The uniform sparse FFT with application to PDEs with random coefficients

Lutz Kämmerer[*], Daniel Potts[†], Fabian Taubert[‡]

September 10, 2021

We develop an efficient, non-intrusive, adaptive algorithm for the solution of elliptic partial differential equations with random coefficients. The sparse Fast Fourier Transform (sFFT) detects the most important frequencies in a given search domain and therefore adaptively generates a suitable Fourier basis corresponding to the approximately largest Fourier coefficients of the function. Our uniform sFFT (usFFT) does this w.r.t. the stochastic domain simultaneously for every node of a finite element mesh in the spatial domain and creates a suitable approximation space for all spatial nodes by joining the detected frequency sets. This strategy allows for a faster and more efficient computation, than just using other algorithms, like for example the sFFT, for each node separately. We then test the usFFT for different examples using periodic, affine and lognormal random coefficients in the PDE problems. The results are significantly better than when using given standard frequency sets and the algorithm does not require any a priori information about the solution.

Key words. partial differential equation with random coefficient, stochastic differential equation, sparse fast Fourier transform, sparse FFT, lattice rule, periodization, uncertainty quantification, high dimensional trigonometric approximation

AMS subject classifications. 35C09, 35R60, 42B05, 42B37, 60-08, 65C20, 65C30, 65D15, 65T40, 65T50

1 Introduction

Parametric operator equations have gained significant attention in recent years. In particular, partial differential equations with random coefficients play an important role in the study of uncertainty quantification, e.g., [7, 18, 19]. Therefore, the numerical solution of these equations and how to compute them in an efficient and reliable way has become more and more important.

[*] Chemnitz University of Technology, Faculty of Mathematics, 09107 Chemnitz, Germany
kaemmerer@mathematik.tu-chemnitz.de

[†] Chemnitz University of Technology, Faculty of Mathematics, 09107 Chemnitz, Germany
potts@mathematik.tu-chemnitz.de

[‡] Chemnitz University of Technology, Faculty of Mathematics, 09107 Chemnitz, Germany
fabian.taubert@mathematik.tu-chemnitz.de

In this work, we consider the parametric, elliptic problem of finding $u : D \times D_{\boldsymbol{y}} \to \mathbb{R}$ such that for every $\boldsymbol{y} \in D_{\boldsymbol{y}}$ there holds

$$\begin{aligned} -\nabla \cdot (a(\boldsymbol{x},\boldsymbol{y})\nabla u(\boldsymbol{x},\boldsymbol{y})) &= f(\boldsymbol{x}) && \boldsymbol{x} \in D,\, \boldsymbol{y} \in D_{\boldsymbol{y}}, \\ u(\boldsymbol{x},\boldsymbol{y}) &= 0 && \boldsymbol{x} \in \partial D,\, \boldsymbol{y} \in D_{\boldsymbol{y}}, \end{aligned} \tag{1.1}$$

describing the diffusion characteristics of inhomogeneous materials and therefore being called diffusion equations with the random diffusion coefficients a. Here, $\boldsymbol{x} = (x_j)_{j=1}^{d} \in D$ is the spatial variable in a bounded Lipschitz domain $D \subseteq \mathbb{R}^d$, typically with spatial dimension $d = 1, 2$ or 3, and $\boldsymbol{y} = (y_j)_{j=1}^{d_{\boldsymbol{y}}} \in D_{\boldsymbol{y}}$ is a high-dimensional random variable with $D_{\boldsymbol{y}} \subseteq \mathbb{R}^{d_{\boldsymbol{y}}}$. The differential operator ∇ is always used w.r.t. the spatial variable $\boldsymbol{x}$ and the one-dimensional random variables y_j are assumed to be i.i.d. with a prescribed distribution.

A common way to define the random coefficient a is via

$$a(\boldsymbol{x},\boldsymbol{y}) = a_0(\boldsymbol{x}) + \sum_{j=1}^{d_{\boldsymbol{y}}} \Theta_j(\boldsymbol{y})\, \psi_j(\boldsymbol{x}), \tag{1.2}$$

where a_0 and ψ_j are assumed to be uniformly bounded on D. This model is commonly used with the stochastic domain $D_{\boldsymbol{y}} = [\alpha, \beta]^{d_{\boldsymbol{y}}}$, typically with $[\alpha, \beta] = [-1, 1]$ or $[-\frac{1}{2}, \frac{1}{2}]$. The $\Theta_j(\boldsymbol{y})$ can also be interpreted as random variables itself and are usually chosen to fulfill $\mathbb{E}[\Theta_j(\boldsymbol{y})] = 0$, such that $\mathbb{E}[a(\boldsymbol{x}, \cdot)] = a_0(\boldsymbol{x})$ holds and the terms of the sum model the stochastic fluctuations.

Often, the model (1.2) is in an affine fashion, using $\Theta_j(\boldsymbol{y}) = y_j$ for all $j = 1, ..., d_{\boldsymbol{y}}$. This so-called affine model is considered in many works on parametric differential equations with random coefficients, e.g., [10, 25, 33, 37, 14, 26, 2, 13, 4, 16, 30]. The so-called periodic model using $\Theta_j(\boldsymbol{y}) = \frac{1}{\sqrt{6}} \sin(2\pi y_j)$ has been recently studied in [19, 18]. For y_j uniformly distributed on $[-\frac{1}{2}, \frac{1}{2}]$ each, these Θ_j are then distributed according to the arcsine distribution on $[-1, 1]$. It turned out that this model should also be considered in addition to the affine model. Further, this model yields some advantages for our new approach due to its periodicity w.r.t. to the random variables, as we will see later.

The second type of the random coefficient a, that is also used in many recent works, e.g., [17, 8, 3, 1, 5, 29], is the so-called lognormal form

$$a(\boldsymbol{x},\boldsymbol{y}) = a_0(\boldsymbol{x}) + \exp(b(\boldsymbol{x},\boldsymbol{y})), \qquad b(\boldsymbol{x},\boldsymbol{y}) = b_0(\boldsymbol{x}) + \sum_{j=1}^{d_{\boldsymbol{y}}} y_j\, \psi_j(\boldsymbol{x}).$$

Here, the random variables y_j are typically normally distributed, i.e., $y_j \sim \mathcal{N}(0, 1)$, and hence $D_{\boldsymbol{y}} = \mathbb{R}^{d_{\boldsymbol{y}}}$. The numerical analysis as well as the computation of approximations for this model is more difficult, but also arises more often from real applications. A more detailed overview on parametric and stochastic PDEs can be found, e.g., in [9, Sec. 1].

In this paper, we design a numerical method for solving the aforementioned problems. To be more precise, we will compute approximations of the solutions u using trigonometric polynomials. A Fourier approach on ordinary differential equations with high-dimensional random coefficients has already been presented in [6]. There, a dimension-incremental method, the so-called *sparse Fast Fourier Transform (sFFT)*, cf. [32], was used to detect the most important frequencies $\boldsymbol{k}$ and corresponding approximations of the Fourier coefficients $c_{\boldsymbol{k}}(u)$ of the solution $u(\boldsymbol{x}, \boldsymbol{y})$. These values can be used to compute an approximation of the solution

u or other quantities of interest as, e.g., the expectation value $\mathbb{E}[u]$. Further, the frequencies and Fourier coefficients can be used to gain detailed information about the influence of the random variables y_j on the solution u and their interaction with each other.

In this work, we present a non-intrusive approach based on the main idea of the algorithm developed in [6]. The main difference is, that we do not include the spatial variable $\boldsymbol{x}$ in the Fourier approach and therefore only apply the sFFT w.r.t. the random variable $\boldsymbol{y}$. Therefore, the sFFT only needs samples of the function values of u for fixed $\boldsymbol{y}$, which can be computed by using any suitable, already available differential equation solver. In consequence, we are not restricted to particular spatial domains D or spatial dimensions d. To be more precise, we consider a finite set $\mathcal{T}_G \subset D$, $|\mathcal{T}_G| = G < \infty$, as spatial discretization and aim for approximations of the functions $u_{\boldsymbol{x}_g} := u(\boldsymbol{x}_g, \cdot)$ for each $\boldsymbol{x}_g \in \mathcal{T}_G$.

Unfortunately, we would need to apply the whole sFFT algorithm G times separately, resulting in an unnecessary huge increase in the number of samples used and therefore, since each sample implies a call of the underlying, probably expensive differential equation solver, also in computation time of the algorithm. Hence, we develop a modification of the sFFT to overcome this problem and compute the approximations of the functions $u_{\boldsymbol{x}_g}$ within one call of the new algorithm. In particular, our so-called *uniform sparse Fast Fourier Transform (usFFT)* combines the candidate sets between each dimension-incremental step and therefore uses the same sampling nodes $\boldsymbol{y}$ for each point $\boldsymbol{x}_g \in \mathcal{T}$. This strategy manages to keep the number of used samples in a reasonable size and hence decreases the computation time drastically compared to G applications of the sFFT algorithm itself. We summarize this in the following Theorem:

Theorem 1.1. *Let the sparsity $s \in \mathbb{N}$, a frequency candidate set $\Gamma \subset \mathbb{Z}^d$, $|\Gamma| < \infty$, and the parameters $G \in \mathbb{N}$ and $\delta \in (0,1)$ be given. Moreover, we define $N_\Gamma := \max_{j=1,\dots,d}\{\max_{\boldsymbol{k}\in\Gamma} k_j - \min_{\boldsymbol{l}\in\Gamma} l_j\}$. Then, there exists a randomized sampling strategy based on the random rank-1 lattice approach in [22] generating a set S of sampling locations with cardinality*

$$|S| \in \mathcal{O}\left(d\, s\, \max(s, N_\Gamma)\, \log^2 \frac{d\, s\, G\, N_\Gamma}{\delta} + \max(sG, N_\Gamma)\, \log \frac{d\, s\, G}{\delta}\right)$$

such that the following holds.

Consider G arbitrary multivariate trigonometric polynomials $p^{(g)}(\boldsymbol{x}) := \sum_{\boldsymbol{k}\in \mathrm{I}_g} \hat{p}_{\boldsymbol{k}}^{(g)} \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}}$, $g = 1, \dots, G$, where we assume $\mathrm{I}_g \subset \Gamma$, $|\mathrm{I}_g| \leq s$ and $\min_{\boldsymbol{k}\in\mathrm{I}_g} |\hat{p}_{\boldsymbol{k}}^{(g)}| > 0$ for each $g = 1, \dots, G$. We generate a random set S via the sampling strategy. Then, with probability at least $1 - \delta$ it holds that

- *all frequencies $\boldsymbol{k} \in \mathrm{I}_g$ as well as*
- *all Fourier coefficients $\hat{p}_{\boldsymbol{k}}^{(g)}$, $\boldsymbol{k} \in \mathrm{I}_g$,*

of all multivariate trigonometric polynomials $p^{(g)}$, $g = 1, \dots, G$, can be reconstructed from their values at the sampling locations in S.

The simultaneous identification of all the frequencies and the computation of all the Fourier coefficients can be realized by a combination of Algorithm 1 and a slight modification of the approach presented in [22] in the role of Algorithm A. *The suggested method has a computational complexity of*

$$\mathcal{O}\left(d^2\, s^2\, G^2\, N_\Gamma\, \log^3 \frac{d\, s\, G\, N_\Gamma}{\delta}\right)$$

with probability at least $1-\delta$ *as well as* $\mathcal{O}\left(d^2\, s^3\, G^2\, N_\Gamma \log^3 \frac{d\, s\, G\, N_\Gamma}{\delta}\right)$ *in the worst case.*

While Theorem 1.1 is stated for trigonometric polynomials $p^{(g)}$ only, the algorithm can be used on the above mentioned functions $u_{\boldsymbol{x}_g}(\boldsymbol{y})$ to compute the support and values of the approximately largest Fourier coefficients of the functions with some suitable thresholding technique aswell, which is also the key idea when applying the sFFT for function approximation. More generally spoken, we could even consider G different periodic functionals $F_g(\boldsymbol{y})$ and approximate them with the same approach we are about to present here. Moreover, Theorem 1.1 does not assume the frequency sets I_g to share any frequencies $\boldsymbol{k}$, i.e., these sets could even be pairwise disjoint in the worst case scenario. Obviously, this will not be the case in our examples later on as the functions $u_{\boldsymbol{x}_g}(\boldsymbol{y})$ and $u_{\boldsymbol{x}_{\tilde{g}}}(\boldsymbol{y})$ are probably very similar for $\boldsymbol{x}_g$ and $\boldsymbol{x}_{\tilde{g}}$ close to each other due to the smoothness of the solution $u(\boldsymbol{x},\boldsymbol{y})$. Hence, the given complexities, especially the quadratic dependency on G of the computational complexity, are very pessimistic and should really be seen as a worst case estimate.

The crucial advantage of the presented approach is the efficient and adaptive choice of the frequency set performed by the underlying sFFT. Most of the approaches in the aforementioned works are based on certain (tensorized) basis functions [10, 8, 2, 4, 3, 1, 5, 18], Quasi-Monte Carlo methods [25, 33, 17, 8, 14, 26, 13, 16, 19, 30, 29] or collocation methods [8, 37] and often assume the particular involved basis functions or kernels needed to be known in advance. Especially when working with certain weights, e.g., to describe some index sets, slightly modified weights may result in tremendously large or way too small index sets, that are computational unfeasible or not capable of yielding a good approximation, respectively. In other words, reasonably estimating these weights is a particular challenge, which might necessitate considerable additional effort. As an example and for additional information, we refer to [23] as a short and general introduction to Quasi-Monte Carlo methods, which is one of the most used approaches as seen above.

The usFFT only needs a candidate set Γ and selects the important frequencies $\boldsymbol{k}$ in this search domain on its own. Also, the cardinality $|\Gamma|$ of this candidate set is not as problematic as for other approaches, since the number of used samples and the computation time suffer only mildly from larger candidate sets. As mentioned above, we may also extract additional information about the influence and the interactions of the random variables $\boldsymbol{y}_j$ from the output of the usFFT. For instance, we detect a maximum of only 4 simultaneously active dimensions in the detected frequencies in our numerical examples, i.e., the detected frequency vectors $\boldsymbol{k}$ have at most 4 non-zero components with $d_{\boldsymbol{y}} = 10$ or even $d_{\boldsymbol{y}} = 20$.

Another main advantage of our algorithm is the non-intrusive and parallelizable behavior. As already mentioned, the usFFT uses existing numerical solvers of the considered differential equation. We can use suitable, reliable and efficient solvers with no need to re-implement them. Further, the different samples needed in each sampling step can be computed on multiple instances. This parallelization allows to reduce the computation time even further and makes a higher number of used samples less time consuming.

The remainder of the paper is organized as follows:

In Section 2 we set up some notation and assumptions and briefly explain the key idea of the sFFT algorithm. Section 3 is devoted to the explanation of the usFFT as well as some periodizations required for the affine and lognormal cases. Finally, in Section 4 we test the new algorithm on different examples using periodic, affine and lognormal random coefficients and investigate the computed approximations under different aspects.

The MATLAB® source code of the algorithm as well as demos for our numerical examples can be downloaded from `https://www-user.tu-chemnitz.de/~tafa/software/software.php`.

2 Prerequisites

We consider the PDE problem (1.1). Note, that we always assume f to be independent of the random variable $\boldsymbol{y}$ and zero boundary conditions just for simplicity and to preserve clarity. Our algorithm (up to some minor changes) may also be applied for right-hand sides $f(\boldsymbol{x},\boldsymbol{y})$ as well as non-zero Dirichlet boundary conditions $u(\boldsymbol{x},\boldsymbol{y}) = h(\boldsymbol{x},\boldsymbol{y})$ for all $\boldsymbol{x} \in \partial D$.

2.1 Problem setting

The weak formulation of our problem reads: Given $f \in H^{-1}(D)$, for every $\boldsymbol{y} \in D_{\boldsymbol{y}}$, find $u(\cdot,\boldsymbol{y}) \in H_0^1(D)$, such that

$$\int_D a(\boldsymbol{x},\boldsymbol{y})\nabla u(\boldsymbol{x},\boldsymbol{y}) \cdot \nabla v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_D f(\boldsymbol{x})v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \forall v \in H_0^1(D).$$

As usual, $H_0^1(D)$ denotes the subspace of the L_2-Sobolev space $H^1(D)$ with vanishing trace on ∂D and $H^{-1}(D)$ denotes the dual space of $H_0^1(D)$. We say, that the diffusion coefficient $a : D \times D_{\boldsymbol{y}} \to \mathbb{R}$ fulfills the uniform ellipticity assumption, if there exist two constants $a_{\min} \in \mathbb{R}$ and $a_{\max} \in \mathbb{R}$, such that

$$0 < a_{\min} \leq a(\boldsymbol{x},\boldsymbol{y}) \leq a_{\max} < \infty \qquad \forall \boldsymbol{x} \in D,\, \forall \boldsymbol{y} \in D_{\boldsymbol{y}}. \tag{2.1}$$

Then, the Lax-Milgram Lemma ensures, that the problem (1.1) possesses a unique solution $u(\cdot,\boldsymbol{y}) \in H_0^1(D)$ for every fixed $\boldsymbol{y} \in D_{\boldsymbol{y}}$, satisfying the a priori estimate

$$\sup_{\boldsymbol{y}\in D_{\boldsymbol{y}}} ||u(\cdot,\boldsymbol{y})||_{H_0^1(D)} \leq \frac{1}{a_{\min}} ||f||_{H^{-1}(D)}.$$

Some further basic information and results on approximation and smoothness of the solution u of high-dimensional parametric PDEs can be found in [9, Sec. 1 and 2]. Additionally, we also refer to the general results on best n-term approximations given in [9, Sec. 3.1], since our Fourier approach fits in this particular framework as well.

In order to compute an approximation of the solution $u_{\boldsymbol{x}_g} := u(\boldsymbol{x}_g,\cdot)$ at a given point $\boldsymbol{x}_g \in D$ using the dimension-incremental method explained below, we need samples of $u_{\boldsymbol{x}_g}$ for a lot of sampling nodes $\boldsymbol{y}$. We aim for a non-intrusive approach and therefore use a finite element method to solve the problem (1.1) for a given $\boldsymbol{y} \in D_{\boldsymbol{y}}$. A similar approach is used e.g. in [30, 29], where the finite element method is used to solve the PDE for any $\boldsymbol{y}_{\mathfrak{u}}$ with $\mathfrak{u} \subset \mathbb{N}$ and $(\boldsymbol{y}_{\mathfrak{u}})_j = y_j$ for $j \in \mathfrak{u}$ and 0 otherwise. The corresponding approximations of the so-called $\mathfrak{u}$-truncated solution are then used for their particular method aswell. In our case, we just evaluate the finite element solution $\check{u}(\cdot,\boldsymbol{y})$ at the given point $\boldsymbol{x}_g \in D$. In particular, instead of the finite element method, any differential equation solver would fit, that is capable of computing the value $u(\boldsymbol{x}_g,\boldsymbol{y})$ for given $\boldsymbol{x}_g$ and $\boldsymbol{y}$. Hence, we also refer to this sampling method as black box sampling later on.

Note, that we will use the finite element solution $\check{u}$ also as an approximation of the true solution u, when we test the accuracy of our computed approximation $u^{\mathtt{usFFT}}$ in Section 4. In detail, we have

$$\mathrm{err}(u, u^{\texttt{usFFT}}) \leq \mathrm{err}(u, \check{u}) + \mathrm{err}(\check{u}, u^{\texttt{usFFT}}),$$

where $\mathrm{err}(\cdot,\cdot)$ is a suitable metric, symbolizing the error. So while we only investigate the second term $\mathrm{err}(\check{u}, u^{\texttt{usFFT}})$ in our numerical tests later, the first term includes other error sources as the modeling, e.g., by a dimension truncation of infinite-dimensional random coefficient a, or the error coming from the finite element approximation itself. For a particular example of this, we refer to the detailed error analysis for the periodic model mentioned in Section 1, that is given in [18, Sec. 4].

2.2 The dimension-incremental method

The following dimension-incremental method was presented in [32]. The aim of this algorithm is to determine the non-zero Fourier coefficients $\hat{p}_{\boldsymbol{k}} \in \mathbb{C}$, $\boldsymbol{k} \in \mathrm{I}$, of a multivariate trigonometric polynomial

$$p(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathrm{I}} \hat{p}_{\boldsymbol{k}} \exp(2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x})$$

with unknown frequency set $\mathrm{I} \subset \mathbb{Z}$, $|\mathrm{I}| < \infty$, based on samples. Obviously, p is a periodic signal and its domain is the d-dimensional torus $\mathbb{T}^d$, $\mathbb{T} \simeq [0,1)$.

The goal is not only to calculate the nonzero Fourier coefficients $\hat{p}_{\boldsymbol{k}}$ but also to detect the frequencies $\boldsymbol{k}$ out of a possibly huge search domain $\Gamma \subset \mathbb{Z}^d$ belonging to the nonzero Fourier coefficients. In particular, we define the set $\operatorname{supp}\hat{p} := \{\boldsymbol{k} \in \Gamma \colon \hat{p}_{\boldsymbol{k}} \neq 0\}$. Assuming $I \subset \Gamma$ and the set $\operatorname{supp}\hat{p}$ is known, the computation of the corresponding Fourier coefficients $\hat{p}_{\boldsymbol{k}}$, $\boldsymbol{k} \in \mathrm{I}$, can be performed efficiently using various, well-known Fourier methods.

First, we introduce some further notation as in [32]. We consider a given search domain $\Gamma \subset \mathbb{Z}^d$, $|\Gamma| < \infty$, that should be large enough to contain the unknown frequency set $\mathrm{I} \subset \Gamma$. We denote the projection of a frequency $\boldsymbol{k} := (k_1, ..., k_d)^\top \in \mathbb{Z}^d$ to the components $\boldsymbol{i} := (i_i, ..., i_m) \in \{\iota \in \{1, ..., d\}^m : \iota_t \neq \iota_{t'} \text{ for } t \neq t'\}$ by $\mathcal{P}_{\boldsymbol{i}}(\boldsymbol{k}) := (k_{i_1}, ..., k_{i_m})^\top \in \mathbb{Z}^m$. Correspondingly, we define the projection of a frequency set $\mathrm{I} \subset \mathbb{Z}^d$ to the components $\boldsymbol{i}$ by $\mathcal{P}_{\boldsymbol{i}}(\mathrm{I}) := \{(k_{i_1}, ..., k_{i_m}) : \boldsymbol{k} \in \mathrm{I}\}$. Using these notations, the general approach is the following:

1. Determine the first components of the unknown frequency set with some sampling values, i.e., determine a set $\mathrm{I}^{(1)} \subseteq \mathcal{P}_1(\Gamma)$ which should be identical to the projection $\mathcal{P}_1(\mathrm{supp}\ \hat{p})$ or contain this projection, $\mathrm{I}^{(1)} \supseteq \mathcal{P}_1(\mathrm{supp}\ \hat{p})$.

2. For dimension increment step $t = 2, ..., d$, i.e., for each additional dimension:

    a) Determine the t-th components of the unknown frequency set with some sampling values, i.e., determine a set $\mathrm{I}^{(t)} \subseteq \mathcal{P}_t(\Gamma)$ which should be identical to the projection $\mathcal{P}_t(\mathrm{supp}\ \hat{p})$ or contain this projection, $\mathrm{I}^{(t)} \supseteq \mathcal{P}_t(\mathrm{supp}\ \hat{p})$.

    b) Determine a suitable sampling set $\mathcal{X}^{(1,...,t)} \subset \mathbb{T}^d$, $|\mathcal{X}^{(1,...,t)}| \ll |\Gamma|$, which allows to detect those frequencies from the set $(\mathrm{I}^{(1,...,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,...,t)}(\Gamma)$ belonging to non-zero Fourier coefficients $\hat{p}_{\boldsymbol{k}}$.

    c) Sample the trigonometric polynomial p along the nodes of the sampling set $\mathcal{X}^{(1,...,t)}$.

    d) Compute the Fourier coefficients $\tilde{\hat{p}}_{(1,...,t),\boldsymbol{k}}$, $\boldsymbol{k} \in (\mathrm{I}^{(1,...,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,...,t)}(\Gamma)$.

e) Determine the non-zero Fourier coefficients from $\tilde{\hat{p}}_{(1,\dots,t),\boldsymbol{k}}$, $\boldsymbol{k} \in (\mathrm{I}^{(1,\dots,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,\dots,t)}(\Gamma)$ and obtain the set $\mathrm{I}^{(1,\dots,t)}$ of detected frequencies. The $\mathrm{I}^{(1,\dots,t)}$ index set should be equal to the projection $\mathcal{P}_{(1,\dots,t)}(\mathrm{supp}\ \hat{p})$.

3. Use the set $\mathrm{I}^{(1,\dots,d)}$ and the computed Fourier coefficients $\tilde{\hat{p}}_{(1,\dots,d),\boldsymbol{k}}$, $\boldsymbol{k} \in \mathrm{I}^{(1,\dots,d)}$ as an approximation for the support supp $\hat{p}$ and the Fourier coefficients $\hat{p}_{\boldsymbol{k}}$, $\boldsymbol{k} \in \mathrm{supp}\ \hat{p}$.

Note, that this method can also be used for the numerical determination of the approximately largest Fourier coefficients

$$c_{\boldsymbol{k}}(f) := \int_{\mathbb{T}^d} f(\boldsymbol{x}) \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{x}} \mathrm{d}\boldsymbol{x}, \quad \boldsymbol{k} \in \mathrm{I},$$

of suffciently smooth periodic signals f using suitable thresholding techniques.

The proposed approach includes the construction of suitable sampling sets in step 2b. To this end, one assumes that an upper bound $s \geq |\,\mathrm{supp}\,\hat{p}|$ is known and one constructs the sampling sets $\mathcal{X}^{(1,\dots,t)}$ such that the Fourier coefficients $\tilde{\hat{p}}_{(1,\dots,t),\boldsymbol{k}}$ computed in step 2d are randomly projected ones. Due to that projection one may observe cancellations with the effect that one misses active frequencies. For that reason, one repeats the computation of the projected Fourier coefficients for a number r of random projections.

Of course, there exist different methods for the computation of the projected Fourier coefficients. The algorithm should work with any sampling method, which reliably computes Fourier coefficients on a given frequency set. Preferable sampling sets combine the four properties:

- relatively low number of sampling nodes,
- stability and thus reliability,
- efficient construction methods,
- fast Fourier transform algorithms.

Popular approaches with such properties are for example based on so-called rank-1 lattices. A rank-1 lattice (R1L) is a set

$$\Lambda(\boldsymbol{z}, M) := \left\{ \frac{i}{M}\boldsymbol{z} \bmod \boldsymbol{1} : i = 0, ..., M-1 \right\}$$

with a so-called generating vector $\boldsymbol{z} \in \mathbb{Z}^d$ and lattice size $M \in \mathbb{N}$. In [32], single rank-1 lattices (single R1Ls) were used as sampling strategy in the dimension-incremental method and provided a perfectly stable, reliable and efficient way to reconstruct the projected Fourier coefficients $\tilde{\hat{p}}_{(1,\dots,t),\boldsymbol{k}}$. In [21] and [22], other approaches based on multiple rank-1 lattices (multiple R1Ls) and random rank-1 lattices (random R1Ls) have been studied. The main advantage of these approaches is a smaller size and a significantly faster construction of the involved sampling sets $\mathcal{X}$, but therefore they involve some failure probability, which is not needed for the perfectly stable single R1L approach. Table 2.1 shows the sampling and arithmetic complexities of the dimension-incremental method when using these different sampling strategies based on R1Ls. Further notes on these approaches and their behavior when used in the dimension-incremental method can be found in the referred works.

Table 2.1: Sampling and arithmetic complexities of the sFFT approach (with high probability, cf. [21, 22]) when using different sampling strategies based on R1Ls, where $\Gamma \subset [-N,N]^d$, $s \geq |\operatorname{supp}\hat{p}|$, and r is the number of random projections computed in each dimenion-incremental step.

	samples	arithmetic operations
single R1Ls [32]	$\mathcal{O}(dr^3s^2N)$	$\mathcal{O}(dr^3s^3 + dr^3s^2N\,\log^{\mathcal{O}(1)}(...))$
multiple R1Ls [21]	$\mathcal{O}(dr^2sN\,\log^{\mathcal{O}(1)}(...))$	$\mathcal{O}(d^2r^2sN\,\log^{\mathcal{O}(1)}(...))$
random R1Ls [22]	$\mathcal{O}(drs\,\log^{\mathcal{O}(1)}(...))$	$\mathcal{O}(d^2rsN\,\log^{\mathcal{O}(1)}(...))$

3 The uniform sparse FFT

Up to now, the sFFT algorithm is a suitable tool in order to compute an approximation of the solution

$$u_{\boldsymbol{x}_g}(\boldsymbol{y}) \coloneqq u(\boldsymbol{x}_g,\boldsymbol{y}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^d} c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g})\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{y}} \approx \sum_{\boldsymbol{k}\in\mathrm{I}_{\boldsymbol{x}_g}} c_{\boldsymbol{k}}^{\mathtt{sFFT}}(u_{\boldsymbol{x}_g})\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{y}}$$

for a single $\boldsymbol{x}_g$. When considering a whole set of points $\boldsymbol{x}_g \in \mathcal{T}_G$, $|\mathcal{T}_G| = G$, we have to call the existing method G times. But multiple, independent calls of the sFFT result in different, adaptively determined sampling sets $\mathcal{X}^{(1,\dots,t)}$ in step 2b). Hence, we cannot guarantee that the solutions of the differential equation from one run of the algorithm can be utilized in another one. So we really need G full calls of the sFFT including all sampling computations and therefore end up with unnecessary many samples, even when using the sample efficient R1L approaches. Remember, that sampling means solving the differential equation with a call of the underlying differential equation solver, that might be very expensive in computation time. Therefore, we now modify the dimension-incremental method, such that we can work on the set $\mathcal{T}_G$ and one call of the algorithm computes approximations of the most important Fourier coefficients $c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g})$, $\boldsymbol{k} \in I_{\boldsymbol{x}_g}$, for each $g = 1,...,G$, including a clever choice of the sampling nodes $\boldsymbol{y}$.

3.1 Expanding the sFFT

We force the dimension-incremental method to select a set I containing the frequencies of the s approximately largest Fourier coefficients $c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g})$ for each $\boldsymbol{x}_g \in \mathcal{T}_G$. Therefore, we compute the set of detected frequencies $\mathrm{I}_{\boldsymbol{x}_g}^{(1,\dots,t)}$ for each $\boldsymbol{x}_g$ in each dimension-increment t, but after that we form the union of these sets $\bigcup_{g=1}^{G} \mathrm{I}_{\boldsymbol{x}_g}^{(1,\dots,t)}$, which will be the set of detected frequencies $\mathrm{I}^{(1,\dots,t)}$ that is given to the next dimension-incremental step $t+1$. So we start each iteration with a larger frequency candidate set $(\mathrm{I}^{(1,\dots,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,\dots,t)}(\Gamma)$, which is suitable for all $\boldsymbol{x}_g \in \mathcal{T}_G$. This way, the first t components of the elements of the sampling set $\mathcal{X}^{(1,\dots,t)}$ are the same for each $\boldsymbol{x}_g$ and the random part, which causes the specific random projection of the Fourier coefficients, can be chosen equally for each $\boldsymbol{x}_g$ without disturbing the algorithm. So now we can take advantage of the fact, that our underlying differential equation solver can evaluate the solutions $u(\boldsymbol{x},\boldsymbol{y})$ for a given $\boldsymbol{y}$ for multiple values of $\boldsymbol{x}$ in the domain D. Accordingly, we only need to solve the differential equation once for each sampling node $\boldsymbol{y}$ and still get all the sampling values $u_{\boldsymbol{x}_g}(\boldsymbol{y})$ for all $\boldsymbol{x}_g$ in our finite set $\mathcal{T}_G$. Note, that this also holds for the one-dimensional detections in steps 1 and 2a. The deterministic parts of the used sampling sets there are independent of the particular function $u_{\boldsymbol{x}_g}$ and the random part can be chosen equally for each $\boldsymbol{x}_g$ again. Also note, that we might need to interpolate or approximate, if some of the values $u_{\boldsymbol{x}_g}(\boldsymbol{y})$, $\boldsymbol{x}_g \in \mathcal{T}_G$ and fixed $\boldsymbol{y} \in \mathcal{X}^{(1,\dots,t)}$, are not directly given by the differential equation solver. Obviously, the larger candidate sets $(\mathrm{I}^{(1,\dots,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,\dots,t)}(\Gamma)$, resulting from the union of the sets $\mathrm{I}_{\boldsymbol{x}_g}^{(1,\dots,t)}$, will also result in larger sampling sets. The overall increase of sampling locations considered is still very reasonable, cf. Theorem 1.1. The computational complexity suffers a bit harder from these modifications, but is not as important as the amount of sampling locations and the estimate given in Theorem 1.1 is probably very pessimistic anyway, cf. Remark 3.1. We could also think of further thresholding methods to cut the number of frequencies back to the sparsity s at the end of each dimension-incremental step or at least at the end of the whole algorithm. In this work, we will not do this, but take a look at the total number of frequencies in the output of the algorithm in relation to the sparsity s, cf. Remark 4.1.

Overall, the proposed method is now capable of computing approximations

$$u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y}) := \sum_{\boldsymbol{k}\in\mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g}) \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{y}}$$

for all nodes $\boldsymbol{x}_g, g = 1, ..., G$, simultaneously. We will call this modified version of the sFFT the *uniform sFFT* or short *usFFT* from now on, where *uniform* is meant w.r.t. the discrete set of points $\mathcal{T}_G$. The full method is stated in Algorithm 1. Note, that the notations in Algorithm 1 are taken from the original works on the sFFT [32], [21] and [22] to set the focus on the modifications and therefore slightly differ from our here used notations. In detail, the Algorithm is stated for a trigonometric polynomial $p(\chi, \boldsymbol{x})$, which will be the function $u(\boldsymbol{x}, \boldsymbol{y})$ in our case, computes the index set $\tilde{\mathrm{I}}$, which is the index set I in our notations, and the corresponding Fourier coefficients $\tilde{\hat{\boldsymbol{p}}}_g$, which we denoted as $c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})$ above. Note, that Algorithm 1 is the same as [22, Alg. 3] up to the gray highlighted modifications.

Proof of Theorem 1.1. Note, that in the following explanations as well as in the corresponding Tables 3.1 and 3.1 'sample complexities' always refers to the cardinality of the set of sampling locations, since we assume the black box sampling algorithm to provide the samples for all G trigonometric polynomials $p^{(g)}$ simultaneously.

Theorem 1.1 is a slight modification of [22, Thm. 2]. To be more precise, we apply Algorithm 1 using a random R1L approach in the role of Algorithm A and a spatial discretization $\mathcal{X}$ based on multiple R1Ls, cf. [20], in step 3. In addition, we assume $s_{\text{local}} \lesssim s$. The crucial difference to [22, Thm. 2] is that we have to take account of the modification that we demand for reconstructing not only one but even G different trigonometric polynomials with possibly differing frequency supports I_g, $g = 1, \dots, G$.

Accordingly, we mainly refer to the analysis of the sample complexity and computational complexity of the sFFT using random R1Ls given in [22, Sec. 3.2] as well as the therefor necessary theoretical results from [21, Sec. 4]. In the following, we discuss the necessary modifications on the bounds and parameter choices discussed, proved and used in [22, Sec. 3.2.2 and 3.2.3] and [21, Lem. 4.4 and Thm. 4.6], such that the corresponding results hold. The modifications when considering the usFFT can be separated in two different parts, first the possibly larger candidate sets J_t and $\mathrm{I}^{(1,\dots,d)}$ in steps 2 and 3, respectively, and second the modified failure probability. Note, that r and $\gamma_\star$ are the decisive parameters which provide estimates on the failure probability later on and, thus, both are discussed in the second part of the proof.

Algorithm 1 The usFFT on a set $\mathcal{T}_G$

```
Input:  $\Gamma \subset \mathbb{Z}^d$          search space in frequency domain, candidate set for $\mathrm{I} = \operatorname{supp} \hat{p}$
        $p(\circ, \circ)$          trigonometric polynomial $p$ as black box (function handle)
        $\mathcal{T}_G$                 discrete set containing the points $\chi_g$, $g = 1, ..., G$
        $s, s_{\mathrm{local}} \in \mathbb{N}$        sparsity parameter, $s \leq s_{\mathrm{local}}$
        Algorithm A          **efficient algorithm** A that guarantees the identification of the frequency support of each $s_{\mathrm{local}}$-sparse trigonometric polynomial w.h.p., cf. Section 2.2, and computes the Fourier coefficients
        $\theta \in \mathbb{R}^+$            absolute threshold
        $r \in \mathbb{N}$             number of detection iterations
(Step 1 & 2a)  [Single frequency component identification]
  for $t := 1, \ldots, d$ do
     Set $K_t := \max(\mathcal{P}_t(\Gamma)) - \min(\mathcal{P}_t(\Gamma)) + 1$, $\mathrm{I}^{(t)} := \emptyset$.
     for $i := 1, \ldots, r$ do
        Choose $x'_j \in \mathbb{T}$, $j \in \{1, \ldots, d\} \setminus \{t\}$ uniformly at random.
        Set $\boldsymbol{x}^{(\ell)} := \left(x_1^{(\ell)}, \ldots, x_d^{(\ell)}\right)^\top$, $x_j^{(\ell)} := \begin{cases} \ell/K_t, & j = t, \\ x'_j, & j \neq t, \end{cases}$   for all $\ell = 0, \ldots, K_t - 1$.
        for $g := 1, \ldots, G$ do
           Compute $\tilde{\hat{p}}_{t,k_t,g} := \frac{1}{K_t} \sum_{\ell=0}^{K_t-1} p\left(\chi_g, \boldsymbol{x}^{(\ell)}\right) \mathrm{e}^{-2\pi \mathrm{i} \ell k_t / K_t}$, $k_t \in \mathcal{P}_t(\Gamma)$, via FFT.
           Set $\mathrm{I}^{(t)} := \mathrm{I}^{(t)} \cup \{k_t \in \mathcal{P}_t(\Gamma) \colon \tilde{\hat{p}}_{t,k_t,g}$ is among the largest $s_{\mathrm{local}}$ (in absolute value)
                      elements of $\{\tilde{\hat{p}}_{t,j,g}\}_{j \in \mathcal{P}_t(\Gamma)}$ and $|\tilde{\hat{p}}_{t,k_t,g}| \geq \theta\}$.
        end for $g$
     end for $i$
  end for $t$
(Step 2)  [Coupling frequency components identification]
  for $t := 2, \ldots, d$ do
     If $t < d$, set $\tilde{r} := r$ and $\tilde{s} := s_{\mathrm{local}}$, otherwise $\tilde{r} := 1$ and $\tilde{s} := s$. Set $\mathrm{I}^{(1,\ldots,t)} := \emptyset$.
     for $i := 1, \ldots, \tilde{r}$ do
        Choose components $x'_{t+1}, \ldots, x'_d \in \mathbb{T}$ of sampling nodes uniformly at random.
(Step 2b)
        Generate a sampling set $\mathcal{X} \subset \mathbb{T}^t$ for $\mathrm{J}_t := (\mathrm{I}^{(1,\ldots,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,\ldots,t)}(\Gamma)$ that allows for the application of Algorithm A. Set $\mathcal{X}_{t,i} := \{\boldsymbol{x} := (\tilde{\boldsymbol{x}}, x'_{t+1}, \ldots, x'_d) \colon \tilde{\boldsymbol{x}} \in \mathcal{X}\} \subset \mathbb{T}^d$.
(Step 2c)
        Sample $p$ along the nodes of the sampling set $\mathcal{X}_{t,i}$ for every $\chi_g$.
        for $g := 1, \ldots, G$ do
(Step 2d)
           Apply Algorithm A to obtain the support $\tilde{\mathrm{J}}_{t,i,g} \subset \mathrm{J}_t$, $|\tilde{\mathrm{J}}_{t,i\,g}| \leq \tilde{s}$, of frequencies belonging to the at most $\tilde{s}$ largest Fourier coefficients, each larger than $\theta$ in absolute value, using the sampling values $p(\chi_g, \boldsymbol{x}_j)$, $\boldsymbol{x}_j \in \mathcal{X}_{t,i}$.
(Step 2e)
           Set $\mathrm{I}^{(1,\ldots,t)} := \mathrm{I}^{(1,\ldots,t)} \cup \tilde{\mathrm{J}}_{t,i,g}$.
        end for $g$
     end for $i$
  end for $t$
```

```
Algorithm 1 continued.
(Step 3) [Computation of Fourier coefficients]
  Generate a sampling set $\mathcal{X} \subset \mathbb{T}^d$ for $\mathrm{I}^{(1,\dots,d)}$ such that the corresponding Fourier matrix
  $\boldsymbol{A}(\mathcal{X}, \mathrm{I}^{(1,\dots,d)})$ is of full column rank and its pseudoinverse can be applied efficiently.
  for $g := 1, \dots, G$ do
    Compute the corresponding Fourier coefficients $\left(\tilde{\hat{p}}_{(1,\dots,d),\boldsymbol{k},\,g}\right)_{\boldsymbol{k}\in\mathrm{I}^{(1,\dots,d)}}$.
  end for $g$
  Set $\tilde{\mathrm{I}} := \mathrm{I}^{(1,\dots,d)}$
 Output: $\tilde{\mathrm{I}} \subset \Gamma \subset \mathbb{Z}^d$            set of detected frequencies
         $\tilde{\hat{\boldsymbol{p}}}_{g} \in \mathbb{C}^{|\tilde{\mathrm{I}}|}$          corresponding Fourier coefficients, $|\tilde{\hat{p}}_{(1,\dots,d),\boldsymbol{k},\,g}| \geq \theta$ for all $\chi_g$
```

We start with the candidate sets J_t in step 2b of Algorithm 1 and observe, that the cardinality $|\mathrm{J}_t|$ of the set of frequency candidates

$$\mathrm{J}_t = (\mathrm{I}^{(1,\dots,t-1)} \times \mathrm{I}^{(t)}) \cap \mathcal{P}_{(1,\dots,t)}(\Gamma) \subset \left(\bigcup_{\substack{i=1,\dots,\tilde{r} \\ g=1,\dots,G}} \tilde{J}_{t-1,i,g} \times \mathcal{P}_t(\Gamma) \right) \cap \mathcal{P}_{(1,\dots,t)}(\Gamma)$$

in each dimension increment t can be simply bounded by $|\mathrm{J}_t| \lesssim r\,s\,G\,N_\Gamma$, which contains an additional factor G compared to [22, Sec. 3.2.2]. Applying the sampling strategy suggested in [21, Sec. 2.1] together with [21, Lem. 4.5] directly yields $|\mathcal{X}_{t,i}| \lesssim \max(s, N_\Gamma)\log(|\mathrm{J}_t|/\gamma_{\mathrm{A}}) \lesssim \max(s, N_\Gamma)\log(r\,s\,G\,N_\Gamma/\gamma_{\mathrm{A}})$.

Further, the cardinality of the finally detected frequency set $\mathrm{I}^{(1,\dots,d)}$ that we are using in step 3 of our algorithm is bounded from above by $s\,G$ due to the same argumentation, since $\tilde{r} = 1$ and $\tilde{s} = s$ when $t = d$ holds in step 2. Accordingly, we can apply [21, Alg. 1] in order to construct a spatial discretization $\mathcal{X}$ of $\mathrm{I}^{(1,\dots,d)}$ based on multiple R1Ls which has a cardinality bounded by $|\mathcal{X}| \lesssim \max(s\,G, N_\Gamma)\log(s\,G/\gamma_{\mathrm{B}})$, where γ_{B} is the failure probability of the construction of this spatial discretization, cf. [20, Thm. 4.1].

Moreover, we need to discuss the computational complexities of the individual steps of the usFFT. Obviously, step 1 applies $d\,r\,G$ different one-dimensional FFTs of lengths at most N_Γ, which yields a computational complexity in $\mathcal{O}\left(d\,r\,G\,N_\Gamma\log(N_\Gamma)\right)$. In step 2, we apply $((d-2)\,r+1)\,G$ times [22, Alg. 4], where the sampling set is a union of $L \in \mathcal{O}\left(\log(r\,s\,G\,N_\Gamma/\gamma_{\mathrm{A}})\right)$ R1Ls of size at most in $\mathcal{O}\left(s\right)$ and the input set of frequencies J_t is bounded from above by $\mathcal{O}\left(r\,s\,G\,N_\Gamma\right)$ in its cardinality, which yields an arithmetic complexity in

$$\begin{aligned}\mathcal{O}\left(d\,r\,G(\max(s, N_\Gamma)\log(s\,N_\Gamma) + d\,r\,s\,G\,N_\Gamma)\log(r\,s\,G\,N_\Gamma/\gamma_{\mathrm{A}})\right)& \\ \subset \mathcal{O}\left(d^2\,r^2\,s\,G^2\,N_\Gamma\log^2(r\,s\,N_\Gamma\,G/\gamma_{\mathrm{A}})\right)&\end{aligned}$$

in worst case (w.c.). Moreover, we observe $|\mathrm{J}_t| \lesssim s\,G\,N_\Gamma$ with a certain probability since the signals p are all trigonometric polynomials and in the case where Algorithm A does not fail in any case, we have $\bigcup_{i=1,\dots,\tilde{r}} \tilde{\mathrm{J}}_{t,i,g} \subset \mathrm{I}_g^{(1,\dots,t)}$ with $|\mathrm{I}_g^{(1,\dots,t)}| \leq I^{(1,\dots,d)} \leq s$. As a consequence, we save a linear r and the r in the log term compared to the worst case arithmetic complexity, cf. also [22, Sec. 3.2.2] for a similar argumentation. In addition, the same argumentation saves a factor r in the logarithmic term of the upper bound on the number of sampling locations in step 2 with the same probability. Later, we specifically choose the parameters r and $\gamma_\star$ such

Table 3.1: Sample complexities and computational complexities for the different steps of Algorithm 1, where the efficient identification by [22, Alg. 4] is used in Step 2 and the multiple R1L approach from [21, Alg. 1] in Step 3.

	sample complexity	computational complexity
Step 1	$d\,r\,N_\Gamma$	$d\,r\,G\,N_\Gamma\,\log N_\Gamma$
Step 2 (w.h.p.)	$d\,r\,\max(s, N_\Gamma)\,\log \frac{s\,G\,N_\Gamma}{\gamma_A}$	$d^2\,r\,s\,G^2\,N_\Gamma\,\log^2 \frac{s\,G\,N_\Gamma}{\gamma_A}$
Step 2 (w.c.)	$d\,r\,\max(s, N_\Gamma)\,\log \frac{r\,s\,G\,N_\Gamma}{\gamma_A}$	$d^2\,r^2\,s\,G^2\,N_\Gamma\,\log^2 \frac{r\,s\,G\,N_\Gamma}{\gamma_A}$
Step 3	$\max(s\,G, N_\Gamma)\,\log \frac{s\,G}{\gamma_\mathrm{B}}$	$G\,\max(s\,G, N_\Gamma)\,\log \frac{s\,G}{\gamma_\mathrm{B}}\,(d + \log(s\,G\,N_\Gamma))$

that the estimates hold with high probability - for that reason, we call these complexities w.h.p. complexities already here.

For computing the G FFTs of step 3 we apply [21, Alg. 2], which yields a computational complexity in

$$\begin{aligned}\mathcal{O}\left(G\log(s\,G/\gamma_\mathrm{B})\big(\max(s\,G, N_\Gamma)\log(s\,G\,N_\Gamma) + s\,G(d+\log(s\,G))\big)\right) \\ \subset \mathcal{O}\left(G\,\max(s\,G, N_\Gamma)\,\log(s\,G/\gamma_\mathrm{B})\,(d+\log(s\,G\,N_\Gamma))\right).\end{aligned}$$

The sample complexities and computational complexities of the usFFT due to these modifications are given in Table 3.1, see [22, Tab. 3.2] for comparison to the sFFT. Here, the only changes are several appearances of the parameter G, i.e., for $G = 1$ we observe the complexities of the sFFT.

We continue with the aforementioned second part, where we need to discuss suitable choices of r, γ_A, and γ_B, cf. also [22, Sec. 3.2.3]. First, we consider the failure probability of the so-called projections, i.e., the failure that may occur due to the fact that projected Fourier coefficients are close to zero and, thus, not detectable, cf. also [21, Lem. 7].

In particular, the number r of iterations determines how many of these projections are considered and, certainly, the more projections are considered the less is the probability that a specific projected Fourier coefficient is small for all different projections.

Let us consider a single trigonometric polynomial $p \not\equiv 0$ with $\min_{\boldsymbol{h}\in\operatorname{supp}\hat{p}} |\hat{p}| \geq 3\theta$ and $\Gamma \supset \operatorname{supp}\hat{p}$, $|\operatorname{supp}\hat{p}| \leq s$. Choosing

$$r = \lceil 2s(\log 3 + \log d + \log s + \log G - \log\delta)\rceil$$

yields a probability of at most $\frac{\delta}{3\,d\,s\,G}$ that all the projected Fourier coefficients are less than θ for at least one frequency. Considering G different of such trigonometric polynomials $p^{(g)}$ and applying the union bound yields that the probability that all the projected Fourier coefficients are less than θ for at least one frequency and at least one signal is bounded by $\frac{\delta}{3\,d\,s}$.

Moreover, we determine the parameter γ_A, which is in fact the failure probability of [22, Alg. 4] in the role of Algorithm A. When choosing $\gamma_\mathrm{A} := \frac{\delta}{3\,d\,s\,G}$, we observe a probability that at least one of the G applications of Algorithm A fails in step 2d (for fixed t and i) is bounded from above by $\frac{\delta}{3\,d\,s}$. Last, we fix the parameter $\gamma_B := \frac{\delta}{3\,d}$, i.e., the failure probability of step 3 is bounded from above by γ_B, cf. [20, Thm. 4.1].

Altogether, the total failure probability can be estimated via union bound similar to [21, Thm. 9] and is bounded by terms less than δ, cf. also [22, Sec. 3.2.3].

Finally, the sample complexity and computational complexity stated in Theorem 1.1 now follow with the same argumentations as in [22, Sec. 3.2.3] using the above discussed choices

Table 3.2: Sample complexities and computational complexities for the different steps of Algorithm 1, where the efficient identification by [22, Alg. 4] is used in Step 2 and $r = \lceil 2\, s \log(\frac{3\, d\, s\, G}{\delta})\rceil$ is chosen in line with [21, Thm. 4.6]. Step 3 is realized via the multiple R1L approach of [21, Alg. 1].

	sample complexity	computational complexity
Step 1	$d\, s\, N_\Gamma \log \frac{d\, s\, G}{\delta}$	$d\, s\, G\, N_\Gamma \log^2 \frac{d\, s\, G\, N_\Gamma}{\delta}$
Step 2 (w.h.p.)	$d\, s \max(s, N_\Gamma) \log^2 \frac{d\, s\, G\, N_\Gamma}{\delta}$	$d^2\, s^2\, G^2\, N_\Gamma \log^3 \frac{d\, s\, G\, N_\Gamma}{\delta}$
Step 2 (w.c.)	$d\, s \max(s, N_\Gamma) \log^2 \frac{d\, s\, G\, N_\Gamma}{\delta}$	$d^2\, s^3\, G^2\, N_\Gamma \log^3 \frac{d\, s\, G\, N_\Gamma}{\delta}$
Step 3	$\max(s\, G, N_\Gamma) \log \frac{d\, s\, G}{\delta}$	$G \max(s\, G, N_\Gamma) \log \frac{d\, s\, G}{\delta} \, (d + \log(s\, G\, N_\Gamma))$

$r = \lceil 2\, s \log(\frac{3\, d\, s\, G}{\delta})\rceil$, $\gamma_A := \frac{\delta}{3\, d\, s\, G}$, and $\gamma_B := \frac{\delta}{3\, d}$. The precise complexities for each step are given in Table 3.1. Again, the only changes to [22, Tab. 3.3] are the additional appearances of G. ■

Remark 3.1. *The sample complexity of Step 2 is the dominating term for the sFFT. When we are talking about the usFFT, G appears linearly in the sample complexity of Step 3, such that it might not be neglectable as in the sFFT case for large G. However, if we can bound G for example by $G \lesssim d\, s$, the sample complexity of Step 3 is again asymptotically smaller than for Step 2. Even more, since G appears only in logarithmic terms of the sample complexity of Step 2, we see, that the overall sample complexity of the usFFT is the same as for the sFFT in this case, i.e., the number of sampling locations is bounded in $\mathcal{O}\left(d\, s \max(s, N_\Gamma) \log^2 \frac{d\, s\, N_\Gamma}{\delta}\right)$ when assuming $G \lesssim d\, s$, cf. also [22, Thm. 1.3] for comparison. This is an important observation, since the amount of sampling locations is the crucial factor for the overall computational complexity of our algorithm due to the expensiveness of the underlying sampling algorithm, i.e., the PDE solver.*

3.2 Periodization

The usFFT allows us to reconstruct a frequency set I and approximations $c_{\boldsymbol{k}}^{\mathtt{usFFT}}(u_{\boldsymbol{x}_g})$ of the corresponding Fourier coefficients $c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g})$ for each $\boldsymbol{x}_g \in \mathcal{T}_G$. Unfortunately, this approach requires the function $u(\boldsymbol{x}, \boldsymbol{y})$ to be 1-periodic w.r.t. $\boldsymbol{y}$ in each stochastic dimension $d_{\boldsymbol{y}}$.

Since our right-hand side $f(\boldsymbol{x})$ does not depend on $\boldsymbol{y}$ in our considerations, the random coefficient a is the only given function involving the random variable $\boldsymbol{y}$ in our problem (1.1). In periodic models, we use the random coefficient (1.2) with 1-periodic functions $\Theta_j(\boldsymbol{y})$. Hence, the random coefficient $a(\boldsymbol{x}, \boldsymbol{y})$ is 1-periodic and thus the solution $u(\boldsymbol{x}, \boldsymbol{y})$ is also 1-periodic w.r.t. each component of $\boldsymbol{y}$. Therefore, we can apply the usFFT directly for this model without any further considerations.

In order to apply the usFFT when using the affine and lognormal models, we need to apply a suitable periodization first, since the random coefficient a and therefore the solution u are not periodic in general. Note, that we assume the random variable to be uniformly distributed in the affine case, i.e., $\boldsymbol{y} \sim \mathcal{U}([\alpha, \beta]^{d_{\boldsymbol{y}}})$, and standard normally distributed in the lognormal case, i.e., $\boldsymbol{y} \sim \mathcal{N}(0, 1)^{d_{\boldsymbol{y}}}$.

3.2.1 Affine case

We consider the in $\tilde{\boldsymbol{y}}$ 1-periodic function

$$\begin{aligned}\tilde{u}: D\times\mathbb{T}^{d_{\boldsymbol{y}}} &\longrightarrow \mathbb{R}\\ \tilde{u}(\boldsymbol{x},\tilde{\boldsymbol{y}}) &:= u(\boldsymbol{x},\varphi(\tilde{\boldsymbol{y}})),\end{aligned}$$

with φ being some suitable transformation function, i.e.,

$$\varphi:\mathbb{T}^{d_{\boldsymbol{y}}}\longrightarrow D_{\boldsymbol{y}}=[\alpha,\beta]^{d_{\boldsymbol{y}}}.$$

With this approach, the usFFT is able to compute approximations of the functions $\tilde{u}_{\boldsymbol{x}_g}:=\tilde{u}(\boldsymbol{x}_g,\cdot)$ for each $\boldsymbol{x}_g\in\mathcal{T}_G$. We want φ to act component-wise on the random variable, i.e., $\varphi(\tilde{\boldsymbol{y}}):=(\varphi_j(\tilde{y}_j))_{j=1}^{d_{\boldsymbol{y}}}$. Further, we assume, that these mappings φ_j fulfill the assumptions

(A1) Each φ_j is continuous, i.e., $\varphi_j\in C(\mathbb{T})$ for each $j=1,...,d_{\boldsymbol{y}}$.

(A2) It holds $\varphi_j(0)=\varphi_j(1)=\alpha$ and $\varphi_j(\frac{1}{2})=\beta$ for each $j=1,...,d_{\boldsymbol{y}}$.

(A3) Each φ_j is symmetric, i.e., $\varphi_j(\frac{1}{2}-\tilde{y})=\varphi_j(\frac{1}{2}+\tilde{y})$ for $\tilde{y}\in[0,\frac{1}{2}]$ and for each $j=1,...,d_{\boldsymbol{y}}$.

(A4) Each φ_j is strictly monotonously increasing in $[0,\frac{1}{2}]$ for each $j=1,...,d_{\boldsymbol{y}}$.

With these restrictions we ensure, that φ is bijective w.r.t. the interval $[0,\frac{1}{2}]^{d_{\boldsymbol{y}}}$. Hence, we define the inverse mapping $\varphi^{-1}(\boldsymbol{y}):[\alpha,\beta]^{d_{\boldsymbol{y}}}\to[0,\frac{1}{2}]^{d_{\boldsymbol{y}}}$.

With this inverse mapping, we are now able to compute approximations of the functions $u_{\boldsymbol{x}_g}(\boldsymbol{y})$ via

$$u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y}):=\tilde{u}_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\varphi^{-1}(\boldsymbol{y}))=\sum_{\boldsymbol{k}\in\mathrm{I}}c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\,\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\varphi^{-1}(\boldsymbol{y})},\tag{3.1}$$

with the finite index set I and the approximated Fourier coefficients $c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})$ which come from the usFFT applied to the functions $\tilde{u}_{\boldsymbol{x}_g}$, $g\in\mathcal{T}_G$.

In this work, we always consider the tent transformation, cf. [27, 36, 11], for each φ_j, i.e.,

$$\varphi_j:\mathbb{T}\longrightarrow[\alpha,\beta],\qquad \varphi_j(\tilde{y})=\beta-|(\beta-\alpha)(1-2\tilde{y})|,\tag{3.2a}$$

$$\varphi_j^{-1}:[\alpha,\beta]\longrightarrow\left[0,\frac{1}{2}\right],\qquad \varphi_j^{-1}(y)=\frac{y-\alpha}{2(\beta-\alpha)}.\tag{3.2b}$$

Although this transformation mapping fulfills the assumptions (A1) - (A4), it might be not the most favorable choice in specific applications due to its lack of smoothness. Smoother periodizations, e.g., [6, Sec. 2.2.2], might yield better approximation results in specific situations due to the faster decay of the Fourier coefficients of $\tilde{u}$. On the other hand, the linear structure of the tent transformation on the interval $[0,\frac{1}{2}]$ allows some simplifications later on.

3.2.2 Lognormal case

As in the affine case, we need a suitable, periodic transformation mapping $\varphi:\mathbb{T}^{d_{\boldsymbol{y}}}\to D_{\boldsymbol{y}}=\mathbb{R}^{d_{\boldsymbol{y}}}$ to receive a periodization $\tilde{u}(\boldsymbol{x},\tilde{\boldsymbol{y}})$. Again, we choose the same functions in each stochastic dimension, so the same φ_j for all $j = 1, ..., d_{\boldsymbol{y}}$, but this time φ_j will consist of two separate steps. First, we consider the transformation

$$\tau_1 : \left(-\frac{1}{2}, \frac{1}{2}\right) \longrightarrow \mathbb{R}, \qquad \tau_1(\breve{y}) := \sqrt{2}\,\mathrm{erf}^{-1}(2\breve{y}),$$

$$\tau_1^{-1} : \mathbb{R} \longrightarrow \left(-\frac{1}{2}, \frac{1}{2}\right), \qquad \tau_1^{-1}(y) = \frac{1}{2}\,\mathrm{erf}\left(\frac{y}{\sqrt{2}}\right),$$

with the error function

$$\mathrm{erf}(y) := \frac{1}{\sqrt{\pi}} \int_{-y}^{y} \mathrm{e}^{-t^2}\,\mathrm{d}t, \quad x \in \mathbb{R}.$$

For further information on this transformation, see [28]. This mapping τ_1 seems like the ideal choice when talking about random variables $\boldsymbol{y} \sim \mathcal{N}(0,1)$, since the error function $\mathrm{erf}(y)$ is closely related to its cumulative distribution function Φ. In detail, it holds

$$\Phi(y) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{y}{\sqrt{2}}\right)\right).$$

The so-called inversion method in stochastic simulation describes, that the cumulative distribution function Φ and its inverse Φ^{-1} map random variables, distributed according to Φ, to uniformly distributed random variables on $[0,1]$ and the other way around, cf. [12, Sec. II.2]. Thus, our transformation τ_1 maps uniformly distributed random variables $\breve{y} \sim \mathcal{U}(-\frac{1}{2}, \frac{1}{2})$ to normally distributed random variables $y \sim \mathcal{N}(0,1)$ and is therefore a great generalization when moving forward from uniformly distributed random variables.

The second part is a suitable periodization $\tau_2 : \mathbb{T} \to (-\frac{1}{2}, \frac{1}{2})$. We choose a similar approach as in the affine case and use a shifted tent transformation

$$\tau_{2,\Delta} : \mathbb{T} \longrightarrow \left[-\frac{1}{2}, \frac{1}{2}\right], \qquad \tau_{2,\Delta}(\tilde{y}) = \begin{cases} -\frac{1}{2} - 2(\tilde{y} - \Delta) & 0 \leq \tilde{y} < \Delta \\ -\frac{1}{2} + 2(\tilde{y} - \Delta) & \Delta \leq \tilde{y} < \frac{1}{2} + \Delta \\ +\frac{3}{2} - 2(\tilde{y} - \Delta) & \frac{1}{2} + \Delta \leq \tilde{y} < 1 \end{cases}$$

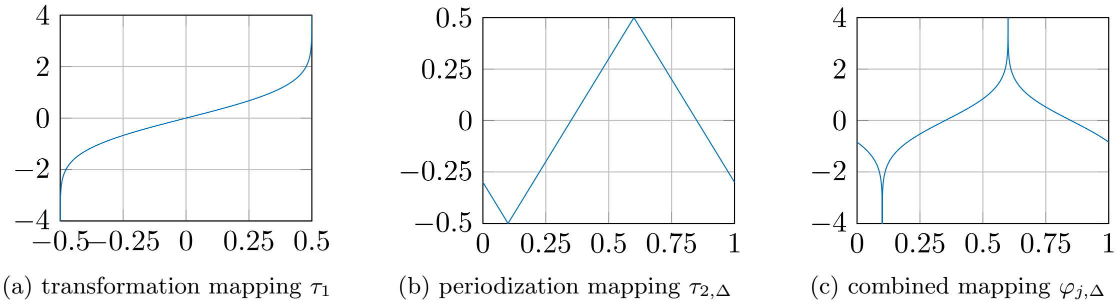
$$\tau_{2,\Delta}^{-1} : \left[-\frac{1}{2}, \frac{1}{2}\right] \longrightarrow \left[\Delta, \frac{1}{2} + \Delta\right], \qquad \tau_{2,\Delta}^{-1}(\breve{y}) = \frac{\breve{y}}{2} + \Delta + \frac{1}{4}$$

with shift $\Delta > 0$. We need this shift, since we can not apply the transformation τ_1 if we have $\tau_{2,\Delta}(\tilde{y}) = \pm\frac{1}{2}$ due to the poles there. Shifting with a suitable Δ ensures, that the deterministic part of the sampling set $\mathcal{X}$ does not contain components equal to Δ or $\frac{1}{2} + \Delta$. The randomly chosen values from the interval $[0,1]$ for the other components will not be equal to Δ or $\frac{1}{2} + \Delta$ almost surely too. Hence, the sampling set $\mathcal{X}$ in Algorithm 1 does not contain any nodes with any component equal to Δ or $\frac{1}{2} + \Delta$ almost surely.

Now we define the transformation mappings $\varphi_{j,\Delta}$ for each $j = 1, ..., d_{\boldsymbol{y}}$ for the lognormal case as

$$\varphi_{j,\Delta} : \mathbb{T} \setminus \left\{\Delta, \frac{1}{2} + \Delta\right\} \longrightarrow \mathbb{R}, \qquad \varphi_{j,\Delta}(\tilde{y}) = (\tau_1 \circ \tau_{2,\Delta})(\tilde{y}), \tag{3.3a}$$

$$\varphi_{j,\Delta}^{-1} : \mathbb{R} \longrightarrow \left(\Delta, \frac{1}{2} + \Delta\right), \qquad \varphi_{j,\Delta}^{-1}(y) = (\tau_{2,\Delta}^{-1} \circ \tau_1^{-1})(y). \tag{3.3b}$$



(a) transformation mapping τ_1 (b) periodization mapping $\tau_{2,\Delta}$ (c) combined mapping $\varphi_{j,\Delta}$

Figure 3.1: The plots of the transformation and periodization mappings τ_1 and $\tau_{2,\Delta}$ and the combined mapping $\varphi_{j,\Delta}$ with shift $\Delta = 0.1$.

The mapping $\varphi_{j,\Delta}$ as well as its two parts τ_1 and $\tau_{2,\Delta}$ are visualized in Figure 3.1. These mappings fulfill slightly modified versions of the assumptions (A1) - (A4) taking into account the shift Δ. Now we can use the transformation $\varphi_\Delta := (\varphi_{j,\Delta})_{j=1}^{d_{\boldsymbol{y}}}$ to receive the in $\tilde{\boldsymbol{y}}$ periodic signals $\tilde{u}(\boldsymbol{x}_g, \tilde{\boldsymbol{y}}) = u(\boldsymbol{x}_g, \varphi_\Delta(\tilde{\boldsymbol{y}}))$, $\boldsymbol{x}_g \in \mathcal{T}_G$, that can be approximated using our usFFT, cf. Algorithm 1. Plugging the inverse mapping φ_Δ^{-1} into the evaluation formula, which is similar to (3.1), we are now able to compute approximations of the solution functions $u_{\boldsymbol{x}_g}(\boldsymbol{y})$ in the lognormal case as well. Again, the periodization φ_Δ is not smooth and therefore might yield non-optimal approximation results. In particular, the periodization mappings $\varphi_{j,\Delta}$ possess two poles instead of two kinks, which is a way worse smoothness behavior than in the affine case.

We now ask for the optimal choice of the parameter Δ, such that the deterministic components of the sampling nodes $\tilde{y}$ of the R1Ls in the usFFT are as far as possible from Δ and $\Delta + 1/2$ to reduce problems at the poles of the transformation mapping φ_Δ. Let

$$\tilde{y}_{i,j} := \frac{i}{M} z_j \mod 1, \qquad i = 0, \dots, M-1 \text{ and } j = 1, \dots, d \tag{3.4}$$

denote the j-th component of the i-th R1L node of the d-dimensional R1L of size M. Then, we are looking for Δ, such that the minimum of the two distances

$$\min_{\substack{i=0,\dots,M-1\\ j=1,\dots,d}} |\tilde{y}_{i,j} - \Delta| \qquad \text{and} \qquad \min_{\substack{i=0,\dots,M-1\\ j=1,\dots,d}} \left|\tilde{y}_{i,j} - \left(\Delta + \frac{1}{2}\right)\right|$$

is maximal.

Lemma 3.2. *Let $\Lambda(\boldsymbol{z}, M)$ be a d-dimensional R1L with prime lattice size $M \in \mathbb{N}$, $M > 2$, generating vector $\boldsymbol{z} \in \mathbb{Z}^d$, $z_{j_0} \not\equiv 0 \pmod M$ for at least one $j_0 \in \{1, \dots, d\}$, and the lattice nodes $\tilde{y}_{i,j}$ as defined in* (3.4)*. Then, we have*

$$\Delta_{opt} := \operatorname*{arg\,max}_{0<\Delta<\frac{1}{2M}} \left\{ \min \left\{ \min_{\substack{i=0,\dots,M-1\\ j=1,\dots,d}} |\tilde{y}_{i,j} - \Delta|, \min_{\substack{i=0,\dots,M-1\\ j=1,\dots,d}} \left|\tilde{y}_{i,j} - \left(\Delta + \frac{1}{2}\right)\right| \right\} \right\} = \frac{1}{4M}.$$

Proof. Since i and z_j in formula (3.4) are integers, we know that $\tilde{y}_{i,j} \in \{\frac{n}{M}, n = 0, \dots, M-1\}$ for all $i = 0, \dots, M-1$ and $j = 1, \dots, d$. In particular, since M is prime and $z_{j_0} \not\equiv 0 \pmod M$, we have that $\{\tilde{y}_{i,j_0}, i = 0, ..., M-1\} = \{\frac{n}{M}, n = 0, ..., M-1\}$, so each $\frac{n}{M}$ is really attained at least once for some i and j. Using this and the fact, that we are only considering $0 = \frac{0}{M} < \Delta < \frac{1}{2}\frac{1}{M} = \frac{1}{2M}$, we have

$$\min_{\substack{i=0,...,M-1\\ j=1,...,d}} |\tilde{y}_{i,j} - \Delta| = \min_{n=0,...,M-1} \left|\frac{n}{M} - \Delta\right| = \left|\frac{0}{M} - \Delta\right| = \Delta.$$

On the other hand, we have

$$\min_{\substack{i=0,...,M-1\\ j=1,...,d}} \left|\tilde{y}_{i,j} - \left(\Delta + \frac{1}{2}\right)\right| = \min_{n=0,...,M-1} \left|\frac{n}{M} - \left(\Delta + \frac{1}{2}\right)\right|,$$

where the minimum is attained for n being the closest integer number to $M\Delta + \frac{M}{2}$. Since $0 < M\Delta < \frac{M}{2M} = \frac{1}{2}$ and M odd, we conclude

$$\min_{n=0,...,M-1} \left|\frac{n}{M} - \left(\Delta + \frac{1}{2}\right)\right| = \left|\frac{M+1}{2M} - \left(\Delta + \frac{1}{2}\right)\right| = \frac{1}{2M} - \Delta.$$

Since the sum of these two minima is constant $\frac{1}{2M}$, we have the upper bound

$$\min\left\{\min_{\substack{i=0,...,M-1\\ j=1,...,d}} |\tilde{y}_{i,j} - \Delta|, \min_{\substack{i=0,...,M-1\\ j=1,...,d}} \left|\tilde{y}_{i,j} - \left(\Delta + \frac{1}{2}\right)\right|\right\} = \min\left\{\Delta, \frac{1}{2M} - \Delta\right\} \leq \frac{1}{4M}.$$

Finally, this upper bound is reached if and only if $\Delta = \frac{1}{4M}$ and hence

$$\Delta_{\text{opt}} = \operatorname*{arg\,max}_{0<\Delta<\frac{1}{2M}} \left\{\min\left\{\Delta, \frac{1}{2M} - \Delta\right\}\right\} = \frac{1}{4M}.$$

■

Remark 3.3. *Note, that the* arg max *above is not unique in general, since there also exist several values for $\Delta \geq \frac{1}{2M}$ attaining this maximum, e.g., $\Delta = \frac{3}{4M}$, which can be proven analogously. In our numerical experiments in Section 4, we will always work with $\Delta_{opt} = \frac{1}{4M}$, which is the smallest optimal $\Delta > 0$ as we saw in the Theorem above.*

Also, if we would neglect the assumption that M is prime, we could run into problems if z_j and M are not coprime for all $j = 1, ..., d$, since then $\{\tilde{y}_{i,j}, i = 0, ..., M-1\}$ is only a proper subset of $\{\frac{n}{M}, n = 0, ..., M-1\}$. But this case is neglectable, since our algorithm only uses prime latttice sizes M anyway.

4 Numerics

We will now test our usFFT on different, two-dimensional numerical examples. In particular, we consider the parametric PDE (1.1) with zero boundary condition and different random coefficients $a(\boldsymbol{x}, \boldsymbol{y})$ and right-hand sides $f(\boldsymbol{x})$.

Table 4.1: Parameter settings η for the numerical tests of Algorithm 1.

η	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII
N	32	32	64	32	64	32	64	128	32	64	128	128	256
s, s_{local}	100	250		500		1000			2000			4000	
θ	$1 \cdot 10^{-12}$												
r	5												

Since our algorithm yields an approximation $u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})$ for each $\boldsymbol{x}_g \in \mathcal{T}_G$ separately, we also compute the approximation error

$$\operatorname{err}_p^{\eta}(\boldsymbol{x}_g) := \left(\frac{1}{n_{\text{test}}} \sum_{j=1}^{n_{\text{test}}} \left| \check{u}\left(\boldsymbol{x}_g, \boldsymbol{y}^{(j)}\right) - u^{\texttt{usFFT}}\left(\boldsymbol{x}_g, \boldsymbol{y}^{(j)}\right) \right|^p \right)^{\frac{1}{p}} \tag{4.1}$$

and

$$\operatorname{err}_\infty^{\eta}(\boldsymbol{x}_g) := \max_{j=1,\dots,n_{\text{test}}} \left| \check{u}\left(\boldsymbol{x}_g, \boldsymbol{y}^{(j)}\right) - u^{\texttt{usFFT}}\left(\boldsymbol{x}_g, \boldsymbol{y}^{(j)}\right) \right| \tag{4.2}$$

for each $\boldsymbol{x}_g \in \mathcal{T}_G$ separately, using $n_{\text{test}} = 10^5$ different, randomly drawn test variables $\boldsymbol{y}^{(j)}$ from the underlying probability distribution. Here, $\check{u}(\circ, \boldsymbol{y}^{(j)})$ are the finite element solutions of the PDE for fixed parameters $\boldsymbol{y}^{(j)}$ and $u^{\texttt{usFFT}}(\boldsymbol{x}_g, \circ)$ are our approximations from the usFFT. The parameter η denotes the used sFFT parameters as given in Table 4.1.

Here, N is the extension of the full grid $[-N, N]^{d_{\boldsymbol{y}}}$, that is used as the search space $\Gamma \subset \mathbb{Z}^{d_{\boldsymbol{y}}}$. Note, that we also choose $s_{\text{local}} = s$. If we miss an important frequency component at one point $\boldsymbol{x}_g$, it is very likely, that it is contained in the detected index set of a neighboring mesh point. Therefore, the union over all points $\boldsymbol{x}_g$ should be enough to avoid losing frequencies and we do not need a larger s_{local}. The choices for Θ and r are common values and the same as in [22]. In particular, we choose the number of detection iterations r as well as the probabilities γ_{A} and γ_{B} as in the case $G = 1$, since we expect a huge overlap of the detected index sets and hence a small failure probability even for these parameter choices instead of the theoretical choices given in the proof of Theorem 1.1.

Further, we always use the random R1L approach in the role of Algorithm A to recover the projected Fourier coefficients in the dimension-incremental method, cf. Section 2.2 and [22]. We also tested the algorithm using the single and multiple R1L approaches mentioned in Section 2.2, but these did not achieve significantly smaller approximation errors and are using larger numbers of samples and therefore result in longer runtimes of the algorithm. Hence, it seems reasonable to stick with the random R1L approach here. We choose the target maximum edge length of the finite element mesh $h_{\max} = 0.075$ in the FE solver. All examples consider the spatial domain $D = [0, 1]^2$, resulting in a finite element mesh $\mathcal{T}_G \subset D$ with $G = 737$ inner and 104 boundary nodes.

Further, we will also analyze the importance of and the interactions between our detected Fourier coefficients. To this end, we use the classical ANOVA decomposition of 1-periodic functions as given in [31] or [37, 24]. Note, that in there the ANOVA decomposition is used already in the proposed methods to receive an adaptive selection of the most important approximation terms, which we realized in our method by simply comparing the size of the projected Fourier coefficients, cf. Sections 2.2 and 3.1.

In particular, we consider the variance of our approximation

$$\sigma^2(\tilde{u}_{\boldsymbol{x}_g}^{\texttt{usFFT}}) := \|\tilde{u}_{\boldsymbol{x}_g}^{\texttt{usFFT}}\|_{L_2(\mathbb{T}^{d_{\boldsymbol{y}}})}^2 - |c_{\boldsymbol{0}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})|^2 = \sum_{\boldsymbol{k}\in \mathrm{I}\setminus\{\boldsymbol{0}\}} |c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})|^2.$$

Now we can study different subsets $\mathrm{J} \subset \mathrm{I}$ and estimate the variance of the approximation using only these subsets. The fraction of variance, that is explained using this subset J, is then called global sensitivity index (short: GSI), see [34, 35],

$$\varrho(\mathrm{J}, \tilde{u}_{\boldsymbol{x}_g}^{\texttt{usFFT}}) := \frac{\sigma^2(\tilde{u}_{\boldsymbol{x}_g,\mathrm{J}}^{\texttt{usFFT}})}{\sigma^2(\tilde{u}_{\boldsymbol{x}_g}^{\texttt{usFFT}})} = \frac{\sum_{\boldsymbol{k}\in \mathrm{J}\setminus\{\boldsymbol{0}\}} |c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})|^2}{\sum_{\boldsymbol{k}\in \mathrm{I}\setminus\{\boldsymbol{0}\}} |c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})|^2} \in [0,1], \tag{4.3}$$

where we define $\tilde{u}_{\boldsymbol{x}_g,\mathrm{J}}^{\texttt{usFFT}}(\boldsymbol{y}) := \sum_{\boldsymbol{k}\in\mathrm{J}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\varphi^{-1}(\boldsymbol{y})}$. In our examples, we will mainly consider the subsets J_ℓ of all frequencies $\boldsymbol{k}$ with exactly ℓ non-zero components, i.e.,

$$\mathrm{J}_\ell := \{\boldsymbol{k} \in \mathrm{I} : \|\boldsymbol{k}\|_0 := |\{i \in \{1, ..., d_{\boldsymbol{y}}\} : k_i \neq 0\}| = \ell\}\,. \tag{4.4}$$

Finally, one can also think about evaluating various quantities of interest of the approximation. Here, we will consider the expectation value $\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ as one example of such quantities. We use a Monte-Carlo approximation of the expectation value

$$\overline{\tilde{u}_{\boldsymbol{x}_g}} := \frac{1}{n_{\mathrm{MC}}} \sum_{j=1}^{n_{\mathrm{MC}}} \check{u}\left(\boldsymbol{x}_g, \boldsymbol{y}^{(j)}\right)$$

of the finite element approximation using n_{MC} random samples for comparison.

4.1 Expectation value of the approximation

Computing the expectation value of our approximation $u_{\boldsymbol{x}_g}^{\texttt{usFFT}}$ requires some additional effort, depending on the particular model and eventually used periodization methods. By definition, the expectation value is given by

$$\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}}) := \int_{D_{\boldsymbol{y}}} u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})\,\mathrm{d}\mu(\boldsymbol{y}) = \int_{D_{\boldsymbol{y}}} u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})\,p(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y},$$

where p is the probability density function of the random variable $\boldsymbol{y}$.

For the periodic model, we do not need any periodization. Therefore, the approximation of the solution reads as

$$u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y}) = \sum_{\boldsymbol{k}\in\mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})\,\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{y}}$$

with I the frequency set and $c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})$ the corresponding approximated Fourier coefficients computed by the usFFT. The random variable $\boldsymbol{y}$ is assumed to be uniformly distributed in

$D_{\boldsymbol{y}} = [-\frac{1}{2}, \frac{1}{2}]^{d_{\boldsymbol{y}}}$ in this case. Hence, we have

$$\begin{aligned}
\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}}) &= \int_{D_{\boldsymbol{y}}} u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})\, p(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} \\
&= \int_{[-\frac{1}{2},\frac{1}{2}]^{d_{\boldsymbol{y}}}} 1^{-d_{\boldsymbol{y}}} \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})\, \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{y}}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g}) \int_{[-\frac{1}{2},\frac{1}{2}]^{d_{\boldsymbol{y}}}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{y}}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})\, \delta_{\boldsymbol{k}} = c_{\boldsymbol{0}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g}).
\end{aligned}$$

In the affine case, we use the tent transformation (3.2), such that our approximation reads as

$$u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y}) = \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, \mathrm{e}^{\pi \mathrm{i} \boldsymbol{k}\cdot \frac{\boldsymbol{y}-\alpha\boldsymbol{1}}{\beta-\alpha}}$$

with $\boldsymbol{1} = (1, 1, ..., 1) \in \mathbb{R}^{d_{\boldsymbol{y}}}$. Again, the random variable $\boldsymbol{y}$ is assumed to be uniformly distributed, but for this computation we work with the more general domain $D_{\boldsymbol{y}} = [\alpha, \beta]^{d_{\boldsymbol{y}}}$. Therefore, we have

$$\begin{aligned}
\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}}) &= \int_{D_{\boldsymbol{y}}} u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})\, p(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} \\
&= \int_{[\alpha,\beta]^{d_{\boldsymbol{y}}}} (\beta-\alpha)^{-d_{\boldsymbol{y}}} \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, \mathrm{e}^{\pi \mathrm{i} \boldsymbol{k}\cdot \frac{\boldsymbol{y}-\alpha\boldsymbol{1}}{\beta-\alpha}}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})(\beta-\alpha)^{-d_{\boldsymbol{y}}} \int_{[\alpha,\beta]^{d_{\boldsymbol{y}}}} \mathrm{e}^{\pi \mathrm{i} \boldsymbol{k}\cdot \frac{\boldsymbol{y}-\alpha\boldsymbol{1}}{\beta-\alpha}}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, D_{\boldsymbol{k}} \qquad (4.5)
\end{aligned}$$

with

$$D_{\boldsymbol{k}} \coloneqq \prod_{j=1}^{d_{\boldsymbol{y}}} D_{k_j} \qquad \text{and} \qquad D_{k_j} \coloneqq \begin{cases} \frac{2\mathrm{i}}{\pi k_j} & k_j \equiv 1 \mod 2 \\ 1 & k_j = 0 \\ 0 & \text{else.} \end{cases}$$

Note, that the parameters α and β vanish completely. Thus, the formula is independent of the particular domain $D_{\boldsymbol{y}} = [\alpha, \beta]^{d_{\boldsymbol{y}}}$.

Finally, the lognormal model involves the more complicated transformation mappings $\varphi_{j,\Delta}$ given in (3.3). Thus, the approximation reads as

$$u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y}) = \sum_{\boldsymbol{k}\in \mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot\varphi_{\Delta}^{-1}(\boldsymbol{y})}.$$

Here, the random variable $\boldsymbol{y}$ is standard normally distributed, i.e., $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ with $\boldsymbol{I}$ the identity matrix of dimension $d_{\boldsymbol{y}}$. Hence, the expectation value can be written as

$$\begin{aligned}
\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}}) &= \int_{D_{\boldsymbol{y}}} u_{\boldsymbol{x}_g}^{\texttt{usFFT}}(\boldsymbol{y})\, p(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} \\
&= \int_{\mathbb{R}^{d_{\boldsymbol{y}}}} (2\pi)^{-\frac{d_{\boldsymbol{y}}}{2}}\, \mathrm{e}^{-\frac{1}{2}\|\boldsymbol{y}\|^2} \sum_{\boldsymbol{k}\in\mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})\, \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\varphi_{\Delta}^{-1}(\boldsymbol{y})}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in\mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g})(2\pi)^{-\frac{d_{\boldsymbol{y}}}{2}} \int_{\mathbb{R}^{d_{\boldsymbol{y}}}} \mathrm{e}^{-\frac{1}{2}\|\boldsymbol{y}\|^2}\, \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\varphi_{\Delta}^{-1}(\boldsymbol{y})}\, \mathrm{d}\boldsymbol{y} \\
&= \sum_{\boldsymbol{k}\in\mathrm{I}} c_{\boldsymbol{k}}^{\texttt{usFFT}}(\tilde{u}_{\boldsymbol{x}_g}) D_{\boldsymbol{k},\Delta}
\end{aligned}$$

with

$$D_{\boldsymbol{k},\Delta} \coloneqq \prod_{j=1}^{d_{\boldsymbol{y}}} D_{k_j,\Delta} \qquad \text{and} \qquad D_{k_j,\Delta} \coloneqq \begin{cases} \frac{2\mathrm{i}}{\pi k_j}\mathrm{e}^{2\pi\mathrm{i}k_j\Delta} & k_j \equiv 1 \mod 2 \\ 1 & k_j = 0 \\ 0 & \text{else.} \end{cases}$$

Note, that the factors $D_{k_j,\Delta}$ are exactly the same as the D_{k_j} in the affine case up to the correction term $\mathrm{e}^{2\pi\mathrm{i}k_j\Delta}$ due to the shift with Δ.

4.2 Periodic example

We consider the example from [18, Sec. 6] using the domain $D = (0,1)^2$ with right-hand side $f(\boldsymbol{x}) = x_2$ and the random coefficient

$$a(\boldsymbol{x},\boldsymbol{y}) \coloneqq 1 + \frac{1}{\sqrt{6}} \sum_{j=1}^{d_{\boldsymbol{y}}} \sin(2\pi y_j)\, \psi_j(\boldsymbol{x}), \qquad \boldsymbol{x} \in D,\ \boldsymbol{y} \in D_{\boldsymbol{y}},$$

with the random variables $\boldsymbol{y} \sim \mathcal{U}\left([-\frac{1}{2},\frac{1}{2}]^{d_{\boldsymbol{y}}}\right)$ and

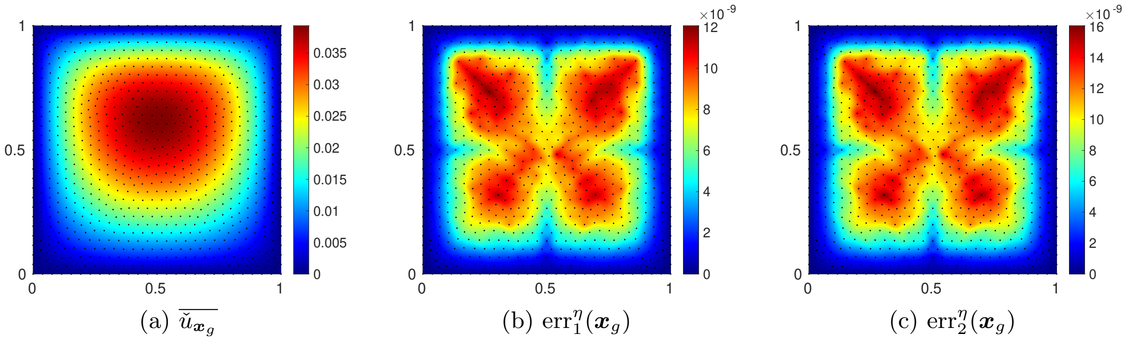
$$\psi_j(\boldsymbol{x}) \coloneqq c j^{-\mu} \sin(j\pi x_1)\sin(j\pi x_2), \qquad \boldsymbol{x} \in D,\ j \geq 1,$$

where $c > 0$ is a constant and $\mu > 1$ is the decay rate. Accordingly, we get

$$a_{\min} = 1 - \frac{c}{\sqrt{6}}\zeta(\mu) \qquad \text{and} \qquad a_{\max} = 1 + \frac{c}{\sqrt{6}}\zeta(\mu),$$

such that for $c < \frac{\sqrt{6}}{\zeta(\mu)}$ the uniform ellipticity assumption (2.1) is fulfilled.

We test the usFFT with the stochastic dimension $d_{\boldsymbol{y}} = 10$ on the two parameter choices $\mu = 1.2$, $c = 0.4$ and $\mu = 3.6$, $c = 1.5$ from [18]. The first choice seems to model a more difficult PDE, since the decay of the functions ψ_j w.r.t. j is very slow and we have $a_{\min} = 0.08690$ and $a_{\max} = 1.91310$. This range of a is wider and $a_{\min}$ is closer to zero than for the quickly decaying second parameter choice with $a_{\min} = 0.31660$ and $a_{\max} = 1.68340$. Figure 4.1 illustrates the total approximation error $\mathrm{err}_p^{\eta}(\boldsymbol{x}_g)$ for $p = 1$ and $p = 2$ as well as the Monte-Carlo approximation of the expectation value $\overline{\tilde{u}_{\boldsymbol{x}_g}}$ using $n_{\mathrm{MC}} = 10^6$ samples for comparison.



(a) $\overline{\breve{u}_{\boldsymbol{x}_g}}$ (b) $\mathrm{err}_1^{\eta}(\boldsymbol{x}_g)$ (c) $\mathrm{err}_2^{\eta}(\boldsymbol{x}_g)$

Figure 4.1: The MC approximation $\overline{\breve{u}_{\boldsymbol{x}_g}}$ and the approximation errors $\mathrm{err}_1^{\eta}(\boldsymbol{x}_g)$ and $\mathrm{err}_2^{\eta}(\boldsymbol{x}_g)$ for the periodic example with $\mu = 1.2$, $c = 0.4$, $d_{\boldsymbol{y}} = 10$, $\eta =$ VII, i.e., $s = 1000$, $N = 64$.

The absolute error err_p^{η} here is very small compared to the function values of $\overline{\breve{u}_{\boldsymbol{x}_g}}$. Thus, our approximation $u_{\boldsymbol{x}_g}^{\texttt{usFFT}}$ is already a very good approximation for these relatively small sparsity and extension parameters s and N. A more detailed insight on the decay of the error is given in Figure 4.2. There, the largest approximation error err_2^{η} w.r.t. the nodes $\boldsymbol{x}_g$ is given with the number of samples used in the corresponding usFFT. Note, that these amounts barely depend on the extension N of our search space Γ. The periodic setting results in very quickly decaying Fourier coefficients. Obviously, the same holds for their projections computed in the dimension-incremental steps. In particular, most of the one-dimensional projections in step 1 & 2a of Algorithm 1, e.g., all projections with component k_t with $|k_t| > 4$, at the start of each iteration are so small, that they are neglected immediately. That means, that the one-dimensional index sets $\mathrm{I}^{(t)}$ are independent of N (for large enough N) and hence the choice of N has a marginale impact only. Note, that we also tested our algorithm with smaller thresholds θ because of the described behavior, but the additionally detected and not neglected frequencies did not change the approximation significantly in the end.

We indicate some kind of linear behavior in this double logarithmic plot. Additional tests showed, that even for smaller sparsities $1 \leq s < 100$ the corresponding samples-error-pair fits into this model, i.e., there seems to be no pre-asymptotic behavior of our algorithm. In [18] the theoretical decay rates are often smaller than the error decay observed in numerical experiments. We also observe a relatively fast decay compared to these theoretical rates. On the other hand, the decay of the approximation error err_2^{η} for the faster decaying random coefficient a with $\mu = 3.6$ is not that much better than the decay of the more complicated example with $\mu = 1.2$. It seems like our algorithm is capable of handling the more difficult problem very well, but also does not yield that much further advantages when being applied to easier problems, i.e., with larger θ, larger $a_{\min}$ and a smaller range of the interval $[a_{\min}, a_{\max}]$. Note, that most of the samples needed are required for the detection of the frequency set I and only a small fraction is really used for the final computation of the corresponding Fourier coefficients, cf. Section 4.5 and Remark 4.2. We also computed the approximation error $\mathrm{err}_{\infty}^{\eta}$ for different η for both parameter choices of μ and c. Obviously, these errors have to be larger than the shown errors err_2^{η}, but the actual magnitude of $\mathrm{err}_{\infty}^{\eta}$ is only about 10 or 15 times as large as the errors err_2^{η}. Hence, the pointwise approximation error seems to stay in a reasonable size for any randomly drawn $\boldsymbol{y}$.

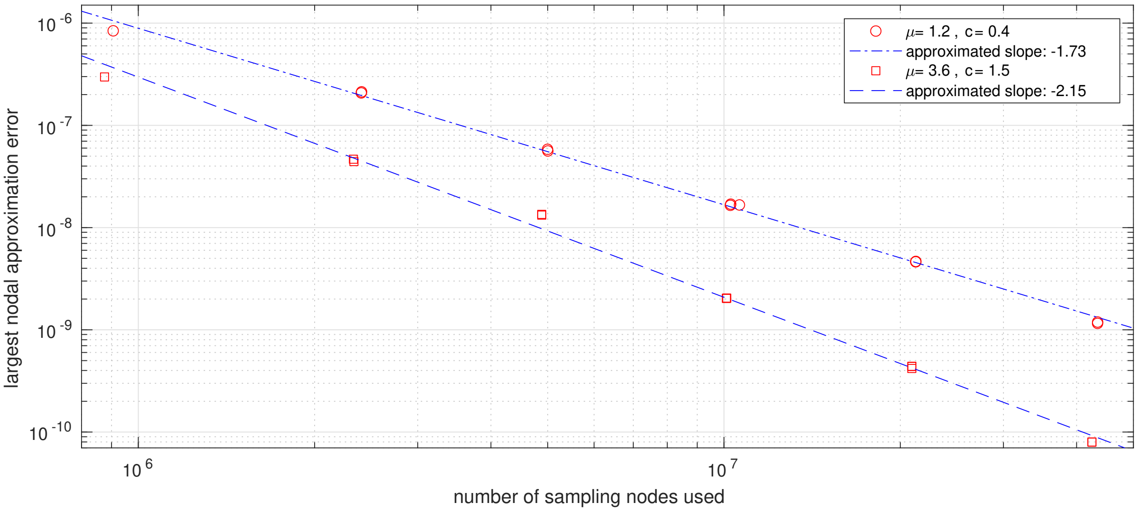


Figure 4.2: Largest error err_2^η w.r.t. the nodes $\boldsymbol{x}_g$ for all parameter settings η displayed in Table 4.1 for the periodic example.

As we saw in Section 4.1, the expectation value of our approximation $\mathbb{E}(u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ is simply its zeroth Fourier coefficient $c_{\mathbf{0}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})$. Since this coefficient is included and computed for each sparsity s anyway, it seems like our different parameter choices would not influence the precision of its approximation at first sight. But for larger sparsities s, we compute more Fourier coefficients in our algorithm, where possible aliasing effects should spread evenly among all of these coefficients, i.e., the particular so-called aliasing error on $c_{\mathbf{0}}^{\texttt{usFFT}}(u_{\boldsymbol{x}_g})$, cf. [21, 22], gets smaller and the approximation improves. Unfortunately, this is not visible in our numerical tests, since the comparison value $\overline{\check{u}_{\boldsymbol{x}_g}}$ behaves too poorly. In detail, we would have to investigate very small sparsities $s < 25$ to observe the described effects. For all of our parameter choices η, the Monte-Carlo approximation $\overline{\check{u}_{\boldsymbol{x}_g}}$ with $n_{\mathrm{MC}} = 5 \cdot 10^6$ samples is not accurate enough to give insight on the particular behavior of our approximation of the expectation value.

Finally, Figure 4.3 shows the cardinality of the sets J_ℓ, i.e., the number of frequencies detected with exactly ℓ non-zero components as given in (4.4), as well as the corresponding global sensitivity indices $\varrho(\mathrm{J}_\ell, u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ given in (4.3). Note, that these values also depend on the considered point $\boldsymbol{x}_g$. Therefore, the bars show the smallest and largest GSI among all nodes $\boldsymbol{x}_g \in \mathcal{T}_G$ as well as their median and mean value. As we would expect, there are no frequencies detected with all or nearly all components being active. Further, even though only 68 of the 4819 frequencies detected (excluding $c_{\mathbf{0}}$) have exactly one non-zero component, i.e., are supported on the axis cross, they contain more than 99% of the variance of our approximation. So the higher-dimensional frequencies with two, three or four non-zero components contribute much less to the variance of $u_{\boldsymbol{x}_g}^{\texttt{usFFT}}$.

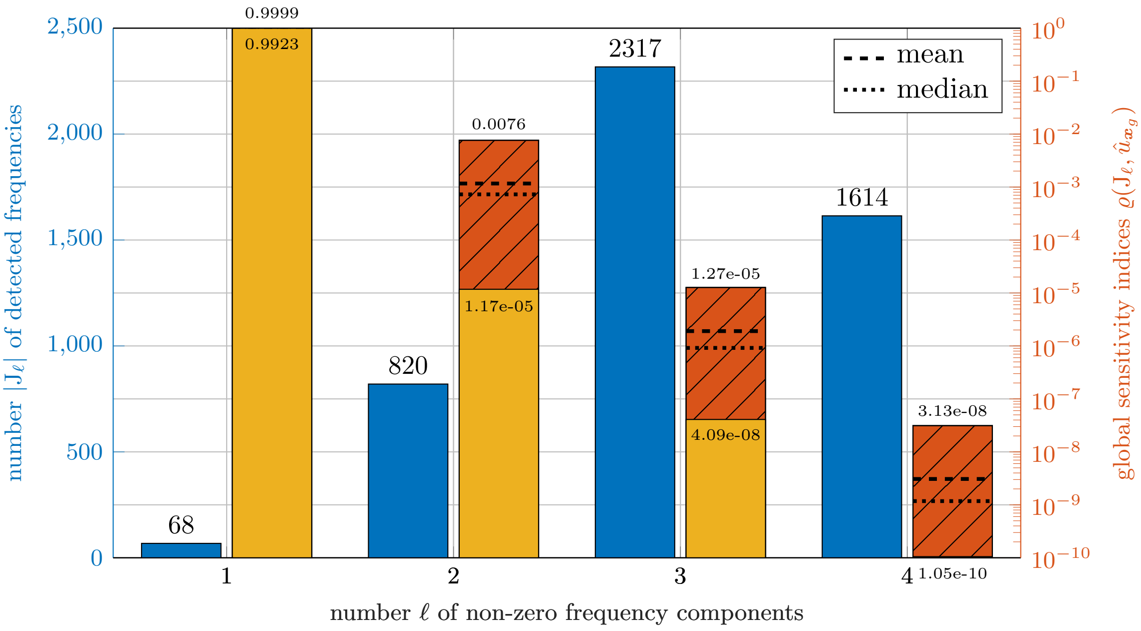


Figure 4.3: Cardinality (left, blue, solid) of the index sets J_ℓ and the corresponding largest (right, orange, striped), smallest (right, yellow, solid), mean (dashed line) and median (dotted line) of the global sensitivity indices $\varrho(J_\ell, u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ w.r.t. $\boldsymbol{x}_g$ for the periodic example with $\mu = 1.2$, $c = 0.4$, $d_{\boldsymbol{y}} = 10$, $\eta = \mathrm{XI}$, i.e., $s = 2000$, $N = 128$.

4.3 Affine example

For the affine case, we consider an example from [15, Sec. 11] with domain $D = (0,1)^2$, right-hand side $f(\boldsymbol{x}) \equiv 1$ and the random coefficient

$$a(\boldsymbol{x}, \boldsymbol{y}) := 1 + \sum_{j=1}^{d_{\boldsymbol{y}}} y_j \psi_j(\boldsymbol{x}), \qquad \boldsymbol{x} \in D,\ \boldsymbol{y} \in D_{\boldsymbol{y}},$$

with the random variables $\boldsymbol{y} \sim \mathcal{U}([-1,1]^{d_{\boldsymbol{y}}})$ and

$$\psi_j(\boldsymbol{x}) := c j^{-\mu} \cos(2\pi m_1(j)\, x_1) \cos(2\pi m_2(j)\, x_2), \qquad \boldsymbol{x} \in D,\ j \geq 1,$$

where again $c > 0$ is a constant and $\mu > 1$ the decay rate. Further, $m_1(j)$ and $m_2(j)$ are defined as

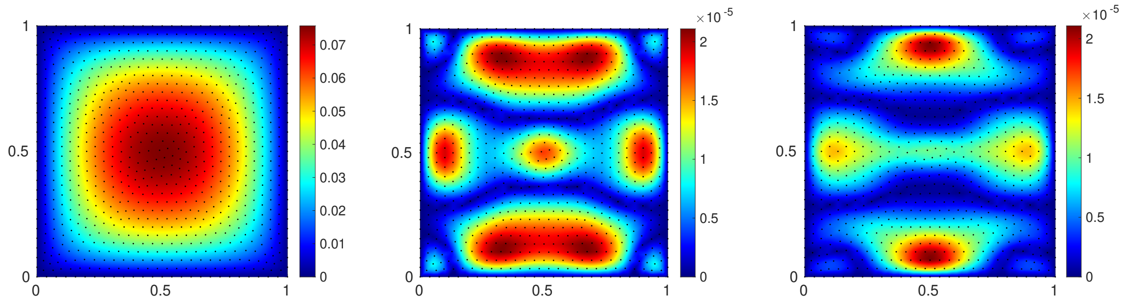
$$m_1(j) := j - \frac{k(j)(k(j)+1)}{2} \quad \text{and} \quad m_2(j) := k(j) - m_1(j)$$

with $k(j) := \lfloor -1/2 + \sqrt{1/4 + 2j} \rfloor$. Table 4.2 shows the numbers $m_1(j), m_2(j)$ and $k(j)$ for a few $j \geq 1$.

As before, we get that $a_{\min} = 1 - c\,\zeta(\mu)$ and $a_{\max} = 1 + c\,\zeta(\mu)$, such that for $c < \frac{1}{\zeta(\mu)}$ the uniform ellipticity assumption (2.1) is fulfilled. Here, we use the parameter choices from [15] with $\mu = 2$ for a relatively slow decay and $c = \frac{0.9}{\zeta(2)} \approx 0.547$ to end up with $a_{\min} = 0.1$ and

Table 4.2: The values of $m_1(j), m_2(j)$ and $k(j)$.

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	$\dots$
$m_1(j)$	0	1	0	1	2	0	1	2	3	0	1	2	3	4	$\dots$
$m_2(j)$	1	0	2	1	0	3	2	1	0	4	3	2	1	0	$\dots$
$k(j)$	1		2			3				4					$\dots$



(a) $\overline{\breve{u}_{\boldsymbol{x}_g}}$ (b) $\eta = \mathrm{I}$ $(s = 100, N = 32)$ (c) $\eta = \mathrm{II}$ $(s = 250, N = 32)$

Figure 4.4: The MC approximation $\overline{\breve{u}_{\boldsymbol{x}_g}}$ and the pointwise errors $|\overline{\breve{u}_{\boldsymbol{x}_g}} - \mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})|$ for $\eta = \mathrm{I}$ and II for the affine example.

$a_{\max} = 1.9$, which is very similar to the first parameter choice in the periodic case. We choose the stochastic dimension $d_{\boldsymbol{y}} = 20$ as in [15].

Figure 4.4 illustrates the Monte-Carlo approximation of the expectation value $\overline{\breve{u}_{\boldsymbol{x}_g}}$ with $n_{\mathrm{MC}} = 10^6$ samples used as well as the pointwise error $|\overline{\breve{u}_{\boldsymbol{x}_g}} - \mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})|$ for two different parameter choices η with $\mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})$ as given in (4.5). We note, that even for small sparsities s the approximation $\mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})$ seems to be quite accurate. Unfortunately, for all other parameter choices η except I and II, the magnitude of the errors does not decrease any further than $1.5 \cdot 10^{-5}$, which is probably again caused by the poor performance of the Monte-Carlo approximation $\overline{\breve{u}_{\boldsymbol{x}_g}}$.

Figure 4.5 again shows the largest error err_2^η w.r.t. the nodes $\boldsymbol{x}_g$ for different parameter settings η. The magnitudes of the errors are already very low for small sparsities s compared to the expected function values shown in Figure 4.4a. This time, we can observe a small increase in the number of used samples for larger extensions N. So in this example, the parameter settings $\eta = \mathrm{I}$ to XI have monotonously increasing sampling sizes, i.e., the data points in Figure 4.5 are ordered from left to right w.r.t. increasing η. We note, that there is an obvious improvement for each sparsity, when we progress from $N = 32$, the first data point in each cluster, to $N = 64$, the second data point. In the periodic case, the important frequencies are very well localized around zero, such that the choice of N has almost no impact. This time, we really lose some of our accuracy, if we choose the smaller extension $N = 32$. For sparsity $s = 2000$ we also see this effect when progressing from $N = 64$ to $N = 128$. The overall decay of the error is a lot slower than for the periodic example. This is probably mainly caused by the non-smooth tent transformation used, cf. Section 3.2.1. Again, some random tests for the error err_∞^η revealed a similar behavior as in the periodic setting and showed, that these errors again are not larger than at most 20 times the error err_2^η.

Finally, the ANOVA shows some other interesting results, visualized in Figure 4.6. Since

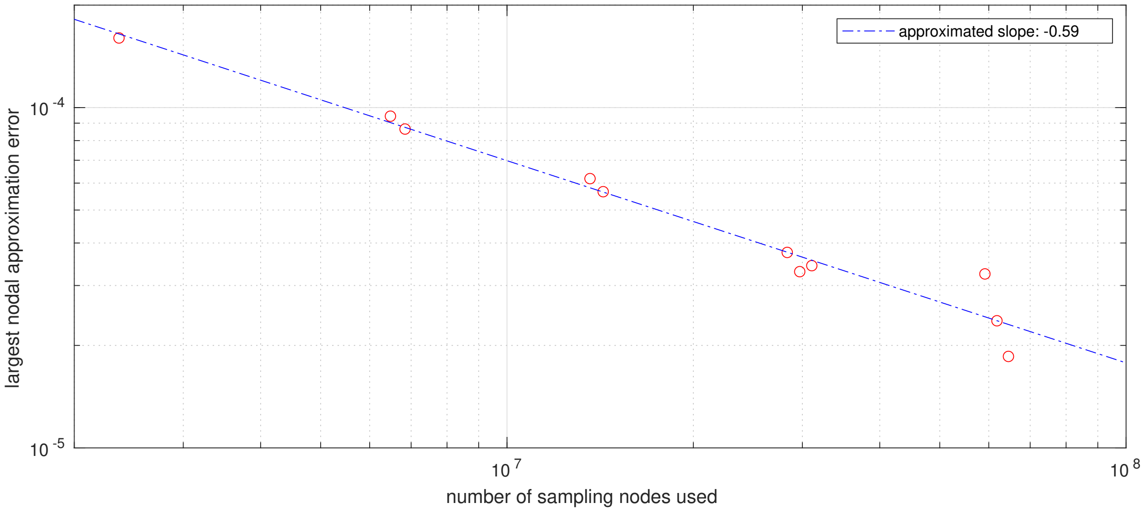


Figure 4.5: Largest error err_2^η w.r.t. the nodes $\boldsymbol{x}_g$ for the parameter settings η = I to XI displayed in Table 4.1 for the affine example.

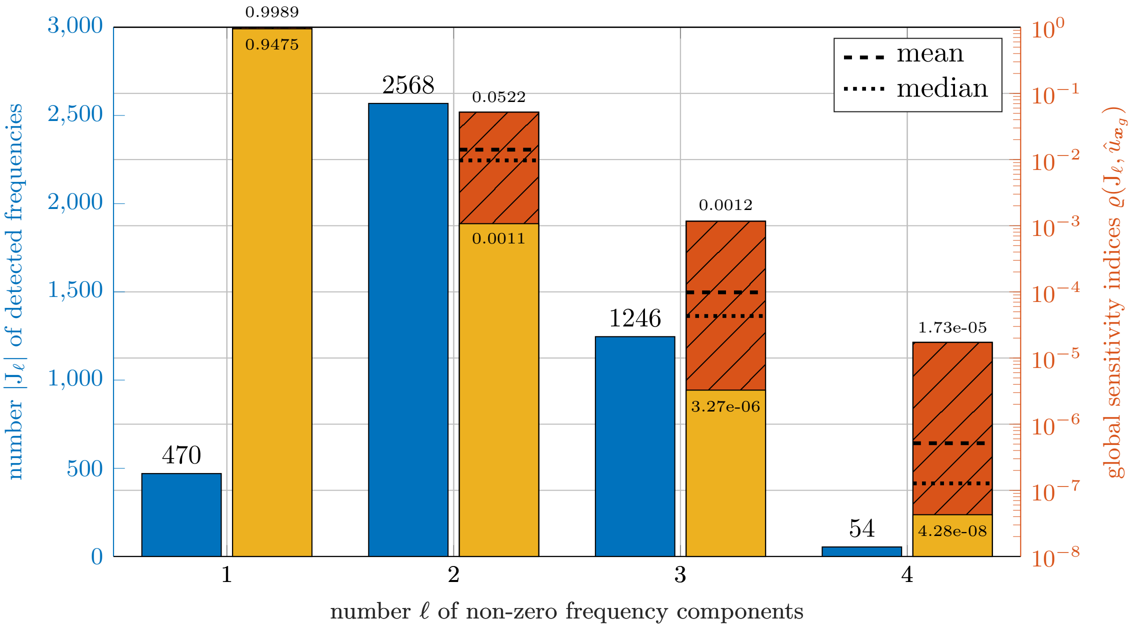
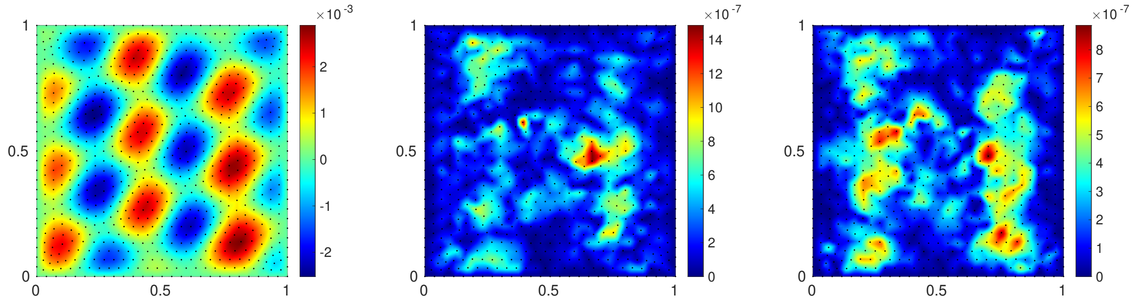


Figure 4.6: Cardinality (left, blue, solid) of the index sets J_ℓ and the corresponding largest (right, orange, striped), smallest (right, yellow, solid), mean (dashed line) and median (dotted line) of the global sensitivity indices $\varrho(\mathrm{J}_\ell, u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ w.r.t. $\boldsymbol{x}_g$ for the affine example with $\eta = \mathrm{IX}$, i.e., $s = 2000,\ N = 32$.

the Fourier coefficients do not decay as fast as in the smooth periodic case, we detected a significantly larger number of one- and two-dimensional couplings. Again, the frequencies with only one non-zero entry explain the largest part of the variance of the function, but this time the minimum percentage is lower than in the periodic example with only about 94.5%. Accordingly, the importance of the two- and three-dimensional pairings did slightly grow.



(a) $\overline{\breve{u}_{\boldsymbol{x}_g}}$ (b) $\eta =$ IV $(s = 500, N = 32)$ (c) $\eta =$ VI $(s = 1000, N = 32)$

Figure 4.7: The MC approximation $\overline{\breve{u}_{\boldsymbol{x}_g}}$ and the pointwise errors $|\overline{\breve{u}_{\boldsymbol{x}_g}} - \mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})|$ for $\eta =$ IV and VI for the lognormal example.

The large number of important coefficients with only one, two or three non-zero entries also results in nearly no detected frequencies with any more non-zero entries. For example, the 54 frequencies in J_4 will vanish when working with larger extensions N, since other frequencies with less entries are preferred in that case. So even though we are working with the moderate stochastic dimension $d_{\boldsymbol{y}} = 20$, we do not detect any frequencies, where the half or even only a quarter of these dimensions are active simultaneously.

4.4 Lognormal example

We consider a two-dimensional problem based on the example in [8] on the domain $D = [0, 1]^2$ with right-hand side $f(\boldsymbol{x}) = \sin(1.3\pi x_1 + 3.4\pi x_2)\cos(4.3\pi x_1 - 3.1\pi x_2)$. The lognormal random coefficient is given by

$$a(\boldsymbol{x}, \boldsymbol{y}) := \exp(b(\boldsymbol{x}, \boldsymbol{y})) \qquad \text{and} \qquad b(\boldsymbol{x}, \boldsymbol{y}) := \sum_{j=1}^{d_{\boldsymbol{y}}} \frac{1}{j} y_j \psi_j(\boldsymbol{x})$$

with the functions

$$\psi_j(\boldsymbol{x}) := \sin(2\pi j x_1)\cos(2\pi(d_{\boldsymbol{y}} + 1 - j)x_2).$$

In [8], the stochastic dimension $d_{\boldsymbol{y}} = 4$ has been used. Here, we will work with $d_{\boldsymbol{y}} = 10$ to receive a more complicated and higher-dimensional problem setting. We use a standard normally distributed random variable $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ with $\boldsymbol{I}$ the identity matrix of dimension $d_{\boldsymbol{y}}$ as before. Hence, we have, that for each $\boldsymbol{x}$ there holds $0 < a(\boldsymbol{x}, \boldsymbol{y}) < \infty$ for any $\boldsymbol{y}$. However, there do not exist the constants $0 < a_{\min} \leq a_{\max} < \infty$ in this example, since $b(\boldsymbol{x}, \boldsymbol{y})$ can become arbitrarily small or large. Therefore, the problem is neither uniformly elliptic nor uniformly bounded. This complicates the analysis of this problem tremendously. We can still stick with it for our numerical tests, since we only need the solvability of the differential equation for fixed values of $\boldsymbol{y}$. Further, we have $b(\boldsymbol{x}, \boldsymbol{y}) \in [-3, 3]$ and therefore $\exp(b(\boldsymbol{x}, \boldsymbol{y})) \in [\mathrm{e}^{-3}, \mathrm{e}^{3}] \approx [0.05, 20.09]$ with a probability of more than 99% for each $\boldsymbol{x} \in D$, i.e., tremendously small or large values of $a(\boldsymbol{x}, \boldsymbol{y})$ are very unlikely to appear anyway.

Figure 4.7a once again illustrates the Monte-Carlo approximation of the expectation value $\overline{\breve{u}_{\boldsymbol{x}_g}}$ with $n_{\mathrm{MC}} = 10^6$ samples used. We note, that the solution has a more interesting structure

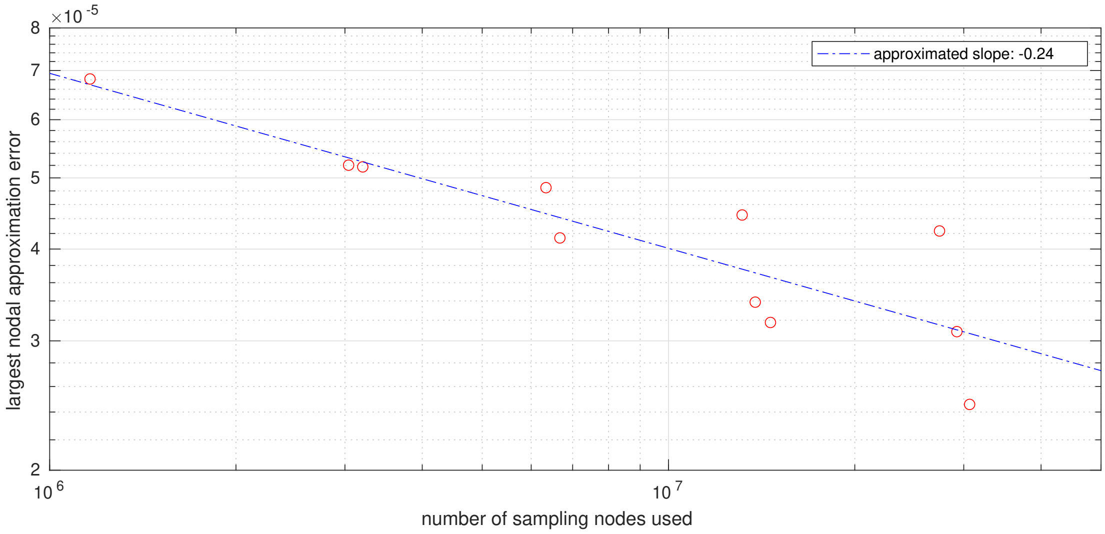


Figure 4.8: Largest error err_2^η w.r.t. the nodes $\boldsymbol{x}_g$ for the parameter settings η = I to XI displayed in Table 4.1 for the lognormal example.

than for the other examples above, mainly caused by the lognormal random coefficient and the non-constant right-hand side $f(\boldsymbol{x})$. Nevertheless, the approximations $\mathbb{E}(u_{\boldsymbol{x}_g}^{\mathtt{usFFT}})$ achieve small errors, which are shown in Figure 4.7. This time, a further increase of the sparsity s and the extension N still increase the accuracy of our approximations, so the stagnation due to the limitations of the Monte-Carlo approximation $\overline{\tilde{u}_{\boldsymbol{x}_g}}$, that we saw in the previous examples, does not occur yet.

The pointwise errors $\mathrm{err}_2^\eta(\boldsymbol{x}_g)$ behave slightly worse but still very good, as we see in Figure 4.8. As in Figure 4.5 for the affine example, the data points are ordered from left to right w.r.t. increasing η. Again, the increase of the extension N shows visible improvements of the approximation error err_2^η. The decay rate is lower than before, matching our expectations since the lognormal example is far more difficult than the affine or periodic examples. Note, that once again the slope considers all data points shown, while specific decays for fixed extensions N might be slower or faster. Further, the size of the error err_∞^η is again about 10 times the size of err_2^η, revealing also a good pointwise approximation w.r.t. the random variable $\boldsymbol{y}$ in this scenario.

We notice a similar distribution of the detected frequencies $\boldsymbol{k}$ to the index sets $\mathrm{J}_\ell \cap I$ as before, cf. Figure 4.9. The key difference is the size of the GSI for each of these index sets. The range of the GSI for J_1 increased significantly, the minimal portion of variance is now only about 30%. Obviously, the GSI for the other index sets J_ℓ grew accordingly. This is probably caused by the more difficult structure of the lognormal diffusion coefficient a and the corresponding more difficult structure of the solution which is reflected in larger differences in the optimal frequency sets $\mathrm{I}_{\boldsymbol{x}_g}$, $g = 1, \ldots, G$, cf. Remark 4.1. Nevertheless, we again detect nearly no significant frequencies $\boldsymbol{k}$ with 4 or more active dimensions as in the previous examples.

Remark 4.1. *As mentioned before, the output of the usFFT contains more than sparsity s frequencies since we join the detected index sets in each dimension increment and use no thresholding technique to reduce the number of found frequencies after that. While we have*

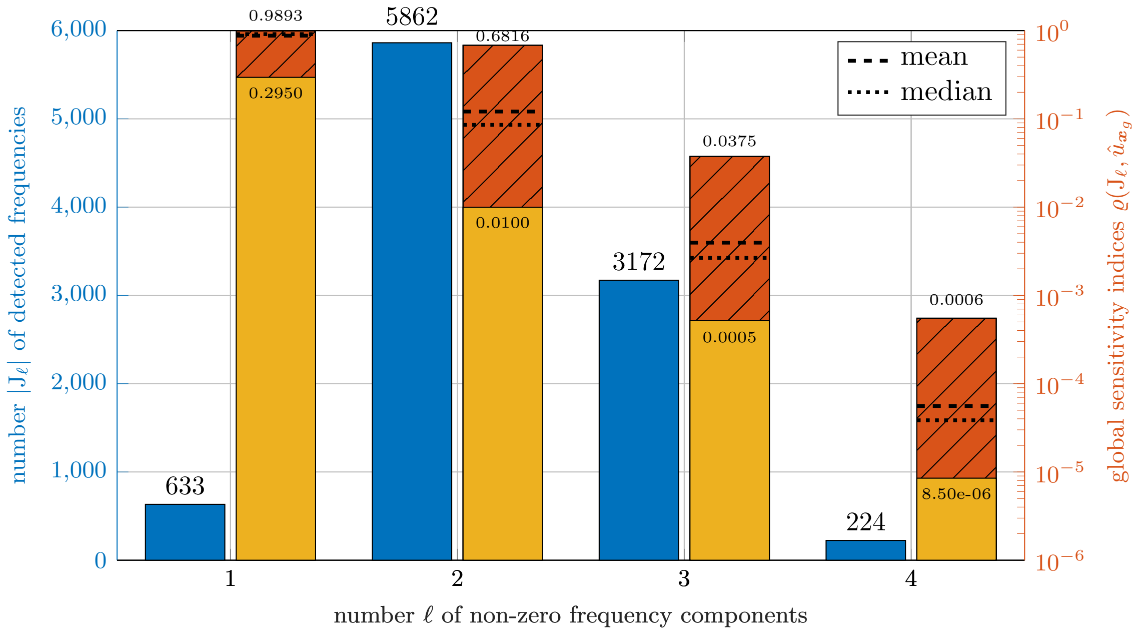


Figure 4.9: Cardinality (left, blue, solid) of the index sets J_ℓ and the corresponding largest (right, orange, striped), smallest (right, yellow, solid), mean (dashed line) and median (dotted line) of the global sensitivity indices $\varrho(\mathrm{J}_\ell, u_{\boldsymbol{x}_g}^{\texttt{usFFT}})$ w.r.t. $\boldsymbol{x}_g$ for the lognormal example with $s = 2000,\ N = 32$.

no reasonably tight theoretical bounds on the size of the output yet, we can further investigate the number of output frequencies in our numerical tests. In detail, we express the detected output sparsity s_{real} as a multiple of the given sparsity s, i.e., $s_{real} = q \cdot s$ with some factor $q \in \mathbb{R}$.

In the numerical tests for the first periodic example in Section 4.2, i.e., $\mu = 1.2$ and $c = 0.4$, we have $q \in [2.41, 2.74]$, where the larger values of q tend to appear for smaller sparsities s. For the quickly decaying example, i.e., $\mu = 3.6$ and $c = 1.5$, we have $q \in [1.9042, 2.45]$ and again the larger values of q are attained for small sparsities s.

The affine model in Section 4.3 results in $q \in [2.06, 2.186]$, where $q < 2.1$ is only attained for $\eta =$ I, II and IV, so parameter settings with small sparsities and $N = 32$.

Finally, in the complicated lognormal case in Section 4.4, we observe $q \in [4.776, 5.15]$. While the values above 5 only appear for $\eta =$ I and II, we still have significantly larger factors q than before because of the higher difficulty of the approximation problem due to the normal distribution of the random variables y_j. Anyway, the magnitude of q is still very small compared to the size $|\mathcal{T}_G| = G = 739$ in our examples.

Our observation is consistent with recent results presented in [18] which considers the periodic model only. The crucial common feature is that the pointwise approximations $u_{\boldsymbol{x}_g}$ can be regarded as elements of a joint reproducing kernel Hilbert space with uniformly good kernel approximants.

Overall, the factor q in our examples is much smaller than G, which would be the worst factor possible in the case that all $\mathrm{I}_{\boldsymbol{x}_g}$ are disjoint. Hence, as already mentioned in Section 1, the given complexities in Theorem 1.1 are way too pessimistic and the true amount of *sampling locations and computational steps needed is much smaller in all of our numerical examples.*

4.5 Comparison to given frequency sets

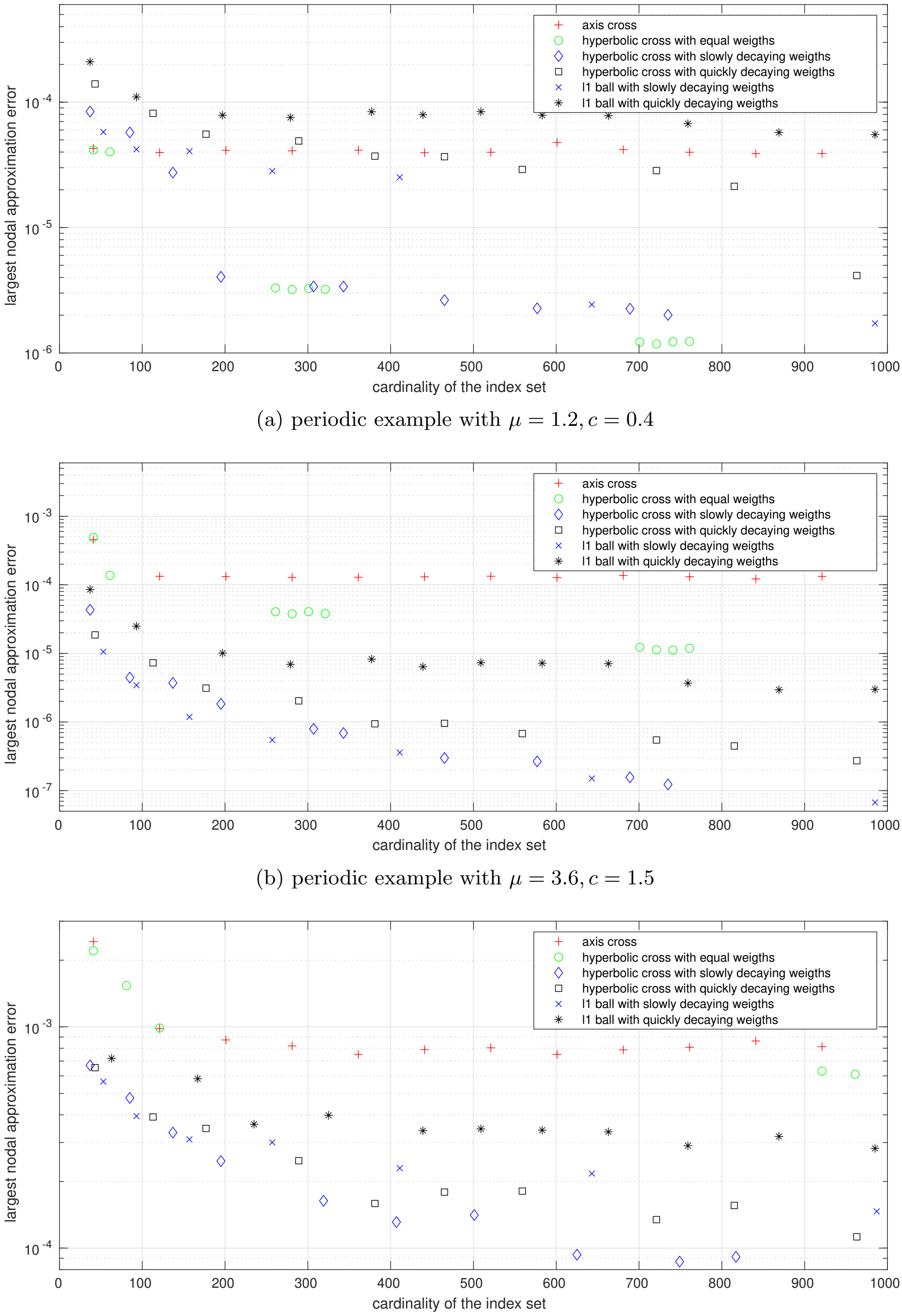
The main effort of the usFFT lies in detecting the index set $\mathrm{I} \subset \Gamma$. The computation of the corresponding Fourier coefficients in the final step of Algorithm 1 needs significantly less samples than the detection steps before. Hence, the question arises, if an a priori choice of the index set I should be preferred to reduce the computational cost, cf. Remark 4.2. Therefore, we now consider the following kinds of index sets:

- axis cross with uniform weight 1: $\mathrm{I} = \{\boldsymbol{k} \in \mathbb{Z}^{d_{\boldsymbol{y}}} : \|\boldsymbol{k}\|_0 = 1, \|\boldsymbol{k}\|_1 \leq N\}$
- hyperbolic cross with uniform weight $\frac{1}{4}$: $\mathrm{I} = \{\boldsymbol{k} \in \mathbb{Z}^{d_{\boldsymbol{y}}} : \prod_{j=1}^{d_{\boldsymbol{y}}} \max(1, 4|k_j|) \leq N\}$
- hyperbolic cross with slowly or quickly ($q = 1$ or 2) decaying weights $\frac{1}{j^q}$: $\mathrm{I} = \{\boldsymbol{k} \in \mathbb{Z}^{d_{\boldsymbol{y}}} : \prod_{j=1}^{d_{\boldsymbol{y}}} \max(1, j^q|k_j|) \leq N\}$
- l_1-ball with slowly or quickly ($q = 1$ or 2) decaying weights $\frac{1}{j^q}$: $\mathrm{I} = \{\boldsymbol{k} \in \mathbb{Z}^{d_{\boldsymbol{y}}} : \sum_{j=1}^{d_{\boldsymbol{y}}} j^q|k_j| \leq N\}$

The Fourier coefficients $c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g})$ are approximated using the same multiple R1L approach as in step 3 of our Algorithm 1, i.e., we just skipped steps 1 and 2 by choosing the index set I instead of detecting it. Figure 4.10 illustrates the largest error err_2^{η} w.r.t. the nodes $\boldsymbol{x}_g$ for the previously considered periodic and affine examples, cf. Sections 4.2 and 4.3, with these given frequency sets I for various refinements N.

The magnitude of the errors is considerably larger than for comparable parameter settings of the usFFT, e.g., $\eta =$ I to III, especially for the periodic example. Further, we see that the particular choice of the structure of the index set also plays a huge role. Obviously, a cleverly chosen index set reduces the size of the approximation error tremendously, especially in the periodic settings. But finding a good or even optimal choice of the index set is highly non-trivial, since it requires sufficient a priori information about the PDE and the structure of its solution or additional computational effort, e.g., to determine suitable weights for a given index set structure. This can be observed for example when comparing the hyperbolic cross index sets for the periodic examples. The uniform weights achieve the best results for $\mu = 1.2$, but can not keep up at all with the decaying weights for the faster decay rate $\mu = 3.6$. On the other hand, even if we know, that there is a certain decay in our random coefficient, it is not clear how to choose suitable decay rates for the weights in order to guarantee reasonable results – specifically in pre-asymptotic settings, which is the rule rather than the exception when numerically determining solutions of high-dimensional problems.

The usFFT does not depend on these kind of information, as its choice of the frequency set is fully adaptive and the only required a priori information is the search space Γ, which can be chosen sufficiently large without disturbing the results of the algorithm. Further, the detected frequency set I provides these additional information about the structure of the solution u as well as the dependence on the random variables $\boldsymbol{y}$. In other words, the additional amount of samples needed for the usFFT makes these structural information unnecessary, detects them on its own and provides a possibility to extract them afterwards from the output.



(a) periodic example with $\mu = 1.2, c = 0.4$

(b) periodic example with $\mu = 3.6, c = 1.5$

(c) affine example

Figure 4.10: Largest error err_2^η w.r.t. the nodes $\boldsymbol{x}_g$ for the periodic and affine examples with given frequency sets.

Remark 4.2. *The here and in step 3 of the usFFT used multiple R1L approach for the efficient computation of Fourier coefficients for a given frequency set* I *was proposed in [20]. From [20, Cor. 3.7] we get a bound on the number of sampling nodes M used. Since we are working with $c = 2$ and $\delta = 0.5$ as in [21, Alg. 3], we arrive at $M \leq \lceil 2\ln(2|\mathrm{I}|)\rceil 4(|\mathrm{I}|-1)$. Note, that this upper bound is very rough and the actual number of used sampling nodes in almost all numerical experiments is much lower.*

As stated above several times, this number of samples used in step 3 of the usFFT is just a small fraction of the total number of used samples when applying the usFFT. In particular, the computation of the actual Fourier coefficients $c_{\boldsymbol{k}}^{\mathit{usFFT}}(u_{\boldsymbol{x}_g})$ for the detected frequency set I *requires roughly* 0.4% *or* 0.3% *of the total sampling amount for the two different parameter choices of the periodic example in Section 4.2, around* 0.1% *in the affine case in Section 4.3 and about* 0.65% *for the lognormal model from Section 4.4.*

In [8, 37], a data-driven method was proposed, which is capable of computing approximations for multiple right-hand sides $f(\boldsymbol{x})$ from a certain function class. Our usFFT approach can also be generalized in a similar, data-driven way: For a given class of functions $f(\boldsymbol{x})$ or even $f(\boldsymbol{x}, \boldsymbol{y})$, we can use the usFFT in order to compute the frequency set I for one randomly selected right-hand side f or randomly select multiple right-hand sides f and compute unions of the corresponding index sets I_f by means of (a slight modification of) the presented usFFT. In each case, we end up with a frequency set I, which is probably a good choice for all the functions f in the given class, since they are hopefully very similar to each other. Hence, we can use this index set I as a starting point and compute approximations of the corresponding Fourier coefficients $c_{\boldsymbol{k}}(u_{\boldsymbol{x}_g}), \boldsymbol{k} \in \mathrm{I}$, as done above. This approximation of $u_{\boldsymbol{x}_g}$ is then probably a lot better, i.e., the detected index set I is a better localization of the largest Fourier coefficients than some a priori choice.

Acknowledegement

L. Kämmerer gratefully acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) with the project number 380648269 and Daniel Potts with the project number 416228727 – SFB 1410.

References

[1] M. Bachmayr, A. Cohen, D. Dũng, and C. Schwab. Fully discrete approximation of parametric and stochastic elliptic PDEs. *SIAM J. Numer. Anal.*, 55(5):2151–2186, 2017.

[2] M. Bachmayr, A. Cohen, and W. Dahmen. Parametric PDEs: sparse or low-rank approximations? *IMA J. Numer. Anal.*, 38(4):1661–1708, 2018.

[3] M. Bachmayr, A. Cohen, R. DeVore, and G. Migliorati. Sparse polynomial approximation of parametric elliptic PDEs. Part II: Lognormal coefficients. *ESAIM Math. Model. Numer. Anal.*, 51(1):341–363, 2017.

[4] M. Bachmayr, A. Cohen, and G. Migliorati. Sparse polynomial approximation of parametric elliptic PDEs. Part I: Affine coefficients. *ESAIM Math. Model. Numer. Anal.*, 51(1):321–339, 2017.

[5] M. Bachmayr, A. Cohen, and G. Migliorati. Representations of Gaussian random fields and approximation of elliptic PDEs with lognormal coefficients. *J. Fourier Anal. Appl.*, 24(3):621–649, 2018.

[6] M. Bochmann, L. Kämmerer, and D. Potts. A sparse FFT approach for ODE with random coefficients. *Adv. Comput. Math.*, 46(5):Paper No. 65, 21, 2020.

[7] J.-L. Bouchot, H. Rauhut, and C. Schwab. Multi-level Compressed Sensing Petrov-Galerkin discretization of high-dimensional parametric PDEs. *ArXiv e-prints*, 2017. arXiv:1701.01671 [math.NA].

[8] M. Cheng, T. Y. Hou, M. Yan, and Z. Zhang. A data-driven stochastic method for elliptic PDEs with random coefficients. *SIAM/ASA J. Uncertain. Quantif.*, 1(1):452–493, 2013.

[9] A. Cohen and R. DeVore. Approximation of high-dimensional parametric PDEs. *Acta Numer.*, 24:1–159, 2015.

[10] A. Cohen, R. DeVore, and C. Schwab. Convergence rates of best N-term Galerkin approximations for a class of elliptic sPDEs. *Found. Comput. Math.*, 10(6):615–646, 2010.

[11] R. Cools, F. Y. Kuo, D. Nuyens, and G. Suryanarayana. Tent-transformed lattice rules for integration and approximation of multivariate non-periodic functions. *J. Complexity*, 36:166–181, 2016.

[12] L. Devroye. *Nonuniform random variate generation.* Springer-Verlag, New York, 1986.

[13] J. Dick, F. Y. Kuo, Q. T. Le Gia, and C. Schwab. Multilevel higher order QMC Petrov-Galerkin discretization for affine parametric operator equations. *SIAM J. Numer. Anal.*, 54(4):2541–2568, 2016.

[14] J. Dick, Q. T. Le Gia, and C. Schwab. Higher order quasi-Monte Carlo integration for holomorphic, parametric operator equations. *SIAM/ASA J. Uncertain. Quantif.*, 4(1):48–79, 2016.

[15] M. Eigel, C. J. Gittelson, C. Schwab, and E. Zander. Adaptive stochastic Galerkin FEM. *Comput. Methods Appl. Mech. Engrg.*, 270:247–269, 2014.

[16] R. N. Gantner, L. Herrmann, and C. Schwab. Multilevel QMC with product weights for affine-parametric, elliptic PDEs. In *Contemporary computational mathematics—a celebration of the 80th birthday of Ian Sloan. Vol. 1, 2*, pages 373–405. Springer, Cham, 2018.

[17] I. G. Graham, F. Y. Kuo, J. A. Nichols, R. Scheichl, C. Schwab, and I. H. Sloan. Quasi-Monte Carlo finite element methods for elliptic PDEs with lognormal random coefficients. *Numer. Math.*, 131(2):329–368, 2015.

[18] V. Kaarnioja, Y. Kazashi, F. Kuo, F. Nobile, and I. Sloan. Fast approximation by periodic kernel-based lattice-point interpolation with application in uncertainty quantification. *ArXiv e-prints*, 2020. arXiv:2007.06367 [math.NA].

[19] V. Kaarnioja, F. Y. Kuo, and I. H. Sloan. Uncertainty quantification using periodic random variables. *SIAM J. Numer. Anal.*, 58(2):1068–1091, 2020.

[20] L. Kämmerer. Constructing spatial discretizations for sparse multivariate trigonometric polynomials that allow for a fast discrete Fourier transform. *Appl. Comput. Harmon. Anal.*, 47(3):702–729, 2019.

[21] L. Kämmerer, D. Potts, and T. Volkmer. High-dimensional sparse FFT based on sampling along multiple rank-1 lattices. *Appl. Comput. Harmon. Anal.*, 51:225–257, 2021.

[22] L. Kämmerer, F. Krahmer, and T. Volkmer. A sample efficient sparse FFT for arbitrary frequency candidate sets in high dimensions. *ArXiv e-prints*, 2020. arXiv:2006.13053 [math.NA].

[23] F. Y. Kuo and D. Nuyens. Application of quasi–Monte Carlo methods to PDEs with random coefficients—an overview and tutorial. In *Monte Carlo and quasi–Monte Carlo methods*, volume 241 of *Springer Proc. Math. Stat.*, pages 53–71. Springer, Cham, 2018.

[24] F. Y. Kuo, D. Nuyens, L. Plaskota, I. H. Sloan, and G. W. Wasilkowski. Infinite-dimensional integration and the multivariate decomposition method. *J. Comput. Appl. Math.*, 326:217–234, Dec. 2017.

[25] F. Y. Kuo, C. Schwab, and I. H. Sloan. Quasi-Monte Carlo finite element methods for a class of elliptic partial differential equations with random coefficients. *SIAM J. Numer. Anal.*, 50(6):3351–3374, 2012.

[26] F. Y. Kuo, C. Schwab, and I. H. Sloan. Multi-level quasi-Monte Carlo finite element methods for a class of elliptic PDEs with random coefficients. *Found. Comput. Math.*, 15(2):411–449, 2015.

[27] D. Li and F. J. Hickernell. Trigonometric spectral collocation methods on lattices. In *Recent advances in scientific computing and partial differential equations (Hong Kong, 2002)*, volume 330 of *Contemp. Math.*, pages 121–132. Amer. Math. Soc., Providence, RI, 2003.

[28] R. Nasdala and D. Potts. Transformed rank-1 lattices for high-dimensional approximation. *Electron. Trans. Numer. Anal.*, 53:239–282, 2020.

[29] D. T. P. Nguyen and D. Nuyens. MDFEM: Multivariate decomposition finite element method for elliptic PDEs with lognormal diffusion coefficients using higher-order QMC and FEM. *ESAIM Math. Model. Numer. Anal.*, 55(4):1461–1505, 2021.

[30] D. T. P. Nguyen and D. Nuyens. MDFEM: Multivariate decomposition finite element method for elliptic PDEs with uniform random diffusion coefficients using higher-order QMC and FEM. *Numer. Math.*, 148(3):633–669, 2021.

[31] D. Potts and M. Schmischke. Approximations of high-dimensional periodic functions with Fourier-based methods. *SIAM J. Numer. Anal.*, 2019. Accepted, arXiv:1907.11412 [math.NA].

[32] D. Potts and T. Volkmer. Sparse high-dimensional FFT based on rank-1 lattice sampling. *Appl. Comput. Harmon. Anal.*, 41(3):713–748, 2016.

[33] C. Schwab. QMC Galerkin discretization of parametric operator equations. In *Monte Carlo and quasi-Monte Carlo methods 2012*, volume 65 of *Springer Proc. Math. Stat.*, pages 613–629. Springer, Heidelberg, 2013.

[34] I. M. Sobol. On sensitivity estimation for nonlinear mathematical models. *Keldysh AppliedMathematics Institute*, 1:112–118, 1990.

[35] I. M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math. Comput. Simulation*, 55(1-3):271–280, 2001.

[36] G. Suryanarayana, D. Nuyens, and R. Cools. Reconstruction and collocation of a class of non-periodic functions by sampling along tent-transformed rank-1 lattices. *J. Fourier Anal. Appl.*, 22(1):187–214, 2016.

[37] Z. Zhang, X. Hu, T. Y. Hou, G. Lin, and M. Yan. An adaptive ANOVA-based data-driven stochastic method for elliptic PDEs with random coefficient. *Commun. Comput. Phys.*, 16(3):571–598, 2014.