Quasioptimality of maximum–volume cross interpolation of tensors

D. V. Savostyanov[a,b,1]

[a]*Institute of Numerical Mathematics of Russian Academy of Sciences, Gubkina 8, Moscow 119333, Russia*
[b]*University of Southampton, School of Chemistry, Highfield Campus, Southampton SO17 1BJ, UK*

Abstract

We consider a cross interpolation of high–dimensional arrays in the tensor train format. We prove that the maximum–volume choice of the interpolation sets provides the quasioptimal interpolation accuracy, that differs from the best possible accuracy by the factor which does not grow exponentially with dimension. For nested interpolation sets we prove the interpolation property and propose greedy cross interpolation algorithms. We justify the theoretical results and measure speed and accuracy of the proposed algorithm with numerical experiments.

Keywords: high–dimensional problems, tensor train format, maximum–volume principle, cross interpolation
2010 MSC: 15A69, 15A23, 65D05, 65F99

1. Introduction

As demand for big data analysis grows, algorithms for high–dimensional data have become increasingly important in scientific computing. The total number of entries in a *tensor* (an array with d indices) grows exponentially with dimension d. Even for a moderate d, it is impossible to process, store, and even compute all elements of a tensor by standard methods. This issue is known in numerical analysis and related areas as the *curse of dimensionality*. Different techniques are used to relax or to overcome this problem, e.g. low parametrical representation on Smolyak's *sparse grids* [62, 9], (Markov chain) *Monte Carlo* sampling [30, 5], *reduced basis* methods [56], in particular those based on the *greedy approximation* [2, 45, 1, 8, 10, 63]. Significant progress was recently made in the development and understanding of the *tensor product* methods [40, 37, 28, 27].

The tensor product methods implement the *separation of variables* at the discrete level, known in the two–dimensional case as the *low rank decomposition* of a matrix. Several approaches have been developed to generalise rank–structured low parametrical models to tensors (see e.g. [40]), and a simple and efficient *tensor train* (TT) format has been proposed [49]. It is equivalent to the *matrix product states* (MPS) [19, 39] and the *density matrix renormalization group* (DMRG) [67, 68, 55] representations, introduced in the quantum physics community for quantum states of many–body systems. When the TT format was re-discovered in the numerical linear algebra community, the optimisation techniques developed in the MPS/DMRG framework were adapted for other high–dimensional problems, and novel algorithms were proposed. Now we can use the TT/MPS and the more general HT format [29, 26, 44] to approximate high–dimensional data and perform algebraic operations [61, 49, 48]), solve linear systems [33, 41, 38, 35, 42, 31, 14, 11, 3, 15, 16] and partial eigenproblems [46, 43, 32, 17, 12],

---

Email address: dmitry.savostyanov@gmail.com (D. V. Savostyanov)

compute the multidimensional Fourier transform [13] and discrete convolution [34], etc. With these algorithms in hand, high–dimensional scientific computations become possible as soon as all data are translated into the TT format.

It is important, therefore, to develop algorithms which construct the approximation of a given high–dimensional array in the tensor product format. For some function–related tensors, the TT representation can be written explicitly (see e.g. [36, 50]). In general, although every entry of a tensor can be computed *on demand* (by a formula or as a solution of a feasible problem, e.g. PDE in three dimensions), all elements are inaccessible in a reasonable time. The question arises naturally whether it is possible to *reconstruct* or *approximate* a given tensor in the TT format from a few elements, also known as *samples*.

For matrices, i.e. 2–tensors, this question is well studied. We know that a rank–r matrix is recovered from a *cross* of r rows and columns if the submatrix on their intersection is nonsingular. When data do not fit the low rank model exactly, the accuracy of the *cross interpolation* depends crucially on the chosen cross. A good choice is the *maximum volume* cross, which has the $r \times r$ submatrix with the maximum modulus of determinant on the intersection. For this cross, the interpolation accuracy differs from the accuracy of the best possible approximation by the factor $\mathcal{O}(r^2)$, i.e. is *quasioptimal* [60, 22].

For the tensor train format an analogue of the cross interpolation formula is given in [54]. It reconstructs a tensor from a few samples under mild non-singularity conditions, if the TT representation is exact. The structure of the algorithm suggested in [54] resembles the *one-site* DMRG algorithm, that optimises the target function (e.g. *Rayleigh quotient*) for the factors of the MPS/TT format (TT cores or *sites*). Similarly, the algorithm from [54] searches for better crosses in order to improve the approximation accuracy. This one-site scheme is also known as the *alternating least squares* (ALS) algorithm, and has been used for various tensor formats and applications, see [40] for a review.

The drawback of the ALS scheme is that the ranks of the underlying tensor format should be defined *a priori* and can not be changed during the computations. To update the ranks *adaptively*, the *two-site* DMRG is used [68], which optimises the target function over the *pairs* of neighbouring sites. The corresponding rank–adaptive version of the TT interpolation algorithm is proposed in [59]. The algorithms from [54, 59] are *heuristic*, as well as interpolation algorithms for other tensor formats, e.g. the Tucker [51, 52] and the HT format [4].

The accuracy of the cross interpolation of tensors has not been well studied yet. For the 3–dimensional Tucker model the quasioptimality with the factor $\mathcal{O}(r^3)$ is shown in [51]. In d dimensions we can expect an excessively large coefficient $\mathcal{O}(r^d)$, cf. $\mathcal{O}(r^{2d})$ for the HT format [4]. The main result of this paper is more optimistic. The quasioptimality of the maximum volume cross interpolation is generalised to the TT format with the coefficient $(2r + \kappa r + 1)^{\lceil \log_2 d \rceil + 2}$ that *does not necessarily grow exponentially* with d. This important *existence result* reassures us that for high–dimensional data that allow TT approximation, accurate cross interpolation is also possible. Since the search for maximum volume submatrix is NP–hard problem [6], we can not use exactly this interpolation in practical computations. However, the concept of maximum volumes shows us the way to new powerful (yet heuristic) algorithms for high–dimensional problems, which we propose at the end of the paper.

The rest of the paper is organised as follows. Sec. 2 presents notation and definitions. In Sec. 3 the quasioptimality of the maximum–volume cross interpolation is proven. In Sec. 4 the interpolation on *nested* sets is considered, which reduces the search space, but results in larger quasioptimality constant. In Sec. 5 the interpolation property for the nested sets is shown. In Sec. 6 practical cross interpolation algorithms for matrices are recalled and similar algorithms for tensor trains are proposed. In Sec. 7 the coefficient of the quasioptimality is measured for

randomly generated tensors, and speed and accuracy of the proposed algorithm is demonstrated with numerical experiments.

2. Notation, definitions and preliminaries

2.1. Low rank matrix approximation

For a $n_1 \times n_2$ matrix $A = [A(i_1, i_2)]$ its *rank-r* approximation is written as follows

$$A(i_1, i_2) \approx \sum_{s=1}^{r} X^{(1)}(i_1, s) X^{(2)}(s, i_2). \tag{1}$$

Here and later we use *elementwise* notation, i.e. assume that all equations hold for all possible values of *free* indices. For a given matrix A, the best approximation in the Frobenius norm is given by the *singular value decomposition* (SVD) $A = USV^*$ after the truncation of small singular values in S.

2.2. Cross interpolation of a matrix

Different low rank approximations, that can be computed faster than the SVD, are useful for large–scale problems. A notable example of cheaper low rank approximation is the *cross* (or *skeleton*) interpolation, defined as follows

$$A(i_1, i_2) \approx \tilde{A}(i_1, i_2) = \sum_{s,t=1}^{r} A(i_1, \mathcal{I}_t^2) \left[A(\mathcal{I}_s^1, \mathcal{I}_t^2) \right]^{-1} A(\mathcal{I}_s^1, i_2). \tag{2}$$

The *interpolation sets* $\mathcal{I}^1 = \{\mathcal{I}_s^1\}_{s=1}^r$ and $\mathcal{I}^2 = \{\mathcal{I}_t^2\}_{t=1}^r$ contain the positions of the interpolation rows and columns, respectively. The summation over $s, t = 1, \ldots, r$ 'ties' the pairs of subsets together, similarly to the summation in (1). In the matrix form the right hand side of (2) is the product of $n_1 \times r$ matrix of columns, the inverse of $r \times r$ submatrix at the intersection, and $r \times n_2$ matrix of rows of A. Since the *interpolant* $\tilde{A}$ is computed from only $(n_1 + n_2)r - r^2$ entries, for large matrices the cross interpolation is preferable to the SVD, which needs all the elements. The interpolation formula (2) bears its name because it is *exact* on the cross where the elements are computed

$$A(i_1, i_2) = \tilde{A}(i_1, i_2) = \sum_{s,t} A(i_1, \mathcal{I}_t^2) \left[A(\mathcal{I}_s^1, \mathcal{I}_t^2) \right]^{-1} A(\mathcal{I}_s^1, i_2), \quad \text{if} \quad i_1 \in \mathcal{I}^1 \text{ or } i_2 \in \mathcal{I}^2. \tag{3}$$

2.3. Maximum volume principle

When A is not exactly a rank-r matrix, the choice of interpolation sets $\mathcal{I}^1, \mathcal{I}^2$ affects the interpolation accuracy significantly. A good choice of $A_\square = [A(\mathcal{I}^1, \mathcal{I}^2)]$ is the *maximum–volume* $r \times r$ submatrix [25], such that the *volume* $\mathrm{vol}\, A_\square = |\det A_\square|$ is maximal over all possible choices of $\mathcal{I}^1$ and $\mathcal{I}^2$. Assuming that the rank r (i.e. size of $A_\square$) is defined *a priori*, we denote this choice by

$$\left[\mathcal{I}^1, \mathcal{I}^2 \right] = \arg \max_{\mathcal{J}^1, \mathcal{J}^2} \mathrm{vol}[A(\mathcal{J}^1, \mathcal{J}^2)], \quad \text{or} \quad \left[\mathcal{I}^1, \mathcal{I}^2 \right] = \mathrm{maxvol}\, A.$$

For $\mathcal{I}^1, \mathcal{I}^2$ chosen by the maximum–volume principle, the following *quasioptimality* statements are proven in [21] and [60, 22], respectively.

$$\begin{aligned} |A - \tilde{A}| &\leqslant (r+1) \ \min_X \|A - X\|_2, \\ |A - \tilde{A}| &\leqslant (r+1)^2 \min_X |A - X|, \end{aligned} \quad \text{over } X \text{ s.t. } \mathrm{rank}\, X = r. \tag{4}$$

3

Here $\| \cdot \|_2$ denotes the *spectral* norm of a matrix, and $| \cdot |$ denotes the *Chebyshev* norm, also known as the *uniform*, the *supremum* norm, or the maximum entry in modulus. We will also use the *Frobenius* norm, denoted by $\| \cdot \|$.

An important property of the maximum–volume submatrix is that it is *dominant* [20] in the rows and columns which it occupies, i.e.

$$\left| \sum_t \left[A(\mathcal{I}_t^1, \mathcal{I}_s^2) \right]^{-1} A(\mathcal{I}_t^1, i_2) \right| \leqslant 1, \qquad \left| \sum_s A(i_1, \mathcal{I}_s^2) \left[A(\mathcal{I}_t^1, \mathcal{I}_s^2) \right]^{-1} \right| \leqslant 1. \tag{5}$$

2.4. Low rank tensor train format

Several attempts have been made (see reviews [40, 37, 28, 27]) to generalise the low rank decomposition (1) from matrices to *tensors*, i.e. arrays with many indices $A = [A(i_1, \ldots, i_d)]$. In this paper we consider the *tensor train* (TT) decomposition [49] defined as follows

$$\begin{aligned} A(i_1, \ldots, i_d) &= \sum_{\mathbf{s}} X^{(1)}(i_1, s_1) X^{(2)}(s_1, i_2, s_2) \cdots X^{(d-1)}(s_{d-2}, i_{d-1}, s_{d-1}) X^{(d)}(s_{d-1}, i_d) \\ &= \sum_{\mathbf{s}} \prod_{k=1}^d X^{(k)}(s_{k-1}, i_k, s_k). \end{aligned} \tag{6}$$

Here $i_k = 1, \ldots, n_k$, are *mode* or physical indices for *modes* $k = 1, \ldots, d$, and $s_k = 1, \ldots, r_k$ are auxiliary *rank* indices. Values n_k are referred to as *mode sizes* of a tensor, and r_k are *tensor train ranks* or TT–ranks. Summation over $\mathbf{s} = (s_1, \ldots, s_{d-1})$ means summation over all pairs of auxiliary indices $s_1, \ldots, s_{d-1}$, where each index runs through all its possible values. Thanks to the elementwise notation, Eq. (6) represents every entry of a tensor by the product of matrices, where each $r_{k-1} \times r_k$ matrix $X^{(k)}(i_k) = [X^{(k)}_{s_{k-1}, s_k}(i_k)]$ depends on the *parameter* i_k. The three–dimensional array $X^{(k)} = \left[X^{(k)}(s_{k-1}, i_k, s_k) \right]$ is referred to as *TT–core*. To unify the notation, we introduce the virtual *border ranks* $r_0 = r_d = 1$ and consider $[X^{(1)}(i_1, s_1)] = [X^{(1)}(s_0, i_1, s_1)]$ and $[X^{(d)}(s_{d-1}, i_d)] = [X^{(d)}(s_{d-1}, i_d, s_d)]$ as 3–tensors.

2.5. Tensor notation: reshapes, unfoldings, multi-indices

The elementwise notation allows us to *reshape* tensors into vectors or matrices simply by moving indices. We have done this to present a TT–core $\left[X^{(k)}(s_{k-1}, i_k, s_k) \right]$ as a parameter–dependent matrix $[X^{(k)}_{s_{k-1}, s_k}(i_k)]$. More complicated transformations can be expressed by *index grouping*, which combines indices $i_1, \ldots, i_d$ in a single multi-index[2] $\overline{i_1 \ldots i_d}$. For example, the k–th *unfolding* of a tensor is the $(n_1 \cdots n_k) \times (n_{k+1} \cdots n_d)$ matrix with elements

$$A^{\{k\}}(i_{\leqslant k}, i_{>k}) = A^{\{k\}}(\overline{i_1 \ldots i_k}, \overline{i_{k+1} \ldots i_d}) = A(i_1, \ldots, i_d).$$

Here and further we use the following shortcuts to simplify the notation

$$i_{\leqslant k} = \overline{i_1 \ldots i_k}, \quad i_{>k} = \overline{i_{k+1} \ldots i_d}, \qquad \text{and} \qquad i_{b:c} = \overline{i_b \ldots i_c}.$$

For A in the TT–format (6) it holds $\operatorname{rank} A^{\{k\}} = r_k$. In [49] the reverse is proven: for any tensor A there is a representation (6) with TT–ranks $r_k = \operatorname{rank} A^{\{k\}}$. This gives the term *TT–rank* the definite algebraic meaning.

---

[2] The multi-index is usually defined by either the *big–endian* convention $\overline{i_1 \ldots i_d} = i_d + (i_{d-1} - 1)n_d + \ldots + (i_1 - 1)n_2 \cdots n_d$ or the *little–endian* convention $\overline{i_1 \ldots i_d} = i_1 + (i_2 - 1)n_1 + \ldots + (i_d - 1)n_1 \cdots n_{d-1}$. The big–endian notation is similar to numbers written in the positional system, while the little–endian notation is used in numerals in the Arabic scripts and is consistent with the FORTRAN style of indexing. The equations and algorithms in this paper are valid for any endianness.
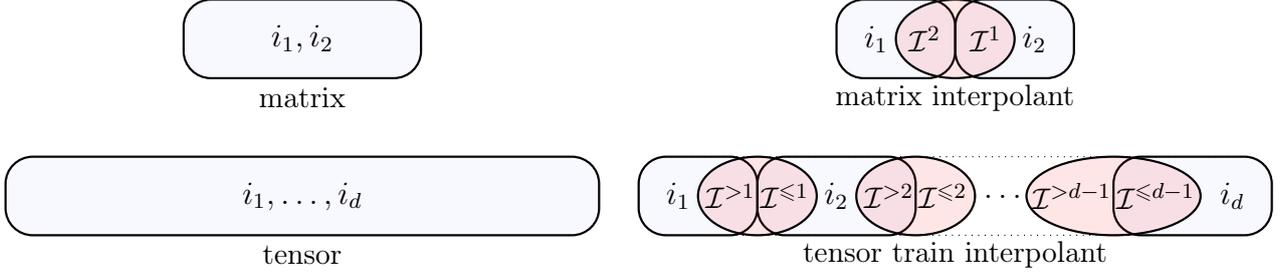
Figure 1: Illustration of matrix and tensor interpolation formulae (2) and (7). Rectangle boxes show the subtensors $\left[A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k})\right]$ which are the building blocks of the decomposition. Ellipses show the inverse matrices $\left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})\right]^{-1}$, which do not carry free indices, but glue pairs of blocks together.

2.6. Cross interpolation of a tensor

The generalisation of cross interpolation formula (2) to the tensor case is suggested in [54] as follows

$$
\begin{aligned}
A(i_1, \ldots, i_d) &= \sum_{\mathbf{s}, \mathbf{t}} A(i_1, \mathcal{I}^{>1}_{t_1}) \left[A(\mathcal{I}^{\leqslant 1}_{s_1}, \mathcal{I}^{>1}_{t_1})\right]^{-1} A(\mathcal{I}^{\leqslant 1}_{s_1}, i_2, \mathcal{I}^{>2}_{t_2}) \cdots A(\mathcal{I}^{\leqslant d-1}_{s_{d-1}}, i_d) \\
&= \sum_{\mathbf{s}, \mathbf{t}} \prod_{k=1}^{d} A(\mathcal{I}^{\leqslant k-1}_{s_{k-1}}, i_k, \mathcal{I}^{>k}_{t_k}) \left[A(\mathcal{I}^{\leqslant k}_{s_k}, \mathcal{I}^{>k}_{t_k})\right]^{-1}.
\end{aligned}
\tag{7}
$$

Here $\mathcal{I}^{\leqslant k}_{s_k}$ and $\mathcal{I}^{>k}_{t_k}$ denote the positions of r_k rows and columns in the k–th unfolding $A^{\{k\}}$. To unify the notation, we introduce the empty border sets $\mathcal{I}^{\leqslant 0} = \emptyset$ and $\mathcal{I}^{>d} = \emptyset$. We denote submatrices on the intersection of interpolation crosses as follows

$$
\left[A(\mathcal{I}^{\leqslant k}_{s_k}, \mathcal{I}^{>k}_{t_k})\right]^{r_k}_{t_k, s_k=1} = \left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})\right] = A_k.
$$

The decomposition is illustrated by Fig. 1.

In [54] it is shown that if a tensor A is *exactly* given by tensor train format (6) with TT–ranks r_k, it is recovered from $\mathcal{O}(dnr^2)$ tensor entries[3] by formula (7). The approximate case was not considered. In this paper we fill this gap by considering the case when the right–hand side of (6) approximates a tensor with certain accuracy. We compare the accuracy provided by a general tensor train (6) and by the tensor train constructed from the entries of a given tensor by formula (7). Throughout the paper we assume that TT–ranks of (6) and (7) are the same, i.e., sets $\mathcal{I}^{\leqslant k} = \{\mathcal{I}^{\leqslant k}_{s_k}\}^{r_k}_{s_k=1}$ and $\mathcal{I}^{>k} = \{\mathcal{I}^{>k}_{t_k}\}^{r_k}_{t_k=1}$ have r_k elements each, and $A_k = \left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})\right]$ is $r_k \times r_k$ matrix, where r_k, $k = 1, \ldots, d-1$, are TT–ranks of (6). When a *choice* of $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$ is considered, it means that we choose r_k 'left' and 'right' multiindices $i_{\leqslant k} \in \mathcal{I}^{\leqslant k}$, $i_{>k} \in \mathcal{I}^{>k}$, i.e. columns and rows in the unfolding matrix $A^{\{k\}}$.

2.7. Tensor norms

For tensors we will use the Chebyshev and the Frobenius norms defined as follows

$$
|A| = \max_{i_1, \ldots, i_d} |A(i_1, \ldots, i_d)|, \qquad \|A\|^2 = \sum_{i_1, \ldots, i_d} |A(i_1, \ldots, i_d)|^2.
$$

---

[3]We always assume $n_1 = n_2 = \ldots = n_d = n$ and $r_1 = \ldots = r_{d-1} = r$ in complexity estimates

3. Maximum–volume principle in higher dimensions

We consider a tensor A which is approximated by the TT format as follows

$$A(i_1, \ldots, i_d) \approx X(i_1, \ldots, i_d) = \sum_{\mathbf{s}} X^{(1)}(i_1, s_1) X^{(2)}(s_1, i_2, s_2) \cdots X^{(d)}(s_{d-1}, i_d),$$
$$|A - X| \leqslant E_C, \qquad \|A - X\| \leqslant E_F, \tag{8}$$

where E_C and E_F are known or estimated from computations or theoretical properties of A. We apply (2) to k–th unfolding and write the cross interpolation

$$A^{\{k\}}(i_{\leqslant k}, i_{>k}) \approx \tilde{A}^{\{k\}}(i_{\leqslant k}, i_{>k}) = \sum_{s_k, t_k} A^{\{k\}}(i_{\leqslant k}, \mathcal{I}_{t_k}^{>k}) \left[A(\mathcal{I}_{s_k}^{\leqslant k}, \mathcal{I}_{t_k}^{>k}) \right]^{-1} A^{\{k\}}(\mathcal{I}_{s_k}^{\leqslant k}, i_{>k}).$$

For $\left[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k} \right] = \operatorname{maxvol} A^{\{k\}}$ the accuracy is estimated by (4) as follows

$$|A^{\{k\}} - \tilde{A}^{\{k\}}| \leqslant (r_k + 1) \, \|A^{\{k\}} - X^{\{k\}}\|,$$
$$|A^{\{k\}} - \tilde{A}^{\{k\}}| \leqslant (r_k + 1)^2 |A^{\{k\}} - X^{\{k\}}|.$$

We can safely omit the superscript for unfoldings when we use the elementwise notation, since the grouping of indices clearly defines the shape of the resulted matrix. The equation for the unfolding is recast for the tensor as follows

$$A(i_1, \ldots, i_d) = \sum_{s_k, t_k} A(i_1, \ldots, i_k, \mathcal{I}_{t_k}^{>k}) \left[A(\mathcal{I}_{s_k}^{\leqslant k}, \mathcal{I}_{t_k}^{>k}) \right]^{-1} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}, \ldots, i_d) + E(i_1, \ldots, i_d),$$
$$|E| \leqslant (r_k + 1) \, E_F, \qquad |E| \leqslant (r_k + 1)^2 E_C. \tag{9}$$

The interpolation step splits a d–tensor into a 'product' of two tensors, which have k and $d - k$ free indices, respectively. The same splitting is made for the *Tree–Tucker* format (later reintroduced as the tensor train format) in [53], and for the HT format in [29, 44, 4]. Using (9) recurrently, we estimate the accuracy of the interpolation–based formula (7).

Lemma 1. *If a tensor A satisfies* (8), *and* $[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}] = \operatorname{maxvol} A^{\{k\}}$ *for* $k = 1, \ldots, d-1$, *then*

$$A(\mathcal{I}^{\leqslant k-1}, i_k, i_{k+1}, \mathcal{I}^{>k+1}) = \sum_{s_k, t_k} A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}_{t_k}^{>k}) \left[A(\mathcal{I}_{s_k}^{\leqslant k}, \mathcal{I}_{t_k}^{>k}) \right]^{-1} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})$$
$$+ E(\mathcal{I}^{\leqslant k-1}, i_k, i_{k+1}, \mathcal{I}^{>k+1}),$$

where $|E|$ is estimated by (9).

Proof. In (9) we reduce free indices $i_{\leqslant k-1}$ to the subset $\mathcal{I}^{\leqslant k-1}$ and $i_{>k+1}$ to $\mathcal{I}^{>k+1}$. $\qquad \square$

Lemma 2. *Under assumptions of Lemma 1, if for some $1 \leqslant p < k < q \leqslant d$ for subtensors*

$$A_{\lhd} = \left[A(\mathcal{I}^{\leqslant p-1}, i_{p:k}, \mathcal{I}^{>k}) \right], \qquad A_{\rhd} = \left[A(\mathcal{I}^{\leqslant k}, i_{k+1:q}, \mathcal{I}^{>q}) \right],$$

it holds $A_{\lhd} = T_{\lhd} + E_{\lhd}$ and $A_{\rhd} = T_{\rhd} + E_{\rhd}$ with $|E_{\lhd}| \leqslant \varepsilon|A|$ and $|E_{\rhd}| \leqslant \varepsilon|A|$, then

$$A(\mathcal{I}^{\leqslant p-1}, i_{p:q}, \mathcal{I}^{>q}) = \sum_{s_k, t_k} T_{\lhd}(\mathcal{I}^{\leqslant p-1}, i_{p:k}, \mathcal{I}_{t_k}^{>k})(A_k^{-1})_{t_k, s_k} T_{\rhd}(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1:q}, \mathcal{I}^{>q})$$
$$+ \mathcal{E}(\mathcal{I}^{\leqslant p-1}, i_{p:q}, \mathcal{I}^{>q}),$$
$$\frac{|\mathcal{E}|}{|A|} \leqslant (2 + \varepsilon \kappa_k) \varepsilon r_k + \frac{|E|}{|A|}, \qquad \kappa_k = r_k |A| \|A_k^{-1}|, \quad A_k = \left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}) \right],$$

where $|E|$ is estimated by (9).
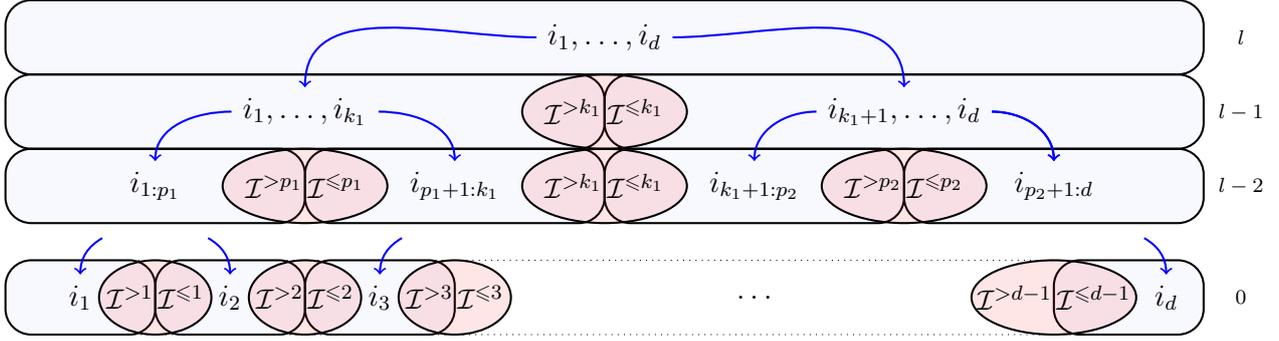
Figure 2: Interpolation steps on the balanced dimension tree for Thm. 1, cf. Fig. 1.

Proof. Like in the previous lemma, we reduce the elementwise equation (9) to the subset of indices with $i_{\leqslant p-1} \in \mathcal{I}^{\leqslant p-1}$ and $i_{>q} \in \mathcal{I}^{>q}$ and obtain

$$A(\mathcal{I}^{\leqslant p-1}, i_{p:q}, \mathcal{I}^{>q}) = \sum\nolimits_{s_k, t_k} A_\triangleleft(\mathcal{I}^{\leqslant p-1}, i_{p:k}, \mathcal{I}^{>k}_{t_k})(A_k^{-1})_{t_k, s_k} A_\triangleright(\mathcal{I}^{\leqslant k}_{s_k}, i_{k+1:q}, \mathcal{I}^{>q}) + E(\mathcal{I}^{\leqslant p-1}, i_{p:q}, \mathcal{I}^{>q}),$$

where $|E|$ is estimated by (9). We write

$$A_\triangleleft A_k^{-1} A_\triangleright = (T_\triangleleft + E_\triangleleft) A_k^{-1} (T_\triangleright + E_\triangleright) = T_\triangleleft A_k^{-1} T_\triangleright + A_\triangleleft A_k^{-1} E_\triangleright + E_\triangleleft A_k^{-1} A_\triangleright - E_\triangleleft A_k^{-1} E_\triangleright.$$

Since $A_k = \left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}) \right]$ is the maximum–volume submatrix in $A^{\{k\}} = \left[A(i_{\leqslant k}, i_{>k}) \right]$, it dominates by (5) in the corresponding rows and columns of the unfolding and *a fortiori* in $A_\triangleleft$ and $A_\triangleright$, i.e. $|A_\triangleleft A_k^{-1}| \leqslant 1$ and $|A_k^{-1} A_\triangleright| \leqslant 1$. It follows that

$$\left| A_\triangleleft A_k^{-1} E_\triangleright \right| \leqslant r_k \left| A_\triangleleft A_k^{-1} \right| |E_\triangleright| \leqslant r_k \varepsilon |A|, \qquad \left| E_\triangleleft A_k^{-1} A_\triangleright \right| \leqslant r_k \varepsilon |A|, \qquad \text{and}$$

$$\left| E_\triangleleft A_k^{-1} E_\triangleright \right| \leqslant r_k^2 \varepsilon^2 \left| A_k^{-1} \right| |A|^2 = r_k^2 \varepsilon^2 \left| A_k^{-1} \right| |A|^2 = r_k \kappa_k \varepsilon^2 |A|,$$

which completes the proof. $\qquad\qquad\square$

Theorem 1. *If a tensor A satisfies* (8), *and E_F and/or E_C are sufficiently small, then $\tilde{A}$ given by the cross interpolation formula* (7) *with interpolation sets $\left[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k} \right] = \text{maxvol } A^{\{k\}}$ provides the accuracy*

$$\begin{aligned}
|A - \tilde{A}| &\leqslant (2r + \kappa r + 1)^{\lceil \log_2 d \rceil} (r+1) \, E_F, \\
|A - \tilde{A}| &\leqslant (2r + \kappa r + 1)^{\lceil \log_2 d \rceil} (r+1)^2 E_C,
\end{aligned} \tag{10}$$

where $r = \max r_k$, $\kappa = \max \kappa_k$. By 'sufficiently small' we mean such values of E_F and/or E_C that the corresponding estimate provides $|A - \tilde{A}|/|A| < 1$.

Proof. We will use the *dimension tree* suggested in [53], see Fig. 2. The interpolation step (9) splits a given group of indices $i_p, \ldots, i_q$ in two parts $i_p, \ldots, i_k$ and $i_{k+1}, \ldots, i_q$, and introduces the auxiliary summation over the sets $\mathcal{I}^{\leqslant k}$ and $\mathcal{I}^{>k}$ at the point of splitting. No more than two auxiliary sets appear in each subtensor when the decomposition goes from the whole tensor down to the *leaves* $\left[A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k}) \right]$, which constitute (7). All leaves consist of the original entries of A, therefore we have zero error at the ground level. The interpolation error at the level 1 is estimated by Lemma 1 as $|\mathcal{E}_1| \leqslant \varepsilon_1 |A|$, where

$$\varepsilon_1 = \min \left\{ (r+1) \frac{E_F}{|A|}, \ (r+1)^2 \frac{E_C}{|A|} \right\}. \tag{11}$$

7

When we move up by one level of the dimension tree, the error is amplified as shown by Lemma 2 and the relative error in Chebyshev norm is $|\mathcal{E}_m| \leqslant \varepsilon_m |A|$, where ε_m propagates as follows

$$\varepsilon_{m+1} = (2 + \varepsilon_m \kappa)\varepsilon_m r + \varepsilon_1 \leqslant (2 + \kappa)\varepsilon_m r + \varepsilon_m = (2r + \kappa r + 1)\varepsilon_m.$$

Here we use the inequality $\varepsilon_m < 1$ provided by the assumption that E_F and E_C are sufficiently small. It follows that $\varepsilon_l \leqslant (2r + \kappa r + 1)^{l-1}\varepsilon_1$. For a balanced tree $2^l \leqslant 2d$ and $l \leqslant \lceil \log_2 d \rceil + 1$, which completes the proof. $\qquad \square$

Remark 1. *The value $\kappa_k = r_k |A| |A_k^{-1}|$ is closely related to the* condition number $\kappa_C(A_k)$ *of the submatrix A_k w.r.t. the Chebyshev norm. In general $|A| \geqslant |A_k|$ and $\kappa_k \geqslant r_k |A_k| |A_k^{-1}| = \kappa_C(A_k)$. However, as shown in [20], the ratio of the Chebyshev norms of a matrix and its maximum–volume submatrix is bounded as $|A|/|A_k| \leqslant 2r_k^2 + r_k$, and often does not grow with rank. Therefore, $\kappa_C(A_k) \leqslant \kappa_k \leqslant (2r_k^2 + r_k)\kappa_C(A_k)$, and usually $\kappa_k \simeq \kappa_C(A_k)$. A similar value $\|A\|_2 \|A_k^{-1}\|_2$ appears in accuracy estimate in the pioneering paper on the cross interpolation of matrices [24, Eq. (1.5)].*

Remark 2. *The splitting of indices in the balanced dimension tree was used to estimate the accuracy of the interpolation in the HT format [4]. The constructive algorithm proposed in [4] approximates a given tensor using $\mathcal{O}(dr^3 + dnr^2 \log d)$ entries [4, Lemma 7]. Under mild assumptions, the upper bound for accuracy of this approximation [4, Remark 15] is of the level $\mathcal{O}(r^{2d})$, but the authors acknowledge that "a further analysis might give more insights into the error amplification". The analysis developed in this paper is equally applicable to the HT format. If a balanced dimension tree is used, and interpolation indices are chosen by the maximum-volume principle, the result of Thm. 1 extends to the HT case, and the quasioptimality constant is reduced from $\mathcal{O}(r^{2d})$ to $(2r + \kappa r + 1)^{\lceil \log_2 d \rceil + 2}$.*

The result of Thm. 1 can be interpreted as the *existence* of a sufficiently good TT approximation computed from a few entries of a tensor by formula (7), provided that the accurate representation in the TT format (6) is possible. The coefficient $\mathcal{O}(r^{\lceil \log_2 d \rceil + 2})$ can be also understood as upper bound for the ratio of the accuracy of the *best cross* interpolation (7) and the *best possible* accuracy of the approximation (6) of the same TT–ranks. Thm. 1 is *constructive* and prescribes the choice of the interpolation sets $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$ to achieve the quasioptimal accuracy. However, the actual *computation* of maximum–volume sets in unfoldings $A^{\{k\}}$ is impossible due to their restrictively large sizes. In the next sections we consider the nested choice of the interpolation sets which reduces the search space.

4. Nested maximum volume indices

In this section we switch to the ultimately unbalanced dimension tree, which splits indices one-by-one, see Fig. 3. In [49] this tree has been used to develop the TT–SVD algorithm which approximates a given d–tensor by the TT format. We apply the same algorithm substituting the SVD approximation steps by the interpolation. As in the previous section, we estimate the accuracy of the resulted approximation w.r.t. the best possible approximation of the same TT–ranks.

Given a tensor $A = [A(i_1, \ldots, i_d)]$ that is approximated by the tensor train (8), we apply the interpolation formula (9) and separate the rightmost index from the others as follows

$$A(i_1, \ldots, i_d) = \sum_{\substack{s_{d-1} \\ t_{d-1}}} A(i_{\leqslant d-1}, \mathcal{I}^{>d-1}_{t_{d-1}}) \left[A(\mathcal{I}^{\leqslant d-1}_{s_{d-1}}, \mathcal{I}^{>d-1}_{t_{d-1}}) \right]^{-1} A(\mathcal{I}^{\leqslant d-1}_{s_{d-1}}, i_d) + E_{d-1}(i_1, \ldots, i_d),$$
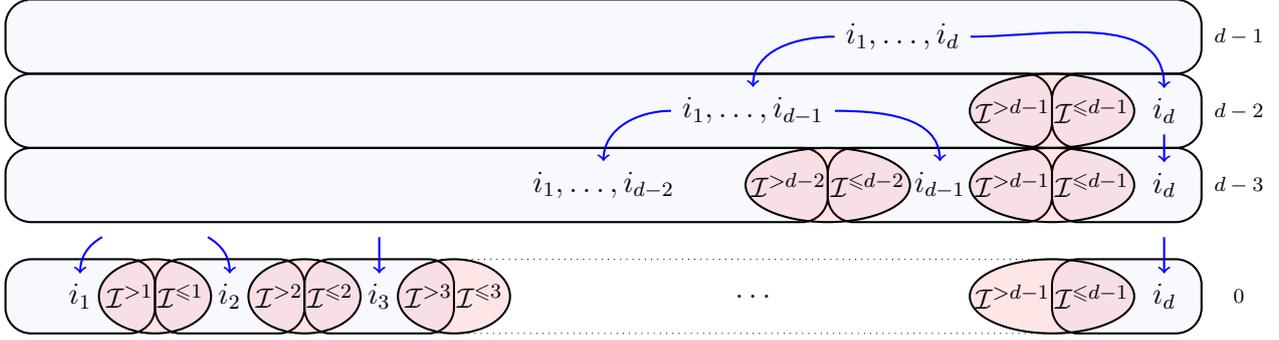
Figure 3: Interpolation steps on the unbalanced dimension tree for Thm. 2, cf. Fig. 2.

where $\left[\mathcal{I}^{\leqslant d-1}, \mathcal{I}^{>d-1}\right] = \text{maxvol}\left[A(i_{\leqslant d-1}, i_d)\right]$. Then we interpolate the subtensor with $d-1$ free indices and separate the rightmost free index as follows

$$A(i_{\leqslant d-1}, \mathcal{I}^{>d-1}) = \sum_{\substack{s_{d-2} \\ t_{d-2}}} A(i_{\leqslant d-2}, \mathcal{I}_{t_{d-2}}^{>d-2}) \left[A(\mathcal{I}_{s_{d-2}}^{\leqslant d-2}, \mathcal{I}_{t_{d-2}}^{>d-2})\right]^{-1} A(\mathcal{I}_{s_{d-2}}^{\leqslant d-2}, i_{d-1}, \mathcal{I}^{>d-1})$$
$$+ E_{d-2}(i_{\leqslant d-1}, \mathcal{I}^{>d-1}),$$

where $\left[\mathcal{I}^{\leqslant d-2}, \mathcal{I}^{>d-2}\right] = \text{maxvol}\left[A(i_{\leqslant d-2}, i_{d-1}\mathcal{I}^{>d-1})\right]$. The elements of $\mathcal{I}^{>d-2}$ are now chosen not from all the possible values of bi-index $\overline{i_{d-1}i_d}$ but from the reduced set $\overline{i_{d-1}\mathcal{I}^{>d-1}}$, where index i_d is restricted to r_{d-1} elements of $\mathcal{I}^{>d-1}$. Hereinafter we omit the overline for the sake of clarity, since the use of comma in the elementwise notation is sufficient to show which indices are grouped together. The maximum–volume subsets $\mathcal{I}^{>d-1}$ and $\mathcal{I}^{>d-2}$ are *right–nested* (cf. [54]) which means that $i_{>d-2} \in \mathcal{I}^{>d-2}$ leads to $i_{>d-1} \in \mathcal{I}^{>d-1}$. As the interpolation develops further, it holds

$$i_{>k} \in \mathcal{I}^{>k} \Rightarrow i_{>k+1} \in \mathcal{I}^{>k+1}, \qquad k = d-1, \ldots, 1. \tag{12}$$

Theorem 2. *If a tensor A satisfies* (8), *then $\tilde{A}$ given by* (7) *with*

$$\left[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}\right] = \text{maxvol}\left[A(i_{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1})\right], \qquad k = d-1, \ldots, 1,$$

provides the following accuracy

$$|A - \tilde{A}| \leqslant \frac{r^{d-1} - 1}{r - 1}(r+1)\ E_F,$$
$$|A - \tilde{A}| \leqslant \frac{r^{d-1} - 1}{r - 1}(r+1)^2 E_C. \tag{13}$$

Proof. At the first level of the dimension tree the interpolation writes as follows

$$A(i_1, i_2\mathcal{I}^{>2}) = \sum_{s_1, t_1} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} A(\mathcal{I}_{s_1}^{\leqslant 1}, i_2\mathcal{I}^{>2}) + E_1(i_1, i_2\mathcal{I}^{>2}),$$

where $A_1 = [A(\mathcal{I}^{\leqslant 1}, \mathcal{I}^{>1})]$. Since $[\mathcal{I}^{\leqslant 1}, \mathcal{I}^{>1}] = \text{maxvol}\left[A(i_1, i_2\mathcal{I}^{>2})\right]$, it holds $|E_1| \leqslant \varepsilon_1|A|$, where ε_1 is defined by (11). This proves the statement of the theorem for $d = 2$, and constitutes the base of recursion. Now we suppose that at the level k of the tree it holds

$$A(i_{\leqslant k}, \mathcal{I}^{>k}) = \sum_{\substack{s_1, \ldots, s_{k-1} \\ t_1, \ldots, t_{k-1}}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} \cdots (A_{k-1}^{-1})_{t_{k-1}, s_{k-1}} A(\mathcal{I}_{s_{k-1}}^{\leqslant k-1}, i_k, \mathcal{I}^{>k}) + \mathcal{E}_k(i_{\leqslant k}, \mathcal{I}^{>k}),$$

9

and $|\mathcal{E}_k| \leqslant \frac{r^{k-1}-1}{r-1}\varepsilon_1|A|$. Interpolation at the next level writes as follows

$$A(i_{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1}) = \sum\nolimits_{s_k,t_k} A(i_{\leqslant k}, \mathcal{I}_{t_k}^{>k})(A_k^{-1})_{t_k,s_k} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1}) + E_k(i_{\leqslant k+1}, \mathcal{I}^{>k+1}).$$

Using the previous equation we obtain

$$A(i_{\leqslant k+1}, \mathcal{I}^{>k+1}) = \sum\nolimits_{\substack{s_1\ldots s_k\\ t_1\ldots t_k}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1,s_1} \cdots (A_k^{-1})_{t_k,s_k} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})$$
$$+ \underbrace{\sum\nolimits_{s_k,t_k} \mathcal{E}_k(i_{\leqslant k}, \mathcal{I}_{t_k}^{>k})(A_k^{-1})_{t_k,s_k} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1}) + E_k(i_{\leqslant k+1}, \mathcal{I}^{>k+1})}_{\mathcal{E}_{k+1}(i_{\leqslant k+1}, \mathcal{I}^{>k+1})}.$$

Since $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$ are chosen by the maximum–volume principle, the approximation error is bounded, $|E_k| \leqslant \varepsilon_1|A|$, and the submatrix $A_k = [A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})]$ is dominant

$$\left| \sum\nolimits_{s_k} (A_k^{-1})_{t_k,s_k} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1}) \right| \leqslant 1,$$

that gives $|\mathcal{E}_{k+1}| \leqslant r|\mathcal{E}_k| + |E_k|$. Using the assumption of the recursion, we write

$$|\mathcal{E}_{k+1}| \leqslant r\frac{r^{k-1}-1}{r-1}\varepsilon_1|A| + \varepsilon_1|A| = \frac{r^k-1}{r-1}\varepsilon_1|A|.$$

Plugging $k = d-1$, we complete the proof. $\qquad\square$

Lemma 3. *If $\tilde{A}$ is given by (7) and the interpolation sets are right–nested as shown by (12), then for all $k = 1, \ldots, d-1$ it holds*

$$\tilde{A}(i_1, \ldots, i_k, \mathcal{I}^{>k}) = \sum\nolimits_{\substack{s_1,\ldots,s_{k-1}\\ t_1,\ldots,t_{k-1}}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1,s_1} \cdots (A_{k-1}^{-1})_{t_{k-1},s_{k-1}} A(\mathcal{I}_{s_{k-1}}^{\leqslant k-1}, i_k, \mathcal{I}^{>k}).$$

Proof. To prove the statement of the lemma for $k = d-1$, in (7) we restrict $i_{>d-1}$ to $\mathcal{I}^{>d-1}$. The last core reduces to

$$\left[A(\mathcal{I}^{\leqslant d-1}, i_d)\right]_{i_d \in \mathcal{I}^{>d-1}} = \left[A(\mathcal{I}^{\leqslant d-1}, \mathcal{I}^{>d-1})\right] = A_{d-1},$$

and cancels out with the neighbouring A_{d-1}^{-1}. This constitute the base of recursion. Suppose now that the statement holds for $k = p+1$, i.e.

$$\tilde{A}(i_1, \ldots, i_{p+1}, \mathcal{I}^{>p+1}) = \sum\nolimits_{\substack{s_1,\ldots,s_p\\ t_1,\ldots,t_p}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1,s_1} \cdots (A_p^{-1})_{t_p,s_p} A(\mathcal{I}_{s_p}^{\leqslant p}, i_{p+1}, \mathcal{I}^{>p+1}).$$

Consider this equation for $i_{>p} \in \mathcal{I}^{>p}$, that by (12) assumes $i_{>p+1} \in \mathcal{I}^{>p+1}$. The rightmost core reduces as follows

$$\left[A(\mathcal{I}^{\leqslant p}, i_{p+1}, \mathcal{I}^{>p+1})\right]_{i_{>p} \in \mathcal{I}^{>p}} = \left[A(\mathcal{I}^{\leqslant p}, \mathcal{I}^{>p})\right] = A_p,$$

and cancels out with A_p^{-1}. This proves the statement for $k = p$ and the lemma by recursion. $\quad\square$

Since $\left[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}\right] = \mathrm{maxvol}\left[A(i_{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1})\right]$, the quasioptimal estimate (9) holds for the entries of this subtensor only. However, $A_k = \left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})\right]$ is nonsingular and we can interpolate the whole unfolding $A^{\{k\}}$ by the cross based on A_k with some (presumably worse) accuracy estimate

$$A(i_1, \ldots, i_d) = \sum\nolimits_{s_k,t_k} A(i_1, \ldots, i_k, \mathcal{I}_{t_k}^{>k})(A_k^{-1})_{t_k,s_k} A(\mathcal{I}_{s_k}^{\leqslant k}, i_{k+1}, \ldots, i_d) + \hat{E}_k(i_1, \ldots, i_d). \quad (14)$$

The following theorem estimates the accuracy of the same interpolation $\tilde{A}$ as in the previous theorem w.r.t. the errors $|\hat{E}_k|$ in (14).

Theorem 3. *Under the conditions of Thm. 2 assume additionally that the interpolation* (14) *provides sufficiently good accuracy* $\hat{\varepsilon} = \max |\hat{E}_k|/|A|$. *Then*

$$|A - \tilde{A}| \leqslant \frac{dr\hat{\varepsilon}}{1 - d\kappa r\hat{\varepsilon}}|A|, \tag{15}$$

where κ *is defined in* (10). *By 'sufficiently small' here we mean such* $\hat{\varepsilon}$, *that the denominator of* (15) *does not approach zero.*

Proof. The interpolation sets (12) have been constructed from right to left according to the dimension tree on Fig. 3. In order to estimate the accuracy we separate indices one-by-one with the interpolation (14) proceeding from left to right. We begin with

$$A(i_1, \ldots, i_d) = \sum_{s_1, t_1} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} A(\mathcal{I}_{s_1}^{\leqslant 1}, i_{>1}) + \hat{E}_1(i_1, \ldots, i_d),$$

and $|\mathcal{E}_1| = |\hat{E}_1| \leqslant \hat{\varepsilon}|A|$. On the second step we write

$$A(i_1, \ldots, i_d) = \sum_{s_2, t_2} A(i_1, i_2, \mathcal{I}_{t_2}^{>2})(A_2^{-1})_{t_2, s_2} A(\mathcal{I}_{s_2}^{\leqslant 2}, i_{>2}) + \hat{E}_2(i_1, \ldots, i_d).$$

Restricting i_1 to $\mathcal{I}^{\leqslant 1}$ and plugging the result into the previous equation, we obtain

$$A(i_1, \ldots, i_d) = \sum_{\substack{s_1, s_2 \\ t_1, t_2}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} A(\mathcal{I}_{s_1}^{\leqslant 1}, i_2, \mathcal{I}_{t_2}^{>2})(A_2^{-1})_{t_2, s_2} A(\mathcal{I}_{s_2}^{\leqslant 2}, i_{>2})$$

$$+ \underbrace{\sum_{s_1, t_1} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} \hat{E}_2(\mathcal{I}_{s_1}^{\leqslant 1}, i_{>1}) + \mathcal{E}_1(i_1, \ldots, i_d)}_{\mathcal{E}_2(i_1, \ldots, i_d)}.$$

Since $[\mathcal{I}^{\leqslant 1}, \mathcal{I}^{>1}] = \text{maxvol}\,[A(i_1, i_2 \mathcal{I}^{>2})]$, the submatrix A_1 dominates in the corresponding rows $\left|\sum_{t_1} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1}\right| \leqslant 1$, and therefore $|\mathcal{E}_2| \leqslant (r+1)\hat{\varepsilon}|A|$.

The third interpolation step writes as follows

$$A(i_1, \ldots, i_d) = \sum_{s_3, t_3} A(i_{\leqslant 2}, i_3, \mathcal{I}_{t_3}^{>3})(A_3^{-1})_{t_3, s_3} A(\mathcal{I}_{s_3}^{\leqslant 3}, i_{>3}) + \hat{E}_3(i_1, \ldots, i_d).$$

Again, we restrict $i_{\leqslant 2}$ to $\mathcal{I}^{\leqslant 2}$ and plug the result into the previous equation.

$$A(i_1, \ldots, i_d) = \sum_{\substack{s_1, s_2, s_3 \\ t_1, t_2, t_3}} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} \cdots (A_3^{-1})_{t_3, s_3} A(\mathcal{I}_{s_3}^{\leqslant 3}, i_{>3}) + \mathcal{E}_3(i_1, \ldots, i_d),$$

$$\mathcal{E}_3(i_1, \ldots, i_d) = \sum_{\substack{s_1, s_2 \\ t_1, t_2}} \underbrace{A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} A(\mathcal{I}_{s_1}^{\leqslant 1}, i_2, \mathcal{I}_{t_2}^{>2})}_{\tilde{A}(i_1, i_2, \mathcal{I}_{t_2}^{>2})}(A_2^{-1})_{t_2, s_2} \hat{E}_3(\mathcal{I}_{s_2}^{\leqslant 2}, i_{>2}) + \mathcal{E}_2(i_1, \ldots, i_d).$$

We need to estimate the norm of the matrix in front of $\hat{E}_3$, avoiding the exponential amplification of the coefficient. To do this, we replace the 'piece' of the interpolation train with the subtensor of A. Since $A = \tilde{A} + \mathcal{E}$, the same holds for the subtensors $A(i_1, i_2, \mathcal{I}^{>2}) = \tilde{A}(i_1, i_2, \mathcal{I}^{>2}) + \mathcal{E}(i_1, i_2, \mathcal{I}^{>2})$, and using Lemma 3 we write

$$\sum_{s_1, t_1} A(i_1, \mathcal{I}_{t_1}^{>1})(A_1^{-1})_{t_1, s_1} A(\mathcal{I}_{s_1}^{\leqslant 1}, i_2, \mathcal{I}^{>2}) = A(i_1, i_2, \mathcal{I}^{>2}) - \mathcal{E}(i_1, i_2, \mathcal{I}^{>2}).$$

Substituting this into the previous equation, we use the domination of the maximum–volume submatrix A_2 to write $\left|\sum_{t_2} A(i_{\leqslant 2}, \mathcal{I}_{t_2}^{>2})(A_2^{-1})_{t_2, s_2}\right| \leqslant 1$ and obtain

$$|\mathcal{E}_3| \leqslant (2r+1)\hat{\varepsilon}|A| + \kappa r\hat{\varepsilon}|\mathcal{E}|.$$

The error accumulates similarly in further interpolation steps. Finally,

$$|\mathcal{E}| = |\mathcal{E}_{d-1}| \leqslant dr\hat{\varepsilon}|A| + d\kappa r\hat{\varepsilon}|\mathcal{E}|,$$

which completes the proof. □

11

Theorems 2 and 3 estimate the accuracy of the interpolation formula (7) with the same interpolation sets. In Thm. 2 the quasioptimality result is proven with the coefficient $\mathcal{O}(r^d)$, which is much larger than the one in (10), cf. the coefficient $\mathcal{O}(r^{2d})$ in [4]. Since the coefficient in (13) grows exponentially with the dimension, it can be hardly used in the real estimates. The result of Thm. 3 improves the estimate of Thm. 2 provided the errors $|\hat{E}_k|$ in (14) do not grow exponentially with d. In general we cannot provide such upper bound for $|\hat{E}_k|$. The estimate (15) is useful in special cases when the theoretical or numerical estimates available for the errors $|\hat{E}_k|$ are bounded or grow moderately with d.

Note that the nestedness of the interpolation sets is essential in the proof of Thm. 3. The result of Thm. 3 cannot be generalised to the 'fully' maximum–volume case described in Thm. 1.

5. Two–side nestedness and the interpolation property

In this section we consider the interpolation (7) where both left and right interpolation sets are nested, i.e. for all valid k it holds

$$i_{>k} \in \mathcal{I}^{>k} \;\Rightarrow\; i_{>k+1} \in \mathcal{I}^{>k+1}, \qquad i_{\leqslant k} \in \mathcal{I}^{\leqslant k} \;\Rightarrow\; i_{\leqslant k-1} \in \mathcal{I}^{\leqslant k-1}. \tag{16}$$

A naive way to construct such sets is to run the right–to–left interpolation pass explained in Sec. 4 and keep the right sets $\mathcal{I}^{>k}$ only. The left sets $\mathcal{I}^{\leqslant k}$ are computed by the left–to–right interpolation pass which separates the index i_1, then i_2, etc. We obtain

$$\left[\mathcal{J}^{\leqslant k}, \mathcal{I}^{>k}\right] = \text{maxvol}\left[A(i_{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1})\right], \quad \left[\mathcal{I}^{\leqslant k}, \mathcal{J}^{>k}\right] = \text{maxvol}\left[A(\mathcal{I}^{\leqslant k-1}i_k, i_{>k})\right].$$

Note that $\left[A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})\right]$ is not necessarily the maximum–volume submatrix neither in matrix $\left[A(i_{\leqslant k}, i_{k+1}\mathcal{I}^{>k+1})\right]$, nor in $\left[A(\mathcal{I}^{\leqslant k-1}i_k, i_{>k})\right]$, nor even in $\left[A(\mathcal{I}^{\leqslant k-1}i_k, i_{k+1}\mathcal{I}^{>k+1})\right]$, where they intersect. Therefore, we cannot use (4) to estimate the accuracy of (7) with these interpolation sets. Due to the restrictive sizes, the computation of the maximum volume submatrix is impossible even with implied nestedness. To make the problem tractable, we should further reduce the search space — the practical recipes will be discussed in the next section.

If both left and right interpolation sets are nested, Eq. (7) is indeed the *cross interpolation formula*, as shown by the following theorem.

Theorem 4. *For a tensor A, the approximation $\tilde{A}$ given by (7) with indices $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$ satisfying (16), is exact on the positions of all entries evaluated in a tensor*

$$A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k}) = \tilde{A}(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k}), \qquad k = 1, \ldots, d. \tag{17}$$

Proof. It is sufficient to repeat the arguments from the proof of Lemma 3 for the left and right interpolation sets. $\qquad\square$

A $n_1 \times n_2$ matrix A of rank r is defined by $(n_1 + n_2)r - r^2$ parameters, e.g. by $n_1 r + n_2 r + r$ elements of the SVD decomposition $A = USV^*$ minus $r(r+1)$ normalisation constraints $U^*U = I$, $V^*V = I$. The cross interpolation formula (2) recovers a rank–r matrix from $(n_1 + n_2)r - r^2$ entries, if a submatrix $[A(\mathcal{I}^1, \mathcal{I}^2)]$ is nonsingular. If rank $A > r$, formula (2) provides the approximation $\tilde{A}$, which is exact on $(n_1 + n_2)r - r^2$ positions of a matrix. These facts are extended to the tensor case by the following theorem.

Theorem 5. *A tensor A with mode sizes $n_1, \ldots, n_d$ and TT–ranks $r_1, \ldots, r_{d-1}$ is defined by*

$$s = \sum_{k=1}^{d} r_{k-1}n_k r_k - \sum_{k=1}^{d-1} r_k^2$$

parameters. If the left and right interpolation sets satisfy (16)*, and the submatrices* $A_k = [A(\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k})]$*,* $k = 1, \ldots, d-1$*, are nonsingular, formula* (7) *recovers* A *from exactly* s *entries. If a tensor* A *is not given by* (6) *exactly, formula* (7) *interpolates it on at least* s *positions.*

Proof. The first statement is proven in [57, Prop. A.3]. Taking into account the result of Thm. 4, the second and the third statements require to calculate the total number of tensor entries in all subtensors in (17). Each block $\left[A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k})\right]$ consists of $r_{k-1}n_k r_k$ elements of a tensor, but some entries contribute to more than one block. For example, if (16) holds, subtensors $[A(i_1, \mathcal{I}^{>1})]$ and $[A(\mathcal{I}^{\leqslant 1}, i_2, \mathcal{I}^{>2})]$ intersect by the submatrix $A_1 = [A(\mathcal{I}^{\leqslant 1}, \mathcal{I}^{>1})]$, which has r_1^2 elements. Similarly, $\left[A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k})\right]$ and $\left[A(\mathcal{I}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})\right]$ have r_k^2 common elements in the submatrix A_k.

The common elements of $\left[A(\mathcal{I}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})\right]$ and $[A(\mathcal{I}^{\leqslant p-1}, i_p, \mathcal{I}^{>p})]$ are described by the following conditions

$$i_{\leqslant p-1} \in \mathcal{I}^{\leqslant p-1}, \quad i_{\leqslant k} \in \mathcal{I}^{\leqslant k}, \qquad i_{>p} \in \mathcal{I}^{>p}, \quad i_{>k+1} \in \mathcal{I}^{>k+1}.$$

If $p < k$, they are reduced by (16) to $\{i_{\leqslant k} \in \mathcal{I}^{\leqslant k}, i_{>p} \in \mathcal{I}^{>p}\} \subset \{i_{\leqslant k} \in \mathcal{I}^{\leqslant k}, i_{>k} \in \mathcal{I}^{>k}\}$. This means that for $p = 1, \ldots, k-1$ all common entries of $[A(\mathcal{I}^{\leqslant p-1}, i_p, \mathcal{I}^{>p})]$ and $\left[A(\mathcal{I}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})\right]$ belong to $A_k = [A^{\leqslant k}, \mathcal{I}^{>k}]$. The total number of entries shared between blocks equals $\sum_{k=1}^{d-1} r_k^2$, which completes the proof. $\square$

6. Interpolation algorithms for matrices and tensors

We start this section with a short overview of the cross interpolation algorithms for matrices. The idea of reconstruction and approximation of a matrix from several columns and rows by the *skeleton decomposition* (2) or the *pseudoskeleton decomposition* $\tilde{A} = CGR$, $C = [A(i, \mathcal{J})]$, $R = [A(\mathcal{I}, j)]$, has been suggested by Goreinov and Tyrtyshnikov [23]. In [24] the accuracy of the pseudoskeleton approximation has been studied, and it has been pointed out that a good cross should intersect by a well bounded submatrix. The connection with the maximum–volume submatrix has been mentioned in [24], and the maximum–volume principle has been presented in more detail in [21].

The search for the maximum–volume submatrix *per se* is an NP–hard problem [6]. For practical computations, it is necessary to find a *sufficiently good* submatrix reasonably fast. The alternating direction algorithm has been proposed in [64], which adaptively increases the size of the interpolation cross following the maximum–volume principle at each step, and computes the approximation of a matrix in linear time w.r.t. the size. The greedy algorithm of such kind, equivalent to the Gaussian elimination with rook pivoting, was then suggested by Bebendorf [7]. Due to its particular simplicity, it has become widely known as the *adaptive cross approximation* (ACA). In practical computations, ACA and similar methods with minimal information are liable to breakdowns, i.e. they may quit when a good approximation is not yet obtained. A cheap remedy proposed in [58] is to check the accuracy on the random set of entries and restart the algorithm if necessary.[4] Another well–known sampling method is the CUR algorithm of Mahoney et al [18], which is the pseudoskeleton CGR decomposition where positions of the rows and columns are chosen randomly.

The accuracy of the maximum–volume cross approximation is estimated for any matrix [21, 60, 22]. Algorithms which use a few elements (e.g. ACA) are *heuristic* and construct the approximation which can be arbitrarily bad for other matrix elements. The accuracy of such

---

[4]Published in English later as [52, Alg. 3]

Algorithm 1 Greedy cross interpolation algorithm for tensor trains

---
Require: Function to compute entries of a tensor $A = [A(i_1, \ldots, i_d)]$
Ensure: Cross interpolation (7) with the nested interpolation sets (16)
  1: $\mathcal{I}^{\leqslant k} = \emptyset$, $\mathcal{I}^{>k} = \emptyset$, $k = 1, \ldots, d$, $\tilde{A} = 0$, $E = A$
  2: **while** $|A - \tilde{A}|$ is not sufficiently small **do**
  3:      Find a pivot $i^\star = (i_1^\star, \ldots, i_d^\star)$ s.t. $|E(i_1^\star, \ldots, i_d^\star)| \simeq |A - \tilde{A}|$
  4:      Add $i_{\leqslant k}^\star$ to $\mathcal{I}^{\leqslant k}$, and $i_{>k}^\star$ to $\mathcal{I}^{>k}$, $k = 1, \ldots, d-1$
  5:      Update the interpolation $\tilde{A}$ by (7)
  6: **end while**

---

algorithms can be estimated in special cases, e.g. for matrices generated by asymptotically smooth functions on quasi–uniform grids, see [7, 66], cf. [65] in many dimensions.

The existing cross interpolation algorithms for tensors can be classified similarly. The skeleton decomposition is generalised to the tensor case in [54] by formula (7). The *existence result* is generalised from the matrix case [21, 60, 22] to the TT case by Thm. 1. Algorithm proposed in [54] approximates the maximum–volume positions in the ALS way, similarly to the one from [64].

A greedy cross interpolation algorithm for the TT format can be suggested similarly to the matrix case, see Alg. 1. Similarly to the ACA for matrices, Alg. 1 relies on the interpolation property for the tensor trains, established by Thm. 4. On each step Alg. 1 searches for a pivot $i^\star$ where the error of the current approximation is (quasi)maximum in modulus. Then it adds the indices of $i^\star = \overline{i_{\leqslant k}^\star i_{>k}^\star}$ to all subsets $\mathcal{I}^{\leqslant k}$ and $\mathcal{I}^{>k}$, $k = 1, \ldots, d-1$, to maintain the two–side nestedness (16). The updated interpolation is exact on all *lines* $(i_1^\star, \ldots, i_{k-1}^\star, i_k, i_{k+1}^\star, \ldots, i_d^\star)$, $i_k = 1, \ldots, n_k$, $k = 1, \ldots, d$.

The full pivoting in higher dimensions is impossible due to the curse of dimensionality, and we need cheaper alternatives to find a new pivot and estimate the accuracy for the stopping criterion. There are several options how to do this:

1. to choose indices randomly, following the tensor–CUR algorithm of Mahoney et al [47];
2. to choose the maximum in modulus element of the current residual among a randomly sampled set;
3. to choose the pivot from a *restricted set*, then check the accuracy of the approximation over a random set of entries, and restart if necessary, similarly to [52, Alg. 3].

The *restricted pivoting set* can naturally arise from the *locality* requirement. By this we mean that with a new pivot we should modify only a few interpolation sets $\mathcal{I}^{\leqslant k}$ and $\mathcal{I}^{>k}$ and increase only a few TT–ranks of the approximation, not all of them. To put it differently, a pivoting algorithm should update only a few TT–cores of (7) at each step, similarly to the ALS and DMRG algorithms introduced in quantum physics.

Following the two-site DMRG algorithm, in Alg. 2 we choose a new pivot $i^\star$ in the *supercore* $A' = \left[A(\mathcal{I}^{\leqslant k-1}, i_k, i_{k+1}, \mathcal{I}^{>k+1}) \right]$. This choice provides $i_{\leqslant k-1}^\star \in \mathcal{I}^{\leqslant k-1}$, and by (16) $i_{\leqslant p}^\star \in \mathcal{I}^{\leqslant p}$ for $p \leqslant k - 1$. Similarly, $i_{>k+1}^\star \in \mathcal{I}^{>k+1}$ and by nestedness $i_{>p}^\star \in \mathcal{I}^{>p}$ for $p \geqslant k + 1$. When we add $i_{\leqslant k}^\star$ to $\mathcal{I}^{\leqslant k}$ and $i_{>k}^\star$ to $\mathcal{I}^{>k}$, the two–side nestedness (16) is preserved *ipso facto*.

The greedy algorithm with pivoting in A' can be implemented as a simple modification of the cross interpolation algorithm TT–RC from [59]. The TT–RC algorithm is of the DMRG type, which means that it updates two neighbouring TT–cores at each step, computing the matrix A' in full. The proposed Alg. 2 substitutes this step with the cross interpolation and

---

Algorithm 2 Greedy restricted cross interpolation algorithm for tensor trains

---

Require: Function to compute entries of a tensor $A = [A(i_1, \ldots, i_d)]$

Ensure: Cross interpolation (7) with the nested interpolation sets (16)

 1: Choose $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$, $k = 1, \ldots, d$, which satisfy (16), and compute $\tilde{A}$ by (7)
 2: **while** stopping criterion is not satisfied **do**
 3:    **for** $k = 1, \ldots, d-1$ **do** {Left–to–right half–sweep}
 4:       Apply the cross interpolation (e.g. [52, Alg. 3]) to the DMRG supercore matrix $\left[A(\mathcal{I}^{\leqslant k-1} i_k, i_{k+1} \mathcal{I}^{>k+1}) \right]$. Use interpolation sets $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$ as the initial guess, and expand them to $\mathcal{J}^{\leqslant k}, \mathcal{J}^{>k}$, adding a few crosses. Obtain (18) such that $\mathcal{I}^{\leqslant k} \subset \mathcal{J}^{\leqslant k}$ and $\mathcal{I}^{>k} \subset \mathcal{J}^{>k}$
 5:       Substitute $\mathcal{I}^{\leqslant k}$ and $\mathcal{I}^{>k}$ by the expanded sets $\mathcal{J}^{\leqslant k}$ and $\mathcal{J}^{>k}$
 6:    **end for**
 7:    Perform right–to-left half–sweep in the same way
 8: **end while**

---

approximates

$$A(\mathcal{I}^{\leqslant k-1} i_k, i_{k+1} \mathcal{I}^{>k+1}) \approx \sum_{s_k, t_k} A(\mathcal{I}^{\leqslant k-1} i_k, \mathcal{J}^{>k}_{t_k}) \left[A(\mathcal{J}^{\leqslant k}_{s_k}, \mathcal{J}^{>k}_{t_k}) \right]^{-1} A(\mathcal{J}^{\leqslant k}_{s_k}, i_{k+1} \mathcal{I}^{>k+1}), \quad (18)$$

where $\mathcal{J}^{\leqslant k}$ and $\mathcal{J}^{>k}$ are computed by the matrix cross interpolation algorithm, s.t.

$$\left[\mathcal{J}^{\leqslant k}, \mathcal{J}^{>k} \right] \simeq \mathrm{maxvol} \left[A(\mathcal{I}^{\leqslant k-1} i_k, i_{k+1} \mathcal{I}^{>k+1}) \right].$$

Since we already have the interpolation sets $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$, which deliver a (hopefully accurate) approximation to the unfolding $A^{\{k\}}$ and hence to the supercore A', it is natural to re-use them and look for $\mathcal{J}^{\leqslant k}, \mathcal{J}^{>k}$ that contain $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$, respectively. The resulting *greedy* algorithm requires $\mathcal{O}(dnr^2)$ evaluation of tensor elements and $\mathcal{O}(dnr^3)$ additional operations, i.e. scales linearly in the mode size and very moderately in the TT–rank. The algorithm is rank–revealing, i.e. will not increase the TT–ranks of the approximation (7) over the TT–ranks of a given tensor.

7. Numerical experiments

The numerical results have been obtained using the Iridis High Performance Computing Facility at the University of Southampton.[5] Cross interpolation and auxiliary tensor train subroutines are written in FORTRAN90 by the author. The code was compiled using the Intel Composer and linked with LAPACK/BLAS subroutines provided with the MKL library.

In the experiments we use a very simple version of Alg. 2. On each step (Line 4) we improve the current approximation by adding *only one* cross to $[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}]$. The position of the new cross is computed as follows. First, a random sampling is performed on $r_{k-1} n_k + n_{k+1} r_{k+1}$ entries of the matrix $\left[A(\mathcal{I}^{\leqslant k-1} i_k, i_{k+1} \mathcal{I}^{>k+1}) \right]$, and an element is chosen where the error of the current interpolation is maximum in modulus. Then the residual for the row or column (for left and right half–sweep, resp.) which contains this element is evaluated, and the pivot $i^\star$ is chosen among its entries. If pivot is not zero up to the machine precision, the obtained cross is added to interpolation sets $\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}$. If pivot is machine null, the rank r_k is not increased.

The interpolation sets are always initialised by the index $(1, 1, \ldots, 1)$.
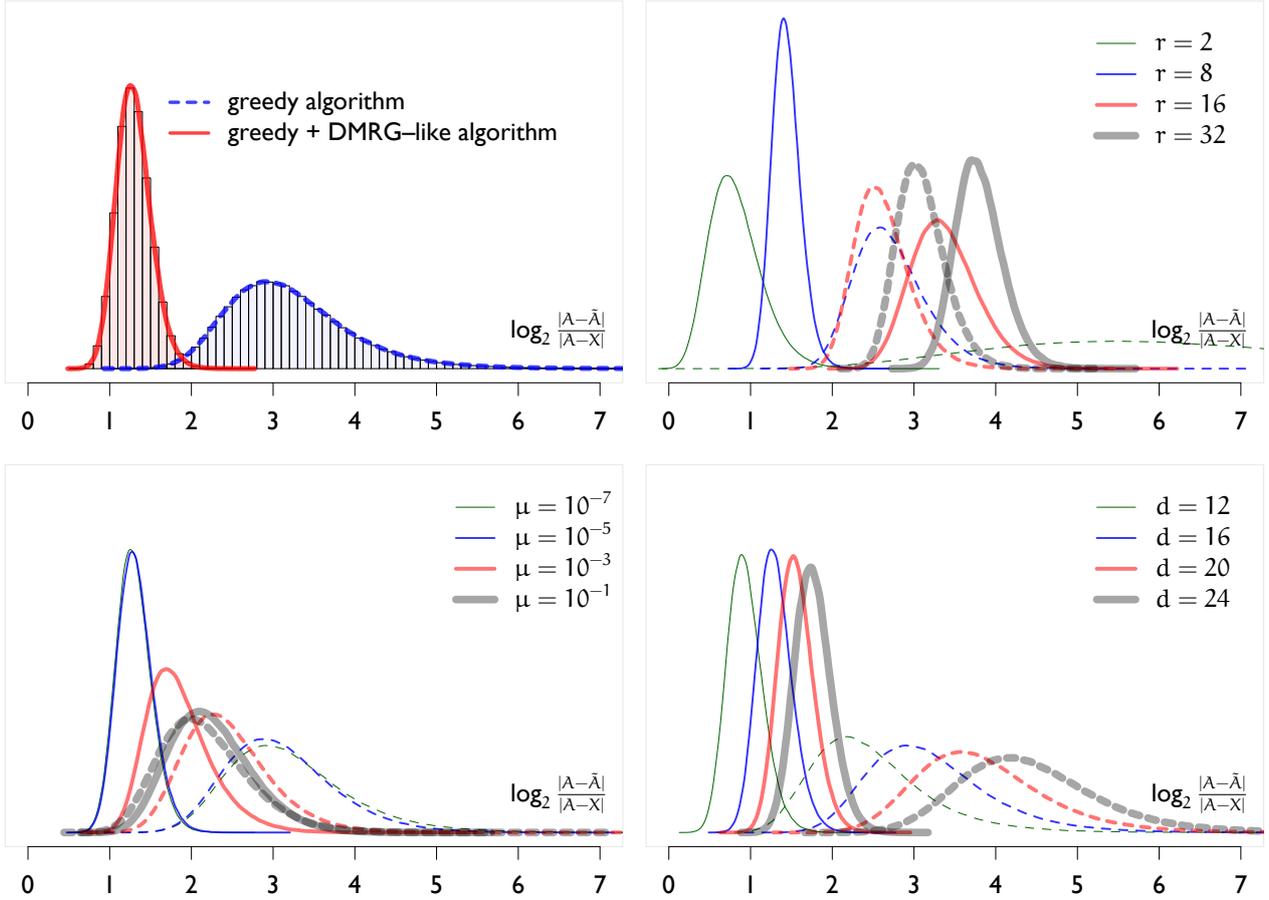
Figure 4: Distribution of $\log_2(|A - \tilde{A}|/|A - X|)$ for randomly generated tensors A given by (19) and the cross interpolation $\tilde{A}$, computed by Alg. 2 (dashed lines), and additionally improved by the TT-RC algorithm from [59] (solid lines). Dimension $d = 16$, mode size $n = 2$, noise level $\mu = 10^{-7}$, rank $r = 5$, unless other values are shown on the graph.

7.1. The quasioptimality coefficient

For a number of randomised experiments we measure the ratio between the accuracy of the approximation in the TT format (6) and the cross interpolation (7) with the same TT–ranks. Given dimension d, mode size $n = 2$, mode ranks r and *noise level* μ, we consider the tensor

$$A = X + \mu R, \qquad |X| = 1, \quad |R| = 1, \tag{19}$$

where R is random and X is given by the TT format (6) with TT–ranks r and random TT–cores. All random elements are independently and uniformly distributed on the unit set and we seed them using the internal pseudorandom generator provided with the compiler.

We apply Alg. 2 to compute the initial cross interpolation $\tilde{A}_{\mathrm{greedy}}$ with TT–ranks not larger than r. Then we run 10 additional sweeps of the DMRG–like TT–RC algorithm [59] to improve the positions of the interpolation crosses and obtain $\tilde{A}_{\mathrm{DMRG}}$. Density distributions of the logarithm of the quasioptimality coefficient for $\tilde{A}_{\mathrm{greedy}}$ and $\tilde{A}_{\mathrm{DMRG}}$ are shown on Fig. 4. The number of tests for each density distribution curve is at least 2^{20}.

We note that for the randomly generated tensors, the quasioptimality coefficient is not very large. For example, the top left graph on Fig. 4 corresponds to $d = 16$ and $r = 5$. The estimate of

---

[5]Iridis4 is based on Intel 2.6 GHz processors, for more specifications see cmg.soton.ac.uk/iridis.

Thm. 1 provides the upper bound for the quasioptimality coefficient $(2r + \kappa r + 1)^{\lceil \log_2 d \rceil}(r+1)^2 \geqslant 16^4 6^2 \geqslant 2^{21}$. The computed value is

$$\log_2(|A - \tilde{A}_{\text{greedy}}|/|A - X|) = 3.2 \pm 2.1, \qquad \log_2(|A - \tilde{A}_{\text{DMRG}}|/|A - X|) = 1.3 \pm 0.6.$$

Therefore, for the considered experiment the upper bound 2^{21} provided by Thm. 1 overestimates the actual value by a factor $\geqslant 2^{19.5}$.

It is important to check how the accuracy of the interpolation depends on the dimension d and the TT–ranks r. The results of these experiments are shown in the right column of Fig. 4. We see that the coefficient grows with rank and dimension slower than the upper bound (10). For example, for $r = 32$ the upper bound is $\gtrsim 2^{58}$, assuming $\kappa = 1$. The actual coefficient computed in the numerical experiments is of the order 2^3 for $\tilde{A}_{\text{greedy}}$ and 2^4 for $\tilde{A}_{\text{DMRG}}$. Note that in this case the interpolation improved by the DMRG–like algorithm has worse accuracy than the interpolation returned by Alg. 2. This may be explained by the fact that the TT–RC algorithm has the truncation step which reduces TT–ranks and introduces a perturbation to the tensor. The double–side nestedness (16) is not preserved during this step which may result in the loss of the interpolation property and deteriorate the accuracy. This emphasises the importance of the interpolation property given by Thm. 4.

Finally, we analyse how the accuracy of the cross interpolation depends on the noise level μ. On the bottom left graph on Fig. 4 we see that this parameter does not change the distribution significantly. When $\mu \leqslant 10^{-5}$, further reduction of the noise level has no effect on the distribution of the quasioptimality coefficient.

We summarise that for random tensors the accuracy of the computed cross interpolation behaves much better than the upper bound in (10).

7.2. Speed and accuracy of the greedy interpolation algorithm

We apply Alg. 2 to the tensor $A = [A(i_1, \ldots, i_d)]$ with elements

$$A(i_1, \ldots, i_d) = 1/\sqrt{i_1^2 + \ldots + i_d^2}. \tag{20}$$

This example is the standard test considered in e.g. [51, 54, 4]. We test the algorithm for large mode sizes n and dimensions d, where the evaluation of the accuracy $|A - \tilde{A}|$ is impossible due to the restrictively large number of entries. We substitute the exact evaluation by estimates computed on a large number of randomly distributed elements as follows

$$|A|_\sim = \max_{i \in \mathcal{I}} |A(i_1, \ldots, i_d)|, \qquad \|A\|_\sim^2 = \frac{n_1 \cdots n_d}{\#\mathcal{I}} \sum_{i \in \mathcal{I}} |A(i_1, \ldots, i_d)|^2, \tag{21}$$

where indices $i = (i_1, \ldots, i_d) \in \mathcal{I}$ are chosen randomly, and $\#\mathcal{I}$ denotes the number of elements in the random set $\mathcal{I}$. In our tests $\#\mathcal{I} \geqslant 2^{30}$.

The relative accuracy of the interpolation computed by Alg. 2 is shown on Fig. 5. We note that for significantly large ranks it reduces almost to the machine precision threshold and does not stagnate near the square root of machine precision, cf. [54, 4].

The runtime of the Alg. 2 is shown on Fig. 6. It is not difficult to notice the linear scaling w.r.t. the mode size n. The scaling in dimension is between $\mathcal{O}(d)$ and $\mathcal{O}(d^2)$, since the algorithm requires $\mathcal{O}(d)$ evaluations of tensor elements, and each tensor element depends on d indices. The scaling w.r.t. TT–rank is quadratic, which shows that the evaluation of tensor elements takes longer than other operations.

Finally, we compare the performance of Alg. 2 with the HT cross interpolation algorithm [4] applied to the same problem, see Table 1. One of the largest problems reported in [4] is the one
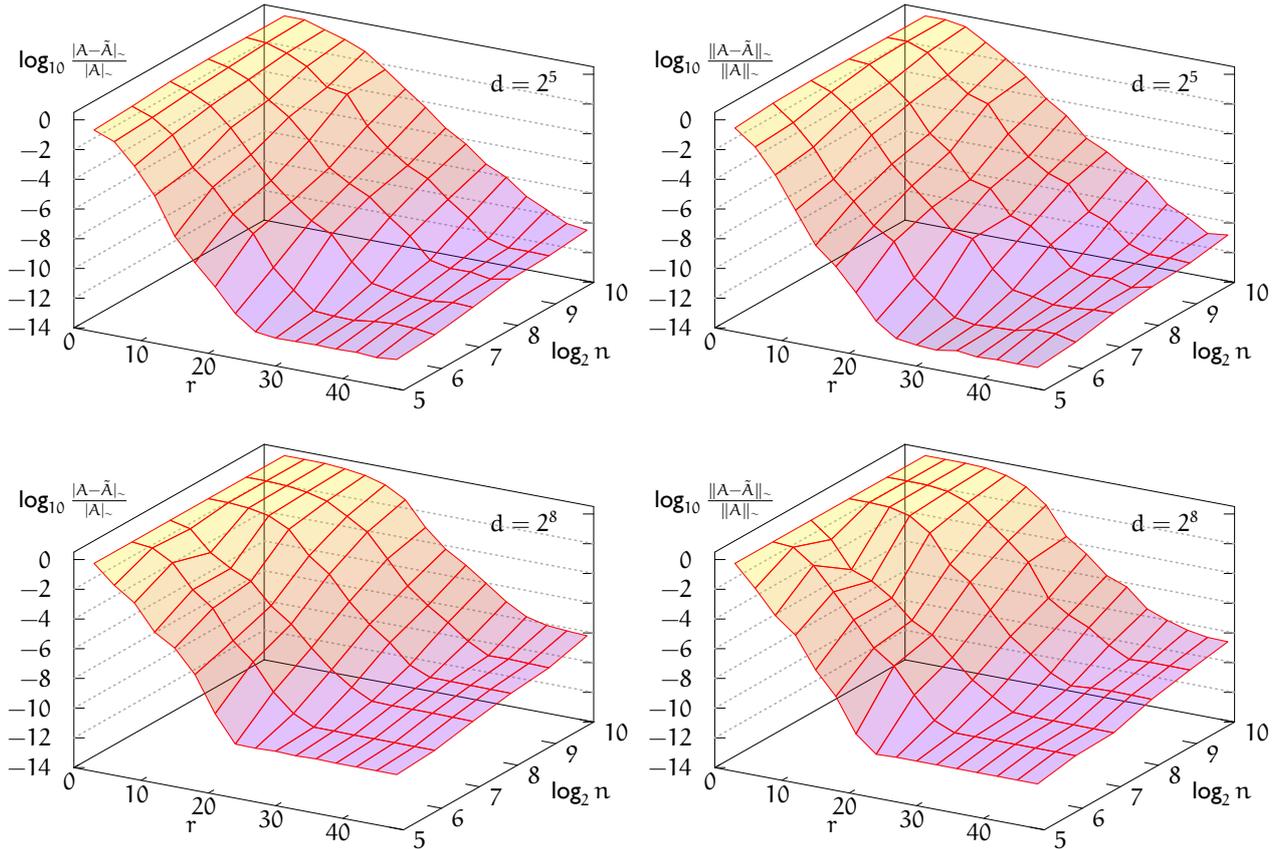
Figure 5: Accuracy of the approximation $\tilde{A}$ delivered by Alg. 2 for tensor (20) with dimension d, mode size n and TT–ranks r. The graphs show the estimates (21) of the relative error in the Chebyshev norm (left) and in the Frobenius norm (right).

with $d = 32$, $n = 32$ and $r = 27$, where the relative accuracy 10^{-12} is reached. The computation in HT format requires 5.17 seconds; the TT cross interpolation is constructed by Alg. 2 in 0.50 seconds, and is of the same accuracy. Therefore, Alg. 2 is ten times faster than the algorithm from [4].

We also consider a larger problem with $d = 1024$, $n = 1024$ and $r = 50$. The accuracy of the approximation computed by Alg. 2 is of the order 10^{-8}, and the computation takes 7 hours and 10 minutes on 1 core of Iridis4. This can be used as a milestone for comparison with the future algorithms.

8. Conclusions and future work

We have generalised two results on the matrix cross interpolation to the tensor case, using the cross interpolation formula (7) proposed by Oseledets and Tyrtyshnikov [54] for the tensor train format. First, we have shown (Thm. 1) that the maximum–volume cross interpolation is quasioptimal, i.e. its accuracy in the Chebyshev norm differs from the best possible accuracy by the factor which does not grow exponentially with dimension. This extends the matrix result of Goreinov and Tyrtyshnikov [22]. Second, we have shown (Thm. 4) that for the nested interpolation indices formula (7) computes $\sum_{k=1}^{d} r_{k-1} n_k r_k - \sum_{k=1}^{d-1} r_k^2$ parameters of the TT format inspecting exactly the same number of tensor entries, and on these elements the interpolation is exact. This generalises the classical result on the skeleton approximation of matrices to the TT case.

18
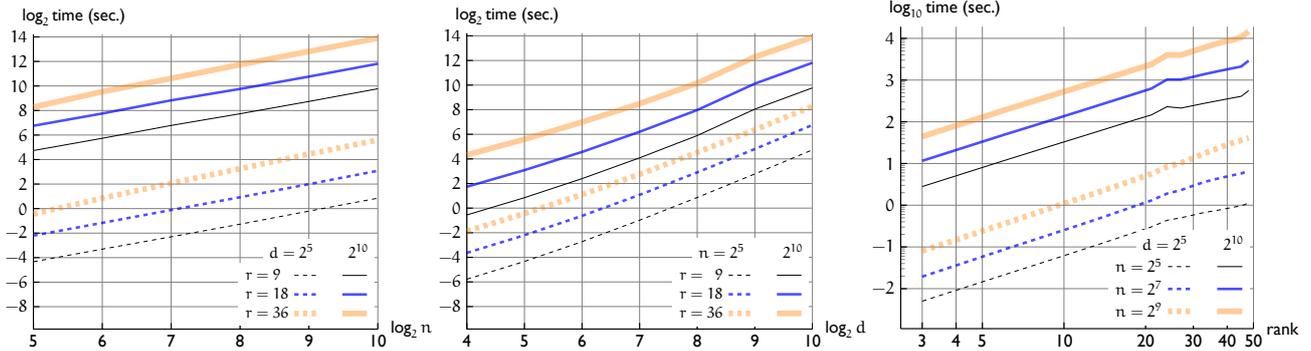
Figure 6: Computation time of Alg. 2 applied to the tensor (20) with dimension d, mode size n and TT–ranks r.

			TT		HT	
d	n	r	acc	time	acc	time
32	32	27	10^{-12}	0.50s	10^{-12}	5.17s
1024	1024	50	10^{-8}	7.16h	—	

Table 1: Accuracy $|A - \tilde{A}|_\sim / |A|_\sim$ and runtime of Alg. 2 (TT) and the HT cross interpolation algorithm [4] (HT) for tensor (20).

In the tensor case, the maximum–volume interpolation sets in general are not nested, and currently we are not able to suggest the choice of sets which provide quasioptimality and the interpolation property simultaneously. It would be interesting to find the nested interpolation sets which provide a moderate coefficient of the quasioptimality.

Using the interpolation property, we have proposed the fast and simple greedy cross interpolation algorithm, which provides very accurate results for the standard test, and is several times faster than other methods. Many variants of this algorithm can be developed, taking in account the interpolation property and the available information on the error of the interpolation for different entries of a tensor. It is easy to overcome the breakdowns, if they occur, simply by taking random pivots in larger subtensors or in the whole tensor, as is suggested in Alg. 1. In our experiments we have never had a breakdown using the restricted pivoting in Alg. 2.

The theoretical and experimental results of this paper show that the curse of dimensionality cannot stop us from developing fast and reliable cross interpolation methods in higher dimensions. The cross interpolation allows to convert a given high–dimensional data array into the tensor train format, for which many operations essential for the scientific computing are already possible. For many high–dimensional problems we can try to substitute the randomised (Monte Carlo) sampling by the cross interpolation in order to benefit from its adaptivity. This is a subject of further work.

Software

The implementation of Alg. 2 made by the author is available at:

- `github.com/savostyanov/ttcross`.

The Matlab implementation of Alg. 2 is made by Sergey Dolgov as:

- `cross/greedy2_cross.m` in `github.com/oseledets/TT-Toolbox`.

19

Acknowledgements

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. Meth. Biomed. Engng., 28 (2012), pp. 960–973.

[2] A. AMMAR, B. MOKDAD, F. CHINESTA, AND R. KEUNINGS, *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids*, Journal of Non-Newtonian Fluid Mechanics, 139 (2006), pp. 153 – 176.

[3] J. BALLANI AND L. GRASEDYCK, *A projection method to solve linear systems in tensor format*, Numerical Linear Algebra with Applications, 20 (2013), pp. 27–43.

[4] J. BALLANI, L. GRASEDYCK, AND M. KLUGE, *Black box approximation of tensors in hierarchical Tucker format*, Linear Alg. Appl., 428 (2013), pp. 639–657.

[5] A. BARTH, C. SCHWAB, AND N. ZOLLINGER, *Multi-level Monte Carlo finite element method for elliptic PDEs with stochastic coefficients*, Numerische Mathematik, 119 (2011), pp. 123–161.

[6] J. J. BARTHOLDI, *A good submatrix is hard to find*, School of industrial and systems engineering, Georgia Institute of technology, 1982.

[7] M. BEBENDORF, *Approximation of boundary element matrices*, Numer. Mathem., 86 (2000), pp. 565–589.

[8] P. BINEV, A. COHEN, W. DAHMEN, R. DEVORE, G. PETROVA, AND P. WOJTASZCZYK, *Convergence rates for greedy algorithms in reduced basis methods*, SIAM J. Math. Anal., 43 (2011), pp. 1457–1472.

[9] H.-J. BUNGATRZ AND M. GRIEBEL, *Sparse grids*, Acta Numerica, 13 (2004), pp. 147–269.

[10] E. CANCÉS, V. EHRLACHER, AND T. LELIÉVRE, *Convergence of a greedy algorithm for high-dimensional convex nonlinear problems*, Mathematical Models and Methods in Applied Sciences, 21 (2011), pp. 2433–2467.

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

[12] 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*, Computer Phys. Comm., (2014).

[13] 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.

[14] 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.

[15] S. V. Dolgov and D. V. Savostyanov, *Alternating minimal energy methods for linear systems in higher dimensions. Part I: SPD systems*, arXiv preprint 1301.6068, 2013.

[16] ——, *Alternating minimal energy methods for linear systems in higher dimensions. Part II: Faster algorithm and application to nonsymmetric systems*, arXiv preprint 1304.1222, 2013.

[17] ——, *One-site density matrix renormalization group and alternating minimum energy algorithm*, arXiv preprint 1312.6542, 2013.

[18] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition*, SIAM J Comput, 36 (2006), pp. 184–206.

[19] M. Fannes, B. Nachtergaele, and R. Werner, *Finitely correlated states on quantum spin chains*, Communications in Mathematical Physics, 144 (1992), pp. 443–490.

[20] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *How to find a good submatrix*, in Matrix Methods: Theory, Algorithms, Applications, V. Olshevsky and E. Tyrtyshnikov, eds., World Scientific, Hackensack, NY, 2010, pp. 247–256.

[21] S. A. Goreinov and E. E. Tyrtyshnikov, *The maximal-volume concept in approximation by low-rank matrices*, Contemporary Mathematics, 208 (2001), pp. 47–51.

[22] ——, *Quasioptimality of skeleton approximation of a matrix in the Chebyshev norm*, Doklady Math., 83 (2011), pp. 374–375.

[23] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *Pseudo–skeleton approximations of matrices*, Reports of Russian Academy of Sciences, 342 (1995), pp. 151–152.

[24] ——, *A theory of pseudo–skeleton approximations*, Linear Algebra Appl., 261 (1997), pp. 1–21.

[25] S. A. Goreinov, N. L. Zamarashkin, and E. E. Tyrtyshnikov, *Pseudo–skeleton approximations by matrices of maximum volume*, Mathematical Notes, 62 (1997), pp. 515–519.

[26] L. Grasedyck, *Hierarchical singular value decomposition of tensors*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2029–2054.

[27] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, arXiv preprint 1302.7121, 2013.

[28] W. Hackbusch, *Tensor spaces and numerical tensor calculus*, Springer–Verlag, Berlin, 2012.

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

[30] W. K. Hastings, *Monte Carlo sampling methods using Markov chains and their applications*, Biometrika, 57 (1970), pp. 97–109.

[31] 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.

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

[33] E. Jeckelmann, *Dynamical density–matrix renormalization–group method*, Phys Rev B, 66 (2002), p. 045114.

[34] V. Kazeev, B. Khoromskij, and E. Tyrtyshnikov, *Multilevel Toeplitz matrices generated by tensor-structured vectors and convolution with logarithmic complexity*, SIAM J. Sci. Comp., 35 (2013), pp. A1511–A1536.

[35] B. Khoromskij and C. Schwab, *Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs*, SIAM J. of Sci. Comp., 33 (2011), pp. 1–25.

[36] B. N. Khoromskij, $\mathcal{O}(d \log n)$–*Quantics approximation of N–d tensors in high-dimensional numerical modeling*, Constr. Appr., 34 (2011), pp. 257–280.

[37] ——, *Tensor-structured numerical methods in scientific computing: Survey on recent advances*, Chemometr. Intell. Lab. Syst., 110 (2012), pp. 1–19.

[38] B. N. Khoromskij and I. V. Oseledets, *Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs*, Comput. Meth. Appl. Math., 10 (2010), pp. 376–394.

[39] A. Klümper, A. Schadschneider, and J. Zittartz, *Matrix product ground states for one-dimensional spin-1 quantum antiferromagnets*, Europhys. Lett., 24 (1993), pp. 293–297.

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

[41] 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.

[42] ——, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 273–290.

[43] ——, *Preconditioned low-rank methods for high-dimensional elliptic PDE eigenvalue problems*, Computational Methods in Applied Mathematics, 11 (2011), pp. 363–381.

[44] S. Kühn, *Hierarchische Tensordarstellung*, PhD thesis, Uni. Leipzig, Faculty of Mathematics and Informatics, 2012.

[45] C. Le Bris, T. Leliévre, and Y. Maday, *Results and questions on a nonlinear approximation approach for solving high-dimensional partial differential equations*, Constr. Approx., 30 (2009), pp. 621–651.

[46] O. S. Lebedeva, *Tensor conjugate-gradient-type method for Rayleigh quotient minimization in block QTT-format*, Russ. J. Numer. Anal. Math. Modelling, 26 (2011), p. 465–489.

[47] M. W. Mahoney, M. Maggioni, and P. Drineas, *Tensor-CUR decompositions for tensor-based data*, SIAM J. Matr. Anal. Appl., 30 (2008), pp. 957–987.

[48] I. V. Oseledets, *DMRG approach to fast linear algebra in the TT–format*, Comput. Meth. Appl. Math., 11 (2011), pp. 382–393.

[49] ——, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[50] ——, *Constructive representation of functions in low-rank tensor formats*, Constr. Appr., 37 (2013), pp. 1–18.

[51] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov, *Tucker dimensionality reduction of three-dimensional arrays in linear time*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 939–956.

[52] I. V. Oseledets, D. V. Savostyanov, and E. E. Tyrtyshnikov, *Cross approximation in tensor electron density computations*, Numer. Linear Algebra Appl., 17 (2010), pp. 935–952.

[53] 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.

[54] ——, *TT-cross approximation for multidimensional arrays*, Linear Algebra Appl., 432 (2010), pp. 70–88.

[55] S. Östlund and S. Rommer, *Thermodynamic limit of density matrix renormalization*, Phys. Rev. Lett., 75 (1995), pp. 3537–3540.

[56] T. A. Porsching, *Estimation of the error in the reduced basis method solution of nonlinear equations*, Math. Comp., 45 (1985), pp. 487–496.

[57] T. Rohwedder and A. Uschmajew, *Local convergence of alternating schemes for optimization of convex problems in the TT format*, SIAM J. Num. Anal., 51 (2013), pp. 1134–1162.

[58] D. V. Savostyanov, *Polilinear approximation of matrices and integral equations*, PhD thesis, INM RAS, Moscow, 2006. (in Russian).

[59] D. V. Savostyanov and I. V. Oseledets, *Fast adaptive interpolation of multidimensional arrays in tensor train format*, in Proceedings of 7th International Workshop on Multidimensional Systems (nDS), IEEE, 2011.

[60] J. Schneider, *Error estimates for two–dimensional cross approximation*, J. Approx. Theory, 162 (2010), pp. 1685–1700.

[61] U. Schollwöck, *The density–matrix renormalization group*, Rev. Mod. Phys., 77 (2005), pp. 259–315.

[62] S. A. Smolyak, *Quadrature and interpolation formulas for tensor products of certain class of functions*, Dokl. Akad. Nauk SSSR, 148 (1964), pp. 1042–1053. Transl.: Soviet Math. Dokl. 4:240-243, 1963.

[63] V. Temlyakov, *Greedy Approximation*, Cambridge University Press, 2011.

[64] E. E. Tyrtyshnikov, *Incomplete cross approximation in the mosaic–skeleton method*, Computing, 64 (2000), pp. 367–380.

[65] ——, *Tensor approximations of matrices generated by asymptotically smooth functions*, Sbornik: Mathematics, 194 (2003), pp. 941–954.

[66] ——, *Kronecker-product approximations for some function-related matrices*, Linear Algebra Appl., 379 (2004), pp. 423–437.

[67] S. R. White, *Density matrix formulation for quantum renormalization groups*, Phys. Rev. Lett., 69 (1992), pp. 2863–2866.

[68] ——, *Density-matrix algorithms for quantum renormalization groups*, Phys. Rev. B, 48 (1993), pp. 10345–10356.