ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

MATHICSE

Mathematics Institute of Computational Science and Engineering

School of Basic Sciences - Section of Mathematics

MATHICSE Technical Report
Nr. 13.2011
December 2011

Low-rank tensor Krylov subspace methods for parametrized linear systems

Daniel Kressner, Christine Tobler

http://mathicse.epfl.ch

Phone: +41 21 69 32555

Address:
EPFL - SB - MATHICSE (Bâtiment MA)
Station 8 - CH-1015 - Lausanne - Switzerland

Fax: +41 21 69 32545

LOW-RANK TENSOR KRYLOV SUBSPACE METHODS FOR PARAMETRIZED LINEAR SYSTEMS[*]

DANIEL KRESSNER[†] AND CHRISTINE TOBLER[†]

Abstract. We consider linear systems $A(\alpha)x(\alpha) = b(\alpha)$ depending on possibly many parameters $\alpha = (\alpha_1, \dots, \alpha_p)$. Solving these systems simultaneously for a standard discretization of the parameter range would require a computational effort growing drastically with the number of parameters. We show that a much lower computational effort can be achieved for sufficiently smooth parameter dependencies. For this purpose, computational methods are developed that benefit from the fact that $x(\alpha)$ can be well approximated by a tensor of low rank. In particular, low-rank tensor variants of short-recurrence Krylov subspace methods are presented. Numerical experiments for deterministic PDEs with parametrized coefficients and stochastic elliptic PDEs demonstrate the effectiveness of our approach.

1. Introduction.

Consider a parameter-dependent linear system

$$A(\alpha)x(\alpha) = b(\alpha), \qquad A(\cdot) : \Omega \to \mathbb{R}^{n\times n}, \quad b(\cdot) : \Omega \to \mathbb{R}^n, \tag{1.1}$$

on a compact parameter set $\Omega \subset \mathbb{R}^p$. It is assumed that $A(\alpha)$ is invertible for every $\alpha = \big(\alpha^{(1)}, \dots, \alpha^{(p)}\big) \in \Omega$. This paper is concerned with numerical methods for solving (1.1) for a *large* number of parameter samples. We have mainly two scenarios in mind. One goal of such a computation could be to gather statistics about the solutions over the range of parameters. Another goal could be to interpolate sampled solutions for rapidly solving (1.1) with respect to a parameter configuration that is not known a priori, similar to the reduced basis method [3].

The computational cost of any standard numerical solver applied to (1.1) individually for each parameter sample $\alpha \in \Omega$ inevitably grows proportionally with the number of parameter samples. Already in the one-parameter case ($p = 1$) this may not always be desirable, especially if n is large. As the number of parameters increases, a straightforward discretization of the parameter set would imply an exponentially growing cost, rendering such a naive approach quickly infeasible. In this paper, we combine existing short-recurrence Krylov subspace methods for linear systems with low-rank tensor approximation to achieve a computational cost that is significantly lower and allows the treatment of many parameters.

Existing approaches. In the following, we briefly summarize existing approaches for solving parameter-dependent linear systems of the form (1.1).

Classical linear algebra methods are applicable when A depends linearly on a single parameter, i.e., $A(\alpha_1) = A_0 + \alpha_1 I$ or $A(\alpha_1) = A_0 + \alpha_1 A_1$. Direct methods based on the (generalized) Schur decomposition [18] have been applied to the computation of pseudospectra [45, 48]. Iterative methods exploit the fact that Krylov subspaces are invariant under shifts [16, 17]. These approaches can be easily extended to polynomial dependence on a single parameter by means of (exact) linearization, see [19, 22, 42]. More recently, recycling has been proposed as a means to speed up Krylov subspace methods for linear systems smoothly depending on a single parameter, see, e.g., [10, 27, 38]. While recycling reduces the computational effort, sometimes considerably, it still results in a cost that grows proportionally with the number of parameter samples.

A different class of linear algebra methods applicable for linear dependence on a single parameter is based on reformulating the linear systems (1.1) into a (generalized) Sylvester matrix equation. This point of view admits the application of existing low-rank methods for solving matrix equations, as demonstrated in [41] for the so called extended Krylov subspace method, see also Section 2. An extension of such Krylov subspace methods to *several* parameters is described in [31] under the condition that the coefficients $A_1, \dots, A_p$ in $A(\alpha) = A_0 + \alpha_1 A_1 + \cdots + \alpha_p A_p$ are (or can be transformed to) identities.

In applications with smooth parameter dependence, one can often avoid the use of a parameter sample size that grows exponentially with p. Sparse grid techniques have been successfully used

[*] Supported by the SNF research module *Preconditioned methods for large-scale model reduction* within the SNF ProDoc *Efficient Numerical Methods for Partial Differential Equations*.

[*] Seminar for Applied Mathematics, D-MATH, ETH Zurich, Raemistr. 101, CH-8092 Zurich. {kressner,ctobler}@math.ethz.ch

in collocation methods for stochastic and parameter-dependent PDEs, see, e.g., [35, 36, 5, 6, 9]. These techniques are fairly easy to implement using an existing library [28, 29] for generating the collocation points. Note that these techniques rely on smooth parameter dependence. Smoothness is helpful but not necessary for the success of the tensor-based methods presented in this paper. In comparison to sparse grids, tensor-based methods employ a full tensorized grid and address the challenges of dimensionality at a later stage. In principle, our methods could be combined with sparse grids that can be written as a sum of tensor grids.

Low-rank tensor methods for solving parametrized linear systems have been proposed by Khoromskij and Schwab [26] as well as Ballani and Grasedyck [2]. In [26], linear elliptic PDEs with stochastic coefficients have been considered and a Richardson-like method has been proposed to obtain a low-rank approximation of solutions in the so called CP format. In [2], general high-dimensional linear systems have been considered and a GMRES-like method has been proposed to obtain a low-rank approximation of solutions in the hierachical Tucker format also used in this paper.

Notation. In the following, capital letters denote matrices: A, X, B; capital calligraphic letters denote tensors: $\mathcal{X}, \mathcal{B}$. Calligraphic letters are also used to denote linear operators on matrices or tensors: $\mathcal{A}(X), \mathcal{A}(\mathcal{X})$. The vectorization $\operatorname{vec}(\mathcal{X})$ stacks the entries of a tensor $\mathcal{X}$ into a long vector, with the indices in reverse lexicographic order. The Kronecker product of two matrices is defined as

$$(A \otimes B) = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \text{with } A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{p\times r}, A \otimes B \in \mathbb{R}^{mp\times nr}.$$

All descriptions involving tensors in this paper will be based on vectorizations of tensors and Kronecker products of matrices.

Outline. The rest of this paper is organized as follows. We first discuss the one-parameter case in some detail in Section 2. This mainly serves as an illustration for the algorithmic ideas intended for the multi-parameter case discussed in Section 3. Note, however, that some of the theoretical results are particular to the one-parameter case and do not admit a direct extension to more than one parameter. In Sections 4 and 5, we discuss numerical results for two typical applications of (1.1): linear elliptic PDEs with parametrized coefficients and linear PDEs with stochastic coefficients, respectively. Section 6 illustrates the application of a non-symmetric solver to a parametrized convection-diffusion equation. Finally, Section 7 presents a numerical comparison of our approach with existing methods.

2. One parameter.

To illustrate the main ideas of this paper, we first consider linear systems depending on *one* parameter $\alpha \in [\alpha_{\min}, \alpha_{\max}]$:

$$A(\alpha)x(\alpha) = b(\alpha), \quad A : [\alpha_{\min}, \alpha_{\max}] \to \mathbb{R}^{n\times n}, \quad x, b : [\alpha_{\min}, \alpha_{\max}] \to \mathbb{R}^n. \tag{2.1}$$

After choosing parameter samples $\alpha_{\min} = \alpha_1 < \cdots < \alpha_m = \alpha_{\max}$, we define the matrices

$$B = [b(\alpha_1), \ldots, b(\alpha_m)], \quad X = [x_1, \ldots, x_m] \in \mathbb{R}^{n\times m} \tag{2.2}$$

containing the right-hand sides and solutions of $A(\alpha_i)x_i = b(\alpha_i)$, respectively.

Example 2.1. *Of particular interest is the case of linear dependence: $A(\alpha) = A_0 + \alpha A_1$, where $b(\alpha) \equiv b$ is constant. The corresponding linear systems $(A_0 + \alpha_i A_1)x_i = b$ can be collected into an $mn \times mn$ system*

$$\big(I \otimes A_0 + D_1 \otimes A_1\big)x = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \otimes b, \tag{2.3}$$

where $D_1 = \operatorname{diag}(\alpha_1, \ldots, \alpha_m)$. Alternatively, (2.3) can be written as

$$A_0X + A_1XD_1 = b\,\big[1, \ldots, 1\big], \tag{2.4}$$

which amounts to a Sylvester matrix equation. Low-rank matrix methods for solving such linear matrix equations have been actively developed in the last two decades; we refer to [41] for the application of such a method to (2.4).

2.1. Singular value decay of X. In the following, we use standard arguments to show that the singular values of the matrix X in (2.2) decay exponentially if the entries of both A and b depend analytically on α. We first verify this for the matrix B containing the sampled right-hand sides. Without loss of generality, we may assume that α is in the interval $[-1, 1]$. In the following, $\mathcal{E}_\rho \subset \mathbb{C}$ denotes the open elliptic disc with foci ± 1 and the sum of the half axes equal to ρ.

Lemma 2.2. *Consider a vector valued function $b : [-1, 1] \to \mathbb{R}^n$ with entries having an analytic extension to $\mathcal{E}_{\rho_0}$ for some $\rho_0 > 1$. Then there exists an approximation*

$$\hat{b}(\alpha) = \sum_{j=0}^{k-1} p_j(\alpha) v_j, \tag{2.5}$$

with constant vectors $v_j \in \mathbb{R}^n$ and polynomials $p_j : [-1, 1] \to \mathbb{R}$, such that

$$\max_{\alpha\in[-1,1]} \|b(\alpha) - \hat{b}(\alpha)\| \le \frac{2}{1-\rho^{-1}} \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\|\rho^{-k}.$$

for any $1 < \rho < \rho_0$ and any vector norm $\|\cdot\|$.

Proof. This proof is a straightforward extension of the classical one for functions [33, p. 77]. As b is analytic, we can expand its entries as a Chebyshev series

$$b(\alpha) = \frac{1}{2} v_0 + \sum_{j=1}^{\infty} p_j(\alpha) v_j, \quad p_j(\alpha) = \cos(j \arccos \alpha), \quad v_j = \pi^{-1} \int_{-\pi}^{\pi} b(\cos(t)) \cos(jt) \mathrm{d}t.$$

Formally setting $p_0 \equiv 1/2$, the truncated expansion $\hat{b}(\alpha) = \sum_{j=0}^{k-1} p_j(\alpha) v_j$ satisfies

$$\max_\alpha \|b(\alpha) - \hat{b}(\alpha)\| \le \max_\alpha \Big\| \sum_{j=k}^{\infty} p_j(\alpha) v_j \Big\| \le \sum_{j=k}^{\infty} \|v_j\|. \tag{2.6}$$

To determine an upper bound on $\|v_j\|$ we substitute $z = e^{it}$ and set $g(z) = b(\frac{z+z^{-1}}{2})$, resulting in

$$v_j = \frac{1}{2\pi i} \oint_{|z|=1} g(z)(z^{j-1} + z^{-j-1}) \mathrm{d}z,$$

Since b is analytic on the closure of the elliptic disc, $\overline{\mathcal{E}_\rho}$, g is analytic in the annulus with radii $1/\rho, \rho$. Hence, by changing the path of integration, we obtain

$$v_j = \frac{1}{2\pi i} \oint_{|z|=\rho^{-1}} g(z) z^{j-1} \mathrm{d}z + \frac{1}{2\pi i} \oint_{|z|=\rho} g(z) z^{-j-1} \mathrm{d}z.$$

This shows

$$\begin{aligned}
\|v_j\| &\le \frac{1}{2\pi} \oint_{|z|=\rho^{-1}} \|g(z)\| |z^{j-1}| \mathrm{d}z + \frac{1}{2\pi} \oint_{|z|=\rho} \|g(z)\| |z^{-j-1}| \mathrm{d}z \\
&\le \frac{1}{2\pi} \rho^{-j+1} 2\pi\rho^{-1} \max_{|z|=\rho^{-1}} \|g(z)\| + \frac{1}{2\pi} \rho^{-j-1} 2\pi\rho \max_{|z|=\rho} \|g(z)\| \\
&\le 2 \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\| \rho^{-j},
\end{aligned}$$

which – combined with (2.6) – completes the proof. □

Corollary 2.3. *Under the assumptions of Lemma 2.2, consider the matrix*

$$B = [b(\alpha_1),\, b(\alpha_2),\, \cdots,\, b(\alpha_m)], \quad \alpha_1, \ldots, \alpha_m \in [-1, 1].$$

Then the kth singular value $\sigma_k(B)$ of B satisfies

$$\sigma_k(B) \leq \frac{2\rho\sqrt{m}}{1-\rho^{-1}} \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\|_2 \rho^{-k}. \tag{2.7}$$

Proof. Set $\hat{B} = [\hat{b}(\alpha_1), \ldots, \hat{b}(\alpha_m)]$ with $\hat{b}$ defined in Lemma 2.2. Then the form (2.5) of $\hat{b}$ implies that $\hat{B}$ is a matrix of rank at most $k-1$:

$$\hat{B} = [v_0, v_1, \ldots, v_{k-2}] \cdot \begin{pmatrix} p_0(\alpha_1) & \cdots & p_0(\alpha_m) \\ \vdots & & \vdots \\ p_{k-2}(\alpha_1) & \cdots & p_{k-2}(\alpha_m) \end{pmatrix}.$$

Moreover, the error bound of Lemma 2.2 reveals

$$\|B - \hat{B}\|_F^2 \leq \sum_{i=1}^m \|b(\alpha_i) - \hat{b}(\alpha_i)\|_2^2 \leq m \cdot \Big(\frac{2}{1-\rho^{-1}} \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\|_2 \rho^{-(k-1)}\Big)^2.$$

This completes the proof by the well-known fact that the error of the best rank $k-1$ approximation in the Frobenius norm is given by $\sqrt{\sigma_k^2(B) + \cdots + \sigma_m^2(B)} \geq \sigma_k(B)$. □

Theorem 2.4. *Let $b : [-1,1] \to \mathbb{R}^n$ and $A : [-1,1] \to \mathbb{R}^{n\times n}$ both have analytic extensions to $\mathcal{E}_{\rho_0}$ for some $\rho_0 > 1$. Moreover, the matrix $A(\alpha)$ is assumed to be invertible for all $\alpha \in \mathcal{E}_{\rho_0}$. Consider*

$$X = [x(\alpha_1), x(\alpha_2), \cdots, x(\alpha_m)], \quad \alpha_1, \ldots, \alpha_m \in [-1,1],$$

where each $x(\alpha_i)$ is the solution of the linear system $A(\alpha_i)x(\alpha_i) = b(\alpha_i)$. Then the kth singular value $\sigma_k(X)$ of X satisfies

$$\sigma_k(X) \leq \frac{2\rho\sqrt{m}}{1-\rho^{-1}} \max_{\eta\in\partial\mathcal{E}_\rho} \|A^{-1}(\eta)\|_2 \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\|_2 \rho^{-k},$$

for any $1 < \rho < \rho_0$.

Proof. The entries of $A(\alpha)^{-1}$ are analytic on $\overline{\mathcal{E}_\rho}$ as they can be written as polynomials in the entries of $A(\alpha)$. Hence, $x(\alpha) = A(\alpha)^{-1}b(\alpha)$ is also analytic on $\overline{\mathcal{E}_\rho}$. The statement of the theorem is proven by applying Corollary 2.3 to $x(\alpha)$ and using the estimate

$$\max_{\eta\in\partial\mathcal{E}_\rho} \|x(\eta)\| \leq \max_{\eta\in\partial\mathcal{E}_\rho} \|A^{-1}(\eta)\| \, \|b(\eta)\| \leq \max_{\eta\in\partial\mathcal{E}_\rho} \|A^{-1}(\eta)\| \max_{\eta\in\partial\mathcal{E}_\rho} \|b(\eta)\|.$$

□

Theorem 2.4 shows that the singular values of the solution matrix X decay exponentially. The strength of this decay depends on the domain of analyticity of $A(\cdot)$ and $b(\cdot)$. (For entire functions, the decay will be superexponential.) Hence, we can expect that X can be well approximated by a matrix of very low rank. In the following, we will develop algorithms that benefit from this property.

2.2. Algorithms.

We consider the linear systems $A(\alpha_i)x_i = b(\alpha_i)$ for $i = 1, \ldots, m$. It will be convenient to combine these systems into one large linear system

$$\mathcal{A}x = \begin{pmatrix} A(\alpha_1) & & \\ & \ddots & \\ & & A(\alpha_m) \end{pmatrix} x = \begin{pmatrix} b(\alpha_1) \\ \vdots \\ b(\alpha_m) \end{pmatrix}. \tag{2.8}$$

This can be interpreted as a linear matrix equation $\mathcal{A}(X) = B$ for the matrix $X \in \mathbb{R}^{n\times m}$ with $x = \mathrm{vec}(X)$, where we define $\mathcal{A}(X)$ as the linear operator satisfying $\mathrm{vec}(\mathcal{A}(X)) = \mathcal{A}\mathrm{vec}(X)$.

The operator $\mathcal{A}(\cdot)$ should be in a form that allows for the economic application to low rank matrices. This is the case, for example, if $A(\alpha)$ has the form*

$$A(\alpha) = \sum_{j=1}^{q} f_j(\alpha) A_j, \tag{2.9}$$

with a small number of terms q. Assuming a low rank decomposition of $Y = UV^T$ with $U \in \mathbb{R}^{n\times k}, V \in \mathbb{R}^{m\times k}$, this implies a low rank decomposition for $\mathcal{A}(Y)$:

$$\mathcal{A}(UV^T) = \sum_{j=1}^{q} (A_jU)(V^T f_j(D)) = [A_1U, \ldots, A_qU][f_1(D)V, \ldots, f_q(D)V]^T,$$

with $D = \operatorname{diag}(\alpha_1, \ldots, \alpha_m)$, and therefore significantly reduces the computational cost if $k, q \ll n$.

To derive efficient algorithms for computing low rank approximations to X, we combine existing iterative methods for solving linear systems with low rank truncation. In the following, we consider three iterative methods: *preconditioned Richardson*, *preconditioned CG*, and *preconditioned BiCGstab*.

2.2.1. Preconditioned Richardson method. Formally, we apply the preconditioned first order Richardson method to the block diagonal linear system (2.8), but rephrase all vectors in $\mathbb{R}^{nm}$ as matrices in $\mathbb{R}^{n\times m}$, leading to matrix iterates $X_k \in \mathbb{R}^{n\times m}$. To exploit the singular value decay of B and X shown in Section 2.1, we represent X_k by a low-rank approximation $X_k \approx U_kV_k^T$, and similarly the residuals R_k. All operations of the algorithm can be applied efficiently to matrices in such a low-rank format. In terms of matrices, the preconditioner is a linear operator $\mathcal{M} : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$, which should have a structure that allows $\mathcal{M}^{-1}$ to benefit from low-rank matrices as well. For example, we could choose $\mathcal{M} = I \otimes M$, corresponding to the use of the same preconditioner M for all linear systems $A(\alpha_i)x_i = b(\alpha_i)$.

As the rank will rapidly grow in the course of the iteration, the iterates X_k should be truncated in every iteration. Algorithm 1 describes the final algorithm.

Algorithm 1 Preconditioned Richardson method

```
Input: Matrix functions $\mathcal{A}, \mathcal{M} : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$, right-hand side $B \in \mathbb{R}^{n\times m}$ in low-rank format. Parameter $\omega > 0$, truncation operator $\mathcal{T}$ w.r.t. relative accuracy $\epsilon_{\mathsf{rel}}$.
Output: Matrix $X \in \mathbb{R}^{n\times m}$ fulfilling $\|\mathcal{A}(X) - B\|_F \le \mathsf{tol}$.
  $X_0 = 0$, $R_0 = B$
  while $\|\mathcal{A}(X_k) - B\|_F > \mathsf{tol}$ do
    $X'_{k+1} = X_k + \omega\mathcal{M}^{-1}(R_k)$,    $X_{k+1} = \mathcal{T}(X'_{k+1})$
    $R_{k+1} = B - \mathcal{A}(X_{k+1})$
    $k = k + 1$
  end while
  $X = X_k$
```

In the following, we discuss various aspects of Algorithm 1.

Low-rank truncation. The truncation operator $Y = \mathcal{T}(Y')$ compresses a matrix $Y' = UV^T$ in low-rank format with $U \in \mathbb{R}^{n\times k}, V \in \mathbb{R}^{m\times k}$ such that $\|Y - Y'\|_F \le \epsilon_{\mathsf{rel}}\|Y'\|_F$. For this purpose, QR factorizations $U = Q_UR_U$, $V = Q_VR_V$ with upper triangular $R_U, R_V \in \mathbb{R}^{k\times k}$ are computed. Then a singular value decomposition†

$$R_UR_V^T = \check{U}\operatorname{diag}(\sigma_1, \ldots, \sigma_k)\,\check{V}^T$$

is computed. The truncation rank $\tilde{k} \le k$ is the smallest integer such that

$$\sqrt{\sigma_{\tilde{k}+1}^2 + \cdots + \sigma_k^2} \le \epsilon_{\mathsf{rel}}\sqrt{\sigma_1^2 + \cdots + \sigma_k^2}. \tag{2.10}$$

*Any analytic $A(\alpha)$ can be approximately written as (2.9) by polynomial expansion and truncation.

†Optionally, a product singular value decomposition [15] may be computed, potentially allowing for higher precision if ϵ_{rel} is tiny.

Then, using MATLAB notation, we set $\tilde{U} = U\check{U}(:, 1 : \tilde{k})$ and $\tilde{V} = V\check{V}(:, 1 : \tilde{k})\mathrm{diag}(\sigma_1, \dots, \sigma_{\tilde{k}})$ and obtain the compressed low-rank matrix $Y = \tilde{U}\tilde{V}^T$. Instead of (2.10), one could also use an absolute criterion to determine which singular values to truncate. This would be appropriate if we were to compress the residuals R_k in Algorithm 1.

Choice of ω and $\mathcal{M}$. The choice of the parameter ω strongly influences the convergence of the Richardson method. For symmetric positive definite $\mathcal{A}$ and $\mathcal{M}$, it is well known [39] that the best convergence rate is achieved by

$$\omega = \frac{2}{\lambda_{\min}(\mathcal{M}^{-1}\mathcal{A}) + \lambda_{\max}(\mathcal{M}^{-1}\mathcal{A})}.$$

If $\mathcal{M} = I \otimes M$ then

$$\lambda_{\min}(\mathcal{M}^{-1}\mathcal{A}) = \min_{i=1,\dots,m} \lambda_{\min}\big(M^{-1}A(\alpha_i)\big), \quad \lambda_{\max}(\mathcal{M}^{-1}\mathcal{A}) = \max_{i=1,\dots,m} \lambda_{\max}\big(M^{-1}A(\alpha_i)\big).$$

In general, it is hard to find a matrix M that is optimal in the sense that it minimizes $\kappa\big(M^{-1}A(\alpha)\big)$ uniformly for all $\alpha \in [\alpha_{\min}, \alpha_{\max}]$. Only in the special case of linear parameter dependence $A(\alpha) = A_0 + \alpha A_1$, it is straightforward to show that among all preconditioners of the form $M = A(\tilde{\alpha}) = A_0 + \tilde{\alpha} A_1$, the choice

$$\tilde{\alpha} = \frac{\tilde{\kappa} - 1}{\lambda_{\max}(A_0^{-1}A_1) - \tilde{\kappa}\lambda_{\min}(A_0^{-1}A_1)}$$

is optimal, where

$$\tilde{\kappa} = \sqrt{\kappa_1 \cdot \kappa_2}, \quad \kappa_1 = \frac{1 + \alpha_{\min}\lambda_{\max}}{1 + \alpha_{\min}\lambda_{\min}}, \quad \kappa_2 = \frac{1 + \alpha_{\max}\lambda_{\max}}{1 + \alpha_{\max}\lambda_{\min}}.$$

Convergence. In the absence of low-rank truncation, the error of the Richardson method satisfies

$$\|X^k - X\|_F \le C\gamma^k \|B\|_F,$$

for any γ larger than the spectral radius of $I - \omega\mathcal{M}^{-1}\mathcal{A}$ and some constant $C > 0$. Truncations introduce nonlinear perturbations and henceforth affect the convergence of the Richardson method. Such perturbed fixed point iterations have been analyzed, e.g., in [34, 43]. For general $\mathcal{A}$ and $\mathcal{M}$, this analysis is hard to turn into practical insights due to the particular choice of norms necessary to deal with the effects of non-normality.

If $\mathcal{A}$ is symmetric positive definite then the induced norm $\|Y\|_{\mathcal{A}}^2 := \mathrm{trace}(Y^T\mathcal{A}(Y))$ yields

$$\begin{aligned} \|X - X_{k+1}\|_{\mathcal{A}} &\le \|X - X'_{k+1}\|_{\mathcal{A}} + \|X_{k+1} - X'_{k+1}\|_{\mathcal{A}} \\ &\le \gamma\|X - X_k\|_{\mathcal{A}} + \epsilon_{\mathsf{rel}}\sqrt{\|\mathcal{A}\|_2}\|X'_{k+1}\|_F \end{aligned}$$

with $\gamma = \|I - \omega\mathcal{M}^{-1}\mathcal{A}\|_2$. Hence, convergence progresses as long as

$$\epsilon_{\mathsf{rel}} < \frac{(1-\gamma)\|X - X_k\|_{\mathcal{A}}}{\sqrt{\|\mathcal{A}\|_2}\|X'_{k+1}\|_F} \approx \frac{(1-\gamma)\|X - X_k\|_{\mathcal{A}}}{\sqrt{\|\mathcal{A}\|_2}\|X\|_F}.$$

While this bound is difficult to check in practice, it at least allows for the conclusion that ϵ_{rel} needs to be kept roughly proportional to the current residual norm to retain convergence.

2.2.2. Preconditioned CG method. Similarly to the Richardson method, we apply the preconditioned CG method to the block diagonal linear system (2.8). Using low-rank truncations of the iterates X_k, P_k results in Algorithm 2. Optionally, the iterates R_k and Q_k can also be truncated, to reduce memory requirements and computational effort. Numerical experiments (see Section 2.3.1) reveal that these optional truncations have little or no impact on the convergence of the CG method. This can be explained by the fact that the observed singular value decays for R_k, Q_k are often similar to those of X_k, P_k and hence the truncation error can be expected to be at the same level. It is important to note that we have replaced the standard residual recursion formula $R_{k+1} = R_k - \omega_k \mathcal{A}(P_k)$ by the explicit formula $R_{k+1} = B - \mathcal{A}(X_k)$, because otherwise we observed the method to stagnate much earlier due to truncation error. This replacement also forces the use of non-standard formulas for the coefficients ω_k and β_k, whose derivation does not assume the residual recursion.

Low-rank computation of inner products. Algorithm 2 requires the computation of the matrix inner product

$$\langle Y, Z\rangle = \operatorname{vec}(Y)^T \operatorname{vec}(Z) = \operatorname{trace}(Y^T Z)$$

for two low-rank matrices $Y = U_Y V_Y^T$, $Z = U_Z V_Z^T$ with $U_Y \in \mathbb{R}^{n\times k_Y}$, $V_Y \in \mathbb{R}^{m\times k_Y}$, $U_Z \in \mathbb{R}^{n\times k_Z}$, $V_Z \in \mathbb{R}^{m\times k_Z}$. Trivially,

$$\operatorname{trace}(Y^T Z) = \operatorname{trace}(V_Y U_Y^T U_Z V_Z^T) = \operatorname{trace}\big((V_Z^T V_Y)(U_Y^T U_Z)\big),$$

and hence we first compute $V_Z^T V_Y \in \mathbb{R}^{k_Z\times k_Y}$ ($2mk_Y k_Z$ flops), $U_Y^T U_Z \in \mathbb{R}^{k_Y\times k_Z}$ ($2nk_Y k_Z$ flops), and then the diagonal elements of the product of these two matrices ($2k_Y k_Z$ flops). In total we require $2(m+n+1)k_Y k_Z$ flops.

Convergence. In the absence of truncation error, the convergence of Algorithm 2 can be estimated from the classical bounds:

$$\|X - X_k\|_{\mathcal{A}} \leq \frac{2c^k}{1+c^{2k}} \|X - X_0\|_{\mathcal{A}}, \quad c = \frac{\sqrt{\kappa(\mathcal{A})}-1}{\sqrt{\kappa(\mathcal{A})}+1} < 1,$$

where $\|Y\|_{\mathcal{A}} := \sqrt{\langle Y, \mathcal{A}(Y)\rangle}$. Not displayed by this bound, a merit of the CG method is the occurrence of superlinear convergence effects [47] in the presence of separate eigenvalues clusters. Unfortunately, the eigenvalues of the block diagonal matrix $\mathcal{A}$ tend to fill up intervals as the samples fill up $[\alpha_{\min}, \alpha_{\max}]$. In such a situation, superlinear convergence effects can be expected to disappear.

Finally, note that Algorithm 2 without low-rank truncations coincides for $A(\alpha) \equiv A$ with a so called global Krylov subspace method [24].

Algorithm 2 Preconditioned CG method

```
Input: Matrix functions $\mathcal{A}, \mathcal{M} : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$, right-hand side $B \in \mathbb{R}^{n\times m}$ in low-rank format. Truncation operator $\mathcal{T}$ w.r.t. relative accuracy $\epsilon_{\mathsf{rel}}$.
Output: Matrix $X \in \mathbb{R}^{n\times m}$ fulfilling $\|\mathcal{A}(X) - B\|_F \leq \mathsf{tol}$.
  $X_0 = 0$, $R_0 = B$, $Z_0 = \mathcal{M}^{-1}(R_0)$, $P_0 = Z_0$, $Q_0 = \mathcal{A}(P_0)$
  $\xi_0 = \langle P_0, Q_0\rangle$, $k = 0$
  while $\|R_k\|_F > \mathsf{tol}$ do
    $\omega_k = \langle R_k, P_k\rangle / \xi_k$
    $X_{k+1} = X_k + \omega_k P_k$,            $X_{k+1} \leftarrow \mathcal{T}(X_{k+1})$
    $R_{k+1} = B - \mathcal{A}(X_{k+1})$,       Optionally: $R_{k+1} \leftarrow \mathcal{T}(R_{k+1})$
    $Z_{k+1} = \mathcal{M}^{-1}(R_{k+1})$
    $\beta_k = -\langle Z_{k+1}, Q_k\rangle / \xi_k$
    $P_{k+1} = Z_{k+1} + \beta_k P_k$,          $P_{k+1} \leftarrow \mathcal{T}(P_{k+1})$
    $Q_{k+1} = \mathcal{A}(P_{k+1})$,           Optionally: $Q_{k+1} \leftarrow \mathcal{T}(Q_{k+1})$
    $\xi_{k+1} = \langle P_{k+1}, Q_{k+1}\rangle$
    $k = k + 1$
  end while
  $X = X_k$
```

2.2.3. Preconditioned BiCGstab method. For the case of non-symmetric linear systems, we employ the BiCGstab method as described in [4, Sec. 2.3.8]. Similarly to the Richardson and

CG methods, applying preconditioned BiCGstab to the block diagonal system (2.8) results in Algorithm 3.

As in the case of the CG method, we have experimented with replacing the standard residual recursion formula $R_{k+1} = S_k - \xi_k T_k$ (Variant 1) by the explicit formula $R_{k+1} = B - \mathcal{A}(X_{k+1})$ (Variant 2), aiming at preventing early stagnation of the residual.

Algorithm 3 Preconditioned BiCGstab method

```
Input: Matrix functions $\mathcal{A}, \mathcal{M} : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$, right-hand side $B \in \mathbb{R}^{n\times m}$ in low-rank format, $\tilde{R} \in \mathbb{R}^{n\times m}$ in low-rank format (e.g., $\tilde{R} = B$). Truncation operator $\mathcal{T}$ w.r.t. relative accuracy $\epsilon_{\mathsf{rel}}$.
Output: Matrix $X \in \mathbb{R}^{n\times m}$ fulfilling $\|\mathcal{A}(X) - B\|_F \le \mathsf{tol}$.
  $X_0 = 0$, $R_0 = B$, $\rho_0 = \langle \tilde{R}, R_0\rangle$, $P_0 = R_0$, $\hat{P}_0 = \mathcal{M}^{-1}(P_0)$, $V_0 = \mathcal{A}(\hat{P}_0)$, $k = 0$
  while $\|R_k\|_F > \mathsf{tol}$ do
    $\omega_k = \langle \tilde{R}, R_k\rangle / \langle \tilde{R}, V_k\rangle$
    $S_k = R_k - \omega_k V_k$,                          Optionally: $S_k \leftarrow \mathcal{T}(S_k)$
    $\hat{S}_k = \mathcal{M}^{-1}(S_k)$,                         Optionally: $\hat{S}_k \leftarrow \mathcal{T}(\hat{S}_k)$
    $T_k = \mathcal{A}(\hat{S}_k)$,                           Optionally: $T_k \leftarrow \mathcal{T}(T_k)$
    if $\|S_k\|_F \le \mathsf{tol}$ then $X = X_k + \omega_k \hat{P}_k$, return, end if
    $\xi_k = \langle T_k, S_k\rangle / \langle T_k, T_k\rangle$
    $X_{k+1} = X_k + \omega_k \hat{P}_k + \xi_k \hat{S}_k$,           $X_{k+1} \leftarrow \mathcal{T}(X_{k+1})$
    Variant 1: $R_{k+1} = S_k - \xi_k T_k$,              $R_{k+1} \leftarrow \mathcal{T}(R_{k+1})$
    Variant 2: $R_{k+1} = B - \mathcal{A}(X_{k+1})$,          Optionally: $R_{k+1} \leftarrow \mathcal{T}(R_{k+1})$
    if $\|R_k\|_F \le \mathsf{tol}$ then $X = X_k$, return, end if
    $\rho_{k+1} = \langle \tilde{R}, R_{k+1}\rangle$
    $\beta_k = \rho_{k+1}/\rho_k \, \omega_k/\xi_k$
    $P_{k+1} = R_{k+1} + \beta_k(P_k - \xi_k V_k)$,          $P_{k+1} \leftarrow \mathcal{T}(P_{k+1})$
    $\hat{P}_{k+1} = \mathcal{M}^{-1}(P_{k+1})$,                 Optionally: $\hat{P}_{k+1} \leftarrow \mathcal{T}(\hat{P}_{k+1})$
    $V_{k+1} = \mathcal{A}(\hat{P}_{k+1})$,                    Optionally: $V_{k+1} \leftarrow \mathcal{T}(V_{k+1})$
    $k = k + 1$
  end while
```

2.3. Numerical Examples.

2.3.1. Parametrized stationary heat equation. As an example, we consider the stationary heat equation

$$\begin{aligned} -\nabla(\sigma(x)\nabla u) &= f \quad \text{in } \Omega = [-1,1]^2 \\ u &= 0 \quad \text{on } \Gamma := \partial\Omega. \end{aligned}$$

The heat conductivity coefficient $\sigma(x)$ is assumed to be piecewise constant:

$$\sigma(x) = \begin{cases} 1+\alpha & \text{for } x \in \mathcal{D}, \\ 1 & \text{for } x \notin \mathcal{D}, \end{cases}$$

where $\mathcal{D} \subset \Omega$ is a disc of radius 0.5 and $\alpha \in [0, 100]$ is the parameter. This system is discretized by a finite element formulation with piecewise linear basis functions on the mesh displayed in Figure 2.1. The resulting 371×371 linear system takes the form $(A_0 + \alpha A_1)x(\alpha) = b$. We choose the preconditioner $\mathcal{M} = I \otimes M$ with $M = A_0 + \tilde{\alpha} A_1$, where $\tilde{\alpha}$ is optimally chosen as discussed in Section 2.2.1. The source term is assumed to be constant: $f \equiv 1$.

The set of parameter samples is $\{\alpha_1, \ldots, \alpha_{101}\} = \{0, \ldots, 100\}$. The singular values of the resulting solution matrix X are displayed in Figure 2.1, which confirms the exponential decay predicted by Theorem 2.4. (Note that singular values smaller than 10^{-14} are corrupted by roundoff error.)

Figure 2.2 displays the residual norm $\|\mathcal{A}(X_k) - B\|_F / \|B\|_F$ for the iterates of the preconditioned Richardson and CG methods, respectively. For the Richardson method, the observed convergence is monotone albeit rather slow. More importantly, turning on low-rank truncation

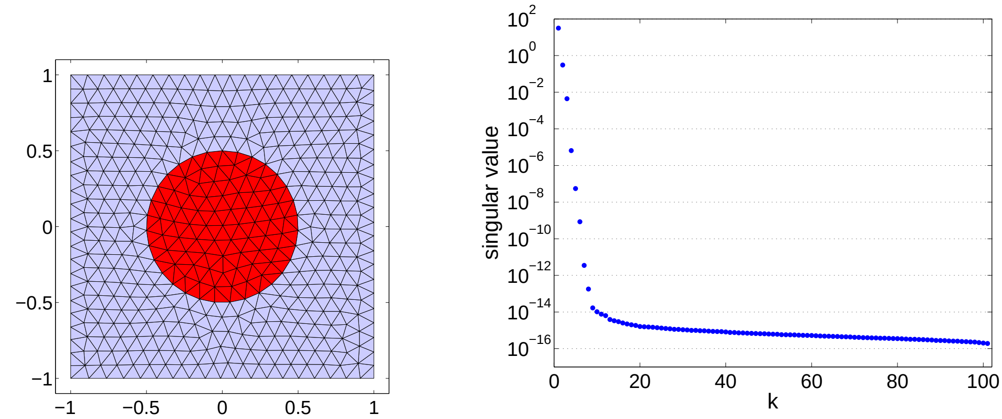

Fig. 2.1. *Left: Mesh discretization. Right: Singular value decay of the solution matrix X.*

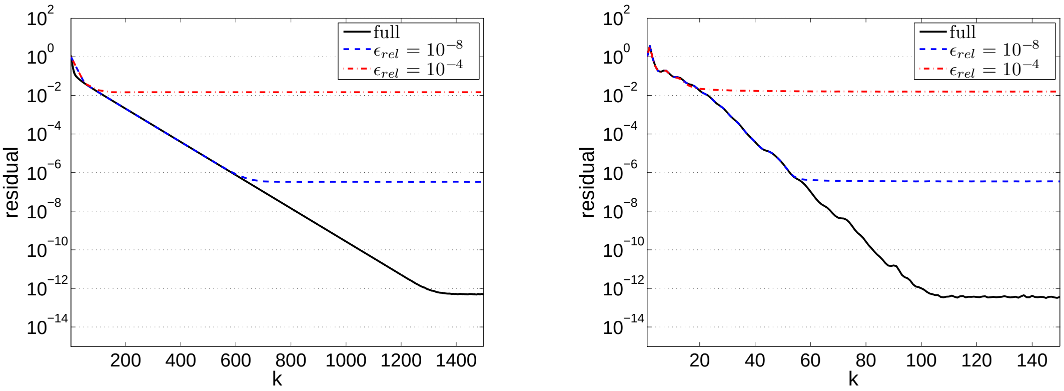

Fig. 2.2. *Left: Preconditioned Richardson method without (full) and with low-rank truncations ($\epsilon_{\mathsf{rel}} \in \{10^{-8}, 10^{-4}\}$). Right: Preconditioned CG method without (full) and with low-rank truncations ($\epsilon_{\mathsf{rel}} \in \{10^{-8}, 10^{-4}\}$). Note the different scales of the x-axis in the two cases.*

does not spoil the convergence until the final accuracy determined by ϵ_{rel} is reached. The convergence of the CG method is significantly faster compared to the Richardson method, without and with low-rank truncations. Again, truncations do not spoil the convergence until the final accuracy is reached. We observed no visible difference in the convergence plots when turning on or turning off the truncations marked optional in Algorithm 2.

2.3.2. Parametrized convection-diffusion equation. Let us now consider the stationary convection-diffusion equation

$$\begin{aligned} -\nabla(\sigma(x)\nabla u) + c^T \nabla u = f \quad & \text{in } \Omega = [-1,1]^2 \\ u = 0 \quad & \text{on } \Gamma := \partial\Omega. \end{aligned}$$

We choose $c = (2,0)^T$, and proceed with the discretization as for the case of the stationary heat equation (Section 2.3.1).

The singular value decay of the solution matrix X (see Figure 2.3) is almost as strong as for the heat equation example above. Figure 2.3 displays the results from applying the preconditioned CG method to the normal equations, which exhibits – as expected – rather slow convergence.

Figure 2.4 displays results from applying the two variants of the BiCGstab method described in Algorithm 3. Variant 1 uses formula $R_{k+1} = S_k + \xi_k T_k$, which gets affected by low-rank truncations. In effect, the norm of R_k becomes much smaller than the actual residual norm $\|B - \mathcal{A}(X_k)\|_F / \|B\|_F$, which stagnates roughly at the level of the truncation error. Variant 2, which uses the true residual $R_{k+1} = B - \mathcal{A}(X_{k+1})$, converges initially at a similar rate. However, the convergence behavior becomes more erratic when the final accuracy is attained. Note that

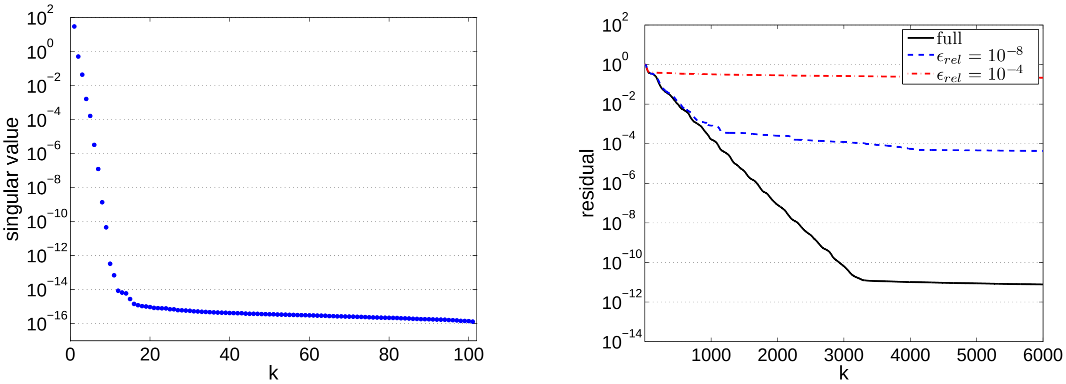

FIG. 2.3. *Left: Singular value decay of the solution matrix X for convection-diffusion example. Right: Preconditioned CG method applied to normal equations without (full) and with low-rank truncations* ($\epsilon_{\text{rel}} \in \{10^{-8}, 10^{-4}\}$).

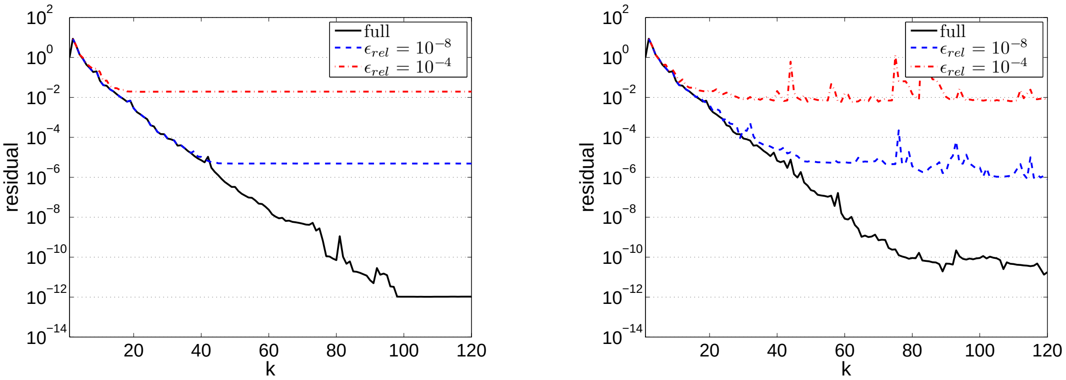

FIG. 2.4. *Preconditioned BiCGstab method without (full) and with low-rank truncations* ($\epsilon_{\text{rel}} \in \{10^{-8}, 10^{-4}\}$). *Left:* Variant 1. *Right:* Variant 2.

such an erratic behavior was avoided in the CG method by the use of non-standard recursion formulas. Unfortunately, it is not clear how this idea can be extended to BiCGstab. Turning on or turning off optional truncations in Algorithm 3 was observed to have no significant impact on the convergence behavior.

3. Multiple parameters. The basic ideas for the one-parameter case extend in a direct fashion to the multi-parameter case:

$$A(\alpha)\, x(\alpha) = b(\alpha), \qquad \alpha \in \Omega := \big[\alpha_{\min}^{(1)}, \alpha_{\max}^{(1)}\big] \times \cdots \times \big[\alpha_{\min}^{(p)}, \alpha_{\max}^{(p)}\big],$$

where $A : \Omega \to \mathbb{R}^{n\times n}, b : \Omega \to \mathbb{R}^{n}$, and $A(\alpha)$ invertible for all $\alpha \in \Omega$. We sample each parameter individually: $\{\alpha_1^{(\mu)}, \dots, \alpha_{m_\mu}^{(\mu)}\} \subset [\alpha_{\min}^{(\mu)}, \alpha_{\max}^{(\mu)}]$ for $\mu = 1, \dots, p$, resulting in a tensor-grid sampling of Ω:

$$\alpha_{\mathfrak{I}} = (\alpha_{i_1}, \dots, \alpha_{i_p}), \qquad i_\mu = 1, \dots, m_\mu, \qquad \mu = 1, \dots, p,$$

with multi-index $\mathfrak{I} = (i_1, \dots, i_p)$. This leads to $m_1 m_2 \cdots m_p$ linear systems

$$A(\alpha_{\mathfrak{I}})\, x_{\mathfrak{I}} = b(\alpha_{\mathfrak{I}}), \qquad \text{with } x_{\mathfrak{I}} = x(\alpha_{\mathfrak{I}}) \in \mathbb{R}^{n}.$$

The solutions $x_{i_1,\dots,i_p} \in \mathbb{R}^n$ are assembled in a tensor $\mathcal{X} \in \mathbb{R}^{n \times m_1 \times \cdots \times m_p}$ or stacked into a long vector $x = \operatorname{vec}(\mathcal{X}) \in \mathbb{R}^{n m_1 \cdots m_p}$, and similarly for the right-hand sides $b(\alpha_{\mathfrak{I}})$. This leads to a linear system

$$\mathcal{A} x = b, \tag{3.1}$$

where $\mathcal{A}$ is a block diagonal matrix containing the system matrices $A(\alpha_{i_1,\ldots,i_p})$ on the diagonal.

Example 3.1. *In the case of linear parameter dependence, $A(\alpha) = A_0 + \alpha^{(1)} A_1 + \cdots + \alpha^{(p)} A_p$, the matrix $\mathcal{A}$ takes the form*

$$\mathcal{A} = I \otimes I \otimes \cdots \otimes A_0 + I \otimes \cdots \otimes D_1 \otimes A_1 + \cdots + D_p \otimes I \otimes \cdots \otimes I \otimes A_p, \tag{3.2}$$

with $D_\mu = \operatorname{diag}(\alpha_1^{(\mu)}, \ldots, \alpha_{m_\mu}^{(\mu)})$.

3.1. Approximation of x by low-rank tensors. The exact solution of the linear system (3.1) is computationally intractable for more than a few parameters unless the number of samples per parameter is ridiculously small. To approach this problem, we will approximate the solution in a low-rank tensor format; the aim of this section is to provide the theoretical justification for this approximation. In the following, we adapt an error bound from sparse tensor approximation of stochastic PDEs [6] to the case of low-rank tensor approximation. We start by showing that a multivariate analytic vector-valued function can be well approximated by a short sum of separable functions.

By a suitable transformation, we may assume without loss of generality that the parameter range is $\Omega = [-1, 1]^p$. We first consider a scalar-valued function $f : \Omega \to \mathbb{R}$, which is expanded in terms of a Fourier-Legendre series:

$$f(\alpha) = \sum_{\mathfrak{J} \in \mathbb{N}^p} c_{\mathfrak{J}} \mathcal{P}_{\mathfrak{J}}(\alpha), \qquad \mathfrak{J} = (j_1, \ldots, j_p),$$

where $\mathcal{P}_{\mathfrak{J}}(\alpha) = P_{j_1}(\alpha_1) \cdots P_{j_p}(\alpha_p)$ is a product of Legendre polynomials and

$$c_{\mathfrak{J}} = \Big(\prod_{\mu=1}^{p} \frac{2 j_\mu + 1}{2} \Big) \int_{[-1,1]^p} f(\alpha) \mathcal{P}_{\mathfrak{J}}(\alpha) \mathrm{d}\alpha. \tag{3.3}$$

We further define the open elliptic polydisc $\mathcal{E}_{\rho_0}^{\times} = \mathcal{E}_{\rho_0} \times \cdots \times \mathcal{E}_{\rho_0}$, where $\mathcal{E}_{\rho_0} \subset \mathbb{C}$ is again the open elliptic disc with foci ±1 and sum of half axes equal to ρ_0.

Lemma 3.2. *Consider a function $f : [-1, 1]^p \to \mathbb{R}$ having an analytic extension to the polydisc $\mathcal{E}_{\rho_0}^{\times}$. Then the Fourier-Legendre coefficients $c_{\mathfrak{J}}$ defined in (3.3) satisfy*

$$|c_{\mathfrak{J}}| \le \Big(\prod_{\mu=1}^{p} \frac{2 j_\mu + 1}{2} \Big) (\rho - 1)^{-p} \cdot \rho^{-\sum_{\mu=1}^{p} j_\mu} \cdot \|f\|_{L^1(\Gamma)} =: \gamma_{\mathfrak{J}},$$

for any $1 \le \rho < \rho_0$ and $\Gamma := \partial\mathcal{E}_\rho \times \cdots \times \partial\mathcal{E}_\rho$.

Proof. Our proof follows the proof of Lemma A.3 in [7], which considers a slightly different setting in the context of deterministic expansions of stochastic PDEs. By repeatedly applying the Cauchy integral formula in each variable, we find

$$f(\alpha) = \frac{1}{(2\pi i)^p} \int_\Gamma \frac{f(z)}{(z_1 - \alpha_1) \cdots (z_p - \alpha_p)} \mathrm{d}z,$$

where $\Gamma := \partial\mathcal{E}_\rho \times \cdots \times \partial\mathcal{E}_\rho$. Inserting into (3.3) gives

$$\begin{aligned} c_{\mathfrak{J}} &= \Big(\prod_{\mu=1}^{p} \frac{2 j_\mu + 1}{2} \Big) \frac{1}{(2\pi i)^p} \int_{[-1,1]^p} \int_\Gamma \frac{f(z) \mathcal{P}_{\mathfrak{J}}(\alpha)}{(z_1 - \alpha_1) \cdots (z_p - \alpha_p)} \mathrm{d}z \, \mathrm{d}\alpha \\ &= \Big(\prod_{\mu=1}^{p} \frac{2 j_\mu + 1}{2} \Big) \frac{1}{(2\pi i)^p} \int_\Gamma f(z) \Big(\prod_{\mu=1}^{p} \int_{-1}^{1} \frac{P_{j_\mu}(\alpha_\mu)}{(z_\mu - \alpha_\mu)} \mathrm{d}\alpha_\mu \Big) \mathrm{d}z. \end{aligned}$$

Using the fact that $Q_{j_\mu}(z_\mu) = \frac{1}{2} \int_{-1}^{1} \frac{P_{j_\mu}(\alpha_\mu)}{(z_\mu - \alpha_\mu)} \mathrm{d}\alpha_\mu$ is the Legendre polynomial of the second kind, and setting $\mathcal{Q}_{\mathfrak{J}}(z) = \prod_{\mu=1}^{p} Q_{j_\mu}(z_\mu)$, we thus have

$$c_{\mathfrak{J}} = \Big(\prod_{\mu=1}^{p} \frac{2 j_\mu + 1}{2} \Big) \frac{1}{(\pi i)^p} \int_\Gamma f(z) \mathcal{Q}_{\mathfrak{J}}(z) \mathrm{d}z.$$

Using $\sup\limits_{z\in\Gamma}|\mathcal{Q}_{\mathfrak{J}}(z)| \leq \prod\limits_{\mu=1}^{p} \pi\frac{\rho^{-j_\mu-1}}{1-\rho^{-1}}$, see (A.21) in [7], leads to the bound

$$|c_{\mathfrak{J}}| \leq \Big(\prod_{\mu=1}^{p}\frac{2j_\mu+1}{2}\Big)\frac{1}{\pi^p}\sup_{z\in\Gamma}|\mathcal{Q}_{\mathfrak{J}}(z)|\int_\Gamma |f(z)|\mathrm{d}z \leq \Big(\prod_{\mu=1}^{p}\frac{2j_\mu+1}{2}\Big)\prod_{\mu=1}^{p}\frac{\rho^{-j_\mu-1}}{1-\rho^{-1}}\|f\|_{L^1(\Gamma)},$$

which completes the proof. □

As a next step, we find an upper bound on the best approximation error $\inf\limits_{f_k}\|f-f_k\|_\infty$ in the supremum norm on Ω, where f_k is any function with only k non-zero coefficients $c_{\mathfrak{J}}$. The following lemma, attributed to Stechkin in [11], will prove very useful for this purpose.

Lemma 3.3. *Consider $q, r \in \mathbb{R}$ with $0 < q \leq r < \infty$, and the coefficients $(c_{\mathfrak{J}})_{\mathfrak{J}\in\mathbb{N}^p} \in \ell^r(\mathbb{N}^p)$. For $k \in \mathbb{N}$, choose $\Lambda_k \subset \mathbb{N}^p$ of cardinality k such that $|c_{\mathfrak{J}}| \geq |c_{\mathfrak{L}}|$ for all $\mathfrak{J} \in \Lambda_k$ and $\mathfrak{L} \in \mathbb{N}^p\backslash\Lambda_k$. Then*

$$\Big(\sum_{\mathfrak{J}\in\Lambda\backslash\Lambda_k}|c_{\mathfrak{J}}|^r\Big)^{1/r} \leq k^{-s}\|c_{\mathfrak{J}}\|_{\ell^q}, \qquad \textit{with } s = \frac{1}{q}-\frac{1}{r} \geq 0.$$

Proof. Construct a rearrangement $(\gamma_n)_{n\geq 1}$ of $|c_{\mathfrak{J}}|$ fulfilling $\gamma_n \geq \gamma_{n+1}$ for all n. We then have

$$\Big(\sum_{n=k+1}^{\infty}\gamma_n^r\Big)^{1/r} \leq \gamma_k^{1-q/r}\Big(\sum_{n=k+1}^{\infty}\gamma_n^q\Big)^{1/r} \leq \gamma_k^{1-q/r}\|\gamma_n\|_{\ell^q}^{q/r}. \tag{3.4}$$

Using $k\gamma_k^q \leq \|\gamma_n\|_{\ell^q}^q$ gives $k^s\gamma_k^{1-q/r} = k^s\gamma_k^{sq} \leq \|\gamma_n\|_{\ell^q}^{sq} = \|\gamma_n\|_{\ell^q}^{1-q/r}$ which, combined with (3.4), proves the statement. □

Lemma 3.4. *Consider a function $f : [-1,1]^p \to \mathbb{R}$ as in Lemma 3.2 with the bounds $\gamma_{\mathfrak{J}}$ on its Fourier-Legendre coefficients. Choose $\Lambda_k \subset \mathbb{N}^p$ such that $\{\gamma_{\mathfrak{J}} : \mathfrak{J} \in \Lambda_k\}$ contains the k largest $\gamma_{\mathfrak{J}}$. Setting $f_k(\alpha) = \sum_{\mathfrak{J}\in\Lambda_k} c_{\mathfrak{J}}\mathcal{P}_{\mathfrak{J}}(\alpha)$, we have*

$$\|f-f_k\|_\infty \leq k^{-s}\|\gamma_{\mathfrak{J}}\|_{\ell^q}, \qquad \textit{for any } 0 < q \leq 1,\ s = \frac{1}{q}-1.$$

Proof. Using that the supremum norm of Legendre polynomials is 1, we obtain

$$\|f(\alpha)-f_k(\alpha)\|_\infty = \Big\|\sum_{\mathfrak{J}\in\Lambda\backslash\Lambda_k} c_{\mathfrak{J}}\mathcal{P}_{\mathfrak{J}}(\alpha)\Big\|_\infty \leq \sum_{\mathfrak{J}\in\Lambda\backslash\Lambda_k}|c_{\mathfrak{J}}|\cdot\|\mathcal{P}_{\mathfrak{J}}(\alpha)\|_\infty \leq \sum_{\mathfrak{J}\in\Lambda\backslash\Lambda_k}\gamma_{\mathfrak{J}}.$$

Applying Stechkin's lemma with $r = 1$ yields the desired result. □

Remark 3.5. *Lemma 3.4 implies that the error decays stronger than any polynomial in k. However, note that the constant $\|\gamma_{\mathfrak{J}}\|_{\ell^q} \to \infty$ as $q \to 0$. A good choice of $q \in (0,1]$ that balances these factors depending on k appears to be difficult to derive analytically. Inserting the bound from Lemma 3.2 into the result of Lemma 3.4 leads to*

$$\begin{aligned}
\|f(\alpha)-f_k(\alpha)\|_\infty &\leq k^{-s}\Big(\frac{1}{\rho-1}\Big)^p\|f\|_{L^1(\Gamma)}\Big(\sum_{\mathfrak{J}\in\mathbb{N}^p}\Big(\prod_{\mu=1}^{p}\frac{2j_\mu+1}{2}\rho^{-j_\mu}\Big)^q\Big)^{1/q}\\
&= k^{-s}\Big(\frac{1/2}{\rho-1}\Big)^p\|f\|_{L^1(\Gamma)}\Big(\sum_{j=0}^{\infty}(2j+1)^q\rho^{-jq}\Big)^{p/q}\\
&= k^{-s}(\rho-1)^{-p}\|f\|_{L^1(\Gamma)}\Phi(\rho^{-q},-q,1/2)^{p/q},
\end{aligned}$$

where Φ denotes the Lerch transcendent.

The *tensor rank* of a tensor $\mathcal{X}$ is defined as the minimal k such that $\mathcal{X}$ can be decomposed as a sum of k rank-one tensors:

$$\operatorname{vec}(\mathcal{X}) = \sum_{j=1}^{k} v_j^{(1)} \otimes \cdots \otimes v_j^{(p)}. \tag{3.5}$$

This is also called *CP decomposition* of $\mathcal{X}$. The tensor rank provides an upper bound on the multilinear ranks and hierarchical ranks discussed in Section 3.2 below. The following theorem gives a bound on the best approximation error by a tensor of tensor rank k.

THEOREM 3.6. *Let $b : [-1,1]^p \to \mathbb{R}^n$ and $A : [-1,1]^p \to \mathbb{R}^{n\times n}$, where each element of b, A is assumed to have an analytic extension to the open polydisc $\mathcal{E}_{\rho_0}^{\times}$. Moreover, the matrix $A(\alpha)$ is assumed to be invertible for all $\alpha \in \mathcal{E}_{\rho_0}^{\times}$. Consider $x(\alpha) = A(\alpha)^{-1}b(\alpha)$, and the tensor $\mathcal{X} \in \mathbb{R}^{n\times m_1\times\cdots\times m_p}$ defined for $\mathfrak{I} = (i_1, \ldots i_p)$ by $(x_{\mathfrak{I}})_{i_0} = x_{i_0}(\alpha_{\mathfrak{I}})$, where $i_0 = 1, \ldots, n$ and $i_\mu = 1, \ldots, m_\mu$ for $\mu = 1, \ldots, p$.*

Then there is an approximation $\mathcal{X}^{(k)}$ of tensor rank k such that, for any choice of $s = \frac{1}{q} - 1$ with $0 < q \le 1$,

$$\|\operatorname{vec}(\mathcal{X} - \mathcal{X}^{(k)})\|_\infty \le C\, k^{-s},$$

where

$$C := \Big(\frac{1/2}{\rho - 1}\Big)^p \max_{i_0=1,\ldots,n} \|x_{i_0}(\alpha)\|_{L^1(\Gamma)} \Big(\sum_{j=0}^{\infty} (2j+1)^q \rho^{-jq}\Big)^{p/q}.$$

Proof. By the same argument used in the proof of Theorem 2.4, the function $x : [-1,1]^p \to \mathbb{R}^n$ is analytic in each variable on $\mathcal{E}_{\rho_0}^{\times}$. We apply Lemma 3.4 to $x_{i_0}(\alpha)$:

$$x_{i_0}^{(k)}(\alpha) = \sum_{\mathfrak{J}\in\Lambda_k} c_{\mathfrak{J}}^{(i_0)} \mathcal{P}_{\mathfrak{J}}(\alpha), \qquad \text{with } \|x_{i_0}(\alpha) - x_{i_0}^{(k)}(\alpha)\|_\infty \le Ck^{-s}.$$

Note that the choice of Λ_k only depends on ρ, which is the same for all $i_0 = 1, \ldots, n$, allowing us to write

$$\mathcal{X}_{i_0,i_1,\ldots,i_p}^{(k)} = \sum_{\mathfrak{J}\in\Lambda_k} c_{\mathfrak{J}}^{(i_0)} P_{j_1}(\alpha_{i_1}^{(1)}) \cdots P_{j_p}(\alpha_{i_p}^{(p)}).$$

By construction, $\mathcal{X}^{(k)}$ has tensor rank k or smaller. □

An error bound in the Euclidean norm can be easily obtained from Theorem 3.6 using the inequality

$$\|\operatorname{vec}(\mathcal{X} - \mathcal{X}^{(k)})\|_2^2 \le n \cdot m_1 \cdots m_p \|\operatorname{vec}(\mathcal{X} - \mathcal{X}^{(k)})\|_\infty^2.$$

3.2. Low-rank multilinear decompositions.

Applying low-rank methods – as, e.g., in Section 2.2 – to linear systems with more than one parameter requires a suitable low-rank tensor decomposition. As the storage requirements for an explicitly stored tensor increase exponentially with its order, such a decomposition becomes mandatory alone for the storage of the solution tensor. On the other hand, we must also be able to perform certain operations with this decomposition in a robust and efficient manner. In the context of the iterative solvers considered in this paper, we require the following operations.

- *Addition of two tensors.*
- *Truncation to low-rank tensor:* Approximate a low-rank tensor by a tensor of even lower tensor rank. For our purpose, it is important that this truncation can be implemented as a black box, in particular without parameter tuning. On the other hand, there is no need to obtain a best or nearly best approximation.
- *μ-mode matrix product:* The multiplication of a matrix on the μth mode of a tensor is defined as

$$\Big((A)_\mu \mathcal{X}\Big)_{\mathfrak{I}} := \sum_{l=1}^{n_\mu} A_{i_\mu,l}\, \mathcal{X}_{i_1,\ldots,i_{\mu-1},l,i_{\mu+1},\ldots,i_d}, \quad A \in \mathbb{R}^{m_\mu\times n_\mu}, \quad \mathcal{X} \in \mathbb{R}^{n_1\times\cdots\times n_d}.$$

In the case of linear parameter dependence, all matrix-tensor multiplications can be performed by a combination of μ-mode matrix products and additions.

- *Tensor inner product and Euclidean norm:* The tensor scalar product is defined as
$$\langle \mathcal{X}, \mathcal{Y} \rangle = \langle \mathrm{vec}(\mathcal{X}), \mathrm{vec}(\mathcal{Y}) \rangle = \sum_{\mathfrak{I} \le \mathfrak{N}} \mathcal{X}_{\mathfrak{I}} \mathcal{Y}_{\mathfrak{I}}, \qquad \text{where } \mathfrak{N} = (n_1, \ldots, n_d),$$
with the induced Euclidean norm $\|\mathcal{X}\|_2 = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}$.

3.2.1. Review of CP and Tucker decomposition. In view of the requirements above, it turns out that classical low-rank tensor formats are not perfectly suited for our purpose. In the following, we briefly explain this for the CP and the Tucker decompositions.

The CP decomposition is defined in (3.5) as the decomposition into a sum of rank-one tensors. Even though its storage requirements are minimal, the decomposition is affected by mathematical as well as algorithmic subtleties [13]. Recently developed methods for low-rank approximation in CP decomposition can be found in [1, 14, 30] and overcome some of these subtleties. However, many of these methods have been tuned for data analysis applications and low-rank recovery; further research is needed to investigate their applicability to highly accurate approximation of function-related tensors.

The Tucker decomposition [46] for a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ takes the form
$$\mathrm{vec}(\mathcal{X}) = (U_1 \otimes \cdots \otimes U_d)\,\mathrm{vec}(\mathcal{C}), \quad U_\mu \in \mathbb{R}^{n_\mu \times r_\mu}, \quad \mathcal{C} \in \mathbb{R}^{r_1 \times \cdots \times r_d},$$
where each matrix U_μ has orthonormal columns. The obvious drawback of this decomposition is that the storage for the so called *core tensor* $\mathcal{C}$ still grows exponentially with the number of dimensions. The great benefit, however, is that low-rank truncation can be easily achieved by means of the Higher Order SVD (HOSVD) introduced in [12].

For each mode μ of the tensor $\mathcal{X}$, the HOSVD considers the corresponding matricization $X_{(\mu)} \in \mathbb{R}^{n_\mu \times n_1 \cdots n_{\mu-1} n_{\mu+1} \cdots n_d}$. Given a user-specified rank $r_\mu \le n_\mu$, the matrix U_μ is defined to contain the r_μ most significant left singular vectors of $X_{(\mu)}$. Once all U_μ are computed, the core tensor is set to $\mathrm{vec}(\mathcal{C}) = (U_1^T \otimes \cdots \otimes U_d^T)\mathrm{vec}(\mathcal{X})$. We can thus interpret the truncation as a projection:
$$\mathrm{vec}(\tilde{\mathcal{X}}) = (U_1 U_1^T \otimes \cdots \otimes U_d U_d^T)\mathrm{vec}(\mathcal{X}). \tag{3.6}$$

The Tucker decomposition introduces a new rank concept, the *multilinear ranks* (or Tucker ranks) $r_1, \ldots, r_d$, where $r_\mu = \operatorname{rank} X_{(\mu)}$ for $\mu = 1, \ldots, d$. Note that every multilinear rank r_i is bounded by the tensor rank R of $\mathcal{X}$, as the matricization $X_{(\mu)}$ can be represented as the sum of R rank-one matrices. Since $U_\mu U_\mu^T X_{(\mu)}$ is the best rank-r_μ approximation of $X_{(\mu)}$ in the Frobenius norm, and the Euclidean norm of $\mathcal{X}$ is the Frobenius norm of $X_{(\mu)}$, the approximation error of the low-rank truncation (3.6) is bounded by
$$\|\mathcal{X} - \tilde{\mathcal{X}}\|_2 \le \sqrt{d}\|\mathcal{X} - \mathcal{X}_r^{\mathsf{best}}\|_2,$$
where $\mathcal{X}_r^{\mathsf{best}}$ is a best possible approximation with multilinear ranks $r = (r_1, \ldots, r_d)$.

3.2.2. Review of hierarchical Tucker decomposition. The shortcomings of the classical decompositions have sparked the development of alternative decompositions, aiming at combining the advantages of CP and Tucker while avoiding their disadvantages. In the following, we consider the hierarchical Tucker decomposition (HTD) recently proposed by Hackbusch and Kühn [23] as well as Grasedyck [21].

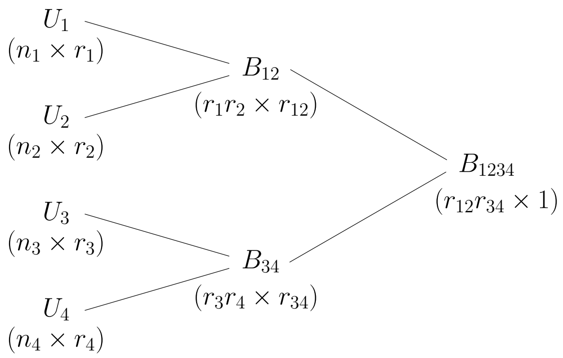

FIG. 3.1. *Example of a dimension tree for $d = 4$ dimensions.*

The HTD can be viewed as an extension of the HOSVD described above. Let $t = \{\mu_1, \ldots, \mu_q\}$ represent a set of dimensions, and $X_{(t)}$ a matricization with respect to these dimensions (e.g., $X_{12} \in \mathbb{R}^{n_1 n_2 \times n_3 n_4}$ for $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$). We define U_t to contain the r_t most significant left singular vectors of $X_{(t)}$. The matricization can be built up hierarchically, e.g., for the case $d = 4$ we obtain the hierarchical projection:

$$\begin{aligned} \operatorname{vec}(\tilde{\mathcal{X}}) &= (U_1 U_1^T \otimes U_2 U_2^T \otimes U_3 U_3^T \otimes U_4 U_4^T)(U_{12} U_{12}^T \otimes U_{34} U_{34}^T) \operatorname{vec}(\mathcal{X}) \\ &= (U_1 \otimes U_2 \otimes U_3 \otimes U_4)(B_{12} \otimes B_{34}) B_{1234} \end{aligned}$$

where $U_\mu \in \mathbb{R}^{n_\mu \times r_\mu}$, $B_{12} = (U_1^T \otimes U_2^T) U_{12} \in \mathbb{R}^{r_1 r_2 \times r_{12}}$, $B_{34} = (U_3^T \otimes U_4^T) U_{34} \in \mathbb{R}^{r_3 r_4 \times r_{34}}$ and $B_{1234} = (U_{12}^T \otimes U_{34}^T) \operatorname{vec}(\mathcal{X}) \in \mathbb{R}^{r_{12} \times r_{34}}$. More generally, a binary *dimension tree* $\mathcal{T}$ is constructed, where each node t represents a set of dimensions, which are split up among its child nodes t_1 and t_2. Each leaf node represents a single dimension. Any tensor is then represented by the matrices U_μ in each dimension, the *transfer matrices* B_t for each node (which can be reinterpreted as 3-tensors), and the last vector $B_{1 \ldots d}$, see also Figure 3.1.

A natural extension of the concept of multilinear ranks, the *hierarchical ranks* r_t are defined as $r_t = \operatorname{rank} X_{(t)}$. The storage requirements for a HTD are bounded by $dnr + (d-1)r^3$, where r is an upper bound on all ranks r_t. The singular value tree is a good way to visualize the general structure and approximability of a tensor in HTD. The error made when choosing a smaller rank r_t can be directly read from the singular values at node t. In Figure 3.3, for example, node $(2, 3)$ has the slowest singular value decay, followed by node 1. Node 6, on the other hand, has a very steep singular value decay; a small rank r_6 is enough, while a larger rank r_{23} is needed.

For a tensor in HTD, there is a recursive algorithm for computing the singular values as well as the left singular vectors for each node [21]. This allows for the efficient low-rank truncation of a tensor in HTD with a computational complexity of $O(dr^4 + dnr^2)$. As for the HOSVD, the obtained approximation is not optimal but satisfies the bound

$$\|\mathcal{X} - \tilde{\mathcal{X}}\|_2 \leq \sqrt{2d-2} \|\mathcal{X} - \mathcal{X}_r^{\mathsf{best}}\|_2,$$

where $\mathcal{X}_r^{\mathsf{best}}$ is the best possible approximation with hierarchical ranks $r = \{r_t\}_{t \in \mathcal{T}}$. Truncating a tensor in HTD involves calculating the eigenvalue decompositions of the Gramian matrices $V_t^T V_t$, where $X_{(t)} = U_t V_t^T$, as well as making the matrices U_t columnwise orthonormal by calculating the QR decompositions of the matrices U_μ, B_t in each node. If all ranks are constants, the matrices U_μ, B_t have size $r \times r$ and $r^2 \times r$, respectively. In our experiments, we have observed that QR decompositions represent the most expensive operation in all our algorithms, which requires the use of fairly small hierarchical ranks r_t.

Structured matrix-tensor multiplication is similarly efficient as for the Tucker decomposition: $(A_1 \otimes \cdots \otimes A_d)(U_1 \otimes \cdots \otimes U_d) \operatorname{vec}(\mathcal{C}) = (A_1 U_1 \otimes \cdots \otimes A_d U_d) \operatorname{vec}(\mathcal{C})$ where $\mathcal{C}$ represents the core tensor part of HTD, $(B_{12} \otimes \cdots \otimes B_{d-1d}) \cdots B_{1 \ldots d}$. Note that the core tensor is not directly affected by the multiplication, but enforcing orthogonality in $A_1 U_1, A_2 U_2, \ldots$ requires QR decompositions

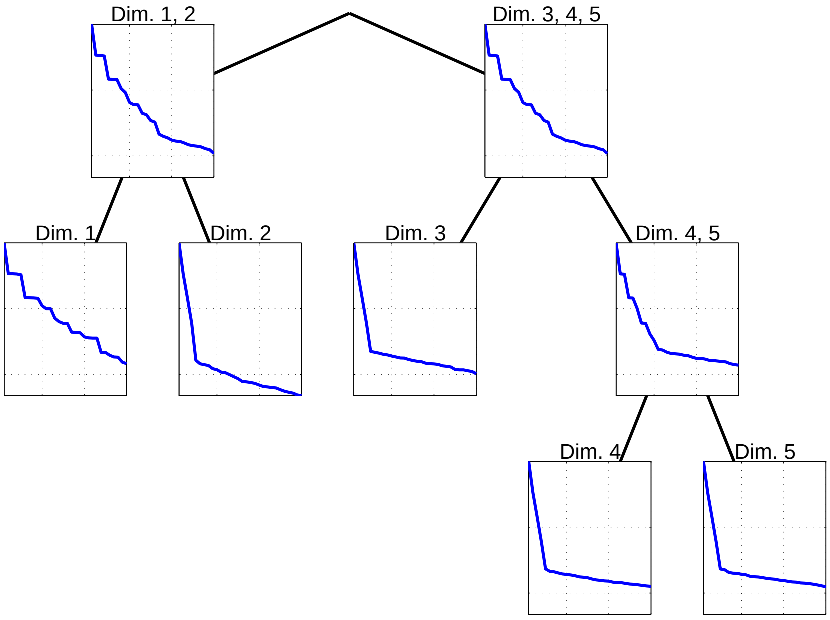

Fig. 3.2. *Singular value tree for the approximate solution tensor of the elliptic parametrized PDE from Section 4 with 4 parameters (2×2 discs). Note that the displayed singular values are affected by roundoff and truncation error; in particular the kinks are artifacts that would not be present in the exact solution tensor.*

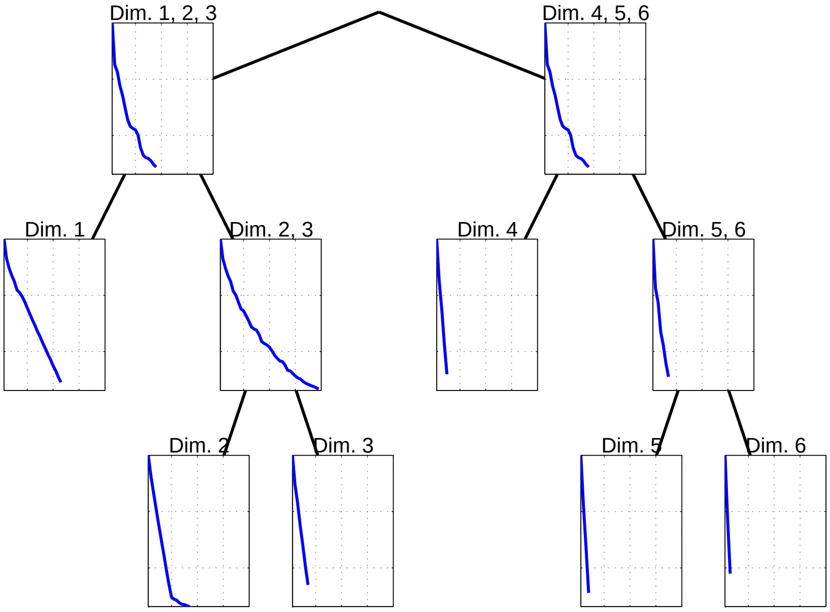

Fig. 3.3. *Singular value tree for the approximate solution tensor of the stochastic elliptic PDE from Section 5 with 5 parameters (corresponding to 5 terms in the truncated Karhunen-Loève expansion) and Karhunen-Loève eigenvalues $\sqrt{\lambda_\mu} = 5\exp(-2\mu)$.*

of these matrices and the propagation of the corresponding factors R into the components of the core tensor.

Addition of two HTD tensors is possible without any arithmetic operations, simply by appropriately concatenating the components of both tensors. Note, however, that the size of B_t increases significantly after addition:

$$B_t \in \mathbb{R}^{r_1 r_2 \times r_{12}},\, \tilde{B}_t \in \mathbb{R}^{\tilde{r}_1 \tilde{r}_2 \times \tilde{r}_{12}} \;\Rightarrow\; B_t^{\mathsf{sum}} \in \mathbb{R}^{(r_1+\tilde{r}_1)(r_2+\tilde{r}_2)\times(r_{12}+\tilde{r}_{12})}.$$

For example, when both summands have identical ranks, this increases the storage requirements by a factor of 8. Consequently, addition is only practical combined with frequent truncation to lower rank.

Remark 3.7. *Apart from the hierarchical Tucker decomposition, the Tensor Train (TT) decomposition [37], as well as the related Tensor Chain and Quantics TT decompositions [25] have been proposed in the literature. In theory, the TT decomposition can be interpreted as a special case of the HTD, where the dimension tree is degenerate.*

In the quantum mechanics community, the more general notion of tensor networks has been proposed for the solution of two-dimensional quantum systems, in an extension of the density-matrix renormalization group (DMRG) method for one-dimensional quantum systems, see, e.g., [40].

3.3. Combination of hierarchical tensor decompositions with iterative algorithms. In summary, the HTD fulfills the requirements listed in the beginning of this section. We can now combine the HTD with iterative algorithms in the same way as described in Section 2 for the two-dimensional case. This gives rise to low-rank tensor variants of the Richardson, CG, and BiCGstab methods.

Note that a low-rank tensor variant of the Richardson method has already been described in [26], on the basis of the CP decomposition. A conceptually different approach has been described by Ballani and Grasedyck [2], where the low-rank HTD structure is directly incorporated into the search space of GMRES.

4. Application to parametrized elliptic PDEs. We extend the elliptic one-parameter PDE from Section 2.3.1 to multiple parameters. Again, we consider the stationary heat equation on a square domain Ω. However, instead of only one disc the square now contains p mutually disjoint discs, see Figure 4.1. The heat conductivity coefficient is piecewise constant, assuming a parameter α_μ on each of the discs:

$$\begin{aligned} -\nabla(\sigma(x)\nabla u) &= f \qquad \text{in } \Omega = [0,L]^2 \\ u &= 0 \qquad \text{on } \Gamma := \partial\Omega, \end{aligned} \tag{4.1}$$

with

$$\sigma(x) = \begin{cases} 1+\alpha_\mu & \text{for } x \in \mathcal{D}_\mu,\ \mu = 1,\dots,p, \\ \quad 1 & \text{for } x \notin \bigcup_{\mu=1}^{p} \mathcal{D}_\mu. \end{cases}$$

As before, this PDE is discretized by finite elements with piecewise linear basis functions, resulting in a linear system of the form

$$(A_0 + \sum_{\mu=1}^{p} \alpha_\mu A_\mu)x(\alpha) = b, \tag{4.2}$$

where each of the matrices $A_1, \dots, A_p$ contains contributions from the corresponding disc. For our tests, we used $p = 4$ and $p = 9$ resulting in the system sizes $n = 1580$ and $n = 3644$, respectively, see also Figure 4.1. The right-hand side b is obtained from discretizing the source term $f \equiv 1$. In all experiments, the matrix $I \otimes \cdots \otimes I \otimes A_0$ is chosen as the preconditioner.

For the discretization of the parameters, we choose $\{\alpha_1^{(\mu)}, \dots, \alpha_m^{(\mu)}\} = \{0, 1, \dots, 100\}$, and hence $m_\mu = 101$ for $\mu = 1, \dots, p$. The number of entries in the tensor $\mathcal{X}$, containing the solutions for all parameter samples, is therefore $1580 \times 101^4 = 1.64 \times 10^{11}$ for $p = 4$ and $3644 \times 101^9 = 3.98 \times 10^{21}$ for $p = 9$.

Compared to the one-parameter case, the low-rank truncation of the iterates is a more complicated matter. During the HTD low-rank compression it would be preferable to truncate only singular values that are negligible in the sense of an absolute or relative accuracy, as discussed in Section 2.2.1. However, as the storage cost increases cubically with the hierarchical ranks, it may be necessary to also impose a maximal hierarchical rank. In all examples, we used a relative accuracy of 10^{-10} and maximal hierarchical ranks of 10, 30 or 50.

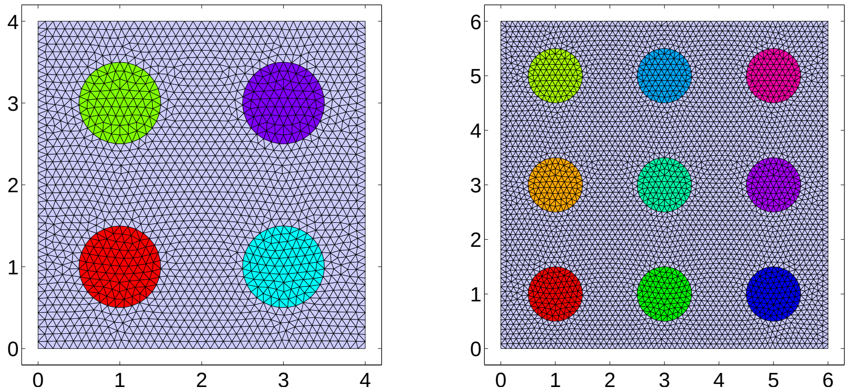

FIG. 4.1. *Left: Mesh for* 2×2 *discs. Right: Mesh for* 3×3 *discs.*

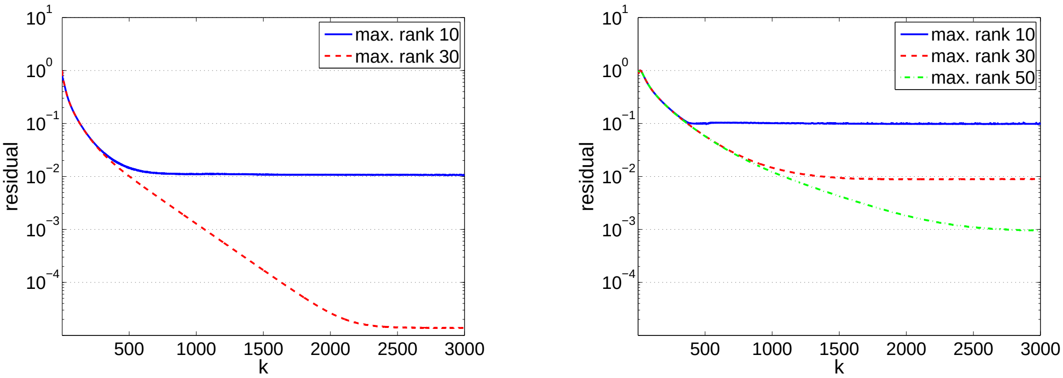

FIG. 4.2. *Left: Richardson method for* 2×2 *discs. Right: Richardson method for* 3×3 *discs.*

Figure 4.2 displays the convergence of the residual norm $\|b - \mathcal{A}x\|/\|b\|$ for the preconditioned Richardson method with a heuristic choice of the parameter ω. As in the one-parameter case, the convergence is monotone and slow. While the example with $p = 4$ parameters eventually settles at a residual norm of 10^{-4} when using a maximal hierarchical rank of 30, the case $p = 9$ parameters proves more difficult. Even when using a maximal hierarchical rank of 50, the final accuracy is only about 10^{-3}.

Figure 4.3 displays the convergence for the preconditioned CG method. The attained accuracy is at the same level as for the Richardson method but the convergence is – as expected – much faster. In contrast to the one-parameter case the convergence is not monotone, which is likely due to the maximal hierarchical rank truncation. The singular value tree for the 2×2 case, with maximal hierarchical rank 30, is shown in Figure 3.2.

5. Application to stochastic elliptic PDEs. Consider an elliptic PDE with stochastic coefficients:

$$\begin{aligned} -\nabla(a(x,y)\nabla u(x,y)) &= f(x) && \text{in } \Omega \times \Gamma, \\ u(x,y) &= 0 && \text{on } \partial\Omega \times \Gamma, \end{aligned} \tag{5.1}$$

where $y \in \Gamma$ is a random variable.

In the following, we give a brief description of how (5.1) can be turned into a deterministic parametrized PDE and refer to, e.g., [44] for more details. Representing the random variable y by an infinite number of parameters $\alpha \in [-1, 1]^\infty$, we employ the Karhunen-Loève expansion of $a(x, \alpha)$:

$$a(x,\alpha) = a_0(x) + \sum_{\mu=1}^{\infty} \sqrt{\lambda_\mu} a_\mu(x)\alpha_\mu, \tag{5.2}$$

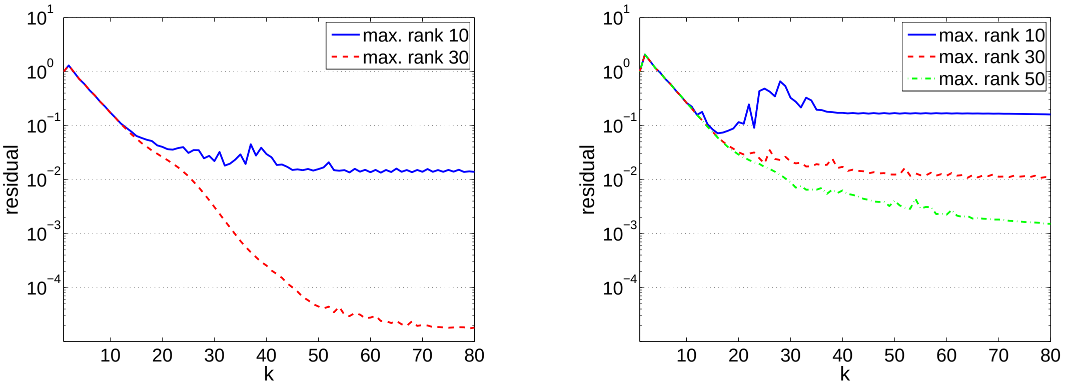

FIG. 4.3. *Left: CG method for 2×2 discs. Right: CG method for 3×3 discs.*

where $a_\mu(x), \mu \in \mathbb{N}$ are normalized $L^2(\Omega)$-functions and the coefficients $\lambda_\mu \geq 0$ are monotonically decreasing. Truncating the Karhunen-Loève expansion after p terms then results in a linearly parameter-dependent PDE, essentially of the form (4.1).

Again, a piecewise linear finite element discretization is used to yield a parametrized linear system (4.2). As above, the parameters α_μ can be discretized by sampling on a tensor grid. Alternatively, one could also use a Galerkin approach based on Legendre polynomials to approximate each α_μ, see, e.g., [6].

Remark 5.1. *Recall that the linear system arising from gathering all sampled linear systems into a large block diagonal matrix takes the form*

$$\mathcal{A} = I \otimes I \otimes \cdots \otimes A_0 + I \otimes \cdots \otimes D_1 \otimes A_1 + \cdots + D_p \otimes I \otimes \cdots \otimes I \otimes A_p, \tag{5.3}$$

where D_μ contain the parameter samples. The simplest nontrivial preconditioner uses the mean value of the random variable, $\mathcal{M}_{\mathsf{mean}} = I \otimes \cdots \otimes I \otimes A_0$.

A more intricate preconditioner, which also takes the parameter samples into account, has recently been proposed in [26, Proposition 2.6]. For this purpose, consider the following approximation of the Karhunen-Loève expansion (5.2):

$$a(x, \alpha) \approx \bar{a}_0 + \sum_{\mu=1}^{\infty} \sqrt{\lambda_\mu} \bar{a}_\mu \alpha_\mu,$$

where $\bar{a}_\mu = \int_\Omega a_\mu(x) dx$ is the mean value of $a_\mu(x)$. The finite element discretization applied to this approximation leads to

$$(A_\mu)_{ij} = \int_\Omega a_\mu(x) \nabla b_i(x) \nabla b_j(x) dx \approx \bar{a}_\mu \int_\Omega \nabla b_i(x) \nabla b_j(x) dx =: \bar{a}_\mu (L)_{ij},$$

where L corresponds to the discretized Laplacian. This yields the preconditioner $\hat{\mathcal{M}} \cdot \big(I \otimes \cdots \otimes I \otimes L\big)$ with

$$\hat{\mathcal{M}} = \big(I \otimes \cdots \otimes I \otimes \bar{a}_0 D_0 + I \otimes \cdots \otimes I \otimes \bar{a}_1 D_1 \otimes I + \cdots + \bar{a}_p D_p \otimes I \otimes \cdots \otimes I\big),$$

where we formally set $D_0 = I$. The inverse of $\hat{\mathcal{M}}$ can be approximated by an exponential sum [20],

$$\hat{\mathcal{M}}^{-1} = \sum_{k=-\infty}^{\infty} c_k \bigotimes_{\mu=0}^{p} \exp(-t_k \bar{a}_\mu D_\mu) \approx \sum_{k=-K}^{K} c_k \bigotimes_{\mu=0}^{p} \exp(-t_k \bar{a}_\mu D_\mu) =: \hat{\mathcal{M}}_K^{-1}.$$

Depending on the choice of the parameters c_k, t_k, see [8, 20], the approximation error $\hat{\mathcal{M}}_K^{-1} - \hat{\mathcal{M}}^{-1}$ decays exponentially with $\sqrt{K}$ or K. Eventually, we obtain the preconditioner

$$\mathcal{M}_{\mathsf{para}} := \hat{\mathcal{M}}_K \cdot \big(I \otimes \cdots \otimes I \otimes L\big).$$

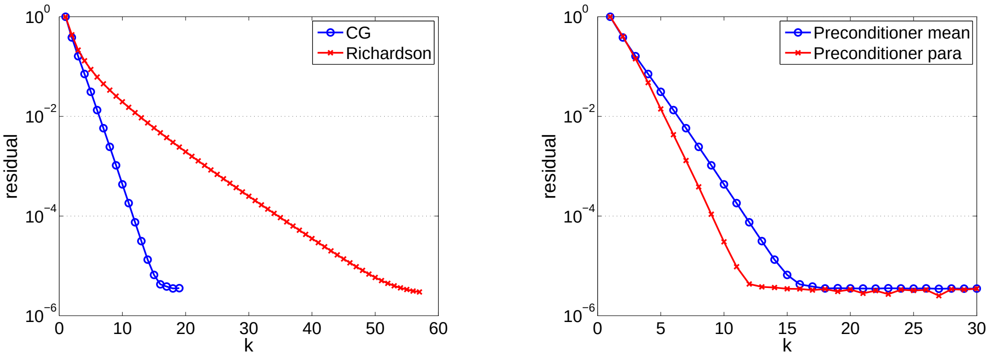

FIG. 5.1. *Left: Richardson and CG methods with preconditioner $\mathcal{M}_{\text{mean}}$ for $p = 20$, maximal hierarchical rank 20, with $\sqrt{\lambda_\mu} = 5\exp(-2\mu)$. Right: CG method using the preconditioners $\mathcal{M}_{\text{mean}}$ and $\mathcal{M}_{\text{para}}$ with $K = 5$.*

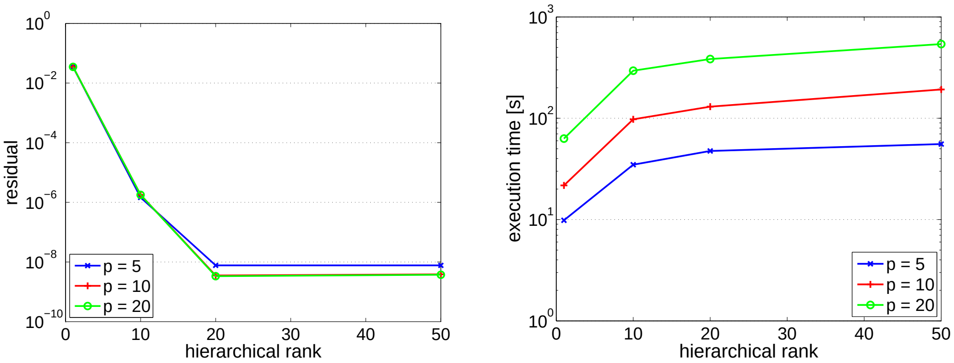

FIG. 5.2. *Rank-dependence of the preconditioned CG method for $\sqrt{\lambda_\mu} = 0.5\exp(-2\mu)$ and different values of p. Left: Final accuracy. Right: Execution time.*

Note that multiplication with $\hat{\mathcal{M}}_K^{-1}$ only requires multiplication with diagonal matrices and summation. As explained in Section 3.2.2, summation of low-rank HTD tensors becomes quickly expensive and need to accompanied by repetitive low-rank truncations. The computational effort for applying $\mathcal{M}_{\text{para}}^{-1}$ can therefore be expected to be significantly higher than for applying $\mathcal{M}_{\text{mean}}^{-1}$. It depends on the application whether this additional effort is compensated by convergence gains.

In our examples, we use the synthetic Karhunen-Loève eigenfunctions

$$a_0(x) = 1, \quad a_\mu(x) = \sin(\mu x), \quad x \in [0, \pi].$$

in the Karhunen-Loève expansion (5.2). The parameters α_μ are sampled at 50 equidistant points in $[-1, 1]$. The source term is is $f(x) = \sin(x)$. Note that this example was chosen to match the numerical example in [26, Section 4.3].

In our first test, we choose the Karhunen-Loève eigenvalues $\sqrt{\lambda_\mu} = 5\exp(-2\mu)$. Figure 5.1 displays the obtained convergence of the low-rank tensor preconditioned Richardson and CG methods. The singular value tree for 5 parameters and maximal hierarchical rank 50 is shown in Figure 3.3. Since the variation of the parameter values is quite narrow, especially compared with the example from Section 4, the preconditioner $\mathcal{M}_{\text{mean}}$ is very effective. This is reflected in two ways in Figure 5.1: (i) the Richardson and CG methods both converge quite quickly, (ii) the preconditioner $\mathcal{M}_{\text{para}}$ described in Remark 5.1 only leads to moderate improvements.

Figures 5.2 and 5.3 display the dependence of the eventually attained accuracy of the solution on the choice of the maximal hierarchical rank, when using the Karhunen-Loève eigenvalues $\sqrt{\lambda_\mu} = 0.5\exp(-2\mu)$ and $\sqrt{\lambda_\mu} = (1+\mu)^{-2}$, respectively. In both cases, the residual norm decreases rapidly as the maximal hierarchical rank increases. In the first case, increasing the number of parameters from 5 to 10 or 20 has little effect on the attained accuracy, as the coefficients λ_μ are nearly negligible for $\mu \geq 6$. In the second case, with polynomially decaying $\sqrt{\lambda_\mu}$, increasing the number of parameters has a negative impact on the accuracy when keeping the hierarchical rank fixed.

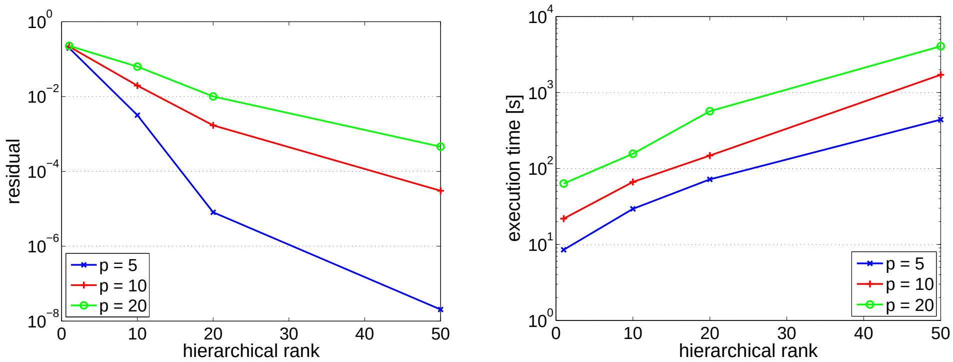

FIG. 5.3. *Rank-dependence of the preconditioned CG method for* $\sqrt{\lambda_\mu} = (1+\mu)^{-2}$ *and different values of* p. *Left: Final accuracy. Right: Execution time.*

Figures 5.2 and 5.3 also display execution times of the algorithms. As explained in Section 3.2.2, the computational effort for low-rank truncations grows proportionally with pr^4, where r denotes the hierarchical rank. This growth is clearly reflected in Figure 5.3.

Remark 5.2. *In applications, one is typically interested in computing statistics for the solution of the stochastic PDE (5.1). The sample mean value of the discretized solutions* $x(\alpha)$ *can be easily retrieved from the solution tensor* $\mathcal{X}$*:*

$$\bar{x} = \frac{1}{m_1 m_2 \cdots m_p} \Big(\big[1, \dots, 1\big] \otimes \cdots \otimes \big[1, \dots, 1\big] \otimes I \Big) \operatorname{vec}(\mathcal{X}).$$

For a maximal hierarchical rank r*, this can be evaluated within* $O(pnr + pr^3)$ *operations. The vector containing the sample variance for each entry of* x *is given by*

$$\operatorname{Var}(x) = \frac{1}{N} \operatorname{diag}(YY^T), \tag{5.4}$$

where $Y = X_{(0)} - \bar{x}\big[1, \dots, 1\big]$ *and* $N = m_1 m_2 \cdots m_p$ *or* $N = m_1 m_2 \cdots m_p - 1$*. It can be shown that (5.4) can be calculated recursively, requiring* $O(\ell r^4 + nr)$ *operations in total, where* ℓ *represents the depth of the dimension tree [32].*

6. Application to parametrized convection-diffusion equation.

As a final, non-elliptic example, we consider the stationary convection-diffusion equation on the domain introduced in Section 4:

$$\begin{aligned} -\nabla(\sigma(x)\nabla u) + c^T \nabla u = f \quad & \text{in } \Omega = [0, L]^2 \\ u = 0 \quad & \text{on } \Gamma := \partial\Omega, \end{aligned}$$

with

$$\sigma(x) = \begin{cases} 1 + \alpha_\mu & \text{for } x \in \mathcal{D}_\mu, \\ 1 & \text{for } x \notin \bigcup_{\mu=1}^{p} \mathcal{D}_\mu. \end{cases}$$

The finite element discretization for the domain with 2 discs ($p = 4$), see Figure 4.1, once again results in a linear system of the form

$$\Big(A_0 + \sum_{\mu=1}^{p} \alpha_\mu A_\mu \Big) x(\alpha) = b,$$

with system size $n = 1580$. As the convection term is not parameter-dependent, it only affects the matrix A_0. The source term is $f(x) = 1$. The parameters samples are $\{\alpha_1^{(\mu)}, \dots, \alpha_m^{(\mu)}\} = \{0, 0.1, \dots, 10\}$, hence $m_\mu = 101$ for $\mu = 1, \dots, p$. Consequently, the number of entries in the tensor $\mathcal{X}$ is $1580 \times 101^4 = 1.64 \times 10^{11}$.

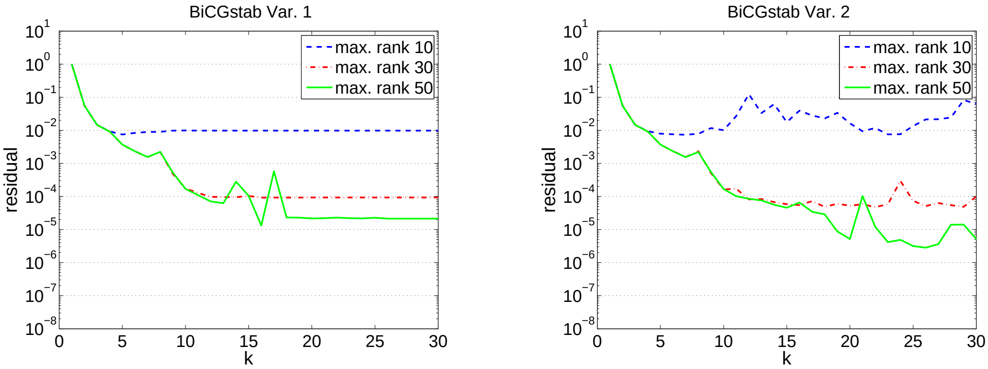

FIG. 6.1. *Convergence behavior of preconditioned BiCGstab. Left: Variant 1, Right: Variant 2.*

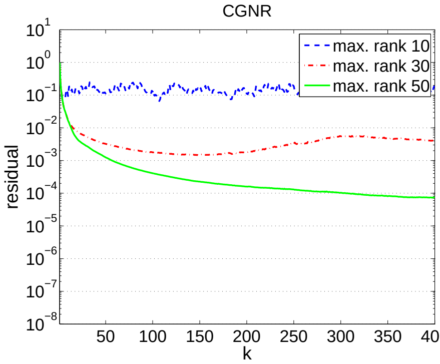

FIG. 6.2. *Convergence behavior of preconditioned CG applied to the normal equations.*

Figure 6.2 displays the convergence of the preconditioned CG method applied to the normal equations $\mathcal{A}^T\mathcal{A}x = \mathcal{A}^T b$ (i.e., CGNR) with different choices of the maximal ranks. The preconditioner $I \otimes \cdots \otimes I \otimes A_0^T A_0$ is used. Figure 6.1 displays the convergence of the two variants of the preconditioned BiCGstab method described in Algorithm 3, with preconditioner $I \otimes \cdots \otimes I \otimes A_0$. As expected, both BiCGstab variants converge much faster than CGNR. Additionally, CGNR stagnates at a higher residual norm than the best accuracy attained by both BiCGstab variants. As in the one-parameter case, the convergence behavior of Variant 2 becomes erratic when the final accuracy is attained. Even worse, the residual norm appears to increase again when the iteration is continued beyond this point. Thus, stopping the iteration becomes a subtle issue, which may render this variant impractical.

7. Comparison with other methods. In this section, we present a first numerical comparison of our approach with existing methods.

Restarted GMRES. As mentioned in Section 3.3, Ballani and Grasedyck [2] have developed variants of the GMRES method originally intended for high-dimensional linear systems but also suitable for parameter-dependent linear systems. Our experiments include GMRES without and with restarts. In the latter case, a restart is performed every 5 iterations. The convergence behavior of GMRES for the non-symmetric example from Section 6 is shown in Figure 7.1, with respect to iteration number and execution time. For this example, BiCGstab appears to be less robust (and requires monitoring of the residual), but it turns out to be faster in terms of execution time for our MATLAB implementation. For symmetric problems, the CG method is robust and fast, and therefore the method of choice (see Figure 7.2, left).

Sparse Grid interpolation. As an alternative approach to the approximate solution of a parametrized linear system $A(\alpha)x(\alpha) = b(\alpha)$, sparse grid interpolation can be applied to the vector function $x(\alpha) = A(\alpha)^{-1}b(\alpha)$. This requires the solution of the corresponding linear system at each interpolation point. For the experiments, we have used the Sparse Grid Interpolation Toolbox [29, 28]

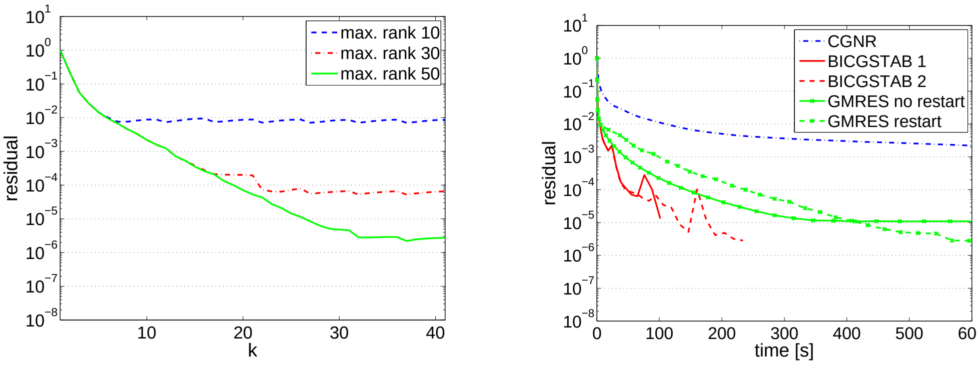

FIG. 7.1. *Left: Convergence behavior of preconditioned GMRES. Restart every* 5 *steps. Right: Comparison of times of the different algorithms, with maximum rank* 50*. GMRES restart every* 5 *steps*

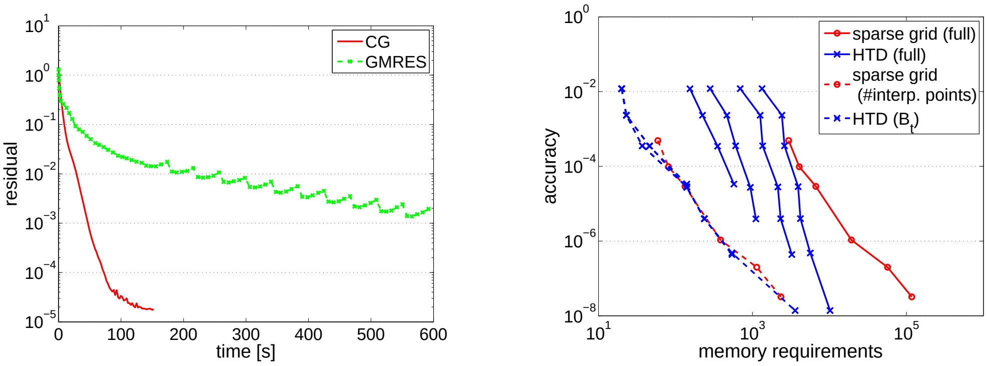

FIG. 7.2. *Left: Comparison of times for CG and GMRES for the symmetric problem, with maximum rank* 50*, restart every* 5 *steps. Right: Comparison with a sparse grid interpolation method.*

to adaptively generate the interpolation points for a piecewise linear interpolation. In particular, this implies that the sampling strategy for the parameters is chosen by the algorithm, while our approach requires an a priori choice.

As both approaches are very different, it is beyond the scope of the paper to provide a comprehensive analytical and numerical comparison. The following experiment should be considered as a first step in this direction. We apply both approaches to the stochastic PDE from Section 5 with exponential decay and $p = 20$ terms in the truncated Karhunen-Loève expansion. Figure 7.2, right plot, compares memory requirements (in terms of #floating point numbers) with the achieved accuracy, i.e., the estimated worst-case error $\|\tilde{u}(\alpha) - u(\alpha)\|_{L^2(\Omega)}/\|u(\alpha)\|_{L^2(\Omega)}$ across all $\alpha \in [-1,1]^p$. This is determined heuristically, by choosing 100 random points in the parameter range. We have applied HTD to equidistant samplings of the parameter domain, with the finest sampling corresponding to the highest level used in sparse grid interpolation. For HTD, solid lines correspond to memory requirements for the entire HTD, while dashed lines account for the transfer matrices B_t only (without the leaf matrices U_t, which are more directly affected by the sampling strategy). For sparse grids, solid lines correspond to the number of interpolation points multiplied with n, while dashed lines represent the number of interpolation points only. For this example, it can be observed that the storage requirements of HTD are significantly smaller, but it is striking that the general tendency is rather similar in both cases. However, it should be emphasized that the computational time of the sparse grid approach is much smaller. This is due to the fact that a linear system of size $n = 50$ is solved at each interpolation point, which is extremely fast. HTD involves significant overhead in the low-rank operations, which dominates the entire calculation but can be expected to become negligible for much larger n. A careful investigation of this aspect is subject to future work.

8. Conclusions. Solving linear systems depending on many parameters is a computationally demanding task. In this paper, we have shown – theoretically as well as numerically – that combining standard iterative methods with low-rank tensor decompositions allows to handle parametrized linear system that are computationally inaccessible to standard methods. For an example described in Section 4, the solution of about 10^{18} linear systems of order 3644 with accuracy 10^{-3} requires 61 minutes with our low-rank tensor variant of the preconditioned CG method. In comparison, with a standard solver that requires 10 milliseconds for each linear system, the overall solution time would be 3×10^8 CPU years!

Several aspects of the paper merit further investigation. On the theoretical side, it is not clear whether the approximation bound by Theorem 3.6 could be improved to yield a truly exponential error decay. Also, the result of the theorem is tailored to the CP decomposition, possibly resulting in rather loose upper bounds for the hierarchical Tucker decomposition used in this paper. It may be possible to obtain better bounds by considering approximation problems more natural for the latter decomposition. This would also provide more insight into the optimal order in the dimension tree. On the algorithmic side, further investigation is required to understand which variants of Krylov subspace methods are robust to low-rank truncations, particularly in the nonsymmetric case. Our numerical examples only cover linear parameter-dependence, for which the Kronecker structure in the matrix $\mathcal{A}$ is particularly evident. To address nonlinear dependencies, transformation techniques and polynomial expansion combined with (exact) linearization can be used.

9. Acknowledgements. The authors thank the referees, Roman Andreev and Michael L. Overton for helpful remarks and constructive criticism.

REFERENCES

[1] E. Acar, D. Dunlavy, and T. Kolda. A scalable optimization approach for fitting canonical tensor decompositions. *J. Chemometrics (to appear)*, 2010.

[2] J. Ballani and L. Grasedyck. A projection method to solve linear systems in tensor format. Preprint 46, DFG-Schwerpunktprogramm 1324, May 2010.

[3] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *C. R. Math. Acad. Sci. Paris*, 339(9):667–672, 2004.

[4] R. Barrett, M. Berry, T. F. Chan, J. W. Demmel, J. Donato, J. J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, PA, 1994.

[5] U. Baur, P. Benner, A. Greiner, J. G. Korvink, J. Lienemann, and C. Moosmann. Parameter preserving model order reduction for MEMS applications. 2010. To appear in Mathematical and Computer Modelling of Dynamical Systems.

[6] M. Bieri, R. Andreev, and C. Schwab. Sparse tensor discretization of elliptic SPDEs. *SIAM J. Sci. Comput.*, 31(6):4281–4304, 2009.

[7] M. Bieri, R. Andreev, and C. Schwab. Sparse tensor discretization of elliptic SPDEs. Report 2009-07, Seminar for Applied Mathematics, ETH Zurich, 2009.

[8] D. Braess and W. Hackbusch. Approximation of $1/x$ by exponential sums in $[1, \infty)$. *IMA J. Numer. Anal.*, 25(4):685–697, 2005.

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

[10] T. F. Chan and M. K. Ng. Galerkin projection methods for solving multiple linear systems. *SIAM J. Sci. Comput.*, 21(3):836–850, 1999.

[11] A. Cohen, R. DeVore, and C. Schwab. Analytic regularity and polynomial approximation of parametric and stochastic elliptic PDEs. Report 2010-03, Seminar for Applied Mathematics, ETH Zurich, 2010.

[12] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

[13] V. de Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 20(3):1084—1127, 2008.

[14] M. Espig and W. Hackbusch. A regularized Newton method for the efficient approximation of tensors represented in the canonical tensor format. Preprint 78/2010, Max-Planck-Institut für Mathematik in den Naturwissenschaften, 2010.

[15] K. V. Fernando and S. Hammarling. A product induced singular value decomposition (πSVD) for two matrices and balanced realization. In *Linear algebra in signals, systems, and control (Boston, MA, 1986)*, pages 128–140. SIAM, Philadelphia, PA, 1988.

[16] R. W. Freund. Solution of shifted linear systems by quasi-minimal residual iterations. In *Numerical linear algebra (Kent, OH, 1992)*, pages 101–121. de Gruyter, Berlin, 1993.

[17] A. Frommer and U. Glässner. Restarted GMRES for shifted linear systems. *SIAM J. Sci. Comput.*, 19(1):15–26, 1998.

[18] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[19] L. Grammont, N. Higham, and F. Tisseur. A framework for analyzing nonlinear eigenproblems and parametrized linear systems. MIMS EPrint 2009.51, 2009. To appear in Linear Algebra Appl.

[20] L. Grasedyck. Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure. *Computing*, 72(3-4):247–265, 2004.

[21] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2010.

[22] G.-D. Gu and V. Simoncini. Numerical solution of parameter-dependent linear systems. *Numer. Linear Algebra Appl.*, 12(9):923–940, 2005.

[23] W. Hackbusch and S. Kühn. A new scheme for the tensor representation. *J. Fourier Anal. Appl.*, 15(5):706–722, 2009.

[24] K. Jbilou, A. Messaoudi, and H. Sadok. Global FOM and GMRES algorithms for matrix equations. *Appl. Numer. Math.*, 31(1):49–63, 1999.

[25] B. N. Khoromskij and I. V. Oseledets. Quantics-TT approximation of elliptic solution operators in higher dimensions. Preprint 79/2009, Max-Planck-Institut für Mathematik in den Naturwissenschaften, 2009.

[26] B. N. Khoromskij and C. Schwab. Tensor-structured galerkin approximation of parametric and stochastic elliptic PDEs. *SIAM J. Sci. Comput.*, 33(1):364–385, 2011.

[27] M. E. Kilmer and E. de Sturler. Recycling subspace information for diffuse optical tomography. *SIAM J. Sci. Comput.*, 27(6):2140–2166, 2006.

[28] A. Klimke. Sparse Grid Interpolation Toolbox – user's guide. Technical Report IANS report 2007/017, University of Stuttgart, 2007. See `http://www.ians.uni-stuttgart.de/spinterp/`.

[29] A. Klimke and B. Wohlmuth. Algorithm 847: spinterp: piecewise multilinear hierarchical sparse grid interpolation in MATLAB. *ACM Trans. Math. Software*, 31(4):561–579, 2005.

[30] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[31] D. Kressner and C. Tobler. Krylov subspace methods for linear systems with tensor product structure. *SIAM J. Matrix Anal. Appl.*, 31(4):1688–1714, 2010.

[32] D. Kressner and C. Tobler. `htucker` – a MATLAB toolbox for tensors in hierarchical Tucker format, 2011.

[33] G. G. Lorentz. *Approximation of functions.* Chelsea Publishing Co., New York, second edition, 1986.

[34] J. C. Miellou, P. Cortey-Dumont, and M. Boulbrachene. Perturbation of fixed point iterative methods. *Advances in Parallel Computing*, I:81–122, 1990.

[35] F. Nobile, R. Tempone, and C. G. Webster. An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numer. Anal.*, 46(5):2411–2442, 2008.

[36] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM J. Numer. Anal.*, 46(5):2309–2345, 2008.

[37] I.V. Oseledets. Compact matrix form of the d-dimensional tensor decomposition. Preprint 09-01, Institute of Numerical Mathematics RAS, Moscow, Russia, 2009.

[38] M. L. Parks, E. de Sturler, G. Mackey, D. D. Johnson, and S. Maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM J. Sci. Comput.*, 28(5):1651–1674, 2006.

[39] Y. Saad. *Iterative Methods for Sparse Linear Systems, 2nd edition.* SIAM, Philadelpha, PA, 2003.

[40] U. Schollwöck. The density-matrix renormalization group. *Rev. Mod. Phys.*, 77(1):259–315, Apr 2005.

[41] V. Simoncini. Extended Krylov subspace for parameter dependent systems. *Appl. Numer. Math.*, 60(5):550–560, 2010.

[42] V. Simoncini and F. Perotti. On the numerical solution of $(\lambda^2 A + \lambda B + C)x = b$ and application to structural dynamics. *SIAM J. Sci. Comput.*, 23(6):1875–1897, 2002.

[43] P. Spiteri, J.-C. Miellou, and D. El Baz. Perturbation of parallel asynchronous linear iterations by floating point errors. *Electron. Trans. Numer. Anal.*, 13:38–55 (electronic), 2002.

[44] R. A. Todor and C. Schwab. Convergence rates for sparse chaos approximations of elliptic problems with stochastic coefficients. *IMA J. Numer. Anal.*, 27(2):232–261, 2007.

[45] L. N. Trefethen and M. Embree. *Spectra and Pseudospectra. The Behavior of Nonnormal Matrices and Operators.* Princeton University Press, Princeton, NJ, 2005.

[46] L. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966.

[47] A. van der Sluis and H. A. van der Vorst. The rate of convergence of conjugate gradients. *Numer. Math.*, 48(5):543–560, 1986.

[48] T. G. Wright. EigTool, 2002. See `http://www.comlab.ox.ac.uk/pseudospectra/eigtool/`.