A Projection Method to Solve Linear Systems in Tensor Format

Jonas Ballani*, and Lars Grasedyck†

Bericht Nr. 309 **April 2010**

Key words: Low rank, Tucker, hierarchical Tucker, Kronecker–product matrix.

AMS subject classifications: 15A69, 90C06, 65F10

Institut für Geometrie und Praktische Mathematik
RWTH Aachen

Templergraben 55, D–52056 Aachen (Germany)

* Max Planck Insitute for Mathematics in the Sciences, Inselstr. 22–26, D–04109 Leipzig, Germany.

† Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Templergraben 55, 52056 Aachen.
Financial support from the DFG SPP–1324 under grant GRA2179/2–1 gratefully acknowledged.

A Projection Method to Solve Linear Systems in Tensor Format

Jonas Ballani [*], Lars Grasedyck [†]

April 30, 2010

In this paper we propose a method for the numerical solution of linear systems of equations in low rank tensor format. Such systems may arise from the discretisation of PDEs in high dimensions but our method is not limited to this type of application. We present an iterative scheme which is based on the projection of the residual to a low dimensional subspace. The subspace is spanned by vectors in low rank tensor format which — similarly to Krylov subspace methods — stem from the subsequent (approximate) application of the given matrix to the residual. All calculations are performed in hierarchical Tucker format which allows for applications in high dimensions. The mode size dependency is treated by a multilevel method. We present numerical examples that include high-dimensional convection-diffusion equations and shift-invert eigenvalue solvers.

1 Introduction

This paper is concerned with the solution of linear systems

$$Ax = b \tag{1}$$

where the system matrix A is given as the sum of d Kronecker products of matrices, i.e.

$$A = \sum_{j=1}^{k_A} \bigotimes_{\mu=1}^{d} A_{j,\mu}, \tag{2}$$

and the right-hand side b is given as a tensor of low (canonical) rank. A matrix of the form (2) is said to be given in *Kronecker format*. Every solution strategy which considers A as a full and unstructured matrix will result in a computational complexity that grows exponentially in d and is therefore intractable for $d \gg 2$. If x can be approximated by a vector of low tensor rank, solution strategies which exploit the tensor structure of the linear system become applicable. We will propose a projection method with a complexity of $\mathcal{O}(dk_A)$ which can be applied even in high dimensions.

[*]Max Planck Institute for Mathematics in the Sciences, Inselstraße 22–26, D-04109 Leipzig, Germany
[†]Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Templergraben 55, D-52056 Aachen, Germany. Financial support from the DFG SPP-1324 under grant GRA2179/2-1 gratefully acknowledged.

1.1 PDEs in High Dimension

As an example where a linear system (1) with A given in Kronecker format may arise, we consider the partial differential equation

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega := (0,1)^d, \\ u &= 0 \quad \text{on } \Gamma := \partial\Omega. \end{aligned} \tag{3}$$

A finite difference discretisation of (3) leads to a linear system (1) where A is of the special form

$$A = \sum_{j=1}^{d} I_{n_1} \otimes \cdots \otimes I_{n_{j-1}} \otimes A_j \otimes I_{n_{j+1}} \otimes \cdots \otimes I_{n_d}. \tag{4}$$

Here, $A_j \in \mathbb{R}^{n_j \times n_j}$ denotes the stiffness matrix of the 1d Laplacian and I_{n_j} denotes the $n_j \times n_j$ identity matrix. A similar structure arises for finite element discretisations and for other operators of the form $\operatorname{div} \sigma \nabla + \nabla \cdot b + c$ where a, b, c are constant.

In [8] we have shown that u has an integral representation which can be approximated by vectors of low rank. The approximation is based on the computation of integrals of matrix exponentials by a quadrature rule using sums of exponentials. The method scales linearly in the dimension d and it is only applicable for definite systems of the form (4) (commutativity of the summands is required).

1.2 Related Solution Strategies

Beylkin and Mohlenkamp [3] reformulate (1) as a linear least squares problem. For a given tensor rank of x, they fix all but one direction and solve the associated normal equations. Alternating cyclically through all possible directions, the residual is decreased at each iteration step. This procedure is commonly known as an alternating least squares (ALS) approach (cf. e.g. [2], [3]).

Espig, Hackbusch, Rohwedder, and Schneider [7] have proposed a direct minimisation of the functional

$$f(x) = \|Ax - b\|^2$$

where x is assumed to be of fixed rank. The computation of the gradient of f is performed with respect to the low rank structure of x. This entails the existence of local minima that are no solutions of (1) which cannot be trivially avoided. To apply the method of [7], we therefore have to rely on additional assumptions to avoid local minima.

Kressner and Tobler [15] have introduced a so-called tensor Krylov subspace method. Using the special structure of (4), they construct a tensorised Krylov space which is the tensor product of usual Krylov subspaces. The basis of the tensorised Krylov space can be used to transform the original linear system to a system of smaller (mode) size. If the smaller system can be solved efficiently, this approach leads to acceptable convergence rates for typical model problems. As a major drawback, the computational complexity for the solution of the smaller system still scales exponentially in d.

Krylov subspace methods for tensor computations have also been analysed by Eldén and Savas [5] but not for the solution of linear systems in low rank tensor format.

1.3 Generalised Low Rank GMRES

In this article, we propose a new method for the solution of (1) which is derived from a typical projection method. In analogy to a GMRES method, we will construct a low dimensional

subspace spanned by tensors of low rank. The linear system projected onto this subspace is solved in each iteration step and the iterate is truncated to low rank. After this projection and truncation step the process is restarted by the construction of a new subspace. The properties of our new method are:

- It has a complexity of $\mathcal{O}(dk_A)$ and is therefore applicable in high dimensions
- Except the Kronecker format, no additional structure of the addends of A is needed, in particular we do not require the form (4)
- The residual is non-increasing at each iteration step
- All calculations are performed in hierarchical Tucker format and rely on simple linear algebra tools like the SVD for the truncation of tensors

The hierarchical Tucker format has been introduced by Hackbusch and Kühn [14]. It allows for an efficient representation of tensors with linear complexity in d. Moreover, a truncation procedure is available [9] which scales linearly in d and hence permits an approximate arithmetic in high dimensions. As a side-effect of our method, the capabilities of the approximate arithmetic in hierarchical Tucker format for the usage in iterative schemes can be demonstrated.

1.4 Iterate and Approximate

Hackbusch, Khoromskij and Tyrtyshnikov [13] analyse the convergence of a truncated iteration process of the form

$$x_{\ell+1} := \mathcal{T}(\Phi_\ell(x_\ell)) \tag{5}$$

where Φ_ℓ is a one-step operator and $\mathcal{T}$ is some truncation operator. They show that if (5) converges for $\mathcal{T} = \mathrm{Id}$, then the process will also converge for small perturbations of the iterates induced by a general truncation operator $\mathcal{T}$. Unfortunately, this concept will not directly apply to our method as we will work with an operator that truncates tensors to a fixed rank. Alternatively, one could choose all involved ranks adaptively to guarantee a certain precision. As some numerical experiments show, this would lead to a successive growth in the ranks which increases the computing time significantly.

The rest of this article is organised as follows. In the second section, we introduce some basic definitions related to tensor decompositions. In the third section, we will derive our new solution method for (1) from a general projection method. In order to make our setting more accessible, we will afterwards introduce the hierarchical Tucker format which allows for an efficient and accurate representation of tensors. Moreover, an approximate arithmetic is defined which relies on the truncation of tensors in hierarchical Tucker format. In section five, we will adapt the projection method in tensor format to the case of the hierarchical Tucker format. Afterwards, we will briefly discuss the acceleration of our method by multigrid methods and the application to eigenvalue problems. In the last section, we illustrate the potential of our method by some numerical examples.

2 Basic Definitions

In this section, we introduce some basic definitions related to tensors that will be used throughout the whole article.

Notation 1 (Index set)**.** Let $d \in \mathbb{N}$ and $n_1, \dots, n_d \in \mathbb{N}$. We consider tensors as vectors over product index sets. For this purpose, we introduce the d-fold product index set

$$\mathcal{I} := \mathcal{I}_1 \times \cdots \times \mathcal{I}_d, \qquad \mathcal{I}_\mu := \{1, \dots, n_\mu\}, \quad \mu \in \{1, \dots, d\}.$$

Definition 2 (Elementary tensor, order)**.** A tensor $X \in \mathbb{R}^{\mathcal{I}}$ is called an *elementary tensor*, if it can be represented as the tensor product of vectors $x_\mu \in \mathbb{R}^{n_\mu}$, $\mu \in \{1, \dots, d\}$, i.e.

$$X = \bigotimes_{\mu=1}^{d} x_\mu. \tag{6}$$

The entries of X are given by

$$X_{(i_1,\dots,i_d)} = \prod_{\mu=1}^{d} (x_\mu)_{i_\mu}, \quad i_\mu \in \mathcal{I}_\mu.$$

Tensors of the form (6) are often called *rank 1* tensors. The integer d is the *order* or *dimension* of the tensor X.

Definition 3 (Tensor rank, k-term representation)**.** The *rank* k of a tensor $X \in \mathbb{R}^{\mathcal{I}}$ is the minimal number $k \in \mathbb{N}_0$ such that there exist elementary tensors $X_1, \dots, X_k$ with

$$X = X_1 + \ldots + X_k = \sum_{j=1}^{k} \bigotimes_{\mu=1}^{d} x_{j,\mu}, \quad x_{j,\mu} \in \mathbb{R}^{n_\mu}. \tag{7}$$

A tensor of the form (7) is said to be given in a *k-term* representation. In the literature, the representation (7) is sometimes referred to as *PARAFAC* (parallel factors), *CANDECOMP* (canonical decomposition), (CP), or *Kronecker* format.

Definition 4 (Tucker rank, Tucker format)**.** The *Tucker rank* of a tensor $X \in \mathbb{R}^{\mathcal{I}}$ is the tuple $(k_1, \dots, k_d)$ with minimal entries $k_\mu \in \mathbb{N}_0$ such that there exist orthonormal vectors $u_{j,\mu} \in \mathbb{R}^{n_\mu}$ and a so-called *core tensor* $C \in \mathbb{R}^{k_1 \times \ldots \times k_d}$ with

$$X = \sum_{j_1=1}^{k_1} \cdots \sum_{j_d=1}^{k_d} C_{(j_1,\dots,j_d)} \bigotimes_{\mu=1}^{d} u_{j_\mu,\mu}, \quad \langle u_{i,\mu}, u_{j,\mu} \rangle = \delta_{i,j}. \tag{8}$$

The representation of the form (8) is called the *Tucker format*.

Definition 5 (Matrix in Kronecker format)**.** The Kronecker product for matrices $A_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$, $\mu \in \{1, \dots, d\}$, is defined by

$$\left(\bigotimes_{\mu=1}^{d} A_\mu \right)_{(i_1,\dots,i_d),(j_1,\dots,j_d)} := \prod_{\mu=1}^{d} (A_\mu)_{i_\mu,j_\mu}, \qquad i_\mu, j_\mu \in \mathcal{I}_\mu.$$

A matrix A of the form

$$A = \sum_{j=1}^{k_A} \bigotimes_{\mu=1}^{d} A_{j,\mu}, \quad A_{j,\mu} \in \mathbb{R}^{n_\mu \times n_\mu}, \tag{9}$$

is said to be given in *Kronecker format*. The multiplication of a matrix $A = \otimes_{\mu=1}^{d} A_\mu$ by a vector $x = \otimes_{\mu=1}^{d} x_\mu$ reads

$$\left(\bigotimes_{\mu=1}^{d} A_\mu \right) \left(\bigotimes_{\mu=1}^{d} x_\mu \right) = \bigotimes_{\mu=1}^{d} (A_\mu x_\mu).$$

3 Projection Methods

3.1 A General Projection Method

Consider the linear system

$$Ax = b$$

where $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ is a non-singular matrix and $b \in \mathbb{R}^{\mathcal{I}}$. Let x_0 be an initial guess and let $\mathcal{V}$, $\mathcal{W}$ be two m-dimensional subspaces of $\mathbb{R}^{\mathcal{I}}$. By means of a projection method, we try to find an approximate solution $\tilde{x} \in x_0 + \mathcal{V}$ such that the residual $b - A\tilde{x}$ is orthogonal to $\mathcal{W}$. Given the initial residual $r_0 = b - Ax_0$, the approximate solution can be defined as $\tilde{x} = x_0 + \delta$, $\delta \in \mathcal{V}$, such that

$$\langle r_0 - A\delta, w\rangle = 0 \quad \text{for all} \quad w \in \mathcal{W}.$$

Let $V_m = [v_1, \ldots, v_m] \in \mathbb{R}^{\mathcal{I}\times m}$ and $W_m = [w_1, \ldots, w_m] \in \mathbb{R}^{\mathcal{I}\times m}$ be such that the column vectors of V_m and W_m form a basis of $\mathcal{V}$ and $\mathcal{W}$, respectively. If the approximate solution is written in the form $x = x_0 + V_m y$, $y \in \mathbb{R}^m$, the orthogonality condition leads immediately to the following system of equations

$$W_m^\top A V_m y = W_m^\top r_0.$$

If the matrix $W_m^\top A V_m$ is non-singular, the approximate solution is uniquely defined. In particular, $W_m^\top A V_m$ is non-singular if A is non-singular and $\mathcal{W} = A\mathcal{V}$ (cf. [17] Prop. 5.1). If $\mathcal{W} = A\mathcal{V}$ where $\mathcal{V}$ is the m-th Krylov subspace, i.e. $\mathcal{V} = \operatorname{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^{m-1} r_0\}$, the projection method is known as GMRES. The projection algorithm is summarised in Algorithm 1. Note that we have assumed a fixed dimension m of the space $\mathcal{V}$. In a GMRES method, this parameter is usually chosen dynamically as the residual can be estimated cheaply in the inner loop when the column vectors of V_m are orthonormal. Later on, we will not be able to rely on orthonormality.

Algorithm 1 Projection method (GMRES)

choose x_0
for $\ell = 0, 1, \ldots$ **do**
  $r_\ell := b - Ax_\ell$
  if $\|r_\ell\| / \|b\| < \varepsilon$ **then**
    return x_ℓ
  end if
  $v_1 := r_\ell / \|r_\ell\|$
  for $j = 1, \ldots, m$ **do**
    $w_j := Av_j$
    solve $(V_j^\top V_j)\alpha = V_j^\top w_j$
    $\tilde{v}_{j+1} := w_j - \sum_{i=1}^{j} \alpha_i v_i$
    $v_{j+1} := \tilde{v}_{j+1} / \|\tilde{v}_{j+1}\|$
  end for
  solve $(W_m^\top A V_m) y = W_m^\top r_\ell$
  $x_{\ell+1} := x_\ell + V_m y$
end for

3.2 A Projection Method in Tensor Format

Let us assume that the matrix $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ is given in Kronecker format, i.e.

$$A = \sum_{j=1}^{k_A} \bigotimes_{\mu=1}^{d} A_{j,\mu}, \quad A_{j,\mu} \in \mathbb{R}^{n_\mu \times n_\mu},$$

and that the right-hand side possesses the representation

$$b = \sum_{j=1}^{k_b} \bigotimes_{\mu=1}^{d} b_{j,\mu}, \quad b_{j,\mu} \in \mathbb{R}^{n_\mu}.$$

Every naive strategy which aims at solving (1) will have a complexity that scales exponentially in d and will be prohibitively expensive. We therefore want to introduce a new method which is inspired by the general projection method that completely relies on calculations with tensors of low rank. This will effectively reduce the complexity to $\mathcal{O}(dk_A)$ which allows us to solve systems in high dimensions.

In analogy to the general projection method, we will generate a sequence of iterates $\{x_\ell\}_{l\in\mathbb{N}}$, $x_\ell \in \mathbb{R}^{\mathcal{I}}$, in tensor format of low rank such that the norm of the residual does not increase at each iteration step. To this end, assume that $x_\ell \in \mathbb{R}^{\mathcal{I}}$ is given in low rank format and let $r_\ell := b - Ax_\ell$. We first construct a subspace $\mathcal{V} = \operatorname{span}\{v_1, \dots, v_m\} \subset \mathbb{R}^{\mathcal{I}}$ such that all basis vectors v_j are given in low rank format. Define the first basis vector by

$$\tilde{v}_1 := \mathcal{T}(r_\ell), \qquad v_1 := \tilde{v}_1 / \left\|\tilde{v}_1\right\|,$$

where $\mathcal{T} : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}^{\mathcal{I}}$ is some truncation operator that truncates a tensor either to a fixed rank or to a given accuracy. Now let $v_1, \dots, v_j \in \mathbb{R}^{\mathcal{I}}$ be given in low rank format and let $V_j := [v_1, \dots, v_j] \in \mathbb{R}^{\mathcal{I}\times j}$. As in a Krylov subspace method, define $w_j := Av_j$ and solve $(V_j^\top V_j)\alpha = V_j^\top w_j$. A new basis vector is now defined by

$$\tilde{v}_{j+1} := \mathcal{T}\left(w_j - \sum_{i=1}^{j} \alpha_i v_i\right), \qquad v_{j+1} := \tilde{v}_{j+1} / \left\|\tilde{v}_{j+1}\right\|.$$

Note that, due to the truncation, the columns of V_m are no longer orthogonal. Moreover, the subspace $\mathcal{V} = \operatorname{span}\{v_1, \dots, v_m\}$ is not a Krylov subspace. Nonetheless, the affine space $x_\ell + \mathcal{V}$ will contain an element which does not increase the norm of the residual. Let $z_{\ell+1} := x_\ell + V_m y$, $y \in \mathbb{R}^m$. To perform the projection of the new residual $b - Az_{\ell+1}$ onto $\mathcal{W} = \operatorname{span}\{w_1, \dots, w_m\}$, let $W_m := [w_1, \dots, w_m] \in \mathbb{R}^{\mathcal{I}\times m}$ and solve $(W_m^\top A V_m)y = W_m^\top r_\ell$. The new iterate is now defined by

$$z_{\ell+1} = x_\ell + V_m y, \qquad x_{\ell+1} := \mathcal{T}(z_{\ell+1}).$$

For the exact residual $b - Az_{\ell+1}$, we have

$$\|b - Az_{\ell+1}\| \le \|r_\ell\|$$

because of the projection of the residual onto $\mathcal{W}$. To ensure the convergence of the projection method, the truncation $x_{\ell+1} = \mathcal{T}(z_{\ell+1})$ has to be done in such a way that the norm of the residual $r_{\ell+1} = b - Ax_{\ell+1}$ is not increased, i.e.

$$x_{\ell+1} = \mathcal{T}(z_{\ell+1}) \quad \text{s.t.} \quad \|r_{\ell+1}\| \le \|r_\ell\|.$$

Algorithm 2 Projection method in tensor format

choose x_0
for $\ell = 0, 1, \ldots$ **do**
  $r_\ell := b - Ax_\ell$
  if $\|r_\ell\| / \|b\| < \varepsilon$ **then**
    return x_ℓ
  end if
  $\tilde{v}_1 := \mathcal{T}(r_\ell)$
  $v_1 := \tilde{v}_1 / \|\tilde{v}_1\|$
  for $j = 1, \ldots, m$ **do**
    $w_j := Av_j$
    solve $(V_j^\top V_j)\alpha = V_j^\top w_j$
    $\tilde{v}_{j+1} := \mathcal{T}\left(w_j - \sum_{i=1}^{j} \alpha_i v_i\right)$
    $v_{j+1} := \tilde{v}_{j+1} / \|\tilde{v}_{j+1}\|$
  end for
  solve $(W_m^\top A V_m)y = W_m^\top r_\ell$
  $z_{\ell+1} := x_\ell + V_m y$
  $x_{\ell+1} := \mathcal{T}(z_{\ell+1})$ s.t. $\|b - Ax_{\ell+1}\| \leq \|r_\ell\|$
end for

The algorithm is summarised in Algorithm 2.

We have still left open which low rank format we want to use for the representation of vectors in $\mathbb{R}^{\mathcal{I}}$ and, in particular, how the truncation operator $\mathcal{T}$ shall look like. Assume for a moment that all vectors in Algorithm 2 were represented as rank k tensors. Each operation that requires the addition of two rank k tensors will result in a tensor of rank $2k$. Hence, the truncation of a rank $2k$ tensor to rank k is necessary if we do not want to increase the rank at each iteration step. A typical way to perform the truncation is to apply an alternating least squares (ALS) approach which – in most cases – converges rather slowly. An efficient algorithm for the truncation has been proposed by Espig [6] which is based on Newton-like techniques. Both algorithms are local optimisation procedures which crucially depend on the choice of initial values. Unfortunately, up to now there are no known algorithms which guarantee an a priori error bound for the truncation.

Consider on the other hand the representation of tensors in Tucker format. Here, reliable truncation procedures like the higher order SVD and a priori error bounds are available [16]. The major drawback of this format is its exponential complexity in d for the storage of the core tensor and thus for all relevant arithmetic operations. For low dimensions where $d \leq 6$, this might not be a problem and Algorithm 2 can be adapted to this case but for high dimensions with $d > 6$ we need something else.

In the following section we will present the so-called *hierarchical Tucker format* introduced by Hackbusch and Kühn [14] which addresses the above mentioned problems. The format provides both a representation of tensors and reliable truncation procedures with linear complexity in d. Therefore the hierarchical Tucker format is an efficient means for performing arithmetic operations with tensors without sacrificing the low rank structure. Hence, it is exactly what we need for our projection method in tensor format.

4 The Hierarchical Tucker Format

For the definition of the hierarchical Tucker format, we adopt the notation from [9].

Definition 6 (Dimension tree)**.** A *dimension tree* $T_{\mathcal{I}}$ for a dimension $d \in \mathbb{N}$ is a tree with root $\mathrm{Root}(T_{\mathcal{I}}) = \{1, \dots, d\}$ and depth $p = \lceil \log_2(d) \rceil := \min\{i \in \mathbb{N}_0 \mid i \geq \log_2(d)\}$ such that each node $t \in T_{\mathcal{I}}$ is either

1. a leaf and singleton $t = \{\mu\}$ on level $\ell \in \{p-1, p\}$ or
2. the disjoint union of two successors $S(t) = \{s_1, s_2\}$,

$$t = s_1 \dot{\cup} s_2.$$

The level ℓ of the tree is defined as the set of all nodes having a distance of exactly ℓ to the root. The set of leaves of the tree is denoted by $\mathcal{L}(T_{\mathcal{I}})$ and the set of interior (non-leaf) nodes is denoted by $\mathcal{J}(T_{\mathcal{I}})$. A node of the tree is a so-called *mode cluster* (a union of modes). The *canonical dimension tree* is a dimension tree where each node $t = \{\mu_1, \dots, \mu_q\}$, $q > 1$, has two successors

$$t_1 = \{\mu_1, \dots, \mu_r\}, \qquad t_2 = \{\mu_{r+1}, \dots, \mu_q\}, \quad r := \lfloor q/2 \rfloor := \max\{i \in \mathbb{N}_0 \mid i \leq q/2\}.$$

Definition 7 (Matricisation)**.** For a mode cluster t in a dimension tree $T_{\mathcal{I}}$ we define the complementary cluster $t' := \{1, \dots, d\} \setminus t$,

$$\mathcal{I}_t := \bigtimes_{\mu \in t} \mathcal{I}_\mu, \qquad \mathcal{I}_{t'} := \bigtimes_{\mu \in t'} \mathcal{I}_\mu,$$

and the corresponding *t-matricisation*

$$\mathcal{M}_t : \mathbb{R}^{\mathcal{I}} \mapsto \mathbb{R}^{\mathcal{I}_t \times \mathcal{I}_{t'}}, \qquad (\mathcal{M}_t(A))_{(i_\mu)_{\mu \in t}, (i_\mu)_{\mu \in t'}} := A_{(i_1, \dots, i_d)}.$$

We use the short notation $A^{(t)} := \mathcal{M}_t(A)$.

Definition 8 (Hierarchical rank)**.** Let $T_{\mathcal{I}}$ be a dimension tree. The *hierarchical rank* $(k_t)_{t \in T_{\mathcal{I}}}$ of a tensor $A \in \mathbb{R}^{\mathcal{I}}$ is defined by

$$\forall t \in T_{\mathcal{I}} : \quad k_t := \mathrm{rank}(A^{(t)}).$$

The set of all tensors of hierarchical rank (node-wise) at most $(k_t)_{t \in T_{\mathcal{I}}}$ is denoted by

$$\mathcal{H}\text{-Tucker}((k_t)_{t \in T_{\mathcal{I}}}) := \{A \in \mathbb{R}^{\mathcal{I}} \mid \forall t \in T_{\mathcal{I}} : \mathrm{rank}(A^{(t)}) \leq k_t\}.$$

Definition 9 (Frame tree, *t*-frame, transfer tensor)**.** Let $t \in T_{\mathcal{I}}$ be a mode cluster and $(k_t)_{t \in T_{\mathcal{I}}}$ a family of non-negative integers. We call a matrix $U_t \in \mathbb{R}^{\mathcal{I}_t \times k_t}$ a *t-frame* and the tuple $(U_s)_{s \in T_{\mathcal{I}}}$ of frames a *frame tree*. A frame is called orthogonal if its columns are orthonormal. A frame tree is called orthogonal if each non-root frame is. A frame tree is nested if for each interior mode cluster t with successor $S(t) = \{t_1, t_2\}$ the following relations holds:

$$\mathrm{span}\{(U_t)_i \mid 1 \leq i \leq k_t\} \subset \mathrm{span}\{(U_{t_1})_i \otimes (U_{t_2})_j \mid 1 \leq i \leq k_{t_1}, 1 \leq i \leq k_{t_2}\}.$$

The corresponding tensor $B_t \in \mathbb{R}^{k_t \times k_{t_1} \times k_{t_2}}$ of coefficients for the representation of the columns $(U_t)_i$ of U_t by the columns of U_{t_1} and U_{t_2},

$$(U_t)_i = \sum_{j=1}^{k_{t_1}} \sum_{l=1}^{k_{t_2}} (B_t)_{i,j,l} (U_{t_1})_j \otimes (U_{t_2})_l,$$

is called the *transfer tensor*.

For a nested frame tree it is sufficient to provide the transfer tensors B_t for all interior mode clusters $t \in \mathcal{J}(T_\mathcal{I})$ and the t-frames U_t for the leaves $t \in \mathcal{L}(T_\mathcal{I})$. Note that we have not yet imposed an orthogonality condition on the t-frames.

Definition 10 (Hierarchical Tucker format)**.** Let $T_\mathcal{I}$ be a dimension tree, $(k_t)_{t\in T_\mathcal{I}}$ a family of non-negative integers and $A \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$. Let $(U_t)_{t\in T_\mathcal{I}}$ be a nested frame tree with transfer tensors $(B_t)_{t\in\mathcal{J}(T_\mathcal{I})}$ and

$$\forall t \in T_\mathcal{I} : \operatorname{image}(A^{(t)}) = \operatorname{image}(U_t), \qquad A = U_{\{1,\dots,d\}}.$$

Then the representation $((B_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ is a *hierarchical Tucker representation* of A. The family $(k_t)_{t\in T_\mathcal{I}}$ is the hierarchical representation rank. Note that the columns of U_t need not be linear independent.

Definition 11 (Orthogonal frame projection)**.** Let $T_\mathcal{I}$ be a dimension tree, $t \in T_\mathcal{I}$ and U_t an orthogonal t-frame. We define the orthogonal frame projection $\pi_t : \mathbb{R}^\mathcal{I} \mapsto \mathbb{R}^\mathcal{I}$ in matricised form by

$$(\pi_t A)^{(t)} := U_t U_t^\top A^{(t)}.$$

Theorem 12 (Hierarchical truncation error, [9] Theorem 17 and Remark 18)**.** *Let $T_\mathcal{I}$ be a dimension tree and $A \in \mathbb{R}^\mathcal{I}$. Let A^{best} denote the best approximation of A in $\mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$ and let π_t be the orthogonal frame projection for the t-frame U_t that consists of the left singular vectors of A^t corresponding to the k_t largest singular values $\sigma_{t,i}$ of $A^{(t)}$. Then for any order of the projections π_t, $t \in T_\mathcal{I}$, holds*

$$\left\| A - \prod_{t\in T_\mathcal{I}} \pi_t A \right\| \le \sqrt{\sum_{t\in T_\mathcal{I}} \sum_{i>k_t} \sigma_{t,i}^2} \le \sqrt{2d-3}\left\| A - A^{\text{best}} \right\|.$$

Theorem 13 (Characterisation of hierarchical approximability, [9] Theorem 24)**.** *Let $T_\mathcal{I}$ be a dimension tree, $A \in \mathbb{R}^\mathcal{I}$, $(k_t)_{t\in T_\mathcal{I}}$ a family of non-negative integers and $\varepsilon > 0$. If there exists a tensor A^{best} of hierarchical rank $(k_t)_{t\in T_\mathcal{I}}$ and $\left\|A - A^{\text{best}}\right\| \le \varepsilon$, then the singular values of $A^{(t)}$ for each node t can be estimated by*

$$\sqrt{\sum_{i>k_t} \sigma_i^2} \le \varepsilon.$$

On the other hand, if the singular values fulfil the bound $\sqrt{\sum_{i>k_t} \sigma_i^2} \le \varepsilon/\sqrt{2d-3}$, then the truncation yields an $\mathcal{H}$-Tucker tensor $A_\mathcal{H} := \prod_{t\in T_\mathcal{I}} \pi_t A$ such that

$$\|A - A_\mathcal{H}\| \le \varepsilon.$$

This means that we can truncate a tensor $A \in \mathbb{R}^\mathcal{I}$ either to a given hierarchical rank $(k_t)_{t\in T_\mathcal{I}}$, or we can prescribe node-wise tolerances $\varepsilon/\sqrt{2d-2}$ to obtain a guaranteed error bound of $\|A - A_\mathcal{H}\| \le \varepsilon$. As we are especially interested in calculations of fixed rank, we will introduce a truncation operator that bounds the node-wise ranks by a constant.

Definition 14 (Truncation operator)**.** Let $k \in \mathbb{N}$ and $T_\mathcal{I}$ a dimension tree. For $A \in \mathbb{R}^\mathcal{I}$ we introduce the truncation operator $\mathcal{T}_k : \mathbb{R}^\mathcal{I} \to \mathbb{R}^\mathcal{I}$ defined by

$$\mathcal{T}_k(A) := \prod_{t\in T_\mathcal{I}} \pi_t A, \tag{10}$$

where for $t \in T_\mathcal{I}$ the columns of the frame $U_t \in \mathbb{R}^{\mathcal{I}_t \times k}$ are the first k left singular vectors of $A^{(t)}$.

If a tensor is already given in hierarchical Tucker format, $A \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$, then the truncation can be performed in a complexity of

$$\mathcal{O}\left(d \max_{t\in T_\mathcal{I}} k_t^4 + \sum_{\mu=1}^{d} n_\mu k_\mu^2\right).$$

The hierarchical Tucker format provides a favourable setting for common arithmetic operations like addition and multiplication of tensors and the matrix-vector product. Since they follow directly from the definition of the hierarchical Tucker format, the following three lemmas are given without proof.

Lemma 15 (Addition)**.** *Let $T_\mathcal{I}$ be a dimension tree and $A \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$, $A' \in \mathcal{H}\text{-Tucker}((k'_t)_{t\in T_\mathcal{I}})$. Moreover, let $((B_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ and $((B'_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U'_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ be hierarchical Tucker representations of A and A', respectively. Define a family of non-negative integers $(k''_t)_{t\in T_\mathcal{I}}$ by*

$$\forall t \in T_\mathcal{I} \setminus \operatorname{Root}(T_\mathcal{I}): \quad k''_t := k_t + k'_t,$$

and $k''_{\{1,\dots,d\}} := 1$, the t-frames

$$\forall t \in \mathcal{L}(T_\mathcal{I}): \quad U''_t := [U_t \mid U'_t],$$

and the transfer tensors

$$\forall t \in \mathcal{J}(T_\mathcal{I}) \setminus \operatorname{Root}(T_\mathcal{I}): \quad B''_t \in \mathbb{R}^{k''_t \times k''_{t_1} \times k''_{t_2}}$$

where $t = t_1 \dot\cup t_2$ and

$$(B''_t)_{i,j,l} := \begin{cases} (B_t)_{i,j,l}, & 1 \le i \le k_t,\ 1 \le j \le k_{t_1},\ 1 \le l \le k_{t_2}, \\ (B'_t)_{i-k_t, j-k_{t_1}, l-k_{t_2}}, & 1 \le i-k_t \le k'_t,\ 1 \le j-k_{t_1} \le k'_{t_1},\ 1 \le l-k_{t_2} \le k'_{t_2}, \\ 0, & \text{otherwise}, \end{cases}$$

and $B''_{\{1,\dots,d\}} \in \mathbb{R}^{1\times k''_{t_1}\times k''_{t_2}}$ where $\{1,\dots,d\} = t_1 \dot\cup t_2$ and

$$(B''_{\{1,\dots,d\}})_{1,j,l} := \begin{cases} (B_{\{1,\dots,d\}})_{1,j,l}, & 1 \le j \le k_{t_1},\ 1 \le l \le k_{t_2}, \\ (B'_{\{1,\dots,d\}})_{1,j-k_{t_1},l-k_{t_2}}, & 1 \le j-k_{t_1} \le k'_{t_1},\ 1 \le l-k_{t_2} \le k'_{t_2}, \\ 0, & \text{otherwise}. \end{cases}$$

Then $((B''_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U''_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ is a hierarchical Tucker representation of $A'' := A + A'$.

Lemma 16 (Scalar multiplication)**.** *Let $T_\mathcal{I}$ be a dimension tree and $A \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$ with hierarchical Tucker representation $((B_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U_t)_{t\in\mathcal{L}(T_\mathcal{I})})$. For $c \in \mathbb{R}$ define $B'_{\{1,\dots,d\}} := cB_{\{1,\dots,d\}}$ and for all other $t \in \mathcal{J}(T_\mathcal{I}) \setminus \operatorname{Root}(T_\mathcal{I})$ let $B'_t := B_t$. Then $((B'_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ is a hierarchical Tucker representation of $A' := cA$.*

Lemma 17 (Matrix-vector multiplication)**.** *Let $T_\mathcal{I}$ be a dimension tree and $A \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$ with hierarchical Tucker representation $((B_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U_t)_{t\in\mathcal{L}(T_\mathcal{I})})$. Let $M \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ be a matrix given by*

$$M = \bigotimes_{\mu=1}^{d} M_\mu, \quad M_\mu \in \mathbb{R}^{n_\mu \times n_\mu}.$$

For all $t = \{\mu\} \in \mathcal{L}(T_\mathcal{I})$ define

$$U'_t := M_\mu U_t.$$

Then $((B_t)_{t\in\mathcal{J}(T_\mathcal{I})}, (U'_t)_{t\in\mathcal{L}(T_\mathcal{I})})$ is a hierarchical Tucker representation of $A' := MA$.

It is straight-forward to see that the multiplication has a complexity of $\mathcal{O}(\sum_{\mu=1}^{d} n_\mu^2 k_\mu)$ if the M_μ are unstructured and $\mathcal{O}(\sum_{\mu=1}^{d} n_\mu k_\mu)$ if the M_μ are sparse (i.e. allow for a matrix-vector multiplication in $\mathcal{O}(n_\mu)$).

Remark 18 (Norm, scalar product)**.** In [9] we have shown how to compute the Euclidean norm of a tensor in hierarchical Tucker format in $\mathcal{O}\left(d \max_{t\in T_\mathcal{I}} k_t^4 + \sum_{\mu=1}^{d} n_\mu k_\mu^2\right)$. The scalar product of two vectors $v, w \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$ may then be calculated via the elementary formula

$$\langle v, w\rangle = \frac{1}{2}\left(\|v+w\|^2 - \|v\|^2 - \|w\|^2\right).$$

5 A Projection Method in Hierarchical Tucker Format

In this section, we adapt the projection method in tensor format to the setting of the hierarchical Tucker format. For this purpose, we will follow the algorithmic concept introduced in Section 3.2. The basic idea is to construct a low-dimensional subspace which is spanned by vectors of low rank in hierarchical Tucker format. Let us assume that $x_\ell, b \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$. According to the arithmetic operations introduced in the last section, we may calculate $r_\ell = b - Ax_\ell$ directly in hierarchical Tucker format. We now can use the residual to construct a low-dimensional subspace of vectors of low rank. To keep computations as cheap as possible, we will bound the node-wise ranks of the vectors v_j spanning the subspace by a constant $k_v \in \mathbb{N}$. In analogy to Section 3.2 we define

$$\tilde{v}_1 := \mathcal{T}_{k_v}(r_\ell), \qquad v_1 := \tilde{v}_1 / \|\tilde{v}_1\|, \tag{11}$$

where we have specified the truncation operator to be of the form (10). Now let $v_1, \dots, v_j \in \mathcal{H}\text{-Tucker}((k_t)_{t\in T_\mathcal{I}})$ be given such that $k_t \le k_v$ for all $t \in T_\mathcal{I}$. Define $V_j := [v_1, \dots, v_j] \in \mathbb{R}^{\mathcal{I}\times j}$ and let $w_j := Av_j$. For the definition of a new basis vector, we have to solve the system $(V_j^\top V_j)\alpha = V_j^\top w_j$. The calculation of the entries of the matrix $V_j^\top V_j$ and of the vector $V_j^\top w_j$ merely requires the evaluation of scalar products which can easily be done according to Remark 18. A new basis vector is now defined by

$$\tilde{v}_{j+1} := \mathcal{T}_{k_v}\left(w_j - \sum_{i=1}^{j} \alpha_i v_i\right), \qquad v_{j+1} := \tilde{v}_{j+1} / \|\tilde{v}_{j+1}\|.$$

The projection of the residual onto the subspace $\mathcal{W}$ spanned by the columns of $W_m = [w_1, \dots, w_m]$ requires the solution of $(W_m^\top A V_m)y = W_m^\top r_0$. Again, the matrix $W_m^\top A V_m$ and the vector $W_m^\top r_0$ may be calculated by the evaluation of scalar products of tensors in hierarchical Tucker format.

Until now, we do not know whether the subspace $\mathcal{V}$ spanned by the columns of V_m was "good" enough to lead to a sufficient decrease in the residual. What we expect is that for increasing values of k_v, $\mathcal{V}$ attains similar properties as the exact Krylov subspace. To control the decrease of the residual, we introduce a parameter $\rho \in (0,1)$ which is assumed to be small. If the new residual $r_\ell - \sum_{i=1}^{m} y_i w_i$ fulfils

$$\left\| r_\ell - \sum_{i=1}^{m} y_i w_i \right\| < (1-\rho)\|r_\ell\|, \tag{12}$$

we have chosen a subspace $\mathcal{V}$ that guarantees a decrease of the residual with rate $1-\rho$. If (12) is not fulfilled, we have to modify the subspace $\mathcal{V}$ such that it contains an element which

decreases the residual faster. We strongly expect that this is the case if we approximate the exact Krylov space better. Therefore, we increase k_v and construct a new set of basis vectors by starting at (11) where we substitute the truncation operator $\mathcal{T}_{k_v}$ by $\mathcal{T}_{k_v+1}$.

Assume now that (12) is fulfilled. Formally, the new iterate may be written as $z_{\ell+1} := x_\ell + V_m y$. As the addition of tensors increases the rank, we also would like to apply a truncation operator to the new iterate. In a similar way as before, we may truncate $z_{\ell+1}$ by $\mathcal{T}_{k_x}$ where $k_x \in \mathbb{N}$ is chosen such that the convergence is preserved. This means that we have to require

$$\|b - Ax_{\ell+1}\| < \|r_\ell\| \tag{13}$$

where $x_{\ell+1} := \mathcal{T}_{k_x}(z_{\ell+1})$. By subsequently increasing k_x, we can find a value such that (13) is fulfilled since for the exact iterate $z_{\ell+1}$ we can rely on the bound (12). Thus we have arrived at a new iterate $x_{\ell+1}$ which is again given in tensor format of low rank. We have summarised all steps in Algorithm 3.

As the truncation of tensors in hierarchical Tucker format scales like $\mathcal{O}(k^4)$ for the node-wise ranks, it is advantageous to keep all ranks as small as possible throughout the whole iteration process. At first, this may seem unnatural as for this purpose we have to choose a small value of ρ resulting in a slow convergence behaviour. But – as the numerical evidence shows – this policy pays off, since we can perform a much higher number of iterations at the same computational costs.

Algorithm 3 Projection method in hierarchical Tucker format

choose x_0 and $\rho \in (0, 1)$
$k_v := 1$, $k_x := 1$
for $\ell = 0, 1, \ldots$ **do**
  $r_\ell := b - Ax_\ell$
  if $\|r_\ell\| / \|b\| < \varepsilon$ **then**
    return x_ℓ
  end if
  repeat
    construct $v_1, \ldots, v_m$ with $\mathcal{T} := \mathcal{T}_{k_v}$ as in lines 7 to 14 of Algorithm 2
    solve $(W_m^\top A V_m) y = W_m^\top r_\ell$
    $k_v := k_v + 1$
  until $\|r_\ell - \sum_{i=1}^m y_i w_i\| < (1 - \rho) \|r_\ell\|$
  $z_{\ell+1} := x_\ell + V_m y$
  repeat
    $x_{\ell+1} := \mathcal{T}_{k_x}(z_{\ell+1})$
    $k_x := k_x + 1$
  until $\|b - Ax_{\ell+1}\| < \|r_\ell\|$
end for

6 Multigrid Acceleration

In the previous section we have considered the solution of a general linear system in Kronecker form. Many problems of practical interest however, require the solution of a linear system (1) where the system matrix stems from the discretisation of a partial differential equation. In this case GMRES type methods can suffer from large mode sizes n_μ and their convergence rate tends to 1 as $n_\mu \to \infty$.

For elliptic problems of large scale, multigrid methods have become the method of choice due to their fast convergence and linear scaling. A comprehensive introduction may be found in [11], and we summarise the necessary basic ingredients in the following. It will turn out that one can use the multigrid method for iterates in low rank tensor format and achieve linear scaling in the mode size n_μ as well as the order d of the tensor.

The main idea is to construct a hierarchy of grids (hence the name multigrid) which are used to reduce different frequency components of the error. The basic ingredients of a multigrid method for the solution of a linear system (1) are:

1. A hierarchy of discrete problems

$$A_\ell x_\ell = b_\ell, \quad A_\ell \in \mathbb{R}^{N_\ell \times N_\ell}, \quad \ell = 1, \dots, L,$$

   where $N_1 < \dots < N_L$, $N_L = \#\mathcal{I}$, such that the problem on the coarsest level $\ell = 1$ may be solved directly, two subsequent systems are strongly related, and the problem on the finest level $\ell = L$ is the original problem with $A_L = A$ that has to be solved.

2. Prolongation and restriction operators

$$P_{\ell-1}^\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}, \quad R_\ell^{\ell-1} : \mathbb{R}^{N_\ell} \to \mathbb{R}^{N_{\ell-1}}$$

   which transfer a vector from a grid on level ℓ to a vector on the next finer or coarser grid on level $\ell + 1$ or $\ell - 1$.

3. A smoothing operator

$$x_\ell^{i+1} = S_\ell(x_\ell^i, b_\ell), \quad \ell = 1, \dots, L$$

   that reduces high frequency components of the error.

Consider as an example a one-dimensional domain $\Omega = (0, 1)$. A hierarchy of grids with equidistant nodes may be defined by $\Omega_\ell := \{i/2^\ell \mid 1 \le i < 2^\ell\}$. A typical choice of the prolongation and restriction operator is then given by the corresponding matrices

$$P_{\ell-1}^\ell = \frac{1}{2}\begin{pmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & \ddots \end{pmatrix}, \quad R_\ell^{\ell-1} = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & \\ & & 1 & 2 & 1 & \\ & & & & & \ddots \end{pmatrix}.$$

The idea of a hierarchy of grids can be adapted to the multidimensional setting. Let $\Omega := \Omega_1 \times \dots \times \Omega_d$ and let $\Omega_{\mu,\ell}$ be a hierarchy of grids in mode-μ direction $\mu \in \{1, \dots, d\}$. Then

$$\Omega_\ell := \Omega_{1,\ell} \times \dots \times \Omega_{d,\ell}, \quad \ell = 1, \dots, L,$$

defines a hierarchy of grids for the domain Ω. Due to the product structure of the hierarchy of grids, the prolongation and restriction operators possess a nice tensor product structure. Let $P_{\mu,\ell-1}^\ell$ and $R_{\mu,\ell}^{\ell-1}$ be prolongation and restriction operators in mode-μ-direction $\mu \in \{1, \dots, d\}$, respectively. Then

$$P_{\ell-1}^\ell := P_{1,\ell-1}^\ell \otimes \dots \otimes P_{d,\ell-1}^\ell, \qquad R_{\ell-1}^\ell := R_{1,\ell}^{\ell-1} \otimes \dots \otimes R_{d,\ell}^{\ell-1},$$

define prolongation and restriction operators for the hierarchy of grids Ω_ℓ.

The smoothing operator S_ℓ is not required to be a good solver, but it should remove high frequency components of the error. A typical choice of S_ℓ is a Jacobi or a Gauss-Seidel method. Here, for simplicity, we choose a Richardson method which is defined by

$$x_\ell^{i+1} := x_\ell^i + \omega_\ell(b_\ell - A_\ell x_\ell^i)$$

where $0 < \omega_\ell < 2/\varrho(A_\ell)$ and $\varrho(A_\ell)$ is the spectral radius of A_ℓ.

We still have to address how the linear systems on the coarsest level should be solved. For the low-dimensional case, this is typically a small system which can be solved iteratively or by direct methods. However, for high dimensions even the system on the coarsest level might be too large. We therefore propose to use the projection method in hierarchical Tucker format to solve this system. We have summarised the whole multigrid procedure in Algorithm 4.

Algorithm 4 $x_\ell = \text{multigrid}(\ell, x_\ell, b_\ell)$

```
if ℓ = 1 then
  solve A_1 x_1 = b_1 by Algorithm 3
  return x_1
else
  for i = 1 to ν_1 do
    x_ℓ := x_ℓ + ω_ℓ(b_ℓ − A_ℓ x_ℓ)
  end for
  d_{ℓ−1} := R_ℓ^{ℓ−1}(b_ℓ − A_ℓ x_ℓ)
  e_{ℓ−1} := 0
  for i = 1 to γ do
    e_{ℓ−1} = multigrid(ℓ − 1, e_{ℓ−1}, d_{ℓ−1})
  end for
  x_ℓ := x_ℓ + P_{ℓ−1}^ℓ e_{ℓ−1}
  for i = 1 to ν_2 do
    x_ℓ := x_ℓ + ω_ℓ(b_ℓ − A_ℓ x_ℓ)
  end for
  return x_ℓ
end if
```

In practice it might be advisable not to use the multigrid method directly, but as a preconditioner in an iterative solver. For the iterative solver one can use again the GMRES-type projection method that we have presented here.

7 Application in an Eigenvalue Solver

We consider the problem to find eigenvalues and corresponding eigenvectors of large matrices, i.e. we seek a pair $(\lambda, x) \in \mathbb{R} \times \mathbb{R}^{\mathcal{I}} \setminus \{0\}$ such that

$$Ax = \lambda x.$$

In many applications, one specifically wants to find the smallest or the largest eigenvalue $\lambda_{\min}$ or $\lambda_{\max}$ for a given matrix, the eigenvectors corresponding to a few largest or smallest eigenvalues, or all eigenvectors corresponding to eigenvalues in a specific part of the complex plane. The

largest eigenvalue may be found by the well-known power method which only requires matrix-vector products. The smallest eigenvalue can be determined by the inverse power method which relies on the subsequent solution of linear systems of the form $Ax^{(\ell+1)} = y^{(\ell)}$ where $y^{(\ell+1)} := x^{(\ell+1)} / \left\|x^{(\ell+1)}\right\|$. The convergence of this iteration process may be controlled by means of the Rayleigh quotient $\Lambda_A(x) := \langle x, Ax\rangle / \langle x, x\rangle$. For an exact eigenpair (λ, x), we have $\lambda = \Lambda_A(x)$. It is therefore a good strategy to use the Rayleigh quotient $\lambda^{(\ell)} := \Lambda_A(x^{(\ell)})$ to estimate an eigenvalue from its approximate eigenvector. The inverse power method may be stopped if $\left\|Ax^{(\ell)} - \lambda^{(\ell)} x^{(\ell)}\right\| < \varepsilon$ for some $\varepsilon > 0$. We have summarised the inverse power method in Algorithm 5.

Algorithm 5 Inverse power method

choose $\varepsilon > 0$ and $y^{(1)} \in \mathbb{R}^{\mathcal{I}}$ with $\left\|y^{(1)}\right\| = 1$
for $\ell = 1, 2, \ldots$ **do**
  solve $Ax^{(\ell+1)} = y^{(\ell)}$
  $y^{(\ell+1)} := x^{(\ell+1)} / \left\|x^{(\ell+1)}\right\|$
  $\lambda^{(\ell+1)} := \Lambda_A(y^{(\ell+1)})$
  if $\left\|Ay^{(\ell+1)} - \lambda^{(\ell+1)} y^{(\ell+1)}\right\| < \varepsilon$ **then**
    return $y^{(\ell+1)}$
  end if
end for

In general, any eigenvalue of A which is sufficiently separated from the rest of the spectrum, may be found by introducing a shift $\sigma \in \mathbb{R}$ which is assumed to be close to the sought eigenvalue. More precisely, let us assume that A has the eigenvalues $\lambda_1, \ldots, \lambda_N$ and that

$$|\sigma - \lambda_i| < \min\{|\sigma - \lambda_j| : 1 \le j \le N, j \ne i\}. \tag{14}$$

If $\sigma \ne \lambda_i$, we have

$$Ax = \lambda_i x \quad \Longleftrightarrow \quad (A - \sigma I)^{-1} x = \frac{1}{\lambda_i - \sigma} x.$$

Hence, $(A - \sigma I)^{-1}$ has the same eigenvectors as A. Moreover, $1/(\lambda_i - \sigma)$ is the largest eigenvalue of $(A - \sigma I)^{-1}$ which can be found by the inverse power method. The application of the inverse power method to the shifted matrix $A - \sigma I$ is commonly known as a shift-invert strategy. Note that if A is given in Kronecker format, also the matrix $A - \sigma I$ possesses this structure as the identity may be written as $I = I_1 \otimes \ldots \otimes I_d$. In this case, the linear system in Algorithm 5 may therefore be solved by the projection method in hierarchical Tucker format.

For eigenvalue problems of large scale it might not be advisable to apply the shift-invert strategy directly since the involved solution of a linear system by a GMRES-type method may suffer from a slow convergence rate. In analogy to the solution of linear systems of large scale, elliptic eigenvalue problems may efficiently be treated using a multigrid method [10]. A detailed analyses of this multigrid eigenvalue scheme has been given in [1]. As in the usual multigrid method, the eigenvalue problem has only to be solved on the coarsest level. The eigenvectors on the finer levels result from the application of prolongation, restriction and smoothing operators. This setting can easily be adapted to the multi-dimensional case where we propose to use the projection method in hierarchical Tucker format as a solver on the coarsest level. This allows us to treat elliptic eigenvalue problems of large scale in high dimensions.

8 Numerical Examples

The following numerical examples shall illustrate the potential of the projection method for the solution of linear systems in high dimensions. We are specifically interested in the following questions:

- Does the method converge sufficiently fast for all dimensions d?
- Does the hierarchical rank of x only increase moderately throughout the iteration process?
- Can we combine large dimensions d with a large mode size n_μ?

Fortunately, we can answer all questions in the affirmative, at least for the range of examples that we consider. For simplicity, let now $n := n_1 = \ldots = n_d$. In all examples, the dimension m of the subspace $\mathcal{V}$ is bounded by $m = 10$ but it may also be smaller when the basis vectors of $\mathcal{V}$ become linearly dependent. The parameter ρ introduced in (12) has a strong impact on both the convergence rate of the method and on the hierarchical ranks of the basis vectors spanning $\mathcal{V}$. The higher the value of ρ, the higher is the convergence rate of the method implying a low number of iteration steps. On the other hand, a higher value of ρ leads to higher ranks for the basis vectors spanning $\mathcal{V}$ which implies a much longer time for the truncation of vectors. The choice of ρ is therefore a trade-off and we are not aware of an optimal value. As a conservative choice which works well in all examples we have set $\rho := 10^{-4}$.

Example 19 (Symmetric case, [8])**.** As a symmetric example, we consider the Poisson equation in d dimensions, i.e.

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega = (0,1)^d, \\ u &= 0 \quad \text{on } \Gamma := \partial\Omega. \end{aligned}$$

A finite difference discretisation with mesh-width h leads to a linear system $Ax = b$ where A is of the form (4) with

$$A_\mu = \frac{1}{h^2} \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}.$$

Let the right-hand side be given such that it corresponds to the solution $u = \prod_{\mu=1}^{d}(x_\mu - x_\mu^2)$. We now fix the value of $n := 10$ and check the convergence of the projection method for dimensions $d = 4, 8, 16, 32, 64$. The decrease of the relative residual $\|Ax - b\| / \|b\|$ is shown in Fig. 1. Remarkably, the maximal hierarchical rank k_x remains bounded by 1 throughout the whole solution process.

Example 20 (Comparison to CG method)**.** In the symmetric case, it is also possible to implement a conjugate gradient method that completely relies on truncated calculations in hierarchical Tucker format. To investigate this method, we made two experiments. First, we fixed $\varepsilon = 10^{-10}$ and truncated all vectors in the CG method with an accuracy of ε where all node-wise ranks were determined adaptively. Second, we fixed an upper bound on the hierarchical rank $k_x = 5$ and compared both cases for $d = 8$ and $n = 10$ in Example 19. In both experiments the decrease of the residual is similar up to an accuracy of 10^{-3} (cf. Fig. 2) but then the sequence of iterates with fixed rank stagnates. The convergence can only be maintained by increasing the

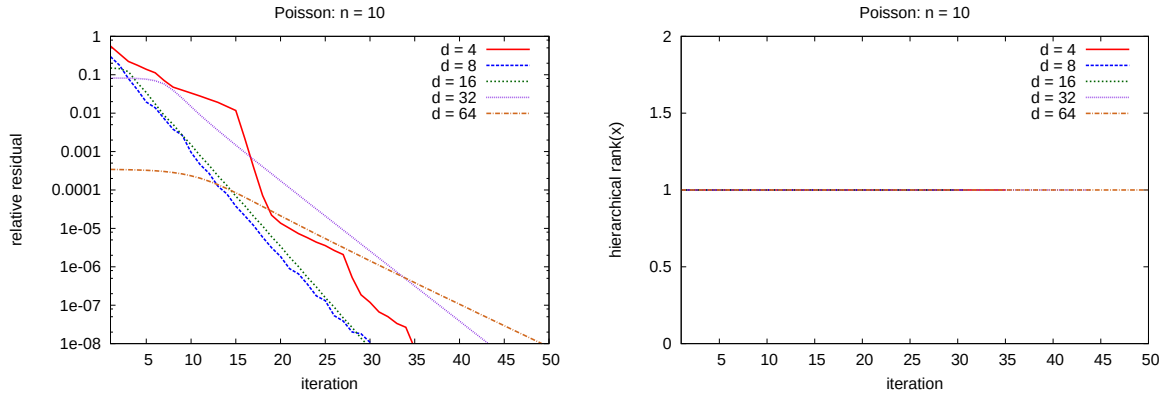

Figure 1: Left: relative residual $\|Ax - b\| / \|b\|$ for the Poisson equation with $n = 10$, right: maximal hierarchical rank k_x.

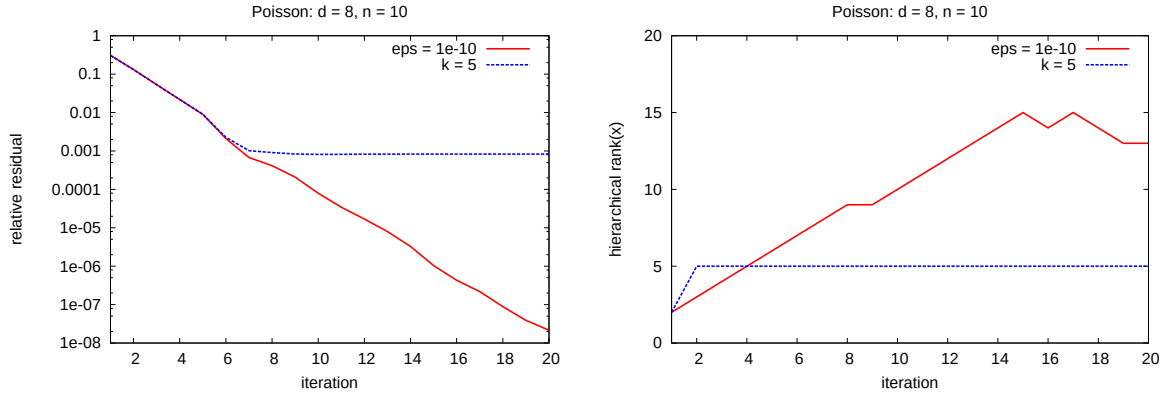

Figure 2: Truncated CG method applied to the Poisson equation for $d = 8$ and $n = 10$ with fixed accuracy or fixed rank. Left: relative residual $\|Ax - b\| / \|b\|$. Right: hierarchical rank k_x.

hierarchical rank up to 15. It seems clear that this behaviour stems from the fact that for fixed ranks the gradient directions in the CG method lose orthogonality. This can only be overcome by increasing the ranks further which has a strong impact on running times as the truncation of tensors in hierarchical Tucker format scales like $\mathcal{O}(k^4)$. It therefore pays off to work with an iterative scheme which is based on calculations with tensors of fixed rank instead of determining all ranks adaptively for a fixed accuracy. This in turn means that it is advantageous to develop solution methods in tensor format that do not only rely on orthogonal directions like the CG method but to consider more general strategies like the GMRES-type method that we present here.

Example 21 (Non-symmetric case, [8])**.** As a non-symmetric example, we consider the convection-diffusion equation

$$\begin{aligned} -\Delta u + c^\top \nabla u &= f \quad \text{in } \Omega = (0,1)^d, \\ u &= 0 \quad \text{on } \Gamma := \partial\Omega. \end{aligned}$$

A finite difference discretisation combined with a second order convergent scheme for the con-

vection term leads to a linear system $Ax = b$ where A is of the form (4) with

$$A_\mu = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix} + \frac{c_\mu}{4h}\begin{pmatrix} 3 & -5 & 1 & & \\ 1 & 3 & -5 & \ddots & \\ & \ddots & \ddots & \ddots & 1 \\ & & 1 & 3 & -5 \\ & & & 1 & 3 \end{pmatrix}.$$

As in the last example, let the right hand side be given such that it corresponds to the solution $u = \prod_{\mu=1}^{d}(x_\mu - x_\mu^2)$ and fix $n := 10$. For $c_\mu := 10$, the matrix A is still positive definite. We now check the convergence for dimensions $d = 4, 8, 16, 32, 64$. The decrease of the relative residual $\|Ax - b\| / \|b\|$ is shown in Fig. 3. In this case, the maximal hierarchical rank of x increases moderately throughout the whole iteration process.

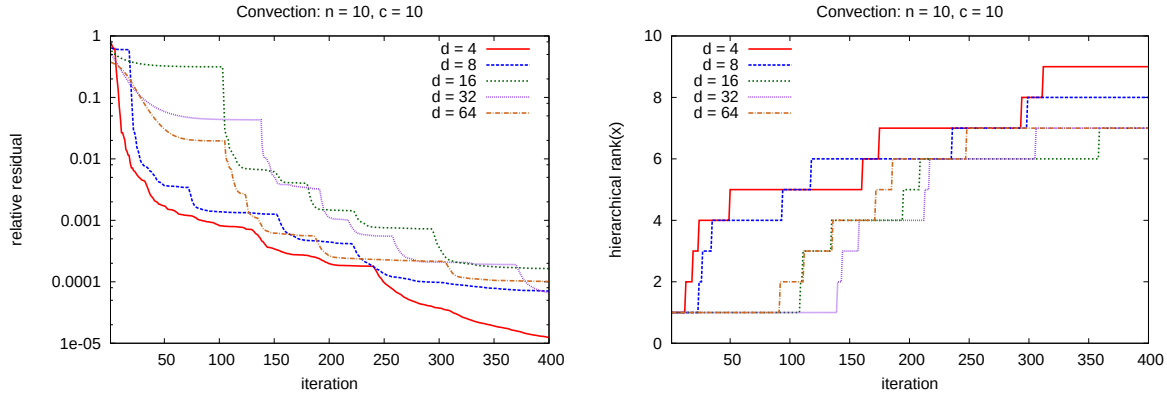

Figure 3: Left: Relative residual $\|Ax - b\| / \|b\|$ for the convection-diffusion equation with $n = 10$ and $c_\mu = 10$, right: maximal hierarchical rank k_x.

Example 22 (Multigrid)**.** For matrices stemming from a finite difference or finite element discretisation of a partial differential equation, typically the condition number increases with the refinement level of the discretisation. One way to overcome this, is to use a preconditioner in the solution process. Here, we focus on a multigrid method which has similar effects. Consider first a naive way of solving Example 19 for fixed dimension $d = 4$ and increasing mode size n. As Fig. 4 illustrates, this leads to a rapid increase in the number of iteration steps. Using a multigrid method, the dependence on n can be completely removed as Fig. 5 illustrates. In another experiment, we fix the mode size $n = 1023$ and test the convergence of the multigrid method for dimensions $d = 4, 8, 16, 32$. As Fig. 5 demonstrates, the convergence seems to be independent of the dimension which allows for the solution of really large problems.

Example 23 (Eigenvalues)**.** As an example for finding eigenvalues and corresponding eigenvectors of a given matrix, let us look at the eigenvalue problem for the Laplace operator

$$\begin{aligned} -\Delta u &= \lambda u \quad \text{in } \Omega := (0,1)^d, \\ u &= 0 \quad \text{on } \Gamma := \partial\Omega. \end{aligned}$$

The eigenvalues of the corresponding finite difference matrix A are given by

$$\lambda_{(i_1,\ldots,i_d)} = 2(n+1)^2 \sum_{\mu=1}^{d} \left(1 - \cos\frac{i_\mu \pi}{n+1}\right), \quad 1 \le i_\mu \le n.$$

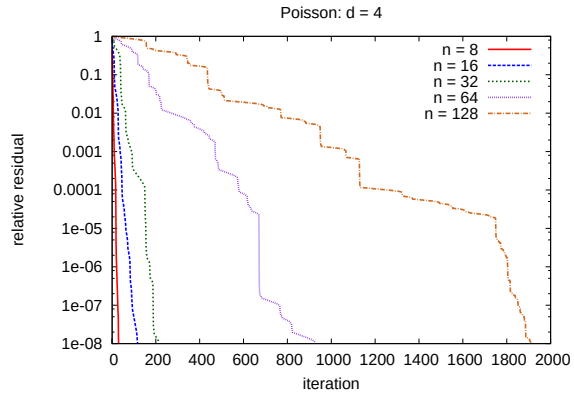

Figure 4: Relative residual $\|Ax - b\| / \|b\|$ for the Poisson equation with $d = 4$

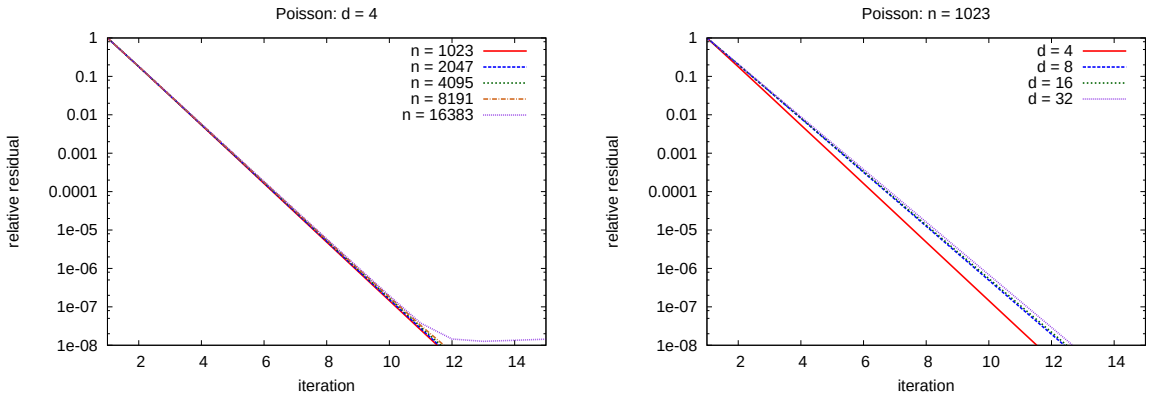

Figure 5: Relative residual $\|Ax - b\| / \|b\|$ for the multigrid method applied to the Poisson equation. Left: $d = 4$. Right: $n = 1023$.

We are interested in the calculation of eigenvalues and corresponding eigenvectors for the following three cases:

- the minimal eigenvalue $\lambda_{\underline{1}} := \lambda_{(1,\dots,1)}$,
- an intermediate eigenvalue $\lambda_{\underline{2}} := \lambda_{(2,\dots,2)}$,
- the maximal eigenvalue $\lambda_{\underline{n}} := \lambda_{(n,\dots,n)}$.

Note that $\lambda_{\underline{1}}$, $\lambda_{\underline{2}}$, and $\lambda_{\underline{n}}$ are simple eigenvalues of A. The shift-invert strategy may be applied to all three cases if we choose a shift σ that is close enough to the sought eigenvalue. In this example, we take

$$\begin{aligned}
\sigma_{\underline{1}} &:= \lambda_{\underline{1}} - 2(n+1)^2(1 - \cos(\pi/(n+1))), \\
\sigma_{\underline{2}} &:= \lambda_{\underline{2}} - \tfrac{1}{8}(n+1)^2 \left|\cos(3\pi/(n+1)) - \cos(2\pi/(n+1))\right|, \\
\sigma_{\underline{n}} &:= \lambda_{\underline{n}} + 2(n+1)^2(1 - \cos(\pi/(n+1))),
\end{aligned}$$

respectively. Note that in all three cases condition (14) is fulfilled. In a first experiment, we apply the shift-invert strategy for $\lambda_{\underline{1}}$ in dimension $d = 4, 8, 16, 32$ and fix $n = 10$. The result is

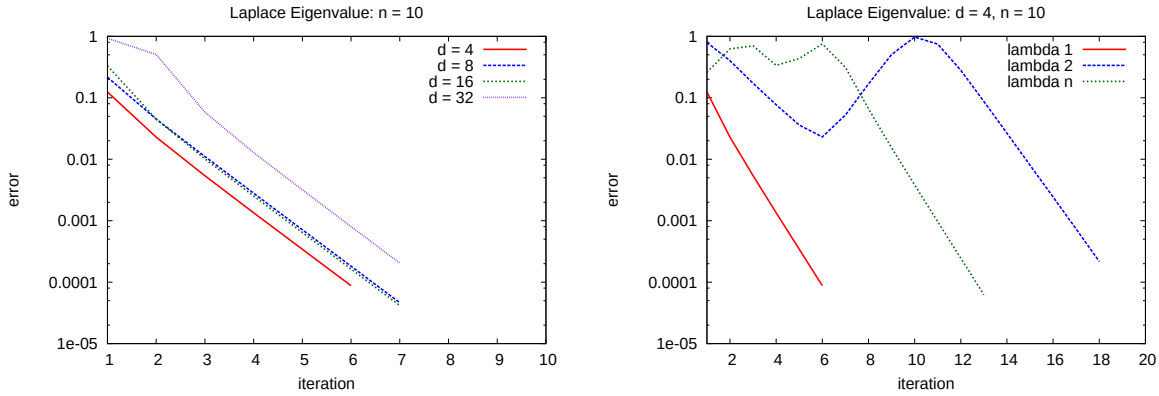

Figure 6: Left: error $\|Ax^{(\ell)} - \lambda^{(\ell)}x^{(\ell)}\|$ of the shift-invert strategy with shift $\sigma_{\underline{1}}$ applied to the Laplace eigenvalue problem with $n = 10$. Right: error for $d = 4$ and $n = 10$ with shifts $\sigma_{\underline{1}}$, $\sigma_{\underline{2}}$ and $\sigma_{\underline{n}}$.

shown in Fig. 6. Second, we apply the shift-invert strategy for $\lambda_{\underline{1}}$, $\lambda_{\underline{2}}$, $\lambda_{\underline{n}}$ with fixed $d = 4$ and $n = 10$ (cf. Fig. 6).

As the first experiment illustrates, the convergence of the shift invert-strategy for the smallest eigenvalue seems to be independent of the dimension. The second experiment demonstrates that we also obtain convergence in the case of an intermediate and the maximal eigenvalue. Here, the convergence behaviour slightly differs from the previous case. At first, the approximate eigenvector tends to converge to an eigenvector belonging to an eigenvalue different from the sought one. After some iterations, the error increases again and thereafter the correct eigenvector is found. In particular, one has to take care if an intermediate eigenvalue needs to be calculated. On the one hand, the shift has to be chosen quite carefully in order to approximate the correct eigenvalue. On the other hand, the system matrix $(A - \sigma I)$ becomes indefinite which has a negative influence on the convergence behaviour of our method resulting in a very high number of iterations. The calculation of an intermediate eigenvalue may therefore require some more sophisticated techniques which are out of the scope of this article.

Example 24 (Eigenvalue Multigrid)**.** Elliptic eigenvalue problems with a large mode size n may efficiently been treated using a multigrid eigenvalue scheme, cf. [10], [1]. Here, we apply the eigenvalue multigrid method to the previous example with fixed mode size $n = 1023$ and look for the smallest eigenvalue $\lambda_{\underline{1}}$. In the multigrid scheme, we use eight different levels where the eigenvalue problem is only solved on the coarsest level. The decrease of the error for dimensions $d = 4, 8, 16$ is shown in Fig. 7.

As this final example demonstrates, the projection method in hierarchical Tucker format perfectly fits into the setting of multidimensional eigenvalue problems. The combination with a multigrid scheme allows for applications of large scale in high dimensions.

References

[1] L. Banjai, S. Börm, and S. Sauter. FEM for elliptic eigenvalue problems: how coarse can the coarsest mesh be chosen? An experimental study. *Comput. Visual. Sci.*, 11(4–6):363–372, 2008.

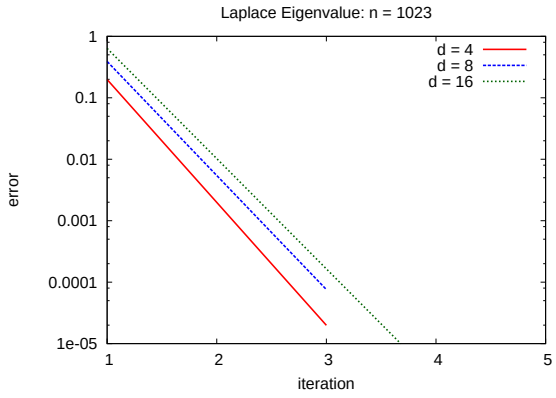

Figure 7: Error $\left\|Ax^{(\ell)} - \lambda^{(\ell)}x^{(\ell)}\right\|$ of the eigenvalue multigrid method applied to the Laplace eigenvalue problem on the finest level with $n = 1023$.

[2] Gregory Beylkin and Martin J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proc. Natl. Acad. Sci. USA*, 99:10246–10251, 2002.

[3] Gregory Beylkin and Martin J. Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM J. Sci. Comput.*, 26(6):2133–2159, 2005.

[4] S. Börm and R. Hiptmair. Analysis of tensor product multigrid. *Numer. Algorithms*, 26(3):219–234, 2001.

[5] Lars Eldén and Berkant Savas. Krylov subspace methods for tensor computations. Technical Report 2, Linköpings Universitet, 2009.

[6] Mike Espig. *Effiziente Bestapproximation mittels Summen von Elementartensoren in hohen Dimensionen*. PhD thesis, Universität Leipzig, 2008.

[7] Mike Espig, Wolfgang Hackbusch, Thorsten Rohwedder, and Reinhold Schneider. Variational calculus with sums of elementary tensors of fixed rank. Technical Report 52, Max Planck Institute for Mathematics in the Sciences, 2009.

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

[9] Lars Grasedyck. Hierarchical singular value decomposition of tensors. Technical Report 27, Max Planck Institute for Mathematics in the Sciences, 2009. accepted for SIMAX.

[10] Wolfgang Hackbusch. On the computation of approximate eigenvalues and eigenfunctions of elliptic operators by means of a multi-grid method. *SIAM J. Numer. Anal.*, 6(2):201–215, 1979.

[11] Wolfgang Hackbusch. *Multi-grid methods and applications*. Springer, 1985.

[12] Wolfgang Hackbusch, Boris N. Khoromskij, Stefan A. Sauter, and Eugene E. Tyrtyshnikov. Use of tensor formats in elliptic eigenvalue problems. Technical Report 78, Max Planck Institute for Mathematics in the Sciences, 2008.

[13] Wolfgang Hackbusch, Boris N. Khoromskij, and Eugene E. Tyrtyshnikov. Approximate iterations for structured matrices. *Numer. Math.*, 109(3):365–383, 2008.

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

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

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

[17] Yousef Saad. *Iterative Methods for Sparse Linear Systems, Second Edition.* SIAM, 2003.