SIAM J. SCI. COMPUT.
Vol. 36, No. 5, pp. A2248–A2271

ALTERNATING MINIMAL ENERGY METHODS FOR LINEAR SYSTEMS IN HIGHER DIMENSIONS*

SERGEY V. DOLGOV† AND DMITRY V. SAVOSTYANOV‡

Abstract. We propose algorithms for the solution of high-dimensional symmetrical positive definite (SPD) linear systems with the matrix and the right-hand side given and the solution sought in a low-rank format. Similarly to density matrix renormalization group (DMRG) algorithms, our methods optimize the components of the tensor product format subsequently. To improve the convergence, we expand the search space by an inexact gradient direction. We prove the geometrical convergence and estimate the convergence rate of the proposed methods utilizing the analysis of the steepest descent algorithm. The complexity of the presented algorithms is linear in the mode size and dimension, and the demonstrated convergence is comparable to or even better than the one of the DMRG algorithm. In the numerical experiment we show that the proposed methods are also efficient for non-SPD systems, for example, those arising from the chemical master equation describing the gene regulatory model at the mesoscopic scale.

Key words. high-dimensional problems, tensor train format, alternating linear scheme, density matrix renormalization group, steepest descent, Poisson equation, chemical master equation

AMS subject classifications. 15A69, 33F05, 65F10

DOI. 10.1137/140953289

1. Introduction.

When high-dimensional problems are concerned, not many algorithms are able to beat the *curse of dimensionality* and solve them efficiently. In this paper we consider a *symmetrical positive definite* (SPD) linear system $Ax = b$, where the solution x and the right-hand side b may be seen as $n_1 \times n_2 \times \cdots \times n_d$ arrays, and $A = A^* > 0$. The number of unknowns $N = n_1 n_2 \cdots n_d$ grows exponentially with d, and even for moderate *dimension* d and *mode sizes* $n_1, \ldots, n_d$ the problem cannot be solved by standard algorithms.

Tensor product methods [36, 33, 19, 18] are promising for the problems of high dimension. They implement the *separation of variables* for all vectors and matrices in consideration and significantly reduce the number of representation parameters. In this paper we consider a particularly simple tensor product format that was proposed in quantum physics as the *matrix product states* (MPS) [15] and the *density matrix renormalization group* (DMRG) [59, 60] formalism, and later rediscovered in numerical linear algebra (NLA) as the *tensor train* (TT) format [48, 45].

Although the MPS/DMRG representation is equivalent to the TT, the algorithms developed in the quantum physics and the NLA community are quite different. The MPS/DMRG methods are based on the *optimization* in tensor formats and were originally developed to find the *ground state* of a system, i.e., the extreme eigenpair of a Hermitian matrix A. This problem is equivalent to the minimization of

*Submitted to the journal's Methods and Algorithms for Scientific Computing section January 17, 2014; accepted for publication (in revised form) July 8, 2014; published electronically September 25, 2014.

http://www.siam.org/journals/sisc/36-5/95328.html

†Max-Planck-Institut für Mathematik in den Naturwissenschaften, Inselstaße 22-26, Leipzig D-04103, Germany (sergey.v.dolgov@gmail.com). This work was supported by the Stipend of President of Russia at the Institute of Numerical Mathematics of Russian Academy of Sciences.

‡School of Chemistry, University of Southampton, Highfield Campus, Southampton SO17 1BJ, United Kingdom (dmitry.savostyanov@gmail.com). This author was supported by EPSRC grant EP/H003789/2 at the University of Southampton.

the *Rayleigh quotient* $\mathtt{Q}_A(x) = (x, Ax)/(x, x)$. To solve an SPD linear system $Ax = b$ using the same optimization scheme, it is sufficient to substitute $\mathtt{Q}_A$ by the *energy function*

$$\mathtt{J}_{A,b}(x) = \|x_\star - x\|_A^2 = (x, Ax) - 2\Re(x, b) + \text{const}, \tag{1.1}$$

where $x_\star = A^{-1}b$ denotes the exact solution. The A-norm $\|\cdot\|_A$ and the corresponding A-scalar product are defined as follows:

$$(u, v)_A = (u, Av), \qquad \|u\|_A^2 = (u, u)_A, \qquad \|u\|_I^2 = (u, u) = \|u\|^2,$$

where $\|\cdot\|$ denotes the Frobenius norm. The optimization of (1.1) over MPS vectors was first introduced as the *dynamical* DMRG [27]; see also the reviews on other MPS/DMRG techniques [53, 54, 25]. The DMRG algorithm substitutes a highly nonlinear *simultaneous* optimization over all cores of a tensor format (or *sites*) by a *subsequent* optimization over single sites (*one-site* DMRG) or pairs of neighboring sites (*two-site* DMRG). The two-site DMRG algorithm demonstrates remarkably fast convergence and is a method of choice for many systems considered in quantum physics, particularly for linear lattices. However, its theoretical understanding is still very limited, and only *local* convergence estimates for the simplest one-site DMRG algorithm, also known as the *alternating linear scheme* (ALS), are available [49].

In the NLA community, the TT format has been proposed as a generalization of a matrix *low-rank* decomposition to multidimensional arrays, or *tensors*. The methods developed in the NLA community often can be seen as classical (e.g., Krylov-type) solvers, where all vectors are truncated to a low-rank format: see, e.g., algorithms for the solution of linear systems [51, 21, 47, 38, 34, 31, 39, 2, 8, 4, 3], partial eigenproblems [41, 40, 24, 20], and Fourier transform [11]. However, tensor ranks of vectors typically grow with iterations, and the methods are often struggling, because the compact representation of intermediate vectors (e.g., the basis vectors of the Krylov subspace) is not guaranteed by a "background" of the problem, in contrast to the solution, for which such estimates may be available. The possibility to use the relaxed approximation in the latter iterations in the spirit of *inexact Krylov methods* [55, 57] has been considered only in a few papers; see, e.g., [21, 8], where the rank accumulation is still observed for the latter Krylov vectors. For certain problems, the possibility to control the ranks was recently discussed in [3].

Here we propose the *alternating minimal energy* (AMEn) algorithm, which bridges the optimization in tensor formats and the classical iterative methods of the NLA. The first algorithm of such a kind is the *corrected one-site density matrix renormalization group* (DMRG1c) [61] that targets the Krylov vector Ax and uses it (with some weight) to modify the search space in the optimization step. In contrast to the DMRG1c, the proposed AMEn algorithm targets the *gradient* direction, i.e., the *residual* $r = b - Ax$ for the linear system and $r = Ax - \mathtt{Q}_A(x)x$ for the ground state problem. This and other differences are emphasized in [13], where it is shown that the AMEn algorithm is stable to perturbations and free from tuning parameters and heuristics and ultimately converges better than the DMRG1c for the original numerical experiment from [61].

In this paper we introduce the AMEn algorithm for the solution of SPD linear systems, analyze its convergence on the base of the *steepest descent* (SD) algorithm [50], and demonstrate that the AMEn has (1) proven geometrical convergence with the global upper bound for the convergence rate, (2) practical convergence competitive

with the one of the two-site DMRG in numerical experiments, (3) numerical complexity similar to the one-site DMRG or ALS.

The rest of the paper is organized as follows. Section 2 gives necessary definitions, section 3 recalls DMRG algorithms, section 4 discusses several variants of the SD method, section 5 presents the AMEn algorithm with analysis, section 6 gives practical implementation details, and section 7 contains numerical experiments.

2. TT notation and definitions. In this section we introduce the notation and definitions for the high-dimensional arrays (tensors) and the TT format.

2.1. Tensors and vectorizations. A d-tensor $\mathbf{x} = [\mathbf{x}(i_1, \dots, i_d)] \in \mathbb{C}^{n_1 \times \dots \times n_d}$ has d *mode indices* $i_p = 1, \dots, n_p$, $p = 1, \dots, d$, where n_p are referred to as the *mode sizes* of $\mathbf{x}$. In a linear system $Ax = b$ we have to consider the same data x as a vector, and for this purpose we use the *vectorization* $x = \operatorname{vec} \mathbf{x} \in \mathbb{C}^{n_1 \cdots n_d}$ defined as follows:

$$x = \operatorname{vec} \mathbf{x} \qquad \Leftrightarrow \qquad x(\overline{i_1 i_2 \dots i_d}) = \mathbf{x}(i_1, i_2, \dots, i_d). \tag{2.1}$$

Here and later $\overline{i_1 i_2 \dots i_d}$ denotes a single index combined from $i_1, i_2, \dots, i_d$, and the equation is entrywise, i.e., holds for all values of all indices. In this paper the *index grouping* operation is *big-endian*,

$$\overline{i_1 i_2 \dots i_{d-1} i_d} = i_d + (i_{d-1} - 1) n_d + \dots + (i_1 - 1) n_2 \cdots n_d,$$

and therefore is consistent with the *Kronecker (tensor) product* in the following way: for vectors $x^{(p)} = [x^{(p)}(i_p)]$, $p = 1, \dots, d$, it holds that

$$x = x^{(1)} \otimes x^{(2)} \otimes \dots \otimes x^{(d)} \qquad \Leftrightarrow \qquad x(\overline{i_1 i_2 \dots i_d}) = x^{(1)}(i_1) x^{(2)}(i_2) \cdots x^{(d)}(i_d).$$

2.2. TT format. The TT format for $x = \operatorname{vec} \mathbf{x}$ is defined as follows:

$$\begin{aligned} x &= \tau(\bar{\mathbf{x}}) = \tau(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(d)}) \in \mathbb{C}^{n_1 \cdots n_d}, \\ x(\overline{i_1 \dots i_d}) &= \sum_{\alpha} \mathbf{x}^{(1)}_{\alpha_0, \alpha_1}(i_1) \mathbf{x}^{(2)}_{\alpha_1, \alpha_2}(i_2) \cdots \mathbf{x}^{(d-1)}_{\alpha_{d-2}, \alpha_{d-1}}(i_{d-1}) \mathbf{x}^{(d)}_{\alpha_{d-1}, \alpha_d}(i_d). \end{aligned} \tag{2.2}$$

Here $\alpha_p = 1, \dots, k_p$ are the *rank indices*, k_p are the TT ranks, $\mathbf{x}^{(p)} \in \mathbb{C}^{k_{p-1} \times n_p \times k_p}$ are the TT cores, and $\bar{\mathbf{x}} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(d)})$ denotes the whole TT. We assume that $k_0 = k_d = 1$, i.e., the *border indices* α_0 and α_d are trivial and used only to unify the notation. The vector of TT ranks is denoted by $\mathtt{k}(\bar{\mathbf{x}}) = (k_1, \dots, k_{d-1}) = k$. The summation over $\alpha = (\alpha_1, \dots, \alpha_{d-1})$ goes over all pairs of repeated indices; cf. the *Einstein notation* [14] and see Figure 1.

Similarly, the TT format $A = \tau(\bar{\mathbf{A}})$ for a matrix $A = [A(i, j)]$ in d dimensions is

$$A(i, j) = A(\overline{i_1 \dots i_d}, \overline{j_1 \dots j_d}) = \sum_{\gamma} \mathbf{A}^{(1)}_{\gamma_0, \gamma_1}(i_1, j_1) \cdots \mathbf{A}^{(d)}_{\gamma_{d-1}, \gamma_d}(i_d, j_d).$$

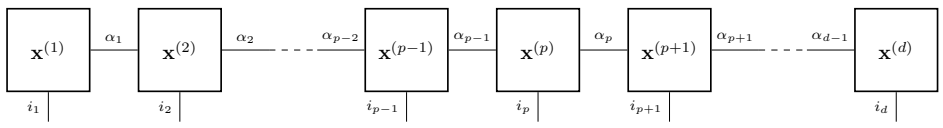

FIG. 1. *The TT format* (2.2). *Each box is a TT core with lines (*legs*, or* bonds*) showing its indices. A bond between two cores assumes a summation over the joint index.*

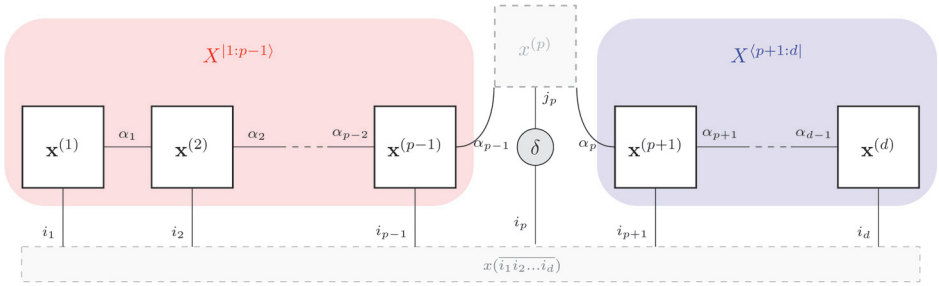

FIG. 2. *The frame matrix* (2.7) *maps a TT core (above) to a large vector (below).*

2.3. Subtrains. For nontrivial border indices the operation τ works as follows:

$$\mathbf{x}^{(p:q)} = \tau(\mathbf{x}^{(p)}, \dots, \mathbf{x}^{(q)}) \in \mathbb{C}^{k_{p-1} \times (n_p \cdots n_q) \times k_q}, \tag{2.3}$$

$$\mathbf{x}^{(p:q)}_{\alpha_{p-1},\alpha_q}(\overline{i_p \dots i_q}) = \sum_{\alpha} \mathbf{x}^{(p)}_{\alpha_{p-1},\alpha_p}(i_p)\mathbf{x}^{(p+1)}_{\alpha_p,\alpha_{p+1}}(i_{p+1}) \cdots \mathbf{x}^{(q)}_{\alpha_{q-1},\alpha_q}(i_q),$$

where α is again a shortcut for the repeated indices $\alpha_p, \dots, \alpha_{q-1}$. The three-dimensional array $\mathbf{x}^{(p:q)}$ is given here by a *subtrain* of $\bar{\mathbf{x}}$, composed of the TT cores $\mathbf{x}^{(p)}, \dots, \mathbf{x}^{(q)}$. At the same time, $\mathbf{x}^{(p:q)}$ may itself be seen as a TT core, which merges the cores of the subtrain; the τ notation allows us to rewrite (2.2) as

$$x = \tau(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(p-1)}, \mathbf{x}^{(p:q)}, \mathbf{x}^{(q+1)}, \dots, \mathbf{x}^{(d)}). \tag{2.4}$$

2.4. Left and right matricizations. In (2.1) we stretch a tensor $\mathbf{x}$ into a vector $x = \operatorname{vec} \mathbf{x}$ with the same entries. We will apply this and other *reshapes* to the TT cores $\mathbf{x}^{(p)}$, turning them into vectors $x^{(p)}$ and matrices $X^{|p\rangle}$ and $X^{\langle p|}$ as follows:

$$\mathbf{x}^{(p)} \in \mathbb{C}^{k_{p-1} \times n_p \times k_p}, X^{|p\rangle} \in \mathbb{C}^{k_{p-1} n_p \times k_p}, X^{\langle p|} \in \mathbb{C}^{k_{p-1} \times n_p k_p}, x^{(p)} \in \mathbb{C}^{k_{p-1} n_p k_p},$$

$$\mathbf{x}^{(p)}_{\alpha_{p-1},\alpha_p}(i_p) = X^{|p\rangle}(\overline{\alpha_{p-1} i_p}, \alpha_p) = X^{\langle p|}(\alpha_{p-1}, \overline{i_p \alpha_p}) = x^{(p)}(\overline{\alpha_{p-1} i_p \alpha_p}).$$

The introduced symbols mirror the *Dirac notation* [6] used in quantum physics: $X^{|p\rangle}$ is a *tall* matrix (a set of columns, or *ket*s), and $X^{\langle p|}$ is a *wide* matrix (a set of rows, or *bra*s). We always assume that $\mathbf{x}^{(p)}$, $X^{|p\rangle}$, $X^{\langle p|}$, and $x^{(p)}$ share the same entries, i.e., when one of them is defined, all the family of reshapes can be immediately used.

This notation also applies to subtrains, e.g., $\mathbf{x}^{(p:q)}$ in (2.3) may be reshaped into

$$x^{(p:q)} \in \mathbb{C}^{k_{p-1}(n_p \cdots n_q) k_q}, \quad X^{|p:q\rangle} \in \mathbb{C}^{k_{p-1}(n_p \cdots n_q) \times k_q}, \quad X^{\langle p:q|} \in \mathbb{C}^{k_{p-1} \times (n_p \cdots n_q) k_q}.$$

2.5. Interface matrices. The *interface* matrices are a special case of subtrains,

$$X^{|1:p\rangle} = \mathbf{x}^{(1:p)} \in \mathbb{C}^{(n_1 \cdots n_p) \times k_p}, \qquad X^{\langle p+1:d|} = \mathbf{x}^{(p+1:d)} \in \mathbb{C}^{k_p \times (n_{p+1} \cdots n_d)}. \tag{2.5}$$

Note that $\mathbf{x}^{(1:p)}$ and $\mathbf{x}^{(p+1:d)}$ are matrices, since one border index vanishes due to the $k_0 = k_d = 1$ condition. Ultimately, $\mathbf{x}^{(1:d)} = \tau(\bar{\mathbf{x}}) = x$ by (2.2).

We will formally extend the definition (2.5) to include $X^{|1:0\rangle} = X^{\langle d+1:d|} = 1$.

2.6. Frame matrices. An important property of the TT format (2.2) is that it is *linear* w.r.t. each TT core $\mathbf{x}^{(p)}$. Indeed,

$$x = X_{\neq p} x^{(p)}, \tag{2.6}$$

where $x^{(p)} = \operatorname{vec} \mathbf{x}^{(p)} \in \mathbb{C}^{k_{p-1} n_p k_p}$ is a vector of parameters of a TT core, and $X_{\neq p}$ is the *frame matrix*. It is easy to check (see Fig. 2) that

$$X_{\neq p} = X^{|1:p-1\rangle} \otimes I_{n_p} \otimes \left(X^{\langle p+1:d|}\right)^\top \in \mathbb{C}^{(n_1 \cdots n_d) \times (k_{p-1} n_p k_p)}, \tag{2.7}$$

where $I_{n_p} \in \mathbb{R}^{n_p \times n_p}$ is the identity matrix. Similarly, the *two-site* frame matrix is written as follows:

$$X_{\neq p,p+1} = X^{|1:p-1\rangle} \otimes I_{n_p} \otimes I_{n_{p+1}} \otimes \left(X^{\langle p+2:d|}\right)^\top \in \mathbb{C}^{(n_1 \cdots n_d) \times (k_{p-1} n_p n_{p+1} k_{p+1})}. \tag{2.8}$$

For a given $\bar{\mathbf{x}}$, the frame matrix $X_{\neq p}$ always maps to a subspace containing the vector $x = \tau(\bar{\mathbf{x}})$, i.e., $\tau(\bar{\mathbf{x}}) \in \operatorname{span} X_{\neq p} \subset \mathbb{C}^{n_1 \cdots n_d}$.

3. Alternating minimization methods. In this section we briefly recall the DMRG algorithm, which is based on the alternating optimization in the TT format. The original DMRG scheme [59, 60] was proposed to find the ground state of a system by the minimization of the *Rayleigh quotient* $\mathtt{Q}_A(x) = (x, Ax)/(x, x)$. It was later applied [27, 23] to solve an SPD linear system $Ax = b$ by the minimization of the *energy function* (1.1). In the following we focus on the latter problem.

3.1. One-site DMRG algorithm. We restrict the optimization of $\mathtt{J}_{A,b}(x)$ to vectors $x = \tau(\bar{\mathbf{x}})$ represented by (2.2) with *fixed* TT ranks $\mathtt{k}(\bar{\mathbf{x}}) = (k_1, \ldots, k_{d-1})$,

$$\bar{\mathbf{x}}_\star = \arg\min_{\bar{\mathbf{x}}} \mathtt{J}_{A,b}(\tau(\bar{\mathbf{x}})) \quad \text{over} \quad \bar{\mathbf{x}} \in \bigtimes_{p=1}^{d} \mathbb{C}^{k_{p-1} \times n_p \times k_p}.$$

Even for small TT ranks, this highly nonlinear problem cannot be solved at once. In a one-site DMRG (or ALS) algorithm it is substituted by a sequence of *microsteps*, i.e., *consecutive* optimizations over TT cores $\mathbf{x}^{(p)}$. Each such *local problem* is written as

$$\mathbf{u}^{(p)} = \arg\min_{\mathbf{x}^{(p)}} \mathtt{J}_{A,b}(\tau(\bar{\mathbf{x}})) \quad \text{over} \quad \mathbf{x}^{(p)} \in \mathbb{C}^{k_{p-1} \times n_p \times k_p}. \tag{3.1}$$

The TT core $\mathbf{x}^{(p)}$ is then replaced by $\mathbf{u}^{(p)}$, and the next core is considered; usually the cores are updated one by one back and forth the chain: first we set $p = 1, \ldots, d$ (*forward half-sweep*), then $p = d, \ldots, 1$ (*backward half-sweep*), and so on again.

The linearity (2.6) of the format (2.2) allows us to rewrite (3.1) as follows:

$$\begin{aligned} u^{(p)} &= \arg\min_{x^{(p)}} \mathtt{J}_{A_p,b_p}(x^{(p)}) \quad \text{over} \quad x^{(p)} \in \mathbb{C}^{k_{p-1} n_p k_p}, \\ A_p &= X_{\neq p}^* A X_{\neq p} \in \mathbb{C}^{(k_{p-1} n_p k_p) \times (k_{p-1} n_p k_p)}, \qquad b_p = X_{\neq p}^* b \in \mathbb{C}^{k_{p-1} n_p k_p}. \end{aligned} \tag{3.2}$$

The unique minimum is delivered by the solution of the *local system* $A_p u^{(p)} = b_p$, which is of a reasonable size and within capabilities of standard methods, e.g., iterative schemes [50]. As shown in Figure 3, A_p and b_p can be assembled from the TT cores of $A = \tau(\bar{\mathbf{A}})$, $x = \tau(\bar{\mathbf{x}})$, and $b = \tau(\bar{\mathbf{b}})$, avoiding appearance of exponentially large arrays.

The accuracy of the obtained solution depends on the *conditioning* of the local system. Fortunately, it can be put under control exploiting the nonuniqueness of the TT format (2.2): TTs $\bar{\mathbf{s}}$ and $\bar{\mathbf{t}}$ map to the same vector $\tau(\bar{\mathbf{s}}) = \tau(\bar{\mathbf{t}})$ as soon as

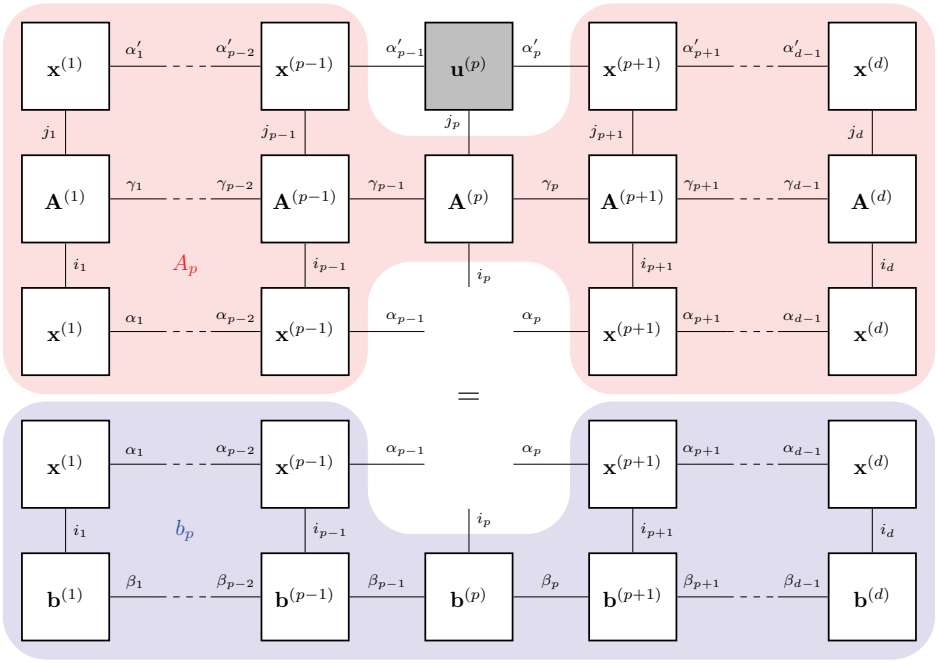

FIG. 3. *A local linear system* $A_p u^{(p)} = b_p$ *defined by* (3.2)*, assembled from the cores of the TT formats of* A*,* x*, and* b*.*

$$\mathbf{t}^{(p)}(i_p) = H_{p-1}^{-1}\mathbf{s}^{(p)}(i_p)H_p, \qquad p = 1, \dots, d, \tag{3.3}$$

where $H_0 = H_d = 1$ and $H_p \in \mathbb{C}^{k_p \times k_p}$, $p = 1, \dots, d-1$, are arbitrary nonsingular matrices. This gives $\mathtt{J}_{A,b}(\tau(\bar{\mathbf{s}})) = \mathtt{J}_{A,b}(\tau(\bar{\mathbf{t}}))$ and $S_{\neq p} \neq T_{\neq p}$, providing us with the necessary flexibility. Given any $\bar{\mathbf{s}}$ and p, we can construct the transformation (3.3) such that $T_{\neq p}$ is *unitary*, which for a tall matrix means $T_{\neq p}^* T_{\neq p} = I_{k_{p-1}n_p k_p}$. This representation is known as a *gauge condition* in the MPS literature and a *canonical form* in the DMRG literature; see, e.g., [53]. The result is based on the following statements.

PROPOSITION 3.1 (left TT orthogonality). *If in a TT* $\bar{\mathbf{x}}$ *the left TT cores* $\mathbf{x}^{(j)}$ *are left-orthogonal, i.e.,* $(X^{|j\rangle})^* X^{|j\rangle} = I_{k_j}$*, for* $j = 1, \dots, p$*, then the left interface matrix is unitary* $(X^{|1:p\rangle})^* X^{|1:p\rangle} = I_{k_p}$*.*

PROPOSITION 3.2 (right TT orthogonality). *If in a TT* $\bar{\mathbf{x}}$ *the right TT cores* $\mathbf{x}^{(q)}$ *are right-orthogonal, i.e.,* $X^{\langle q|}(X^{\langle q|})^* = I_{k_{q-1}}$*, for* $q = p+1, \dots, d$*, then the right interface matrix is unitary* $X^{\langle p+1:d|}(X^{\langle p+1:d|})^* = I_{k_p}$*.*

PROPOSITION 3.3 (orthogonality of the frame). *If in* $\bar{\mathbf{x}}$ *the TT cores* $\mathbf{x}^{(j)}$ *are left-orthogonal for* $j = 1, \dots, p-1$*, and the TT cores* $\mathbf{x}^{(q)}$ *are right-orthogonal for* $q = p+1, \dots, d$*, then the frame matrix* $X_{\neq p}$ *is unitary* $X_{\neq p}^* X_{\neq p} = I_{k_{p-1}n_p k_p}$*.*

In the following, "orthogonal" is used as a synonym of "unitary."

The left and the right TT orthogonality can be ensured using the standard QR and LQ algorithms [17] applied to the matricizations of TT cores. In Algorithm 1 we recall the one-site DMRG algorithm that enforces the orthogonality of $X_{\neq p}$ when $\mathbf{x}^{(p)}$ is updated. For orthogonal $X_{\neq p}$ the spectrum of A_p lies within the spectral range of A,

ALGORITHM 1. One-site DMRG for SPD linear system $Ax = b$ (forward half-sweep).

Require: Initial guess $t = \tau(\bar{\mathbf{t}})$ in the TT format (2.2) with $\mathtt{k}(\bar{\mathbf{t}}) = k$.
Ensure: Updated vector $x = \tau(\bar{\mathbf{x}})$ with $\mathtt{k}(\bar{\mathbf{x}}) = k$ s.t. $\mathtt{J}_{A,b}(x) \leqslant \mathtt{J}_{A,b}(t)$.
Copy $\mathbf{x}^{(p)} = \mathbf{t}^{(p)}$, $p = 1, \dots, d$.
for $p = d, \dots, 2$ **do** {Orthogonalization of the right interface}
  Compute LQ decomposition $X^{\langle p|} = LQ$, $QQ^* = I$.
  Replace $X^{\langle p|} := Q$, and $X^{|p-1\rangle} := X^{|p-1\rangle} L$.
end for
for $p = 1, \dots, d$ **do** {Optimization over TT cores}
  Form A_p and b_p by (3.2).
  Solve $A_p u^{(p)} = b_p$. {In iterative solver take $x^{(p)}$ as an initial guess}
  Replace $\mathbf{x}^{(p)} := \mathbf{u}^{(p)}$.
  if $k \neq d$ **then** {Orthogonalization of the left interface, if required}
    Compute QR decomposition $X^{|p\rangle} = QR$, $Q^*Q = I$.
    Replace $X^{|p\rangle} := Q$, and $X^{\langle p+1|} := RX^{\langle p+1|}$.
  end if
end for
return $x = \tau(\bar{\mathbf{x}})$, where $\bar{\mathbf{x}} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(d)})$.

$$\lambda_{\min}(A_p) = \lambda_{\min}(X_{\neq p}^* A X_{\neq p}) = \min_{\|v\|=1} (X_{\neq p} v, A X_{\neq p} v) = \min_{\substack{\|u\|=1 \\ u \in \operatorname{span} X_{\neq p}}} (u, Au)$$
$$\geqslant \min_{\|u\|=1} (u, Au) = \lambda_{\min}(A),$$

and similarly $\lambda_{\max}(A_p) \leqslant \lambda_{\max}(A)$. It follows that the *condition numbers* satisfy $\kappa(A_p) \leqslant \kappa(A)$, i.e., the local system (3.2) is conditioned not worse than $Ax = b$.

In the following we will silently assume the orthogonality of interfaces whenever necessary, omitting the required operations in the algorithms.

3.2. Two-site DMRG algorithm. The drawback of the one-site DMRG is that the TT ranks remain the same during the computations. Therefore, we have to guess the TT ranks of the solution a priori, which might be difficult. If we underestimate them, the returned solution will be far from the exact one; if we overestimate them, the local problems will be more difficult to solve.

The two-site DMRG allows us to *change* (usually to *increase*) the TT ranks adaptively during the computations. It considers the vector in the following form:

$$x = \tau(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(p-1)}, \mathbf{x}^{(p,p+1)}, \mathbf{x}^{(p+2)}, \dots, \mathbf{x}^{(d)}), \tag{3.4}$$

similarly to (2.4). The local optimization step is performed similarly to (3.1), but over the elements of the *supercore* $\mathbf{x}^{(p,p+1)}$, as follows:

$$u^{(p,p+1)} = \arg \min_{x^{(p,p+1)}} \mathtt{J}_{A_{p,p+1}, b_{p,p+1}}(x^{(p,p+1)}) \quad \text{over} \quad x^{(p,p+1)} \in \mathbb{C}^{k_{p-1} n_p n_{p+1} k_{p+1}}. \tag{3.5}$$

The minimizer is the solution of the *local system* $A_{p,p+1} u^{(p,p+1)} = b_{p,p+1}$ with

$$A_{p,p+1} = X_{\neq p,p+1}^* A X_{\neq p,p+1} \in \mathbb{C}^{(k_{p-1} n_p n_{p+1} k_{p+1}) \times (k_{p-1} n_p n_{p+1} k_{p+1})}, \tag{3.6}$$
$$b_{p,p+1} = X_{\neq p,p+1}^* b \in \mathbb{C}^{k_{p-1} n_p n_{n+1} k_{p+1}},$$

ALGORITHM 2. Two-site DMRG for SPD linear system $Ax = b$ (forward half-sweep).

Require: Initial guess $t = \tau(\bar{\mathbf{t}})$ with $\mathtt{k}(\bar{\mathbf{t}}) = k$, accuracy ε, and/or rank bounds $k^{\max}$.
Ensure: Updated vector $x = \tau(\bar{\mathbf{x}})$ with $\mathtt{k}(\bar{\mathbf{x}}) = k' \leqslant k^{\max}$ if $k^{\max}$ are given.

1: Copy $\mathbf{x}^{(p)} = \mathbf{t}^{(p)}$, $p = 1, \dots, d$.
2: **for** $p = 1, \dots, d-1$ **do** {Optimization over TT cores}
3: Ensure the orthogonality of the frame matrix $X^*_{\neq p,p+1} X_{\neq p,p+1} = I$.
4: Form $A_{p,p+1}$ and $b_{p,p+1}$ by (3.6).
5: Solve $A_{p,p+1} u^{(p,p+1)} = b_{p,p+1}$. {Use $x^{(p,p+1)}$ as an initial guess}
6: Decompose $\mathbf{u}^{(p,p+1)}$ into $\mathbf{u}^{(p)}$ and $\mathbf{u}^{(p+1)}$ by (3.7). Choose new rank k'_p s.t. $\|U^{(p,p+1)} - \tilde{U}^{(p,p+1)}\| \leqslant \varepsilon \|U^{(p,p+1)}\|$ and/or $k'_p \leqslant k_p^{\max}$.
7: Replace $\mathbf{x}^{(p)} := \mathbf{u}^{(p)}$ and $\mathbf{x}^{(p+1)} := \mathbf{u}^{(p+1)}$.
8: **end for**
9: **return** $x = \tau(\bar{\mathbf{x}})$, where $\bar{\mathbf{x}} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(d)})$.

where $X_{\neq p,p+1}$ is defined in (2.8). As previously, we assume that the orthogonality $X^*_{\neq p,p+1} X_{\neq p,p+1} = I_{k_{p-1} n_p n_{p+1} k_{p+1}}$ is enforced for the frame matrix, and hence the local problem is conditioned not worse than the system $Ax = b$.

The updated TT core $\mathbf{u}^{(p,p+1)}$ is then decomposed to $\mathbf{u}^{(p)}$ and $\mathbf{u}^{(p+1)}$. We reshape $\mathbf{u}^{(p,p+1)}_{\alpha_{p-1},\alpha_{p+1}}(\overline{i_p i_{p+1}}) = U^{(p,p+1)}(\overline{\alpha_{p-1} i_p}, \overline{i_{p+1} \alpha_{p+1}})$ and apply a rank-revealing matrix decomposition algorithm, e.g., the *singular value decomposition* (SVD) [17]

$$U^{(p,p+1)} \approx \tilde{U}^{(p,p+1)} = U^{|p\rangle} U^{\langle p+1|}, \tag{3.7}$$
$$U^{|p\rangle} \in \mathbb{C}^{k_{p-1} n_p \times k'_p}, \qquad U^{\langle p+1|} \in \mathbb{C}^{k'_p \times n_{p+1} k_{p+1}}.$$

The perturbation introduced to $\mathbf{u}^{(p,p+1)}$ on this step impacts the whole vector x, and the orthogonality of the frame matrix $X_{\neq p,p+1}$ ensures that the Frobenius norms of the local perturbation (to $\mathbf{u}^{(p,p+1)}$) and the global perturbation (to x) are the same.

There are several ways to choose the rank k'_p of the decomposition (3.7). We may

- define the upper *rank bound* $k_p^{\max}$, which will be hit almost certainly if $k_p^{\max} \leqslant \min(k_{p-1} n_p, n_{p+1} k_{p+1})$;
- set the relative *accuracy level* ε, and find the lowest rank k'_p that provides the approximation (3.7) such that $\|U^{(p,p+1)} - \tilde{U}^{(p,p+1)}\| \leqslant \varepsilon \|U^{(p,p+1)}\|$; or
- use both requirements simultaneously.

After the decomposition is done, the TT cores $\mathbf{x}^{(p)}$ and $\mathbf{x}^{(p+1)}$ are replaced by $\mathbf{u}^{(p)}$ and $\mathbf{u}^{(p+1)}$, and the TT rank k_p is substituted by k'_p. The tensor structure changes in each step, and further optimization is carried out over the updated *tensor manifold*. This generally speeds up the convergence of $\tau(\bar{\mathbf{x}})$ toward the solution $x_\star$ (or to some local minimum of $\mathtt{J}_{A,b}$) but makes the process more difficult to analyze.

4. SD schemes. In this section we discuss several variants of the SD algorithm. An iterative algorithm starts from the *initial guess* x_0 and generates a sequence of *solutions* $x_1, x_2, \dots$ s.t. $x_i \to x_\star$. We say that the algorithm converges *geometrically* if $\|x_\star - x_i\| \leqslant \|x_\star - x_0\| \Omega^i$, where $\Omega < 1$ is the *convergence rate*, which does not depend on the initial vector x_0.

The SD is a one-step algorithm, i.e., it uses the same rule to compute x_{i+1} from x_i for all iterations i. Therefore, we can study just one iteration and obtain the uniform bound $\|x_\star - x_1\| / \|x_\star - x_0\| \leqslant \Omega$ for all x_0. It is convenient for us to remove the index of iterations and consider only one SD step with the initial guess $t = x_0$ and

result $x = x_1$. The errors before and after the SD step are denoted as $c = x_\star - t$ and $d = x_\star - x$. The lower subindex will be used to enumerate the *microsteps* related to the optimization over TT cores; cf. Algorithm 1.

4.1. Basic definitions and convergence analysis. The SD step minimizes the energy function (1.1) in the gradient direction as follows:

$$r = -\operatorname{grad} \mathtt{J}_{A,b}(t) = b - At, \tag{4.1}$$
$$h = \arg\min_{h'} \mathtt{J}_{A,b}(t + rh') = \frac{(r,r)}{(r,Ar)}.$$

The result $x = t + rh$ satisfies the *Galerkin condition* $(r, b - Ax) = 0$. To analyze the *progress* we need to compare $\mathtt{J}_{A,b}(t) = \|x_\star - t\|_A^2$ with $\mathtt{J}_{A,b}(x) = \|x_\star - x\|_A^2$. Starting from the relation between errors $d = x_\star - x = x_\star - t - rh = c - rh$, we write

$$d = c - \frac{r(r,r)}{(r,Ar)} = (I - \mathcal{P}_{A,r})c, \qquad \text{where} \quad \mathcal{P}_{A,r} = \frac{rr^*A}{r^*Ar}, \tag{4.2}$$
$$\frac{\|x_\star - x\|_A^2}{\|x_\star - t\|_A^2} = \frac{(c, (I - \mathcal{P}_{A,r})^* A (I - \mathcal{P}_{A,r})c)}{(c, Ac)} = 1 - \frac{(c, \mathcal{P}_{A,r}c)_A}{(c,c)_A} = \omega_{r,r}^2.$$

Here $\mathcal{P}_{A,r}$ is the A-orthogonal projector on r, and the first line shows the *monotone decrease* $\mathtt{J}_{A,b}(x) = \|x_\star - x\|_A^2 \leqslant \|x_\star - t\|_A^2 = \mathtt{J}_{A,b}(t)$. The *progress* $\omega_{r,r}$ depends on the current iterate—for example, $\omega_{r,r} = 0$ (the SD converges instantly) for such t that $c = x_\star - t$ is an eigenvector of A. However, $\omega_{r,r}$ can be uniformly bounded from above using the *Kantorovich inequality* [28] as follows:

$$\omega_{r,r} = \sqrt{1 - \frac{(r,r)}{(r,Ar)}\frac{(r,r)}{(r,A^{-1}r)}} \leqslant \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)} = \frac{\kappa(A) - 1}{\kappa(A) + 1} = \Omega < 1. \tag{4.3}$$

Remark 4.1. The Kantorovich inequality is sharp. The equality is achieved for $r = u_{\min}(A) + u_{\max}(A)$, where $u_{\min}(A)$ and $u_{\max}(A)$ denote the normalized eigenvectors of A for the minimum and the maximum eigenvalue.

The upper bound Ω is an a priori estimate of the *convergence rate* of the SD algorithm. In the following analysis we assume that the spectral bounds of A are known or estimated and therefore Ω is known as well.

4.2. Inexact SD algorithm. In many large-scale problems, including those of high dimension, the residual $r = b - At$ is difficult to compute exactly, and an approximate direction $z \approx r$ is used. Consider the *inexact* SD step

$$x = t + zh, \qquad h = \arg\min_{h'} \mathtt{J}_{A,b}(t + zh') = \frac{(z,r)}{(z,Az)}, \tag{4.4}$$

which provides the error $\tilde{d} = x_\star - x = (I - \mathcal{P}_{A,z})c$. Since $(z,r) = (z,b) - (Az,t)$, the computation of h requires z and Az, but not r. Comparison with the exact SD (4.1) gives $\tilde{d} - d = -(\mathcal{P}_{A,z} - \mathcal{P}_{A,r})\,c$, where $\mathcal{P}_{A,z} - \mathcal{P}_{A,r}$ can be estimated for small $\|r - z\|$ in the spirit of [56]. Even better analysis is provided in [44], where a bound on the *angle* $\angle(r,z)$ between r and z is assumed instead of a bound on the norm of $r - z$.

PROPOSITION 4.2 (convergence of the inexact SD [44]). *Given an SPD linear system $Ax = b$ and an initial vector t, consider the residual $r = b - At$ and a vector z s.t. $\angle(r,z) \leqslant \theta < \pi/2$. The inexact SD step* (4.4) *provides the following progress:*

$$\omega_{r,z} = \frac{\|x_\star - x\|_A}{\|x_\star - t\|_A} \leqslant \frac{\tilde{\kappa} - 1}{\tilde{\kappa} + 1} = \tilde{\Omega} < 1, \qquad \tilde{\kappa} = \kappa\frac{1 + \sin\theta}{1 - \sin\theta}, \quad \kappa = \kappa(A). \tag{4.5}$$

Proof. Directly from $\tilde{d} = (I - \mathcal{P}_{A,z})c$ we obtain

$$\omega_{r,z}^2 = \frac{\|x_\star - x\|_A^2}{\|x_\star - t\|_A^2} = \frac{\|\tilde{d}\|_A^2}{\|c\|_A^2} = 1 - \frac{|(z,r)|^2}{(z,Az)(r,A^{-1}r)}. \tag{4.6}$$

To bound $\omega_{r,z}$ from above similarly to (4.3), we use the generalization of the Kantorovich inequality from [5, Corollary IV], which is written as follows:

$$\frac{(z,Az)(r,A^{-1}r)}{\|z\|^2\|r\|^2} \leqslant \frac{\left((\kappa+1)+(\kappa-1)\sin\theta\right)^2}{4\kappa}. \tag{4.7}$$

Together with $\cos\angle(r,z) \geqslant \cos\theta$ and identity $\frac{1+\sin\theta}{\cos\theta} = \frac{\cos\theta}{1-\sin\theta} = \sqrt{\frac{1+\sin\theta}{1-\sin\theta}}$ it gives

$$\frac{|(z,r)|^2}{(z,Az)(r,A^{-1}r)} \geqslant \frac{4\kappa\cos^2\theta}{\left(\kappa(1+\sin\theta)+(1-\sin\theta)\right)^2} = \left(\frac{2}{\tilde{\kappa}^{1/2}+\tilde{\kappa}^{-1/2}}\right)^2,$$

where $\tilde{\kappa}$ is defined in (4.5). Plugging this into (4.6), we complete the proof. ∎

Remark 4.3. For $r = z$ inequality (4.7) is the Kantorovich inequality (4.3).

In tensor product algorithms proposed in this paper we assume that

$$r = z + \delta, \qquad (z,\delta) = 0, \qquad \|\delta\| \leqslant \varepsilon\|r\|, \tag{4.8}$$

with some $\varepsilon < 1$, that is controlled a priori or estimated a posteriori. This gives $\angle(r,z) < \pi/2$, and $\sin\angle(r,z) = \|\delta\|/\|r\| \leqslant \varepsilon$. Plugging this into (4.5), we obtain $\tilde{\kappa}/\kappa \leqslant (1+\varepsilon)/(1-\varepsilon)$, and for small ε it follows $\tilde{\kappa}/\kappa \lesssim 1 + 2\varepsilon$, i.e., the perturbation of a residual has almost no effect on convergence.

Remark 4.4. For $\Omega \approx 1$ we may want to control $1 - \tilde{\Omega}$ as follows:

$$1 \leqslant \frac{1-\Omega}{1-\tilde{\Omega}} \leqslant \frac{\frac{1+\varepsilon}{1-\varepsilon}\kappa + 1}{\kappa+1} = \frac{1+\varepsilon\frac{\kappa-1}{\kappa+1}}{1-\varepsilon} \leqslant 3 \quad \text{for} \quad \varepsilon \leqslant \frac{1}{2}.$$

This shows that

- for a given Ω we may provide any desired $\tilde{\Omega} \geqslant \Omega$ choosing a sufficiently small ε, and
- even a rough approximation with the relative accuracy $\varepsilon = 1/2$ provides a satisfactory good convergence bound $\tilde{\Omega}$.

In the following we assume that Ω is known and ε is chosen a priori or controlled during the computations in such a way that $\tilde{\Omega}$ in (4.5) is a reasonable a priori bound for the convergence rate of the inexact SD algorithm.

4.3. Wide SD algorithm. Given a matrix Z of a suitable size and full column rank, we consider a *wide* SD step $x = t + Zv$ with

$$v = \arg\min_{v'} \mathtt{J}_{A,b}(t + Zv') = (Z^*AZ)^{-1}Z^*r. \tag{4.9}$$

Compared to the standard SD step (4.1), or to the inexact step (4.4), the optimization (4.9) is performed over vectors in the *wider* manifold $t + \operatorname{span} Z$, hence the name. Errors before and after the wide SD step (4.9) are related as $d = (I - \mathcal{P}_{A,Z})c$, where the A-orthogonal projector $\mathcal{P}_{A,Z}$ on the subspace $\operatorname{span} Z$ is defined as follows:

$$\mathcal{P}_{A,Z} = Z(Z^*AZ)^{-1}Z^*A. \tag{4.10}$$

ALGORITHM 3. ALS(SD) for SPD linear system $Ax = b$ (single iteration).

Require: Initial guess $t = \tau(\bar{\mathbf{t}})$ with $\mathtt{k}(\bar{\mathbf{t}}) = k$, accuracy ε, and/or rank bounds $k^{\max}$.
Ensure: Updated vector $x = \tau(\bar{\mathbf{x}})$ with $\mathtt{k}(\bar{\mathbf{x}}) = k' \leqslant k + k^{\max}$, if $k^{\max}$ is given.
1: Approximate $b - At = r \approx z = \tau(\bar{\mathbf{z}})$ s.t. $\|r - z\| \leqslant \varepsilon\|r\|$ and/or $\mathtt{k}(\bar{\mathbf{z}}) \leqslant k^{\max}$.
2: Compute $\bar{\mathbf{x}} = \bar{\mathbf{t}} + \bar{\mathbf{z}}$ by (4.13).
3: Apply one half-sweep of the one-site DMRG optimization (Algorithm 1) to $\bar{\mathbf{x}}$.

The progress of the wide SD step (4.9) is written as follows:

$$\omega_{r,Z}^2 = \frac{\|x_\star - x\|_A^2}{\|x_\star - t\|_A^2} = 1 - \frac{(c, \mathcal{P}_{A,Z} c)_A}{(c,c)_A}. \tag{4.11}$$

LEMMA 4.5. *If $z \in \operatorname{span} Z$, then $\omega_{r,Z} \leqslant \omega_{r,z}$, i.e., the progress* (4.11) *of the wide SD step* (4.9) *is not worse than the progress* (4.6) *of the inexact step* (4.4).

Proof. The proof follows immediately from $\|\mathcal{P}_{A,Z} c\|_A \geqslant \|\mathcal{P}_{A,z} c\|_A$. ∎

COROLLARY 4.6. *If $\angle(r, Z) = \min_{z \in \operatorname{span} Z} \angle(r,z) \leqslant \theta < \pi/2$, then*

$$\omega_{r,Z} \leqslant \frac{\tilde{\kappa} - 1}{\tilde{\kappa} + 1} = \tilde{\Omega} < 1, \qquad \tilde{\kappa} = \kappa \frac{1 + \sin\theta}{1 - \sin\theta}, \quad \kappa = \kappa(A). \tag{4.12}$$

Proof. Apply Lemma 4.5 for $z \in \operatorname{span} Z$ s.t. $\angle(r,z) = \angle(r,Z) \leqslant \theta$ and estimate $\omega_{r,z}$ by (4.5). ∎

Remark 4.7. The inequality in Lemma 4.5 is sharp. For example, $\omega_{r,Z} = \omega_{r,z}$ for $Z = \begin{bmatrix} z & s \end{bmatrix}$ with such s that $(z,s)_A = 0$ and $(c,s)_A = 0$.

Proof. For $(z,s)_A = 0$ it holds $\mathcal{P}_{A,Z} = \mathcal{P}_{A,z} + \mathcal{P}_{A,s}$, and the first condition $(z,s)_A = 0$ gives $\|\mathcal{P}_{A,Z} c\|_A^2 = \|\mathcal{P}_{A,z} c\|_A^2 + \|\mathcal{P}_{A,s} c\|_A^2$. The condition $(c,s)_A = 0$ annihilates the second term, which proves the sharpness. ∎

4.4. SD and one-site optimization in higher dimensions. For a high-dimensional SPD linear system $Ax = b$ we combine the SD step with the one-site optimization, as proposed by Algorithm 3. Starting from an initial guess $t = \tau(\bar{\mathbf{t}})$ in the TT format (2.2), we approximate the current residual $r = b - At$ in the TT format $r \approx z = \tau(\bar{\mathbf{z}})$, e.g., by the TT-SVD algorithm [45]. In TT-SVD we can either

- use accuracy criteria $\|r - z\| \leqslant \varepsilon\|r\|$ with ε requested a priori and obtain quasi-optimal TT ranks $\mathtt{k}(\bar{\mathbf{z}})$ adaptively, or
- set the upper rank bounds $k^{\max} = (k_1^{\max}, \dots, k_{d-1}^{\max})$ and obtain the approximation $z = \tau(\bar{\mathbf{z}})$ with $\mathtt{k}(\bar{\mathbf{z}}) \leqslant k^{\max}$ (i.e., $\mathtt{k}_p(\bar{\mathbf{z}}) \leqslant k_p^{\max}$ for $p = 1, \dots, d-1$) and a quasi-optimal accuracy $\|r - z\| \leqslant \varepsilon\|r\|$, where ε is computed a posteriori.

When z is computed, the SD step is applied to compute $x = t + zh$. For $t = \tau(\bar{\mathbf{t}})$ and $s = \tau(\bar{\mathbf{s}})$ the sum $x = t + s$ has the TT representation $x = \tau(\bar{\mathbf{x}})$ with

$$\begin{aligned} \mathbf{x}^{(1)}(i_1) &:= \begin{bmatrix} \mathbf{t}^{(1)}(i_1) & \mathbf{s}^{(1)}(i_1) \end{bmatrix}, \\ \mathbf{x}^{(p)}(i_p) &:= \begin{bmatrix} \mathbf{t}^{(p)}(i_p) & 0 \\ 0 & \mathbf{s}^{(p)}(i_p) \end{bmatrix}, \quad \mathbf{x}^{(d)}(i_d) := \begin{bmatrix} \mathbf{t}^{(d)}(i_d) \\ \mathbf{s}^{(d)}(i_d) \end{bmatrix}, \end{aligned} \tag{4.13}$$

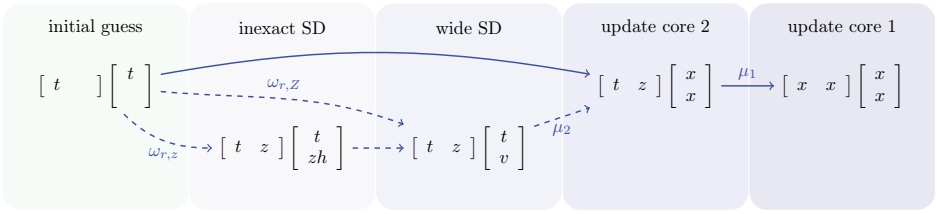

FIG. 4. *Illustration of the ALS(SD) algorithm in two dimensions.*

where $p = 2, \dots, d-1$. We will denote this TT as $\bar{\mathbf{x}} = \bar{\mathbf{t}} + \bar{\mathbf{s}}$. Other representations (probably with smaller TT ranks) can also exist for $x = t + s$.

The inexact SD step guarantees the progress (4.6) with the upper bound (4.5). To further improve the target function, one half-sweep of the one-site DMRG optimization Algorithm 1 is applied to x. We immediately arrive at the following theorem.

THEOREM 4.8. *A single iteration of Algorithm* 3 *with* $\varepsilon < 1$ *provides the progress*

$$\omega_{\mathrm{ALS(SD)}} = \frac{\|x_\star - x\|_A}{\|x_\star - t\|_A} \leqslant \omega_{r,z}\, \mu_1 \cdots \mu_d \leqslant \tilde{\Omega} < 1, \tag{4.14}$$

where $\omega_{r,z}$ *is defined by* (4.6) *and bounded by* (4.5)*, and* $\mu_p \leqslant 1$, $p = 1, \dots, d$, *denote additional progress obtained by the optimization over the TT core* $\mathbf{x}^{(p)}$.

In the two-dimensional case Algorithm 3 consists (see Figure 4) of the following steps:

1. For a given initial guess t approximate $z \approx r = b - At$ with a desired accuracy or with bounded ranks.
2. Expand the TT cores $\mathbf{t}^{(p)}$ with the residual information $\mathbf{z}^{(p)}$ by (4.13).
3. Find the optimal step size h, and multiply $\mathbf{z}^{(d)} := \mathbf{z}^{(d)} h$.
4. Optimize over the bottom part of the last TT core, i.e., the entries of $\mathbf{z}^{(d)}$.
5. Optimize further over the whole TT core $\mathbf{x}^{(d)}$, and proceed for $\mathbf{x}^{(d-1)}, \dots, \mathbf{x}^{(1)}$.

steps 3 and 4 are not technically present in the computations (they are covered by step 5). They allow us to estimate the progress of Algorithm 3 compared to the inexact SD (4.4) (step 3) or wide SD (4.9) (step 4). The latter may be used to improve the estimate (4.14), substituting $\omega_{r,z}$ with $\omega_{r,Z_{\neq d}} \leqslant \omega_{r,z}$, but at the moment it is not clear how to quantify the difference between these estimates.

Note that values $\omega_{r,z}$ and $\omega_{r,Z}$ have the *global* upper bound (4.5), which does not depend on the current iterate (see also Remark 4.4). Currently the values $\mu_1, \dots, \mu_d$ in (4.14) can be estimated only *locally*, i.e., near the extremum of $\mathtt{J}_{A,b}(x)$; see [49].

5. Alternating minimal energy algorithm. The main result of this paper is Algorithm 4, the AMEn algorithm. Similarly to Algorithm 3, it expands the search space with the gradient information, trying to improve the convergence of the optimization steps. However, the TT cores of the residual which are used in the ALS(SD) algorithm become "outdated" (i.e., do not match the gradient precisely) after several updates of the TT cores of the solution. In the AMEn we always use the "fresh" residual that accounts for the latest changes to the solution in microsteps and better approximates the current gradient. The implementation of the AMEn requires only *local* operations (involving one or two cores at a time), while the ALS(SD) has the *global* basis enrichment step; see Figure 5.

ALGORITHM 4. AMEn for SPD linear system $Ax = b$ (single iteration).

Require: Initial guess $t = \tau(\bar{\mathbf{t}})$ with $\mathtt{k}(\bar{\mathbf{t}}) = k$, accuracy ε, and/or rank bounds $k^{\max}$.
Ensure: Updated vector $x = \tau(\bar{\mathbf{x}})$ with $\mathtt{k}(\bar{\mathbf{x}}) = k' \leqslant k + k^{\max}$ if $k^{\max}$ is given.

Form $A_1 = T^*_{\neq 1} A T_{\neq 1}$, $b_1 = T^*_{\neq 1} b$, and solve $A_1 u^{(1)} = b_1$.
Let $u = \tau(\mathbf{u}^{(1)}, \mathbf{t}^{(2)}, \ldots, \mathbf{t}^{(d)})$ and $r = b - Au$.
Find $\mathbf{z}^{(1)}$ from $\bar{\mathbf{z}}$, s.t. $r \approx z = \tau(\bar{\mathbf{z}})$, where $\|z - r\| \leqslant \varepsilon \|r\|$ and/or $\mathtt{k}_1(\bar{\mathbf{z}}) \leqslant k_1^{\max}$.
Expand the basis $\mathbf{x}^{(1)} := \begin{bmatrix} \mathbf{u}^{(1)} & \mathbf{z}^{(1)} \end{bmatrix}$, $\mathbf{t}^{(2)} := \begin{bmatrix} \mathbf{t}^{(2)} \\ 0 \end{bmatrix}$
Consider the $(d-1)$-dimensional system $A_{\geqslant 2} x^{\geqslant 2} = b_{\geqslant 2}$ given by (5.1)
if $d = 2$ **then**
    Form the system (5.1) and solve it directly
else
    Solve the system (5.1) by the AMEn, obtain $x^{\geqslant 2} = \operatorname{vec} \tau(\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(d)})$.
end if
return $x = \tau(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(d)})$

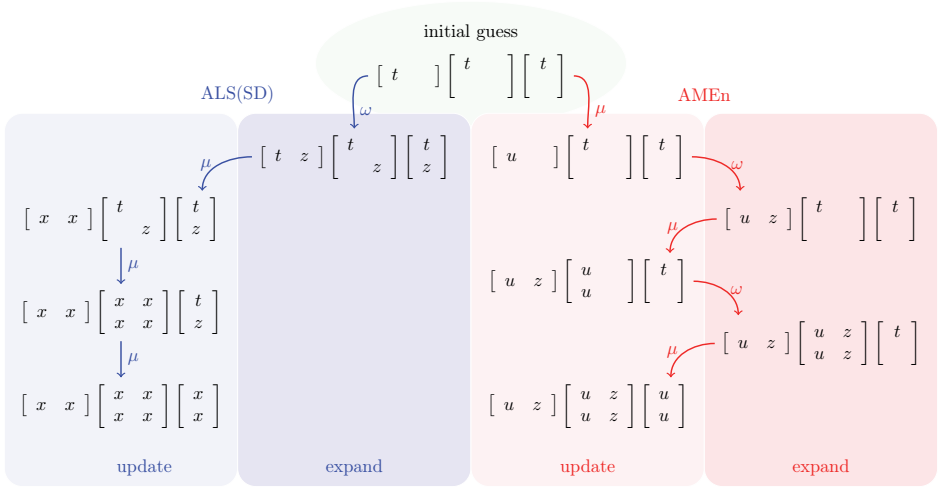

FIG. 5. *Schematic representation of ALS(SD) and AMEn algorithms.*

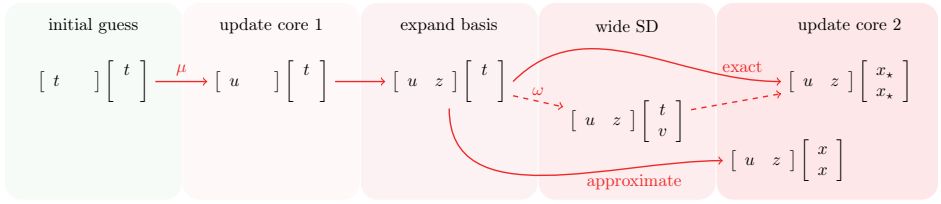

FIG. 6. *Illustration of the AMEn algorithm in two dimensions.*

5.1. Convergence analysis in two dimensions. As shown in Figure 6, AMEn (Algorithm 4) consists of the following steps:

1. Optimize the first TT core $\mathbf{t}^{(1)} \to \mathbf{u}^{(1)}$.
2. Expand the TT core $\mathbf{x}^{(1)} = \begin{bmatrix} \mathbf{u}^{(1)} & \mathbf{z}^{(1)} \end{bmatrix}$ by the first TT core of the residual.

3. Reduce the system $Ax = b$ to $A_{\geqslant 2}x^{\geqslant 2} = b_{\geqslant 2}$ of lower dimension with

$$A_{\geqslant 2} = X_1^* A X_1, \quad b_{\geqslant 2} = X_1^* b, \qquad X_1 = X^{|1\rangle} \otimes I_{n_2 \cdots n_d} \in \mathbb{C}^{(n_1 \cdots n_d)\times(k_1 n_2 \cdots n_d)}. \tag{5.1}$$

4. Solve $A_{\geqslant 2}x^{\geqslant 2} = b_{\geqslant 2}$ exactly (if $d = 2$), or approximately with the AMEn.

For $d = 2$ the convergence is estimated by the one of the wide SD (4.9) similarly to the ALS(SD) algorithm. Indeed, if $x_\star^{\geqslant 2} = A_{\geqslant 2}^{-1} b_{\geqslant 2} = (X_1^* A X_1)^{-1} X_1^* b$ is computed, then $x = X_1 x_\star^{\geqslant 2} = \mathcal{P}_{A,X_1} x_\star$, and $x_\star - x = (I - \mathcal{P}_{A,X_1})x_\star$. Since $u \in \operatorname{span} X_1$, we can write $x_\star - x = (I - \mathcal{P}_{A,X_1})(x_\star - u) = (I - \mathcal{P}_{A,X_1})c$ with $c = x_\star - u$. It gives

$$\frac{\|x_\star - x\|_A^2}{\|x_\star - u\|_A^2} = 1 - \frac{(c, \mathcal{P}_{A,X_1} c)_A}{(c,c)_A} = \omega_{r,X_1}^2, \tag{5.2}$$

where ω_{r,X_1} is defined by (4.11). Since $z \in \operatorname{span} Z_1 \subset \operatorname{span} X_1$ for $Z_1 = Z^{|1\rangle} \otimes I_{n_2 \cdots n_d}$, it holds that $\omega_{r,X_1} \leqslant \omega_{r,Z_1} \leqslant \omega_{r,z}$ with upper bounds provided by (4.5) and (4.12).

5.2. Convergence analysis in higher dimensions. In higher dimensions we should also account for the fact that $A_{\geqslant 2}x^{\geqslant 2} = b_{\geqslant 2}$ is not solved exactly, i.e., the solution $x^{\geqslant 2}$ delivered by a recursive call to the AMEn differs from $x_\star^{\geqslant 2} = A_{\geqslant 2}^{-1} b_{\geqslant 2}$. We expand $x_\star - x = x_\star - X_1 x_\star^{\geqslant 2} + X_1 x_\star^{\geqslant 2} - X_1 x^{\geqslant 2} = (I - \mathcal{P}_{A,X_1})x_\star + X_1(x_\star^{\geqslant 2} - x^{\geqslant 2})$, where two terms are A-orthogonal. It gives

$$\begin{aligned}\frac{\|x_\star - x\|_A^2}{\|x_\star - u\|_A^2} &= \omega_{r,X_1}^2 + \frac{\|X_1(x_\star^{\geqslant 2} - x^{\geqslant 2})\|_A^2}{\|x_\star - u\|_A^2} \\ &= \omega_{r,X_1}^2 + \frac{\|x_\star^{\geqslant 2} - x^{\geqslant 2}\|_{A_{\geqslant 2}}^2}{\|x_\star^{\geqslant 2} - t^{\geqslant 2}\|_{A_{\geqslant 2}}^2} \, \frac{\|X_1(x_\star^{\geqslant 2} - t^{\geqslant 2})\|_A^2}{\|x_\star - u\|_A^2},\end{aligned}$$

where $t^{\geqslant 2} = \operatorname{vec} \mathbf{t}^{(2:d)}$ is the initial guess in the reduced problem. Similarly to the case $d = 2$, it holds that $X_1 x_\star^{\geqslant 2} = \mathcal{P}_{A,X_1} x_\star$. Also, $X_1 t^{\geqslant 2} = \tau(\mathbf{x}^{(1)}, \mathbf{t}^{(2)}, \ldots, \mathbf{t}^{(d)}) = u = \mathcal{P}_{A,X_1} u$, since $u \in \operatorname{span} X_1$. We plug $X_1(x_\star^{\geqslant 2} - t^{\geqslant 2}) = \mathcal{P}_{A,X_1}(x_\star - u) = \mathcal{P}_{A,X_1} c$ into the previous equation and compare it to (5.2). This gives the following estimate:

$$\frac{\|x_\star - x\|_A^2}{\|x_\star - u\|_A^2} = \omega_{r,X_1}^2 + (1 - \omega_{r,X_1}^2) \, \frac{\|x_\star^{\geqslant 2} - x^{\geqslant 2}\|_{A_{\geqslant 2}}^2}{\|x_\star^{\geqslant 2} - t^{\geqslant 2}\|_{A_{\geqslant 2}}^2}. \tag{5.3}$$

Since the solution $x^{\geqslant 2}$ is computed by the same AMEn algorithm, we can quantify the last term in (5.3) in a recurrent way. One half-sweep can be seen, therefore, as a sequence of embedded *reduced* problems $A_{\geqslant p}x^{\geqslant p} = b_{\geqslant p}$, for $p = 1, \ldots, d$, where

$$\begin{aligned} A_{\geqslant p} &= X_{<p}^* A X_{<p} \in \mathbb{C}^{(k_{p-1} n_p \cdots n_d)\times(k_{p-1} n_p \cdots n_d)}, & b_{\geqslant p} &= X_{<p}^* b, \\ X_{<p} &= X^{|1:p-1\rangle} \otimes I_{n_p \cdots n_d} \in \mathbb{C}^{(n_1 \cdots n_d)\times(k_{p-1} n_p \cdots n_d)}, & X_{<1} &= I_{n_1 \cdots n_d}. \end{aligned} \tag{5.4}$$

For each reduced problem we start from the initial guess $t^{\geqslant p} = \operatorname{vec} \tau(\mathbf{t}^{(p)}, \ldots, \mathbf{t}^{(d)})$, update its first core and obtain $u_p = \operatorname{vec} \tau(\mathbf{u}^{(p)}, \mathbf{t}^{(p+1)}, \ldots, \mathbf{t}^{(d)})$, expand it with the core $\mathbf{z}^{(p)}$ of the approximate residual $r_p = b_{\geqslant p} - A_{\geqslant p} u_p \approx z_p = \operatorname{vec} \tau(\mathbf{z}^{(p)}, \ldots, \mathbf{z}^{(d)})$, and reduce the problem to the one of smaller dimension. We define

$$\mu_p^2 = \frac{\|x_\star^{\geqslant p} - u_p\|_{A_{\geqslant p}}^2}{\|x_\star^{\geqslant p} - t^{\geqslant p}\|_{A_{\geqslant p}}^2} \leqslant 1, \qquad \omega_p^2 = 1 - \frac{(c_p, \mathcal{P}_{A_{\geqslant p}, X_p} c_p)_{A_{\geqslant p}}}{(c_p, c_p)_{A_{\geqslant p}}} < 1, \tag{5.5}$$

where $x_\star^{\geqslant p} = A_{\geqslant p}^{-1} b_{\geqslant p}$, $c_p = x_\star^{\geqslant p} - u_p$, and $X_p = X^{|p\rangle} \otimes I_{n_{p+1} \cdots n_d}$, $X^{|p\rangle} = \begin{bmatrix} U^{|p\rangle} & Z^{|p\rangle} \end{bmatrix}$.

LEMMA 5.1. *A single iteration (half-sweep) of Algorithm* 4 *provides the progress*

$$\omega^2_{\mathrm{AMEn}} = \frac{\|x_\star - x\|_A^2}{\|x_\star - t\|_A^2} = \sum_{p=1}^{d-1} \omega_p^2 \prod_{j=1}^{p-1}(1-\omega_j^2) \prod_{j=1}^{p} \mu_j^2. \tag{5.6}$$

Proof. For $d = 2$ we use (5.2) and the first definition in (5.5) to establish

$$\frac{\|x_\star - x\|_A}{\|x_\star - t\|_A} = \frac{\|x_\star - u\|_A}{\|x_\star - t\|_A} \frac{\|x_\star - x\|_A}{\|x_\star - u\|_A} = \mu_1\omega_1, \tag{5.7}$$

which gives the base of induction. Now we suppose that (5.6) holds in $d-1$ dimensions and prove it for d dimensions, which constitutes the induction step. From (5.3),

$$\frac{\|x_\star - x\|_A^2}{\|x_\star - t\|_A^2} = \mu_1^2 \left(\omega_1^2 + (1-\omega_1^2) \frac{\|x_\star^{\geqslant 2} - x^{\geqslant 2}\|^2_{A_{\geqslant 2}}}{\|x_\star^{\geqslant 2} - t^{\geqslant 2}\|^2_{A_{\geqslant 2}}} \right),$$

where by the assumption of induction

$$\frac{\|x_\star^{\geqslant 2} - x^{\geqslant 2}\|^2_{A_{\geqslant 2}}}{\|x_\star^{\geqslant 2} - t^{\geqslant 2}\|^2_{A_{\geqslant 2}}} = \sum_{p=2}^{d-1} \omega_p^2 \prod_{j=2}^{p-1}(1-\omega_j^2) \prod_{j=2}^{p} \mu_j^2.$$

Plugging this into the last equation, we obtain (5.6) and complete the proof. ∎

5.3. Convergence rate bounds. In terms of definition (4.11), $\omega_p = \omega_{r_p, X_p}$, and since $z_p \in \operatorname{span} Z_p \subset \operatorname{span} X_p$, the upper bound (4.12) applies as follows:

$$\omega_p \leqslant \frac{\tilde\kappa_p - 1}{\tilde\kappa_p + 1} = \tilde\Omega_p < 1, \qquad \tilde\kappa_p = \kappa_p \frac{1+\sin\theta_p}{1-\sin\theta_p}, \quad \kappa_p = \kappa(A_{\geqslant p}), \tag{5.8}$$

where $\theta_p = \angle(r_p, X_p) \leqslant \angle(r_p, z_p)$. If the accuracy criterion $\|r_p - z_p\| \leqslant \varepsilon \|r_p\|$ is used for all approximation steps (Algorithm 4, line 3), then $\sin\theta_p \leqslant \varepsilon$ with ε known a priori. If the rank bound criterion is applied, θ_p is estimated a posteriori by $\angle(r_p, z_p)$. Since

$$A_{\geqslant p+1} = X^*_{<p+1} A X_{<p+1} = (X^{|p\rangle} \otimes I_{n_{p+1}\cdots n_d})^* A_{\geqslant p} (X^{|p\rangle} \otimes I_{n_{p+1}\cdots n_d}), \tag{5.9}$$

the condition numbers $\kappa(A_{\geqslant p})$ are related as follows:

$$\kappa(A) = \kappa(A_{\geqslant 1}) \geqslant \kappa(A_{\geqslant 2}) \geqslant \cdots \geqslant \kappa(A_{\geqslant p-1}) \geqslant \kappa(A_{\geqslant p}) \geqslant \cdots \geqslant \kappa(A_{\geqslant d}). \tag{5.10}$$

Plugging $\theta_p \leqslant \theta = \max_p \angle(r_p, z_p)$ and $\kappa_p \leqslant \kappa = \kappa(A)$ into (5.8), we obtain

$$\omega_p \leqslant \tilde\Omega_p = \frac{\tilde\kappa_p - 1}{\tilde\kappa_p + 1} \leqslant \frac{\tilde\kappa - 1}{\tilde\kappa + 1} = \tilde\Omega < 1, \qquad \tilde\kappa = \kappa \frac{1+\sin\theta}{1-\sin\theta}, \quad \kappa = \kappa(A), \tag{5.11}$$

which gives a uniform upper bound for all ω_p.

THEOREM 5.2. *The AMEn (Algorithm* 4*) is convergent if the approximation error allowed in line* 3 *satisfies* $\theta = \max_{p=1,\ldots,d-1} \angle(r_p, z_p) < \pi/2$. *The convergence rate* (5.6) *is bounded from above, s.t. the following inequality holds:*

$$1 - \omega^2_{\mathrm{AMEn}} \geqslant (1 - \tilde\Omega^2)^{d-1}, \qquad \tilde\Omega = \frac{\tilde\kappa - 1}{\tilde\kappa + 1}, \qquad \tilde\kappa = \kappa(A) \frac{1+\sin\theta}{1-\sin\theta}. \tag{5.12}$$

Proof. Note that the right-hand side in (5.6) is monotonous for $0 \leqslant \omega_p \leqslant \tilde{\Omega}$ and $0 \leqslant \mu_p \leqslant 1$, $p = 1, \dots, d-1$. By plugging the upper bounds $\mu_p = 1$ and $\omega_p = \tilde{\Omega}$ from (5.11) into (5.6), we prove (5.12). ∎

The upper bound for ω_{AMEn} provided by (5.12) is much weaker than the upper bound for $\omega_{\mathrm{ALS(SD)}}$ given by (4.14). If $\tilde{\Omega} \approx 1$, the value of $(1 - \tilde{\Omega}^2)^{d-1}$ decays exponentially fast with d. Interestingly, the practical convergence of the AMEn is usually better than that of the ALS(SD) algorithm, as shown in section 7.

6. Practical aspects. In this section we discuss the implementation details that improve the performance of Algorithms 3 and 4.

6.1. Computation of local systems. In each optimization step we create and solve the local system $A_p u^{(p)} = b_p$ by (3.2). As shown in Figure 3, it can be assembled from the TT cores of $A = \tau(\bar{\mathbf{A}})$, $x = \tau(\bar{\mathbf{x}})$ and $b = \tau(\bar{\mathbf{b}})$ in $\mathcal{O}(d)$ time. However, only small corrections, similar to the one in (5.9), are required to update the local systems between microsteps $p = 1, \dots, d$. The total complexity of these steps for one iteration (half-sweep) is linear in d; see [12] for more details.

6.2. Approximation of the residual. In ALS(SD) (Algorithm 3, line 1) the exact residual $r = b - At = \tau(\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(d)})$ is a TT sum (4.13) of two TT formats $\bar{\mathbf{b}} - \bar{\mathbf{A}}\bar{\mathbf{t}}$, where $\bar{\mathbf{A}}\bar{\mathbf{t}} = \bar{\mathbf{s}} = (\mathbf{s}^{(1)}, \dots, \mathbf{s}^{(d)})$ with $\mathbf{s}^{(p)}(i_p) = \sum_{j_p} \mathbf{A}^{(p)}(i_p, j_p) \otimes \mathbf{t}^{(p)}(j_p)$. The TT ranks $\mathtt{k}(\bar{\mathbf{r}}) = \mathtt{k}(\bar{\mathbf{b}}) + \mathtt{k}(\bar{\mathbf{A}})\,\mathtt{k}(\bar{\mathbf{t}})$ are usually too large to work with this representation efficiently, and r is approximated by $z = \tau(\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(d)})$ with smaller TT ranks, using, e.g., the TT-SVD algorithm [53, 45] with complexity $\mathcal{O}(dnk^6)$, where $k^2 = \max \mathtt{k}(\bar{\mathbf{r}})$. Since one approximation step is required per iteration, the complexity of the ALS(SD) algorithm is linear with d.

In the AMEn there is one approximation per microstep (Algorithm 4, line 3). Fortunately, only the first TT core of the approximate residual $z_p \approx r_p = b_{\geqslant p} - A_{\geqslant p} u_p$ is required, and computation can be organized in such a way that the total complexity is still linear in d. Indeed, since $u_p = \operatorname{vec} \tau(\mathbf{u}^{(p)}, \mathbf{t}^{(p+1)}, \dots, \mathbf{t}^{(d)})$, where $\mathbf{t}^{(p+1)}, \dots, \mathbf{t}^{(d)}$ are the TT cores of the initial guess $t = \tau(\bar{\mathbf{t}})$, the *exact* residual has the TT representation $r_p = \operatorname{vec} \tau(\hat{\mathbf{r}}^{(p)}, \mathbf{r}^{(p+1)}, \dots, \mathbf{r}^{(d)})$ with the same TT cores $\mathbf{r}^{(p+1)}, \dots, \mathbf{r}^{(d)}$ as the ones in $\tau(\bar{\mathbf{r}}) = r = b - At$. The TT representation $\bar{\mathbf{r}}$ may be precomputed before each AMEn iteration with the right TT orthogonality ensured by Proposition 3.2. This allows us to compute the first TT core $\mathbf{z}^{(p)}$ of the approximate residual $z_p = \operatorname{vec} \tau(\mathbf{z}^{(p)}, \hat{\mathbf{r}}^{(p+1)}, \mathbf{r}^{(p+2)}, \dots, \mathbf{r}^{(d)})$ by applying SVD to just one TT core $\hat{\mathbf{r}}^{(p)}$. The complexity of this step is $\mathcal{O}(nk^6)$ operations, where $k^2 = \max \mathtt{k}(\bar{\mathbf{r}}) \simeq \max \mathtt{k}(\bar{\mathbf{A}}) \max \mathtt{k}(\bar{\mathbf{t}})$. The overall cost of approximation steps is again $\mathcal{O}(dnk^6)$. This version of Algorithm 4 will be referred to as the $\mathrm{AMEn}_{(\mathrm{SVD})}$ algorithm.

Remark 6.1. The approximation z_p in the $\mathrm{AMEn}_{(\mathrm{SVD})}$ algorithm has all the usual properties of the approximation, computed by the SVD algorithm. In particular, it satisfies (4.8), where one can either require an arbitrary accuracy $\varepsilon < 1$ and obtain a TT core $\mathbf{z}^{(p)} \in \mathbb{C}^{\mathtt{k}_{p-1}(\bar{\mathbf{x}}) \times n_p \times k_p}$ with the optimal rank k_p or set up a desired rank bound $k_p^{\max}$ and compute $\mathbf{z}^{(p)}$ with $k_p \leqslant k_p^{\max}$ and the optimal accuracy $\varepsilon < 1$.

To ensure that the upper bounds on the convergence rates in (4.14) and (5.12) do not approach 1, it is sufficient to use $\varepsilon \leqslant \hat{\varepsilon} < 1$ with some fixed $\hat{\varepsilon}$. When the accuracy-based approximation is used, this is straightforward. For the rank-based criteria, the upper bound $\hat{\varepsilon}$ also exists naturally, as explained in the following remark.

Remark 6.2. If the approximation in Algorithm 4, line 3, is computed by the SVD (as explained above) with the rank bounded by $k_p^{\max}$, then the obtained accuracy satisfies $\varepsilon^2 \leqslant \hat{\varepsilon}^2 = 1 - k_p^{\max}/n^{d/2} < 1$, where $n = \max_p n_p$.

Proof. The SVD is effectively applied to the *unfolding* $R_p \in \mathbb{C}^{\mathtt{k}_{p-1}(\bar{\mathbf{x}})\mathbf{n_p}\times(\mathbf{n_{p+1}}\cdots\mathbf{n_d})}$, which consists of the elements of $r_p = \operatorname{vec}\tau(\mathbf{r}^{(p)},\dots,\mathbf{r}^{(d)})$ and has no more than $\mathtt{k}_p(\bar{\mathbf{r}}) \leqslant n^{d/2}$ nonzero singular values. The truncated SVD algorithm chooses the largest $k_p^{\max}$ among them and discards the others as the error. The worst estimate $\hat{\varepsilon}$ corresponds to the case when all singular values are the same. ∎

6.3. Fast heuristic approximation. The complexity of the AMEn$_{(\mathrm{SVD})}$ algorithm is linear in d but scales as $\mathcal{O}(k^6)$ w.r.t. the TT-ranks $k \simeq \mathtt{k}(\bar{\mathbf{A}}) \simeq \mathtt{k}(\bar{\mathbf{t}})$, that may significantly slow down the computation. To reduce the complexity, we can substitute the SVD truncation by the one-site DMRG (ALS) Algorithm 1 optimization, applied to the error $\mathtt{J}_{I,r}(z) = \|r - z\|^2$ between the exact and approximate residuals. It appears in practice that even one *microstep* of ALS (Algorithm 1, lines 7–9) is sufficient to obtain an approximation z_p that is good enough to guide the following optimization steps in the right direction. It allows us to conduct AMEn (Algorithm 4) and the ALS (Algorithm 1) simultaneously, synchronizing the steps in both algorithms. This version will be referred to as the AMEn$_{(\mathrm{ALS})}$ algorithm. If the rank bound $k^{\max}$ is used for the approximation of the residual, AMEn$_{(\mathrm{ALS})}$ has $\mathcal{O}(dk^3k^{\max}\,\mathtt{matvec}(n))$ complexity, where $\mathtt{matvec}(n)$ denotes the cost of matrix-by-vector product of one $\mathbf{A}^{(p)}(i_p, j_p)$ by one $\mathbf{t}^{(p)}(j_p)$. For moderate $k^{\max}$ this is three orders lower than the complexity of the approximation in the AMEn$_{(\mathrm{SVD})}$ algorithm.

Remark 6.3. The approximation strategy in the AMEn$_{(\mathrm{ALS})}$ algorithm provides orthogonality $(z_p - r_p, z_p) = 0$.

Proof. The approximation step (Algorithm 4, line 3) in the AMEn$_{(\mathrm{ALS})}$ algorithm is done by Algorithm 1 with $A = I$. It boils down to the solution of $Z_{\neq 1}z^{(1)} = r_1 = b - Au_1$, where $Z_{\neq 1}$ is the frame matrix of the current approximation $z \approx r$. If the orthogonality is imposed on $Z_{\neq 1}$, the solution $z^{(1)} = Z_{\neq 1}^* r_1$ provides $z_1 = Z_{\neq 1}Z_{\neq 1}^* r_1$, which ensures $(z_1 - r_1, z_1) = 0$. The same argument holds for other z_p when AMEn is applied recursively for $p = 1,\dots,d$. ∎

We are not currently able to quantify the quality of approximation $\|z_p - r_p\|$ computed by this method, i.e., to estimate ε in (4.8). However, if the subspace spanned by columns of the current frame matrix is not orthogonal to the exact residual (i.e., $Z_{\neq 1}^* r_1 \neq 0$), which happens almost surely, the approximation satisfies $\varepsilon < 1$.

6.4. Truncation of the solution. ALS(SD) and AMEn increase the TT ranks in each step. Sometimes, however, it is useful to reduce them. To do this, we apply the TT-SVD recompression algorithm [53, 45] to the solution after each microstep. Since the interface matrices are orthogonal, the SVD step may be applied at low cost: we just add the truncation of $U^{|1\rangle}$ between lines 1 and 2 of Algorithm 4. Our convergence analysis does not account for the error which appears on this step.

7. Numerical experiments. In this section we compare the performance of the ALS(SD) (Algorithm 3), the AMEn (Algorithm 4), and the two-site DMRG (Algorithm 2) algorithms, which are implemented in MATLAB and publicly available in the TT-Toolbox (http://github.com/oseledets/TT-Toolbox) as files `alstpz_solve.m`, `amen_solve2.m`, and `dmrg_solve3.m`, respectively. Currently, the ALS(SD) uses the TT-SVD algorithm to approximate the residual, while AMEn can use either SVD or ALS.

The algorithms are compared also to the GMRES algorithm, in which all vectors are represented in the TT format and relaxed TT-SVD truncations are used; see details in [8] and the implementation in the file `tt_gmres.m`. GMRES is featured as a reference classical method with a proven convergence. The comparison is used

to demonstrate the difficulty arising from large tensor ranks of the Krylov vectors, outlined in the introduction.

In section 7.2 we compare the AMEn techniques with the recent KSL scheme [42], an ODE integrator on the TT manifold. The KSL algorithm is implemented by Sergey Dolgov in communication with Ivan Oseledets in the `tt_ksl_ml.m` file.

The experiments described in sections 7.1 and 7.2 can be reproduced by running the scripts `test_amen_laplace.m` and `test_amen_cme.m`, respectively. Unless stated otherwise, the parameters of the algorithms are the following: the residual in AMEn and ALS(SD) is approximated with the rank bound $k^{\max} = 4$, the solution in GMRES is truncated with the tolerance $\varepsilon = 10^{-3}$ (for the Krylov vectors, the accuracy is relaxed according to [55]), and the GMRES is restarted each 15 iterations. The computations were done on a Linux workstation with 2.6-GHz Intel Xeon CPU and MATLAB R2013b at the MPI MiS, Leipzig, Germany.

7.1. Poisson equation. We consider the high-dimensional Poisson equation

$$-\Delta x(q) = e, \qquad q \in D = [0,1]^d, \qquad x|_{\partial D} = 0, \tag{7.1}$$

discretized with the finite difference scheme on a uniform $n \times n \times \cdots \times n$ grid with $n = 64$, $d = 16$. The total number of unknowns is $N = 64^{16} \approx 8 \cdot 10^{28}$. The right-hand side is $e = (1, \ldots, 1)^\top$.

The results are presented in Figure 7. All optimization-based methods behave similarly in terms of the number of iterations but differently on the time scale, where the AMEn methods outperform the two-site DMRG by a factor of approximately $n = 64$. In this example, all alternating algorithms reach the accuracy ε (i.e., there is no stagnation), and both realizations of the AMEn and the ALS(SD) deliver the solution in the same time. The GMRES converges slowly due to the large $\kappa(\Delta) \sim n^2$.

7.2. Chemical master equation. Consider the discretized *chemical master equation* (CME) [58], which describes a d-dimensional *cascade gene regulatory network* [22, 1]. The nonsymmetric linear system $Ax = b$ arises after the backward Euler discretization $x(t+h) = (I + hM)^{-1}x(t)$. Each *snapshot* $x(t_l)$, $l = 1, \ldots, L$, is a $n \times \cdots \times n$ tensor with $N = n^d$ entries. Following [9], all snapshots are stored *simultaneously* in $n \times \cdots \times n \times L$ tensor $x = [x(hl)]$ of dimension $d+1$. The right-hand-side $b = e \otimes x(0)$, where $e = (1, \ldots, 1)^\top$, $x(0) = (1, 0, \ldots, 0)^\top$. The matrix

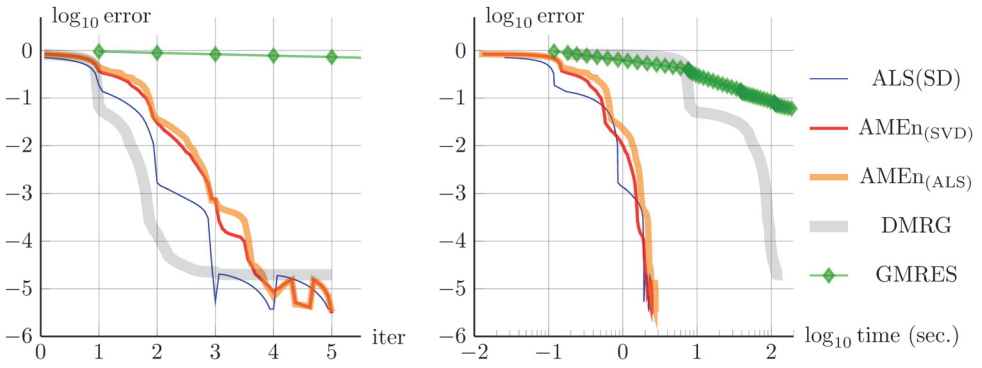

FIG. 7. *Relative A-norm error in different methods versus the number of iterations (left) and CPU time (right) for the Poisson equation* (7.1) *with dimension* $d = 16$ *and mode sizes* $n = 64$. *In AMEn, ALS(SD), and DMRG the solution is truncated using the relative threshold* $\varepsilon = 10^{-5}$; *in GMRES we use* $\varepsilon = 10^{-3}$.

$A = I_L \otimes I_N + hG_L^{-1} \otimes M$, with the discrete gradient $G_L = \mathrm{tridiag}(-1, 1, 0) \in \mathbb{R}^{L\times L}$ and the CME matrix $M \in \mathbb{R}^{N\times N}$, is defined as follows:

$$M = 0.7 \cdot G_n \otimes I_{n^{d-1}} + \sum_{p=1}^{d-1} I_{n^{p-1}} \otimes W_n \otimes G_n \otimes I_{n^{d-p-1}} + \sum_{p=1}^{d} I_{n^{p-1}} \otimes G_n^\top D_n \otimes I_{n^{d-p}}$$

with coefficient matrices $W_n = \mathrm{diag}\{i/i+5\}_{i=0}^{n-1}$ and $D_n = 0.07\,\mathrm{diag}\{i\}_{i=0}^{n-1}$. The size parameters are chosen in accordance with [7]: $d = 20$, $n = 64$, $L = 2^{12}$, $h = 10/L$.

Additional cost reduction is achieved using the *quantized* TT (QTT) format [32], for which we increase the dimensionality of x and b by the reshape

$$\underbrace{n \times \cdots \times n}_{d \text{ dimensions}} \times L \quad \Rightarrow \quad \underbrace{2 \times \cdots \times 2}_{d \log_2 n \text{ dimensions}}, \times \underbrace{2 \times \cdots \times 2}_{\log_2 L \text{ dimensions}}$$

and apply the TT format (2.2) with dimension $D = d \log_2 n + \log_2 L$ and mode sizes 2. Both A and b have moderate QTT ranks [7, 30].

The full problem size $Ln^d \simeq 10^{40}$ makes the straightforward solution impossible. Existing techniques either leave the CME aside (the SSA method [16] and its descendants) or employ high-dimensional techniques like Smolyak's *sparse grids* [22], *greedy approximations* [1], or dynamics on tensor manifolds [26]. In the current work we apply the AMEn algorithm to the CME problem and compare it with the two-site DMRG Algorithm 2. For some systems with moderate dimensions and small time steps, the DMRG method can be of good use, as was demonstrated in [29]. However, we show that the AMEn algorithm performs better for this more difficult problem.

We set the relative tensor truncation threshold for the solution $\varepsilon = 10^{-6}$ and track the convergence of different methods toward the reference solution, which is computed via $\mathrm{AMEn}_{(\mathrm{SVD})}$ with high accuracy $\varepsilon = 10^{-9}$. The results are given in Figure 8. Since $Ax = b$ is not an SPD system, we can apply the traditional symmetrization $A^*Ax = A^*b$, which squares both the condition number and the TT ranks of the matrix, and usually slows the computation. We can also apply all algorithms directly to $Ax = b$, sacrificing the proven convergence estimates.

We observe that the DMRG fails to solve both the initial system and the symmetrization—it returns the approximation with significantly underestimated TT

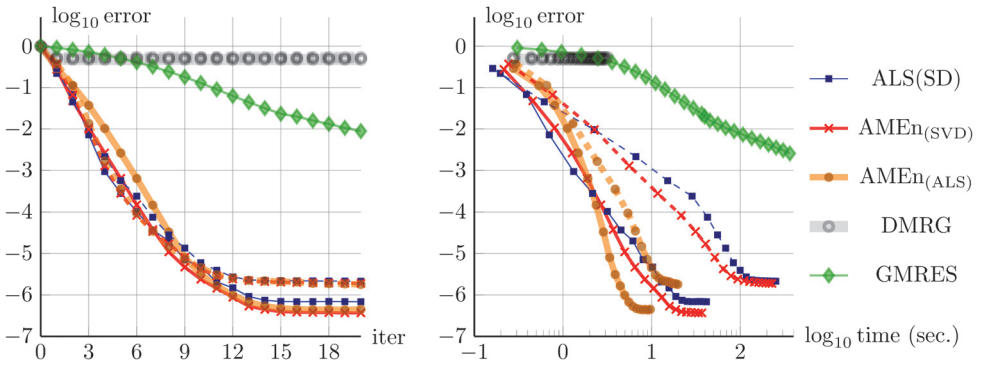

FIG. 8. *Relative Frobenius-norm error in different methods versus the number of iterations (left) and CPU time (right) for the CME equation described in section* 7.2. Solid lines*: nonsymmetric system $Ax = b$;* dashed lines*: symmetrized system $A^*Ax = A^*b$. In AMEn, ALS(SD), and DMRG the solution is truncated using the relative threshold $\varepsilon = 10^{-6}$; in GMRES we use $\varepsilon = 10^{-3}$.*

TABLE 1
Errors in final time snapshots (err) and the CPU times in seconds (time) of different methods.

	ALS(SD)		$\text{AMEn}_{(\text{SVD})}$		$\text{AMEn}_{(\text{ALS})}$		DMRG		KSL
	A	A^*A	A	A^*A	A	A^*A	A	A^*A	
Err., $\cdot 10^{-5}$	1.3	2.6	0.77	2.6	0.88	2.4	98,000	180,000	84
Time, sec	15.7	133	13.1	93.7	4.97	12.1	1.60	1.19	90.4

TABLE 2
CPU time in seconds, number of iterations, maximal TT rank hit during iterations, and the final error of the solution in the GMRES and the AMEn algorithms.

	TT-GMRES				$\text{AMEn}_{(\text{ALS})}$			
ε	Time	Iter	Rank	Error	Time	Iter	Rank	Error
10^{-2}	$3.1 \cdot 10^1$	17	42	$2.5 \cdot 10^{-2}$	1.3	6	14	$3.1 \cdot 10^{-3}$
10^{-3}	$3.6 \cdot 10^2$	27	156	$2.6 \cdot 10^{-3}$	2.0	8	22	$3.1 \cdot 10^{-4}$
10^{-4}	$4.0 \cdot 10^3$	37	264	$2.4 \cdot 10^{-4}$	2.4	9	31	$4.3 \cdot 10^{-5}$
10^{-5}	$6.1 \cdot 10^4$	57	520	$2.6 \cdot 10^{-5}$	3.5	11	41	$4.9 \cdot 10^{-6}$
10^{-6}	*out of our patience*				5.3	13	49	$9.1 \cdot 10^{-7}$

ranks, which is also reflected by small CPU times; see Table 1. All proposed algorithms deliver a satisfactory solution. Interestingly, the nonsymmetric versions appear to be even faster and more accurate than their symmetrized counterparts. Focusing on two AMEn realizations, we note that $\text{AMEn}_{(\text{ALS})}$ speeds up the computations essentially compared to the $\text{AMEn}_{(\text{SVD})}$ algorithm, while the convergence does not deteriorate. This is especially useful when TT ranks of the matrix are large, e.g., for the symmetrized problem.

The GMRES manifests a reasonable linear convergence for this well-conditioned problem but is clearly uncompetitive with AMEn on the time scale. This is a natural consequence of the large TT ranks of the GMRES vectors, while AMEn does not overshoot the quasi-optimal TT ranks of the solution, as shown in Table 2.

7.3. AMEn as an ODE solver. Since the CME problem is an ODE, we may be interested in the final time snapshot $x(hL)$. In Table 1 we show the relative Frobenius norm errors for $x(hL)$ verified w.r.t. the reference solution. The observed errors are at the same level as the accuracies of the total solution x in Figure 8.

Now we can compare the last snapshot computed by the AMEn with the result obtained by another ODE integrator. A dynamical problem may be put onto the tensor manifold via the so-called Dirac–Frenkel principle (see, e.g., [35]): we solve the problem $\min_{\bar{\mathbf{x}}:\mathbf{k}(\bar{\mathbf{x}})=k} \|d\tau(\bar{\mathbf{x}})/dt - dy/dt\|$, projecting the exact velocity dy/dt onto the tangent space of the TT manifold. Several theoretical issues were addressed in [43], and the numerical splitting w.r.t. the TT cores $\mathbf{x}^{(p)}$ was proposed in [42] as the KSL scheme. The latter may be applied to our ODE if dy/dt is substituted by $M\tau(\bar{\mathbf{x}})$.

The KSL scheme is efficient in simulation of molecular vibrations [46]. However, it possesses the same drawback as the ALS Algorithm 1: TT ranks of the solution are predefined and fixed. Moreover, we observe that the KSL delivers a much larger error than the other methods, even if the TT ranks are set to the proper values. When we decrease the time step, the computational cost of the KSL grows (90 seconds for $L = 50$ time steps) and becomes larger than that of AMEn. However, the error of the KSL stagnates at the level of $8 \cdot 10^{-4}$ and does not improve.

8. Conclusion and future work. We propose new methods for the solution of high-dimensional linear systems, bridging optimization in tensor product formats (DMRG/MPS schemes) and classical iterative methods (SD) together. Each iteration of the developed algorithms has asymptotically the same complexity as the one-site DMRG (or ALS), and the convergence demonstrated in the numerical experiments is competitive with that of the two-site DMRG. The SD step, which is used to expand the search space, secures the geometrical convergence of the proposed methods with the global upper bound for the convergence rate.

The developed ALS(SD) and AMEn algorithms are tested for the model Poisson problem in high dimension and demonstrate fast convergence and accurate results. We also applied them to a non-SPD problem, arising from the high-dimensional chemical master equation for the mesoscopic scale gene regulatory model. The first experiment with the symmetrized system shows that the DMRG stagnates, while the ALS(SD) and the AMEn are efficient in terms of both iterations and the computational time. The new algorithms converge even faster and smoother when they are applied to the non-SPD system directly, although such behavior is not yet theoretically explained. This demonstrates a significant advantage of ALS(SD) and AMEn over the DMRG approach.

We also compared the proposed methods with the GMRES algorithm, where all vectors are approximated in the TT format. In numerical experiments considered in this paper the TT ranks of GMRES vectors grow rapidly with iterations, and the algorithm struggles to find a sufficiently accurate solution, despite the relaxation of approximation accuracy on the latter iterations. In contrast, the AMEn does not need to approximate the basis vectors of the Krylov subspace and only stores the current iterate and an approximation to the residual, which may be computed with a (very) rough accuracy and (very) small TT ranks. This makes the AMEn algorithm more efficient in practical computations and at the same time fully supported by the theory.

Several directions of future work are seen on this stage. The combination of the one-site update and basis enrichment steps (the AMEn scheme) can be generalized to other problems—first, naturally, to the high-dimensional ground state problem. The DMRG/MPS schemes were originally developed to solve it for a quantum many-body system, and these algorithms are implemented in several well-established numerical packages for quantum physics computations. To adjust the AMEn method to this problem, it is sufficient to replace the energy function $\mathtt{J}_{A,b}(x)$ with the Rayleigh quotient $\mathtt{Q}_A(x)$. It is natural to compare AMEn with the optimization methods developed in the DMRG/MPS community. This work is started in [10, 13, 37].

There is a certain mismatch between the theoretical convergence estimates, which are on the level of the SD algorithm, and the practical convergence pattern, which is superlinear. This indicates that our understanding of the convergence of the AMEn and similar methods can be improved, and it inspires us to look for possible connections with the theory of iterative methods of Krylov and Newton type. The convergence rate bounds can benefit from sharp estimates of the progress of the one-dimensional update steps μ_p, which are available at the moment only in a small vicinity of a true solution, which is hard to satisfy in practice. The use of preconditioners should also be discussed, and this work is started in [40, 37].

The AMEn algorithm already has been applied to challenging high-dimensional problems, such as the chemical master equation for mesoscopic scale gene regulatory networks [7], the ground state problem for a periodic Heisenberg spin chain [13], and the quantum spin dynamics simulation for nuclear magnetic resonance spectroscopy

experiments [52]. We look forward to solving more high-dimensional problems and are sure that they will bring new understanding of the advantages and drawbacks of the proposed methods and open new directions for a future research.

REFERENCES

[1] A. Ammar, E. Cueto, and F. Chinesta, *Reduction of the chemical master equation for gene regulatory networks using proper generalized decompositions*, Int. J. Numer. Methods Biomed. Eng., 28 (2012), pp. 960–973.

[2] R. Andreev and C. Tobler, *Multilevel Preconditioning and Low Rank Tensor Iteration for Space-Time Simultaneous Discretizations of Parabolic PDEs*, Numer. Linear. Alg. Appl., 2014, DOI: 10.1002/nla.1951.

[3] M. Bachmayr and W. Dahmen, *Adaptive near-optimal rank tensor approximation for high-dimensional operator equations*, Found. Comput. Math., (2014), pp. 1–60.

[4] J. Ballani and L. Grasedyck, *A projection method to solve linear systems in tensor format*, Numer. Linear Algebra Appl., 20 (2013), pp. 27–43.

[5] F. L. Bauer and A. S. Householder, *Some inequalities involving the Euclidian condition of a matrix*, Numer. Math., 2 (1960), pp. 308–311.

[6] P. A. M. Dirac, *A new notation for quantum mechanics*, Math. Proc. Cambridge Philos. Soc., 35 (1939), pp. 416–418.

[7] S. Dolgov and B. Khoromskij, *Simultaneous state-time approximation of the chemical master equation using tensor product formats*, Numer. Linear Algebra Appl., 2014.

[8] S. V. Dolgov, *TT-GMRES: Solution to a linear system in the structured tensor format*, Russian J. Numer. Anal. Math. Modelling, 28 (2013), pp. 149–172.

[9] S. V. Dolgov, B. N. Khoromskij, and I. V. Oseledets, *Fast solution of multi-dimensional parabolic problems in the tensor train/quantized tensor train–format with initial application to the Fokker-Planck equation*, SIAM J. Sci. Comput., 34 (2012), pp. A3016–A3038.

[10] S. V. Dolgov, B. N. Khoromskij, I. V. Oseledets, and D. V. Savostyanov, *Computation of extreme eigenvalues in higher dimensions using block tensor train format*, Comput. Phys. Comm., 185 (2014), pp. 1207–1216.

[11] S. V. Dolgov, B. N. Khoromskij, and D. V. Savostyanov, *Superfast Fourier transform using QTT approximation*, J. Fourier Anal. Appl., 18 (2012), pp. 915–953.

[12] S. V. Dolgov and I. V. Oseledets, *Solution of linear systems and matrix inversion in the TT-format*, SIAM J. Sci. Comput., 34 (2012), pp. A2718–A2739.

[13] S. V. Dolgov and D. V. Savostyanov, *Corrected One-Site Density Matrix Renormalization Group and Alternating Minimal Energy Algorithm*, arXiv:1312.6542, 2013.

[14] A. Einstein, *Die Grundlage der allgemeinen Relativitätstheorie*, Ann. Phys., 354 (1916), pp. 769–822.

[15] M. Fannes, B. Nachtergaele, and R. Werner, *Finitely correlated states on quantum spin chains*, Comm. Math. Phys., 144 (1992), pp. 443–490.

[16] D. Gillespie, *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*, J. Comput. Phys., 22 (1976), pp. 403–434.

[17] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1996.

[18] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78.

[19] W. Hackbusch, *Tensor Spaces and Numerical Tensor Calculus*, Springer-Verlag, Berlin, 2012.

[20] W. Hackbusch, B. N. Khoromskij, S. A. Sauter, and E. E. Tyrtyshnikov, *Use of tensor formats in elliptic eigenvalue problems*, Numer. Linear Algebra Appl., 19 (2012), pp. 133–151.

[21] W. Hackbusch, B. N. Khoromskij, and E. E. Tyrtyshnikov, *Approximate iterations for structured matrices*, Numer. Math., 109 (2008), pp. 365–383.

[22] M. Hegland, C. Burden, L. Santoso, S. MacNamara, and H. Booth, *A solver for the stochastic master equation applied to gene regulatory networks*, J. Comput. Appl. Math., 205 (2007), pp. 708–724.

[23] S. Holtz, T. Rohwedder, and R. Schneider, *The alternating linear scheme for tensor optimization in the tensor train format*, SIAM J. Sci. Comput., 34 (2012), pp. A683–A713.

[24] T. Huckle and K. Waldherr, *Subspace iteration method in terms of matrix product states*, Proc. Appl. Math. Mech., 12 (2012), pp. 641–642.

[25] T. Huckle, K. Waldherr, and T. Schulte-Herbrüggen, *Computations in quantum tensor*

networks, Linear Algebra Appl., 438 (2013), pp. 750–781.
[26] T. Jahnke and W. Huisinga, *A dynamical low-rank approach to the chemical master equation*, Bull. Math. Biol., 70 (2008), pp. 2283–2302.
[27] E. Jeckelmann, *Dynamical density-matrix renormalization–group method*, Phys. Rev. B, 66 (2002), 045114.
[28] L. V. Kantorovich, *Funktsionallniy analyz i prikladnaya matematika*, Uspekhi Mat. Nauk, 3 (1945), pp. 89–185.
[29] V. Kazeev, M. Khammash, M. Nip, and C. Schwab, *Direct solution of the chemical master equation using quantized tensor trains*, PLOS Comput. Biol., 10 (2014), p. e100359.
[30] V. A. Kazeev and B. N. Khoromskij, *Low-rank explicit QTT representation of the Laplace operator and its inverse*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 742–758.
[31] B. Khoromskij and C. Schwab, *Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs*, SIAM J. Sci. Comput., 33 (2011), pp. 1–25.
[32] B. N. Khoromskij, *$\mathcal{O}(d \log n)$-Quantics approximation of N-d tensors in high-dimensional numerical modeling*, Constr. Approx., 34 (2011), pp. 257–280.
[33] B. N. Khoromskij, *Tensor-structured numerical methods in scientific computing: Survey on recent advances*, Chemometr. Intell. Lab. Syst., 110 (2012), pp. 1–19.
[34] B. N. Khoromskij and I. V. Oseledets, *Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs*, Comput. Methods Appl. Math., 10 (2010), pp. 376–394.
[35] O. Koch and C. Lubich, *Dynamical tensor approximation*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2360–2375.
[36] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500.
[37] D. Kressner, M. Steinlechner, and A. Uschmajew, *Low-Rank Tensor Methods with Subspace Correction for Symmetric Eigenvalue Problems*, MATHICSE preprint 40.2013, EPFL, Lausanne, 2013.
[38] D. Kressner and C. Tobler, *Krylov subspace methods for linear systems with tensor product structure*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 1688–1714.
[39] D. Kressner and C. Tobler, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 273–290.
[40] D. Kressner and C. Tobler, *Preconditioned low-rank methods for high-dimensional elliptic PDE eigenvalue problems*, Comput. Methods Appl. Math., 11 (2011), pp. 363–381.
[41] O. S. Lebedeva, *Tensor conjugate-gradient-type method for Rayleigh quotient minimization in block QTT-format*, Russian J. Numer. Anal. Math. Modelling, 26 (2011), pp. 465–489.
[42] C. Lubich and I. V. Oseledets, *A projector-splitting integrator for dynamical low-rank approximation*, BIT, 54 (2014), pp. 171–188.
[43] C. Lubich, T. Rohwedder, R. Schneider, and B. Vandereycken, *Dynamical approximation by hierarchical Tucker and tensor-train tensors*, SIAM J. Matrix. Anal. Appl., 34 (2013), pp. 470–494.
[44] H. Munthe-Kaas, *The Convergence Rate of Inexact Preconditioned Steepest Descent Algorithm for Solving Linear Systems*, Numerical Analysis Report NA-87-04, Stanford University, Standford, CA, 1987; also available online from http://i.stanford.edu/pub/cstr/reports/na/m/87/04/NA-M-87-04.pdf.
[45] I. V. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.
[46] I. V. Oseledets, B. N. Khoromskij, and R. Schneider, *Efficient Time-Stepping Scheme for Dynamics on TT-manifolds*, Preprint 24, MPI MIS, 2012.
[47] I. V. Oseledets, D. V. Savostyanov, and E. E. Tyrtyshnikov, *Linear algebra for tensor problems*, Computing, 85 (2009), pp. 169–188.
[48] I. V. Oseledets and E. E. Tyrtyshnikov, *Breaking the curse of dimensionality, or how to use SVD in many dimensions*, SIAM J. Sci. Comput., 31 (2009), pp. 3744–3759.
[49] T. Rohwedder and A. Uschmajew, *On local convergence of alternating schemes for optimization of convex problems in the tensor train format*, SIAM J. Numer. Anal., 51 (2013), pp. 1134–1162.
[50] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, 2003.
[51] D. V. Savostyanov, *Polilinear Approximation of Matrices and Integral Equations* (in Russian), Ph.D. thesis, INM RAS, Moscow, 2006; also available online from http://www.inm.ras.ru/library/Tyrtyshnikov/savostyanov_disser.pdf.
[52] D. V. Savostyanov, S. V. Dolgov, J. M. Werner, and I. Kuprov, *Exact NMR simulation of protein-size spin systems using tensor train formalism*, Phys. Rev. B, 90 (2014), p. 085139.

[53] U. Schollwöck, *The density–matrix renormalization group*, Rev. Modern Phys., 77 (2005), pp. 259–315.
[54] U. Schollwöck, *The density-matrix renormalization group in the age of matrix product states*, Ann. Phys., 326 (2011), pp. 96–192.
[55] V. Simoncini and D. B. Szyld, *Theory of inexact Krylov subspace methods and applications to scientific computing*, SIAM J. Sci. Comput., 25 (2003), pp. 454–477.
[56] G. W. Stewart, *On the perturbation of pseudo–inverses, projections and linear least squares problems*, SIAM Rev., 19 (1977), pp. 634–662.
[57] J. van den Eshof and G. L. G. Sleijpen, *Inexact Krylov subspace methods for linear systems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 125–153.
[58] N. G. van Kampen, *Stochastic Processes in Physics and Chemistry*, North-Holland, Amsterdam, 1981.
[59] S. R. White, *Density matrix formulation for quantum renormalization groups*, Phys. Rev. Lett., 69 (1992), pp. 2863–2866.
[60] S. R. White, *Density-matrix algorithms for quantum renormalization groups*, Phys. Rev. B, 48 (1993), pp. 10345–10356.
[61] S. R. White, *Density matrix renormalization group algorithms with a single center site*, Phys. Rev. B, 72 (2005), 180403.