SIAM J. SCI. COMPUT.
Vol. 34, No. 2, pp. A683–A713

THE ALTERNATING LINEAR SCHEME FOR TENSOR OPTIMIZATION IN THE TENSOR TRAIN FORMAT*

SEBASTIAN HOLTZ†, THORSTEN ROHWEDDER†, AND REINHOLD SCHNEIDER†

Abstract. Recent achievements in the field of tensor product approximation provide promising new formats for the representation of tensors in form of tree tensor networks. In contrast to the canonical r-term representation (CANDECOMP, PARAFAC), these new formats provide stable representations, while the amount of required data is only slightly larger. The tensor train (TT) format [*SIAM J. Sci. Comput.*, 33 (2011), pp. 2295–2317], a simple special case of the hierarchical Tucker format [*J. Fourier Anal. Appl.*, 5 (2009), p. 706], is a useful prototype for practical low-rank tensor representation. In this article, we show how optimization tasks can be treated in the TT format by a generalization of the well-known alternating least squares (ALS) algorithm and by a modified approach (MALS) that enables dynamical rank adaptation. A formulation of the component equations in terms of so-called retraction operators helps to show that many structural properties of the original problems transfer to the micro-iterations, giving what is to our knowledge the first stable generic algorithm for the treatment of optimization tasks in the tensor format. For the examples of linear equations and eigenvalue equations, we derive concrete working equations for the micro-iteration steps; numerical examples confirm the theoretical results concerning the stability of the TT decomposition and of ALS and MALS but also show that in some cases, high TT ranks are required during the iterative approximation of low-rank tensors, showing some potential of improvement.

Key words. density matrix renormalization group, alternating least squares, optimization problem, iterative methods for linear systems, eigenvalue problem, tensor product approximation, tensor decompositions, tensor train, high-dimensional systems, matrix product states, hierarchical tensors

AMS subject classifications. 15A69, 65K10, 90C06

DOI. 10.1137/100818893

1. Introduction.

The approximation of high-dimensional functions, often defined implicitly as the solution of, e.g., a partial differential equation, is a major challenge in numerical analysis. Using common discretization techniques, problems of this kind are usually cast into optimization problems on corresponding discrete tensor product spaces, say, $\bigotimes_{i=1}^{d} \mathbb{R}^n$, where then a linear or nonlinear equation or eigenvalue problem has to be solved. Since the dimension n^d of this space scales exponentially with d, this task is ususally prohibitively expensive for larger d (and/or n)—the well-known *curse of dimensionality*. Instead, approximating the objects under consideration by sums of tensor products of lower rank often provides a powerful tool for tackling such problems.

In the matrix case $d = 2$, the best low-rank approximation in the least squares sense is provided by what today is called the singular value decomposition (SVD). Although this result due to E. Schmidt [31] is a cornerstone of numerical mathematics, a direct optimization of matrices in low-rank form [16] has only recently gained interest for model reduction. Unfortunately, Schmidt's results, usually referred to as the Eckhardt–Young theorem [6], does not generalize to dimensions $d > 2$ except in

*Submitted to the journal's Methods and Algorithms for Scientific Computing section December 21, 2010; accepted for publication (in revised form) October 31, 2011; published electronically March 22, 2012. This work has been supported by the DFG SPP 1324, "Mathematische Methoden zur Extraktion quantifizierbarer Information aus komplexen Systemen."
http://www.siam.org/journals/sisc/34-2/81889.html

†Institute of Mathematics, TU Berlin, Straße des 17.Juni 136, Berlin, 10623 Germany (sholtz@math.tu-berlin.de, rohwedde@math.tu-berlin.de, schneidr@math.tu-berlin.de).

relatively trivial cases. Two distinct ansatz that globalize the SVD are given by

$$U = \sum_{k=1}^{r} \bigotimes_{i=1}^{d} U_{i,k} \quad \text{and} \quad U = \sum_{k_1=1}^{r_1} \dots \sum_{k_d=1}^{r_d} c_{k_1,\dots,k_d} \bigotimes_{i=1}^{d} U_{i,k_i},$$

i.e., the *canonical format* [18], based on a representation by r elementary products of singled valued functions, and the *Tucker format* [33], determining optimal subspaces of dimension r_i in each coordinate direction or mode x_i. While both parametrizations share the property of being highly nonlinear but multilinear, they are in some sense antipoles exhibiting the advantages and disadvantages one can be faced with in tensor approximation. In terms of complexity, the canonical format provides an ideal representation, with the number of required parameters depending linearly on the dimension d, the size of the single variable vector spaces n, and the number of terms r involved. In contrast to this, representation of a tensor in the Tucker format requires $r^d + nrd$ parameters assuming all $r_i = r$ are equal. Although this usually offers tremendous computational saving of parameters by a factor $\sim (r/n)^d$, it does not prevent an exponential scaling in degrees of freedom in principle. On the other hand, the treatment of even the simplest problems turns out to be hard to handle practically when approached in the canonical format. Unpredictable redundancies and instabilities lead to typical problems of nonlinear optimization like nonuniqueness, nonexistence, etc. [9], so that for the treatment of optimization tasks in canonical format [5], artificial stabilization techniques have to be utilized as side conditions [4]. In contrast to this, the Tucker format is mathematically more sound: The Tucker tensors of fixed rank being an embedded manifold [17] provides a stable parametrization, reflected in reliable practical computations, e.g., in the context of the multiconfigural time-dependent Hartree–Fock (MCTDHF) method of quantum chemistry [3, 20, 22, 23]. A perfect example for the practical experiences delineated above is the alternating least squares (ALS) method popular for the computation of best approximations in the least squares sense: Fixing all but one component at a time, the parametrization turns the above multilinear parametrizations into a series of linear problems for each component. When applied to the canonical format [1, 2, 24], ALS often shows slow or no convergence at all, in particular when higher accuracy is demanded and when the rank r is large. Although for the Tucker format the convergence properties of Tucker-ALS depend on the tensor at hand and a convergence analysis is not yet available, its convergence properties are quite satisfactory in many cases [14, 19].

A recent approach proposed by Hackbusch and Kühn [11] combines the advantages of both methods by extending the subspace approximation philosophy to a hierarchical framework, resulting in what is called the hierarchical Tucker (HT) format. Almost simultaneously, Oseledets and Tyrtyshnikov presented the similar "tree tensor" decomposition format [29] and the tensor train format (*TT format*) [25, 26, 28], the latter a powerful special case of HT and tensor tree structure due to its structural simplicity. One main advantage over the Tucker format lies in the fact that the representation of higher-order tensors is reduced to d tensors of order at most 3; in contrast to the Tucker format, the HT/TT formats are thus formally free from the curse of dimensionality. Although identical approaches have been known for decades in quantum physics, e.g., in quantum dynamics [22, 23] and in the context of matrix product states (MPS), [34] and of the density matrix renormalization group (DMRG) algorithm [32, 35], and have been generalized further to tensor networks [13, 21], these ideas are completely new in the fields of numerics and tensor optimization. Because

the practically motivated perspectives taken in quantum physics are quite different from the systematic approach taken in numerical analysis, a rigorous mathematical treatment of these new tensor representations is still in its early stage. Recently, it has been shown that a stable decomposition providing a quasi-optimal approximation in the ℓ_2-sense can be computed by hierarchical SVDs [8, 26] and that best approximations exist [7]. In a recent publication [12], we proved certain existence and uniqueness statements for TT decompositions and showed that the manifold of TT tensors of fixed rank shares many of the favorable properties of the corresponding manifold of Tucker tensors proved in [17]. Thus, the TT format combines two main advantages: On the one hand, it is stable from an algorithmic point of view; on the other, it is computationally affordable provided that the TT ranks of the tensors used stay reasonably small. Therefore, its potential for application to a broad variety of problems is very promising from the present point of view.

We take these results as a motivation to approach the practical treatment of optimization problems in the TT format by a suitable generalization of the ALS algorithm mentioned above, i.e., fixing all indices except one, thus reducing the multilinear parametrization in the TT form to a *linear* parametrization in the free index. Motivated by this fact, we will call the resulting method the *alternating linearized scheme,* again abbreviated by ALS. The ALS method for the TT format is derived rather straightforwardly. We would like to stress, however, that in the present work, we combine the ALS alorithm with a stabilization technique by QR decompositions introduced in [25], and only this combination leads to what to our knowledge is the first stable algorithm for tensor computations, while without this technique, the equations to be solved often become unstable in practice; without this step, one only obtains approximation results with single precision. We will invesigate the basic properties of our ALS method in a systematic way, and we propose a modified ALS (MALS) algorithm, inspired by the DMRG algorithm of quantum physics, which facilitates the self-adaptation of ranks either by using SVDs or by employing a greedy algorithm. A particular emphasis will then be laid on the treatment of linear equations and eigenvalue problems. Although we confine our treatment to the case of the TT format for ease of presentation, many of the methods and results in this paper generalize to the more general HT format without difficulty. Note also that in this paper, we restrict our attention mainly to practical aspects of the ALS and MALS algorithms. In subsequent publications, we examine the theoretical convergence behavior of the linearization approach being the main idea behind these methods [30] and compare it to an algorithm formulated on the full tangent space as computed in [12].

The paper is organized as follows. We start in section 2 by introducing our notation for tensors and by explicitly formulating the optimization problems to be treated; we then give a short review on the TT format. Section 3 introduces the basic ideas of the ALS and MALS methods as relaxation methods and defines so-called retraction operators to develop precise algorithmic formulations for ALS and MALS for the TT format, with which certain monotonicity statements for ALS/MALS are proved. In section 4, we take a closer look at the important applications of linear equations and the eigenvalue problem. Using the properties of the retraction operators, we find that in this case, ALS and MALS lead to a series of reduced problems of the same kind, inheriting basic features as positive definiteness and condition numbers and eigenvalues that are bounded by respective quantities of the original problems, a result that is essential for the computational stability of the ALS and MALS approach. In section 5, we turn toward implementational issues. We formulate the steps that have to be performed to assemble the component equations for ALS and MALS for linear and

eigenvalue problems and analyze the complexity of the ALS and MALS algorithm. Finally, in section 6 we report some of our practical experience with ALS and MALS when applied to some sample linear and eigenvalue equations.

2. Tensors, tensor optimization tasks, and the TT format. To set the stage, this section introduces the objects and problems subject to further treatment in the later sections. We declare some notational conventions used in this work (including the graphical representation already used in [12]) in section 2.1 and formulate the optimization tasks to be treated in section 2.2. We then review the TT representation for tensors $U \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ (section 2.3).

2.1. Notation and problems to be treated. As in [12], we will treat tensors as multivariate functions, depending on d discrete index sets

$$\mathcal{I}_i = \{1, \dots, n_i\}, \qquad i \in \{1, \dots, d\}, \qquad n_i \in \mathbb{N},$$

that is, an order-d tensor is given by

$$U : \mathcal{I}_1 \times \cdots \times \mathcal{I}_d \to \mathbb{R}, \qquad \underline{x} = (x_1, \dots, x_d) \mapsto U(x_1, \dots, x_d). \tag{2.1}$$

As in the above, we will be concerned only with real-valued discrete tensors U; the case where U maps to $\mathbb{C}$ only demands some obvious modifications. Tensors from spaces $\mathbb{R}^{n_1 \times \cdots \times n_d}$ will be denoted by capital letters $U, B, X, \dots$. For the special cases $d = 1$ and $d = 2$, corresponding to vectors and matrices, we use boldface font and the notation

$$\mathbf{v} := \big[\mathbf{v}_x\big] := \big[\mathbf{v}_x\big]_{x=1}^{n_1} \in \mathbb{R}^{n_1},$$
$$\mathbf{A} := \big[\mathbf{A}_y^x\big] := \big[\mathbf{A}_y^x\big]_{x=1,y=1}^{n_1,n_2} \in \mathbb{R}^{n_1 \times n_2}.$$

In the context of linear equations, we will sometimes use the (column) vectorization of a tensor U, denoted in the vein of the above vector notation by the according boldface lowercase letter

$$\mathbf{u} := \big[U_{x_1,\dots,x_d}\big] = \big[U_{x_1,\dots,x_d}\big]_{\pi(x_1,\dots,x_d)=1}^{n_1 \cdots n_d} \in \mathbb{R}^{n_1 \cdots n_d},$$

where the indices of $\mathbf{u}$ are determined uniquely by a suitable bijection

$$\pi : \mathcal{I}_1 \times \cdots \times \mathcal{I}_d \to \{1, \dots, n_1 \cdot \ldots \cdot n_d\}.$$

The transpose of $\mathbf{u}$, a row vector, will sometimes be denoted as $\big[U^{x_1,\dots,x_d}\big]$. We will also encounter matrifications of tensors, labeled by subsets of the indices $x_1, \dots, x_d$ of the tensor (2.1). Again choosing a suitable bijection,

$$\big[U^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}\big] \in \mathbb{R}^{(n_1 \cdots n_i) \times (n_{i+1} \cdots n_d)} \tag{2.2}$$

denotes the matrix obtained by taking $x_1, \dots, x_i$ as row indices and $x_{i+1}, \dots, x_d$ as column indices.

In this publication, we will be concerned with the solution of equations for linear operators A on the tensor space $\mathbb{R}^{n_1 \times \cdots \times n_d}$, given by tensors

$$A = A\big(x_1, \dots, x_n, y_1, \dots, y_n\big) \in \mathbb{R}^{n_1 \times \cdots \times n_d \times n_1 \times \cdots \times n_d}, \tag{2.3}$$

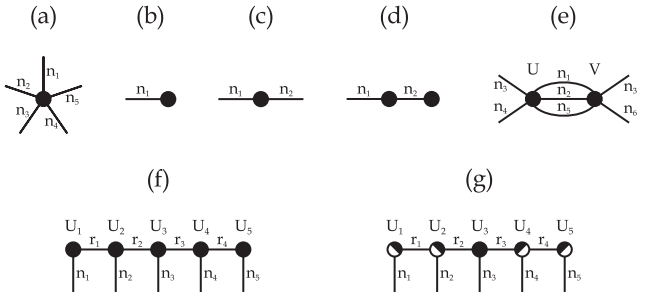

FIG. 2.1. *Examples for diagrammatic notation of tensors:* (a) *tensor of order* 5, (b) *tensor of order* 1 *(vector),* (c) *tensor of order* 2 *(matrix),* (d) *matrix-vector multiplication,* (e) *index contraction of tensors* U, V *(see text),* (f) *TT decomposition of* (a), (g) *orthonormal TT decomposition.*

and acting on $U \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ by the pointwise definition

$$(AU)(y_1, \ldots, y_n) = \sum_{x_1=1}^{n_1} \cdots \sum_{x_d=1}^{n_d} A\big(x_1, \ldots, x_n, y_1, \ldots, y_n\big) U(x_1, \ldots, x_d). \tag{2.4}$$

Using the vectorization $\mathbf{u} = [U_{x_1,\ldots,x_n}]$ of U and the matrification $\mathbf{A} = [A^{x_1,\ldots,x_n}_{y_1,\ldots,y_n}]$ of A, (2.4) can also be computed by retensorizing the column vector that results from the matrix-vector multiplication

$$[A^{x_1,\ldots,x_n}_{y_1,\ldots,y_n}][U_{x_1,\ldots,x_n}] = \mathbf{A}\mathbf{u},$$

giving a neater, equivalent representation for (2.4). As in this example, we will see that it will often be useful to skip between tensors, denoted in regular letters U, their vectorization $\mathbf{u} = \big[U_{x_1,\ldots,x_d}\big]$, and their matrifications $\big[U^{x_{i+1},\ldots,x_d}_{x_1,\ldots,x_i}\big]$.

Diagrammatic notation. Even with the above notational conventions, a precise description of the tensor objects and operations is often overburdened with the indices involved. In our opinion, the description of tensors and their computation is facilitated by the use of diagrammatic representations borrowed from physics and quantum chemistry. This notation is relatively simple, and with some experience one can describe most linear tensor operations in a very instructive way since all summations are visualized by edges of a graph. The graphical notation has been of great help in studying and implementing the tensor algorithms, and we strongly recommend this diagrammatic notation for the description of tensor manipulations.

In this notation, a tensor U of the above form (2.1) is represented by a dot with d "arms," depicting the d free variables $x_1, \ldots, x_d$. The vector and matrix cases $d = 1$ and $d = 2$ as well as the case $d = 5$ are given in Figure 2.1. Summation over common indices is symbolized by joining the repective arms of the involved tensors: Figure 2.1(d) shows the matrix-vector multiplication

$$\mathbf{A}\mathbf{v} = \big[\mathbf{A}^x_y\big]\big[\mathbf{v}_x\big],$$

while 2.1(e) is an example for the more complicated summation

$$W(x_3, x_3', x_4, x_6) = \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \sum_{x_5=1}^{n_5} U(x_1, x_2, x_3, x_4, x_5) V(x_1, x_2, x_3', x_5, x_6)$$

for U as above (with $d = 5$) and some tensor $V : \mathcal{I}_1 \times \mathcal{I}_2 \times \mathcal{I}_3 \times \mathcal{I}_5 \times \mathcal{I}_6 \to \mathbb{R}$. The diagrams 2.1(f) and (g) belong to the TT decomposition to be introduced below.

2.2. Minimization tasks on tensor spaces. We will test the ALS and MALS algorithms that are main subject of this paper on two very common problems, i.e., the solution of a linear equation

$$AU = B, \tag{2.5}$$

where $U, B \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ are tensors, A is a symmetric positive definite operator mapping $\mathbb{R}^{n_1 \times \cdots \times n_d} \to \mathbb{R}^{n_1 \times \cdots \times n_d}$, and AU is defined by (2.4), and as a second problem the eigenvalue equation

$$AU = \lambda U, \qquad U \neq 0. \tag{2.6}$$

Both (2.5) and (2.6) are special cases of the minimization problem for a given functional $\mathcal{J}$, U being the minimizer of

$$\mathcal{J} : \mathbb{R}^{n_1 \times \cdots \times n_d} \to \mathbb{R}, \quad \mathcal{J}(V) = \frac{1}{2}\langle \mathbf{A}\mathbf{v}, \mathbf{v}\rangle - \langle \mathbf{b}, \mathbf{v}\rangle \tag{2.7}$$

and

$$\mathcal{J} : \mathbb{R}^{n_1 \times \cdots \times n_d} \setminus \{0\} \to \mathbb{R}, \quad \mathcal{J}(V) = \frac{1}{2}\frac{\langle \mathbf{A}\mathbf{v}, \mathbf{v}\rangle}{\langle \mathbf{v}, \mathbf{v}\rangle}. \tag{2.8}$$

Note that in (2.7) and (2.8), we deliberately define the functional via the matrification $\mathbf{A}$ of the operator A and the vectorization $\mathbf{v}$ of the tensor V; this viewpoint will be useful later on, when the properties of the resulting vector equations are analyzed in section 4.1.

2.3. The TT decomposition of a tensor. We review the TT decomposition of tensors, emphasizing the points that are of importance for this work. For more detailed information, see [12].

Every tensor $U \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ may be represented exactly in the so-called TT format, i.e., U is decomposed into d component tensors U_i, $i = 1, \ldots, d$, of at most order 3, depending on the variable $x_i \in \mathcal{I}_i$, respectively, and on two auxiliary variables

$$k_{i-1} \in \mathcal{K}_{i-1}, \quad k_i \in \mathcal{K}_i, \quad \mathcal{K}_i := \{1, \ldots, r_i\} \text{ for } i = 1, \ldots, d-1, \quad \mathcal{K}_0 = \mathcal{K}_d = \{1\},$$

connecting the components U_{i-1} and U_i via summation over k_i. Thus,

$$U_i \in \mathcal{K}_{i-1} \times \mathcal{I}_i \times \mathcal{K}_i \to \mathbb{R}$$

for all $i \in \{1, \ldots, d\}$. The value of a tensor U at $\underline{x} = (x_1, \ldots, x_d)$ is then given by a successive matrix-vector multiplication using the component tensors,

$$U(\underline{x}) = \sum_{k_1=1}^{r_1} \cdots \sum_{k_{d-1}=1}^{r_{d-1}} U_1(1, x_1, k_1) \cdot U_2(k_1, x_2, k_2) \cdot \cdots \cdot U_d(k_{d-1}, x_d, 1). \tag{2.9}$$

The numbers r_i are the rank numbers of the TT tensor, giving the TT rank vector $\underline{r} = (r_1, \ldots, r_{d-1})$ that mainly governs the complexity of the TT representation. Note also that for notational convenience, we let $r_0 := r_d := 1$ in the above, so that for all i the values of U_i are given by $U_i(k_{i-1}, x_i, k_i)$ despite the fact that U_1 and U_d depend only on two variables (indices) while all other U_i are order-3 tensors. See also Figure 2.1(f) for a depiction of the TT decomposition (2.9).

As in (2.9), it is often more instructive to refer to the values $U(\underline{x})$ since the tensor U is therein equipped with the indices or variables $\underline{x} = (x_1, \dots, x_d)$ it depends on. This is also the case in what is probably the most comprehensive form for displaying a TT tensor, the *matrix product representation* (called matrix product state (MPS) in quantum physics). This notation makes use of the component function belonging to a component tensor $U_i(k_{i-1}, x_i, k_i)$, defined as

$$\big[(\mathbf{U}_i)_{k_{i-1}}^{k_i}(\cdot)\big] : \quad x_i \;\mapsto\; [U_i(k_{i-1}, x_i, k_i)]_{k_{i-1}=1, k_i=1}^{r_{i-1} \quad r_i} \in \mathbb{R}^{r_{i-1} \times r_i}, \tag{2.10}$$

by which the value of U at $\underline{x} = (x_1, \dots, x_d)$ can be written as

$$U(\underline{x}) = \big[(\mathbf{U}_1)^{k_1}(x_1)\big] \cdot \dots \cdot \big[(\mathbf{U}_i)_{k_{i-1}}^{k_i}(x_i)\big] \cdot \dots \cdot \big[(\mathbf{U}_d)_{k_{d-1}}(x_d)\big]. \tag{2.11}$$

Using the abbreviation

$$\big[\mathbf{U}_{k_{i-1}}^{k_i}\big] = \big[(\mathbf{U}_i)_{k_{i-1}}^{k_i}(\cdot)\big],$$

the tensor U is then represented in the shorthand notation by

$$U = \big[\mathbf{U}^{k_1}\big]\big[\mathbf{U}_{k_1}^{k_2}\big] \cdot \dots \cdot \big[\mathbf{U}_{k_{i-1}}^{k_i}\big] \cdot \dots \cdot \big[\mathbf{U}_{k_{d-2}}^{k_{d-1}}\big]\big[\mathbf{U}_{k_{d-1}}\big]. \tag{2.12}$$

The representation (2.12) is extremely compact and sometimes heavily facilitates the use of TT tensors, but it is also rather sloppy and more useful for abbreviation than for pointing to the nature of the tensor decompositions. At several points in this paper we will therefore avoid the notation (2.12) and rather use the more lengthy ones above.

Before we can define the ALS and MALS algorithms, we need a notion of orthonormality for TT tensors introduced in [25] for practical reasons and also used in [12] in a more theoretical context. For a component tensor U_i of U, its unfoldings

$$\big[U_{k_{i-1},x_i}^{k_i}\big] := \big[(U_i)_{k_{i-1},x_i}^{k_i}\big] \quad \text{and} \quad \big[U_{k_i,x_i}^{k_{i-1}}\big] := \big[(U_i)_{k_i,x_i}^{k_{i-1}}\big] \tag{2.13}$$

are called the *left and right unfoldings* of U_i, respectively. U_i is *left, respectively, right, orthonormal* iff the left, respectively, right, unfoldings are orthonormal, i.e.,

$$\big[U_{k_i}^{k_{i-1},x_i}\big]\big[U_{k_{i-1},x_i}^{k_i}\big] = \mathbf{I} \in \mathbb{R}^{r_i \times r_i}, \;\; \text{resp.}, \;\; \big[U_{k_{i-1}}^{k_i,x_i}\big]\big[U_{k_i,x_i}^{k_{i-1}}\big] = \mathbf{I} \in \mathbb{R}^{r_{i-1} \times r_{i-1}}. \tag{2.14}$$

A TT decomposition is called a *minimal decomposition* if all component functions have full left and right rank,

$$\operatorname{rank}\big[U_{k_i,x_i}^{k_{i-1}}\big] = r_{i-1}, \;\; \operatorname{rank}\big[U_{k_{i-1},x_i}^{k_i}\big] = r_i \qquad \text{for all} \;\; i = 1, \dots, d,$$

and every such decomposition is optimal in the sense that the ranks r_i equal the respective separation ranks r_i^S of U (i.e., the rank of the unfolding $[U_{x_1,\dots,x_i}^{x_{i+1},\dots,x_d}]$), while they provide a lower bound for any other TT decomposition. Such a decomposition exists and can be computed by successive SVDs [8, 25]; see [12, 25] for the proof.

We finally note for later purposes that as a straightforward globalization of Theorem 3.1(b) in [12], for fixed i, the decomposition can be chosen such that the components are left-orthonormal for $j < i$ and right-orthonormal for $j > i$,

$$\big[(U_j)_{k_j}^{k_{j-1},x_j}\big]\big[(U_j)_{k_{j-1},x_j}^{k_j}\big] = \mathbf{I} \in \mathbb{R}^{r_j \times r_j} \qquad \text{for} \;\; j < i, \tag{2.15}$$

$$\big[(U_j)_{k_{j-1}}^{k_j,x_j}\big]\big[(U_j)_{k_j,x_j}^{k_{j-1}}\big] = \mathbf{I} \in \mathbb{R}^{r_{j-1} \times r_{j-1}} \;\; \text{for} \;\; j > i, \tag{2.16}$$

while the component U_i, sometimes called the "core" of the tensor, carries the nonorthogonal share of U. In diagrammatic notation, this may be depicted as in Figure 2.1(g), where the black-and-white dots indicate left- and right-orthonormality.

3. ALS and MALS algorithms for the TT format. In this section, we devise and discuss the ALS and MALS algorithms that will be applied to the introduced optimization tasks in the later sections (section 3). In sections 3.1 and 3.2, respectively, we give for each of the two methods an introduction to the basic ideas of the two algorithms and then devise a formulation of the algorithms with the help of retraction operators to be defined below. For later purposes, we prove some properties of retraction operators in section 3.3. We then derive a strict monotonicity result for the ALS and MALS algorithms in section 3.4.

The TT format represents a tensor U as a multilinear combination of component tensors U_i depending only on a few variables (or indices). The ALS and its modification MALS discussed below are relaxation procedures, where only one, respectively, two, components U_i are optimized at a time while the other components U_j, are fixed. Such relaxation methods are the method of choice in many optimization and iterative methods, with the Gauss–Seidel iterative method for solving linear equations probably the best known example. In the treatment of multilinear tensor formats as the canonical, the Tucker, and also the TT format, relaxation methods have the attractive property that the parametrization for the component U_i subject to the momentary micro-iteration enters *linearly* into the tensor product ansatz, so that a global quadratic optimization problem, for example, then also induces quadratic optimization problems for the micro-iterations. Convergence to a minimum need not be guaranteed for such relaxation methods, and a rigorous analysis of the convergence properties of the ALS and MALS algorithms will be performed in a forthcoming publication [30]. Here, we will concentrate on the algorithmic aspects of the methods and only note that the iteration is monotonous in the sense that the values $\mathcal{J}(U^{(n)})$ decrease during the iteration; see Lemma 3.3.

3.1. ALS. The first scheme we tested for the treatment of optimization tasks is the ALS, subsequently optimizing single component functions of a TT tensor of given rank. The well-known alternating least squares algorithm that is common for approximation of full given tensors is a special case of this approach, applied to the functional (2.7) with $\mathbf{A} = \mathbf{I}$. Starting with a TT tensor (2.9) with right-orthonormal component functions $U_2, \dots, U_d$, the first micro-iteration optimizes the nonorthonormal component U_1, obtaining a component V_1. Afterward, a numerically motivated stabilization procedure is performed: The nonorthonormal part (or "core") is moved to the next component function U_2 by performing a QR decomposition

$$[V^{k_1'}_{k_0,x_1}] = [\tilde{U}^{k_1}_{k_0,x_1}][\mathbf{R}^{k_1'}_{k_1}] \tag{3.1}$$

of the left unfolding of V_1. A new, left-orthonormal component function $\tilde{U}_1$ is then determined by its left unfolding taken as $\tilde{U}^{k_1}_{k_0,x_1}$, while the second component U_2 is updated to the tensor $\tilde{U}_2$, determined by its right unfolding chosen as

$$[\tilde{U}^{x_2,k_2}_{k_1}] := [\mathbf{R}^{k_1'}_{k_1}][U^{x_2,k_2}_{k_1'}], \tag{3.2}$$

so that $\tilde{U}_2$ now carries the nonorthonormal part of the tensor. This optimization/orthogonalization procedure is now performed for $i = 2, \dots, d-1$, moving the nonorthonormal part in each step: A nonorthonormal component U_i is thus optimized at a time, while the remaining components U_j are left-orthonormal for $j < i$ and right-orthonormal for $j > i$ in each step. Note that this is an important detail of the ALS algorithm, because it ensures the stability of the method (see Theorem 4.1) for the

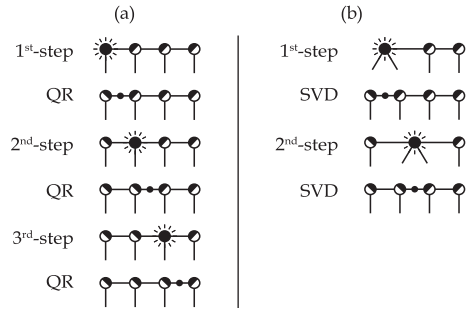

FIG. 3.1. *A "half sweep" of the ALS algorithm* (a) *and of the MALS algorithm* (b) *for* $d = 4$. *In each* i*th step, the highlighted component* U_i *(or contracted component* $U_{i,i=1}$ *in the MALS case) is optimized. Afterwards, by QR decomposition of* U_i *(resp.* $U_{i,i+1}$*), the nonorthogonal component of the tensor is moved to the next component* U_{i+1}. *Upon reaching the right-hand side of the tensor, the procedure is then repeated in the opposite direction in the second half sweep (not depicted here).*

linear and eigenvalue case. Such a "half sweep" is depicted in Figure 3.1(a). The relaxation procedure is then performed in the opposite direction (the "back sweep"), resulting on the whole in an iteration sweep for ALS.

An algorithmic formulation of the ALS scheme can be given in terms of one-component retraction operators $P_{i,1}$, defined via the current iterate $U = U^{(n)}$ of the ALS scheme. For given U, represented by a vector $\underline{U} = (U_1, \dots, U_d)$ of d component functions, and for $i \in \{1, \dots, d\}$, the operator

$$P_{i,1} \;:=\; P_{i,1,\underline{U}} : \quad \mathbb{R}^{r_{i-1} \times n_i \times r_{i+1}} \;\to\; \bigotimes_{i=1}^{d} \mathbb{R}^{n_i} \tag{3.3}$$

is defined by its action on $V_i \in \mathbb{R}^{r_{i-1} \times n_i \times r_{i+1}}$, given by

$$P_{i,1} V_i = [(\mathbf{U}_1)_{k_0}^{k_1}] \cdot \dots \cdot [(\mathbf{U}_{i-1})_{k_{i-2}}^{k_{i-1}}][(\mathbf{V}_i)_{k_{i-1}}^{k_i}][(\mathbf{U}_{i+1})_{k_i}^{k_{i+1}}] \cdot \dots \cdot [(\mathbf{U}_d)_{k_{d-1}}^{k_d}], \tag{3.4}$$

where we used the notation introduced in (2.12). Although $P_{i,1}$ depends on the component vector $\underline{U}$ representing the actual iterate, this dependency will be suppressed as long as no confusion can arise. Given a functional $\mathcal{J}$ to be optimized, the retraction operators $P_{i,1}$ can now be used to conveniently formulate the ALS relaxation scheme algorithm as given in Algorithm 1. In the case of linear equations, respectively, eigenvalue problems, the functional $\mathcal{J}$ is given by (2.7) and (2.8), respectively, and the resulting equations and properties will be analyzed in more detail in section 4 and also serve for numerical examples tested in section 6. We note that with some obvious modifications, the above ALS algorithm may also be applied for the treatment of optimization tasks in the more global hierachical Tucker format described in [11].

3.2. MALS. Although our numerical results below suggest that in general, ALS provides a stable, recommendable algorithm, it has two principal disadvantages: Besides the orthogonalization required in each micro-step to keep the method stable, ALS carries the main deficit that the ranks cannot easily be adapted to obtain a desired accuracy. We therefore propose the computationally simple MALS utilizing a greedy strategy which is only slightly more expensive than the ALS for a fixed rank provided the numbers n_i are appropriately small. This scheme is designed exclusively for the TT format and cannot be easily transferred to the general HT format without its

ALGORITHM 1. The ALS algorithm for TT optimization problems.

Require: Functional $\mathcal{J} : \mathbb{R}^{n_1 \times \cdots \times n_d} \to \mathbb{R}$,
Starting vector of component functions $(U_1, \dots, U_d)$.

Right-orthonormalize the components $U_2, \dots, U_d$.

while termination criterion is not fulfilled **do**

for $i = 1, \dots, d-1$ **do**
Compute

$$V_i = \operatorname{argmin}\{\mathcal{J} \circ P_{i,1}(V) \mid \ V \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}\}. \tag{3.5}$$

Apply the core movement steps (3.1), (3.2) for V_i, giving $\tilde{U}_i, \tilde{U}_{i+1}$.
Set $U_i, U_{i+1} \leftarrow \tilde{U}_i, \tilde{U}_{i+1}$.
end for
Repeat the above loop in reverse order.

end while

return vector of component functions $(U_1, \dots, U_d)$

efficiency deteriorating. The idea is to contract two subsequent component functions, say, U_i, U_{i+1}, into one compound component function $W \in \mathbb{R}^{r_{i-1} \times n_i \times n_{i+1} \times r_{i+1}}$,

$$W : \ (k_{i-1}, x_i, x_{i+1}, k_{i+1}) \ \mapsto \ \sum_{k_i=1}^{r_i} U_i(k_{i-1}, x_i, k_i) U_{i+1}(k_i, x_{i+1}, k_{i+1}). \tag{3.6}$$

Starting with a TT tensor U with right-orthonormal component functions $U_3, \dots, U_n$, the first micro-iteration in MALS optimizes the nonorthonormal contracted component W for $i = 1$, while leaving $U_3, \dots, U_n$ fixed. The component W is then optimized and then reseparated to a left-orthonormal component function $\tilde{U}_1 \in \mathbb{R}^{r_0 \times n_1 \times r_1}$ and a part $\tilde{U}_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}$ (see (a), (b) below for details). The resulting $\tilde{U}_1$ is used to update the component function U_1; the procedure then moves on to compute optimal two-component functions $W_{2,3}, W_{3,4}, \dots W_{d-2,d-1}$ for $i = 2, \dots, d-2$ successively in the same way, updating $U_2, \dots, U_{d-1}$ on the way, and then repeats the procedure in the opposite direction, computing $W_{d-1,d}, \dots, W_{2,3}$ and updating $U_d, \dots, U_2$. Although it is in principle possible to discard the part $\tilde{U}_{i+1}$ resulting from decomposition step of $W_{i,i+1}$ because it is optimized in the following step anyway, one can also use $\tilde{U}_{i+1}$ and the current component U_{i+2} to construct a reasonable starting guess for the optimization of the next contracted component $W_{i+1,i+2}$ along the lines of (3.6), as will be done in the algorithm below. For the (approximate) solution of the inner systems, we have two possibilities.

(a) Exact solution and truncation. As a first variant of MALS, one can directly compute the minimizer $W_{1,2}$ of the functional $\mathcal{J} \circ P_{i,2} : \mathbb{R}^{r_{i-1} \times n_i \times n_{i+1} \times r_{i+1}} \to \mathbb{R}$, then reseparate the tensor approximately afterward. This second step can be done, e.g., by a truncated SVD,

$$\big[(W_{1,2})_{k_0,x_1}^{x_2,k_2}\big] \approx \big[\widetilde{U}_{k_0,x_1}^{k_1}\big]\big[\Sigma_{k_1}^{k_1}\big]\big[\widetilde{V}_{k_1}^{k_2,x_2}\big] =: \big[(\widetilde{U}_1)_{k_0,x_1}^{k_1}\big]\big[(\widetilde{U}_2)_{k_1}^{x_2,k_2}\big], \tag{3.7}$$

with $\big[(\widetilde{U}_1)_{k_0,x_1}^{k_1}\big]$ orthonormal and $k_1 = 1, \dots, r_1$ with the truncation rank r_1 chosen

such that the error is below a certain threshold, determined in practice, for instance, by the current residual.

(b) A greedy strategy for the micro-iterations in MALS. As an alternative to the above procedure, we may also use a greedy algorithm strategy as an inner loop of the MALS algorithm. In this, the unfolding $[(W_{i,i+1})^{x_{i+1},r_{i+1}}_{r_{i-1},x_i}] \in \mathbb{R}^{(r_{i-1}n_i)\times(n_{i+1}r_{i+1})}$ of the two-compontent solution $W = W_{i,i+1}$ is approximated by TT tensors of order 2,

$$[(W_{i,i+1})^{x_{i+1},k_{i+1}}_{k_{i-1},x_i}] \approx \left[\left(\sum_{k_i=1}^{r_i} \tilde{U}_i\left((k_{i-1},x_i),k_i\right) \tilde{U}_{i+1}\left(k_i,(x_{i+1},k_{i+1})\right) \right)^{x_{i+1},k_{i+1}}_{k_{i-1},x_i} \right].$$

The rank r_i is to be adapted corresponding to a sufficient accuracy. A suitable r_i may be obtained by a greedy algorithm, computing the minimizer of $\mathcal{J} \circ P_{i,1}$ over all tensors of rank r_i, where the rank r_i is increased until the desired accuracy is reached. We note that by suitable modification of the functional under consideration, the above procedure also provides a strategy for computing the best approximation of the two-component function $[(W_{i,i+1})^{x_{i+1},r_{i+1}}_{r_{i-1},x_i}]$ in terms of norms different from the ℓ_2-norm, being of importance, for example, in the treatment of differential equations.

For linear problems with rank-1 operators A, as, for instance, the approximation problem with $A = I$, approximation with respect to the A-norm, and solution of linear systems for A, this strategy can be refined by reusing the lower-rank approximations. Let us note for sake of clarity that the following strategy can be used only for the subproblems of order $d = 2$ (= matrices) and does not generalize to higher-order tensors. Starting with $r_i = 1$ gives a rank-1 best approximation $\mathbf{T}_1$ of the unfolding $[(W_{i,i+1})^{x_{i+1},r_{i+1}}_{r_{i-1},x_i}]$ of the sought solution with respect to the A-norm, i.e., the rank-1-matrix $\mathbf{T}_1 = s_1\mathbf{x}_1\mathbf{y}_1^T$ with $\mathbf{x}_1, \mathbf{y}_1$ normalized; for $A = I$, this corresponds to the first singular value and corresponding vectors. Subtracting this approximation, i.e., letting $\mathbf{B} \leftarrow [(W_{i,i+1})^{x_{i+1},k_{i+1}}_{k_{i-1},x_i}] - \mathbf{T}_1$, the new matrix $\mathbf{B}$ can now again be approximated by rank 1, etc., until a desired accuracy is reached. The algorithm provides linearly independent components (orthogonal in the A-inner product) that can comprise a matrix $\mathbf{T} = \sum_{i=1}^{r_1} \mathbf{T}_i$. Because $d' = 2$, $n'_1 = r_{i-1}n_i$, $n'_2 = n_{i+1}r_{i+1}$, this is a rather small problem compared to the original one, causing no severe additional costs.

Since $d = 2$, further matrix decomposition techniques also provide other alternatives. We note though that this seems to be a rather juvenile field; see [16] for a possibility of low-rank approximation of matrices.

For the formulation of the MALS in terms of retraction operators, we introduce analogously to (3.4) the two-index retraction operator $P_{i,2}$ for $i = 1, \dots, d-1$,

$$P_{i,2} \; : \quad \mathbb{R}^{r_{i-1}\times n_i \times n_{i+1}\times r_{i+1}} \; \to \; \bigotimes_{i=1}^{d} \mathbb{R}^{n_i}, \tag{3.8}$$

$$\begin{aligned} P_{i,2}W_{i,i+1} = {} & [(\mathbf{U}_1)^{k_1}_{k_0}] \cdot \dots \cdot [(\mathbf{U}_{i-1})^{k_{i-1}}_{k_{i-2}}][(\mathbf{W}_{i,i+1})^{k_{i+1}}_{k_{i-1}}] \\ & [(\mathbf{U}_{i+2})^{k_{i+2}}_{k_{i+1}}] \cdot \dots \cdot [(\mathbf{U}_d)^{k_d}_{k_{d-1}}] \end{aligned} \tag{3.9}$$

in which

$$[(\mathbf{W}_{i,i+1})^{k_{i+1}}_{k_{i-1}}] : \quad (x_i, x_{i+1}) \;\; \mapsto \;\; [W(k_{i-1}, x_i, x_{i+1}, k_{i+1})]^{r_{i-1}\quad r_{i+1}}_{k_{i-1}=1,k_{i+1}=1} \in \mathbb{R}^{r_{i-1}\times r_{i+1}}$$

is the two-component function defined by W, and the tensor $P_{i,2}W \in \bigotimes_{i=1}^{d} \mathbb{R}^{n_i}$ is

built from $U_1, \dots, U_{i-1}, W, U_{i+2}, \dots, U_d$ analogously to (2.12). In terms of $P_{i,2}$ and a given functional $\mathcal{J}$, the MALS algorithm is now given in Algorithm 2.

ALGORITHM 2. The relaxation TT-MALS algorithm.

Require: Functional $\mathcal{J} : \mathbb{R}^{n_1 \times \cdots \times n_d} \to \mathbb{R}$,
Starting vector of component functions $(U_1, \dots, U_d)$.

Right-orthonormalize the components $(U_3, \dots, U_d)$.

while termination criterion is not fulfilled **do**

for $i = 1, \dots, d-1$ **do**
(a) Optimization step: Compute or approximate the minimizer

$$W_{i,i+1} : \quad (k_{i-1}, x_i, x_{i+1}, k_{i+1}) \quad \mapsto \quad W_{i,i+1}(k_{i-1}, x_i, x_{i+1}, k_{i+1})$$

of the functional

$$\{\mathcal{J} \circ P_{i,2} : \mathbb{R}^{r_{i-1} \times n_i \times n_{i+1} \times r_{i+1}} \to \mathbb{R}\}.$$

(b) Decomposition step: Represent this approximation or exact solution as product of one-component functions,

$$\big[(W_{i,i+1})^{x_{i+1},k_{i+1}}_{k_{i-1},x_i}\big] \approx \big[\tilde{U}^{k_i}_{k_{i-1},x_i}\big]\big[\tilde{V}^{x_{i+1},k_{i+1}}_{k_i}\big].$$

with low rank r_i.
(c) Update step: $U_i \leftarrow \tilde{U},\ U_{i+1} \leftarrow \tilde{V}$.
end for
Repeat the above loop in reverse order.

end while

return vector of component functions $(U_1, \dots, U_d)$

Relation between MALS and the DMRG algorithm used in quantum chemistry. Modification of ALS to obtain the MALS algorithm gives an algorithm that resembles the DMRG algorithm [32, 35] used in quantum theory. Indeed, our MALS algorithm delivers the DMRG approach when the MALS algorithm is applied to a representation of a quantum system in terms of matrix product states [34] to compute the eigenvalues of the Hamiltonian of the system. MALS may therefore also be seen as a generalization of DMRG to generic optimization problems posed for high-dimensional systems and thus may be called the "generalized DMRG." Because the terminology of DMRG is much rooted in the specific settings, viewpoints, and quantities of quantum computation, we preferred to use the notation of the MALS approach.

3.3. Properties of the retraction operator $P_{i,j}$. In the last paragraph, the retraction operator $P_{i,j}$ was used to conveniently formulate the ALS/MALS algorithm. We now define an according matrix form $\mathbf{P}_{i,j}$ of the retraction operator $P_{i,j}$, in terms of which many properties of the ALS and MALS algorithms are more conveniently analyzed, and derive some properties of $\mathbf{P}_{i,j}$ in Lemmas 3.1 and 3.2.

Denoting the vectorization operation by **vec**, we define $\mathbf{P}_{i,j}$ as a linear operator acting on the vectorization of a one-component, respectively, two-component, function V by

$$\mathbf{P}_{i,j}: \quad \mathbb{R}^m \to \mathbb{R}^{n_1 \cdots n_d}, \quad \mathbf{P}_{i,j}\mathbf{v} = \mathbf{vec}\Big(P_{i,j}\big(\mathbf{vec}^{-1}(\mathbf{v})\big)\Big), \tag{3.10}$$

where $m = r_{i-1}n_i r_i$ for $j = 1$ and $m = r_{i-1}n_i n_{i+1} r_{i+1}$ for $j = 2$. Note that then $\mathbf{P}_{i,j}\mathbf{v}$ corresponds to the vectorization of $P_{i,j}V$ for $V \in \mathbb{R}^{r_{i-1}\times n_i \times r_i}$ for $j = 1$, respectively, $V \in \mathbb{R}^{r_{i-1}\times n_i \times n_{i+1}\times r_{i+1}}$ for $j = 2$. The main useful point is that $\mathbf{P}_{i,j}$ can be expressed more explicitly than (3.10). To derive this expression, we fix U and $i \in \{1, \dots, d+1-j\}$ and define for $i \in \{2, \dots, d\}$ the ith left part of U by

$$L_i \in \mathbb{R}^{n_1\times\cdots\times n_{i-1}\times r_{i-1}}, \quad L_i(x_1, \dots, x_{i-1}, k_{i-1}) := \big[\mathbf{U}^{k_1}(x_1)\big] \cdot \cdots \cdot \big[\mathbf{U}^{k_{i-1}}_{k_{i-2}}(x_{i-1})\big]$$

and by $L_i = [1]$ for $i = 1$. The according right part we define by

$$R_i \in \mathbb{R}^{r_i\times n_{i+1}\times\cdots\times n_d}, \quad R_i(k_i, x_{i+1}, \dots, x_d) := \big[\mathbf{U}^{k_i}_{k_{i+1}}(x_{i+1})\big] \cdot \cdots \cdot \big[\mathbf{U}_{k_{d-1}}(x_d)\big].$$

Note that then

$$U = \sum_{k_{i-1}=1}^{r_{i-1}} \sum_{k_i=1}^{r_i} L_i(x_1, \dots, x_{i-1}, k_{i-1}) U_i(k_{i-1}, x_i, k_i) R_i(k_i, x_{i+1}, \dots, x_d).$$

Using the unfoldings

$$\mathbf{L}_i := [(L_i)^{k_{i-1}}_{x_1,\dots,x_{i-1}}], \quad \mathbf{R}_i := [(R_i)^{k_{i+1}}_{x_{i+1},\dots,x_d}], \tag{3.11}$$

$\mathbf{P}_{i,j}$ is now given by

$$\mathbf{P}_{i,1} := \mathbf{L}_i \otimes \mathbf{I}_{n_i} \otimes \mathbf{R}_i \in \mathbb{R}^{(n_1\cdots n_d)\times(r_{i-1}n_i r_i)} \tag{3.12}$$

in the case $j = 1$ designed for the ALS and by

$$\mathbf{P}_{i,2} := \mathbf{L}_i \otimes \mathbf{I}_{n_i} \otimes \mathbf{I}_{n_{i+1}}\mathbf{R}_{i+1} \in \mathbb{R}^{(n_1\cdots n_d)\times(r_{i-1}n_i n_{i+1} r_{i+1})} \tag{3.13}$$

in the case $j = 2$, corresponding to MALS. With this representation, it is easy to prove the following crucial property of $\mathbf{P}_{i,j}$.

LEMMA 3.1. *Let $j = 1, 2$, $U_1, \dots, U_{i-1}$ be left-orthonormal, and $U_{i+j}, \dots, U_d$ be right-orthonormal. Then $\mathbf{P}_{i,j}$ is orthonormal, that is, with $m = r_{i-1}n_i r_i$ for $j = 1$, $m = r_{i-1}n_i n_{i+1} r_{i+1}$ for $j = 2$, there holds*

$$\mathbf{P}^T_{i,j}\mathbf{P}_{i,j} = \mathbf{I} \in \mathbb{R}^{m\times m}.$$

Proof. An easy induction based on the left, respectively, right, orthonormality of the components U_i shows

$$\mathbf{L}_i^T\mathbf{L}_i = \mathbf{I} \in \mathbb{R}^{r_{i-1}\times r_{i-1}}, \quad \mathbf{R}_i\mathbf{R}_i^T = \mathbf{I} \in \mathbb{R}^{r_i\times r_i},$$

from which the result follows by the representations (3.12), (3.13) for $\mathbf{P}_{i,j}$. ∎

Later on, we will also need the following property of the retraction operators for different TT representations of a tensor U.

LEMMA 3.2. *Let* $U \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ *a tensor and*

$$U = [\mathbf{V}^{k_1}][\mathbf{V}^{k_2}_{k_1}] \cdot \cdots \cdot [\mathbf{V}_{k_{d-1}}] = [\mathbf{W}^{k_1}][\mathbf{W}^{k_2}_{k_1}] \cdot \cdots \cdot [\mathbf{W}_{k_{d-1}}]$$

two different TT representations of U *defined via two vectors of component functions*

$$\underline{V} = (V_1, \dots, V_d), \quad \underline{W} = (W_1, \dots, W_d).$$

For fixed $i \in \{1, \dots, d\}$, $j \in \{1, 2\}$, *denote by* $P_V = P_{i,j,\underline{V}}, P_W = P_{i,j,\underline{W}}$ *the retraction operators for the* i*th component as defined via* (3.4)*, respectively,* (3.10)*, by the components of* $\underline{V}, \underline{W}$*. Then there holds image*(P_V) =*image*(P_W)*.*

Proof. We prove only the case $j = 1$; the case $j = 2$ is analogous. For P_V, we use the matrix representation from (3.11),

$$\mathbf{P}_V = \mathbf{L}_V \otimes \mathbf{I}_{n_i} \otimes \mathbf{R}_V := \mathbf{L}_{i,\underline{V}} \otimes \mathbf{I}_{n_i} \otimes \mathbf{R}_{i,\underline{V}}, \tag{3.14}$$

and an analogous representation $\mathbf{P}_W = \mathbf{L}_W \otimes \mathbf{I}_{n_i} \otimes \mathbf{R}_W$ for P_W. Subsequent left-orthogonalization of the components $V_1, \dots, V_{i-1}$ gives a representation for $\mathbf{L}_V$ as $\mathbf{L}_V = \widehat{\mathbf{L}}_V \mathbf{A}$ with $\mathbf{A} \in \mathbb{R}^{r_{i-1} \times r_{i-1}}$ invertible and

$$\widehat{\mathbf{L}}_V = [(\widehat{\mathbf{V}}_1)^{k_1}][(\widehat{\mathbf{V}}_2)^{k_2}_{k_1}] \cdot \cdots \cdot [(\widehat{\mathbf{V}}_{i-1})_{k^{i-1}_{i-2}}]$$

built of left-orthogonal component functions $\widehat{V}_1, \dots, \widehat{V}_{i-1}$. Analogous to this, we can write $\mathbf{L}_W = \widehat{\mathbf{L}}_W \mathbf{B}$ with $\mathbf{B}$ invertible and $\widehat{\mathbf{L}}_W$ built of left-orthogonal component functions, and by right orthogonalization $\mathbf{R}_V = \widehat{\mathbf{R}}_V \mathbf{C}$, $\mathbf{R}_W = \widehat{\mathbf{R}}_W \mathbf{D}$, with $\mathbf{C}, \mathbf{D} \in \mathbb{R}^{r_i \times r_i}$ invertible and $\widehat{\mathbf{R}}_V, \widehat{\mathbf{R}}_W$ built of right-orthogonal component functions. Denoting the resulting left-orthogonalized, respectively, right-orthogonalized, component functions by a tilde and defining $\widehat{V}_i, \widehat{W}_i$ for the fixed index i by

$$\widehat{\mathbf{V}}_i(x) = \mathbf{A}\mathbf{V}_i(x)\mathbf{C}, \qquad \widehat{\mathbf{W}}_i(x) = \mathbf{B}\mathbf{W}_i(x)\mathbf{D},$$

we obtain

$$U = [(\widehat{\mathbf{V}}_1)^{k_1}][(\widehat{\mathbf{V}}_2)^{k_2}_{k_1}] \cdot \cdots \cdot [(\widehat{\mathbf{V}}_d)_{k_{d-1}}] = [(\widehat{\mathbf{W}}_1)^{k_1}][(\widehat{\mathbf{W}}_2)^{k_2}_{k_1}] \cdot \cdots \cdot [(\widehat{\mathbf{W}}_d)_{k_{d-1}}],$$

two TT decompositions of U in which the first $i-1$ components are left-orthogonal and the last components $i+1, \dots, d$ are right-orthogonal. Because of the full left and right ranks of $\widehat{V}_i, \widehat{W}_i$ and the left, respectively, right, orthogonality of the other components, the representations

$$[U^{x_i}_{x_1,\dots,x_{i-1},x_{i+1},\dots,x_d}] = (\widehat{\mathbf{L}}_V \otimes \widehat{\mathbf{R}}_V)[(\widehat{V}_i)^{x_i}_{r_{i-1},r_i}] = (\widehat{\mathbf{L}}_W \otimes \widehat{\mathbf{R}}_W)[(\widehat{W}_i)^{x_i}_{r_{i-1},r_i}]$$

for the unfolding $[U^{x_i}_{x_1,\dots,x_{i-1},x_{i+1},\dots,x_d}]$ correspond to two SVDs

$$[U^{x_i}_{x_1,\dots,x_{i-1},x_{i+1},\dots,x_d}] = \mathcal{U}\Sigma\mathcal{V} = \mathcal{U}'\Sigma'\mathcal{V}'$$

with $\mathcal{U}, \mathcal{U}'$ orthogonal and $\Sigma\mathcal{V} = [(\widehat{V}_i)^{x_i}_{r_{i-1},r_i}]$, $\Sigma'\mathcal{V}' = [(\widehat{W}_i)^{x_i}_{r_{i-1},r_i}]$, and we obtain that

$$\widehat{\mathbf{L}}_V = \widehat{\mathbf{L}}_W \mathbf{Q}_1, \qquad \widehat{\mathbf{R}}_V = \widehat{\mathbf{R}}_W \mathbf{Q}_2 \tag{3.15}$$

has to hold for some orthogonal $\mathbf{Q}_1 \in \mathbb{R}^{r_{i-1} \times r_{i-1}}, \mathbf{Q}_2 \in \mathbb{R}^{r_i \times r_i}$. Thus,

$$\mathbf{L}_V = \mathbf{L}_W \mathbf{B}^{-1} \mathbf{Q}_1 \mathbf{A}, \quad \mathbf{R}_V = \mathbf{R}_W \mathbf{D}^{-1} \mathbf{Q}_2 \mathbf{C}$$

so that using (3.14),

$$\mathbf{P}_V = \mathbf{P}_W \big(\mathbf{B}^{-1} \mathbf{Q}_1 \mathbf{A} \otimes \mathbf{I}_{n_i} \otimes \mathbf{D}^{-1} \mathbf{Q}_2 \mathbf{C}\big) =: \mathbf{P}_W \mathbf{E}$$

with $\mathbf{E}$ invertible, which completes the proof. ∎

3.4. Monotonicity properties of the ALS and MALS algorithms. In section 3, the ALS and MALS algorithms were defined as a procedure updating one after another the component functions U_i by successively minimizing the mini-functionals $\mathcal{M}_{i,j} := \mathcal{J} \circ P_{i,j}$. We now turn to the global properties of these algorithms and show that under certain conditions, ALS and MALS are strictly monotonous in the full tensor space unless a self-consistent stationary point of $\mathcal{J}$ with respect to certain Galerkin subspaces $G \subseteq \mathbb{R}^{n_1 \times \cdots \times n_d}$ is reached, the latter meaning that

$$\langle \mathcal{J}'(U), \delta U \rangle = 0 \quad \text{for all} \quad \delta U \in G. \tag{3.16}$$

Monotonicity in the ALS case. ALS fixes by definition a certain TT rank vector $\underline{r}' = (r_1, \ldots, r_{d-1})$, so that all iterates are at most of rank $\underline{r}'$, that is, every iterate during ALS belongs to a manifold $\mathbb{T}_{\underline{r}}$, $\underline{r} \leq \underline{r}'$, of tensors of TT rank $\underline{r}$ as analyzed in [12]. For given U of TT rank $\underline{r}$, its tangent space in $\mathbb{T}_{\underline{r}}$ is defined as

$$\mathcal{T}_U \mathbb{T}_{\underline{r}} = \big\{ \gamma'(t)|_{t=0} \big| \gamma \in C^1(]-\delta, \delta[, \mathbb{T}), \quad \gamma(t) = [\mathbf{U}^{k_1}][\mathbf{U}^{k_2}_{k_1}] \cdot \cdots \cdot [\mathbf{U}_{k_{d-1}}], \gamma(0) = U \big\}.$$

Using the abbreviation where $C_i = \mathbb{R}^{r_{i-1} \times n_i \times r_i}$, it can be represented (nonuniquely; see [12] for details) as

$$\begin{aligned} \mathcal{T}_U \mathbb{T}_{\underline{r}} &= \operatorname{span}\Big\{ [\mathbf{U}^{k_1}] \cdot \cdots \cdot [\mathbf{U}^{k_{i-1}}_{k_{i-2}}][(\mathbf{W}_i)^{k_i}_{k_{i-1}}][\mathbf{U}^{k_{i+1}}_{k_i}] \cdot \cdots \cdot [\mathbf{U}_{k_{d-1}}] \big| W_i \in C_i, i = 1, \ldots, d \Big\} \\ &= \operatorname{span}\big\{ \operatorname{image}(P_{i,1}) \;\big|\; i = 1, \ldots, d \big\}. \end{aligned} \tag{3.17}$$

We will now show that for ALS, the above strict monotonicity statement holds with $G = \mathcal{T}_U \mathbb{T}_{\underline{r}}$ in (3.16). To formulate the result, we call an inner solver used for minimization of a functional $\mathcal{M}_{i,j}$ (with $\mathcal{M}_{i,j} := \mathcal{J} \circ P_{i,j}$ for ALS/MALS) *strictly monotonous* if for a given starting guess U_i, it either returns U_i in the case that it is a stationary point of $\mathcal{M}_{i,j}$ or otherwise delivers an update $U_{i,+}$ with $\mathcal{M}_{i,j}(U_{i,+}) < \mathcal{M}_{i,j}(U_i)$. Note that this property is in particular given for the canonical treatment of linear and eigenvalue equations subject to section 4, where the inner problems are solved exactly (up to numerical accuracy). With this terminology, the following holds.

LEMMA 3.3. *Let $\mathcal{J}$ be a continuously differentiable functional, and let $\underline{U} = (U_1, \ldots, U_d) \in \times_{i=1}^d \mathbb{R}^{r_{i-1} \times n_i \times r_i}$ a vector of component functions, $\psi : \times_{i=1}^d \mathbb{R}^{r_{i-1} \times n_i \times r_i} \to \mathbb{R}^{n_1 \times \cdots \times n_d}$ the mapping taking a vector of component functions to a full TT tensor, and let $U = \psi(\underline{U})$ of TT rank $\underline{r}$.*

By definition, application of one "left to right" half sweep of an ALS optimization with respect $\mathcal{J}$ then yields an updated component vector $\underline{U}_+ = (U_{1,+}, \ldots, U_{d,+})$ with $U_{1,+}, \ldots, U_{d-1,+}$ left-orthonormal corresponding tensor $U_+ = \psi(\underline{U}_+)$. If the occurring micro-equations are solved via a strictly monotonous solver, either there holds

$$\mathcal{J}(U_+) < \mathcal{J}(U) \tag{3.18}$$

or U is a stationary point of $\mathcal{J}$ on the manifold $\mathbb{T}_{\underline{r}}$,

$$\langle \mathcal{J}'(U), \delta U \rangle = 0 \quad \text{for all} \quad \delta U \in \mathcal{T}_U \mathbb{T}_{\underline{r}}. \tag{3.19}$$

Remark. Condition (3.19) is the projection of the original stationarity condition $\mathcal{J}(U) = 0$ onto the Galerkin space $\mathcal{T}_U \mathbb{T}_{\underline{r}}$, so that ALS computes what is generally called a self-consistent solution U of the minimization problem for $\mathcal{J}$ on $\mathbb{T}_{\underline{r}}$, in agreement with the ansatz normally employed to approximately solve high-dimensional

equations on a given approximation manifold; see, e.g., [10]. Our present result leaves open if the convergence rate of the procedure may deteriorate toward 1, causing the procedure to stagnate although no self-consistency has been reached. We never observed this behavior though, and linear convergence of the ALS algorithm for approximation problems with convergence rate $q < 1$ can be shown if certain properties of the manifold $\mathbb{T}_{\underline{r}}$ and its tangent space can be verified. Because this is a rather lengthy matter of its own, this treatment is postponed to a forthcoming publication [30].

Proof. To begin, we note that by (3.17), the Galerkin condition (3.19) holds for U iff $\mathcal{J}'(U) \circ P_{i,1} = 0$ for all $i = 1, \dots, d$. Assume this is not fulfilled. Then there is a minimal $i \in \{1, \dots, d\}$ such that $\mathcal{J}'(U) \circ P_{i,1} \neq 0$. To establish a connection to the mini-functionals, a technical problem occurring is that the ith micro-iteration does not optimize $\mathcal{J}'(U) \circ P_{i,1}$ but the functional $\mathcal{J}(U) \circ P_{i,1,+}$ with the updated retraction operator

(3.20)
$$P_{i,1,+}V_i = [(\mathbf{U}_{1,+})_{k_0}^{k_1}] \cdot \dots \cdot [(\mathbf{U}_{i-1,+})_{k_{i-2}}^{k_{i-1}}][(\mathbf{V}_i)_{k_{i-1}}^{k_i}][(\mathbf{U}_{i+1})_{k_i}^{k_{i+1}}] \cdot \dots \cdot [(\mathbf{U}_d)_{k_{d-1}}^{k_d}]$$

with the left half built of the left-orthonormal updated components $U_{1,+}, \dots, U_{i-1,+}$, resulting from the subsequent QR-decompositions of $W_j = \operatorname{argmin}(\mathcal{J} \circ P_{j,1,+}), j = 1, \dots, i-1$, in the prior micro-iterations. This is resolved by noting that the component vector $\underline{W}^i = (U_{1,+}, \dots, U_{i-1,+}, V_i, U_{i+1}, \dots, U_d)$ used as input for the ith micro-iteration fulfils $\psi(\underline{W}^i) = \psi(\underline{U})$ due to the minimality of i and the strict monotonicity of the inner solvers. Lemma 3.2 therefore shows that $\operatorname{image}(P_{i,1,+}) = \operatorname{image}(P_{i,1})$; thus, we have $(\mathcal{J} \circ P_{i,1,+})'(V_i) = (\mathcal{J}'(U) \circ P_{i,1,+})(V_i) \neq 0$, the ith step outputs an update W_i for which

$$\big(\mathcal{J} \circ P_{i,1,+}\big)(W_i) < \big(\mathcal{J} \circ P_{i,1,+}\big)(V_i).$$

Due to the strict monotonicity of the inner solvers, the micro-iterations compute updates for which

$$\begin{aligned} \mathcal{J}(U) = \mathcal{J}(P_{1,1}(U_1)) &\geq \mathcal{J}(P_{1,1}(W_1)) \overset{QR}{=} \mathcal{J}(P_{2,1,+}(U_2)) \\ &\geq \mathcal{J}(P_{2,1,+}(W_2)) = \cdots \mathcal{J}(P_{i,1,+}(U_i)) > \quad \mathcal{J}(P_{i,1,+}(W_i)) \\ &= \mathcal{J}(P_{i+1,1,+}(U_{i+1})) \cdots \geq \cdots \mathcal{J}(P_{d,1,+}(W_d)) = \mathcal{J}(U_+) \end{aligned} \tag{3.21}$$

holds, and we have strict inequality in the ith step of (3.21), so that (3.18) is shown. If on the contrary, $\underline{U}$ is stationary on $T_U\mathbb{T}_{\underline{r}}$, we have $\mathcal{J}'(U) \circ P_{i,1} = 0$ for every $i = 1, \dots, d$, so that

$$\big(\mathcal{J}'(U) \circ P_{i,1,+}\big)(W_i) = \big(\mathcal{J}'(U) \circ P_{i,1,+}\big)(V_i)$$

for all i, and equality holds in (3.21). ∎

Monotonicity in the MALS case. In the MALS case, an analogous statement holds.

LEMMA 3.4. *With the notation of Lemma* 3.3 *and using a strictly monotonous inner solver, the MALS algorithm computes in one "left to right" half sweep an updated component vector* $\underline{U}_+ = (U_{1,+}, \dots, U_{d,+})$ *for which either*

$$\mathcal{J}(U_+) < \mathcal{J}(U)$$

or U *is a stationary point of* $\mathcal{J}$ *on* $\mathcal{P}_2 := \mathit{span}\{\mathit{image}(P_{i,2}) |\ i = 1 \dots, d-1\}$, *i.e.,*

$$\langle \mathcal{J}'(U), \delta U \rangle = 0 \quad \textit{for all} \quad \delta U \in \mathcal{P}_2.$$

Using Lemma 3.2, the proof is analogous to that of Lemma 3.3, so we omit it. We remark that Lemma 3.4 displays an unfortunate fact of the MALS algorithm: Although it adapts the rank numbers r_i in the respective ith iteration step, it may still happen that although $\mathcal{J}'(U) \neq 0$, the algorithm terminates if $\mathcal{J}'(U)$ is orthogonal to the momentary approximation space $\mathcal{P}_2$.

Local minima. A general problem in tensor product approximation is the existence of local minima of the respective parametrization of the representation, even if the minimum of the original functional $\mathcal{J}$ is unique. This causes severe problems, e.g., when using the canonical format, and Lemmas 3.3 and 3.4 show that the TT ansatz also may deliver stationary points with respect to the parametrization. In ongoing work, we are examining these stationary points. For the approximation problem, there usually exists an abundance of these stationary points, but it may be shown that stationary points owed to the parametrization are saddle points if a certain gap condition is fulfilled for the best approximation, so that at such points, a descent direction exists which can be used as update direction by the inner solver. This observation is also supported by our experimental experience in which no problems with local minima occurred. The theoretical background for these observations will be delivered in a forthcoming publication.

4. Treatment of linear equations and eigenvalue problems by (M)ALS. In this section, we formulate the micro-equations of the TT-ALS and MALS approaches, applied to some optimization tasks of special interest, in terms of the retraction operator from section 3.3 and derive therefrom some basic properties of the resulting micro-equations. In section 4.1, we will investigate the equations obtained from application of the ALS/MALS scheme to linear equations with symmetric positive definite system matrix. The special case of approximation problems (i.e., $\mathbf{A} = \mathbf{I}$) is treated in section (4.2), and we show that in reconstruction problems, ALS returns a TT decomposition after only one half sweep. We then sketch the modifications needed for the treatment of according eigenvalue problems in section (4.3).

4.1. ALS and MALS for linear equations. We turn our attention to the case where the functional $\mathcal{J}$ corresponds to the linear problem (2.5), in which case we have

$$\mathcal{J}(U) = \frac{1}{2}\langle \mathbf{A}\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{b}, \mathbf{u}\rangle \tag{4.1}$$

with symmetric positive definite system matrix $\mathbf{A}$ mapping $\mathbb{R}^{n_1 \cdots n_d} \to \mathbb{R}^{n_1 \cdots n_d}$ and a right-hand side $\mathbf{b} \in \mathbb{R}^{n_1 \cdots n_d}$, defined by the according tensor equation (2.5) introduced in section 2.1. We will see that the equations resulting from the optimization of a single component function in a micro-iteration step are also linear equations, inheriting basic properties of the global problems (4.1); see Theorem 4.1. Note that for the treatment in this section, the format in which the tensor quantities $\mathbf{A}, \mathbf{b}$ are given is not important.

Micro-iteration equations. For the linear problem, the ALS or MALS approach leads by definition of the algorithms to a sequence of lower-dimensional optimization tasks to be treated in the micro-iteration steps for the single components U_i, $i = 1, \ldots, d$. The according functionals $\mathcal{J} \circ P_{i,j}$ are in this case given by

$$(\mathcal{J} \circ P_{i,j})(V) \;=\; \frac{1}{2}\langle \mathbf{A}\mathbf{P}_{i,j}\mathbf{v}, \mathbf{P}_{i,j}\mathbf{v}\rangle - \langle \mathbf{P}_{i,j}\mathbf{v}, \mathbf{b}\rangle, \quad j = 1, 2, \tag{4.2}$$

In (4.2), we optimize $V \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}$ in the case $j = 1$ corresponding to the ALS and $V \in \mathbb{R}^{r_{i-1} \times n_i \times n_{i+1} \times r_{i+1}}$ for $j = 2$ giving the MALS; $P_{i,j} = P_{i,j,U}$ is defined by

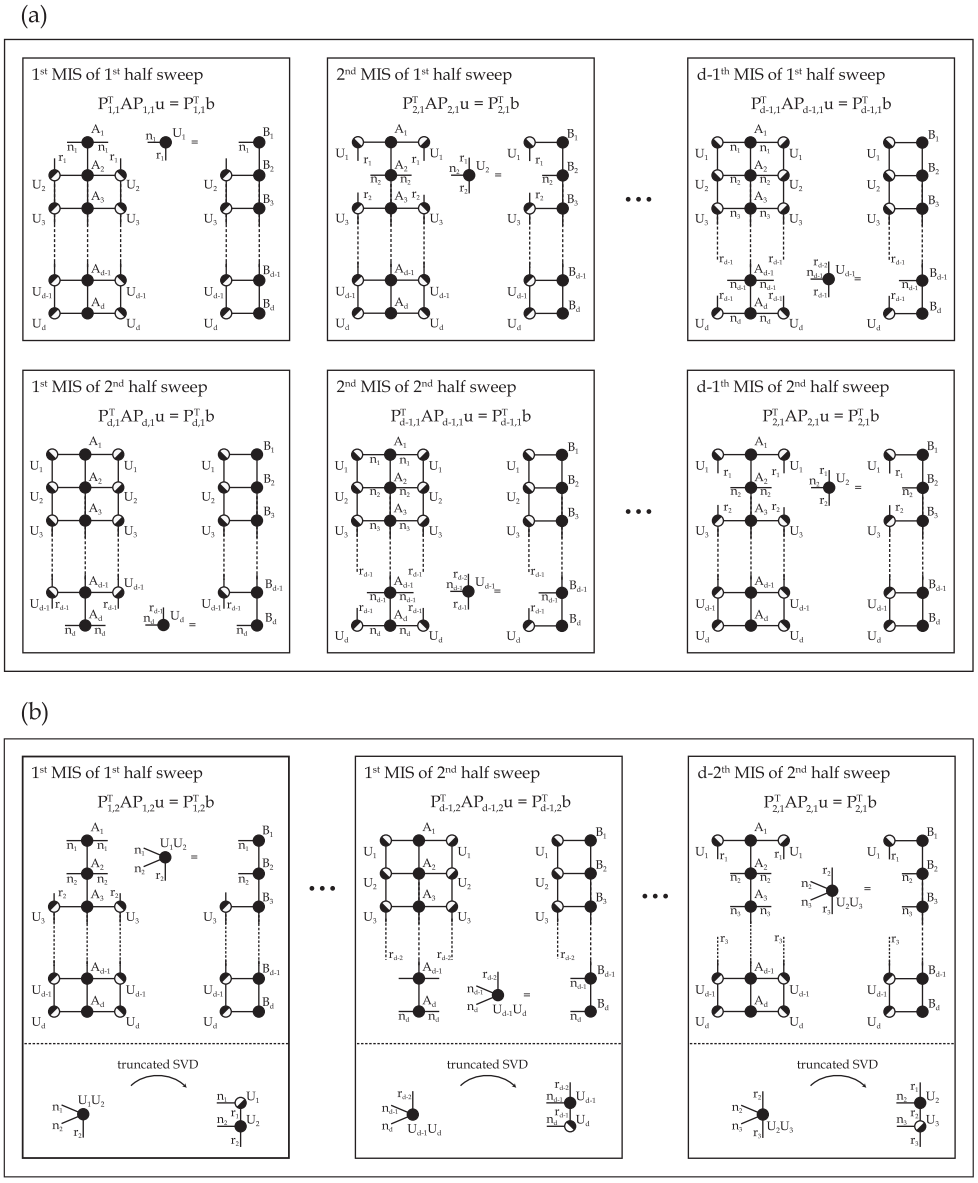

FIG. 4.1. *Some micro-iteration steps constituting the ALS* (a) *and MALS* (b) *schemes for A given in TT format.*

the current iterate U. At stationary points V_i of the functional $\mathcal{J} \circ P_{i,j}$, there holds the first-order condition

$$\mathbf{A}_i \mathbf{v}_i - \mathbf{b}_i \;:=\; \mathbf{P}_{i,j}^T \mathbf{A} \mathbf{P}_{i,j} \mathbf{v}_i - \mathbf{P}_{i,j}^T \mathbf{b} \;=\; \nabla(\mathcal{J} \circ P_{i,j})(V_i) \;=\; \mathbf{0} \tag{4.3}$$

with $D(\mathcal{J} \circ P_{i,j})V_i$ denoting the derivative of $\mathcal{J} \circ P_{i,j}$ taken at V_i, giving a linear equation to be solved in each micro-iteration. Note that $\mathbf{A}_i$ and $\mathbf{b}_i$ depend on the current macro-iterate U via $P_{i,j}$. In Figure 4.1, we depict the according course of micro-equations during ALS and MALS iterations for the case that A is given in TT format.

In the ith micro-iteration step of the ALS and the MALS, the component functions $U_1, \dots, U_{i-1}$ are by construction left-orthonormal and $U_{i+1}, \dots, U_d$ (for ALS), respectively, $U_{i+2}, \dots, U_d$ (for MALS) are right-orthonormal (cf. (2.14)). In this case, the retraction matrix $\mathbf{P}_{i,j}$ is orthogonormal by Lemma 3.1, so that the matrix $\mathbf{A}_i = \mathbf{P}_{i,j}^T \mathbf{A} \mathbf{P}_{i,j} \in \mathbb{R}^{m \times m}$ ($m = r_{i-1} n_i r_i$ for $j = 1$, $m = r_{i-1} n_i n_{i+1} r_{i+1}$ for $j = 2$ as above) directly inherits important properties of the symmetric operator $\mathbf{A}$. In the following theorem, we summarize these consequences.

THEOREM 4.1. *For A symmetric positive definite, let $\mathbf{A}_i = \mathbf{P}_{i,j}^T \mathbf{A} \mathbf{P}_{i,j}$ denote the system matrix belonging to a micro-iteration step of the ALS/MALS algorithm (where $j = 1$ and $j = 2$, respectively) with the components $U_1, \dots, U_{i-1}$ left-orthonormal and $U_{i+j}, \dots, U_d$ right-orthonormal.*

(a) *In each ith micro-iteration step of the ALS and MALS algorithm, $\mathbf{A}_i$ is symmetric positive definite. In particular, the systems* (4.3) *have a unique solution $\mathbf{v}$, defining the update $U_{i,+}$ for the ith component function.*

(b) *Let $\Lambda_{\max}, \Lambda_{\min}$ the largest und smallest eigenvalues of A and $\lambda_{\max}, \lambda_{\min}$ the largest und smallest eigenvalues of $\mathbf{A}_i$. There holds*

$$\Lambda_{\min} \;\le\; \lambda_{\min} \;\le\; \lambda_{\max} \;\le\; \Lambda_{\max}. \tag{4.4}$$

In particular, throughout the iteration, the condition numbers of the matrices $\mathbf{A}_i$ remain uniformly bounded by

$$cond\ \mathbf{A}_i \le cond\ \mathbf{A}.$$

Remark. Although it is in principle possible to choose the component functions constituting $\mathbf{P}_{i,j}$ as nonorthonormal, we will see below that then $\mathbf{P}_{i,j}$ need no longer be an orthogonal matrix. Statement (a) then still holds, but (b) is no longer valid in general. The eigenvalues of $\mathbf{A}_i$ may attain *any* positive value, and we observed the deterioration of the condition numbers of the systems (4.3) in practice. Therefore, we strongly advise the reader to maintain left-orthonormality, respectively, right-orthonormality of the component functions during the iterations. However, the orthogonality may in principle be replaced by any operator-induced orthogonality such that

$$\mathbf{P}_{i,1}^T \mathbf{A} \mathbf{P}_{i,1} = \mathbf{I}_{(r_{i-1} n_i r_i)} \quad \text{resp.,} \quad \mathbf{P}_{i,2}^T \mathbf{A} \mathbf{P}_{i,2} = \mathbf{I}_{(r_{i-1} n_i n_{i+1} r_{i+1})}.$$

Proof. Lemma 3.1 shows that $\mathbf{P}_{i,j}$ is an injection, making $\mathbf{A}_i$ symmetric positive definite and thus proving (a). Part (b) is a consequence of the Rayleigh–Ritz principle, giving with m defined in Lemma 3.1 that

$$\Lambda_{\min} = \min_{\substack{\mathbf{u} \in \mathbb{R}^{n_1 \cdots n_d} \\ \mathbf{u} \neq 0}} \frac{\langle \mathbf{A}\mathbf{u}, \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \le \min_{\substack{\mathbf{v} \in \mathbb{R}^m \\ \mathbf{v} \neq 0}} \frac{\langle \mathbf{A}\mathbf{P}_{i,j}\mathbf{v}, \mathbf{P}_{i,j}\mathbf{v} \rangle}{\langle \mathbf{P}_{i,j}\mathbf{v}, \mathbf{P}_{i,j}\mathbf{v} \rangle} = \lambda_{\min}.$$

An according estimate for $\lambda_{\max}$ completes the proof of Theorem 4.1. ∎

4.2. Approximation problems. As a special case, we consider the problem of approximating a given tensor B. Approximation in the ℓ_2-norm is the special case of the linear equation (2.5) with $\mathbf{A}$ identity operator $\mathbf{I}$. In this case, the component equations read

$$\mathbf{A}_i \mathbf{v} - \mathbf{b}_i := \mathbf{P}_{i,j}^T \mathbf{P}_{i,j} \mathbf{v} - \mathbf{P}_{i,j}^T \mathbf{b} \;=\; \mathbf{0},$$

so that for the components being left, respectively, right, orthonormal, the orthonormality of $\mathbf{P}_{i,j}$ implies the explicit formula

$$\mathbf{v} = \mathbf{P}_{i,j}^T \mathbf{b} \tag{4.5}$$

for the components of the ℓ_2-best approximation.

In some practical cases, e.g., for differential equations, approximation of a given tensor B in the ℓ_2-norm may need to be replaced by approximation in terms of some energy norm induced by an operator A. In this case, we obtain the linear system

$$\mathbf{A}_i\mathbf{v} - \mathbf{b}_i := \mathbf{P}_{i,j}^T\mathbf{A}\mathbf{P}_{i,j}\mathbf{v} - \mathbf{P}_{i,j}^T\mathbf{A}\mathbf{b} = \mathbf{0}$$

that then can be treated in the above way.

Exact TT decomposition for tensors with known TT rank. We observed in practice that when approximating a given tensor $B \in \mathbb{R}^{n_1\times\cdots\times n_d}$ of known TT rank $\underline{r}$, the ALS/ MALS only needs one half sweep for returning a TT decomposition of B if the starting guess U used in the algorithm is also of the given rank $\underline{r}$; cf. the numerical results in section 6. We now give a formal proof of this fact.

LEMMA 4.2. *Let $B \in \mathbb{R}^{n_1\times\cdots\times n_d}$ be of TT rank $\underline{r}$, let U be an inital guess given in TT format, having TT rank $\underline{r}$, and let $\mathbf{R}_i$ be the ith right part of U as defined in* (3.11). *If for all $i \in \{1,\dots,d-1\}$, the matrix*

$$[B_{x_1,\dots,x_i}^{x_{i+1},\dots,x_d}]\mathbf{R}_i^T \in \mathbb{R}^{(n_1\cdots n_{i-1})\times r_i} \tag{4.6}$$

has rank r_i, the ALS algorithm for approximation of B converges to an exact TT decomposition in one sweep when it is started with the initial guess U.

We note that the rank assumption imposed on (4.8) is introduced to avoid pathological cases, in which the ALS algorithm would return an approximation of a rank lower than $\underline{r}$. In exact arithmetic, (4.6) has rank $r_i' < r_i$ with probability 0; in practice, we never observed this situation. For a test sequence of 10,000 random initial guesses, the lowest singular value of the matrix (4.6) was 10^{-4} on average and always stayed bounded from below by 10^{-8}. The proof below shows that the result also holds for nonorthogonal initial guesses U. Nevertheless, the algorithm should start with an initial guess which is orthogonal in the last $d-1$ components to obtain stable micro-equations; see Theorem 4.1.

Proof. We denote the components of the initial guess by U_i and let $\mathbf{L}_i$ be the ith left part of U as defined in (3.11). For the tensor U_+ built up in the first half sweep, we analogously denote its components as $U_{i,+}$ and its ith left, respectively, right, part as $\mathbf{L}_{i,+}$, respectively, $\mathbf{R}_{i,+}$. The retraction operator used in the ith micro-iteration then looks like

$$\mathbf{P}_{i,1} := \mathbf{L}_{i,+} \otimes \mathbf{I}_{n_i} \otimes \mathbf{R}_i \in \mathbb{R}^{(n_1\cdots n_d)\times(r_{i-1}n_ir_i)},$$

i.e., the left part depends only on the first $i-1$ components of U_+ and the right part depends only on the last $d-i$ components of U. By (4.5), the algorithm calculates for every $i \in \{1,\dots,d\}$ in the ith microiteration step the quantity $\mathbf{v}_i = \mathbf{P}_{i,1}^T\mathbf{b}$, that is,

$$[(V_i)_{k_{i-1},x_i}^{k_i}] = (\mathbf{L}_{i,+}^T \otimes \mathbf{I}_{n_i})[B_{x_1,\dots,x_i}^{x_{i+1},\dots,x_d}]\mathbf{R}_{i,U}^T \in \mathbb{R}^{(r_{i-1}n_i)\times r_i}, \tag{4.7}$$

from which in the first $d-1$ mirco-iteration steps, the newly calculated components U_i arise by QR decomposition of V_i; see Algorithm 1. For the dth component function, we have

$$[(U_{d,+})_{r_{d-1},n_d}] = (\mathbf{L}_{d,+}^T \otimes \mathbf{I}_{n_d})[B_{x_1,\dots,x_d}] \in \mathbb{R}^{r_{d-1}n_d},$$

or, in other words,

$$\mathbf{R}_{d-1,+} = [(U_{d,+})_{r_{d-1}}^{n_d}] = \mathbf{L}_{d,+}^T[B_{x_1,\dots,x_{d-1}}^{x_d}] \in \mathbb{R}^{r_{d-1}\times n_d}.$$

This means that

$$[(U_+)^{x_d}_{x_1,\dots,x_{d-1}}] = \mathbf{L}_{d,+}\mathbf{R}_{d-1,+} = \mathbf{L}_{d,+}\mathbf{L}_{d,+}^T[B^{x_d}_{x_1,\dots,x_{d-1}}].$$

We now prove that $\mathbf{L}_{d,+}\mathbf{L}_{d,+}^T$ is a projector on the range of $[B^{x_d}_{x_1,\dots,x_{d-1}}]$; then, $U_+ = B$ holds, and the proof is finished. We proceed by induction, starting by showing that $\mathbf{L}_{2,+}\mathbf{L}_{2,+}^T \in \mathbb{R}^{n_1\times n_1}$ projects onto range$([B^{x_2,\dots,x_d}_{x_1}])$. From V_1, we obtain $\mathbf{U}_{1,+}$ by QR decomposition (here denoted by QS),

$$\begin{aligned}[(V_1)^{k_1}_{x_1}] = [B^{x_2,\dots,x_d}_{x_1}]\mathbf{R}_2^T = \underbrace{\mathbf{Q}_1}_{\in\mathbb{R}^{n_1\times r_1}} \cdot \underbrace{\mathbf{S}_1}_{\in\mathbb{R}^{r_1\times r_1}}\\ \iff \qquad [(U_{1,+})^{r_1}_{x_1}] := \mathbf{Q}_1 = [B^{x_2,\dots,x_d}_{x_1}]\mathbf{R}_2^T\mathbf{S}_1^{-1}.\end{aligned}$$

Note that $\mathbf{S}_1$ is of full rank by the rank condition for (4.6). Because $\mathbf{L}_{2,+} = \mathbf{U}_{1,+}$, $\mathbf{L}_{2,+}$ is orthonormal with rank r_1 and its image lies in the image of $([B^{x_2,\dots,x_d}_{x_1}])$, $\mathbf{L}_{2,+}\mathbf{L}_{2,+}^T$ is a projector onto the image of $([B^{x_2,\dots,x_d}_{x_1}])$. To prove the induction step, we decompose $[(V_i)^{k_i}_{k_{i-1},x_i}] = \mathbf{Q}_i\mathbf{S}_i$ with orthonormal $\mathbf{Q}_i \in \mathbb{R}^{(r_{i-1}n_i)\times r_i}$ and obtain

$$[(U_{i,+})^{k_i}_{k_{i-1},x_i}] := \mathbf{Q}_i = (\mathbf{L}_{i,+}^T \otimes \mathbf{I}_{n_i})[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1}.$$

It follows directly from the definition of L_i and the induction hypothesis that

$$\begin{aligned}\mathbf{L}_{i+1,+} &= (\mathbf{L}_{i,+}\otimes\mathbf{I}_{n_i})[(U_{i,+})^{k_i}_{k_{i-1},x_i}]\\ &= (\mathbf{L}_{i,+}\otimes\mathbf{I}_{n_i})(\mathbf{L}_{i,+}^T\otimes\mathbf{I}_{n_i})[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1}\\ &= (\mathbf{L}_{i,+}\mathbf{L}_{i,+}^T\otimes\mathbf{I}_{n_i})[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1}\\ \iff [(L_{i,+})^{x_i,k_i}_{x_1,\dots,x_{i-1}}] &= \underbrace{\mathbf{L}_{i,+}\mathbf{L}_{i,+}^T}_{\text{proj. on image } [B^{x_i,\dots,x_d}_{x_1,\dots,x_{i-1}}]}[B^{x_i,\dots,x_d}_{x_1,\dots,x_{i-1}}](\mathbf{I}_{n_i}\otimes\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1})\\ &= [B^{x_i,\dots,x_d}_{x_1,\dots,x_{i-1}}](\mathbf{I}_{n_i}\otimes\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1})\\ \iff \mathbf{L}_{i+1,+} &= \mathbf{I}_{n_1\cdot\dots\cdot n_i}[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]\mathbf{R}_{i+1,U}^T\mathbf{S}_i^{-1} = [B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]\mathbf{R}_{i+1}^T\mathbf{S}_i^{-1}.\end{aligned}$$

With the same arguments as before, $\mathbf{L}_{i+1,+}\mathbf{L}_{i+1,+}^T$ is a projector on the range of $[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}]$. Thus, $\mathbf{L}_{d,+}\mathbf{L}_{d,+}^T$ is a projector on the image of $[B^{x_d}_{x_1,\dots,x_{d-1}}]$ which shows the statement. ∎

For the MALS case, an analogous statement holds. For the proof, we assume that the MALS inner equations for the components of U_i are solved exactly, then decomposed by an SVD.

LEMMA 4.3. *Let U, B be as in Lemma 4.2. If for all $i \in \{2,\dots,d-1\}$, the matrix*

$$[B^{x_{i+1},\dots,x_d}_{x_1,\dots,x_i}](\mathbf{I}_{n_i}\otimes\mathbf{R}_{i+1}^T) \in \mathbb{R}^{(n_1\cdots n_{i-1})\times r_i} \tag{4.8}$$

has rank r_i, the MALS algorithm for approximation of B with U as a starting guess converges in one half sweep.

Proof. The proof is analogous to the ALS case. Here, the SVD, which substitutes the QR decomposition, yields components $\mathbf{U}_{i,+}$ of rank r_i because of the rank condition (4.8), $\mathbf{L}_{i,+}\mathbf{L}_{i,+}^T$ projecting on the span of $[B^{x_i,\dots,x_d}_{x_1,\dots,x_{i-1}}]$. ∎

4.3. Eigenvalue equations. The eigenvalue problem can be treated in a similar fashion as the above linear problems. To compute the smallest, respectively, largest eigenvalue of a symmetric matrix $\mathbf{A}$, we can minimize, respectively, maximize, the functional

$$\mathcal{J}(U) = \frac{1}{2}\frac{\langle \mathbf{A}\mathbf{u}, \mathbf{u}\rangle}{\langle \mathbf{u}, \mathbf{u}\rangle}, \qquad \mathbf{u} \neq 0. \tag{4.9}$$

A short computation using the orthogonality of $\mathbf{P}_{i,j}$ yields the first-order condition

$$\begin{aligned}\nabla(\mathcal{J} \circ P_{i,j})(V) &= \frac{1}{\|\mathbf{v}\|}\left(\mathbf{P}_{i,j}^T\mathbf{A}\mathbf{P}_{i,j}\mathbf{v} - \frac{\langle \mathbf{P}_{i,j}^T\mathbf{A}\mathbf{P}_{i,j}\mathbf{v}, \mathbf{v}\rangle}{\langle \mathbf{v}, \mathbf{v}\rangle}\mathbf{P}_{i,j}^T\mathbf{P}_{i,j}\mathbf{v}\right)\\ &= \frac{1}{\|\mathbf{v}\|}\left(\mathbf{A}_i\mathbf{v} - \lambda_{\mathbf{v}}\mathbf{v}\right),\end{aligned}$$

where $\lambda_{\mathbf{v}}$ is the Rayleigh quotient of $\mathbf{v}$ with respect to the matrix $\mathbf{A}_i$. Note that the last line holds only if $\mathbf{P}_{i,j}$ is orthogonal (as is the case for (M)ALS); otherwise the component equations are generalized eigenproblems. If $\mathbf{P}_{i,j}$ is isometric as in the ALS/MALS presented here, a normalized solution from the mircoiteration provides a normalized solution for the original problem.

5. Assembly of the micro-equations for linear and eigenvalue equations, complexity of (M)ALS. In this section we turn to practical aspects of the methods derived for linear systems and eigenvalue problems in section 4. To treat those problems, the ith micro-iteration of a sweep requires the application of the matrix $\mathbf{A}_i := \mathbf{P}_{i,j}^T\mathbf{A}\mathbf{P}_{i,j}$ to $\mathbf{u}$ and computation of the quantity $\mathbf{P}_{i,j}^T\mathbf{b}$ (where $\mathbf{b} = \mathbf{u}$ in the eigenvalue case). We will start this section by showing how in the ALS case these quantities may be computed with reasonable complexity. We then describe the modifications needed in the MALS case, including a discussion of truncation strategies for the separation step utilized in the MALS. We then give some concrete numerical examples.

In the following, we assume that the operator A defined by (2.3) is given in TT format, that is,

$$A\big(x_1, \dots, x_n, y_1, \dots, y_n\big) = \sum_{j_0=1}^{R_0} \dots \sum_{j_d=1}^{R_d} \widehat{A}_1(j_0, x_1, y_1, j_1) \cdot \dots \cdot \widehat{A}_d(j_{d-1}, x_d, y_d, j_d)$$

with $R_0 = R_d = 1$ as in the case of a normal tensor and with $R_1, \dots, R_{d-1}$ defining the TT rank of A. Also, we assume that the right-hand-side B is in TT format. If this is not the case, A and B can be decomposed beforehand by the approximation methods from section 4.1. Note that alternatively, we may also use a tensor A given in canonical format—this then corresponds to a sum of rank-1 TT decompositions, and the procedure given below may then be applied to each summand.

5.1. Computation of matrices for the component equations of ALS. We first treat the ALS case and give the explicit formulas for the computational steps that have to be taken. Unfortunately, the tensor operations to be performed naturally involve an abundance of different indices; we will therefore, along with the formulas, again use the diagrammatic notation that facilitates understanding of the respective operations considerably.

For computation of the matrix $\mathbf{A}_i$, we have to multiply the corresponding matrix $\mathbf{A}$ with the matrix form of the retraction operator $\mathbf{P}_{i,1} \in \mathbb{R}^{(n_1 \cdot \dots \cdot n_d)\times(r_{i-1}n_i r_i)}$.

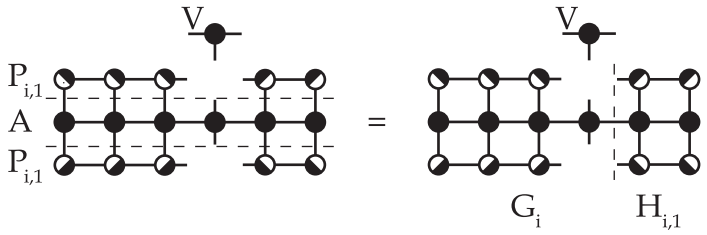

FIG. 5.1. *The tensors G_i and $H_{i,1}$ used for the application of $\mathbf{A}_i$ in the ALS algorithm and their application to a one-component iterate V.*

For notational convenience, we write the matrix $\mathbf{P}_{i,1} = [(\mathbf{P}_{i,1})_y^z]$, for which $y \in \{1, \dots, n_1 n_2 \cdot \dots \cdot n_d\}$ and $z \in \{1, \dots, r_{i-1} n_i r_i\}$, as the unfolding of a tensor

$$[(P_{i,1})_{x_1,\dots,x_d}^{k_{i-1},x_i',k_i}] \simeq [(\mathbf{P}_{i,1})_y^z]$$

with a corresponding d-tuple $(x_1, \dots, x_d) \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_d$ and a corresponding triple $(k_{i-1}, x_i', k_i) \in \mathcal{K}_{i-1} \times \mathcal{I}_i \times \mathcal{K}_i$, connected to y, z by fixed canonical bijections.

The pointwise entries $(P_{i,1})_{x_1,\dots,x_d}^{k_{i-1},x_i',k_i}$ of $[(P_{i,1})_{x_1,\dots,x_d}^{k_{i-1},x_i',k_i}]$ are then given by

$$\begin{aligned}(P_{i,1})_{x_1,\dots,x_d}^{k_{i-1},x_i',k_i} &= L_i(x_1, \dots, x_{i-1}, k_{i-1}) \cdot \delta_{x_i,x_i'} \cdot R_i(x_{i+1}, \dots, x_d, k_i) \\ &= \left(\sum_{k_1=1}^{r_1} \dots \sum_{k_{i-2}=1}^{r_{i-1}} U_1(x_1, k_1) \cdot \dots \cdot U_{i-1}(k_{i-2}, x_{i-1}, k_{i-1}) \right) \cdot \delta_{x_i,x_i'} \\ &\quad \cdot \left(\sum_{k_{i+1}=1}^{r_{i+1}} \dots \sum_{k_{d-1}=1}^{r_{d-1}} U_{i+1}(k_i, x_{i+1}, k_{i+1}) \cdot \dots \cdot U_d(k_{d-1}, x_d) \right). \end{aligned} \tag{5.1}$$

Writing $\mathbf{A}_i = \mathbf{P}_{i,1}^T \mathbf{A} \mathbf{P}_{i,1} \in \mathbb{R}^{(r_{i-1} n_i r_i) \times (r_{i-1} n_i r_i)}$ as an unfolding

$$[(A_i)_{k_{i-1}',y_i,k_i'}^{k_{i-1},x_i,k_i}] \simeq \mathbf{A}_i,$$

$\mathbf{A}_i$ can be computed by performing pointwise the summation (i.e., matrix multiplication)

$$(A_i)_{k_{i-1}',y_i,k_i'}^{k_{i-1},x_i,k_i} = \sum_{x_1=1}^{n_1} \dots \sum_{x_d=1}^{n_d} \sum_{y_1=1}^{n_1} \dots \sum_{y_d=1}^{n_d} (P_{i,1})_{k_{i-1}',y_i,k_i'}^{y_1,\dots,y_d} A_{y_1,\dots,y_d}^{x_1,\dots,x_d} (P_{i,1})_{x_1,\dots,x_d}^{k_{i-1},x_i,k_i}$$

as depicted in the left diagram of Figure 5.1; the indices to be summed over are indicated by the dashed lines. This in-principle high-dimensional summation can be split up in a sensible way by using the component matrices of A and U. We split the assembly of $\mathbf{A}_i$ into that of a left half G_i and a right half $H_{i,1}$ (with the index 1 denoting the ALS case), as seen in the right picture in Figure 5.1. G_i and $H_{i,1}$ can be computed recursively. We define $G_1 := \widehat{A}_1 \in \mathbb{R}^{n_1 \times n_1 \times R_1}$ and $G_{i+1} \in \mathbb{R}^{r_{i-1} \times r_{i-1} \times n_i \times n_i \times R_1}$ by the formula

$$\begin{aligned} &G_{i+1}(k_i, k_i', x_{i+1}, y_{i+1}, j_{i+1}) \\ &\quad = \sum_{k_{i-1}=1}^{r_{i-1}} \sum_{k_{i-1}'=1}^{r_{i-1}} \sum_{x_i=1}^{n_i} \sum_{y_i=1}^{n_i} \sum_{j_i=1}^{R_i} G_i(k_{i-1}, k_{i-1}', x_i, y_i, j_i) \cdot U_i(k_{i-1}, x_i, k_i) \\ &\qquad \cdot \widehat{A}_{i+1}(j_i, x_{i+1}, y_{i+1}, j_{i+1}) \cdot U_i(k_{i-1}', y_i, k_i'). \end{aligned}$$

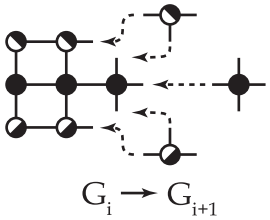

FIG. 5.2. *The update step for the left part G_i of $\mathbf{A}_i$ in the ALS algorithm.*

Note that G_{i+1} may be computed from G_i by a sequence of three matrix multiplications of rather low complexity, e.g., by computing

$$[(M_1)^{k_i}_{k'_{i-1},y_i,j_i}] := [(G_i)^{k_{i-1},x_i}_{k'_{i-1},y_i,j_i}][(U_i)^{k_i}_{k_{i-1},x_i}], \quad [(M_2)^{k'_i}_{j_i,k_i}] := [(M_1)^{k'_{i-1},y_i}_{j_i,k_i}][(U_i)^{k'_i}_{k'_{i-1},y_i}]$$
$$[(G_{i+1})^{x_{i+1},y_{i+1},j_{i+1}}_{k_i,k'_i}] = [(M_2)^{j_i}_{k_i,k'_i}][(\widehat{A}_{i+1})^{x_{i+1},y_{i+1},j_{i+1}}_{j_i}].$$

To determine the best order of the contractions, a rule of thumb is to start with the index having the largest rank r_i, R_i, n_i, so that this index turns up only once in a summation. See also Figure 5.2 for a graphical respresentation of the update step. The other way around, we let $H_{d,1} := (1) \in \mathbb{R}^{r_d \times r_d \times R_d} = \mathbb{R}$ and define $H_{i-1,1} \in \mathbb{R}^{r_{i-1} \times r_{i-1} \times R_{i-1}}$ by

$$H_{i-1,1}(k_{i-1}, k'_{i-1}, j_{i-1}) = \sum_{k_{i-1}=1}^{r_{i-1}} \sum_{k'_{i-1}=1}^{r_{i-1}} \sum_{x_i=1}^{n_i} \sum_{y_i=1}^{n_i} \sum_{j_i=1}^{R_i} H_{i,1}(k_i, k'_i, j_i)$$
$$\cdot U_i(k_{i-1}, x_i, k_i) \cdot \widehat{A}_i(j_{i-1}, x_i, y_i, j_i) \cdot U_i(k'_{i-1}, y_i, k'_i),$$

computable in terms of matrix multiplications, e.g., as

$$[(N_1)^{k_{i-1},x_i}_{k'_i,j_i}] := [(H_{i,1})^{k_i}_{k'_i,j_i}][(U_i)^{k_{i-1},x_i}_{k_i}], \quad [(N_2)^{j_{i-1},y_i}_{k'_i,k_{i-1}}] := [(N_1)^{j_i,x_i}_{k'_i,k_{i-1}}][(\widehat{A}_i)^{j_{i-1},y_i}_{j_i,x_i}]$$
$$[(H_{i-1,1}))^{k'_{i-1}}_{j_{i-1},k_{i-1}} = [(N_2)^{y_i,k'_i}_{j_{i-1},k_{i-1}}][(U_i)^{k'_{i-1}}_{y_i,k'_i}].$$

Before performing the first half sweep (from left to right; cf. section 3), the quantities $H_{d,1}, H_{d-1,1}, \ldots$ have to be computed to obtain $H_{1,1}$. If enough storage is available, the $H_{i,1}$ should be stored and can then be used in the following micro-iterations $i = 2, \ldots, d$ of the first half sweep. In contrast, G_i carries the new iterates $U_{1,+}, \ldots, U_{i-1,+}$ and therefore has to be computed between the micro-iteration steps $i-1$ and i. Storing these G_i computed from G_{i-1} during the first half sweep, they may be used in the second half sweep (from right to left), while this time, the tensors $H_{i,1}$ have to be updated and should be saved for use in the third half sweep, etc.

G_i and H_i computed, the evaluation of $\mathbf{A}_i\mathbf{v}$ can now be performed by first calculating an intermediate tensor K_i by determining the matrix product

$$[(K_i)^{k_i}_{k'_{i-1},y_i,j_i}] := [(G_i)^{k_{i-1},x_i}_{k'_{i-1},y_i,j_i}][(V_i)^{k_i}_{k_{i-1},x_i}], \tag{5.2}$$

then resorting K_i and applying H_i,

$$[(A_i v)^{k'_i}_{k'_{i-1},y_i}] = [(K_i)^{j_i,k_i}_{k'_{i-1},y_i}][(H_i)^{k'_i}_{j_i,k_i}]. \tag{5.3}$$

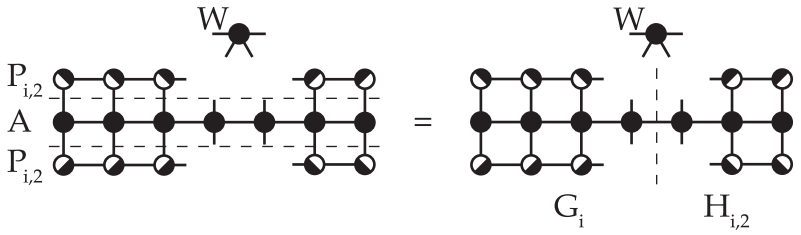

FIG. 5.3. *The tensors G_i and $H_{i,2}$ used for the application of $\mathbf{A}_i$ in the MALS algorithm and their application to a two-component iterate W.*

5.2. Complexity of the ALS scheme. To give complexity estimates for the ALS iteration, we denote by r the maximum of the ranks r_i, s_i of U and B (with $U = B$ in the eigenproblem case) and by R, depending on how A is given, the maximal rank of the TT decomposition of A or the canonical rank of A.

Concerning *storage needs*, storage of U, B clearly consumes $\mathcal{O}(r^2nd)$; for A, we have $\mathcal{O}(R^2n^2d)$ in TT format and $\mathcal{O}(Rn^2d)$ in canonical format, respectively. Additionally, the intermediate quantities G_i, $H_{i,1}$ introduced above have to be stored, each being of $\mathcal{O}(r^2Rn^2)$. Thus, the total storage requirement is $\mathcal{O}(dr^2R^2n^2)$.

Turning to the *computational steps*, we prepare the micro-iterations by computing in one half sweep the tensors G_i, $2 \le i \le d$, respectively, $H_{i,1}$, $1 \le i \le d-1$, i.e., a workload of $\mathcal{O}(r^3R^2n^2(d-1))$ for the TT format and as $\mathcal{O}(r^3Rn^2(d-1))$ in the canoncial format, and by computing the matrices $\mathbf{P}_{i,1}^T\mathbf{b}$, being of $\mathcal{O}(r^3nd)$ on the whole. During the micro-iterations, application of $\mathbf{G}_i$ and $\mathbf{H}_{i,j}$ to $\mathbf{v}$ has to be performed to obtain $\mathbf{A}_i\mathbf{v}$; here, we obtain $\mathcal{O}(r^3Rn^2)$ for the ALS with TT operator A as well as for ALS with canonical operator. Multiplying these quantities by the number of micro-iterations d, one sees that the dominant steps of ALS are of $\mathcal{O}(r^3R^2n^2d)$ for TT operators and of $\mathcal{O}(r^3Rn^2d)$ for canonical operators if the systems are solved approximately, while exact solution of the micro-systems would require $\mathcal{O}(r^6n^3)$ flops with Gauss elimination. The following QR step requires $\mathcal{O}(r^3n)$ flops. Note that the estimates are for the case that the matrices $\widehat{A}_i(k_{i-1}, \cdot, \cdot, k_i)$ are dense; in the sparse case, the n^2-behavior can be reduced to be linear in n.

5.3. The MALS scheme: Modifications and complexity. The implementation of the MALS scheme in principle corresponds to that of the ALS scheme; we therefore only discuss the main differences. As in the ALS case, the assembly of $\mathbf{A}_i$ is performed by computing a left half G_i and a right half $H_{i,2}$ (with the index 2 denoting the MALS case); see Figure 5.3. While G_i is identical to the ALS case, the matrices $H_{i,2}$ are computable by recursion via defining $H_{d-1,2} := \widehat{A}_d \in \mathbb{R}^{R_{d-1}\times n_d\times n_d}$ and $H_{i-1,2} \in \mathbb{R}^{r_{i-1}\times r_{i-1}\times n_i\times n_i\times R_1}$ by

$$[(N_1)^{k_i}_{k'_{i+1},j_i,y_{i+1}}] := [(H_{i,2})^{k_{i+1},x_{i+1}}_{k'_{i+1},y_{i+1},j_i}][(U_{i+1})^{k_i}_{k_{i+1},x_{i+1}}],$$

$$[(N_2)^{k'_i}_{j_i,k_i}] := [(N_1)^{k'_{i+1},y_{i+1}}_{j_i,k_i}][(U_i)^{k'_i}_{k'_{i+1},y_{i+1}}]$$

$$[(H_{i-1,2})^{x_i,y_i,j_{i-1}}_{k_i,k'_i}] = [(M_2)^{j_i}_{k_i,k'_i}][(\widehat{A}_{i+1})^{x_i,y_i,j_{i-1}}_{j_i}].$$

Using G_i and $H_{i,2}$, $\mathbf{A}_i\mathbf{v}$ can now be evaluated analogously to (5.2), (5.3).

Concerning complexity, one easily sees that storage needs are of the same order as for ALS, i.e., $\mathcal{O}(dr^2R^2n^2)$. The assemblies of G_i, $H_{i,2}$, $1 \leq i \leq d-2$ and of $\mathbf{P}_{i,2}^T\mathbf{b}$ are steps of $\mathcal{O}(r^3R^2n^2(d-2))$ for the TT format and of $\mathcal{O}(r^3Rn^2(d-2))$ in the canoncial format, as for the ALS case. Application of G_i and $H_{i,2}$ to $W_i \in \mathbb{R}^{r_{i-1}\times n_i \times n_{i+1}\times r_{i+1}}$ is a step of $\mathcal{O}(Rr^3n^3)$ for both TT and canonical operator, so that the whole computational complexity scales as $\mathcal{O}(r^3R^2n^3d)$ for iterative solvers in contrast to $\mathcal{O}(r^6n^6)$ flops for exact solution by Gauss elimination. The following SVD step requires $\mathcal{O}(2r^3n^3)$ flops.

6. Numerical examples. In this section, we present some sample numerical experiments using the ALS and MALS algorithm, displaying how the algorithms behave in generic situations. All results presented here were computed on an AMD 2.6 GHz with 4 cores and 32 GB RAM with MATLAB 7.9, using built-in functions and some routines from the TT Toolbox [27].

6.1. Approximation with ALS. Our first example, a "reconstruction test," consisted of the approximation of tensors B of known TT rank by TT tensors of same or higher ranks, using the ALS as described in section 4.1. In all experiments, a single half sweep of ALS returned a tensor that was a numerically exact approximation of B, delivering the numerical counterpart of our earlier Lemma 4.2. A sample calculation is displayed in Figure 6.1(a). In this, the TT tensor B to be approximated by ALS was constructed by using the quantized TT approach [15] to turn a tridiagonal matrix of the size $2^d \times 2^d$ (corresponding to a finite difference approximation of the second derivative) to $U \in \mathbb{R}^{(4^d)}$ of order $d = 7$ and of TT ranks $r_i = 3$. For d varying up to $d = 50$ in our calculation, the approximation algorithm always converged after about 25 sweeps (with stagnation of the relative error of U as termination criterion). We also observed this for analogous discretizations of the p-dimensional Laplacian, which in quantized TT approach gives a representation as tensor of order $d \cdot p$. We noted before that a TT approximation is unique up to insertion of orthogonal $r_i \times r_i$-matrices (cf. also [12, Theorem 3.1]), and this reflects the observation that in this case, the components U_i still changed after convergence of the error, while the resulting TT approximation U of B stayed constant.

A peculiar detail we observed is that during the first micro-iteration steps, the errors do not decrease much; nearly all error improvement in a step is gained in the last step of the half sweep. This effect also occurs in the situation where the approximation ranks are chosen lower than that of the original tensors. Because in this case, more than a half sweep is needed to calculate sufficient precision, this typically results in plateaus in the according convergence curves as displayed in Figure 6.1(b).

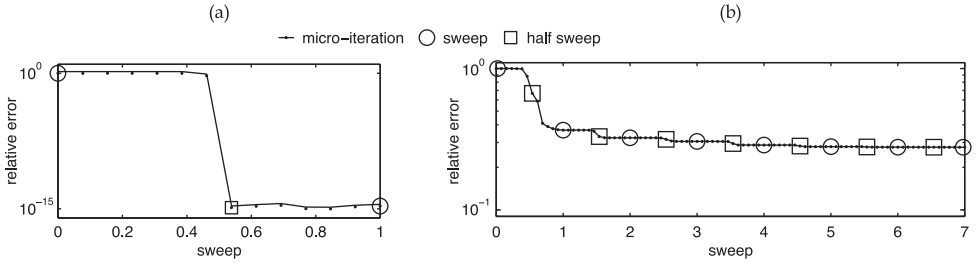

FIG. 6.1. *ALS approximation of the one-dimensional Laplacian, given as an order-*7 *quantized tensor, with TT tensors of constant rank* $r_i = 3$ (a) *and* $r_i = 2$ (b).

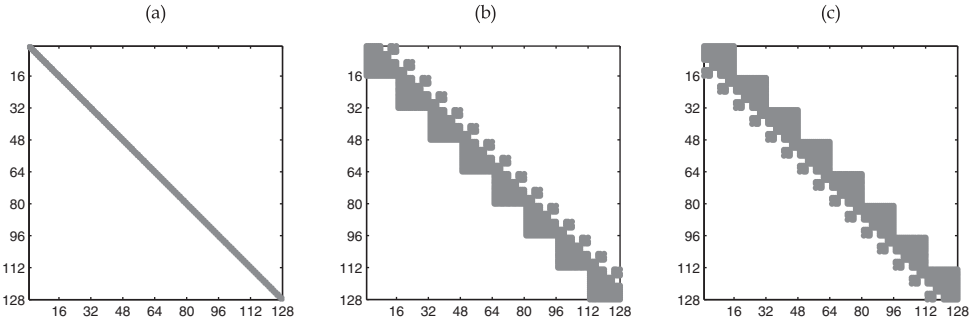

FIG. 6.2. *TT rank-2 ALS approximations of the discrete second derivative:* (a) *original matrix;* (b), (c) *two different approximations, obtained from different starting values. Gray dots represent elements with modulus greater than* 10^{-16} *relative to the Frobenius norm of the matrix.*

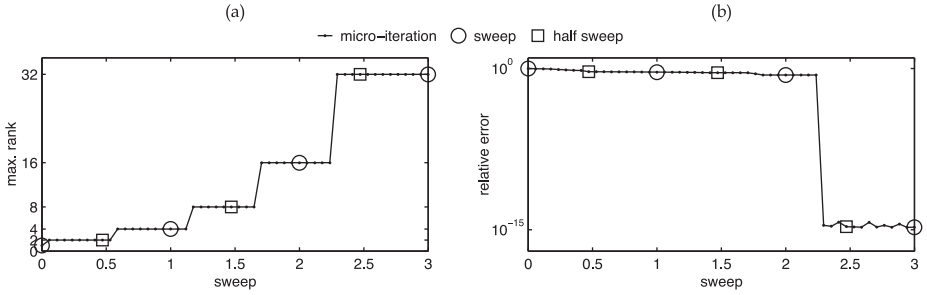

FIG. 6.3. *Development of maximum rank* (a) *and relative error* (b) *during an approximation with MALS.*

When approximating matrices B, approximation of B by truncated SVDs need not be unique (e.g., if B has identical second and third singular value, there is no unique rank-2 approximation of these matrices). In the same vein, a TT approximation, computed by successive SVDs of its unfoldings, does not need to be unique, even in its rematrification. Figure 6.2 shows another example from our "quantized second derivative" test series (see above), where the approximation computed depends on the inital guess. While the approximations have the same distance from the tensor U, they are distinct, and we conjecture that there may be multiple best approximations if TT decompositions corresponding to multiple singular values are computed.

6.2. Approximation with MALS. The big advantage of the MALS algorithm is its feature of automatic rank adaptation. Our experience is that starting with a rank-1 initial guess will always return a tensor with the rank of the approximated tensor. Figure 6.3 shows the relative error and the maximum ranks during a MALS approximation of a random tensor U of order $d = 10$ with $n_i = 2$ for all $i = 1, \ldots, 10$, resulting in a TT rank vector $\underline{r} = (2, 4, 8, 16, 32, 16, 8, 4, 2)$. In this example, it takes five MALS half sweeps to adapt the rank. When starting with an initial guess with the rank of U, MALS always returned U numerically exact after one half sweep.

6.3. Solving a linear system with MALS. We tested the behavior of the MALS algorithm for the solution of linear systems as described in section 4.1, where we utilized a *cg* method as inner solver. In sample calculations, we experienced that the ranks of the iterates may grow to unmanageable sizes during the solution process,

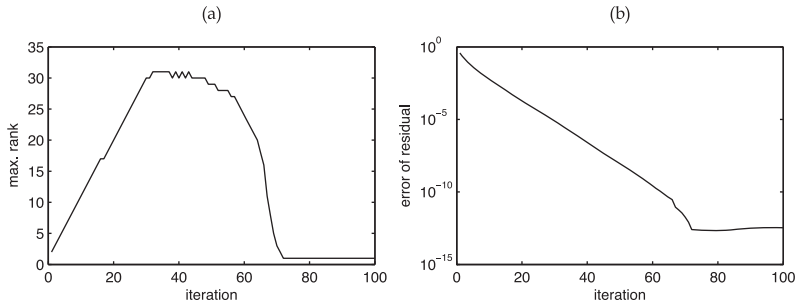

FIG. 6.4. *Solution of a linear system in cg TT arithmetic: maximum ranks of the iterates* (a) *and the norm of the error* (b).

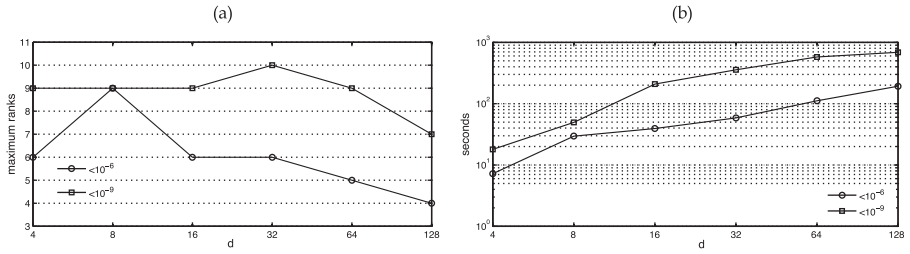

FIG. 6.5. *Solution of a linear system with MALS: maximum ranks of the solutions* (a) *and the timings* (b) *with different accuracies.*

only to decrease again when getting near to the solution; see Figure 6.4 for such an example. To save computational resources, we therefore exploited that the quantities $W_{i,i+1} \in \mathbb{R}^{r_{i-1} \times n_i \times n_{i+1} \times r_{i+1}}$ are tensors of order 4 (resp., 3 for $i = 1, d-1$) and can thus themselves be represented in the TT format. With an according modification of the contractions of G_i and $H_{i,2}$ they can also be represented as order-4 tensors in TT format, and linear algebra steps as, e.g., the addition of two tensors, matrix-vector multiplications and scalar multiplications can then be performed in the TT format. Because in this arithmetic ranks increase during the operations, we performed a recompression step (cf. [25]) for all quantities at the end at one iteration to obtain lower ranks. We did not iterate the *cg* method until convergence, but only did some *cg* steps and then computed a low-rank approximation of the iterate, this step then also replacing the SVD decimation step of the MALS procedure. If this approximation yielded no improvement compared with the former iterate, we raised the truncation accuracy to a new level. Although compared to the original MALS the number of sweeps needed increases by this procedure, the overall cost is reduced due to the lower ranks, thus facilitating the solution of the given linear systems. Another heuristic we found to be useful was to do one *cg* step with the full TT tensor to get a good initial guess for the MALS algorithm. We note that all these strategies are rather heuristic and that it would be desirable to have a rigorous strategy preventing the occurrance of high ranks in the approximation of low-rank quantities.

In Figure 6.5, we display the maximum ranks and computation times needed to compute two different accuracies $(10^{-6}, 10^{-9})$ for the solution of a linear system $AU = B$, using as system matrix A a TT approximation of the (canonical format) finite-difference d-dimensional Laplacian ($d \in \{4, 8, 16, 32, 64, 128\}$), discretized on a grid with 10 equally distributed points with Dirchlet boundary conditions. This TT

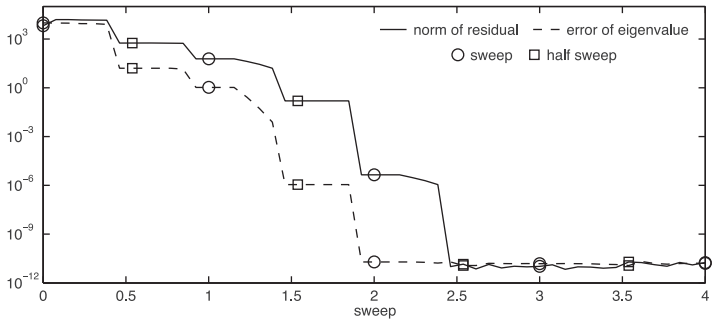

FIG. 6.6. *Eigenvector/eigenvalue approximation for a quantized TT operator.*

approximation has rank 2; see [25] for a sketch of the proof. As a right-hand side, we use the discretization of

$$B(x_1, \ldots, x_d) := \prod_{i=1}^{d} \exp(x_i), \qquad \underline{x} \in [0,1]^d.$$

Obviously, this tensor has canonical and TT rank 1. The analytical rank of the solution U is unknown; a numerical test yields a maximal TT rank $r_i = 13$. Note that the solution times are (sub)linear in d and also that for $d = 8$ and target accuracy of 10^{-6}, MALS "overshoots" this target and returns a solution of higher accuracy, but also of higher rank, which influences the timing of this iteration. This again stresses the importance of devising rigorous strategies for rank capping.

6.4. Solving an eigenvalue problem with ALS. We represented the discretization of the second-order derivate as a quantized TT operator (as in the first experiment) and computed its lowest eigenpair; see Figure 6.6. The solution has the TT ranks $r_i = 2$; see [29]. The errors may again stagnate during the micro-iterations, resulting in the characteristic plateaus as in the case of linear equations; on the whole though, we observed a fast and stable convergence behavior.

7. Conclusion and outlook. In this article, we investigated the behavior and structural properties of the ALS and its modification MALS when applied to some common optimization problems. In order to make use of the structural properties of the underlying optimization problems, we found that it is important to exploit the fact that the TT tensors can be represented as a left, respectively, right, orthonormalized component functions [25]. The ALS, being particularly simple to implement, shows extraordinarily nice convergence behavior similar to that of ALS applied to the Tucker format. This behavior will be complemented by a proof of local linear convergence in a forthcoming publication [30]. Unfortunately, ALS is sequential by definition. An according Jacobi-like procedure would allow for parallelization, and both algorithms allow for generalization to the HT format, for which their convergence properties are still unclear, however.

For MALS, our practical experience concerning stability is not any less positive, although proofs of convergence seem out of reach as long as no rigurous strategy for rank adaption is devised. Optimization of this step and also the arrangement of indices to which TT is sensitive should be investigated further in the future. In the context of quantum chemistry, the MALS algorithm applied to the binary Fock space corresponds to the successful DMRG algorithm, and we conjecture that in the context

of multireference methods, the potential of the MALS/DMRG approach has not fully been exploited and can profit from the connections with the newer developments of tensor approximation methods in the field of numerical mathematics.

Acknowledgment. The authors thank the DFG research center MATHEON and the Berlin Mathematical School for providing travel means used in connection with the development of this work.

REFERENCES

[1] G. Beylkin and M. J. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*, SIAM J. Sci. Comput., 6 (2005), pp. 2133–2159.
[2] J. D. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition*, Psychometrika, 3 (1970), p. 283.
[3] D. Conte and C. Lubich, *An error analysis of the multi-configuration time-dependent Hartree method of quantum dynamics*, M2AN Math. Model. Numer. Anal., 44 (2010), p. 759.
[4] M. Espig, *Effziente Bestapproximation Mittels Summen von Elementartensoren in Hohen Dimensionen*, Ph.D. thesis, 2007.
[5] M. Espig, W. Hackbusch, T. Rohwedder, and R. Schneider, *Variational calculus with sums of elementary tensors of fixed rank*, Numer. Math., to appear; also available online from http://www.mis.mpg.de/de/publications/preprints/2009/prepr2009-52.html.
[6] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika, 3 (1936), p. 211.
[7] A. Falcó and W. Hackbusch, *On Minimal Subspaces in Tensor Representations*, Found. Comput. Math., to appear.
[8] L. Grasedyck, *Hierarchical singular value decomposition of tensors*, SIAM. J. Matrix Anal. Appl., 31 (2010), p. 2029.
[9] V. de Silva and L.-H. Lim, *Tensor rank and the ill-posedness of the best low-rank approximation problem*, SIAM J. Matrix Anal. Appl., 3 (2008), pp. 1084–1127.
[10] E. Hairer, C. Lubich, and G. Wanner, *Geometrical Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, 2nd ed., Springer, Berlin, 2006.
[11] W. Hackbusch and S. Kühn, *A new scheme for the tensor representation*, J. Fourier Anal. Appl., 5 (2009), p. 706.
[12] S. Holtz, T. Rohwedder, and R. Schneider, *On manifolds of tensors of fixed TT rank*, Numer. Math., DOI:10.1007/s00211-011-0419-7.
[13] T. Huckle, K. Waldherr, and T. Schulte-Herbrüggen, *Computations in quantum tensor networks*, Linear Algebra Appl., to appear.
[14] A. Kapteyn, H. Neudecker, and T. Wansbeek, *An approach to n-mode components analysis*, Psychometrika, 51 (1986), p. 269.
[15] V. A. Kazeev and B. N. Khoromskij, *On Explicit QTT Representation of Laplace Operator and Its Inverse*, http://www.mis.mpg.de/preprints/2010/preprint2010_75.pdf (2010).
[16] O. Koch and C. Lubich, *Dynamical low rank approximation*, SIAM J. Matrix Anal. Appl., 2 (2008), p. 434.
[17] O. Koch and C. Lubich, *Dynamical low-rank approximation of tensors*, SIAM J. Matrix Anal. Appl., 31 (2010), p. 2360.
[18] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 3, pp. 455–500.
[19] P. M. Kroonenberg and J. De Leeuw, *Principal component analysis of three-mode data by means of alternating least squares algorithms*, Psychometrika, 45 (1980), p. 69.
[20] C. Lubich, *From Quantum to Classical Molecular Dynamics: Reduced Methods and Numerical Analysis*, Zürich Lectures in Advanced Mathematics, EMS, 2008.
[21] K. H. Marti, B. Bauer, M. Reiher, M. Troyer, and F. Verstraete, *Complete-graph tensor network states: A new fermionic wave function ansatz for molecules*, New J. Phys., 12 (2010), 103008.
[22] D. Meyer, F. Gatti, and G. A. Worth, *The multi-configurational time-dependent Hartree (MCTDH) method*, Phys. Rep., 324 (2000), p. 1.
[23] D. Meyer, U. Manthe, and L. Cederbaum, *The multi-configurational time-dependent Hartree approach*, Chem. Phys. Lett., 165 (1990), p. 73.

[24] M. J. Mohlenkamp, *Musings on multilinear fitting*, Linear Algebra Appl., (2011), DOI:10.1016/j.laa.2011.04.019.
[25] I. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.
[26] I. Oseledets, *On a new tensor decomposition*, Dokl. Math. 427 (2009).
[27] I. Oseledets, *TT Toolbox* 1.0: *Fast Multidimensional Array Operations in MATLAB*, preprint 2009-06, INM RAS, 2009.
[28] I. V. Oseledets and E. E. Tyrtyshnikov, *Tensor tree decomposition does not need a tree*, Linear Algebra Appl., submitted.
[29] I. Oseledets and E. E. Tyrtyshnikov, *Breaking the Curse of Dimensionality, or How to Use SVD in Many Dimensions*, J. Sci. Comput., 5 (2009), p. 3744.
[30] T. Rohwedder and A. Uschmajew, *Local convergence of alternating schemes for optimisation of convex problems in the TT format*, SIAM J. Numer. Anal., submitted.
[31] E. Schmidt, *Zur Theorie der linearen und nichtlinearen Integralgleichungen. I Teil. Entwicklung willkürlichen Funktionen nach System vorgeschriebener*, Math. Ann., 63 (1907), p. 433.
[32] U. Schollwöck, *The density-matrix renormalization group*, Rev. Mod. Phys., 1 (2005), p. 259.
[33] L. R. Tucker, *Some mathematical notes on three-mode factor analysis*, Psychometrica, 3 (1966), pp. 279–311.
[34] G. Vidal, *Efficient classical simulation of slightly entagled quantum computation*, Phys. Rev. Lett., 14 (2003).
[35] S. R. White, *Density matrix formulation for quantum renormalization groups*, Phys. Rev. Lett., 69 (1992), p. 2863.
[36] http://en.wikipedia.org/wiki/Density_matrix_renormalization_group,