FAST EVALUATION OF ADDITIVE KERNELS: FEATURE ARRANGEMENT, FOURIER METHODS, AND KERNEL DERIVATIVES

THERESA WAGNER*, FRANZISKA NESTLER*, AND MARTIN STOLL*

Abstract. One of the main computational bottlenecks when working with kernel based learning is dealing with the large and typically dense kernel matrix. Techniques dealing with fast approximations of the matrix vector product for these kernel matrices typically deteriorate in their performance if the feature vectors reside in higher-dimensional feature spaces. We here present a technique based on the non-equispaced fast Fourier transform (NFFT) with rigorous error analysis. We show that this approach is also well suited to allow the approximation of the matrix that arises when the kernel is differentiated with respect to the kernel hyperparameters; a problem often found in the training phase of methods such as Gaussian processes. We also provide an error analysis for this case. We illustrate the performance of the additive kernel scheme with fast matrix vector products on a number of data sets. Our code is available at https://github.com/wagnertheresa/NFFTAddKer.

Key words. additive kernels, feature grouping, Fourier analysis, kernel derivatives, multiple kernel learning

1. Introduction.

Kernel methods [36, 71, 70, 56] are a crucial tool in many machine learning tasks such as support vector machines (SVMs) [35, 70, 11] or Gaussian processes (GPs) [80, 81, 50]. In the literature many kernel designs can be found and one of the common bottlenecks in their application is dealing with the large and often dense kernel matrix. Our goal in this paper is to analyze a general acceleration technique for additive kernels and their derivatives routed in Fourier analysis.

Out of the many kernel choices possible, the squared-exponential, the periodic, and the linear kernel are among the most commonly used kernels. The underlying structure of real-world data cannot always be described by those kernels immediately. However, combining several of such kernels by addition, multiplication or a mixture of both can add more complexity to the model [24], with such kernel combinations still fulfilling all kernel properties, see Williams and Rasmussen [81]. There has been a growing interest in additive kernels and multiple kernel learning [31]. Common applications are in computer vision for instance such as pedestrian [2, 3, 55] or human activity detection systems [12] and medicine, where additive kernel SVMs are used to detect pedestrians, predict human activities or recognize types of cancer [12]. The main motivation of working with additive kernels is that they can reduce the complexity and increase the interpretability of the problem. When working with GP models for instance the problem is based on similarity and neighborhood relations. This means that in order to sufficiently cover the domain a large amount of data is required, what then increases the computational cost of the kernel evaluations. By incorporating additivity to the model the complexity of the features and the curse of dimensionality can be reduced [22, 23].

Theoretical guarantees for the additive kernel structure are given by Yang and Tokdar [88], who present minimax risks for regression functions that admit an additive structure.

In the literature additive kernels are mainly associated with SVMs [55, 2, 83, 12, 16, 85], GPs [22, 23, 25] or source separation tasks [53]. Below, we briefly review some of the specifics of using additive kernels in SVMs and in GPs.

*University of Technology Chemnitz, Chemnitz, Germany (theresa.wagner@math.tu-chemnitz.de, franziska.nestler@math.tu-chemnitz.de, martin.stoll@math.tu-chemnitz.de).

Additive SVMs can be employed for abstract input spaces for instance and interpreting the resulting predictions is typically easier. Moreover, the favourable robustness properties of SVMs remain the same for additive SVMs [16]. In other works lookup tables are employed to reduce the training and testing time and save memory. Baek et al. [2] suggest to work with a cascade implementation of the additive kernel SVM to reduce the computation time and use lookup tables to avoid kernel expansion. The PmSVM-LUT method proposed by Yang and Wu [85] uses polynomial approximation for the gradient to accelerate the dual-coordinate descent method. Chan et al. [12] develop a similarity measure PLAME (piecewise-linear approximate measure) that is incorporated with the dual-coordinate descent method. By this they approximate the additive kernel to ensure an efficient training of the additive kernel SVM. Furthermore, additive kernels are employed when prior knowledge about the distribution of the data is given or an easily interpretable prediction function is desired [16]. Xie et al. [83] suggest the UKSVM (uncertain kernel SVM) method for classifying uncertain data. Another relevant application of additive kernel SVMs is histogram-based image comparison. Common choices for such additive histogram comparison kernels are the intersection or chi-squared kernel. Linearly combining functions of each coordinate of the histogram yields the comparison [55].

Additive kernels are also used in GP models. Durrande et al. [23] argue that even if the function to be approximated is not purely additive, additive Kriging models can express the additivity of the function well. They are combining the features of GP modeling with generalized additive models what is especially suitable for high-dimensional problems. A similar approach is presented by Durrande et al. [22], where additivity is incorporated in the covariance kernel to obtain GPs for additive models. The response of the generalized additive models (GAMs) [34] simulator can then approximately be separated into a sum of univariate functions. Duvenaud et al. [25] propose an expressive but tractable parameterization of the kernel function. By this all input interaction terms can be evaluated efficiently. The additive structure incorporated into the model which is present in many real data sets leads to increased interpretability and predictive power what yields a better performance compared to standard GP models overall.

In many papers using a kernel of the form

$$\kappa_{\mathrm{A}}^{t}(\boldsymbol{x},\boldsymbol{x}') = \sigma_f^2 \sum_{1\le i_1<i_2<\cdots<i_t\le d} \prod_{j=1}^{t} \kappa_{i_j}(x_{i_j}, x'_{i_j}), \tag{1.1}$$

with $\boldsymbol{x},\boldsymbol{x}' \in \mathbb{R}^d$, $t \le d$ and the signal variance parameter $\sigma_f \in \mathbb{R}_{>0}$, is suggested. It is referred to as the t-th order additive kernel [25] or the ANOVA kernel [71]. The term ANOVA is an abbreviation for analysis of variance. This is reasonable in this setting since we aim to work with a kernel that compares the variance of the data in detail. The kernels κ_{i_j} are one-dimensional base kernels. Depending on the order t of the additive kernel multiple base kernels are multiplied to cover higher order feature interactions. Overall κ_{A}^{t} yields the sum of all possible t-th order interactions of one-dimensional base kernels. Here, the idea is to compare the data on a subset of features first and summing over several of such kernels relying on fewer features [71].

Assume all base kernels are squared-exponential kernels

$$\kappa_{\mathrm{SE}}(\boldsymbol{x},\boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2\ell^2}\right)$$

with the same length-scale parameter $\ell \in \mathbb{R}_{>0}$. Then, the d-th order ANOVA kernel is nothing else than the squared-exponential kernel evaluated at all feature dimensions at once, since

$$\begin{aligned}\kappa_{\mathrm{A}}^{d}(\boldsymbol{x},\boldsymbol{x}') &= \sigma_f^2 \prod_{j=1}^{d} \kappa_j(x_j,x_j') = \sigma_f^2 \prod_{j=1}^{d} \exp\left(-\frac{\|x_j-x_j'\|_2^2}{2\ell^2}\right)\\ &= \sigma_f^2 \exp\left(-\sum_{j=1}^{d}\frac{\|x_j-x_j'\|_2^2}{2\ell^2}\right) = \sigma_f^2 \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2\ell^2}\right)\\ &= \sigma_f^2 \kappa_{\mathrm{SE}}(\boldsymbol{x},\boldsymbol{x}')\end{aligned}$$

by the power law. The sum of all such t-th order ANOVA kernels is the full additive kernel

$$\kappa_{\mathrm{A}}^{\mathrm{full}}(\boldsymbol{x},\boldsymbol{x}') = \sum_{t=1}^{d} \kappa_{\mathrm{A}}^{t}(\boldsymbol{x},\boldsymbol{x}'). \tag{1.2}$$

However, involving all subsets can be too over-determined for a good approximation of the data structure. Instead combining terms only relying on a small number of features can be more promising [76].

In this paper, we propose the use of a special case of the ANOVA kernel (1.1), whose design is discussed in detail in the following.

It is a common phenomenon that many real-world data sets are mainly based on sums of low-order feature interactions [25]. Therefore, we do not want to incorporate the full additive kernel (1.2) merging base kernels of all possible dimensionality. At the same time, we need to ensure to capture non-local structure, which is why we do not incorporate all feature dimensions within one kernel evaluation at once. Hence, we suggest to work with a weighted sum of kernels

$$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j) = \sigma_f^2 \sum_{s=1}^{P} \kappa_s(\boldsymbol{x}_i,\boldsymbol{x}_j), \tag{1.3}$$

where $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^d$, $i,j = 1,\dots,N$, and the sub-kernels κ_s depend on low-dimensional feature interactions only. For this we define sets of feature indices $\mathcal{W}_s$ building the s-th group of features, that is

$$\kappa_s(\boldsymbol{x}_i,\boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s}-\boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{2\ell^2}\right),$$

for $s = 1,\dots,P$ and $P \in \mathbb{N}$ the number of feature groups and sub-kernels. For the bivariate case the groups of features $\mathcal{W}_s \in \{(a,b) : a,b \in \{1,\dots,d\}, a \neq b\}$ are of length $d_s = 2$ and for the trivariate $\mathcal{W}_s \in \{(a,b,c) : a,b,c \in \{1,\dots,d\}, a \neq b \neq c\}$ with $d_s = 3$ for instance.

Accordingly, $\boldsymbol{x}_i^{\mathcal{W}_s} \in \mathbb{R}^{d_s}$, $i = 1,\dots,N$, are the corresponding data points restricted to those indices. The resulting kernel represents the special case of the ANOVA kernel (1.1) with $t = |\mathcal{W}_s| = d_s$ and Gaussian base kernels relying on d_s-dimensional windows $\mathcal{W}_s$ of features, and is referred to as the additive Gaussian kernel from now on. We aim for a method that keeps the computational complexity for large-scale applications low. Since more feature windows lead to more kernels what leads

to solving more linear systems with dense matrices that are square in the number of data points, we cannot work with all possible low order feature interactions. Instead we require a procedure for reasonably reducing the number of windows.

Strategies on how to determine the sets of feature indices $\mathcal{W}_s$ are elaborated in Section 2. If the number of data points N is large, multiplying and solving with $K = \sigma_f^2 \sum_{s=1}^{P} K_s$ is of quadratic or even cubic computational complexity. For this, we suggest employing NFFT-accelerated approximations in Section 3. Naturally, the choice of the parameter values can have a huge impact on the prediction quality and the parameters have to be optimized. For this, we introduce a NFFT-acceleration procedure for multiplying and solving with the matrix representing the derivative of the kernel with respect to the length-scale hyperparameter ℓ, that is $K_{\text{der}} = \sigma_f^2 \sum_{s=1}^{P} K_s^{\ell}$ with

$$\kappa_s^{\ell}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{\ell^3} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{2\ell^2}\right).$$

We demonstrate the corresponding approximation error empirically and provide error estimates analytically. In Section 4 we present an approach on how to determine $\mathcal{W}_s$ analytically in the Fourier setting and run first experiments according to this scheme. We showcase the numerical performance of the presented feature grouping techniques for the additive kernel design with NFFT-accelerated kernel evaluations in Section 5 and present concluding remarks in Section 6.

Main Contributions. We summarize the main contributions of this paper as follows:

- Novel combination of feature grouping techniques for the additive kernel setting with NFFT-accelerated kernel evaluation.
- Development of an NFFT-acceleration procedure for the derivative kernel.
- Derivation of Fourier error estimates for the trivariate Gaussian kernel and its derivative kernel.

2. Feature Engineering Techniques.

In this section we focus on determining the feature arrangement for the additive kernel introduced previously. We give a thorough overview of existing feature engineering methods in the literature. We later choose a few methods that are most suitable for the additive kernel setting. For the sake of comparability and reproducibility we focus on open source Python software. The performance of those methods is analyzed and compared comprehensively in Section 5.

2.1. Feature Selection and Grouping Techniques in the Literature.

We now give an overview of feature selection and feature grouping techniques proposed in the literature. This shall form the basis for feature arrangement techniques in the additive kernel setting.

2.1.1. Feature Selection.

In the literature many definitions for feature selection can be found. It is described as a dimensionality reduction technique that chooses a smaller subset of features from the original feature set trying to fulfill different criteria. In general, the most common objectives of feature selection are improving the prediction quality, reducing the computation (training and utilization) time, reducing measurement and storage requirements, increasing the comprehensibility and interpretability, gaining a better understanding of the data and stemming the curse of dimensionality [13, 49, 46, 32, 17, 57].

The most common approach to categorize feature selection algorithms is classifying them into filter, wrapper and embedded methods [46, 13, 42, 32, 57, 78, 69]. Filter methods select the features based on their importance with respect to the target concept [7]. For this statistical measures such as information gain [87, 67], chi-square test [87], Fisher score, correlation coefficient, variance threshold, reliefF [44, 67], F-statistic [20] or mRMR [61] are employed [78, 57]. Filter methods are independent of the choice of the learning algorithm. Therefore the obtained feature subsets can subsequently be transferred to any learning task on that data set. This is not the case for wrapper methods where feature selection is accomplished by performing the learning task on candidate subsets until a stopping criterion is met [45]. By design, wrapper methods are more computationally complex than filter methods but often more accurate [78]. Examples are recursive feature elimination [84], the sequential feature selection algorithm [39] or genetic algorithms. In embedded methods the feature selection is performed as part of the learning process with a specific learning algorithm [46]. One example for embedded methods are random forests [78]. Embedded methods are based on the same idea as wrapper methods but they are working with an objective function consisting of a goodness-of-fit term and a penalty for large number of variables [32] what makes them more efficient.

Often, the general procedure for feature selection is characterized as 4 key steps: subset generation, evaluation of the subset, stopping criteria and validation of the result [46, 51, 42, 17]. Dash and Liu [17] furthermore define 3 categories of generation procedures (complete, heuristic, random) and 5 categories of evaluation functions (distance, information, dependence [75], consistency, classifier error rate).

Other common categorization approaches found in the literature focus on the availability of the label information, the data perspective or the label and selection strategy perspective. Miao and Niu [57] distinguish between supervised, semi-supervised and unsupervised methods [26]. Yu and Liu [89] present feature selection techniques based on relevance and redundancy.

In general, when looking for a suitable feature selection method the following aspects should be considered: simplicity, stability, the desired number of reduced features, the classification accuracy and storage and computational requirements [13].

2.1.2. Feature Grouping. In addition to classical feature selection approaches as described above several feature selection techniques based on previous feature grouping can be found in the literature. The main idea behind this approach is to generate groups of features where the intra-group similarity is maximized and the inter-group similarity is minimized [48]. Afterwards feature selection is performed based on this group arrangement.

This procedure is motivated by discovering that relevant features are highly correlated in a high-dimensional setting. Therefore, groups of correlated features resistant to sample size variations can be formed [30].

To that effect, feature grouping comes with many benefits when learning with high-dimensional data [30]. The search space being reduced with feature grouping and the higher resistance to sample variations [48] leads to an improved stability of feature selection [41] and effectiveness of the search [30], helps to reduce the complexity of the model and increase the generalization capability [48] and potentially reduces the estimator variance [72]. Popular applications are text mining [5, 19] or microarray domains [6].

Among the most basic feature grouping methods are exhaustive or explicit search for feature groups [92, 93]. However, this combinatorial optimization problem is often

computationally infeasible when working with large data sets. To overcome this, greedy hill climbing strategies have been proposed, with features leading to the largest gain in the subset score greedily being added to the candidate subset individually [92]. However, they typically only lead to local optima.

Regularization techniques are very common algorithms for generating feature groups and belong to the category of embedded methods [30, 93, 86, 33]. By adding a regularization term to the objective function the model is fitted minimizing the coefficients what results in features with coefficients close to zero being dropped [30]. Common sparse-modeling algorithms are the lasso regularization [77] and its extensions such as group lasso [91], adaptive lasso, fused lasso and clustered lasso. Further regularization techniques worth mentioning are Bridge regularization [37], elastic net regularization [94] and the orthogonal shrinkage and clustering algorithm for regression (OSCAR) method introduced by Bondell and Reich [8]. However, many of the aforementioned methods suffer from the problem that they cannot distinguish groups of features that are similar but still different well and often tend to merge those groups together.

As a remedy, discriminative feature grouping (DFG) is proposed by Han and Zhang [33]. By introducing a novel regularizer in the feature coefficients fusing and discriminating feature groups is balanced out. Moreover, they present an adaptive DFG (ADFG) aiming to yield a better asymptotic property.

Subspace clustering methods represent another type of feature grouping techniques that intend to detect clusters in subspaces rather than the whole data space [15]. They are distinguished between hard subspace clustering [1] and soft subspace clustering methods [38, 21, 40]. While hard subspace clustering methods detect the exact subspaces of the clusters, in soft clustering methods subspaces with large weights are identified by assigning weights to features instead [15, 28]. Many of such methods have been proposed in the literature, such as CLIQUE [1], W-k-means [38], fuzzy subspace clustering (FSC) [29], EWKM [40], LAC [21] and EEW-SC [18]. Alternatives that are less sensitive to noise and missing values [15, 28] are FG-k-means, introduced by Chen et al. [15] as an iterative alternative soft subspace clustering method, and AFG-k-means [28]. While the feature groups are assumed to be given as inputs in FG-k-means, the groups are detected automatically by dynamically updating them during the iterative process in AFG-k-means instead.

Regarding stability, several group-based feature selection methods were developed for the purpose of improving robustness. The main reason for instability in feature selection techniques originates in the small number of samples in a high-dimensional domain and the goal of selecting the minimal subset without redundant features [90]. For this two frameworks have been proposed: dense feature groups (DFG) [90] and consensus group stable feature selection (CGS) [54]. Overall, the concept of those methods originates from the observation that features close to core (peak) regions have a high correlation.

2.2. Feature Arrangement Techniques for Additive Kernels. After having presented existing feature engineering techniques above, we want to choose methods suitable for arranging the features in the additive kernel setting. In this context, we refer to feature grouping as separating feature dimensions into multiple kernels by defining corresponding feature windows $\mathcal{W}_s$ as introduced in (1.3). For this, several requirements have to be fulfilled.

First, we do not only want to get rid of less relevant features but also need a sensible scheme for separating the feature indices into several small groups. Addi-

tionally, we want to keep the number of kernels P small since more kernels lead to higher computational costs as more matrix vector products need to be evaluated. The kernel matrix vector product approximations are more expensive the larger the size of the corresponding feature subset d_s. In order to exploit the full computational power of those approximation techniques d_s is required to be small. Since both demands are opposed to each other, the number P of kernels or the number of feature groups respectively needs to be balanced carefully with the cardinality of the feature groups.

Second, we do not necessarily have the kernel entries given explicitly. When working with large-scale problems fast approximation methods are often employed for speeding up multiplications with the kernel matrices. Then, the routine operates as a black box, where the data points and a vector are given as inputs and the kernel vector product is returned as the output. We go into more detail in Section 3. Indeed, a number of feature selection techniques require having those entries available explicitly.

In the remainder of this section we discuss several feature grouping techniques that aim to determine the feature groups in a sophisticated way. We describe some very basic feature grouping strategies first before we consider more elaborated ones next. In Section 5 we analyze and compare their performance.

Note that we refrain from adding certain feature grouping methods to our investigations even though they are somewhat prevalent in the literature. Examples are OSCAR, CLIQUE, FGOC (feature grouping and orthogonal constraints) that are not competitive regarding their computational complexity. Hierarchical clustering, (adaptive) discriminative feature grouping, k-means and fuzzy c-means clustering are methods where the strategy on how to define feature windows are not suited to the setting we want to employ.

2.2.1. Straightforward Feature Grouping Methods. A very basic strategy on how to separate the feature dimensions is to simply group the features following their feature indices determined by the column arrangement in the original data set. For $d = 6$ this yields windows $\mathcal{W}_1 = \{1, 2\}$, $\mathcal{W}_2 = \{3, 4\}$, $\mathcal{W}_3 = \{5, 6\}$ for $t = d_s = 2$ and windows $\mathcal{W}_1 = \{1, 2, 3\}$, $\mathcal{W}_2 = \{4, 5, 6\}$ for $t = d_s = 3$ respectively, for instance. For $t = d_s = 1$ this represents a special case of the feature index based allocation. Then, the feature dimensions are split into d one–dimensional windows and the features are arranged as $\mathcal{W}_s = \{s\}$ for $s = 1, \ldots, d$.

Even though this strategy is very basic it constitutes a valuable comparative method for examining whether putting more effort in terms of computational complexity and runtime into more complex techniques pays off in achieving higher predictive accuracy.

2.2.2. Methods Based on Feature Importance Ranking. In previous works [59, 79] we ranked all features by their mutual information score (MIS) [4] and arranged them into groups of 3 following their importance scores starting with the largest one. The MIS quantifies how much information about the label can be obtained by knowing the feature value. It is a univariate measure that does not examine the impact of a combination of several features on predicting the target. However, the MIS ranking method only requires the original data as input and is of low computational complexity. After having employed this feature arrangement technique in previous papers already we now want to analyze its performance more extensively by comparing it to several other methods.

Instead of computing the importance scores via MIS, other measures can be applied. Common alternatives are the Fisher score and reliefF. Alternatively, feature

importance can be obtained by fitting a decision tree model. By introducing a threshold, features with an importance score below this value are dropped.

Based on the feature importance scores, the features are ranked and arranged into groups of the desired size. The feature arrangement can follow different strategies. Features can either be arranged consecutively following the ranking such that the features with the 3 highest scores are arranged into the first window and so on for $|\mathcal{W}_s| = 3$ for instance or the features are separated into different feature groups successively so that features with similar importance scores do not end up in the same group. We refer to these arrangement strategies as 'consec' and 'distr' from now on.

2.2.3. Regularization Techniques. In contrast to computing each feature's importance score individually as described above, one can work with a regression model for estimating sparse coefficients. In the well-known Lasso regularization, the objective function

$$Z_{\text{Lasso}} = \frac{1}{2N}\|Xw - y\|_2^2 + \lambda_{\text{Lasso}}\|w\|_1$$

is minimized with respect to the coefficients w and $\lambda_{\text{Lasso}} > 0$, the regularization parameter that regulates the degree of sparsity of the estimated coefficient vector.

Elastic-Net is a regression model incorporating both the L1-norm and the L2-norm of the coefficients to the model. The corresponding objective

$$Z_{\text{EN}} = \frac{1}{2N}\|Xw - y\|_2^2 + \lambda_{\text{EN}}\rho\|w\|_1 + \frac{\lambda_{\text{EN}}(1-\rho)}{2}\|w\|_2$$

is again minimized with respect to the coefficients w and the ratio between the penalty terms is balanced with the L1-ratio ρ. Note that with $\rho = 1$ the objective of Elastic-Net equals with Lasso's objective.

By combining L1 and L2 regularization, Elastic-Net benefits from both the sparsity of the Lasso model and the regularization properties of ridge, such as stability. However, in settings with two correlated features, Lasso randomly chooses one of them while Elastic-Net encourages a grouping effect and tends to select both [94].

Note that the features are selected based on classical regression on the data matrix $X \in \mathbb{R}^{N\times d}$ rather than on how they perform in a non-linear context and hence we are working in a different context than in the kernel setting here.

In addition to 'consec' and 'distr' we introduce the 'direct' feature arrangement strategy for Lasso and Elastic-Net. In 'direct' the features with nonzero coefficients are immediately assigned to the windows consecutively following their indices without ranking them first.

2.2.4. Feature Arrangement Based on Clustering. Another approach for detecting feature groups is via feature clustering techniques. One way of doing this is via connected components. This method is based on the correlation matrix holding the Pearson product-moment correlation coefficients. In clustering via connected components two features are considered to be connected if the magnitude of their correlation value is larger than some predefined threshold. Based on those pairs the feature clusters are detected.

Different to the feature importance ranking and regularization techniques described above, clustering methods are unsupervised and do not incorporate the target values into the clustering process.

In addition to 'consec' and 'distr' we introduce 'single' as a feature arrangement strategy for the connected components method. In 'single' all centroid features build a window of length 1.

2.2.5. Feature Grouping Optimization via Regularization. Alternatively to the aforementioned approaches, we propose to determine the feature groups via an optimization. The objective Z_{fg} of this feature grouping optimization consists of the objective Z of the original classification/regression method plus a L1 regularization term, that is

$$Z_{\text{fg}}(\theta) = Z(\theta) + \lambda_{\text{fg}} \|\sigma_f^{\text{fg}}\|_1,$$

with $\lambda_{\text{fg}} > 0$ the regularization parameter balancing the impact of the L1 penalty,

$$K_{\text{fg}} = \underbrace{\sigma_{f_1}^2 K_1}_{K_1^{\text{fg}}} + \cdots + \underbrace{\sigma_{f_P}^2 K_P}_{K_P^{\text{fg}}}$$

and $\theta_{\text{fg}} = (\sigma_f^{\text{fg}}, \ell, \sigma_\varepsilon)$, where σ_ε is the noise parameter and $\sigma_f^{\text{fg}} = [\sigma_{f_1}, \ldots, \sigma_{f_P}]^\intercal$. Note that in contrast to the definition of the additive Gaussian kernel κ in (1.3), all sub-kernels K_s^{fg} are assigned a separate kernel weight σ_{f_s} now, with respect to which the optimization is performed. The noise σ_ε and length-scale ℓ parameters are kept fixed during the feature grouping optimization.

The model is initialized with all possible feature subsets with $d_s = 2$, what leads to $P = \binom{d}{2}$ sub-kernels K_s^{fg}. Through the L1 regularization term sparsity is ensured and most of the kernel weights σ_{f_s} are pushed to zero. By this, only a few non-zero kernel weights are obtained, what yields the desired optimal feature groups immediately.

Since the binomial coefficient grows big even for moderate feature dimensions, the feature grouping optimization is performed on a small subset of the data set only. Additionally, one can perform a feature ranking initially, using the MIS ranking for instance, to reduce the number of pairs P by dropping the features least relevant in the very beginning. While the feature importance ranking based and clustering techniques do not allow for repeated feature indices, this feature grouping optimization approach enables feature indices to appear in multiple feature windows $\mathcal{W}_s$.

3. NFFT-Accelerated Kernel Vector Products and Multiplications with the Derivative Kernel When working with large-scale data, multiplying and solving with the dense kernel matrix is the classical computational bottleneck. In this section we give an overview of techniques for accelerating evaluations with the kernel matrix and its derivative.

Examples for such methods are structured kernel interpolation (SKI), subset of regressors (SoR), deterministic training conditional (DTC), fully independent training conditional (FITC) and partially independent training conditional (PITC) approximation or hierarchical matrices (H-matrices). SKI is an approach based on inducing points that accelerates kernel approximations through kernel interpolation [82]. Another inducing point approach is SoR that approximates kernel vector multiplications based on inducing points with a specific prior for the vector. The DTC approximation works similarly to the SoR except from the relation between the function value and the inducing points being described by an exact test conditional instead of being deterministic such as for SoR. This means that with DTC the predictive response has a prior variance of its own [66]. FITC is another likelihood approximation with an extensive covariance. Different to SoR and DTC, FITC does not introduce a deterministic relation between the function value and the inducing points. For this, it

employs an approximation to the training conditional distribution as an independence assumption [66]. PITC further improves this approximation by equipping the training conditional with a block diagonal covariance [66]. An alternative approach are hierarchical matrices that are data-sparse approximations of non-sparse matrices by partitioning them into low-rank factorized sub-blocks [9].

In this paper we emphasize the NFFT-based fast summation technique we employed in Nestler et al. [59] and Wagner et al. [79]. In fact, all the above-mentioned fast kernel vector product approximation techniques have in common that their effectiveness and high efficiency is restricted to small feature space dimensions. This again motivates the need to splitting the feature space and working with additive kernels.

Moreover, we demonstrate the effect of hyperparameter choices on learning tasks and highlight the importance of hyperparameter optimization. Naturally, for this the derivatives with respect to the hyperparameters are required and one typically is faced with the task of multiplying also with the derivative matrix for that particular hyperparameter. We here advocate for an explicit computation employing the kernel structure as much as possible as finite difference approximations are typically not stable enough. Another alternative would be automatic differentiation techniques such as the one employed in Charlier et al. [14]. We introduce an NFFT-based technique for approximating multiplications with derivative kernels. A typical example would be the parameter training for Gaussian process regression where due to the log-determinant one typically requires matrix-vector products with the derivative matrix as part of a matrix function approximation for the correct computation of the parameter gradient.

3.1. Fourier Theory. In many kernel learning tasks such as the GP hyperparameter optimization, multiplying with the kernel matrix $K \in \mathbb{R}^{N,N}$ is most computationally complex. The cost of computing its product Kv with a vector $v \in \mathbb{R}^N$ through the conjugate gradient method is $\mathcal{O}(N^2)$ in each iteration, for instance. This scales badly for large-scale applications. Therefore, we approximate these products leveraging the computational power of the non-equispaced fast Fourier transform (NFFT) instead. This is realized by applying the fast summation approach, in which the NFFT and the adjoint NFFT, confer Potts and Steidl [64], are combined to compute sums of the form

$$h(\boldsymbol{x}'_i) \coloneqq \sum_{j=1}^{N} v_j \kappa(\boldsymbol{x}'_i, \boldsymbol{x}_j) \quad \forall i = 1, \dots, N' \tag{3.1}$$

efficiently. For this, the kernel κ is approximated by a trigonometric polynomial, what can be written as

$$\kappa(\boldsymbol{x}', \boldsymbol{x}) = \kappa(\boldsymbol{r}) \approx \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{c}_{\boldsymbol{k}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{r}/L}, \tag{3.2}$$

where $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$, $\boldsymbol{r} \coloneqq \boldsymbol{x}' - \boldsymbol{x}$, L is the period that has to be chosen appropriately, $\hat{c}_{\boldsymbol{k}}$ are the Fourier coefficients and $\boldsymbol{M} = (M_1, \dots, M_d)^\intercal \in 2\mathbb{N}^d$ is a multivariate grid size, that is a d-dimensional vector with even integer components, what gives multivariate index sets

$$\mathcal{I}_{\boldsymbol{M}} \coloneqq \{-\tfrac{M_1}{2}, \dots, \tfrac{M_1}{2} - 1\} \times \cdots \times \{-\tfrac{M_d}{2}, \dots, \tfrac{M_d}{2} - 1\}$$

of cardinality $|\mathcal{I}_{\boldsymbol{M}}| = M_1 \cdot \ldots \cdot M_d$. Typically, we set $M_1 = \cdots = M_d = m$, so that the grid size is the same respective all dimensions. Replacing κ in (3.1) by its Fourier

representation (3.2) and rearranging the sums gives

(3.3)
$$h(\boldsymbol{x}_i') \approx \tilde{h}(\boldsymbol{x}_i') = \sum_{j=1}^{N} v_j \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{c}_{\boldsymbol{k}} \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^{\mathsf{T}}(\tilde{\boldsymbol{x}}_i'-\tilde{\boldsymbol{x}}_j)} = \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{c}_{\boldsymbol{k}} \left(\sum_{j=1}^{N} v_j \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}^{\mathsf{T}}\tilde{\boldsymbol{x}}_j} \right) \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^{\mathsf{T}}\tilde{\boldsymbol{x}}_i'},$$

where $\tilde{\boldsymbol{x}}_i'$ and $\tilde{\boldsymbol{x}}_j$ are now scaled nodes, for which $\tilde{\boldsymbol{x}}_i' - \tilde{\boldsymbol{x}}_j \in [-1/2, 1/2]^d$ holds. Note that $\tilde{h}$ is now a periodic function with period 1 in each coordinate direction. By this, we can reduce the arithmetic complexity for computing Kv from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$, provided that the parameters involved are chosen appropriately. The inner sums for all $\boldsymbol{k}$ can be computed via a so-called adjoint NFFT (or type-2 nonuniform FFT) and the approximation of the outer sums are then realized by a d-variate NFFT (or type-1 nonuniform FFT). This procedure is known as NFFT-based fast summation. NFFT and adjoint NFFT themselves are approximate algorithms for an efficient evaluation of the required trigonometric sums at equidistant nodes. The accuracy of these algorithms is controlled by several parameters, which we do not further discuss here. For detailed information concerning NFFT and related algorithms we refer to Keiner et al. [43] and references therein.

In our investigations, the kernel function κ is non-periodic. Thus, the approximation by a trigonometric polynomial is not straightforward. We refer to our previous paper [59] and references therein for more details on the underlying theory.

In 1D, the easiest periodization approach just continues the kernel function periodically in order to obtain a continuous 1-periodic function $\tilde{\kappa}(r) := \kappa(r + k)$, where $k \in \mathbb{Z}$ is chosen such that $r + k \in [-1/2, 1/2]$, see Figure 1 for an illustration. The Fourier coefficients of that $C_0(\mathbb{T})$-continuation will tend to zero like $\mathcal{O}(k^{-2})$. A faster decay of the Fourier coefficients can be achieved by a smoother periodization, in which the function is regularized at the edges by a smooth transition, see Potts and Steidl [64].

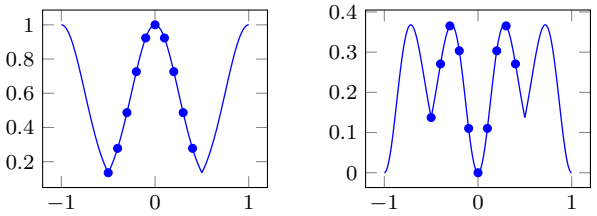

FIG. 1. *An even function of the form* $f(r) = \exp(-r^2/2\ell^2)$ *(left) or* $f(r) = \exp(-r^2/(2\ell^2)) \cdot (r^2/2\ell^2)$ *(right), defined on* $[-1/2, 1/2]$*, is periodized via simple periodic continuation. The resulting periodic function is at least continuous, but in general not smooth. A finite number of approximating Fourier coefficients can be obtained by sampling the function in equidistant points (marked by the dots) and applying the FFT. Alternatively, one can make use of the analytic Fourier coefficients, provided they are known.*

The presented periodization technique is also applicable to multivariate radial kernels in order to periodize the function and approximate it for $\boldsymbol{r} \in [-1/2, 1/2]^d$. In the numerical experiments we make use of the NFFT-based fast summation approach [43], where only a radial section of the function is approximated, that is $\|\boldsymbol{r}\| \leq \frac{1}{2}$. For details we refer to Potts et al. [65].

For the Gaussian kernel $\kappa_{\mathrm{gauss}}(r) := \mathrm{e}^{-r^2/2\ell^2}$ we can compute the Fourier coeffi-

cients of $\tilde{\kappa}_{\text{gauss}}(r)$ as

$$\begin{aligned} c_k(\tilde{\kappa}_{\text{gauss}}) &= \int_{-1/2}^{1/2} \mathrm{e}^{-\frac{r^2}{2\ell^2}} \mathrm{e}^{2\pi \mathrm{i} k r} \mathrm{d}r \\ &= \mathrm{e}^{-2\pi^2 k^2 \ell^2} \int_{-1/2}^{1/2} \mathrm{e}^{-(\frac{r}{\sqrt{2}\ell} - \pi \mathrm{i} k \sqrt{2}\ell)^2} \mathrm{d}r \\ &= \cdots = \ell\sqrt{2\pi} \mathrm{e}^{-2\pi^2 k^2 \ell^2} \operatorname{Re}\left[\operatorname{erf}\left(\frac{1}{2\sqrt{2}\ell} + \pi \mathrm{i} k \sqrt{2}\ell\right)\right], \end{aligned}$$

where $\operatorname{erf}(z) \coloneqq \frac{2}{\sqrt{\pi}} \int_0^z \mathrm{e}^{-t^2} \mathrm{d}t$ is the complex-valued error function. The final result is obtained by making use of simple integration techniques as well as symmetry properties of the error function, that is $\operatorname{erf}(-z) = -\operatorname{erf}(z)$ and $\operatorname{erf}(\overline{z}) = \overline{\operatorname{erf}(z)}$. The complex-valued error function is rather difficult to evaluate numerically and is also hard to handle analytically. For the calculation of the approximating Fourier coefficients, we prefer to use the FFT in practice, as described above. We also do not use this analytical form of the Fourier coefficients in our error estimation later on. It is only given here for the sake of completeness.

Note that

$$\kappa'_{\text{gauss}}(r) = -\frac{r}{\ell^2} \kappa_{\text{gauss}}(r), \quad \kappa''_{\text{gauss}}(r) = \left(\frac{r^2}{\ell^4} - \frac{1}{\ell^2}\right) \kappa_{\text{gauss}}(r),$$

that is for the kernel $\kappa_{\text{der}}(r) \coloneqq r^2/(2\ell^2)\mathrm{e}^{-r^2/2\ell^2}$ we obtain

$$c_k(\tilde{\kappa}_{\text{der}}) = \frac{1}{2}\left(c_k(\tilde{\kappa}_{\text{gauss}}) + \ell^2 c_k(\tilde{\kappa}''_{\text{gauss}})\right) = \frac{1}{2}\left(1 - 4\pi^2 k^2 \ell^2\right) c_k(\tilde{\kappa}_{\text{gauss}}),$$

where we apply the well-known differentiation properties for Fourier series.

So far we just considered the univariate case, where we approximate a certain kernel in terms of m Fourier coefficients. Considering uniform grids in higher dimensions, the number of coefficients m^d on the grid grows exponentially fast. Thus, the computational efficiency of the NFFT approach pays off most for rather small input-dimensions, say $d < 4$. As the presented method is designed for large-scale data with many features, a strategy on how to arrange small groups of feature combinations and to detect the most relevant ones is required. By this, several fast NFFT multiplications each relying on a small number of features can be combined via the additive kernel setting as introduced above.

3.2. Scaling the Data. As described above, we make use of periodic functions and Fourier approximations in order to compute the matrix-vector products efficiently. Since we work in a periodic setting, that is on a finite interval and not on $\mathbb{R}$, we have to ensure that the data points are scaled into a finite interval.

In order to apply the fast summation approach, as explained above, we have to scale the data such that $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \frac{1}{2}$ for all pairs i, j, which is fulfilled if all the data are scaled such that $\|\boldsymbol{x}_j\| \leq \frac{1}{4}$. Therefore, the d-dimensional data points are scaled such that $\boldsymbol{x}_j \in [-1/4, 1/4]^d$ first. If we denote by $d_{\max} = \max_s d_s$ the maximal number of features incorporated in the sub-kernels, then the maximum norm of a data point, restricted to a set of $d_{\max}$ features, is given by

$$\Delta_{\max} = \frac{\sqrt{d_{\max}}}{4}.$$

Thus, we define the scaled nodes via

$$\tilde{\boldsymbol{x}}_j \coloneqq \frac{1}{4} \cdot \frac{\boldsymbol{x}_j}{\Delta_{\max}} = \frac{\boldsymbol{x}_j}{\sqrt{d_{\max}}}.$$

If a length-scale parameter ℓ has already been chosen to be applied to the nodes $\boldsymbol{x}_j \in [-1/4, 1/4]^d$, we scale it with the same scaling factor, that is $\tilde{\ell} := \ell/\sqrt{d_{\max}}$, so that

$$\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\ell^2} = \frac{\|\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j\|_2^2}{2\tilde{\ell}^2}.$$

The advantage of prescaling the data is that the scaling parameter is computed based on the scaled data and is scaled equally in all dimensions. Without this prescaling, the scaling is different for every fastadj object being constructed for the particular windows each. This can turn out to be problematic when performing global sensitivity analysis for instance, see Section 4. Note that the ℓ values displayed for the empirical results are the initially chosen length-scale parameters for the data already scaled to $[-1/4, 1/4]^d$. The length-scales are then scaled based on the corresponding scaling factor before running the model.

We provide the GitHub repository `prescaledFastAdj`[1] in which the prescaling is incorporated as described above.

3.3. Implementing the NFFT Approach. Above, we explain the theory behind the fast NFFT-based approximation technique for matrix-vector multiplications with a kernel K and its derivatives. In our setting, K is defined by the additive Gaussian kernel (1.3)

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \sum_{s=1}^{P} \underbrace{\exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{2\ell^2}\right)}_{\kappa_s}. \tag{3.4}$$

As introduced in Nestler et al. [59] this is implemented as a black box approach, where only the data points restricted to the windows $\mathcal{W}_s$, the kernel parameter ℓ and a vector v, the kernel shall be multiplied with, are required as inputs and the corresponding approximation of $K_s v$ is returned. The underlying kernel is defined as

$$\kappa_s^{\text{gauss}}(\boldsymbol{x}_i^{\mathcal{W}_s}, \boldsymbol{x}_j^{\mathcal{W}_s}) = \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{c^2}\right) \tag{3.5}$$

in the implementation, that is with $c = \sqrt{2}\ell$, $\kappa_s^{\text{gauss}} = \kappa_s$. Summing over several of such approximations each relying on another window $\mathcal{W}_s$ and multiplying this sum by the signal variance parameter σ_f^2 we obtain $Kv = \sigma_f^2 \sum_{s=1}^{P} K_s^{\text{gauss}} v$.

As motivated earlier, the choice of hyperparameters affects the performance of the learning algorithm tremendously. Figure 2 visualizes the impact of the parameter choices on the prediction quality with additive kernel ridge regression (KRR) on the Protein and KEGGundir data sets for instance, where the solution to the system

$$(K + \beta I)v = y, \tag{3.6}$$

[1] https://github.com/wagnertheresa/prescaledFastAdj

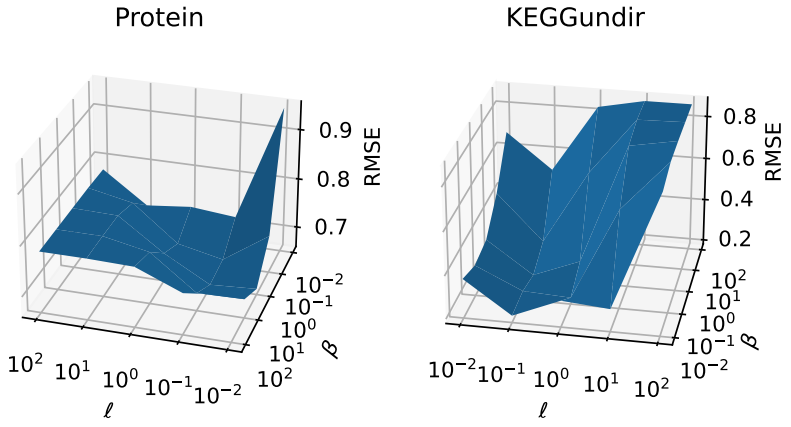

FIG. 2. *RMSE surface for additive kernel ridge regression and different length-scale and regularization parameters ℓ and β, where $N = 1000$ and the windows are determined consecutively via MIS ranking.*

with $\beta \in \mathbb{R}$ the regularization parameter, is sought. The plot shows the root mean square error (RMSE) on a grid of different values for the length-scale parameter ℓ and β and highlights the significance of hyperparameter optimization. For more information on the data we refer to Section 5.

For optimizing the objective of a regression model for instance, the kernel vector product Kv must be differentiated with respect to the kernel parameters σ_f and ℓ. Differentiation with respect to the signal variance can easily be realized for the kernel evaluation κ in (3.4) since

$$\frac{\partial K_{ij}}{\partial \sigma_f} = 2\sigma_f \sum_{s=1}^{P} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{2\ell^2}\right) = \frac{2}{\sigma_f} K_{ij}.$$

Therefore, the multiplication of the derivative kernel $\frac{\partial K}{\partial \sigma_f}$ and v can be performed similarly to the product Kv using the same implementation except that the sum of the approximations $K_s v$ is multiplied by $2\sigma_f$ instead of σ_f^2, that is $\frac{\partial K}{\partial \sigma_f} v = \frac{2}{\sigma_f}\sigma_f^2 \sum_{s=1}^{P} K_s^{\text{gauss}} v$. Differentiation with respect to the regularization parameter is straightforward.

However, differentiation with respect to the length-scale parameter ℓ gives

$$\frac{\partial K_{ij}}{\partial \ell} = \sigma_f^2 \sum_{s=1}^{P} \frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{\ell^3} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{2\ell^2}\right) = \sigma_f^2 \sum_{s=1}^{P} \underbrace{\frac{C_{s_{ij}}}{\ell^3} \circ K_{s_{ij}}}_{K^{\ell}_{s_{ij}}},$$

with $C_{s_{ij}} = \|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2$, is more complicated. For the entry-wise multiplication in the Hadamard products $C_s \circ K_s$ many nice properties as the associative law do not hold. Thus, we cannot employ our technique from approximating Kv with the kernel κ_s^{gauss} as in (3.5) directly. Instead, we introduce a derivative kernel

$$\kappa_s^{\text{der}}(\boldsymbol{x}_i^{\mathcal{W}_s}, \boldsymbol{x}_j^{\mathcal{W}_s}) = \frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{c^2} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2^2}{c^2}\right), \tag{3.7}$$

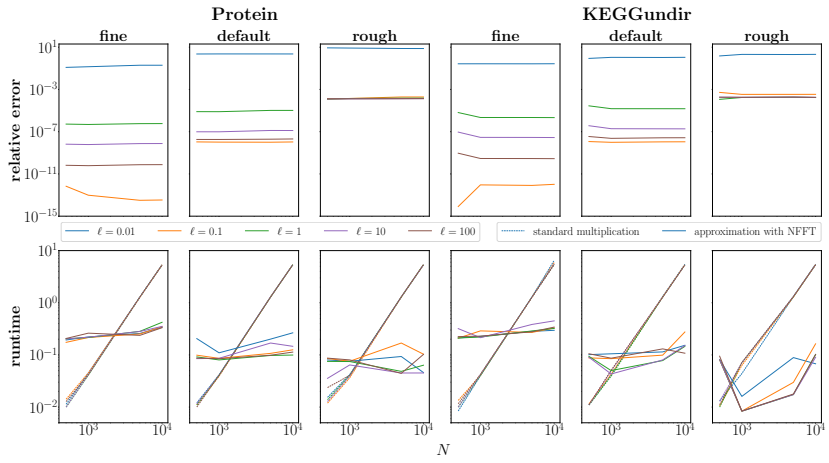

FIG. 3. *Fourier approximation error for computing Kv with different values of ℓ, where $v = \mathbf{1}$, $N_{fg} = 1000$, $d_{max} = 3$ and the windows are determined consecutively via MIS ranking, in comparison with standard multiplication.*

with $c = \sqrt{2}\ell$. For implementation reasons, the parameter in the denominator of both terms in κ_s^{der} is chosen equally, that is $K_s^{\mathrm{der}} = \frac{\ell}{2} K_s^{\ell}$ and $K^{\mathrm{der}} v = \frac{\partial K}{\partial \ell} v = \sigma_f^2 \sum_{s=1}^{P} \frac{2}{\ell} K_s^{\mathrm{der}} v$.

The corresponding implementations can be found in the `prescaledFastAdj` repository, in which the NFFT-accelerated kernel and derivative kernel evaluations are implemented. $\kappa_s^{\mathrm{gauss}}$ and κ_s^{der} are referred to as kernel = 1 and kernel = 2, respectively. Within the 'fastsum' module of the underlying `NFFT`[2] repository [43], $\kappa_s^{\mathrm{gauss}}$ is embedded as the 'gaussian' kernel and κ_s^{der} as the 'xx_gaussian' kernel.

In Figure 3 we illustrate the Fourier approximation error for multiplying the kernel K with the $\mathbf{1}$ vector for subsets of the data sets Protein and KEGGundir, several choices for the length-scale parameter ℓ and the three different setup presets for the parameters of the NFFT fastsum method 'fine' ($m = 64$), 'default' ($m = 32$) and 'rough' ($m = 16$). Here, by relative error we denote the relative difference of the Euclidean norms of the exact product Kv and its Fourier approximation. Note that we restrict the size of the considered subsets to 10^4 at a max since the computations break for bigger matrices in the standard multiplication due to a lack of memory. However, the NFFT-based approximation runs smoothly in such cases. The setups control the number of Fourier coefficients and therefore describe the degree of accuracy in the approximation, where 'fine' provides the most precise approximations, 'rough' is least precise and 'default' is in between. This characteristic is also displayed in the figure. The relative error plots for the 'rough' setup are on a higher level than the 'default' ones that lie above the 'fine' ones for the corresponding parameters ℓ. Moreover, we compare the runtime for computing Kv via standard multiplication and approximation with the NFFT approach in the second row of the plot. While the NFFT approach has a basic complexity for setting up the fast adjacency object and for computing the Fourier coefficients the runtime does not ascent steeply for larger scales.

[2] https://github.com/NFFT/nfft

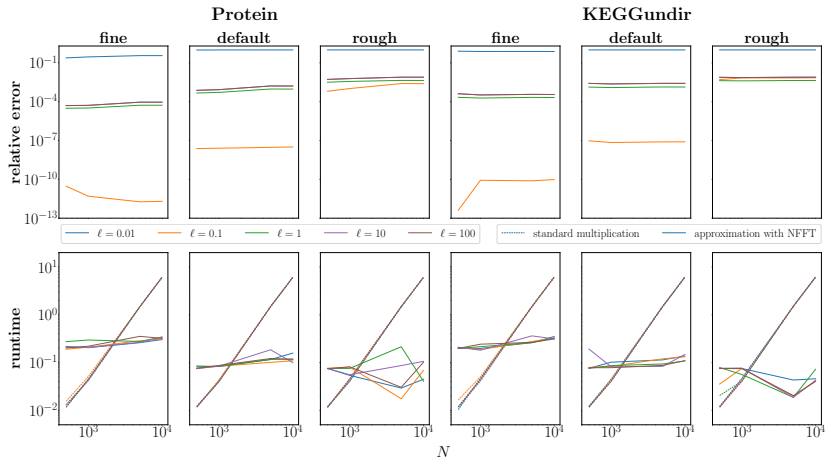

FIG. 4. *Fourier approximation error for computing* $\frac{\partial K}{\partial \ell} v$ *with different values of* ℓ*, where* $v = \mathbf{1}$*,* $N_{fg} = 1000$*,* $d_{max} = 3$ *and the windows are determined consecutively via MIS ranking, in comparison with standard multiplication.*

In contrast, the standard multiplication starts at a very low level for small subset sizes but increases strongly for larger kernels. Whereas the NFFT setups obviously do not impact the runtime plots for the standard multiplication, the runtime of the NFFT approximations differs. The most Fourier coefficients have to be computed in 'fine' and the least in 'rough'. Therefore, 'fine' naturally has a higher runtime than 'default' that has a higher runtime than 'rough'. While the value of ℓ mostly does not seem to have a huge impact on the runtime, the Fourier approximation error evidently highly differs for various values of ℓ. For very small values of ℓ the relative error can become larger than 10. In contrast, for medium sized values of ℓ the relative error ranges between 10^{-3} and 10^{-15}, depending on the setup, and for large values the relative error is between 10^{-4} for 'rough' and 10^{-10} for 'fine'.

Figure 4 shows the analogous results for the Fourier approximation error for multiplying the derivative kernel $\frac{\partial K}{\partial \ell}$ with the $\mathbf{1}$ vector. Overall, the relative errors and runtimes show the same trend as in Figure 3. The main difference is that the relative error is by far the smallest for $\ell = 0.1$ for the 'fine' and 'default' setup presets. In contrast, $\ell = 0.01$ clearly yields the largest error and for the length-scales larger or equal 1 the relative error is at the same level in between. In contrast, the relative errors for length-scales larger or equal 1 show a greater variation for the distinct values of ℓ up to several orders of magnitude in Figure 3.

The relative approximation errors are not satisfactory for all values of ℓ, of course. The NFFT approach does not approximate the product Kv well when the value of ℓ is very small. Note that ℓ always appears squared in the denominator of the exponential. With that, very small values ℓ lead to kernel matrices K_s^{gauss} with all entries close to zero except the diagonal being ones. This gives an identity matrix of full rank. The other extreme case is when the values of ℓ are very large. Then, all entries in K_s^{gauss} are close to one what gives a rank 1 matrix. In the derivative case the approximation error is biggest for very small and very large values of ℓ. In both cases, all entries of K_s^{der} are close to zero, what yields a zero matrix of zero rank.

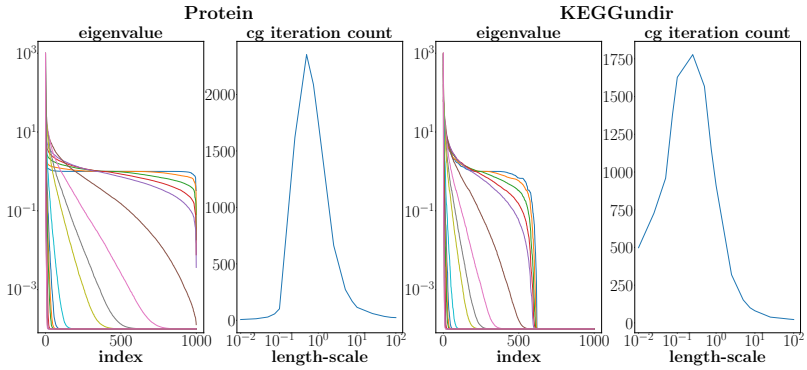

FIG. 5. *Eigenvalue decay and iteration count of unpreconditioned CG to solve* (3.6) *to reach the relative residual tolerance* 10^{-4} *for fixed* $N = 1000$ *and regularization parameter* $\beta = 0.0001$ *but different length-scales* ℓ*, where the windows are determined consecutively via MIS ranking.*

When the kernel matrices are of such special structure, multiplying and solving with the kernel matrix is not as challenging. This is emphasized by Figure 5 where the eigenvalue decay is shown alongside the CG iteration count for different values of ℓ. As before K has beneficial properties for very small and very large values of ℓ and as such CG does not require many iterations until convergence. More iterations are needed for very small values ℓ than for very large ones for the KEGGundir data set. That effect can be attributed to K being of full rank in contrast to having rank 1, what is a more difficult system to solve, naturally. More importantly the number of iterations clearly has a peak for moderate values of ℓ, when K is dense and not of a special structure. In the eigenvalue plots the slowly decaying curves correspond to a large CG iteration count.

Since we are mostly concerned with settings in which the system is not as well-posed and those cases align with values ℓ for which the NFFT-approach provides a good Fourier approximation error, the NFFT-based approximation is a competitive acceleration method independently of that effect.

3.4. Fourier Error Estimates. After illustrating the empirical Fourier approximation error in different settings above, we derive analytical error estimates next. For that, we assume that the kernels are considered on $[-1/2, 1/2]^3$ and periodized by simple periodic continuation, as explained above. Moreover, we assume that the length-scale parameter ℓ is already scaled by the scaling factor for the corresponding data points. Since kernels in higher dimensions can be derived via tensor products, the heart of the presented proofs consists of estimating the analytical Fourier coefficients and their sums for the univariate case. As we exclusively work with dimensions smaller or equal three in the presented additive kernel design, we derive the resulting error estimates for $d = 3$. The case $d = 2$ can be treated similarly, whereas the estimates for $d = 1$ immediately follow from the computations, see Remarks 3.3 and 3.5. The error estimates for the Gaussian kernel in two dimensions have already been presented in a slightly different form in Potts and Steidl [64]. Our proofs follow a similar baseline. To the best of our knowledge the Fourier approximation error for the derivative Gaussian kernel, see Theorem 3.4, is estimated for the first time in this paper.

As introduced above in (3.1) and (3.3), kernel vector products can be represented in a summation form as denoted by h and approximated by a truncated Fourier series denoted by $\tilde{h}$, that is

$$h(\boldsymbol{x}_i') \coloneqq \sum_{j=1}^{N} v_j \kappa(\boldsymbol{x}_i', \boldsymbol{x}_j) \quad \forall i = 1, \dots, N',$$

$$h(\boldsymbol{x}_i') \approx \tilde{h}(\boldsymbol{x}_i') = \sum_{j=1}^{N} v_j \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{c}_{\boldsymbol{k}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal (\boldsymbol{x}_i' - \boldsymbol{x}_j)} = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{c}_{\boldsymbol{k}} \left(\sum_{j=1}^{N} v_j \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}_j} \right) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}_i'},$$

for $\boldsymbol{x}_i', \boldsymbol{x}_j \in [-1/2, 1/2]^3$ already scaled.

Then, by the Hölder inequality the Fourier approximation error is determined by

$$\left| h(\boldsymbol{x}_i') - \tilde{h}(\boldsymbol{x}_i') \right| = \left| \sum_{j=1}^{N} \alpha_j \kappa_{\mathrm{ERR}}(\boldsymbol{x}_i', \boldsymbol{x}_j) \right| \leq \|\boldsymbol{\alpha}\|_1 \|\kappa_{\mathrm{ERR}}\|_\infty, \tag{3.8}$$

for $i = 1, \dots, N'$ and κ_{ERR} the difference between the kernel representation by a Fourier series and its approximation by a truncated one, where

$$\|\boldsymbol{\alpha}\|_1 \coloneqq \sum_{j=1}^{N} |\alpha_j|,$$

$$\|\kappa_{\mathrm{ERR}}\|_\infty \coloneqq \max_{\boldsymbol{x}, \boldsymbol{x}' \in [-1/4, 1/4]^d} |\kappa_{\mathrm{ERR}}(\boldsymbol{x}', \boldsymbol{x})| = \max_{\boldsymbol{r} \in [-1/2, 1/2]^d} |\kappa_{\mathrm{ERR}}(\boldsymbol{r})|.$$

We present theoretical bounds on the achievable approximation error $\|\kappa_{\mathrm{ERR}}\|_\infty$ in the following, where we set

$$\mathcal{I}_m \coloneqq \{\boldsymbol{k} \in \mathbb{Z}^d : -\tfrac{m}{2} \leq k_j < \tfrac{m}{2} \ \forall j = 1, \dots, d\}.$$

To this end, we consider the periodized Gaussian and derivative Gaussian kernels in $d = 3$ dimensions. For $\boldsymbol{r} \coloneqq \boldsymbol{x}' - \boldsymbol{x} \in [-1/2, 1/2]^3$, we obtain

$$\kappa(\boldsymbol{r}) = \sum_{\boldsymbol{k} \in \mathbb{Z}^3} c_{\boldsymbol{k}}(\kappa) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{r}} \approx \sum_{\boldsymbol{k} \in \mathcal{I}_m} c_{\boldsymbol{k}}(\kappa) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{r}} \approx \sum_{\boldsymbol{k} \in \mathcal{I}_m} \hat{c}_{\boldsymbol{k}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{r}} =: \kappa_{\mathrm{F}}(\boldsymbol{r}),$$

where $c_{\boldsymbol{k}}(\kappa)$ are the analytical Fourier coefficients and $\hat{c}_{\boldsymbol{k}}$ the discrete Fourier coefficients obtained from m^3 equidistant samples. The Fourier approximation error that is studied in this subsection is now defined as

$$\kappa_{\mathrm{ERR}}(\boldsymbol{r}) \coloneqq \kappa(\boldsymbol{r}) - \kappa_{\mathrm{F}}(\boldsymbol{r}).$$

Note that the sums $h(\boldsymbol{x}_i')$ are ultimately not evaluated directly, but approximated by the NFFT algorithms, that is

$$\tilde{h}(\boldsymbol{x}_i') \approx \underbrace{\sum_{\boldsymbol{k} \in \mathcal{I}_m} \hat{c}_{\boldsymbol{k}} \underbrace{\left(\sum_{j=1}^{N} v_j \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}_j} \right)}_{\text{approx. via adjoint NFFT}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}_i'}}_{\text{approx. via NFFT}},$$

what introduces further approximation errors. However, these approximation errors are neglected at this point and the pure Fourier approximation error is considered in the following. The NFFT algorithms depend on several parameters controlling the accuracy. Choosing these parameters appropriately, the NFFT approximation errors can be made negligibly small.

LEMMA 3.1. *(Aliasing error) Let $f \in L_2(\mathbb{T}^3)$ be a function with absolutely convergent Fourier series*

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^3} c_{\boldsymbol{k}}(f)\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}}, \quad c_{\boldsymbol{k}}(f) = \int_{\mathbb{T}^3} f(\boldsymbol{x})\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}}\,\mathrm{d}\boldsymbol{x}$$

and let an approximation of f be given by (replacing the analytic Fourier coefficients by the discrete Fourier coefficients using m equidistant samples on the grid $\mathcal{I}_m$)

$$f_F(\boldsymbol{x}) \coloneqq \sum_{\boldsymbol{k}\in\mathcal{I}_m} \hat{c}_{\boldsymbol{k}}\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}}, \quad \hat{c}_{\boldsymbol{k}} = \frac{1}{m^3}\sum_{\boldsymbol{j}\in\mathcal{I}_m} f\left(\frac{\boldsymbol{j}}{m}\right)\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{j}\boldsymbol{k}/m}.$$

Then

$$\hat{c}_{\boldsymbol{k}} = c_{\boldsymbol{k}}(f) + \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} c_{\boldsymbol{k}+m\boldsymbol{r}}(f) \tag{3.9}$$

and the approximation error can be estimated for all $\boldsymbol{x} \in \mathbb{T}^3$ by

$$|f(\boldsymbol{x}) - f_F(\boldsymbol{x})| \le 2\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}}\sum_{\boldsymbol{k}\in\mathcal{I}_m} |c_{\boldsymbol{k}+m\boldsymbol{r}}(f)| = 2\sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_m} |c_{\boldsymbol{k}+m\boldsymbol{r}}(f)|\,.$$

Proof. For the derivation of the well-known aliasing formula (3.9), which states the relationship between the analytic and the discrete Fourier coefficients, we refer to the standard literature on Fourier analysis, see for instance Plonka et al. [62] and references therein.

The stated estimate between f and f_F is then a simple consequence of the triangle inequality, as sketched in the following. We rewrite the function f as

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathcal{I}_m} c_{\boldsymbol{k}}(f)\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}} + \sum_{\boldsymbol{k}\in\mathcal{I}_m}\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} c_{\boldsymbol{k}+m\boldsymbol{r}}(f)\mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+m\boldsymbol{r})^\intercal\boldsymbol{x}}$$

and its approximation f_F as

$$f_F(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathcal{I}_m} c_{\boldsymbol{k}}(f)\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}} + \sum_{\boldsymbol{k}\in\mathcal{I}_m}\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} c_{\boldsymbol{k}+m\boldsymbol{r}}(f)\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}},$$

and conclude

$$f(\boldsymbol{x}) - f_F(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathcal{I}_m}\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \left(\mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+m\boldsymbol{r})^\intercal\boldsymbol{x}} - \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal\boldsymbol{x}}\right) c_{\boldsymbol{k}+m\boldsymbol{r}}(f).$$

Consequently, applying the triangle inequality gives

$$|f(\boldsymbol{x}) - f_F(\boldsymbol{x})| \le \sum_{\boldsymbol{k}\in\mathcal{I}_m}\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} 2\cdot|c_{\boldsymbol{k}+m\boldsymbol{r}}(f)|.$$

∎

3.4.1. Gaussian Kernel. We review the theoretical results presented in Potts et al. [65], where the authors consider the Gaussian kernel in two variables and derive an upper bound for $\|\kappa_{\mathrm{ERR}}\|_\infty$. This result can be extended to higher dimensions, where formulas follow the same rules but become somewhat more extensive. In the following theorem we improve the error estimates of Potts et al. [65] and restrict our considerations to the case $d = 3$.

THEOREM 3.2. *Let the kernel matrix be defined by the trivariate Gaussian* $\kappa_s^{\mathrm{gauss}}$ *in* (3.5). *Then, for* $\eta := \frac{\ell\pi m}{\sqrt{2}} \geq 1$ *the following estimate holds true*

$$\|\kappa_{\mathrm{ERR}}\|_\infty \leq 15\gamma(\eta,\ell)\left(\gamma(\eta,\ell)+\frac{5}{2}\right)+102A(\eta,\ell),$$

where

$$\gamma(\eta,\ell) := \ell\sqrt{2\pi}\mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{\eta^2} & : \ell < \frac{1}{2} \\ \dfrac{\ell\mathrm{e}^{-1/2}}{\eta^2} & : \ell \geq \frac{1}{2} \end{cases}$$

and

$$A(\eta,\ell) := \frac{1}{2\eta\sqrt{\pi}}\mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{\sqrt{2}\ell\pi\eta} & : \ell < \frac{1}{2} \\ \dfrac{\sqrt{2}\mathrm{e}^{-1/2}}{\pi\eta} & : \ell \geq \frac{1}{2} \end{cases}.$$

Proof. Throughout this proof we make use of the short hand notation $f(x) := \mathrm{e}^{-x^2/2\ell^2}$. Further, we will employ the following simple estimates

$$\sum_{k=1}^{m/2}\frac{1}{k^2} \leq \sum_{k=1}^{\infty}\frac{1}{k^2} = \frac{\pi^2}{6}, \tag{3.10}$$

$$\sum_{k=1}^{m/2}\mathrm{e}^{-2k^2\pi^2\ell^2} \leq \int_0^\infty \mathrm{e}^{-x^2\pi^2 2\ell^2}\,\mathrm{d}x = \frac{1}{2\ell\sqrt{2\pi}}, \tag{3.11}$$

$$\sum_{k=m/2+1}^{\infty}\frac{1}{k^2} \leq \int_{m/2}^\infty \frac{1}{x^2}\,\mathrm{d}x = \frac{2}{m}, \tag{3.12}$$

$$\sum_{k=m/2+1}^{\infty}\mathrm{e}^{-2k^2\pi^2\ell^2} \leq \int_{m/2}^\infty \mathrm{e}^{-x^2\pi^2 2\ell^2}\,\mathrm{d}x \leq \frac{1}{2\ell^2 m\pi^2}\mathrm{e}^{-\pi^2 m^2\ell^2/2}, \tag{3.13}$$

where the last line follows from

$$\int_a^\infty \mathrm{e}^{-cx^2}\,\mathrm{d}x \leq \int_0^\infty \mathrm{e}^{-c(x+a)^2}\,\mathrm{d}x \leq \mathrm{e}^{-ca^2}\int_0^\infty \mathrm{e}^{-2acx}\,\mathrm{d}x = \frac{\mathrm{e}^{-ca^2}}{2ac}$$

and the estimates (3.11)–(3.13) are simply obtained by estimating the sum from above by an integral over a monotonically decreasing function.

The Fourier transform of the univariate Gaussian is defined as

$$\hat{f}(k) := \int_{-\infty}^{\infty} f(x)\mathrm{e}^{-2\pi\mathrm{i}kx}\,\mathrm{d}x = \ell\sqrt{2\pi}\mathrm{e}^{-2k^2\pi^2\ell^2}. \tag{3.14}$$

By applying integration by parts twice, we obtain for the Fourier coefficients and $k \neq 0$

$$\begin{aligned} c_k(f) &:= \int_{-1/2}^{1/2} f(x) \mathrm{e}^{-2\pi \mathrm{i} k x} \,\mathrm{d}x \\ &= (-1)^{k+1} \frac{1}{4\ell^2 \pi^2 k^2} \mathrm{e}^{-1/8\ell^2} - \frac{1}{4\pi^2 k^2} \int_{-1/2}^{1/2} f''(x) \mathrm{e}^{-2\pi \mathrm{i} k x} \,\mathrm{d}x \\ &= (-1)^{k+1} \frac{1}{4\ell^2 \pi^2 k^2} \mathrm{e}^{-1/8\ell^2} - \frac{1}{4\pi^2 k^2} \int_{-\infty}^{\infty} f''(x) \mathrm{e}^{-2\pi \mathrm{i} k x} \,\mathrm{d}x \\ &\quad + \frac{1}{2\pi^2 k^2} \int_{1/2}^{\infty} f''(x) \cos(2\pi k x) \,\mathrm{d}x, \end{aligned}$$

where $f''(x) = \ell^{-2} \mathrm{e}^{-x^2/2\ell^2} (\ell^{-2} x^2 - 1)$. The second last integral is simply the Fourier transform of f'', which is given by $4\pi^2 k^2 \hat{f}(k)$.

In order to obtain an estimate for $|c_k(f)|$ we may simply use the triangle inequality and it remains to estimate

$$\left| \int_{1/2}^{\infty} f''(x) \cos(2\pi k x) \,\mathrm{d}x \right| \leq \int_{1/2}^{\infty} |f''(x)| \,\mathrm{d}x$$

further. First, we note that f'' changes its sign in $x_0 := \ell$ and, thus, the value of the integral depends on whether $\ell \geq \frac{1}{2}$ or $\ell < \frac{1}{2}$.

If $\ell < \frac{1}{2}$ we compute

$$\int_{1/2}^{\infty} |f''(x)| \,\mathrm{d}x = -f'(\tfrac{1}{2}) = \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}$$

and for $\ell \geq \frac{1}{2}$ it holds

$$\int_{1/2}^{\infty} |f''(x)| \,\mathrm{d}x = f'(\tfrac{1}{2}) - 2f'(\ell) = \frac{2\mathrm{e}^{-1/2}}{\ell} - \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2} \geq 0.$$

Therefore,

$$|c_k(f)| \leq \ell \sqrt{2\pi} \mathrm{e}^{-2\pi^2 \ell^2 k^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2 \pi^2 k^2} & : \ell < \frac{1}{2} \\ \dfrac{\mathrm{e}^{-1/2}}{\pi^2 k^2 \ell} & : \ell \geq \frac{1}{2} \end{cases}.$$

With that, we conclude

$$|c_{m/2}(f)| \leq \ell \sqrt{2\pi} \mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{\eta^2} & : \ell < \frac{1}{2} \\ \dfrac{2\ell \mathrm{e}^{-1/2}}{\eta^2} & : \ell \geq \frac{1}{2} \end{cases} =: \gamma(\eta, \ell).$$

Next, applying (3.10) and (3.11) we compute

$$\sum_{k=1}^{m/2} |c_k(f)| \leq \frac{1}{2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{12\ell^2} \leq \dfrac{1}{4} & : \ell < \frac{1}{2} \\ \dfrac{\mathrm{e}^{-1/2}}{6\ell} \leq \dfrac{\mathrm{e}^{-1/2}}{3} \approx 0.202 & : \ell \geq \frac{1}{2} \end{cases} \leq \frac{3}{4},$$

where we make use of $x^{-2}\mathrm{e}^{-1/8x^2} \leq 8\mathrm{e}^{-1} \approx 2.94 < 3$ for all $x \in \mathbb{R}$. Further, with (3.12) and (3.13) we deduce

$$\sum_{k=m/2+1}^{\infty} |c_k(f)| \leq \frac{1}{2\eta\sqrt{\pi}}\mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{\sqrt{2}\ell\pi\eta} & : \ell < \frac{1}{2} \\ \dfrac{\sqrt{2}\mathrm{e}^{-1/2}}{\pi\eta} & : \ell \geq \frac{1}{2} \end{cases} =: A(\eta,\ell).$$

Note that for $\eta \geq 1$ we obtain $A(\eta,\ell) \leq (2\mathrm{e}\sqrt{\pi})^{-1} + \sqrt{2}\pi^{-1}\mathrm{e}^{-1/2} \approx 0.377$, where we make use of the fact that the function $x^{-1}\mathrm{e}^{-1/8x^2}$ is monotonically increasing on $(0, 1/2)$.

Based on Lemma 3.1 and by exploiting the underlying symmetry we have

$$\begin{aligned}\|\kappa_{\mathrm{ERR}}\|_\infty \leq\, & 2 \cdot \sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_m} |c_{k_1}(f)||c_{k_2}(f)||c_{k_3}(f)| \\ =\, & 2\cdot 3|c_{m/2}(f)|\Big(\sum_{k=-m/2}^{m/2}|c_k(f)|\Big)^2 + 2\cdot 3|c_{m/2}(f)|^2\sum_{k=-m/2}^{m/2}|c_k(f)| \\ & + 2\sum_{\|\boldsymbol{k}\|_\infty \geq m/2+1} |c_{k_1}(f)||c_{k_2}(f)||c_{k_3}(f)| =: 6S_1 + 6S_2 + 2S_3,\end{aligned}$$

where we exploit the tensor product structure $c_{\boldsymbol{k}}(\mathrm{e}^{-\|\boldsymbol{x}\|^2/2\ell^2}) = c_{k_1}(f)c_{k_2}(f)c_{k_3}(f)$. Based on the above derived estimates and by using $|c_0(f)| < 1$ we obtain

$$\begin{aligned} S_1 &\leq \gamma(\eta,\ell)\left(1 + 2\cdot\tfrac{3}{4}\right)^2 = \frac{25}{4}\gamma(\eta,\ell), \\ S_2 &\leq \gamma(\eta,\ell)^2\left(1 + 2\cdot\tfrac{3}{4}\right) = \frac{5}{2}\gamma(\eta,\ell)^2 \end{aligned}$$

and

$$\begin{aligned} S_3 \leq\, & 8\left(\sum_{k=m/2+1}^{\infty}|c_k(f)|\right)^3 + 4\cdot 3\left(\sum_{k=-m/2}^{m/2}|c_k(f)|\right)\left(\sum_{k=m/2+1}^{\infty}|c_k(f)|\right)^2 \\ & + 2\cdot 3\left(\sum_{k=-m/2}^{m/2}|c_k(f)|\right)^2\left(\sum_{k=m/2+1}^{\infty}|c_k(f)|\right) \\ \leq\, & 8A(\eta,\ell)^3 + 12\cdot\frac{5}{2}A(\eta,\ell)^2 + 6\cdot\frac{25}{4}A(\eta,\ell). \end{aligned}$$

Now, we summarize

$$\|\kappa_{\mathrm{ERR}}\|_\infty \leq \frac{75}{2}\gamma(\eta,\ell) + 15\gamma(\eta,\ell)^2 + 16A(\eta,\ell)^3 + 60A(\eta,\ell)^2 + 75A(\eta,\ell).$$

Since $A(\eta,\ell) < \frac{2}{5}$, we have $A(\eta,\ell)^2 < \frac{2}{5}A(\eta,\ell)$ and $A(\eta,\ell)^3 < \frac{4}{25}A(\eta,\ell)$. With that, we obtain a somewhat more simple estimate of the form

$$\|\kappa_{\mathrm{ERR}}\|_\infty < 15\gamma(\eta,\ell)\left(\gamma(\eta,\ell) + \frac{5}{2}\right) + \underbrace{\frac{2539}{25}}_{<102}A(\eta,\ell).$$

∎

We have now established a rigorous error bound for one sub-kernel κ_s^{gauss}. The result for the additive kernel κ follows straightforwardly by applying the bound to each kernel individually.

REMARK 3.3. *Considering the one-dimensional case with* $\kappa^{(1d)}(r) := \mathrm{e}^{-r^2/(2\ell^2)}$ *we obtain*

$$\begin{aligned}\|\kappa_{\text{ERR}}^{(1d)}\|_\infty &\leq 2|c_{m/2}(f)| + 4\sum_{k=m/2+1}^{\infty}|c_k(f)| \\ &\leq 2\gamma(\eta,\ell) + 4A(\eta,\ell),\end{aligned}$$

with $\gamma(\eta,\ell)$ *and* $A(\eta,\ell)$ *as stated above.*

For very small values of the kernel parameter ℓ*, see left plot in Figure 6, the periodized kernel can be considered to be smooth and the exponential decay dominates the error bound. However, the smaller the kernel parameter* ℓ*, the slower the Fourier coefficients decrease to zero. Meaningful values for* m *are limited in order to achieve* $\eta \geq 1$*.*

In the case of moderate values of ℓ*, see second plot in Figure 6, the periodized kernel has a sharp kink and the terms* $\sim \eta^{-2}$ *and* $\sim \eta^{-1}$ *have a greater influence on the estimate. For large values of* ℓ*, see right plot in Figure 6, a constant kernel with zero error is approached, since essentially only* $c_0(f) \neq 0$*.*

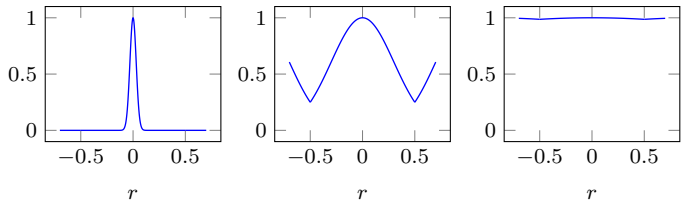

FIG. 6. *Periodized Gaussian kernels in 1D, with parameters* $\ell = 0.03$*,* $\ell = 0.3$ *and* $\ell = 3$ *(from left to right).*

3.4.2. Derivative Gaussian Kernel. For the derivative Gaussian kernel we again consider the case $d = 3$, where

$$\frac{\|\boldsymbol{x}\|_2^2}{2\ell^2}\mathrm{e}^{-\|\boldsymbol{x}\|_2^2/2\ell^2} = \frac{x_1^2}{2\ell^2}\mathrm{e}^{-x_1^2/2\ell^2}\mathrm{e}^{-x_2^2/2\ell^2}\mathrm{e}^{-x_3^2/2\ell^2} + \cdots + \mathrm{e}^{-x_1^2/2\ell^2}\mathrm{e}^{-x_2^2/2\ell^2}\frac{x_3^2}{2\ell^2}\mathrm{e}^{-x_3^2/2\ell^2}$$

and we obtain

$$c_{\boldsymbol{k}}\left(\|\cdot\|_2^2\mathrm{e}^{-\|\cdot\|_2^2/2\ell^2}\right) = c_{k_1}(g)c_{k_2}(f)c_{k_3}(f) + c_{k_1}(f)c_{k_2}(g)c_{k_3}(f) + c_{k_1}(f)c_{k_2}(f)c_{k_3}(g)$$

for the Fourier coefficients, where we define $f(x) := \mathrm{e}^{-x^2/2\ell^2}$ and $g(x) := \frac{x^2}{2\ell^2}\mathrm{e}^{-x^2/2\ell^2}$.

THEOREM 3.4. *Let the kernel matrix be defined by the trivariate derivative Gaussian* κ_s^{der} *in (3.7). Then, for* $\eta := \frac{\ell\pi m}{\sqrt{2}} \geq 1$ *the following estimate holds true*

$$\begin{aligned}\|\kappa_{\text{ERR}}\|_\infty < &\left(\frac{5}{2}\xi(\eta,\ell) + 15\gamma(\eta,\ell)\right)(15 + 12\gamma(\eta,\ell)) \\ &+ 75S(\eta,\ell) + 6A(\eta,\ell)\left(\frac{116}{5}S(\eta,\ell) + 87\right),\end{aligned}$$

where $A(\eta,\ell)$ and $\gamma(\eta,\ell)$ are stated in Theorem 3.2 and

$$\xi(\eta,\ell) \coloneqq \left(\eta^2+\tfrac{1}{2}\right)\ell\sqrt{2\pi}\mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{8\eta^2\ell^2} & : \ell \le \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \\ \dfrac{1}{\eta^2}+\dfrac{3\ell}{2\eta^2} & : \mathit{else} \end{cases},$$

$$S(\eta,\ell) \coloneqq \frac{\operatorname{erfc}(\eta)}{4} + \frac{\eta}{2\sqrt{\pi}}\mathrm{e}^{-\eta^2} + \frac{1}{4\sqrt{\pi}\eta}\mathrm{e}^{-\eta^2} + \begin{cases} \dfrac{\mathrm{e}^{-1/8\ell^2}}{8\sqrt{2}\pi\ell^3\eta} & : \ell \le \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \\ \dfrac{1}{\sqrt{2}\pi\ell\eta}+\dfrac{3}{2\sqrt{2}\pi\eta} & : \mathit{else} \end{cases}.$$

Proof. Throughout this proof we make use of the short hand notations $f(x) = \mathrm{e}^{-x^2/2\ell^2}$ and $g(x) = \frac{1}{2}x^2\ell^{-2}\mathrm{e}^{-x^2/2\ell^2}$, as already introduced above. First, we compute the Fourier transform of the function g. We see $g(x) = \frac{1}{2}\ell^2 f''(x) + \frac{1}{2}f(x)$ and conclude

$$\hat{g}(k) = \int_{-\infty}^{\infty} g(x)\mathrm{e}^{-2\pi\mathrm{i}kx}\mathrm{d}x = (-2\pi^2k^2\ell^2+\tfrac{1}{2})\hat{f}(k), \tag{3.15}$$

where $\hat{f}(k) = \ell\sqrt{2\pi}\mathrm{e}^{-2k^2\pi^2\ell^2}$, as stated in (3.14). In addition to the estimates (3.10)–(3.13), we will make use of the following additional estimates

(3.16)

$$\sum_{k=m/2+1}^{\infty} k^2\mathrm{e}^{-2\pi^2k^2\ell^2} \le \int_{m/2}^{\infty} x^2\mathrm{e}^{-2\pi^2x^2\ell^2}\mathrm{d}x = \frac{1}{\sqrt{2\pi}}\frac{\operatorname{erfc}\left(\pi\ell m/\sqrt{2}\right)}{8\pi^2\ell^3} + \frac{m}{8\pi^2\ell^2}\mathrm{e}^{-\pi^2\ell^2m^2/2},$$

(3.17)

$$\sum_{k=1}^{m/2} k^2\mathrm{e}^{-2\pi^2k^2\ell^2} \le \frac{1}{\sqrt{2\pi}\cdot 8\pi^2\ell^3} + \begin{cases} \dfrac{\mathrm{e}^{-1}}{2\pi^2\ell^2} & : \ell \le \dfrac{1}{\sqrt{2}\pi} \\ \mathrm{e}^{-2\pi^2\ell^2} & : \text{else} \end{cases}.$$

These estimates are obtained as follows. Note that the function $h(x) \coloneqq x^2\mathrm{e}^{-2\pi^2x^2\ell^2}$ with $h'(x) = 2x\mathrm{e}^{-2\pi^2x^2\ell^2}(1-2\pi^2\ell^2x^2)$ is monotonically decreasing for $x \ge (\sqrt{2}\pi\ell)^{-1}$. Thus, presuming $\frac{\ell\pi m}{\sqrt{2}} \ge 1$ we are able to estimate the first sum by the stated integral. The well-known complementary error function is defined by

$$\operatorname{erfc}(x) \coloneqq 1-\operatorname{erf}(x) = 1 - \frac{2}{\sqrt{\pi}}\int_0^x \mathrm{e}^{-t^2}\mathrm{d}t$$

and tends to zero exponentially fast for $x \to \infty$.

The second estimate is obtained as follows. The function $x^2\mathrm{e}^{-2\pi^2x^2\ell^2}$ is monotonically increasing from 0 to $(\sqrt{2}\pi l)^{-1}$, where we may estimate the sum above by the integral of the shifted function. In the area where the function decreases, we obtain an upper estimate by the integral of the function itself. However, the position of the maximum depends on the parameter ℓ. Thus, let $(\sqrt{2}\pi\ell)^{-1} \ge 1 \iff \ell \le (\sqrt{2}\pi)^{-1}$ and $k_\ell \in \mathbb{N}_0$ be the largest natural number $\le (\sqrt{2}\pi l)^{-1}$. Then, we obtain an upper estimate via

$$\int_0^{k_\ell}(x+1)^2\mathrm{e}^{-2\pi^2(x+1)^2\ell^2}\mathrm{d}x + \frac{\mathrm{e}^{-1}}{2\pi^2\ell^2} + \int_{k_\ell+1}^{\infty} x^2\mathrm{e}^{-2\pi^2x^2\ell^2}\mathrm{d}x < \frac{\mathrm{e}^{-1}}{2\pi^2\ell^2} + \int_0^{\infty} x^2\mathrm{e}^{-2\pi^2x^2\ell^2}\mathrm{d}x,$$

which consists of two integrals estimating the sums up to k_ℓ and from $k_\ell + 2$, respectively, plus the area of the rectangle of with 1 and height $(2\pi^2\ell^2\mathrm{e})^{-1}$ being the maximum value of the integrated function.

For large $\ell > (\sqrt{2}\pi)^{-1}$ we obtain the estimate

$$\mathrm{e}^{-2\pi^2\ell^2} + \int_1^\infty x^2\mathrm{e}^{-2\pi^2x^2\ell^2}\mathrm{d}x < \mathrm{e}^{-2\pi^2\ell^2} + \int_0^\infty x^2\mathrm{e}^{-2\pi^2x^2\ell^2}\mathrm{d}x,$$

what is simply the area of the rectangle of width 1 and height $\mathrm{e}^{-2\pi^2\ell^2}$, what is the function value at $x = 1$, plus an integral estimating the remaining sum starting at $k = 2$ from above.

Now, we consider the Fourier coefficients $c_k(g)$, $k \in \mathbb{Z}$, of the 1-periodic continuation of g and state an estimate for the absolute values $|c_k(g)|$. Applying integration by parts two times we obtain

$$\begin{aligned} c_k(g) &= \int_{-1/2}^{1/2} g(x)\mathrm{e}^{-2\pi\mathrm{i}kx}\mathrm{d}x = \left[g'(x)\frac{\mathrm{e}^{-2\pi\mathrm{i}kx}}{4\pi^2k^2}\right]_{-1/2}^{1/2} - \frac{1}{4\pi^2k^2}\int_{-1/2}^{1/2} g''(x)\mathrm{e}^{-2\pi\mathrm{i}kx}\mathrm{d}x \\ &= \frac{(-1)^{k+1}}{4\pi^2k^2\ell^2}\mathrm{e}^{-1/8\ell^2}\left(\frac{1}{8\ell^2} - 1\right) - \frac{\displaystyle\int_{-\infty}^{\infty} g''(x)\mathrm{e}^{-2\pi\mathrm{i}kx}\mathrm{d}x - 2\int_{1/2}^{\infty} g''(x)\cos(2\pi kx)\mathrm{d}x}{4\pi^2k^2}, \end{aligned}$$

where $g''(x) = \ell^{-2}\mathrm{e}^{-x^2/2\ell^2}(1 + x^4/(2\ell^4) - 5x^2/(2\ell^2))$. Thus, by making use of the well-known derivative related properties of the Fourier transform and the triangle inequality, we get

$$|c_k(g)| \le \frac{\mathrm{e}^{-1/8\ell^2}}{32\pi^2k^2\ell^4} + \frac{\mathrm{e}^{-1/8\ell^2}}{4\pi^2k^2\ell^2} + |\hat{g}(k)| + \frac{1}{2\pi^2k^2}\int_{1/2}^{\infty} |g''(x)|\mathrm{d}x,$$

where, $|\hat{g}(k)| \le (2\pi^2k^2\ell^2 + \frac{1}{2})\ell\sqrt{2\pi}\mathrm{e}^{-2k^2\pi^2\ell^2}$, by (3.15) and (3.14).

In order to estimate the last integral, we examine the sign of the function

$$g''(x) = \frac{1}{2\ell^2}\left(\frac{x^4}{\ell^4} - \frac{5x^2}{\ell^2} + 2\right)\mathrm{e}^{-x^2/2\ell^2}.$$

Obviously, $g''(0) > 0$ and changes its sign in the points

$$x_1 \coloneqq \ell\sqrt{\frac{5 - \sqrt{17}}{2}} \text{ and } x_2 \coloneqq \ell\sqrt{\frac{5 + \sqrt{17}}{2}}$$

with $x_1 < x_2$. Depending on whether x_1 and x_2 are smaller or larger than $\frac{1}{2}$ we obtain the following values. If $x_2 < \frac{1}{2} \iff \ell < \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \approx 0.2341$

$$\int_{1/2}^{\infty} |g''(x)|\mathrm{d}x = -g'(\tfrac{1}{2}) = \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(\frac{1}{8\ell^2} - 1\right),$$

where $g'(x) = \frac{1}{2}\ell^{-2}(2 - \ell^{-2}x^2)x\mathrm{e}^{-x^2/2\ell^2}$. In the case $x_1 < \frac{1}{2} < x_2 \iff \ell \in \left(\frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}}, \frac{1}{2}\sqrt{\frac{2}{5-\sqrt{17}}}\right)$

$$\int_{1/2}^{\infty} |g''(x)|\mathrm{d}x = -\int_{1/2}^{x_2} g''(x)\mathrm{d}x + \int_{x_2}^{\infty} g''(x)\mathrm{d}x = \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(1 - \frac{1}{8\ell^2}\right) - 2g'(x_2).$$

Finally, if $x_1 > \frac{1}{2} \iff \ell > \frac{1}{2}\sqrt{\frac{2}{5-\sqrt{17}}} \approx 0.7551$ and splitting the integral into regions of equal sign leads to

$$\int_{1/2}^{\infty} |g''(x)| \mathrm{d}x = 2g'(x_1) - 2g'(x_2) + \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(\frac{1}{8\ell^2} - 1\right).$$

For the derivative evaluated in x_1 and x_2 we compute

$$g'(x_1) = \frac{x_1}{2\ell^2}\left(2 - \frac{5-\sqrt{17}}{2}\right)\mathrm{e}^{(-5+\sqrt{17})/4} \approx 0.83 \cdot \frac{1}{2\ell},$$
$$g'(x_2) = \frac{x_2}{2\ell^2}\left(2 - \frac{5+\sqrt{17}}{2}\right)\mathrm{e}^{(-5-\sqrt{17})/4} \approx -0.56 \cdot \frac{1}{2\ell},$$

and in summary (estimating $0.83 < 0.9$ and $0.56 < 0.6$)

$$\int_{1/2}^{\infty} |g''(x)| \mathrm{d}x \leq \begin{cases} \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(\frac{1}{8\ell^2} - 1\right) & : \ell \leq \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \quad \text{(I)} \\ \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(1 - \frac{1}{8\ell^2}\right) + \frac{3}{5\ell} & : \ell \in \left(\frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}}, \frac{1}{2}\sqrt{\frac{2}{5-\sqrt{17}}}\right) \quad \text{(II)} \\ \frac{\mathrm{e}^{-1/8\ell^2}}{2\ell^2}\left(\frac{1}{8\ell^2} - 1\right) + \frac{3}{2\ell} & : \ell \geq \frac{1}{2}\sqrt{\frac{2}{5-\sqrt{17}}} \quad \text{(III)} \end{cases}.$$

Putting everything together gives

$$\begin{aligned} |c_k(g)| &\leq (2\pi^2k^2\ell^2 + \tfrac{1}{2})\ell\sqrt{2\pi}\mathrm{e}^{-2\pi^2k^2\ell^2} + \frac{1}{2\pi^2k^2} \cdot \begin{cases} \frac{\mathrm{e}^{-1/8\ell^2}}{8\ell^4} & : \text{case (I)} \\ \frac{\mathrm{e}^{-1/8\ell^2}}{\ell^2} + \frac{3}{5\ell} & : \text{case (II)} \\ \frac{\mathrm{e}^{-1/8\ell^2}}{8\ell^4} + \frac{3}{2\ell} & : \text{case (III)} \end{cases} \\ &< (2\pi^2k^2\ell^2 + \tfrac{1}{2})\ell\sqrt{2\pi}\mathrm{e}^{-2\pi^2k^2\ell^2} + \begin{cases} \frac{\mathrm{e}^{-1/8\ell^2}}{16\pi^2k^2\ell^4} & : \ell \leq \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \\ \frac{1}{2\pi^2k^2\ell^2} + \frac{3}{4\pi^2k^2\ell} & : \text{else} \end{cases}, \end{aligned}$$

where we use $x^{-2}\mathrm{e}^{-1/8x^2} < 3$ and $\mathrm{e}^{-1/8x^2} < 1$ for all $x > 0$. It follows

$$\begin{aligned} |c_{m/2}(g)| &< (\tfrac{1}{2}\pi^2m^2\ell^2 + \tfrac{1}{2})\ell\sqrt{2\pi}\mathrm{e}^{-\pi^2\ell^2m^2/2} \\ &\quad + \begin{cases} \frac{\mathrm{e}^{-1/8\ell^2}}{4\pi^2m^2\ell^4} & : \ell \leq \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \\ \frac{2}{\pi^2m^2\ell^2} + \frac{3}{\pi^2m^2\ell} & : \text{else} \end{cases} \\ &= \left(\eta^2 + \tfrac{1}{2}\right)\ell\sqrt{2\pi}\mathrm{e}^{-\eta^2} + \begin{cases} \frac{\mathrm{e}^{-1/8\ell^2}}{8\eta^2\ell^2} & : \ell \leq \frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}} \\ \frac{1}{\eta^2} + \frac{3\ell}{2\eta^2} & : \text{else} \end{cases} \\ &=: \xi(\eta, \ell). \end{aligned}$$

For the sums to be estimated we obtain by the estimates (3.16), (3.13) and (3.12)

$$\begin{aligned}\sum_{k=m/2+1}^{\infty}|c_k(g)| &< \frac{\operatorname{erfc}(\eta)}{4}+\frac{\sqrt{2\pi}m\ell}{4}\mathrm{e}^{-\eta^2}+\frac{\sqrt{2\pi}}{4m\pi^2\ell}\mathrm{e}^{-\eta^2}\\ &\quad+\begin{cases}\dfrac{\mathrm{e}^{-1/8\ell^2}}{8\pi^2\ell^4 m} & :\ell\le\frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}}\\ \dfrac{1}{\pi^2\ell^2 m}+\dfrac{3}{2\pi^2\ell m} & :\text{else}\end{cases}\\ &\le \frac{\operatorname{erfc}(\eta)}{4}+\frac{\eta}{2\sqrt{\pi}}\mathrm{e}^{-\eta^2}+\frac{1}{4\sqrt{\pi}\eta}\mathrm{e}^{-\eta^2}\\ &\quad+\begin{cases}\dfrac{\mathrm{e}^{-1/8\ell^2}}{8\sqrt{2}\pi\ell^3\eta} & :\ell\le\frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}}\\ \dfrac{1}{\sqrt{2}\pi\ell\eta}+\dfrac{3}{2\sqrt{2}\pi\eta} & :\text{else}\end{cases}\\ &=: S(\eta,\ell)\end{aligned}$$

and by (3.17), (3.11) and (3.10)

$$\sum_{k=1}^{m/2}|c_k(g)| < \frac{1}{4}+\begin{cases}\frac{\ell\sqrt{2\pi}}{\mathrm{e}} & :\ell\le\frac{1}{\sqrt{2}\pi}\\ 2\pi^2\ell^3\sqrt{2\pi}\mathrm{e}^{-2\pi^2\ell^2} & :\text{else}\end{cases}+\frac{1}{4}+\begin{cases}\frac{\mathrm{e}^{-1/8\ell^2}}{96\ell^4} & :\ell\le\frac{1}{2}\sqrt{\frac{2}{5+\sqrt{17}}}\\ \frac{1}{12\ell^2}+\frac{1}{8\ell} & :\text{else}\end{cases}.$$

Now, we can simply compute the maximum possible values regarding the single cases and obtain

$$\sum_{k=1}^{m/2}|c_k(g)| < \frac{1}{4}+\frac{1}{2}+\frac{1}{4}+\frac{5}{2}=\frac{7}{2}.$$

The error $\|\kappa_{\mathrm{ERR}}\|_\infty$ can now be estimated by the sum of the non considered Fourier coefficients, see Lemma 3.1. By making use of the underlying symmetry we obtain

$$\begin{aligned}\|\kappa_{\mathrm{ERR}}\|_\infty &\le 2\cdot\sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_m}|c_{k_1}(g)||c_{k_2}(f)||c_{k_3}(f)|+\cdots+|c_{k_1}(f)||c_{k_2}(f)||c_{k_3}(g)|\\ &\le\ 6|c_{m/2}(g)|\Big(\sum_{k=-m/2}^{m/2}|c_k(f)|\Big)^2+12|c_{m/2}(f)|\sum_{k=-m/2}^{m/2}|c_k(g)|\sum_{k=-m/2}^{m/2}|c_k(f)|\\ &\quad+6|c_{m/2}(f)|^2\sum_{k=-m/2}^{m/2}|c_k(g)|+12|c_{m/2}(g)||c_{m/2}(f)|\sum_{k=-m/2}^{m/2}|c_k(f)|\\ &\quad+2\cdot3\cdot\sum_{\|\boldsymbol{k}\|_\infty\ge m/2+1}|c_{k_1}(g)||c_{k_2}(f)||c_{k_3}(f)|\\ &=: 6S_1+12S_2+6S_3+12S_4+6S_5.\end{aligned}$$

We make use of $g(x) \leq \mathrm{e}^{-1}$, implying $|c_0(g)| \leq \mathrm{e}^{-1} < \frac{1}{2}$, and obtain

$$\begin{aligned} S_1 &< \xi(\eta,\ell) \cdot (\tfrac{5}{2})^2, \\ S_2 &< \gamma(\eta,\ell) \cdot \tfrac{15}{2} \cdot \tfrac{5}{2}, \\ S_3 &< \gamma(\eta,\ell)^2 \cdot \tfrac{15}{2}, \\ S_4 &< \xi(\eta,\ell) \cdot \gamma(\eta,\ell) \cdot \tfrac{5}{2} \end{aligned}$$

and

$$\begin{aligned} S_5 = 8 &\left(\sum_{k=m/2+1}^{\infty} |c_k(g)| \right) \left(\sum_{k=m/2+1}^{\infty} |c_k(f)| \right)^2 \\ &+ 4 \left(\sum_{k=-m/2}^{m/2} |c_k(g)| \right) \left(\sum_{k=m/2+1}^{\infty} |c_k(f)| \right)^2 \\ &+ 4 \cdot 2 \cdot \left(\sum_{k=m/2+1}^{\infty} |c_k(g)| \right) \left(\sum_{k=m/2+1}^{\infty} |c_k(f)| \right) \left(\sum_{k=-m/2}^{m/2} |c_k(f)| \right) \\ &+ 2 \left(\sum_{k=-m/2}^{m/2} |c_k(f)| \right)^2 \left(\sum_{k=m/2+1}^{\infty} |c_k(g)| \right) \\ &+ 2 \cdot 2 \left(\sum_{k=-m/2}^{m/2} |c_k(g)| \right) \left(\sum_{k=-m/2}^{m/2} |c_k(f)| \right) \left(\sum_{k=m/2+1}^{\infty} |c_k(f)| \right), \end{aligned}$$

and with the estimates from above

$$\begin{aligned} S_5 &< 8S(\eta,\ell)A(\eta,\ell)^2 + 30A(\eta,\ell)^2 + 20S(\eta,\ell)A(\eta,\ell) \\ &\quad + \frac{25}{2}S(\eta,\ell) + 75A(\eta,\ell) \\ &< \frac{25}{2}S(\eta,\ell) + A(\eta,\ell)\left(\frac{116}{5}S(\eta,\ell) + 87\right), \end{aligned}$$

where we simplify $A(\eta,\ell)^2 < \frac{2}{5}A(\eta,\ell)$, as in the proof of the previous theorem. In summary, the derived estimate reads as

$$\begin{aligned} \|\kappa_{\mathrm{ERR}}\|_\infty &< \frac{5}{2}\xi(\eta,\ell)\left(15 + 12\gamma(\eta,\ell)\right) + 15\gamma(\eta,\ell)\left(15 + 3\gamma(\eta,\ell)\right) \\ &\quad + 75S(\eta,\ell) + 6A(\eta,\ell)\left(\frac{116}{5}S(\eta,\ell) + 87\right) \\ &< \left(\frac{5}{2}\xi(\eta,\ell) + 15\gamma(\eta,\ell)\right)\left(15 + 12\gamma(\eta,\ell)\right) \\ &\quad + 75S(\eta,\ell) + 6A(\eta,\ell)\left(\frac{116}{5}S(\eta,\ell) + 87\right). \end{aligned}$$

□

We have now established a rigorous error bound for the derivative of one sub-kernel κ_s^{der}. The result for the additive kernel κ^{der} follows straightforwardly.

REMARK 3.5. *Considering the one-dimensional case with* $\kappa^{(1d)}(r) = \frac{r^2}{2\ell^2}\mathrm{e}^{-r^2/(2\ell^2)}$ *we obtain*

$$\begin{aligned}\|\kappa_{\mathrm{ERR}}^{(1d)}\|_\infty &\leq 2|c_{m/2}(g)| + 4\sum_{k=m/2+1}^{\infty}|c_k(g)| \\ &\leq 2\xi(\eta,\ell) + 4S(\eta,\ell),\end{aligned}$$

with $\xi(\eta,\ell)$ *and* $S(\eta,\ell)$ *as stated above.*

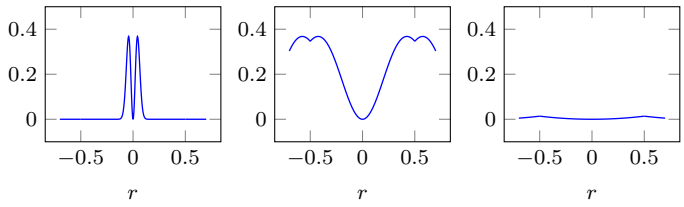

FIG. 7. *Periodized Gaussian derivative kernels in 1D, with parameters* $\ell = 0.03$, $\ell = 0.3$ *and* $\ell = 3$ *(from left to right).*

3.4.3. Comparison of Empirical Approximation Error and Analytical Error Estimates. In order to check if the actual kernel approximation error can indeed be estimated by the error bounds derived above, we perform the following experiments. We generate $N = 10^4$ uniformly distributed random points in $\boldsymbol{r}_j \in [-1/2, 1/2]^3$. Then, we evaluate the kernel functions

$$\kappa^{\mathrm{gauss}}(\boldsymbol{r}) = \mathrm{e}^{-\|\boldsymbol{r}\|^2/2\ell^2} \quad \text{and} \quad \kappa^{\mathrm{deriv}}(\boldsymbol{r}) = \frac{\|\boldsymbol{r}\|^2}{2\ell^2}\mathrm{e}^{-\|\boldsymbol{r}\|^2/2\ell^2}$$

in the points $\boldsymbol{r}_j$, $j = 1, \ldots, N$, for different values of ℓ. In order to approximate the kernel by a trigonometric sum with Fourier coefficients $\hat{c}_{\boldsymbol{k}}$ we evaluate the kernel function on a regular grid of m^3 points in $[-1/2, 1/2]^3$, where we select $m \in \{16, 32, 64\}$. Finally, we evaluate the obtained trigonometric polynomials in the random points $\boldsymbol{r}_j$ and compute the measured worst case error via $\max_{j=1,\ldots,N} |\kappa_{\mathrm{ERR}}(\boldsymbol{r}_j)|$.

In Figure 8 the measured errors for different m and ℓ are represented by the dotted lines. The solid lines show the estimates, as presented in Theorems 3.2 and 3.4. We can see that the estimated errors are indeed below the corresponding estimates, where for some values of ℓ the true error is a few orders of magnitudes smaller than estimated. However, the error behavior is described qualitatively very well by our estimates.

3.4.4. Evaluation of Accuracy. The ultimate goal of the NFFT-accelerated kernel vector multiplication is to obtain fast "accurate" approximate products. For this we want to evaluate the accuracy of the approximation. Let us denote the exact kernel vector product as $p = Kv$, the exact kernel vector product with the Fourier approximation error as $p_E = K_E v$ and the approximate kernel vector product with K_E as $\tilde{p}_E \approx K_E v$. Then, the overall approximation error is determined by

$$|p - \tilde{p}_E| \leq |p - p_E| + |p_E - \tilde{p}_E|.$$

Here, $|p - p_E|$ describes the Fourier approximation error as introduced above in (3.8) and $|p_E - \tilde{p}_E|$ emerges from employing an approximation algorithm, such as the conjugate gradient method for solving with the kernel matrix for instance. Note that an additional approximation error originates from applying the NFFT as already mentioned above.

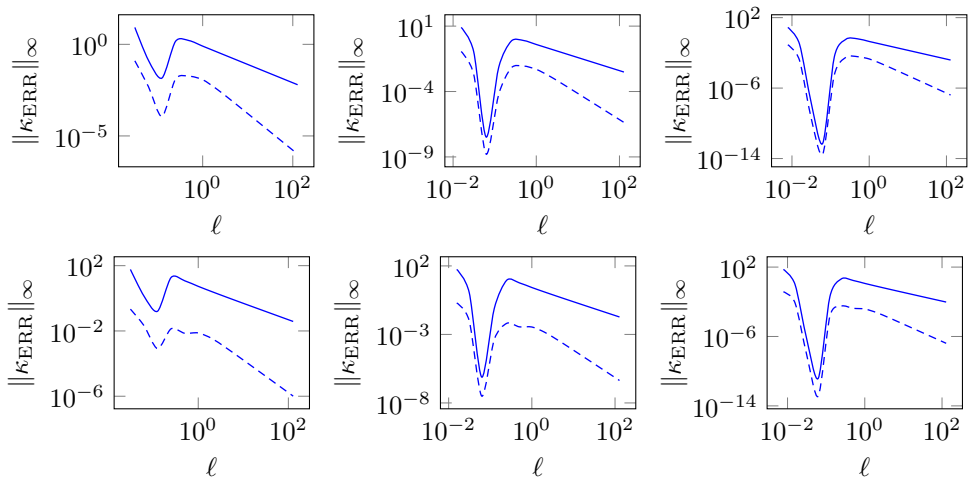

FIG. 8. *Comparison of the measured estimates* $\|\kappa_{ERR}\|_\infty$ *and corresponding error bounds. Results for the Gaussian kernel* $\kappa := \kappa^{gauss}$ *are depicted in the first line, for the Gaussian derivative kernel* $\kappa := \kappa^{deriv}$ *in the second line. The grid size* m *in each direction has been set to* $m = 16$*,* $m = 32$ *or* $m = 64$ *(from left to right).*

4. Global Sensitivity Analysis. The analysis of variance (ANOVA) is a concept studied in the context of statistical methods as well as pure and numerical analysis. In an analytical framework, one may study the so-called classical ANOVA decomposition of functions [10, 52, 47], in order to understand which variables and groups of variables are most important to the function. Expanding a function by using orthonormal systems makes it easy to decompose its variance by means of the basis coefficients, as presented by Potts and Schmischke [63]. We briefly introduce this concept below.

4.1. Sensitivity Analysis in Terms of Fourier Coefficients. We start with some preliminaries and introduce the required notation. In the following, we study periodic functions $f : \mathbb{T}^d \to \mathbb{R}$ on the d-dimensional torus $\mathbb{T} \simeq [-1/2, 1/2)^d$ and denote by

$$[d] \coloneqq \{1, \dots, d\}$$

the set of all dimensions or rather features. As usual, we denote by $\mathcal{P}(S)$ the set of all subsets of a set S.

Subsets of $[d]$, that is elements of $\mathcal{P}([d])$, are denoted by small bold letters $\boldsymbol{u}$. Such a subset is identified with a vector with ascending entries, for example

$$\text{the subset } \boldsymbol{u} = \{1, 4, 3\} \text{ is identified with the vector } \boldsymbol{u} = (1, 3, 4) \in \mathbb{N}^3.$$

For $\boldsymbol{x} \in \mathbb{R}^d$ we denote by $\boldsymbol{x}^{\boldsymbol{u}} \in \mathbb{R}^{|\boldsymbol{u}|}$ the restriction of $\boldsymbol{x}$ to the dimensions present in $\boldsymbol{u}$, for example

$$\boldsymbol{x} = (9, 8, 7, 6, 5), \boldsymbol{u} = \{1, 4, 3\} \Rightarrow \boldsymbol{x}^{\boldsymbol{u}} = (9, 7, 6).$$

Furthermore, we denote by $\mathrm{supp}(\boldsymbol{x})$ the set (or vector) of all dimensions j with $x_j \neq 0$,

for example

$$\boldsymbol{x} = (2,0,0,1,3) \Rightarrow \operatorname{supp}(\boldsymbol{x}) = (1,4,5).$$

Let $\boldsymbol{v}, \boldsymbol{u} \in \mathcal{P}([d])$ with $\boldsymbol{v} \subseteq \boldsymbol{u}$. Then, we define the elements of the vector $\mathcal{F}(\boldsymbol{v}, \boldsymbol{u}) \in \{0,1\}^{|\boldsymbol{u}|}$ via

$$\mathcal{F}(\boldsymbol{v}, \boldsymbol{u})_j = \begin{cases} 1 & : \boldsymbol{u}_j = \boldsymbol{v}_i \text{ for some } i = 1, \dots, |v| \\ 0 & : \text{else} \end{cases},$$

where $j = 1, \dots, |\boldsymbol{u}|$, containing the information which elements of $\boldsymbol{v}$ are also included in $\boldsymbol{u}$. As an example, for $\boldsymbol{v} = (1,4)$ and $\boldsymbol{u} = (1,2,4)$ we obtain $\mathcal{F}(\boldsymbol{v}, \boldsymbol{u}) = (1,0,1)$.

The classical ANOVA decomposition of a function $f \in L_2(\mathbb{T}^d)$ is a unique decomposition of the form

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{u} \subseteq \{1,\dots,d\}} f_{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}) = f_{\emptyset} + f_{\{1\}}(x_1) + f_{\{2\}}(x_2) + \cdots + f_{\{1,\dots,d\}}(\boldsymbol{x}),$$

consisting of 2^d ANOVA terms $f_{\boldsymbol{u}} = f_{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}})$. The ANOVA decomposition is defined in such a way that the single ANOVA terms $f_{\boldsymbol{u}}$ are pairwise orthogonal with respect to the usual L_2 inner product, that is $\langle f_{\boldsymbol{u}}, f_{\boldsymbol{v}} \rangle = \int_{\mathbb{T}^d} f_{\boldsymbol{u}}(\boldsymbol{x}) \overline{f_{\boldsymbol{v}}(\boldsymbol{x})} \mathrm{d}x = 0$ for $\boldsymbol{u} \neq \boldsymbol{v}$. The ANOVA term $f_{\emptyset}$ is a constant, which equals the mean value of the function.

In the special case of a trigonometric polynomial

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathcal{I}_m} \hat{f}_{\boldsymbol{k}} \, \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}}, \tag{4.1}$$

one can show that, confer Potts and Schmischke [63],

$$f_{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}) = \sum_{\substack{\boldsymbol{k} \in \mathcal{I}_m \\ \operatorname{supp}(\boldsymbol{k}) = \boldsymbol{u}}} \hat{f}_{\boldsymbol{k}} \, \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}^\intercal \boldsymbol{x}} = \sum_{\substack{\boldsymbol{k} \in \mathcal{I}_m \\ \operatorname{supp}(\boldsymbol{k}) = \boldsymbol{u}}} \hat{f}_{\boldsymbol{k}} \, \mathrm{e}^{2\pi \mathrm{i} (\boldsymbol{k}^{\boldsymbol{u}})^\intercal \boldsymbol{x}^{\boldsymbol{u}}}. \tag{4.2}$$

From (4.2) we see that an ANOVA term $f_{\boldsymbol{u}}$ includes only the frequencies $\boldsymbol{k}$, which have non-zero entries on the set of indices $\boldsymbol{u}$ and are zero in all dimensions included in $[d] \setminus \boldsymbol{u}$, meaning that the ANOVA decomposition introduces a disjoint decomposition of the trigonometric polynomial (4.1) in terms of its Fourier coefficients.

In order to understand the importance of variables and subsets of variables $\boldsymbol{u}$ to the function, one studies the variance of f and analyzes the contributions of the single ANOVA terms. It is well-known that the variance of a trigonometric polynomial is easily determined by summing over the absolute values of the Fourier coefficients, that is

$$\sigma^2(f) = \sum_{\boldsymbol{k} \in \mathcal{I}_m \setminus \{\boldsymbol{0}\}} \left| \hat{f}_{\boldsymbol{k}} \right|^2.$$

For the single ANOVA terms $f_{\boldsymbol{u}}$ we obtain the same, namely

$$\sigma^2(f_{\boldsymbol{u}}) = \sum_{\substack{\boldsymbol{k} \in \mathcal{I}_m \\ \operatorname{supp}(\boldsymbol{k}) = \boldsymbol{u}}} \left| \hat{f}_{\boldsymbol{k}} \right|^2,$$

so that we conclude

$$\sigma^2(f) = \sum_{\substack{\boldsymbol{u}\subseteq\{1,\dots,d\}\\ \boldsymbol{u}\neq\emptyset}} \sigma^2(f_{\boldsymbol{u}}).$$

Based on that, the so-called global sensitivity indices (GSI), confer Sobol [73, 74] and Potts and Schmischke [63], are defined by

$$\rho_{\boldsymbol{u}}(f) := \frac{\sigma^2(f_{\boldsymbol{u}})}{\sigma^2(f)} \in [0,1],$$

where we may replace the variances by the sums of the corresponding Fourier coefficients, as explained above. Non-important subsets $\boldsymbol{u}$ will not significantly contribute to the overall variance, meaning $\rho_{\boldsymbol{u}}(f) \approx 0$. In contrast, a large GSI is obtained for important $\boldsymbol{u}$.

4.2. Computing the GSI in the Kernel Setting. Now, we consider the matrix vector product (3.1) with coefficients v_j and κ being an additive kernel with windows $\mathcal{W}_s$, $s = 1,\dots,P$, for which $|\mathcal{W}_s| = d_{\max}$ holds true. We obtain

$$\begin{aligned} h(\boldsymbol{x}) &:= \sum_{j=1}^{N} v_j \sum_{s=1}^{P} \kappa_s(\boldsymbol{x}_j^{\mathcal{W}_s}, \boldsymbol{x}^{\mathcal{W}_s}) \\ &\approx \sum_{j=1}^{N} v_j \sum_{s=1}^{P} \sum_{\boldsymbol{k}\in\mathcal{I}_m} \hat{c}_{\boldsymbol{k}}\, \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal(\boldsymbol{x}_j^{\mathcal{W}_s} - \boldsymbol{x}^{\mathcal{W}_s})} \\ &= \sum_{s=1}^{P} \sum_{\boldsymbol{k}\in\mathcal{I}_m} \hat{c}_{\boldsymbol{k}}\, S(\boldsymbol{k},\mathcal{W}_s)\, \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}^\intercal \boldsymbol{x}^{\mathcal{W}_s}} = \tilde{h}(\boldsymbol{x}), \end{aligned} \tag{4.3}$$

where $\mathcal{I}_m \subset \mathbb{Z}^{d_{\max}}$ and

$$S(\boldsymbol{k},\mathcal{W}_s) := \sum_{j=1}^{N} v_j \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}^\intercal \boldsymbol{x}_j^{\mathcal{W}_s}}.$$

Note that exactly the same approximation is used for all windows, that is, the set of Fourier coefficients $\{\hat{c}_{\boldsymbol{k}}\}$ is the same for all $\mathcal{W}_s$. This is possible since all windows have the same length $d_{\max}$ and the same length scale parameter ℓ. The approximation (4.3) is clearly again a trigonometric polynomial and we can now compute the sensitivity indices as explained above. We summarize the procedure of computing the GSI in this setting in the following algorithm.

Note that in Algorithm 4.1 we consider the special case where all given windows in the kernel have exactly the same cardinality, that is $|\mathcal{W}_s| = d_{\max}$. The case $|\mathcal{W}_s| \leq d_{\max}$ can be realized analogously and is not more complicated. The notation will be slightly more complex for this more general case, since the set of Fourier coefficients $\{\hat{c}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_m\}$ differs for windows of different lengths. We would like to mention that the adjoint NFFT that has to be computed in step 2 in the above algorithm is computed using the `pynufft`[3] software package in our Python codes.

[3] https://github.com/jyhmiinlin/pynufft

Algorithm 4.1 Computation of GSI

Input: The set of windows $\mathcal{W}_s \subset \{\boldsymbol{u} \subset [d] : |\boldsymbol{u}| = d_{\max}\}$, $s = 1, \dots, P$, scaled training data $\boldsymbol{x}_j$ with $\|\boldsymbol{x}_j^{\mathcal{W}_s}\| \leq 1/4$, and corresponding coefficients v_j, $j = 1, \dots, N$, superposition dimension $d_{\max}$, length-scale parameter $\ell > 0$ and corresponding kernel Fourier coefficients $\hat{c}_{\boldsymbol{k}}, \boldsymbol{k} \in \mathcal{I}_m \subset \mathbb{Z}^{d_{\max}}$ (precomputed via periodization and FFT).

1. For all $\boldsymbol{u} \in \bigcup_{s=1}^{P} \mathcal{P}(\mathcal{W}_s)$ initialize $\theta_{\boldsymbol{u}} := 0$.
2. For all $s = 1, \dots, P$ do:
   (a) Compute $S(\boldsymbol{k}, \mathcal{W}_s)$, $\boldsymbol{k} \in \mathcal{I}_m$ (this is an adjoint or rather type-2 NFFT).
   (b) For all $\emptyset \neq \boldsymbol{v} \in \mathcal{P}(\mathcal{W}_s)$ compute

$$\theta_{\boldsymbol{v}} = \theta_{\boldsymbol{v}} + \sum_{\mathrm{supp}(\boldsymbol{k}) = \mathrm{supp}(\mathcal{F}(\boldsymbol{v}, \mathcal{W}_s))} |\hat{c}_{\boldsymbol{k}} S(\boldsymbol{k}, \mathcal{W}_s)|^2 .$$

3. Compute the overall variance

$$\sigma^2(\tilde{h}) := \sum_{\emptyset \neq \boldsymbol{v} \in \bigcup_{s=1}^{P} \mathcal{P}(\mathcal{W}_s)} \theta_{\boldsymbol{v}}.$$

4. For all $\emptyset \neq \boldsymbol{v} \in \bigcup_{s=1}^{P} \mathcal{P}(\mathcal{W}_s)$ compute the GSI via

$$\rho_{\boldsymbol{v}}(\tilde{h}) := \frac{\theta_{\boldsymbol{v}}}{\sigma^2(\tilde{h})}.$$

Output: Global sensitivity indices $\rho_{\boldsymbol{v}}(\tilde{h})$ for all $\boldsymbol{v} \in \bigcup_{s=1}^{P} \mathcal{P}(\mathcal{W}_s) \setminus \emptyset$, that is, for all given windows $\mathcal{W}_s$ and all their subsets, except for $\boldsymbol{v} = \emptyset$.

4.3. Variation of the GSI scores. As explained above, sensitivity indices are computed for all subsets of features of cardinality smaller or equal $d_{\max}$. Those subsets are then sorted by their GSI in descending order. Since the sum of those indices is 1 over all feature subsets, we define a $\texttt{GSI}_{\text{score}} \in (0, 1)$ determining how many of those subsets shall be assigned to the feature window. Starting with the subset with the largest GSI, subsets are added to the feature window until the sum of GSI reaches $\texttt{GSI}_{\text{score}}$. The larger the GSI score the more feature subsets are selected. Of course, $\texttt{GSI}_{\text{score}}$ has to be selected carefully in order to obtain good feature windows. The optimal choice can vary for different data sets and is not straightforward. In Figure 9 we compare the RMSE, window size and runtime yielded by models with windows generated for different GSI scores. As expected the RMSE increases when increasing the GSI score up to a certain level. At some point, adding more feature subsets to the window does not lead to better prediction quality. Interestingly, the number of features and windows included in the model is equal for most GSI scores. Only for very large GSI scores, the number of windows goes up steeply. The time for running the model with the corresponding windows behaves accordingly. The prediction quality of the additive model with windows generated with global sensitivity analysis clearly outperforms the full KRR model for the two data sets considered. Note that the performance of sklearn KRR with the full kernel could likely be improved by a more exhaustive grid search. In the following, we set $\texttt{GSI}_{\text{score}} = 0.99$, as we achieve a very high prediction quality with this score and can keep the number of features and windows involved and thus the runtime moderate.

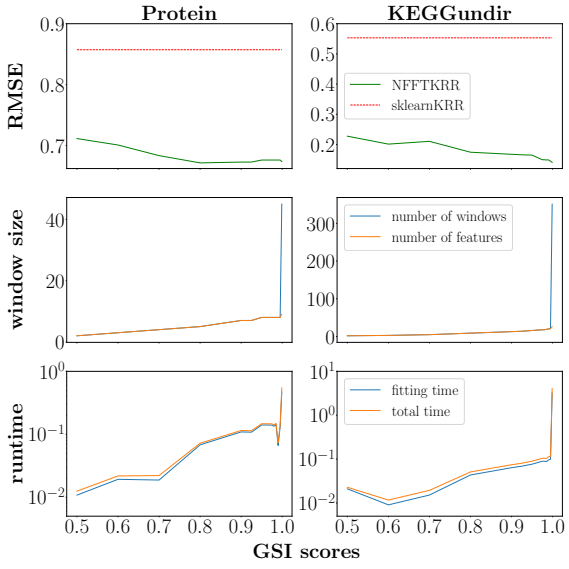

FIG. 9. *Comparison of RMSE, window size and runtime for the additive KRR model for different GSI scores with* $N = 1000$, $d_{max} = 3$, $N_{feat} = d$ *and initial* $\ell = 1$ *and* $\beta = 1$.

5. Numerical Results. To demonstrate the predictive power of the feature arrangement techniques presented above we perform additive kernel ridge regression on benchmark data sets with NFFT-approximations. The corresponding implementations are available in the GitHub repository `NFFTAddKer`, see https://github.com/wagnertheresa/NFFTAddKer. The underlying repository for the fast NFFT-based kernel evaluations is `prescaledFastAdj` as introduced above that accesses parts of the `NFFT` library.

In the following we compare the results of the more sophisticated with the basic techniques to examine whether it is worth putting more effort into the preprocessing phase of learning the windows $\mathcal{W}_s$ and whether additive kernels actually allow for higher accuracy.

Furthermore, we investigate whether the intuition holds true that feature groups covering feature interactions incorporate more information into the model what leads to higher prediction accuracy than groups consisting of single features only.

5.1. Experimental Setup. All experiments were run on a computer with $8 \times$ Intel Core i7 − 7700 CPU @ 3.60 GHz processors with NV106 graphics and 16.0 GiB of RAM. We consider the UCI data sets Protein [68] ($N = 45730$, $d = 9$), KEGGundir [58] ($N = 63608$, $d = 26$) and Bike Sharing [27] ($N = 17379$, $d = 14$) and the StatLib data set Housing [60] ($N = 20640$, $d = 8$). Note that data points with missing entries or entries exceeding the range defined for the feature are dropped in the KEGGundir data set. The data is z-score normalized, the labels are transformed to normalize the target distribution and the data and length-scale parameters are prescaled as described in Subsection 3.2. We perform grid search for the additive kernel ridge regression. All results presented in this paper were generated with the parameter choices listed in Table 1 unless stated otherwise.

General parameter setting	
train–test split	$\texttt{data}_{\text{split}} = 0.5$
signal variance parameter	$\sigma_f = \sqrt{1/P}$
CG convergence tolerance	$\texttt{tol}_{\text{CG}} = 10^{-3}$
NFFT parameter setup	$\texttt{setup}_{\text{NFFT}} =$ "default"
Parameter setting for feature grouping techniques	
subset size for feature grouping	$N^{\text{fg}} = 1000$
subset size for FGO	$N^{\text{fg}}_{\text{FGO}} = 500$
threshold for dropping features	$\texttt{thres} = 0.0$
L1 regularization parameter for lasso	$\beta^{\text{L1}}_{\text{lasso}} = 0.01$
L1 regularization parameter for EN	$\beta^{\text{L1}}_{\text{EN}} = 0.01$
L1 ratio for EN	$\texttt{ratio}^{\text{L1}}_{\text{EN}} = 0.5$
fixed length-scale parameter for FGO	$\ell_{\text{FGO}} = 1$
fixed regularization parameter for FGO	$\beta_{\text{FGO}} = 0.1$
GSI score	$\texttt{GSI}_{\text{score}} = 0.99$
initial length-scale parameter for GSI	$\ell^{\text{init}}_{\text{GSI}} = 1$
initial regularization parameter for GSI	$\beta^{\text{init}}_{\text{GSI}} = 1$
Candidate parameter values for grid search	
length-scale parameter	$\ell \in [10^{-2}, 10^{-1}, 1, 10^1, 10^2]$
regularization parameter	$\beta \in [10^{-2}, 10^{-1}, 1, 10^1, 10^2]$

TABLE 1
Parameter setting for the experiments presented in this paper.

Other parameters that have to be chosen are the maximal length of the windows $d_{\max}$ and the total number of features included N_{feat} that are required for the feature arrangement techniques based on a feature importance ranking. In the remainder of this section we analyze how the choice of these parameters affects the performance of the corresponding regression model. Finally, we compare the feature importance ranking based methods to the approaches based on optimization and global sensitivity analysis.

In this section, we examine the following feature arrangement techniques: consecutive feature grouping (consec), decision tree (DT), mutual information score (MIS), Fisher score (Fisher), RreliefF as filter (relfilt) and wrapper (relwrap) method, lasso, elastic net (EN), feature clustering based on connected components (FC CC), feature grouping optimization (FGO) and global sensitivity indices (GSI).

5.2. Variation of the Maximal Window Length. As motivated above, fast approximation techniques can only exploit their full computational power in small feature spaces. Therefore, a maximal window length $d_{\max}$ must be defined to determine the windows accordingly. For the NFFT-accelerated approximation $d_{\max}$ shall be smaller than 4.

In Figure 10 we analyze the impact of its value on the feature importance ranking based techniques for different arrangement strategies, where the total number of features involved is fixed to two-thirds of d. We compare the RMSE of the additive regression model obtained with the corresponding windows, the time for determining the windows and the mean time for fitting and predicting the model in the grid search routine for the KEGGundir data set. In most RMSE plots the bars shrink the larger $d_{\max}$. While FC CC clearly has the best time for fitting and predicting and yields the second best windows setup time, it cannot keep up with the competitors regarding

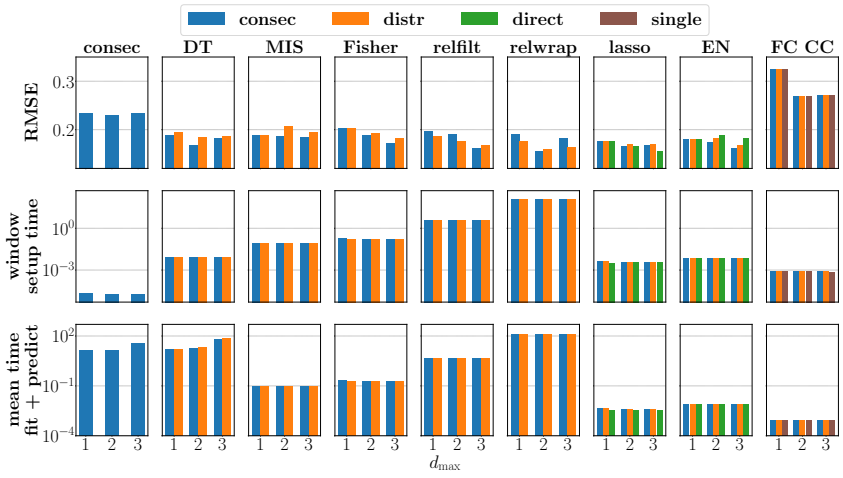

FIG. 10. *Comparison of RMSE, window setup time and time for fitting and predicting the additive KRR model with the corresponding windows for different feature arrangement techniques and strategies, fixed number of total features included $N_{feat} = 2d/3$ and different maximal window length d_{max} for the KEGGundir data set.*

RMSE. relfilt and relwrap take the longest for setting up the windows and are among the slower methods for fitting and predicting. The corresponding RMSE is in the midfield but cannot compensate for the high runtimes however. Lasso and EN provide the second best runtime for fitting and predicting, the third best window setup time and are among the best in RMSE. DT provides one of the best RMSE results and is in the midfield in the window setup time. However, the fitting and predicting takes one of the longest. In comparison, MIS and Fisher yield quite similar RMSE values as DT but take longer for generating the windows. Fitting and predicting is faster by several orders of magnitude though. In total, MIS performs slightly better than Fisher in all categories. Naturally, consec is fastest in determining the windows. The RMSE is far from the best and fitting and predicting is among the slowest. While some feature arrangement strategies beat others in particular techniques, no clear trend of one of them outperforming the others can be identified.

As expected the choice of $d_{\max}$ mostly does not impact the window setup time. However, it generally does not strongly affect the time for fitting and predicting the model either. For most techniques the runtime increases by factor 2 to 4 when changing $d_{\max}$ from 1 to 3, what is barely visible in the figure. The larger $d_{\max}$ the more Fourier coefficients have to be computed per sub-kernel K_s. A smaller value of $d_{\max}$ however leads to a larger number of windows and sub-kernels P for a fixed number N_{feat}. Therefore, both aspects mostly balance each other out.

5.3. Variation of the Total Number of Features Included. The experiment on the maximal window length has illustrated that $d_{\max} = 3$ can be a good choice since it usually yields the smallest RMSE while it only leads to an insignificantly greater computational effort. Next, we investigate how the total number of features included impacts the overall performance for fixed $d_{\max} = 3$.

Figure 11 shows the performance of the feature importance ranking based techniques for different arrangement strategies, fixed $d_{\max} = 3$ and different values $d/3$,

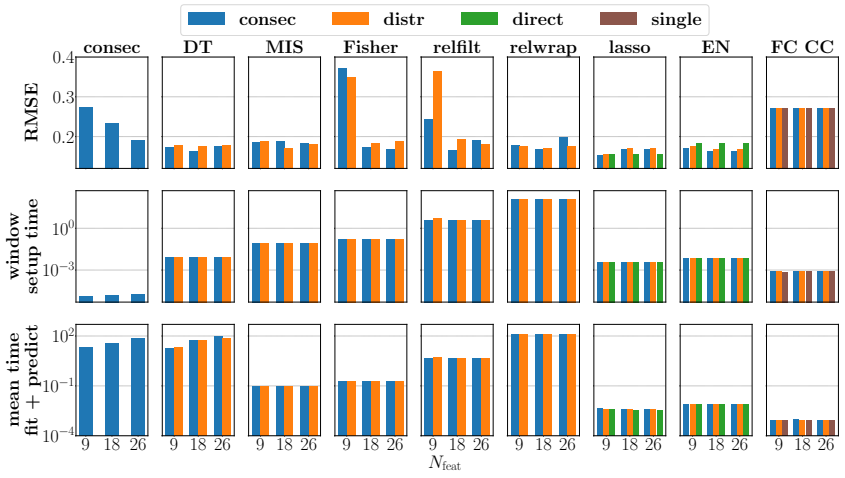

FIG. 11. *Comparison of RMSE, window setup time and time for fitting and predicting the additive KRR model with the corresponding windows for different feature arrangement techniques and strategies, fixed $d_{max} = 3$ and different number of total features included N_{feat} for the KEGGundir data set.*

$2d/3$ and d for N_{feat}. The runtime plots behave similarly as the ones in Figure 10 for the different feature arrangement techniques. Again, we cannot recognize that one of the arrangement strategies is superior to the other ones and the choice of N_{feat} does not seem to have an impact on the time for running the model. For all but one technique, the RMSE is largest for $N_{\text{feat}} = d/3$ and smallest for $N_{\text{feat}} = 2d/3$. In most cases, the RMSE for $N_{\text{feat}} = d$ is either at the same level or larger than for $N_{\text{feat}} = 2d/3$.

5.4. Comparison to GSI, FGO and Full Kernel Ridge Regression. In the previous subsections we observed that using lasso and EN to determine the feature windows usually led to the smallest RMSE. Moreover, the window setup time and the time for fitting and predicting the model with the corresponding windows is superior to most of the other methods. MIS can be considered as the best technique that is not based on a regularization. The only other method that could keep up with MIS is DT that reached similar RMSE. While the window setup time of DT is actually smaller than for MIS, the time for running the model is larger by up to 3 orders of magnitude. Even though none of the feature arrangement strategies is clearly preferable for those techniques, 'distr' might be slightly the best for MIS, 'direct' for lasso and 'consec' for EN.

In Figure 12, we compare those leading feature importance ranking based techniques MIS, lasso and EN to GSI, FGO and the state-of-the-art sklearn kernel ridge regression with the full kernel on 4 benchmark data sets.

Note that other than for the KEGGundir data set $N_{\text{feat}} = d$ can lead to a further RMSE improvement for other data sets, in particular when d is small for instance. Therefore, we choose $N_{\text{feat}} = 2d/3$ for the KEGGundir and Bike Sharing data set and $N_{\text{feat}} = d$ for the Protein and Housing data set. Moreover, we set $d_{\max} = 3$ for the feature importance ranking based techniques and refer to Table 1 for the further parameter setting. For all 4 data sets considered, EN performs better than

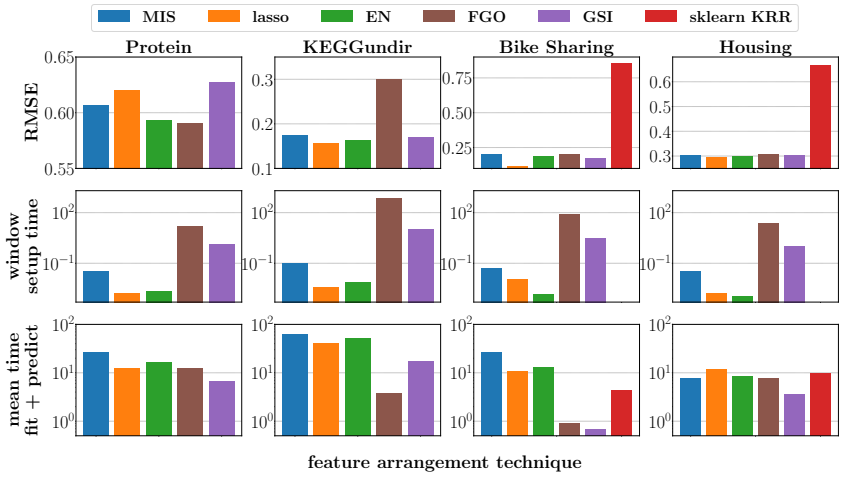

FIG. 12. *Comparison of the feature importance ranking based techniques MIS, lasso and EN with FGO and GSI for the additive KRR model, and sklearn KRR on the full kernel, with* $d_{max} = 3$, $N_{feat}^{Protein} = N_{feat}^{Bike} = 9$, $N_{feat}^{KEGGundir} = 18$, $N_{feat}^{Housing} = 8$, $\beta_{FGO}^{Protein} = 2.5$, $\beta_{FGO}^{KEGGundir} = 0.5$, $\beta_{FGO}^{Bike} = 1.0$, $\beta_{FGO}^{Housing} = 1.5$.

MIS in all three categories. Comparing lasso and EN, we cannot recognize an obvious trend of one outperforming the other. For the Bike Sharing and Housing data, lasso yields a better RMSE than EN but worse runtimes and for the Protein data set EN clearly returns a better RMSE but slightly worse runtimes. As expected, FGO and GSI require by far the longest window setup time but often yield a very competitive runtime for training and fitting the model. Once the windows are set up with these methods running the model is usually quite efficient since FGO and GSI usually return fewer windows of shorter length since they are not affected similarly by the choices of $d_{\max}$ and N_{feat}. However, the RMSE obtained with those windows cannot always keep up with the competitors. Note that a careful adjustment of the model parameters in FGO and GSI can lead to a competitive RMSE for the Protein and KEGGundir data sets. The RMSE obtained with windows generated via GSI is already competitive for the KEGGundir, Bike Sharing and Housing data sets. Since GSI windows usually incorporate many features separately, the intuition that windows with larger d_s yield better RMSE cannot be confirmed in general. The red bar represents the performance of sklearn KRR with the full kernel. The additive models clearly provide a better RMSE that is more than two or three times smaller than for the full KRR model. The time for fitting and predicting the model can be slightly smaller if the number of data points is small such as for the Bike Sharing data set. As motivated above in Figure 3, the computational complexity of NFFT-based additive kernel evaluations is evidently smaller for large scale problems. Note that the red bar is missing in the Protein and KEGGundir plots since the computations break for bigger matrices in the sklearn KRR model due to a lack of memory.

5.5. Extension to Other Kernels. Naturally, the NFFT-accelerated kernel matrix and derivative kernel evaluations and the presented feature arrangement techniques are not only tailored to the Gaussian kernel but can be applied to other kernels

directly. One of the most popular alternatives is the Matérn($\frac{1}{2}$) kernel

$$\kappa^{1/2}_{\text{Matérn}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2}{\ell}\right).$$

Analogously to (1.3) for the additive Gaussian kernel, we can define the Matérn($\frac{1}{2}$) kernel additively as

$$\kappa^{1/2}_{\text{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \sum_{s=1}^{P} \underbrace{\exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell}\right)}_{\kappa^{1/2}_{\text{M}_s}}.$$

In the `prescaledFastAdj` repository $\kappa^{1/2}_{\text{Matérn}}$ is referred to as kernel = 3 and embedded as 'laplacian_rbf' in the underlying `NFFT` repository [43]. Differentiation with respect to the signal variance parameter σ_f gives

$$\frac{\partial K^{1/2}_{\text{M}_{ij}}}{\partial \sigma_f} = 2\sigma_f \sum_{s=1}^{P} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell}\right) = \frac{2}{\sigma_f} K^{1/2}_{\text{M}_{ij}}$$

and with respect to the length-scale parameter ℓ we obtain

$$\frac{\partial K^{1/2}_{\text{M}_{ij}}}{\partial \ell} = \sigma_f^2 \sum_{s=1}^{P} \frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell^2} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell}\right) = \sigma_f^2 \sum_{s=1}^{P} \frac{C_{\text{M}_s}}{\ell^2} \circ K^{1/2}_{\text{M}_{s\,ij}},$$

with $C_{\text{M}_{s\,ij}} = \|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2$. For the latter we added the 'der_laplacian_rbf' kernel

$$\kappa^{1/2}_{\text{derM}_s}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell} \exp\left(-\frac{\|\boldsymbol{x}_i^{\mathcal{W}_s} - \boldsymbol{x}_j^{\mathcal{W}_s}\|_2}{\ell}\right)$$

to the `NFFT` repository that is referred to as kernel = 4 within `prescaledFastAdj`. With that, we obtain $\frac{\partial K^{1/2}_{\text{M}}}{\partial \sigma_f} v = 2\sigma_f \sum_{s=1}^{P} K^{1/2}_{\text{M}_s} v$ and $\frac{\partial K^{1/2}_{\text{M}}}{\partial \ell} v = \sigma_f^2 \sum_{s=1}^{P} \frac{1}{\ell} K^{1/2}_{\text{derM}_s} v$.

In Figure 13 we compare the performance of different feature arrangement techniques for an additive KRR model working with the Matérn($\frac{1}{2}$) instead of the Gaussian kernel as in Figure 12. In comparison to the model with the Gaussian kernel, the Matérn($\frac{1}{2}$) kernel does not lead to huge differences in the runtimes for fitting and predicting the model. The different kernel definition does not modify the MIS, lasso and EN techniques but also for FGO and GSI we cannot recognize huge variations in the window setup time in comparison to Figure 12. The RMSE plots however show greater alternation. Especially for the FGO technique, the RMSE obtained with the Matérn($\frac{1}{2}$) kernel can improve as for the KEGGundir and Bike Sharing data set but also deteriorate as for the Protein data set. The other feature arrangement techniques do not show great variations in performance between the two kernels.

It is also possible to include further kernels such as the Matérn($\frac{3}{2}$) by specifying the function and its derivatives within the `NFFT` package. It remains to derive the corresponding Fourier error estimates in future work.

6. Conclusion. In this paper we have analyzed feature arrangement techniques for additive regression models and their applicability to NFFT-accelerated kernel

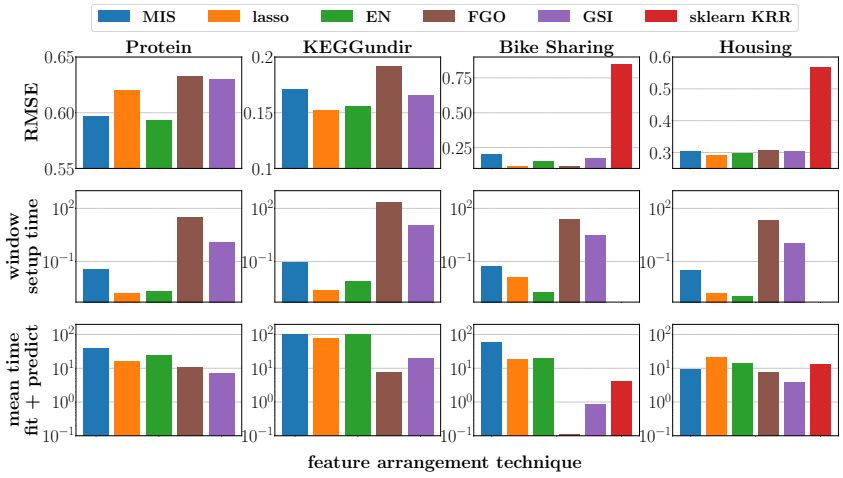

FIG. 13. *Comparison of the feature importance ranking based techniques MIS, lasso and EN with FGO and GSI for the additive KRR model, and sklearn KRR on the full kernel, with* $d_{max} = 3$, $N_{feat}^{Protein} = N_{feat}^{Bike} = 9$, $N_{feat}^{KEGGundir} = 18$, $N_{feat}^{Housing} = 8$, $\beta_{FGO}^{Protein} = 2.5$, $\beta_{FGO}^{KEGGundir} = 0.5$, $\beta_{FGO}^{Bike} = 1.0$, $\beta_{FGO}^{Housing} = 1.5$ *for the Matérn*$(\frac{1}{2})$ *kernel.*

evaluations. We presented several options for splitting the original feature space into smaller feature groups and examined their performance. For simplicity, we demonstrated the numerical results on an additive KRR model, but the computations can easily be applied to other kernel methods. Moreover, we developed an NFFT-acceleration procedure for kernel evaluations with the derivative kernel and motivated its computational power empirically. This is of great relevance in hyperparameter optimization tasks for GPs, for instance. We derived the corresponding Fourier error estimates for the trivariate Gaussian kernel and its derivative kernel analytically and demonstrated its quality. Finally, we compared the additive KRR model to the state-of-the-art sklearn KRR model with the full kernel matrix. In our experiments, the additive model could consistently yield clearly better RMSE while requiring smaller runtimes for fitting and predicting the model if the data is large enough. We mostly focused on the Gaussian kernel in this paper, but briefly motivate the extension to other kernels such as the Matérn$(\frac{1}{2})$ kernel and present first numerical results. It remains to derive additional theoretical guarantees for this kernel in future work.

Acknowledgments. The authors gratefully acknowledge their support from the Bundesministerium für Bildung und Forschung (BMBF) grant 01|S20053A (project SAℓE).

References.

[1] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD International Conference on Management of Data*, pages 94–105, 1998.

[2] Jeonghyun Baek, Jisu Kim, and Euntai Kim. Fast and efficient pedestrian detec-

tion via the cascade implementation of an additive kernel support vector machine. *IEEE Transactions on Intelligent Transportation Systems*, 18(4):902–916, 2016.

[3] Jeonghyun Baek, Junhyuk Hyun, and Euntai Kim. A pedestrian detection system accelerated by kernelized proposals. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1216–1228, 2019.

[4] Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.

[5] Ron Bekkerman, Ran El-Yaniv, Naftali Tishby, and Yoad Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3(Mar):1183–1208, 2003.

[6] Amir Ben-Dor and Zohar Yakhini. Clustering gene expression patterns. In *International Conference on Computational Molecular Biology*, pages 33–42, 1999.

[7] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, 34:483–519, 2013.

[8] Howard D. Bondell and Brian J. Reich. Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. *Biometrics*, 64 (1):115–123, 2008.

[9] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch. Introduction to hierarchical matrices with applications. *Engineering Analysis with Boundary Elements*, 27(5):405–422, 2003.

[10] Russel Caflisch, William Morokoff, and Art Owen. Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. *Journal of Computational Finance*, 1:27–46, 1997.

[11] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[12] Tsz Nam Chan, Zhe Li, Leong Hou U, and Reynold Cheng. Plame: Piecewise-linear approximate measure for additive kernel SVM. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[13] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[14] Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin, and Ghislain Durif. Kernel operations on the GPU, with autodiff, without memory overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021.

[15] Xiaojun Chen, Yunming Ye, Xiaofei Xu, and Joshua Zhexue Huang. A feature group weighting method for subspace clustering of high-dimensional data. *Pattern Recognition*, 45(1):434–446, 2012.

[16] Andreas Christmann and Robert Hable. Consistency of support vector machines using additive kernels for additive models. *Computational Statistics & Data Analysis*, 56(4):854–873, 2012.

[17] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1-4):131–156, 1997.

[18] Zhaohong Deng, Kup-Sze Choi, Fu-Lai Chung, and Shitong Wang. Enhanced soft subspace clustering integrating within-cluster and between-cluster information. *Pattern Recognition*, 43(3):767–781, 2010.

[19] Inderjit S. Dhillon, Subramanyam Mallela, and Rahul Kumar. A divisive information theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, 2003.

[20] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from

microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(02):185–205, 2005.

[21] Carlotta Domeniconi, Dimitrios Gunopulos, Sheng Ma, Bojun Yan, Muna Al-Razgan, and Dimitris Papadopoulos. Locally adaptive metrics for clustering high dimensional data. *Data Mining and Knowledge Discovery*, 14:63–97, 02 2007.

[22] Nicolas Durrande, David Ginsbourger, and Olivier Roustant. Additive kernels for Gaussian process modeling. *arXiv preprint arXiv:1103.4023*, 2011.

[23] Nicolas Durrande, David Ginsbourger, and Olivier Roustant. Additive covariance kernels for high-dimensional Gaussian process modeling. In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, volume 21, pages 481–499, 2012.

[24] David Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge, 2014.

[25] David K. Duvenaud, Hannes Nickisch, and Carl Rasmussen. Additive Gaussian processes. *Advances in Neural Information Processing Systems*, 24, 2011.

[26] Jennifer G. Dy and Carla E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5(Aug):845–889, 2004.

[27] Hadi Fanaee-T. Bike Sharing. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C5W894.

[28] Guojun Gan and Michael Kwok-Po Ng. Subspace clustering with automatic feature grouping. *Pattern Recognition*, 48(11):3703–3713, 2015.

[29] Guojun Gan and Jianhong Wu. A convergence theorem for the fuzzy subspace clustering (FSC) algorithm. *Pattern Recognition*, 41(6):1939–1947, 2008.

[30] Miguel García-Torres, Francisco Gómez-Vela, Belén Melián-Batista, and J. Marcos Moreno-Vega. High-dimensional feature selection via feature grouping: A variable neighborhood search approach. *Information Sciences*, 326:102–118, 2016.

[31] Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011.

[32] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(Mar):1157–1182, 2003.

[33] Lei Han and Yu Zhang. Discriminative feature grouping. In *AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[34] Trevor J. Hastie and Robert Tibshirani. Generalized additive models. *Statistical Science*, 1(3):297–318, 1986.

[35] Marti A. Hearst, Susan T. Dumais, Edgar Osuna, John Platt, and Bernhard Schölkopf. Support vector machines. *IEEE Intelligent Systems and Their Applications*, 13(4):18–28, 1998.

[36] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

[37] Jian Huang, Joel L. Horowitz, and Shuangge Ma. Asymptotic properties of bridge estimators in sparse high-dimensional regression models. *Annals of Statistics*, 36 (2):587 – 613, 2008.

[38] Joshua Zhexue Huang, Michael K. Ng, Hongqiang Rong, and Zichen Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):657–668, 2005.

[39] Anil Jain and Douglas Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

[40] Liping Jing, Michael K. Ng, and Joshua Zhexue Huang. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge and Data Engineering*, 19(8):1026–1041, 2007.

[41] Rebecka Jörnsten and Bin Yu. Simultaneous gene clustering and subset selection for sample classification via MDL. *Bioinformatics*, 19(9):1100–1109, 2003.

[42] Alan Jović, Karla Brkić, and Nikola Bogunović. A review of feature selection methods with applications. In *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1200–1205. IEEE, 2015.

[43] Jens Keiner, Stefan Kunis, and Daniel Potts. Using NFFT3 - a software library for various nonequispaced fast Fourier transforms. *ACM Transactions on Mathematical Software*, 36:Article 19, 1–30, 2009.

[44] Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Machine Learning Proceedings 1992*, pages 249–256. Elsevier, 1992.

[45] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[46] Vipin Kumar and Sonajharia Minz. Feature selection: a literature review. *SmartCR*, 4(3):211–229, 2014.

[47] Frances Y. Kuo, Ian H. Sloan, Grzegorz W. Wasilkowski, and Henryk Woźniakowski. On decompositions of multivariate functions. *Mathematics of Computation*, 79(270):953–966, 2009.

[48] Cihan Kuzudisli, Burcu Bakir-Gungor, Nurten Bulut, Bahjat Qaqish, and Malik Yousef. Review of feature selection approaches based on grouping of features. *PeerJ*, 11:e15666, 2023.

[49] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):1–45, 2017.

[50] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 31(11):4405–4423, 2020.

[51] Huan Liu and Lei Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, 2005.

[52] Ruixue Liu and Art B. Owen. Estimating mean dimensionality of ANOVA decompositions. *Journal of the American Statistical Association*, 101(474):712–721, 2006.

[53] Antoine Liutkus, Derry Fitzgerald, Zafar Rafii, Bryan Pardo, and Laurent Daudet. Kernel additive models for source separation. *IEEE Transactions on Signal Processing*, 62(16):4298–4310, 2014.

[54] Steven Loscalzo, Lei Yu, and Chris Ding. Consensus group stable feature selection. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 567–576, 2009.

[55] Subhransu Maji, Alexander C. Berg, and Jitendra Malik. Efficient classification for additive kernel SVMs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):66–77, 2012.

[56] Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi. Kernel methods through the roof: handling billions of points efficiently. *Advances in Neural Information Processing Systems*, 33:14410–14422, 2020.

[57] Jianyu Miao and Lingfeng Niu. A survey on feature selection. *Procedia Computer Science*, 91:919–926, 2016.

[58] Muhammad Naeem and Sohail Asghar. KEGG Metabolic Reaction Network (Undirected). UCI Machine Learning Repository, 2011. DOI: https://doi.org/10.24432/C5G609.

[59] Franziska Nestler, Martin Stoll, and Theresa Wagner. Learning in high-dimensional feature spaces using ANOVA-based fast matrix-vector multiplication. *Foundations of Data Science*, 4(3):423–440, 2022.
[60] R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.
[61] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.
[62] Gerlind Plonka, Daniel Potts, Gabriele Steidl, and Manfred Tasche. *Numerical Fourier Analysis*. Applied and Numerical Harmonic Analysis. Birkhäuser, 2018.
[63] Daniel Potts and Michael Schmischke. Approximation of high-dimensional periodic functions with Fourier-based methods. *SIAM Journal on Numerical Analysis*, 59(5):2393–2429, 2021.
[64] Daniel Potts and Gabriele Steidl. Fast summation at nonequispaced knots by NFFT. *SIAM Journal on Scientific Computing*, 24(6):2013–2037, 2003.
[65] Daniel Potts, Gabriele Steidl, and Arthur Nieslony. Fast convolution with radial kernels at nonequispaced knots. *Numerische Mathematik*, 98:329–351, 2004.
[66] Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.
[67] Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the Gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41:77–93, 2004.
[68] Prashant Rana. Physicochemical Properties of Protein Tertiary Structure. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C5QW3H.
[69] Yvan Saeys, Inaki Inza, and Pedro Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.
[70] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.
[71] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
[72] Xiaotong Shen and Hsin-Cheng Huang. Grouping pursuit through a regularization solution surface. *Journal of the American Statistical Association*, 105(490): 727, 2010.
[73] Ilya M. Sobol. On sensitivity estimation for nonlinear mathematical models. *Matematicheskoe modelirovanie*, 2(1):112–118, 1990.
[74] Ilya M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Compututers in Simulation*, 55 (1-3):271–280, 2001.
[75] Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13(5), 2012.
[76] Mark Stitson, Alex Gammerman, Vladimir Vapnik, Volodya Vovk, Chris Watkins, and Jason Weston. Support vector regression with ANOVA decomposition kernels. *Advances in Kernel Methods—Support Vector Learning*, pages 285–292, 1999.
[77] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[78] B. Venkatesh and J. Anuradha. A review of feature selection and its methods. *Cybernetics and Information Technologies*, 19(1):3–26, 2019.

[79] Theresa Wagner, John W. Pearson, and Martin Stoll. A preconditioned interior point method for support vector machines using an ANOVA-decomposition and NFFT-based matrix-vector products. *arXiv preprint arXiv:2312.00538*, 2023.

[80] Christopher Williams and Carl Rasmussen. Gaussian processes for regression. *Advances in Neural Information Processing Systems*, 8, 1995.

[81] Christopher K. I. Williams and Carl Edward Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT Press, 2006.

[82] Andrew Wilson and Hannes Nickisch. Kernel interpolation for scalable structured Gaussian processes (kiss-gp). In *International Conference on Machine Learning*, pages 1775–1784. PMLR, 2015.

[83] Zongxia Xie, Yong Xu, and Qinghua Hu. Uncertain data classification with additive kernel support vector machine. *Data & Knowledge Engineering*, 117: 87–97, 2018.

[84] Ke Yan and David Zhang. Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B: Chemical*, 212: 353–363, 2015.

[85] Hao Yang and Jianxin Wu. Practical large scale classification with additive kernels. In *Asian Conference on Machine Learning*, pages 523–538. PMLR, 2012.

[86] Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye. Feature grouping and selection over an undirected graph. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 922–930, 2012.

[87] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *International Conference on Machine Learning*, pages 412–420, 1997.

[88] Yun Yang and Surya T. Tokdar. Minimax-optimal nonparametric regression in high dimensions. *The Annals of Statistics*, 43(2):652–674, 2015.

[89] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[90] Lei Yu, Chris Ding, and Steven Loscalzo. Stable feature selection via dense feature groups. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 803–811, 2008.

[91] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(1):49–67, 2006.

[92] Ling Zheng, Fei Chao, Neil Mac Parthaláin, Defu Zhang, and Qiang Shen. Feature grouping and selection: A graph-based approach. *Information Sciences*, 546: 1256–1272, 2021.

[93] Leon Wenliang Zhong and James T Kwok. Efficient sparse modeling with automatic feature grouping. *IEEE Transactions on Neural Networks and Learning Systems*, 23(9):1436–1447, 2012.

[94] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67 (2):301–320, 2005.