \mathbf{u} \subseteq \mathcal{D}$, describe the dependence of f on the subset of variables contained in $\mathbf{u}$. Analogously to the definition of the variance of $f \in \mathcal{T}^{(d)}$, the variance of the ANOVA term $f_{\mathbf{u}}$ for a $\mathbf{u} \neq \emptyset$ is given by

$$\sigma^2_{\mathbf{u},\mu}(f) = \int_{\Omega^{\mathbf{u}}} f_{\mathbf{u}}^2 \,\mathrm{d}\mu_{\mathbf{u}} - \underbrace{\left(\int_{\Omega^{\mathbf{u}}} f_{\mathbf{u}} \,\mathrm{d}\mu_{\mathbf{u}}\right)^2}_{=0} = \int_{\Omega^{\mathbf{u}}} f_{\mathbf{u}}^2 \,\mathrm{d}\mu_{\mathbf{u}}.$$

Then, due to the orthogonality of the ANOVA-decomposition, the variance of f can be decomposed into the sum of the variances of all ANOVA terms, i.e.

$$\sigma^2_{\mu}(f) = \sum_{|\mathbf{u}|>0} \sigma^2_{\mathbf{u},\mu}.$$

We define the auxiliary quantity

$$\begin{aligned} D_{\mathbf{u}}(f) &:= \sum_{\mathbf{v}\subseteq\mathbf{u}} \sigma^2_{\mathbf{v},\mu}(f) \overset{(2.1)}{=} \int_{\Omega^{\mathbf{u}}} P_{\mathbf{u}}(f)^2 \,\mathrm{d}\mu_{\mathbf{u}} - f_{\emptyset}^2 \\ &= \int_{\Omega^{\mathbf{u}}} \left(\int_{\Omega^{\mathbf{u}^c}} f \,\mathrm{d}\mu_{\mathbf{u}^c}\right)^2 \mathrm{d}\mu_{\mathbf{u}} - f_{\emptyset}^2 \end{aligned}$$

and use it to determine $\sigma^2_{\mathbf{u},\mu}(f)$ recursively via

$$\sigma^2_{\mathbf{u},\mu}(f) = D_{\mathbf{u}}(f) - \sum_{\mathbf{v}\subsetneq\mathbf{u}} \sigma^2_{\mathbf{v},\mu}(f). \tag{2.2}$$

The value of $D_{\mathbf{u}}$ is given by the following Lemma, which we will exploit later. This result has been proven in Theorem 2 of [12].

Lemma 2.1. *For $\mathbf{u} \subseteq \mathcal{D}$ it holds*

$$D_{\mathbf{u}}(f) + f_{\emptyset}^2 = \int_{\Omega^{2d-|\mathbf{u}|}} f(\boldsymbol{x}_{\mathbf{u}}, \boldsymbol{x}_{\mathbf{u}^c})\, f(\boldsymbol{x}_{\mathbf{u}}, \boldsymbol{y}_{\mathbf{u}^c}) \,\mathrm{d}\mu_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}})\,\mathrm{d}\mu_{\mathbf{u}^c}(\boldsymbol{x}_{\mathbf{u}^c})\,\mathrm{d}\mu_{\mathbf{u}^c}(\boldsymbol{y}_{\mathbf{u}^c}).$$

The next proposition shows that the $\sigma_{\mathbf{u},\mu}(f)$ are invariant with respect to component-wise coordinate transformations.

Proposition 2.1. *Let $\Omega, \widehat{\Omega} \subseteq \mathbb{R}$ and let $\mu = \bigotimes_{j=1}^d \mu_j$ be a product measure on Ω^d. Moreover, let $\Phi : \widehat{\Omega}^d \to \Omega^d$ be defined by $\Phi(\boldsymbol{x}) := (\phi_1(x_1), \ldots, \phi_d(x_d))$, where each $\phi_j : \widehat{\Omega} \leftrightarrow \Omega$ is a diffeomorphism. Then it holds*

$$\sigma^2_{\mathbf{u},\mu}(f) = \sigma^2_{\mathbf{u},\mu\circ\Phi}(f \circ \Phi). \tag{2.3}$$

Proof. This is a direct consequence of the change of variables formula.

This result is of special interest if the component functions ϕ_j of the transformation are the inverses of the cumulative distribution functions of the μ_j.

Finally, in order to compare the dependence of functions on their respective ANOVA terms, we define the so-called sensitivity coefficients

$$s_{\mathbf{u}}(f) := \frac{1}{\sigma^2_{\mu}(f)} \sigma^2_{\mathbf{u},\mu}(f) \quad \text{for all } \mathbf{u} \subseteq \mathcal{D}.$$

These coefficients describe the relative importance of the coordinate directions in $\mathbf{u}$. Note that $\sum_{\mathbf{u}\subseteq\mathcal{D}} s_{\mathbf{u}} = 1$.

2.2 Notions of effective dimensionality. The term *effective dimensionality* is based on the insight that the ANOVA terms of higher cardinality contribute much less to the total variance than the lower-order terms for many application-driven problems and that methods for their solution can benefit from this property. In [9] the *mean dimension* is defined by weighting the variances $\sigma^2_{\mathbf{u}}$ of the single ANOVA terms $f_{\mathbf{u}}$ with their cardinalities $|\mathbf{u}|$, i.e. higher-order terms get penalized stronger than lower-order terms. Notions of effective dimensionality which are not based on the classical ANOVA-decomposition, but rather on the anchored ANOVA approach can be found in [7].

In this paper we will employ a *generalization* of the mean dimension. To this end, let $\nu_{\mathbf{u}} > 0$ be an arbitrary set of weights for all $\mathbf{u} \subseteq \mathcal{D}$. We define

$$d_{\boldsymbol{\nu}}(f) := \sum_{\mathbf{u}\subseteq\mathcal{D}} \nu_{\mathbf{u}}\, s_{\mathbf{u}}(f). \tag{2.4}$$

In this way, sensitivity coefficients in the ANOVA decomposition can be weighted differently, e.g. according to the number of variables present in each ANOVA term.

3 Minimizing the effective dimensionality

In the following, we aim to find a coordinate transformation $\psi : \Omega^d \to \mathbb{R}^d$ to reduce the generalized mean dimension (2.4) for a given $f : \mathbb{R}^d \to \mathbb{R}$. This directly leads to the minimization problem

$$\mathfrak{M}_f(\psi) := \sum_{\mathbf{u}\subseteq\mathcal{D}} \nu_{\mathbf{u}}\, s_{\mathbf{u}}(f \circ \psi) \longrightarrow \min_{\psi\in\Psi}!, \tag{3.5}$$

where $\nu_{\mathbf{u}} > 0$ are prescribed weights that should penalize higher-order terms and Ψ is a class of suitable diffeomorphisms. The hope is that there exists a ψ such that $f \circ \psi$ has a substantially smaller dependence on higher-order ANOVA terms than the original f did. At this point we should realize that the approximation of ψ also involves certain costs. Therefore, we need the class Ψ to be both powerful enough to reduce the effective dimension and small enough to rely only on a few degrees of freedom so that we do not just shift the costs from the approximation of the outer function $f \circ \psi$ to the inner function ψ.

3.1 Equivalent maximization problem. Penalizing higher-order ANOVA terms makes the functional (3.5) expensive or even impossible to evaluate as these terms contribute the most to its value and their evaluation is based on the evaluation of all lower-order terms for $\mathbf{v} \subsetneq \mathbf{u}$. Therefore, we are looking for a reformulation of the minimization task (3.5) which circumvents this problem. To this end, note that

$$\begin{aligned}\mathfrak{M}_f(\psi) &= \sum_{\mathbf{u}\subseteq\mathcal{D}} \nu_{\mathbf{u}}\, s_{\mathbf{u}}(f\circ\psi)\\ &= \sum_{|\mathbf{u}|<d} \nu_{\mathbf{u}}\, s_{\mathbf{u}}(f\circ\psi) + \nu_{\mathcal{D}}\, s_{\mathcal{D}}(f\circ\psi)\\ &= \sum_{|\mathbf{u}|<d} (\nu_{\mathbf{u}} - \nu_{\mathcal{D}})\; s_{\mathbf{u}}(f\circ\psi) + \nu_{\mathcal{D}}.\end{aligned}$$

Therefore, a minimizer of $\mathfrak{M}_f(\psi)$ is also a maximizer of $-\frac{1}{\nu_{\mathcal{D}}}\mathfrak{M}_f(\psi)$ and vice versa and we obtain the following equivalent maximization problem

$$(3.6)\quad \hat{\mathfrak{M}}_f(\psi) := \sum_{\mathbf{u}\subsetneq\mathcal{D}} \left(1 - \frac{\nu_{\mathbf{u}}}{\nu_{\mathcal{D}}}\right) s_{\mathbf{u}}(f\circ\psi) \longrightarrow \max_{\psi\in\Psi}!.$$

The main advantage of considering (3.6) instead of (3.5) is that we can now omit sets $\mathbf{u}$ with large $|\mathbf{u}|$ in (3.6) and focus the optimization task to sets with small $|\mathbf{u}|$, i.e. lower-dimensional terms only.

3.2 Choice of the weights. Let $1 \leq k < d$. In the remainder of the article, we will use

$$\nu_{\mathbf{u}} := \begin{cases} 1 - \exp\left(-\max\{j \in \mathbf{u}\}\right) & \text{if } \mathbf{u} \subseteq \{1,\ldots,k\},\\ 1 & \text{else.}\end{cases}$$

Now we only need to evaluate ANOVA terms of $f \circ \psi$ corresponding to subsets of the first k variables since

$$\hat{\mathfrak{M}}_f(\psi) = \sum_{\mathbf{u}\subseteq\{1,\ldots,k\}} \exp\left(-\max\{j\in\mathbf{u}\}\right) s_{\mathbf{u}}(f\circ\psi).$$

In this sense, we try to find a ψ such that ideally all (or at least most) of the variance of $f \circ \psi$ resides in these terms. Such a function is said to have truncation dimension k in the ANOVA sense.

For the subclass of orthogonal projections Ψ, which we will focus on in this paper, and for a measure μ which is invariant under orthogonal transformations, such as the Lebesgue or the Gaussian measure on $\mathbb{R}^d$, we obtain $\sigma^2_\mu(f) = \sigma^2_{\mu\circ\psi}(f\circ\psi)$ for all $\psi \in \Psi$. Therefore, we can simply omit $\sigma^2_\mu(f\circ\psi)$ in the maximization functional in this case and we just maximize

$$(3.7)\quad \hat{\mathfrak{M}}_f(\psi) := \sum_{\mathbf{u}\subseteq\{1,\ldots,k\}} \exp\left(-\max\{j\in\mathbf{u}\}\right) \sigma^2_{\mathbf{u},\mu}(f\circ\psi),$$

which we can evaluate by using (2.2) and Lemma 2.1.

3.3 Orthogonal transformations. Due to our specific choice of weights, we can actually restrict ourselves to transformations $\phi : \Omega^k \subseteq \mathbb{R}^k \to \mathbb{R}^d$ instead of having to look for maps with a domain in $\mathbb{R}^d$. Therefore, let

$$V_k(\mathbb{R}^d) := \left\{\boldsymbol{Q} \in \mathbb{R}^{d\times k} \mid \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}\right\},$$

be our class of valid transformations. Here, the rows of $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ represent an orthogonal k-frame in $\mathbb{R}^d$. This class is actually a submanifold of $\mathbb{R}^{d\times k}$ and it is known as the so-called *Stiefel manifold.*

As we see, maximizing $\hat{\mathfrak{M}}_f(\boldsymbol{Q})$ over $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ is a highly nonlinear task with possibly nonunique maximizers. The existence of maximizers can be guaranteed for continuous functions f since $\hat{\mathfrak{M}}_f$ is a continuous functional in that case and $V_k(\mathbb{R}^d) \subset \mathbb{R}^{d\times k}$ is compact. As mentioned earlier, we will substitute f by a polynomial surrogate for the actual optimization for which we can, therefore, guarantee the existence of maximizers.

3.4 Polynomials as invariant basis. The largest part of the costs in evaluating $\hat{\mathfrak{M}}_f(\boldsymbol{Q})$ is the evaluation of each $\sigma^2_{\mathbf{u},\mu}(f\circ\boldsymbol{Q})$, which requires the approximation of high-dimensional integrals. Therefore, we discretize f in a basis which allows to compute $\sigma^2_{\mathbf{u},\mu}(f\circ\boldsymbol{Q})$ analytically for $\Omega = \mathbb{R}$ and which is closed under orthogonal transformations. To this end, we employ a total degree polynomial space with a homogeneous basis in $\mathbb{R}^k$, i.e. we take the basis set

$$\mathcal{B}^{(k)}_m := \{\boldsymbol{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1}\ldots x_k^{\alpha_k} \mid |\boldsymbol{\alpha}|_1 \leq m\}$$

for some $m \in \mathbb{N}$. Note that $K := |\mathcal{B}^{(k)}_m| = \binom{k+m}{k}$.

LEMMA 3.1. *The basis $\mathcal{B}^{(d)}_m$ spans the total degree polynomial space $\mathcal{P}^{(d)}_m$ on $\mathbb{R}^d$ which is invariant with respect to all orthogonal transformations $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$, i.e.*

$$\phi\circ\boldsymbol{Q} \in \operatorname{span}\{\mathcal{B}^{(k)}_m\} \quad \forall\ \phi \in \mathcal{B}^{(d)}_m, \boldsymbol{Q} \in V_k(\mathbb{R}^d).$$

Proof. Let $\omega_{\boldsymbol{\alpha}}(\boldsymbol{x}) := \boldsymbol{x}^{\boldsymbol{\alpha}}$. Using the multinomial theorem with $\beta \in \mathbb{N}_0^k$, we obtain

$$\begin{aligned} \omega_{\boldsymbol{\alpha}} \circ \boldsymbol{Q}(\boldsymbol{x}) &= \prod_{i=1}^{d} \left(\sum_{j=1}^{k} Q_{ij} x_j \right)^{\alpha_i} \\ &= \prod_{i=1}^{d} \underbrace{\sum_{|\boldsymbol{\beta}|_1 = \alpha_i} \frac{\alpha_i!}{\beta_1! \cdots \beta_k!} \prod_{j=1}^{k} \left(Q_{ij} x_j\right)^{\beta_j}}_{\in \operatorname{span}\{\mathcal{B}_{\alpha_i}^{(k)}\}} . \end{aligned}$$

Since $\deg(P \cdot S) = \deg(P) + \deg(S)$ holds for polynomials P and S and because of $|\boldsymbol{\alpha}|_1 \leq m$, the claim follows.

Lemma 3.1 shows that we just need to evaluate $D_{\mathbf{u}}(p)$ for polynomials $\tilde{p} \in \mathcal{B}_m^{(k)}$ regardless of the transformation $Q \in V_k(\mathbb{R}^d)$ when taking a polynomial surrogate $p \in \operatorname{span}(\mathcal{B}_m^{(d)})$ of f. Next, let us define $\mathcal{I}_M(\boldsymbol{\alpha}) := \int_{\Omega^k} \boldsymbol{x}^{\boldsymbol{\alpha}} \,\mathrm{d}\mu(\boldsymbol{x})$ with the restriction $\mathcal{I}_M(\boldsymbol{\alpha}_{\mathbf{u}}) := \int_{\Omega^{|\mathbf{u}|}} \boldsymbol{x}_{\mathbf{u}}^{\boldsymbol{\alpha}_{\mathbf{u}}} \,\mathrm{d}\mu_{\mathbf{u}}(\boldsymbol{x}_{\mathbf{u}})$ to the directions that are contained in $\mathbf{u}$. Then, we have the following result.

Lemma 3.2. *Let $p := \sum_{|\boldsymbol{\alpha}|_1 \leq m} C_{\boldsymbol{\alpha}} \boldsymbol{x}^{\boldsymbol{\alpha}} \in \mathcal{B}_m^{(k)}$. Then, $\mathcal{A} := D_{\mathbf{u}}(p) + f_{\emptyset}^2$ from (2.2) fulfills*

$$\mathcal{A} = \sum_{\substack{|\boldsymbol{\alpha}|_1 \leq m \\ |\boldsymbol{\beta}|_1 \leq m}} C_{\boldsymbol{\alpha}} C_{\boldsymbol{\beta}} \, \mathcal{I}_M(\boldsymbol{\alpha}_{\mathbf{u}} + \boldsymbol{\beta}_{\mathbf{u}}) \cdot \mathcal{I}_M(\boldsymbol{\alpha}_{\mathbf{u}^c}) \cdot \mathcal{I}_M(\boldsymbol{\beta}_{\mathbf{u}^c}).$$

Proof. Using Lemma 2.1 we obtain

$$\begin{aligned} \mathcal{A} &= \int_{\Omega^{2k - |\mathbf{u}|}} p(\boldsymbol{x}_{\mathbf{u}}, \boldsymbol{x}_{\mathbf{u}^c})\, p(\boldsymbol{x}_{\mathbf{u}}, \boldsymbol{y}_{\mathbf{u}^c}) \mathrm{d}\mu(\boldsymbol{x}) \mathrm{d}\mu_{\mathbf{u}^c}(\boldsymbol{y}_{\mathbf{u}^c}) \\ &= \sum_{\substack{|\boldsymbol{\alpha}|_1 \leq m \\ |\boldsymbol{\beta}|_1 \leq m}} C_{\boldsymbol{\alpha}} C_{\boldsymbol{\beta}} \int_{\Omega^{2k - |\mathbf{u}|}} \boldsymbol{x}_{\mathbf{u}}^{\boldsymbol{\alpha}_{\mathbf{u}} + \boldsymbol{\beta}_{\mathbf{u}}} \, \boldsymbol{x}_{\mathbf{u}^c}^{\boldsymbol{\alpha}_{\mathbf{u}^c}} \, \boldsymbol{y}_{\mathbf{u}^c}^{\boldsymbol{\beta}_{\mathbf{u}^c}} \mathrm{d}\mu(\boldsymbol{x}) \mathrm{d}\mu_{\mathbf{u}^c}(\boldsymbol{y}_{\mathbf{u}^c}) \end{aligned}$$

and applying Fubini's theorem finishes the proof.

In the case $\Omega = \mathbb{R}$ and when μ is the standard Gaussian measure we can compute $\mathcal{I}_M(\boldsymbol{\alpha})$ analytically.

Lemma 3.3. *The expected value of $\boldsymbol{x}^{\boldsymbol{\alpha}}$ with $|\boldsymbol{\alpha}|_1 = m$ for $\boldsymbol{\alpha} \in \mathbb{N}_0^k$ with respect to the Gaussian measure μ is*

$$\mathbb{E}_{\mu}\left[\boldsymbol{x}^{\boldsymbol{\alpha}}\right] = \begin{cases} \prod_{i=1}^{k} (\alpha_i - 1)!! & \textit{if all } \alpha_i \textit{ even,} \\ 0 & \textit{else,} \end{cases}$$

where $n!! := n \cdot (n-2) \cdot (n-4) \cdot \ldots \cdot 1$.

Proof. We have $\mathbb{E}_{\mu}\left[\boldsymbol{x}^{\boldsymbol{\alpha}}\right] = \prod_{i=1}^{k} \mathbb{E}_{\mu_i}(x^{\alpha_i})$ and since the moments of the standard normal distribution are

$$\mathbb{E}(x^q) = \begin{cases} (q-1)!! & q \text{ even,} \\ 0 & q \text{ odd,} \end{cases}$$

the claim follows.

With the help of Lemmata 3.1, 3.2 and 3.3, we can compute $f_{\emptyset}$ and solve all the integrals involved in the computation of $\sigma_{\mathbf{u},\mu}^2(p \circ \boldsymbol{Q})$ for all $\mathbf{u} \subseteq \mathcal{D}$, $p \in \mathcal{P}_m^{(d)}$ and $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ analytically if μ is taken as the standard Gaussian measure on the whole space $\mathbb{R}^k$. Therefore, we know how to evaluate $\hat{\mathfrak{M}}_p(\boldsymbol{Q})$ in this case.

3.5 A manifold CG algorithm on $V_k(\mathbb{R}^d)$. To numerically solve the optimization problem, we propose a descent algorithm on the submanifold $V_k(\mathbb{R}^d) \subset \mathbb{R}^{d \times k}$. Algorithm 1 briefly sketches a conjugate gradient approach for this specific matrix manifold. Before we start the first iteration, we set the solution to the zero vector and choose a random direction. For more details on the algorithm and the theory behind optimization on matrix manifolds, see [1].

Algorithm 1 One step of the matrix manifold CG algorithm on $V_k(\mathbb{R}^d)$ to determine a maximizer of $\hat{\mathfrak{M}}_p$.

```
Input: Iterate $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$, direction $\boldsymbol{M} \in \mathbb{R}^{d \times k}$.
Output: New iterate $\bar{\boldsymbol{Q}} \in V_k(\mathbb{R}^d)$, new direction $\bar{\boldsymbol{M}} \in \mathbb{R}^{d \times k}$.
Line search for $\delta > 0$ along $\boldsymbol{Q} + \delta \boldsymbol{M}$.
QR decomposition: $QR \leftarrow \boldsymbol{Q} + \delta \boldsymbol{M}$ (“retraction”).
Define new iterate: $\bar{\boldsymbol{Q}} \leftarrow Q$.
Polak-Ribiere: $\beta \leftarrow \frac{\nabla \hat{\mathfrak{M}}_p(\bar{\boldsymbol{Q}})^T \left(\nabla \hat{\mathfrak{M}}_p(\bar{\boldsymbol{Q}}) - \nabla \hat{\mathfrak{M}}_p(\boldsymbol{Q})\right)}{\nabla \hat{\mathfrak{M}}_p(\boldsymbol{Q})^T \nabla \hat{\mathfrak{M}}_p(\boldsymbol{Q})}$.
$\boldsymbol{M}^* \leftarrow (\boldsymbol{I} - \bar{\boldsymbol{Q}}\bar{\boldsymbol{Q}}^T)\boldsymbol{M} + \frac{1}{2}\bar{\boldsymbol{Q}}(\bar{\boldsymbol{Q}}^T \boldsymbol{M} - \boldsymbol{M}^T \bar{\boldsymbol{Q}})$ (“parallel transport”).
Define new direction: $\bar{\boldsymbol{M}} \leftarrow -\nabla \hat{\mathfrak{M}}_p(\bar{\boldsymbol{Q}}) + \beta \boldsymbol{M}^*$.
```

As we see, we need matrix-matrix multiplications, a QR decomposition of a $d \times k$ matrix and a gradient computation of $\hat{\mathfrak{M}}_p$, which has to be understood as calculating the usual gradient in $\mathbb{R}^{d \times k}$. We do the latter by using first order forward finite differences.

Proposition 3.1. *Let $\mu = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ be the standard Gaussian measure. The total number of operations to perform one CG iteration of Algorithm 1 is bounded by*

$$\begin{aligned} &\mathcal{O}\left(kd \cdot \left(d \binom{d+m}{d} + \left(k \binom{k+m}{k} \right)^2 \right) \right) \\ &= \mathcal{O}\left(kd^{m+2} + k^{2m+3} d \right). \end{aligned}$$

Proof. Since computing the QR decomposition of a $d \times k$ matrix costs $\mathcal{O}\left(k^2 \cdot d\right)$ operations and the matrix products in the parallel transport step cost $\mathcal{O}\left(k^2 \cdot d + k \cdot d^2\right) = \mathcal{O}\left(k \cdot d^2\right)$ operations, we directly see that the most expensive part is computing the derivative $\nabla \hat{\mathfrak{M}}_p(\bar{\boldsymbol{Q}})$ by finite differences, for which we need to perform $2kd$ evaluations of $\hat{\mathfrak{M}}_p$. To this end, let $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ be given and assume that we want to compute $\hat{\mathfrak{M}}_p(\boldsymbol{Q})$. We first need to calculate the coefficients of $p \circ \boldsymbol{Q}$ in the basis $\mathcal{B}_m^{(k)}$ as done in the proof of Lemma 3.1. Here, we store the intermediate values $s_i := \sum_{j=1}^k Q_{ij} x_j$ for all $i = 1, \ldots, d$ and subsequently compute $\sum_{|\boldsymbol{\alpha}|_1 \leq m} c_{\boldsymbol{\alpha}} \prod_{i=1}^d s_i^{\alpha_i}$ for the coefficients $c_{\boldsymbol{\alpha}}$ of p. This costs $\mathcal{O}(d \cdot \binom{d+m}{d})$ operations since there are $\binom{d+m}{d}$ monomials. Next, we evaluate $\hat{\mathfrak{M}}_p$ at $\boldsymbol{Q}$. To this end, note that the computation of $D_{\mathbf{u}}(p \circ \boldsymbol{Q})$ - with given coefficients for $p \circ \boldsymbol{Q}$ - takes $\mathcal{O}(\binom{k+m}{k} \cdot \binom{k+m}{k} \cdot k)$ operations as proven in Lemma 3.2 and Lemma 3.3. It remains to show that we can evaluate $\hat{\mathfrak{M}}_p(\boldsymbol{Q})$ by using only $\mathcal{O}(k)$ different sets $\mathbf{u} \subset \{1, \ldots, k\}$ and their corresponding $D_{\mathbf{u}}(p \circ \boldsymbol{Q})$. Indeed, let $\hat{\nu}_{\mathbf{u}} := \exp(-\max\{j \in \mathbf{u}\})$ and let $[i] := \{1, \ldots, i\}$. According to (3.7), we actually need to compute

$$\begin{aligned}
\hat{\mathfrak{M}}_p(\boldsymbol{Q}) &= \sum_{\mathbf{u} \subseteq [k]} \hat{\nu}_{\mathbf{u}} \sigma^2_{\mathbf{u},\mu}(f \circ \psi) \\
&= \sum_{\mathbf{u} \subseteq [k-1]} \hat{\nu}_{\mathbf{u}} \sigma^2_{\mathbf{u},\mu}(f \circ \psi) \\
&\quad + \exp(-k) \cdot \big(D_{[k]}(p \circ \boldsymbol{Q}) - D_{[k-1]}(p \circ \boldsymbol{Q})\big) \\
&= \ldots \\
&= \sum_{i=1}^{k} \exp(-i) \cdot \big(D_{[i]}(p \circ \boldsymbol{Q}) - D_{[i-1]}(p \circ \boldsymbol{Q})\big),
\end{aligned}$$

where we set $D_\emptyset := 0$. This can be done with $\mathcal{O}(k)$ evaluations of different $D_{\mathbf{u}}$, which completes the proof.

Finally note that a similar approach with a Newton-type manifold optimizer has been introduced in [4]. However, there the polynomial surrogate p for f and the transformation $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ are optimized at the same time and the least squares loss is considered. In our case, the polynomial surrogate just serves to coarsely represent the ANOVA structure of f. Since we first want to obtain a cost-efficient representation of $f \circ \boldsymbol{Q} \approx p \circ \boldsymbol{Q}$ before actually solving the underlying transformed least-squares problem on a sparse grid, we take the generalized mean dimension $\hat{\mathfrak{M}}_f$ as a complexity indicator to optimize with respect to $\boldsymbol{Q}$. Subsequently, we solve a least squares problem in an optimized sparse grid space as described in the next section.

4 Application to regression with sparse grids

In this section, we briefly recapitulate least squares regression on sparse grids, see e.g. [6, 10], and discuss a space- and dimension-adaptive variant based on the concepts of [5, 10]. Then, to reduce the computational costs of the adaptive regression algorithm, we suggest a preprocessing step based on the preceding section.

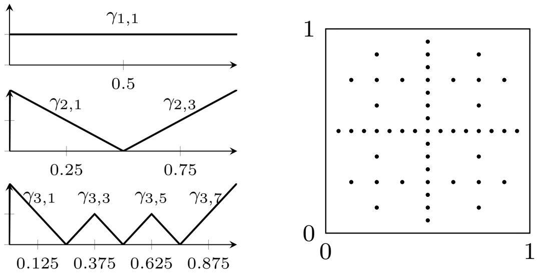

Fig. 1: Hierarchical basis functions up to $l = 3$ (left) and regular $2d$ sparse grid of level $\ell = 4$ (right).

4.1 Multivariate regression on sparse grids

Let $\boldsymbol{z} := \{(\boldsymbol{t}_i, x_i) \in T \times \mathbb{R} \mid i = 1, \ldots, N\}$ be N given samples, where $T := [0,1]^d$. If a different domain is used, the data has to be rescaled appropriately. Generally, least squares regression determines a minimizer to

$$\text{(4.8)} \qquad \min_{f \in S} \mathcal{E}_{\boldsymbol{z}} \quad \text{with} \quad \mathcal{E}_{\boldsymbol{z}}(f) := \frac{1}{N} \sum_{i=1}^{N} (x_i - f(\boldsymbol{t}_i))^2$$

over some set of functions S. If S is a reproducing kernel Hilbert space the famous representer theorem states that the solution can be determined by solving a (usually dense) $N \times N$ system of linear equations [11]. Naively, this would need $\mathcal{O}(N^3)$ floating point operations. While some algorithms solve these kernel systems approximately, they still scale worse than linearly in N in general. Therefore, we will employ a space S of sparse grid functions instead, which naturally leads to an algorithm that scales linearly in N and circumvents the curse of dimensionality of standard tensor grid approaches [2]. Several variants of such sparse grid least squares algorithms have been very successfully employed for different regression tasks, see e.g. [6, 10].

We shortly recall the sparse grid discretization based on the modified linear basis from [10]. Let

$$\Phi(t) := \max(1 - |t|, 0) \quad \text{and} \quad \Phi_{l,i}(t) := \Phi(2^l \cdot t - i)|_{[0,1]}$$

for $l, i \in \mathbb{N}_+$. To construct the modified hierarchical linear basis let $I_l := \{i \in \mathbb{N}_+ \mid 1 \leq i \leq 2^l - 1,\ i \text{ odd}\}$. For $l = 1$ we set $\gamma_{1,1} := 1$. For $l \geq 2$ we define $\gamma_{l,i} := \Phi_{l,i}$ for $i \in I_l \setminus \{1, 2^l - 1\}$ and $\gamma_{l,1}(t) := \max\left(2 - 2^l t, 0\right)|_{[0,1]}$ and $\gamma_{l,2^l-1}(t) := \gamma_{l,1}(1-t)$, see also Figure 1(left).

The d-variate basis functions are then built via the tensor product construction

$$\gamma_{\mathbf{l},\mathbf{i}}(\boldsymbol{t}) := \prod_{j=1}^{d} \gamma_{l_j, i_j}(t_j),$$

where $\mathbf{l} = (l_1, \ldots, l_d) \in \mathbb{N}_+^d$ is the multivariate level and $\mathbf{i} = (i_1, \ldots, i_d) \in \mathbb{N}_+^d$ denotes the multivariate position index. Let $\mathbf{I}_\mathbf{l} := \otimes_{j=1}^{d} I_{l_j}$. Then, $W_\mathbf{l} := \operatorname{span}\{\gamma_{\mathbf{l},\mathbf{i}} \mid \mathbf{i} \in \mathbf{I}_\mathbf{l}\}$ denotes the so-called hierarchical increment space of level $\mathbf{l}$. We now define the (regular) sparse grid space of level $\ell > 0$ by

$$V^\ell := \bigoplus_{\mathbf{k}\in\mathbb{N}_+^d, |\mathbf{k}|_1 \le \ell+d-1} W_\mathbf{k}. \tag{4.9}$$

Instead of $2^{\ell d}$ degrees of freedom as in the full grid case, the sparse grid space only contains $M := \dim(V^l) = \mathcal{O}(2^\ell \ell^{d-1})$ basis functions. A $2d$ sparse grid, i.e. the centers of the supports of all basis function of V^ℓ, can be found in Figure 1(right). For more details on sparse grids and a thorough comparison to full grids regarding cost complexity and approximation rates we refer to [2].

Representing $f \in V^\ell$ in the hierarchical basis yields

$$f(\boldsymbol{t}) = \sum_{|\mathbf{k}|_1 \le \ell+d-1} \sum_{\mathbf{i}\in\mathbf{I}_\mathbf{k}} \beta_{\mathbf{k},\mathbf{i}} \gamma_{\mathbf{k},\mathbf{i}}(\boldsymbol{t}).$$

We now consider the Tikhonov-regularized version

$$\min_{f\in S} \frac{1}{n} \sum_{i=1}^{n} (x_i - f(\boldsymbol{t}_i))^2 + \lambda \|\vec{\boldsymbol{\beta}}\|_2^2 \tag{4.10}$$

of the least squares problem (4.8) for $S = V^\ell$. Then, the coefficient vector $\vec{\boldsymbol{\beta}}$ is given by

$$\left(\boldsymbol{B}^T\boldsymbol{B} + \lambda \boldsymbol{I}\right)\vec{\boldsymbol{\beta}} = \boldsymbol{B}^T\vec{\boldsymbol{x}}, \tag{4.11}$$

where $\boldsymbol{B} \in \mathbb{R}^{N\times M}$ with entries $\boldsymbol{B}_{i,(\mathbf{l},\mathbf{j})} = \gamma_{\mathbf{l},\mathbf{j}}(\boldsymbol{t}_i)$ and $\vec{\boldsymbol{x}} = (x_1, \ldots, x_N)^T$. We employ a conjugate gradient solver to obtain the solution. For details on the linear equation system and the fast numerical treatment in the sparse grid case we refer to [5, 10].

4.2 Adaptive sparse grids. If certain spatial directions or regions are more important than others, e.g. when the solution of (4.10) varies strongly in one part of the domain but is almost constant in others, it is reasonable to adjust the underlying discretization to this behavior. To this end, space- and dimension-adaptive sparse grids can be employed [5, 10]. They adapt according to an error indicator which determines where the grid will be refined. Here, we use a combination

$$\epsilon_{\mathbf{l},\mathbf{i}} := \beta_{\mathbf{l},\mathbf{i}} \sum_{j=1}^{N} \gamma_{\mathbf{l},\mathbf{i}}(\boldsymbol{t}_j) \cdot \left(f(\boldsymbol{t}_j) - x_j\right)^2,$$

between the coefficients and the least-squares error. This serves to indicate how much $\gamma_{\mathbf{l},\mathbf{i}}$ contributes to the least squares error in the actual discretization.

The actual adaptive algorithm starts with $V = V^\ell$ for small ℓ and solves (4.10) to obtain the solution $f \in V$. Then, an initial compression step is performed, i.e. we mark all those basis functions from V for which $\epsilon_{\mathbf{l},\mathbf{i}}$ is smaller than a fixed threshold. Subsequently, all marked basis functions are removed from V. However, due to the hierarchical structure, we do not remove $\gamma_{\mathbf{l},\mathbf{i}}$ if one of its successors, i.e. a basis function whose support is a subset of the support of $\gamma_{\mathbf{l},\mathbf{i}}$, is not marked. Finally, we run a series of refinement steps, which consist of solving (4.10) over V, marking the $\mathcal{L} > 0$ refinable[1] basis functions in V with the largest value of $\epsilon_{\mathbf{l},\mathbf{i}}$ and then refining the marked functions. In this paper, we consider two different kinds of refinement: The first one, referred to as "standard" refinement, inserts all $2d$ children of each marked function. The second one, referred to as "ANOVA" refinement, only inserts children in those directions k, for which $l_k > 1$. This ensures that f remains constant in directions which the compression step has deemed to be irrelevant. The complete space- and dimension-adaptive procedure is described in Algorithm 2. For more details on adaptivity, its relation to the anchored ANOVA decomposition and fast sparse grid traversal algorithms we refer to [5].

Algorithm 2 The adaptive sparse grid algorithm

```
Input: $l \in \mathbb{N}$, Threshold $t > 0$, $\mathcal{L} \in \mathbb{N}$, numIt $\in \mathbb{N}$.
Output: Adaptive sparse grid space $V$.

Solve (4.10) over $V := V^l$ and compress($V$, $t$).
for $i = l \ldots$ numIt do
    Solve (4.10) over $V$ and refine($V$, $\mathcal{L}$).
end for
```

4.3 The final preprocessing method. Let $\boldsymbol{\alpha}_1 := \mathbf{0}$ and let $\boldsymbol{\alpha}_i, i = 2, \ldots, \bar{K} = \binom{d+m}{d}$ be an arbitrary enumeration of all indices $|\boldsymbol{\alpha}|_1 \le m$ corresponding to the basis set $\mathcal{B}_m^{(d)}$ of polynomials with total degree less than m. The final ingredient to our overall algorithm is solving the unregularized least squares problem (4.8) over $\operatorname{span}(\mathcal{B}_m^{(d)})$ to build the polynomial surrogate $p = \sum_{i=1}^{\bar{K}} c_{\boldsymbol{\alpha}_i} \boldsymbol{t}^{\boldsymbol{\alpha}_i}$. Its coefficients are the minimizers of

$$\min_{\vec{\boldsymbol{w}}\in\mathbb{R}^{\bar{K}}} \|\boldsymbol{A}\vec{\boldsymbol{w}} - \vec{\boldsymbol{x}}\|_2 = \min_{\vec{\boldsymbol{w}}\in\mathbb{R}^{\bar{K}}} \|\boldsymbol{W}\vec{\boldsymbol{w}} - \boldsymbol{V}^T\vec{\boldsymbol{x}}\|_2 \tag{4.12}$$

and can be determined by backsubstitution after computing a QR decomposition of the Vandermonde matrix

$$\boldsymbol{V}\boldsymbol{W} = \boldsymbol{A} := \begin{pmatrix} 1 & \boldsymbol{t}_1^{\boldsymbol{\alpha}_2} & \ldots & \boldsymbol{t}_1^{\boldsymbol{\alpha}_{\bar{K}}} \\ \vdots & \vdots & & \vdots \\ 1 & \boldsymbol{t}_N^{\boldsymbol{\alpha}_2} & \ldots & \boldsymbol{t}_N^{\boldsymbol{\alpha}_{\bar{K}}} \end{pmatrix}.$$

[1] We call a basis function refinable if not all of its children are already included in the grid. By children of $\gamma_{\mathbf{l},\mathbf{i}}$ we mean all successors on levels $\mathbf{l} + \mathbf{e}_k$, where $\mathbf{e}_k$ denotes the k-th unit vector.

Now, let $\mathcal{C}$ be the cumulative distribution function of the k-variate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, which we apply to rescale the data to $[0,1]^k$ for the sparse grid algorithm. The final optimally rotated, adaptive sparse grid least-squares method is presented in Algorithm 3. If the distribution of the input data is known to be non-Gaussian, it might be more sensible to use a different transformation for the rescaling onto $[0,1]^k$.

Algorithm 3 Optimally rotated, adaptive sparse grid least-squares algorithm

```
Input: Initial data $\boldsymbol{z} = \{(\boldsymbol{t}_i, x_i) \mid i = 1, \dots, N\}$.
Output: A $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ and a sparse grid function $f : [0,1]^k \to \mathbb{R}$ such that $f \circ \mathcal{C} \circ \boldsymbol{Q}^T$ approximates $\boldsymbol{z}$.
Determine $p$ via (4.12).
Determine $\boldsymbol{Q}$ with Algorithm 1.
Transform data to $\tilde{\boldsymbol{z}} = (\mathcal{C}(\boldsymbol{Q}^T \boldsymbol{t}_i), x_i)_{i=1}^N$.
Compute $f : [0,1]^k \to \mathbb{R}$ with Algorithm 2 on $\tilde{\boldsymbol{z}}$.
```

5 Numerical results

For our computations, we employ the SG++ sparse grid library [10] and choose the following parameters for all experiments: truncation parameter $k = \min(d, 3)$, total degree $m = 3$, compression threshold $t = 0.1$, number of points to refine $\mathcal{L} = 10$, initial grid level $l = 3$. The CG algorithm for solving (4.11) is iterated until the norm of the residual has decreased by a factor of 10^{-12}. To measure our performance, we use the normalized RMSE

$$\text{NRMSE} := \sqrt{\frac{\sum_i \left(f\left(\mathcal{C}\left(\boldsymbol{Q}^T(\tilde{\boldsymbol{t}}_i)\right)\right) - \tilde{x}_i\right)^2}{\sum_i \tilde{x}_i^2}},$$

where $(\tilde{\boldsymbol{t}}_i, \tilde{x}_i)$ is some test data. We compare this value to the NRMSE of Algorithm 2 on the untransformed data. Note that the runtimes of Algorithm 3 were (often magnitudes) smaller than the runtimes of Algorithm 2 on the untransformed data set for all of our experiments.

5.1 Two-dimensional ridge function. We draw $N = 10^5$ i.i.d. $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distributed points $\boldsymbol{t}_i \in \mathbb{R}^2$ and choose the ridge function $x_i = \tanh([\boldsymbol{t}_i]_1 + [\boldsymbol{t}_i]_2) + \varepsilon_i$, i.e. we evaluate tanh on the sum of the coordinates of each data vector and add i.i.d. white noise $\varepsilon_i \sim \mathcal{N}(0, 10^{-8})$. We also create a test data set $(\tilde{\boldsymbol{t}}_i, \tilde{x}_i)$ of size N with the same distribution but without the noise. Since N is significantly larger than the sparse grid sizes we use, we set $\lambda = 0$. The refinement process is iterated until the number of grid points exceeds 500. The resulting errors for each refinement after the initial compression and the employed sparse grids are illustrated in Figure 2 for both Algorithm 3 and Algorithm 2 on original data.

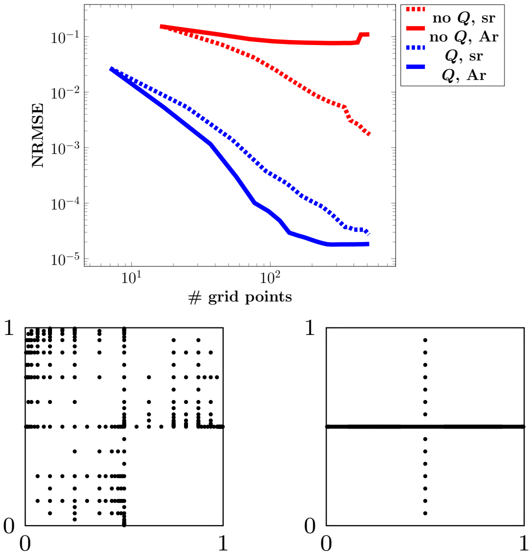

Fig. 2: NRMSE for 2d tanh ridge function (top). Ar = ANOVA refinement, sr = standard refinement. First 200 sparse grid points inserted by Ar for untransformed data (bottom left) and transformed data (bottom right).

As we observe, Algorithm 3 achieves an NRMSE, which is several magnitudes smaller than the NRMSE of the adaptive sparse grid algorithm on the untransformed data for both ANOVA and standard refinement. Obviously, the ANOVA refinement is too restrictive in the untransformed case and only standard refinement seems to converge. The remarkable performance of Algorithm 3 is obvious as the specific ridge function example has a rotated one-dimensional structure, which the preprocessing is able to pick up. This can be seen in the grid in Figure 2(right), where most points are spent along the horizontal line in the middle of the domain.

5.2 Five-dimensional sum of ridge functions. We draw $N = 10^5$ points $\boldsymbol{t}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ in $\mathbb{R}^5$ and use $x_i = \tanh(\sum_{j=1}^5 [\boldsymbol{t}_i]_j) + \max(0, \sum_{j=1}^5 (-1)^j [\boldsymbol{t}_i]_j) + \varepsilon_i$ with $\varepsilon_i \sim \mathcal{N}(0, 10^{-8})$. Because of its non-smoothness, this test function is more complicated than the one from the last section. Nonetheless, it is a simple sum of two ridge functions and our algorithm should be able to exploit this. We set $\lambda = 0$ and terminate the adaptive algorithm after the number of grid points has reached 1000. The results can be found in Figure 3. As in the previous example, we clearly see that the data transformation benefits the adaptive sparse grid algorithm significantly.

5.3 Ten-dimensional PDE problem. In this example from [4], 10^4 vectors $\boldsymbol{t}_i \in \mathbb{R}^{10}$ are drawn according to $\mathcal{N}(\mathbf{0}, \mathbf{I})$. These reflect the parameters of the diffusion

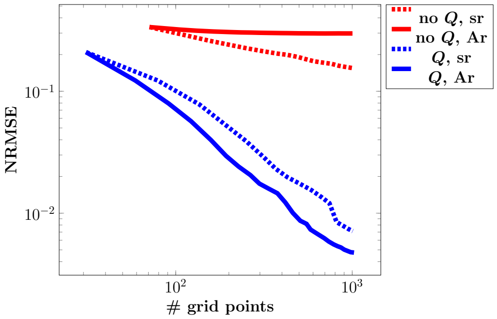

Fig. 3: NRMSE for 5d sum of ridge functions. Ar = ANOVA refinement, sr = standard refinement.

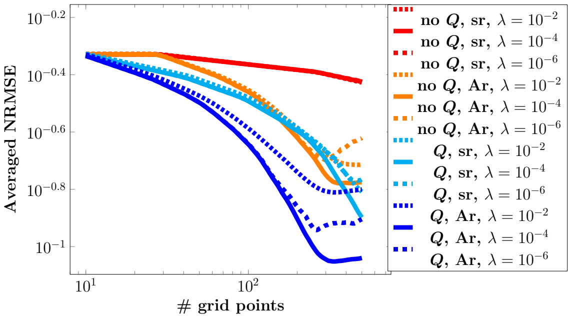

Fig. 4: Averaged NRMSE for the 10d PDE problem. Ar = ANOVA refinement, sr = standard refinement.

coefficient a in the two-dimensional elliptic PDE

$$-\nabla_{\boldsymbol{s}} \cdot (a(\boldsymbol{s},\boldsymbol{t})\nabla_{\boldsymbol{s}} u(\boldsymbol{s},\boldsymbol{t})) = 1 \qquad \boldsymbol{s} \in [0,1]^2$$

with Neumann boundary conditions on the right side of the domain and Dirichlet zero boundary conditions on the other sides. The $\boldsymbol{t}_i$ represent the first ten coefficients of a truncated Karhunen-Loéve decomposition of $\log(a)$ with correlation kernel $\exp(-\|\boldsymbol{r}-\boldsymbol{s}\|_1)$. For each $\boldsymbol{t}_i$, the PDE is solved and x_i is set to the spatial average of the solution on the Neumann boundary. Solving regression problems of this kind is an important task in uncertainty quantification, see [3] for details. We present averaged results over 20 random splits of the data into 5000 training and 5000 test points for different regularization parameters $\lambda \in \{10^{-2}, 10^{-4}, 10^{-6}\}$ in Figure 4.

The smallest error with the least amount of grid points is achieved with transformed data and ANOVA refinement. For this example, the ANOVA refinement performs better than standard refinement also in the case of untransformed data. However, standard refinement seems to produce more stable results with respect to λ. For ANOVA refinement, transformed data and $\lambda = 10^{-4}$ we achieve an averaged NRMSE smaller than 0.1, which is competitive with the best results from [4]. For all choices of λ, we also outperform the LASSO and Gaussian processes approaches tested there.

6 Conclusion

In this paper we have discussed the idea of preprocessing data in regression tasks in order to achieve a beneficial error decay and possibly smaller computational costs of the underlying algorithm. Our approach is motivated by the ANOVA decomposition and works best with regression methods based on tensor-product functions, e.g. on full grids and sparse grids. We provided an efficient algorithm to find the optimal matrix $\boldsymbol{Q} \in V_k(\mathbb{R}^d)$ to transform the data at hand. Subsequently, we discussed an adaptive sparse grid least-squares regression algorithm, which is able to adapt to the underlying regressor function. We showed how our preprocessing method significantly enhances the performance of the adaptive sparse grid algorithm for both artificial toy problems and a real-world application from uncertainty quantification.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds.* Princeton University Press, Princeton, NJ, 2008.

[2] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:1–123, 2004.

[3] P. Constantine. *Active Subspaces: Emerging Ideas in Dimension Reduction for Parameter Studies.* SIAM, Philadelphia, 2015.

[4] P. Constantine and J. Hokanson. Data-driven polynomial ridge approximation using variable projection. 2017. arXiv:1702.05859.

[5] C. Feuersänger. *Sparse Grid Methods for Higher Dimensional Approximation.* Dissertation, Institut für Numerische Simulation, Universität Bonn, 2010.

[6] J. Garcke, M. Griebel, and M. Thess. Data mining with sparse grids. *Computing*, 67(3):225–253, 2001.

[7] M. Griebel and M. Holtz. Dimension-wise integration of high-dimensional functions with applications to finance. *J. Complexity*, 26:455–489, 2010.

[8] J. Imai and K. Tan. Minimizing effective dimension using linear transformation. *Monte Carlo and Quasi-Monte Carlo Methods*, 2002:275–292, 2004.

[9] A. Owen. The dimension distribution and quadrature test functions. *Statistica Sinica*, 13:1–17, 2003.

[10] D. Pflüger, B. Peherstorfer, and H.-J. Bungartz. Spatially adaptive sparse grids for high-dimensional data-driven problems. *J. Complexity*, 26:508–522, 2010.

[11] B. Schölkopf and A. Smola. *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond.* The MIT Press – Cambridge, Massachusetts, 2002.

[12] I. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math. Comput. Simulation*, 55:271–280, 2001.