Journal of Computational Physics 281 (2015) 116–134

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# Randomized interpolative decomposition of separated representations ☆

David J. Biagioni [a], Daniel Beylkin [b], Gregory Beylkin [c,*]

[a] Computational Science Center, National Renewable Energy Laboratory, 15013 Denver W. Pkwy, Golden, CO 80401, United States
[b] Program in Applied Mathematics, Yale University, 51 Prospect St., New Haven, CT 06511, United States
[c] Department of Applied Mathematics, University of Colorado at Boulder, 526 UCB, Boulder, CO 80309-0526, United States

ARTICLE INFO

*Article history:*
Received 16 December 2013
Received in revised form 16 September 2014
Accepted 4 October 2014
Available online 13 October 2014

*Keywords:*
Canonical tensor decomposition
Interpolative decomposition
Alternating least squares algorithm
Randomized projection
Self-guiding tensor iteration

ABSTRACT

We introduce an algorithm to compute tensor interpolative decomposition (dubbed CTD-ID) for the reduction of the separation rank of Canonical Tensor Decompositions (CTDs). Tensor ID selects, for a user-defined accuracy $\epsilon$, a near optimal subset of terms of a CTD to represent the remaining terms via a linear combination of the selected terms. CTD-ID can be used as an alternative to or in combination with the Alternating Least Squares (ALS) algorithm. We present examples of its use within a convergent iteration to compute inverse operators in high dimensions. We also briefly discuss the spectral norm as a computational alternative to the Frobenius norm in estimating approximation errors of tensor ID.

We reduce the problem of finding tensor IDs to that of constructing interpolative decompositions of certain matrices. These matrices are generated via randomized projection of the terms of the given tensor. We provide cost estimates and several examples of the new approach to the reduction of separation rank.



## 1. Introduction

The computational cost of many fast algorithms grows exponentially in the problem dimension, $d$. In order to maintain linear complexity in $d$, we use separated representations as a framework for numerical computations in high dimensions [13, 14]. Separated representations generalize the usual notion of separation of variables by representing a multivariate function as

$$u(x_1, x_2, \dots x_d) = \sum_{l=1}^{r} \sigma_l u_1^{(l)}(x_1) u_2^{(l)}(x_2) \cdots u_d^{(l)}(x_d), \tag{1.1}$$

where the number of terms, $r$, is called the separation rank of the function $u$. Any discretization of (1.1) with $u_{i_j}^{(l)} = u_j^{(l)}(x_{i_j})$, $i_j = 1, \dots M_j$ and $j = 1, \dots d$, leads to a Canonical Tensor Decomposition (CTD),

$$\mathcal{U}_{i_1 \dots i_d} = \sum_{l=1}^{r} \sigma_l u_{i_1}^{(l)} u_{i_2}^{(l)} \cdots u_{i_d}^{(l)}. \tag{1.2}$$

☆ D.J. Biagioni was partially supported by NREL grant UGA-0-41026-08 and NSF grant DMS-1228359. D. Beylkin received support under FA9550-11-C-0028 by the DoD AFOSR NDSEG Fellowship. G. Beylkin was partially supported by NSF grants DMS-1009951, DMS-1228359 and DOE/ORNL grant 4000038129.
* Corresponding author.

http://dx.doi.org/10.1016/j.jcp.2014.10.009
0021-9991/

The functions $u_j^{(l)}(x_j)$ in (1.1) and the corresponding vectors $u_{i_j}^{(l)}$ in (1.2) are normalized to have unit Frobenius norm so that the size of the terms is carried by their positive $s$-values, $\sigma_l$.

Numerical computations using such representations require an algorithm to reduce the number of terms for a given accuracy, $\epsilon$. Such reduction can be achieved via Alternating Least Squares (ALS) (see, e.g., [35,20,17,13,14,56,39]). Specifically, given a CTD of rank $r$, ALS allows us to find a representation of the same form but with fewer terms,

$$\tilde{\mathcal{U}}_{i_1\dots i_d} = \sum_{l=1}^{k} \tilde{\sigma}_l \tilde{u}_{i_1}^{(l)} \tilde{u}_{i_2}^{(l)} \cdots \tilde{u}_{i_d}^{(l)}, \quad k < r, \tag{1.3}$$

so that $\|\mathcal{U} - \tilde{\mathcal{U}}\| \le \epsilon \|\mathcal{U}\|$, where $\epsilon$ is a user-selected accuracy. Standard operations on separated representations of rank $k$, such as multiplication, may result in a large number, e.g., $\mathcal{O}(k^2)$, of intermediate terms. If the intermediate separation rank $r = \mathcal{O}(k^2)$ is reducible to $\mathcal{O}(k)$, then the cost of ALS can be estimated as $\mathcal{O}(d \cdot k^4 \cdot M) \cdot (\textit{number of iterations})$, where we assumed that $M_j = M$, $j = 1, \dots, d$. Noting that the number of iterations is not easily controlled and may be large, the computational cost of ALS can be significant, although it is linear in the dimension $d$.

In this paper we describe randomized tensor Interpolative Decomposition (ID) for the reduction of the separation rank of CTD (1.2) that is faster (for a class of problems) than ALS by $\mathcal{O}(k) \cdot (\textit{number of iterations})$. We adopt the term tensor CTD-ID (by analogy with matrix ID, see e.g. [34]) and to emphasize that our approach is limited to CTD (in the sequel we may abbreviate this to just tensor ID). We note that the reduction of the number of linearly dependent terms in the separated representations has been already performed using Gram matrices.[1] The reason we revisit this type of approach is that the use of the Gram matrix limits the accuracy of computations to about one half of the available digits (e.g., single precision while using double precision arithmetic) and may be problematic due to the dynamic range of its entries if used for sufficiently high dimensions. The randomized approach allows us to avoid these issues, at least for moderate dimensions. While we tested the randomized algorithm on a number of representative examples, the full justification of our approach needs additional work. Currently we verify accuracy of the result *a posteriori*. The randomized tensor ID algorithm can be supplemented by a fixed number of ALS-type iterations which play an auxiliary role of improving the conditioning of CTD. We demonstrate a significant acceleration of the reduction of separation rank on a number of examples.

On the technical level, we extend recently developed ideas of randomized algorithms for matrix ID [21,41,45,34] to a tensor setting. These algorithms use the fact that the projection of columns of a low rank matrix onto a sufficient number of random vectors provides, with high probability, a good approximation of the matrix range. The number of random vectors needed is only slightly greater than the rank of the matrix which, in turn, implies that the cost of computing the matrix ID can be greatly reduced. We extend this randomized framework to CTD by posing the problem as that of computing the matrix ID of a certain associated matrix. We first present the main ideas without details or proofs and lay out technical information in the sections that follow.

Our interest in the development of CTD-ID stems from applications where we compute functions of operators in high dimensions. In particular, we are interested in computing Green's functions (see examples in Section 5). The inverse operator (the Green's function) can be computed via self-correcting, quadratically convergent Schulz iteration [55]. Many other important functions of operators can also be obtained via self-correcting, convergent iterations. In such cases we can use the iteration itself to generate the necessary variety of terms in the separated representation and use the randomized tensor ID as a way to select the desired subset of terms while maintaining accuracy. The intermediate errors incurred by such reduction are then corrected by the iteration itself. We dub such an approach Self-Guiding Tensor Iteration (SGTI) and provide examples of computing Green's functions using it. We plan to address a general problem of computing functions of operators in high dimensions via the SGTI approach separately.

Finally, we have observed that, in the CTD setting, a randomized approach based on sampling may provide an additional speed-up in reducing the separation rank. We plan to address such an approach separately.

### 1.1. Definitions and notation

Throughout the paper, CTDs are denoted by the calligraphic letters $\mathcal{Q}$ through $\mathcal{Z}$. We assume that each direction $j = 1, \dots, d$ may be represented by $M_j$ values, stored in the vectors $\mathbf{u}_j$. Thus, $\mathcal{U} \in \mathbb{V} = \bigotimes_{j=1}^{d} \mathbb{R}^{M_j}$ and, using the Kronecker product notation, can also be written as

$$\mathcal{U} = \sum_{l=1}^{r} \sigma_l \bigotimes_{j=1}^{d} \mathbf{u}_j^{(l)} \tag{1.4}$$

instead of (1.2). It may sometimes be convenient to emphasize that $\mathcal{U}$ is a sum of rank-one tensors, so that

[1] Martin Mohlenkamp (Ohio University) and G.B. developed and used a deterministic algorithm for tensor ID based on pivoted Cholesky decomposition of the Gram matrix (see, e.g., discussion around Theorem 3.5).

$$\mathcal{U} = \sum_{l=1}^{r} \sigma_l \mathcal{U}^{(l)}, \quad \text{where} \quad \mathcal{U}^{(l)} = \bigotimes_{j=1}^{d} \mathbf{u}_j^{(l)}. \tag{1.5}$$

The standard Frobenius inner product between any two tensors $\mathcal{U}$ and $\mathcal{V}$ is defined as

$$\langle \mathcal{U}, \mathcal{V} \rangle = \sum_{i_1=1}^{M_1} \cdots \sum_{i_d=1}^{M_d} \mathcal{U}_{i_1 \ldots i_d} \mathcal{V}_{i_1 \ldots i_d} \tag{1.6}$$

which for CTDs reduces to

$$\langle \mathcal{U}, \mathcal{V} \rangle = \sum_{l=1}^{r_u} \sum_{m=1}^{r_v} \sigma_l^u \sigma_m^v \langle \mathcal{U}^{(l)}, \mathcal{V}^{(m)} \rangle = \sum_{l=1}^{r_u} \sum_{m=1}^{r_v} \sigma_l^u \sigma_m^v \prod_{j=1}^{d} \langle \mathbf{u}_j^{(l)}, \mathbf{v}_j^{(m)} \rangle, \tag{1.7}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product between component vectors. The Frobenius norm is then defined as

$$\|\mathcal{U}\|_F = \sqrt{\langle \mathcal{U}, \mathcal{U} \rangle}. \tag{1.8}$$

**Remark 1.1.** The directional vectors $\mathbf{u}_j^{(l)}$ may represent objects of different types, including proper one dimensional vectors, matrices or even low dimensional tensors. The vectors $\mathbf{u}_j^{(l)}$ can have a complicated structure (e.g., sparse matrices or low dimensional tensors) as long as the Frobenius inner product between them is well defined.

### *1.2. CTD interpolative decomposition*

The tensor ID is motivated by the fact that certain tensor operations, e.g. multiplication, can result in CTDs with many nearly linearly dependent terms. In such case, we formulate the problem of separated rank reduction as that of identifying a near optimal (in a sense to be explained later) subset $\{\mathcal{U}^{(l_m)}\}_{m=1}^{k}$, $k < r$, of linearly independent terms of $\mathcal{U}$ in (1.4) to represent the remaining terms. In contrast to (1.3), we seek a CTD

$$\mathcal{U}_k = \sum_{m=1}^{k} \widehat{\sigma}_m \mathcal{U}^{(l_m)}, \quad k < r, \tag{1.9}$$

with modified $s$-values $\widehat{\sigma}_m$, so that $\|\mathcal{U} - \mathcal{U}_k\| \leq \epsilon \|\mathcal{U}\|$.

The tensor ID extends the concept of the matrix ID introduced and developed in [33,59,31,30,21,41,45,34]. For an $m \times n$ matrix $A$ and desired accuracy $\epsilon$, the construction of the matrix ID entails identifying a set of columns with indices $\mathcal{L}_k = \{l_1, l_2, \ldots, l_k\} \subseteq \{1, 2, \ldots, n\}$ and $k \times n$ (well conditioned) coefficient matrix $P$, such that

$$\|A - A_c P\|_2 < \epsilon \|A\|_2, \tag{1.10}$$

where $A_c$ is the so-called column skeleton of $A$, i.e., a matrix containing the columns with indices $\mathcal{L}_k$.

We associate the problem of selecting linearly independent terms of the tensor $\mathcal{U}$ to that of selecting linearly independent columns of an appropriately chosen matrix. For conceptual purposes, let us introduce the $N \times r$ matrix

$$U = \begin{bmatrix} | & | & & | \\ \sigma_1 \mathcal{U}^{(1)} & \sigma_2 \mathcal{U}^{(2)} & \cdots & \sigma_r \mathcal{U}^{(r)} \\ | & | & & | \end{bmatrix}, \tag{1.11}$$

by treating the terms of $\mathcal{U}$ as vectors in $\mathbb{R}^N$, where $N = \prod_{j=1}^{d} M_j$ is gigantic. Forming such a matrix is in *no way practical* for most (if not all) problems of interest. However, by efficiently constructing the matrix ID of $U$ (without using this matrix explicitly), we show in Section 3 that it is possible to construct a rank-$k$ approximation to the tensor $\mathcal{U}$ as in (1.9).

In Section 3.2, we start by showing how to use the $r \times r$ Gram matrix (corresponding to $U^*U$)

$$G = \begin{bmatrix} \langle \sigma_1 \mathcal{U}^{(1)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \sigma_1 \mathcal{U}^{(1)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \sigma_1 \mathcal{U}^{(1)}, \sigma_r \mathcal{U}^{(r)} \rangle \\ \langle \sigma_2 \mathcal{U}^{(2)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \sigma_2 \mathcal{U}^{(2)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \sigma_2 \mathcal{U}^{(2)}, \sigma_r \mathcal{U}^{(r)} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \sigma_r \mathcal{U}^{(r)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \sigma_r \mathcal{U}^{(r)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \sigma_r \mathcal{U}^{(r)}, \sigma_r \mathcal{U}^{(r)} \rangle \end{bmatrix} \tag{1.12}$$

to compute the tensor ID of $\mathcal{U}$. For this purpose, we use a symmetric rank-$k$ ID of $G$,

$$G_k = P^* G_S P, \tag{1.13}$$

where $G_S$ is a $k \times k$ sub-matrix of $G$, and derive estimates for the resulting error in approximating the tensor $\mathcal{U}$.

**Remark 1.2.** Constructing the tensor ID via the Gram matrix limits the achievable accuracy $\epsilon$ to about half the digits of machine precision. To avoid the loss of accuracy, we develop a randomized algorithm for computing the tensor ID. Specifically (see Section 3), we form a collection of $\ell$ random tensors in separated form,

$$\mathcal{R}^{(l)} = \bigotimes_{j=1}^{d} \mathbf{r}_j^{(l)}, \quad l = 1, \ldots, \ell, \tag{1.14}$$

with independent random variables $r_{i_j}^{(l)}$ of zero mean and unit variance and $\ell$ somewhat larger than the expected separation rank $k$. We then form the $\ell \times r$ projection matrix

$$Y = \begin{bmatrix} \langle \mathcal{R}^{(1)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \mathcal{R}^{(1)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \mathcal{R}^{(1)}, \sigma_r \mathcal{U}^{(r)} \rangle \\ \langle \mathcal{R}^{(2)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \mathcal{R}^{(2)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \mathcal{R}^{(2)}, \sigma_r \mathcal{U}^{(r)} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathcal{R}^{(\ell)}, \sigma_1 \mathcal{U}^{(1)} \rangle & \langle \mathcal{R}^{(\ell)}, \sigma_2 \mathcal{U}^{(2)} \rangle & \cdots & \langle \mathcal{R}^{(\ell)}, \sigma_r \mathcal{U}^{(r)} \rangle \end{bmatrix}, \tag{1.15}$$

and use the matrix ID of $Y$ to construct a rank-$k$ approximation $\mathcal{U}_k$ of the tensor $\mathcal{U}$.

**Remark 1.3.** The reason higher accuracy is achievable with (1.15) than with (1.12) is that, ultimately, to find new coefficients, we have to solve a linear system using sub-matrices of these matrices. If in solving a linear system $Ax = b$, where matrix $A$ is poorly conditioned with condition number $\kappa(A)$, we were to form the normal system, $A^t Ax = b$, the resulting condition number is $\kappa^2(A)$. On the other hand, if instead we solve $RAx = Rb$, where $R$ is a rectangular random matrix, the condition number of the matrix $RA$ is $\kappa(RA) \le \kappa(R)\kappa(A)$. The condition number of random rectangular matrices grows relatively slowly with their size. For example, it is shown in [28, Theorem 7.3] that if $R$ is an $m \times n$ random matrix with independent and identically distributed entries from $N(0, 1)$, and the ratio $m/n$ (or $n/m$) approaches $y \in [0, 1]$ as $n, m \to \infty$, then

$$E(\log \kappa) = \log \frac{1 + \sqrt{y}}{1 - \sqrt{y}} + o(1).$$

While solving linear systems with ordinary matrices it is not necessary to modify the system using projections. However, because the size of matrices such as $U$ in (1.11) is intractably large, we need to do it for tensors in CTD form where the use of random projections leads to better conditioned systems.

While the theoretical underpinnings for this approach require further work, we have found that the method works well in many practical examples, some of which are presented in Section 5.

### *1.3. Contributions and relationship to prior work*

Separated representations of functions and operators were introduced in [13,14] (see also [12]). Since that time, their use as a framework for numerical operator calculus has appeared in a number of contexts, for example, in the construction of Green's functions, quadratures, nonlinear approximations, and solutions of stochastic differential equations (see [32] for a recent survey). The so-called Slater determinants used in quantum chemistry may be viewed as a form of separated representation (see [15] for details).

The Canonical Tensor Decomposition (CTD) has a long history and is also known as PARAFAC (PARAllel FACtor analysis) [35] or CANDECOMP (CANonical DECOMPosition) [20]. The CTD has been used extensively in various areas, including chemometrics, psychometrics, multivariate regression, and signal processing. For additional references, we refer to recent surveys [39,1,32].

While it is well known that CTD can be ill-conditioned (see, e.g., [24] or [51, pp. 132–133]), note that in a practical use of this tensor format one always seeks a finite precision and a near optimal separation rank. For this reason, potential ill-conditioning by itself does not limit practical applications.

Current practice for reducing the separation rank of a CTD is to use the Alternating Least Squares (ALS) algorithm or its variants (see, e.g., [35,20,17,13,14,56,39,47]). Although many problems may indeed be solved using this algorithm, this approach limits both the size of problems and the attainable accuracy. This paper provides an efficient alternative to ALS for applications where a self-correcting convergent iterative algorithm is used for solving numerical problems (cf. Section 5).

We introduce randomized methods into the CTD setting. The use of randomized methods for matrix problems has gained wide popularity within the last several years (see [34] for a recent survey). The theoretical foundation for many of these methods can be traced to the Johnson–Lindenstrauss Lemma [37], which guarantees so-called concentration of measure when vectors in high dimensions are projected into a lower dimensional space. The idea of using random projections in computational problems was proposed and developed in different areas over several years; see, e.g., [50,2,4,18,19]. The application of such an approach to low rank matrix approximation has been further developed in [44,54,41,45]. A variety of sampling methods for the same purpose have been proposed in [26,29,3,53].

We note that several alternative tensor formats, such as the Tucker decomposition, H-Tucker, Tensor Train (TT) have been proposed recently. For example, several recent papers introduced randomized methods to the tensor setting for the Tucker decomposition [22,43,27,57]. Skeleton-type decompositions have been obtained for TT and H-Tucker formats [49,6].

The CTD-ID described in Sections 1.2 and 3 extends the concept of the matrix ID developed in [59,31,30] and further extended in [33,21,44,41,45]. Our method for constructing tensor ID uses a randomized projection method with some similarities to approaches in [44,54,41,45].

Many of the standard arguments to justify the use of random projections of low rank matrices do not translate easily to the tensor setting as they lead to overly pessimistic estimates. For example, the usual arguments on concentration of measure for columns of the matrix $Y$ in (1.15) lead to a lower bound for the number of random projections that is exponential in the dimension $d$. Similar pessimistic estimates appears in the recent works [48,52]. Nevertheless, as we describe in Sections 3 and 5, the method we propose works significantly better than these estimates suggest. Further work on this topic is needed.

We begin by reviewing the necessary mathematical preliminaries in Section 2. We then describe our approach to the reduction of separation rank in Section 3. We set the conceptual framework for the tensor ID in Sections 3.1–3.2 and describe an efficient randomized algorithm for its computation in Section 3.3. We then address the loss of accuracy and the high cost of evaluating the Frobenius norm in Section 4, and discuss rank-one tensor approximation as an alternative method for norm estimation. Finally, we present several examples illustrating the new algorithm in Section 5.

## 2. Preliminaries

We start by summarizing several randomized algorithms for matrices that will be used in Section 3.

### 2.1. Randomized algorithm for approximating the range of a matrix

Let us define Q-factorization as a decomposition of an $m \times n$ matrix $A$ of fixed rank $k$ via

$$A_k = Q\,S \tag{2.1}$$

where $Q$ is an $m \times k$ matrix with orthonormal columns and $S$ is a $k \times n$ coefficient matrix (not necessarily upper-triangular). An efficient approach is to first construct the matrix $Q$ to approximate the range of $A$, and then set

$$S = Q^* A. \tag{2.2}$$

The matrix $A_k$ is then seen to be an orthogonal projection of the columns of $A$ onto the subspace captured by $Q$.

When $A$ is low rank, the construction of (2.1) is amenable to randomized methods. We use Algorithm 4.1 of [34] to construct $Q$, summarized in this paper as Algorithm 1 (see also [54,41,45]).

**Algorithm 1** Randomized algorithm for approximating the range of a matrix [34].

Input: An $m \times n$ matrix $A$ of fixed rank $k$, and integer $\ell > k$.
Output: An $m \times \ell$ matrix $Q$ whose columns comprise an orthonormal basis for the range of $A$.

1. Generate an $m \times \ell$ Gaussian random matrix, $R$.
2. Form the $m \times \ell$ matrix, $Y = AR$.
3. Construct QR factorization of $Y$, yielding the matrix $Q$.

This procedure yields a matrix $Q$ satisfying the following error estimate,

**Theorem 2.1** *(Halko, Martinsson, and Tropp). (See [34].) Select a target rank $k \geq 2$ and an integer $p \geq 2$, where $\ell = k + p \leq \min\{m, n\}$ and generate an $m \times \ell$ matrix $Q$ with orthonormal columns. Then we have*

$$\mathbb{E}\big[\big\|A - Q\,Q^*A\big\|_2\big] \leq \tau_{k+1}(A)\left(1 + \frac{4\sqrt{\ell}}{p-1}\sqrt{\min\{m,n\}}\right), \tag{2.3}$$

*where $\mathbb{E}$ denotes expectation with respect to the random matrix $R$ and $\tau_{k+1}(A)$ is the $k+1$ singular value of $A$.*

The probability of violating the estimate (2.3) decays exponentially in $p$,

$$\Pr\big\{\big\|A - Q\,Q^*A\big\| \geq \tau_{k+1}(A)\big(1 + 9\sqrt{\ell}\cdot\sqrt{\min\{m,n\}}\big)\big\} \leq 3\cdot p^{-p}, \tag{2.4}$$

implying that $p$ need not be large (e.g., $p = 5$ or $p = 10$). Thus, with very high probability, the randomized method comes close to constructing an optimal rank-$k$ approximation (the best possible bound in (2.3) is $\tau_{k+1}(A)$ which follows from the optimality of the SVD).

Let $c_A$ denote the cost of applying $A$ to a vector, and let $t_R$ denote the cost of generating a single random entry of $R$. Constructing $S$ costs $\ell \cdot c_A$ operations, leading to a total complexity on the order of

$$\underset{\text{Form } R}{\ell \cdot n \cdot t_R} + \underset{\text{Form } Y}{\ell \cdot c_A} + \underset{\text{QR of } Y}{\ell^2 \cdot m} + \underset{\text{Form } S}{\ell \cdot c_A}. \tag{2.5}$$

### 2.2. Interpolative matrix decomposition

Our approach also relies on computing the interpolative decomposition of a matrix. The idea of matrix ID is to find, for a given accuracy $\epsilon$, a near optimal set of columns (rows) of a matrix so that the rest of columns (rows) can be represented as a linear combination from the selected set. Algorithmically, this decomposition proceeds via pivoted QR factorization and so is closely related to the factorization (2.1) in Section 2.1. The fact that a basis for the range of $A$ is constructed purely as a subset of its columns (not necessarily orthonormal) will be crucial when dealing with CTDs. While we quote results in [45], early results on this type of matrix decomposition appeared in [30,31,58].

**Lemma 2.2.** *(See [45].) Suppose $A$ is an $m \times n$ matrix. Then, for any positive integer $k$ with $k \leq m$ and $k \leq n$, there exist a real $k \times n$ matrix $P$, and a real $m \times k$ matrix $A_c$ whose columns constitute a subset of the columns of $A$, such that*

1. *Some subset of the columns of $P$ makes up the $k \times k$ identity matrix,*
2. *no entry of $P$ has an absolute value greater than* 1,
3. $\|P\|_2 \leq \sqrt{k(n-k)+1}$,
4. *the smallest singular value of $P$ is at least* 1,
5. $A_c P = A$ *when* $k = m$ *or* $k = n$, *and*
6. $\|A_c P - A\|_2 \leq \sqrt{k(n-k)+1}\,\tau_{k+1}(A)$ *when* $k < m$ *and* $k < n$, *where* $\tau_{k+1}(A)$ *is the* $k+1$ *singular value of* $A$.

Properties 1, 2, 3, and 4 ensure that the interpolative decomposition of $A$ is numerically stable.

**Remark 2.3.** In [21], the authors show how the rank-revealing QR factorization of $A$ is used to compute decomposition

$$A = A_c \left[\, I \,\middle|\, T \,\right] P_c^* + X, \tag{2.6}$$

where $A_c$ is the column skeleton of $A$, $P = [I \mid T] P_c^*$ and $X = A_c P - A$ as in Lemma 2.2. The matrix $P_c$ is an $n \times n$ permutation matrix and $T$ is a $k \times (n-k)$ matrix that solves an associated linear system (see [21] for details). This decomposition can be extended to include both rows and columns, yielding a $k \times k$ skeleton matrix $A_s$ such that

$$A = P_r^* \left[\begin{array}{c} I \\ \hline S \end{array}\right] A_s \left[\, I \,\middle|\, T \,\right] P_c + X. \tag{2.7}$$

Here $P_r$ is an $m \times m$ permutation matrix and $S$ is an $(m-k) \times k$ matrix analogous to $T$. While the full decomposition (2.7) is used in the proof of Theorem 3.5 (see the Online Supplement), the column oriented decomposition (2.6) is the main ingredient in the tensor ID described in this paper.

### 2.3. A randomized algorithm for interpolative matrix decomposition

The deterministic calculation of the interpolative decomposition is as expensive as the standard QR. However, when $A$ is of low rank, the matrix ID can be constructed efficiently using a randomized algorithm described in [41,61,45,34] under slightly weaker conditions than those in Lemma 2.2. The randomized matrix ID algorithm is summarized in this paper as Algorithm 2.

**Algorithm 2** Randomized algorithm for matrix ID [45].

Input: An $m \times n$ matrix $A$ and integer $\ell > k$.
Output: Indices $\mathcal{L}_k$ of the $k$ skeleton columns, the $k \times n$ coefficient matrix $P$, and the $m \times k$ column skeleton matrix $A_c$.

1. Generate an $m \times \ell$ Gaussian random matrix, $R$.
2. Form the $\ell \times n$ matrix, $Y = R^* A$.
3. Construct QR factorization of $Y$.
4. Using Lemma 2.2, construct $\mathcal{L}_k$ and $P$ from the QR of $Y$.
5. Collect the $k$ skeleton columns into the matrix $A_c$.

Steps 1–3 of Algorithm 2 are the same as those of Algorithm 1 applied to $A^*$; the remainder of the steps reorganize the QR factorization of the projection matrix $Y$. The cost of collecting the $k$ columns of $A$ into matrix $A_c$ requires $\mathcal{O}(k \cdot m)$ operations. Hence, the computational cost of the randomized ID is estimated as

$$\underbrace{\ell \cdot m \cdot t_R}_{\text{Form } R} + \underbrace{\ell \cdot c_A}_{\text{Form } Y} + \underbrace{k \cdot \ell \cdot n}_{\text{ID of } Y} + \underbrace{k \cdot m}_{\text{Form } A_c}. \tag{2.8}$$

To provide an informal description of the properties of the randomized matrix ID, we state

**Lemma 2.4.** *(See Observation 3.3 of [45].) The randomized matrix ID algorithm constructs matrices $A_c$ and $P$ such that*

1. *some subset of the columns of $P$ makes up the $k \times k$ identity matrix*
2. *no entry of $P$ has absolute values greater than 2,*
3. *$\|P\|_2 \leq \sqrt{4k(n-k)+1}$,*
4. *the smallest singular value of $P$ is at least 1,*
5. *$A_c P = A$ when $k = m$ or $k = n$,*
6. *$\|A_c P - A\|_2 \leq \sqrt{4k(n-k)+1}\,\tau_{k+1}(A)$ when $k < m$ and $k < n$, where $\tau_{k+1}(A)$ is the $k+1$ singular value of $A$.*

Comparing this lemma with Lemma 2.2, the only difference is the appearance of an extra factor of size $\sim 2$ in Properties 2, 3 and 6.

**Remark 2.5.** The proofs of the results summarized above assume that the random variables are Gaussian with zero mean and unit variance. However, it is well known that effectiveness of many randomized algorithms is somewhat independent of the distribution. Numerical experiments show that uniform, Bernoulli, log normal and several other distributions also work well in this context. This observation is an important for the discussion of randomized tensor projections (see Section 3.3.1).

### 2.4. Q-factorization for tensors

Collecting together the vectors $\mathbf{u}_j^{(l)}$ in (1.4) from all terms $l = 1, \ldots, r$, we define the $M_j \times r$ directional component matrices,

$$U_{(j)} = \begin{bmatrix} | & | & & | \\ \mathbf{u}_j^{(1)} & \mathbf{u}_j^{(2)} & \cdots & \mathbf{u}_j^{(r)} \\ | & | & & | \end{bmatrix}, \quad j = 1, \ldots, d. \tag{2.9}$$

We refer to the component index $j$ as the direction – rather than dimension – since these components are not necessarily univariate.

Factorizing the directional component matrices of the tensor $\mathcal{U}$ leads to a factorization of $\mathcal{U}$ itself. Consider

$$U_{(j)} = Q_{(j)} S_{(j)}, \quad j = 1, \ldots, d, \tag{2.10}$$

where $Q_{(j)}$ is an $M_j \times k_j$ matrix with orthonormal columns and $S_{(j)}$ is a $k_j \times r$ directional component matrix,

$$S_{(j)} = \begin{bmatrix} | & | & & | \\ \mathbf{s}_j^{(1)} & \mathbf{s}_j^{(2)} & \cdots & \mathbf{s}_j^{(r)} \\ | & | & & | \end{bmatrix}, \quad j = 1, \ldots, d. \tag{2.11}$$

Here $k_j$ is the rank of the matrix $U_{(j)}$ (for a given accuracy $\epsilon$). Given (2.10), it is straightforward to demonstrate that tensor $\mathcal{U}$ in (1.4) admits the decomposition

$$\mathcal{U} = \mathcal{Q}\mathcal{S}, \quad \text{where } \mathcal{Q} = \bigotimes_{j=1}^{d} Q_{(j)}, \; \mathcal{S} = \sum_{l=1}^{r} \sigma_l \bigotimes_{j=1}^{d} \mathbf{s}_j^{(l)}. \tag{2.12}$$

We call this – the $d$ independent factorizations of the component matrices $U_{(j)}$ – the Q-factorization of tensor $\mathcal{U}$. Notice that $\mathcal{Q}$ is a separation rank-one operator, and $\mathcal{S}$ is a CTD that typically has significantly smaller component dimensions than $\mathcal{U}$. Since $\mathcal{S} \in \bigotimes_{j=1}^{d} \mathbb{R}^{k_j}$ with $k_j \leq r$, for many problems of practical interest, we have $k_j \ll M_j$ for some or all of the directions $j = 1, \ldots d$. We note that Q-factorization is not new and is known under different names, e.g., CANDELINC or simply compression (see Section 5 of [39] and references therein).

It is easy to show

**Lemma 2.6.** *Suppose that a rank-r canonical tensor admits the Q-factorization $\mathcal{U} = \mathcal{Q}\mathcal{S}$ as in (2.12). Then we obtain*

$$\|\mathcal{U}\|_F = \|\mathcal{S}\|_F. \tag{2.13}$$

Due to this lemma, the accuracy of the rank reduction for the tensor $\mathcal{S}$ is the same as that for the original tensor $\mathcal{U}$. To see this, let $\mathcal{S}_k = \sum_{l=1}^{k} \hat{\sigma}_l \bigotimes_{j=1}^{d} \hat{\mathbf{s}}_j^{(l)}$ and assume it satisfies

$$\|\mathcal{S} - \mathcal{S}_k\|_F \leq \epsilon \|\mathcal{S}\|_F. \tag{2.14}$$

Then, setting $\mathcal{U}_k = \mathcal{Q}\mathcal{S}_k$, we have

$$\|\mathcal{U} - \mathcal{U}_k\|_F = \left\| \mathcal{Q}(\mathcal{S} - \mathcal{S}_k) \right\|_F = \|\mathcal{S} - \mathcal{S}_k\|_F \leq \epsilon \|\mathcal{U}\|_F. \tag{2.15}$$

## 3. Randomized tensor interpolative decomposition

### 3.1. On the interpolative decomposition of symmetric matrices

We briefly describe a matrix ID for symmetric matrices because it parallels the development for tensors (and, to our knowledge, its description is not available in the literature). Suppose $A$ is an $m \times n$ matrix of rank $k$ (for a given accuracy $\epsilon$) where $m \gg n$, and let $B = A^*A$ denote its $n \times n$ Gram matrix (see (1.11) and (1.12)). Theorem 3.1 below relates the error in approximating $B$ by a symmetric matrix ID to that of approximating $A$ by a corresponding matrix ID that uses only the decomposition of $B$. The error estimate is of interest in cases where the cost of constructing the matrix ID of $B$ is relatively small compared with that of $A$.

**Theorem 3.1.** *Suppose $A$ is an $m \times n$ matrix of rank $k$. Then the $n \times n$ matrix $B = A^*A$ admits a symmetric interpolative decomposition of the form*

$$B_k = P^* B_s P, \tag{3.1}$$

*where $B_s = A_c^* A_c$. The $m \times k$ column skeleton $A_c$ of $A$, and $k \times n$ coefficient matrix $P$ can be computed using only $B$. In addition,*

$$\|A - A_c P\|_2 = \|B - B_k\|_2^{1/2} \le C(n,k)\tau_{k+1}(A), \tag{3.2}$$

*where*

$$C(n,k) = \big(4k(n-k)+1\big)^{1/4}\big(1+\sqrt{nk(n-k)}\big)^{1/2} = \mathcal{O}\big(n^{3/4}\big), \tag{3.3}$$

*and $\tau_{k+1}(A)$ is the $k+1$ singular value of $A$.*

The proof of Theorem 3.1 is similar to that of Theorem 3 in [21] and appears in the Online Supplement.

Theorem 3.1 allows us to compute the interpolative decomposition of $A$ via its Gram matrix, $B$. However, for the matrix ID of $A$ to have accuracy $\epsilon$, we have to compute the ID of $B$ to an accuracy of $\epsilon^2$. This presents a limitation since, if $\epsilon^2 < \epsilon_{machine}$, this approach will select all columns of $B$. Thus, in general, the ID of $A$ can be computed in this manner only for accuracies $\sqrt{\epsilon_{machine}} < \epsilon$.

**Remark 3.2.** An exception to the last statement occurs if $B$ happens to be a structured matrix of a particular type (see [25]) since then its singular values can be computed with relative precision. For example, if the univariate functions in separated representation (1.1) are exponentials or Gaussians, then $B$ is a Cauchy matrix which is the key example in [25].

### 3.2. Tensor CTD-ID problem

Our goal is to reduce the tensor ID problem to that of skeletonization of certain matrices and to relate the error of the tensor ID to that of the corresponding matrix ID. A CTD in (1.2) can always be interpreted as a dense tensor with $N = \prod_{j=1}^{d} M_j$ elements and represented as the $N \times r$ matrix $U$ in (1.11). While the formation of such a matrix is in *no way practical*, it allows us to establish a connection between the matrix and tensor settings.

Assume for a moment that we are able to compute the matrix ID of $U$ directly as well as the corresponding tensor ID of $\mathcal{U}$. As already mentioned in Section 1.2, given a rank-$k$ matrix ID of $U$, we can construct a rank-$k$ approximation $\mathcal{U}_k$ via

$$\mathcal{U}_k = \sum_{m=1}^{k} \alpha_m \mathcal{U}^{(l_m)}, \quad \alpha_m = \sum_{l=1}^{r} \sigma_l P_{ml}, \tag{3.4}$$

where $l_m \in \mathcal{L}_k$ are the $k$ indices of the skeleton columns of $U$ and $P$ is the $k \times r$ coefficient matrix. Each of the original rank-one terms $\mathcal{U}^{(l)}$ of $\mathcal{U}$ in (1.4) has its own representation via the skeleton terms,

$$\mathcal{U}^{(l)} = \sum_{m=1}^{k} P_{ml} \mathcal{U}^{(l_m)}, \quad l = 1, \ldots, r. \tag{3.5}$$

Now define the $N \times r$ matrix

$$U - U_k = \begin{bmatrix} | & | & & | \\ \sigma_1 \mathcal{U}^{(1)} - \mathcal{U}_k^{(1)} & \sigma_2 \mathcal{U}^{(2)} - \mathcal{U}_k^{(2)} & \cdots & \sigma_r \mathcal{U}^{(r)} - \mathcal{U}_k^{(r)} \\ | & | & & | \end{bmatrix}, \tag{3.6}$$

where tensors are interpreted as vectors in $\mathbb{R}^N$ (but are not necessarily of separation rank one). The error of the tensor ID can be bounded from above by the error in the matrix ID of $U$.

**Theorem 3.3.** *Suppose $\mathcal{U}$ is a rank-r canonical tensor, and let $U$ denote the $N \times r$ matrix in (1.11). Further, suppose that a rank-k tensor ID of $\mathcal{U}_k$ of $\mathcal{U}$ is given as in (3.4). Then we have*

$$\|\mathcal{U} - \mathcal{U}_k\|_F \leq \sqrt{r}\|U - U_k\|_F \leq r\|U - U_k\|_2. \tag{3.7}$$

For the proof, we need the following

**Lemma 3.4.** *For any rank-r canonical tensor $\mathcal{U}$ and corresponding matrix $U$ in (1.11), we have*

$$\|\mathcal{U}\|_F \leq \sqrt{r}\|U\|_F. \tag{3.8}$$

**Proof.** By definition, we have

$$\|U\|_F^2 = \sum_{i_1=1}^{M_1} \cdots \sum_{i_d=1}^{M_d} \sum_{l=1}^{r} \left(\sigma_l \mathcal{U}^{(l)}_{i_1 \ldots i_d}\right)^2.$$

Since any real numbers $\{a_l\}_{l=1}^r$ satisfy $(\sum_{l=1}^r a_l)^2 \leq r \sum_{l=1}^r a_l^2$, it follows that

$$\begin{aligned}\|\mathcal{U}\|_F^2 &= \sum_{i_1=1}^{M_1} \cdots \sum_{i_d=1}^{M_d} \mathcal{U}^2_{i_1 \ldots i_d} = \sum_{i_1=1}^{M_1} \cdots \sum_{i_d=1}^{M_d} \left( \sum_{l=1}^{r} \sigma_l \mathcal{U}^{(l)}_{i_1 \ldots i_d} \right)^2 \\ &\leq r \sum_{i_1=1}^{M_1} \cdots \sum_{i_d=1}^{M_d} \sum_{l=1}^{r} \left(\sigma_l \mathcal{U}^{(l)}_{i_1 \ldots i_d}\right)^2 = r\|U\|_F^2. \qquad \square\end{aligned}$$

To prove Theorem 3.3, we observe that the first inequality follows directly from Lemma 3.4. Letting $\tau_1 \geq \tau_2 \geq \cdots \geq \tau_r \geq 0$ denote the eigenvalues of the $r \times r$ matrix $(U - U_k)^*(U - U_k)$, we have

$$\|U - U_k\|_F^2 = \operatorname{tr}(U - U_k)^*(U - U_k) = \sum_{j=1}^{r} \tau_j \leq r\tau_1 = r\|U - U_k\|_2^2.$$

For computing the tensor ID, instead of the $N \times r$ matrix $U$ we can use the $r \times r$ Gram matrix. The next theorem relates the error in the tensor ID of $\mathcal{U}$ computed via its Gram matrix to that of the matrix ID of the Gram matrix itself.

**Theorem 3.5.** *Suppose $G$ is the $r \times r$ Gram matrix defined in (1.12). Let a symmetric interpolative decomposition of $G$ be given by*

$$G_k = P^* G_s P, \tag{3.9}$$

*where $\|G - G_k\|_2 \leq \epsilon_k$, $G_s$ is a $k \times k$ sub-matrix of $G$, and $P$ is a $k \times r$ coefficient matrix. Then the corresponding tensor ID $\mathcal{U}_k$ of $\mathcal{U}$ satisfies*

$$\|\mathcal{U} - \mathcal{U}_k\|_F \leq r\sqrt{\epsilon_k}. \tag{3.10}$$

**Proof.** The proof is a straightforward consequence of Theorem 3.1 and is obtained by setting $A = U$ and $B = G$, and using Theorem 3.3. $\square$

**Remark 3.6.** The matrix $P$ appearing in Theorem 3.5 coincides with that of $P_r$ in (2.7). In other words, computing the symmetric ID of $G$ requires computing the full skeletonization. We refer the reader to the proof in the Online Supplement for additional details.

### 3.3. Projection algorithm with random tensors in canonical form

#### 3.3.1. Random tensors in canonical form

In order to form the matrix $Y$ in (1.15), we generate random rank-one tensors in canonical form,

$$\mathcal{R} = \bigotimes_{j=1}^{d} \mathbf{r}_j = \prod_{j=1}^{d} r_{i_j}, \quad i_j = 1, \ldots, M_j. \tag{3.11}$$

If the entries of $\mathbf{r}_j$ are realizations of independent random variables with zero mean and unit variance, then each entry of $\mathcal{R}$ has zero mean and unit variance as well:

**Table 1**
Estimates of computational complexity.

| Reduction method | Computational complexity |
|---|---|
| ALS | $(d \cdot k^4 \cdot M) \cdot (\textit{number of iterations}) \sim (d \cdot k^6) \cdot (\textit{number of iterations})$ |
| Tensor ID: random projection | $d \cdot k^3 \cdot M + k^4 \sim d \cdot k^5$ |

$$\mathbb{E}[\mathcal{R}_{i_1 \dots i_d}] = \mathbb{E}\left[\prod_{j=1}^{d} r_{i_j}\right] = \prod_{j=1}^{d} \mathbb{E}[r_{i_j}] = 0, \tag{3.12}$$

and

$$\mathbb{E}\left[\left(\mathcal{R}_{i_1 \dots i_d} - \mathbb{E}[\mathcal{R}_{i_1 \dots i_d}]\right)^2\right] = \mathbb{E}\left[\mathcal{R}^2_{i_1 \dots i_d}\right] = \prod_{j=1}^{d} \mathbb{E}\left[(r_{i_j})^2\right] = 1. \tag{3.13}$$

In our numerical experiments, we consider several distributions for the entries of the vectors $\mathbf{r}_j$, specifically,

- Normal
- Uniform
- Bernoulli
- $1/d$ power distribution, i.e., $r_{i_j} = \text{sign}(\tilde{r}_{i_j}) \times |\tilde{r}_{i_j}|^{1/d}$, where, e.g., $\tilde{r}_{i_j} \sim N(0,1)$

Algorithm 3 below shows the steps for constructing the tensor ID via random projections. As before, $\ell$ is an integer slightly larger than the expected separation rank of the approximation, $k$.

**Algorithm 3** Tensor ID via random projection.

Input: A rank-$r$ CTD $\mathcal{U}$ and anticipated separation rank $\ell > k$.
Output: A rank-$k$ tensor ID, $\mathcal{U}_k$.

1. Form the $\ell$ random CTDs $\mathcal{R}^{(l)}$ as in (3.11).
2. Form the $\ell \times r$ projection matrix $Y$ as in (1.15).
3. Compute the rank-$k$ matrix ID of $Y$ via Algorithm 2.
4. Form the rank-$k$ tensor ID $\mathcal{U}_k$ via (3.4).

*3.3.2. Cost estimate*

We estimate the cost of the random projection method using $\ell$ random tensors assuming that $M_1 = \cdots = M_d = M$. As we did in Section 2.1, let $t_R$ denote the cost of generating a single random number used to construct $\mathcal{R}^{(l)}$, $l = 1, \dots \ell$. Forming the collection of $\ell$ random tensors costs $\mathcal{O}(d \cdot \ell \cdot t_R \cdot M)$. Computing each element of matrix $[Y]_{lm} = \langle \mathcal{R}^{(l)}, \sigma_m \mathcal{U}^{(m)} \rangle$ requires $\mathcal{O}(d \cdot M)$ operations so that formation of $Y$ takes $\mathcal{O}(d \cdot \ell \cdot r \cdot M)$ operations. The cost of obtaining the matrix ID of $Y$ is given in Section 2.3. Finally, constructing the rank-$k$ tensor ID by collecting the $k$ rank-one terms of $\mathcal{U}$ requires $\mathcal{O}(d \cdot k \cdot M)$ operations, and computing the new $s$-values, an additional $\mathcal{O}(k \cdot r)$ operations. Hence, the total cost is on the order of

$$\underbrace{d \cdot \ell \cdot M \cdot t_R}_{\text{Form tensors } \mathcal{R}^{(l)}} + \underbrace{d \cdot \ell \cdot r \cdot M}_{\text{Form } Y} + \underbrace{k \cdot \ell \cdot r}_{\text{ID of } Y} + \underbrace{k \cdot (d \cdot M + r)}_{\text{Form } \mathcal{U}_k}. \tag{3.14}$$

*3.3.3. Comparison of computational costs*

For comparison of the computational complexity of the algorithms, let us consider the case where the nominal (original) separation rank $r \sim \mathcal{O}(k^2)$ is reducible to $\mathcal{O}(k)$ via tensor ID and $M \gg r$ is the same in all directions. This situation commonly occurs when, e.g., two functions are multiplied together, or when an operator is applied to a function represented in separated form. The situation where $M \gg r$ is typical when the component vectors $\mathbf{u}_j^{(l)}$ are two- or three-dimensional. For simplicity, let us ignore the cost of generating a random numbers since this cost is not significant.

Before comparing the algorithms, we remark on the roles that Q-factorization and tensor ID play in reducing the size of relevant computational parameters. After performing Q-factorization on a given tensor, the size $M$ is reduced to at most $r$. Therefore, in all of the estimates below we replace $M$ by $k^2$.

Combining these assumptions with the estimates (3.14), we arrive at computational complexities shown in Table 1. The tensor ID approach is faster than ALS by a remarkable factor $k \cdot$ (number of iterations), where the number of iterations in ALS method is quite large.

## 4. Frobenius norm and $s$-norm of a tensor

Since we have no way to add/subtract two tensors in CTD form directly, computing the Frobenius norm of the difference of two tensors via

$$\|\mathcal{U}-\mathcal{V}\|_F = \left(\|\mathcal{U}\|_F^2 - 2\langle \mathcal{U},\mathcal{V}\rangle + \|\mathcal{V}\|_F^2\right)^{1/2} \tag{4.1}$$

results in the loss of significant digits if the two tensors are close. As an alternative to the Frobenius norm, we use the spectral, or *s-norm,* of a tensor by computing its rank-one approximation. Rank-one approximations of a tensor are well understood (see, e.g., [23,62,24,36]). We use the fact that the largest $s$-value of the rank-one approximation has all the necessary properties to be used as a norm (see Lemma 4.1 below).

The rank-one approximation of tensor $\mathcal{U}$ can be found by solving the system of multi-linear equations [23,62,42,36],

$$\sigma \mathbf{x}_{j'} = \sum_{l=1}^{r} \sigma_l \prod_{j=1,\ j\neq j'}^{d} \langle \mathbf{u}_j^{(l)}, \mathbf{x}_j\rangle, \quad j' = 1,2,\ldots,d, \tag{4.2}$$

where $\|\mathbf{x}_1\|_F = \|\mathbf{x}_2\|_F = \cdots = \|\mathbf{x}_d\|_F = 1$. The solution of this system maximizes

$$\sigma = \sum_{l=1}^{r} \sigma_l \prod_{j=1}^{d} \langle \mathbf{u}_j^{(l)}, \mathbf{x}_j\rangle, \tag{4.3}$$

so that $\sigma$ coincides with $s$-norm $\|\mathcal{U}\|_s$ if the global maximum is attained in solving (4.2).

**Lemma 4.1.** *Suppose* $\mathcal{U} = \sum_{l=1}^{r} \sigma_l \bigotimes_{j=1}^{d} \mathbf{u}_j^{(l)}$ *is a rank-r CTD. Consider*

$$\|\mathcal{U}\|_s = \sup_{\|\hat{\mathbf{x}}_j\|_F=1,\ j=1,\ldots,d} \left( \sum_{l=1}^{r} \sigma_l \prod_{j=1}^{d} \langle \mathbf{u}_j^{(l)}, \hat{\mathbf{x}}_j\rangle \right), \tag{4.4}$$

*where* $\hat{\mathcal{X}} = \bigotimes_{j=1}^{d} \hat{\mathbf{x}}_j$ *is the best rank-one approximation of* $\mathcal{U}$*. Then* $\|\mathcal{U}\|_s$ *satisfies the properties of a norm,*

1. $\|\alpha\mathcal{U}\|_s = |\alpha|\,\|\mathcal{U}\|_s$ *for any* $\alpha \in \mathbb{R}$.
2. $\|\mathcal{U}\|_s = 0$ *if and only if* $\mathcal{U} = 0$.
3. $\|\mathcal{U}+\mathcal{V}\|_s \le \|\mathcal{U}\|_s + \|\mathcal{V}\|_s$.

**Proof.** First, we observe that for a rank-one tensor $\mathcal{X} = \sigma\hat{\mathcal{X}}$ satisfying (4.2) and (4.3),

$$\begin{aligned} \|\mathcal{U}-\mathcal{X}\|_F^2 &= \|\mathcal{U}\|_F^2 - 2\langle \mathcal{U},\mathcal{X}\rangle + \|\mathcal{X}\|_F^2 \\ &= \|\mathcal{U}\|_F^2 - \sigma^2. \end{aligned} \tag{4.5}$$

Thus, the best rank-one approximation with unit norm, $\hat{\mathcal{X}}$, maximizes the quantity $\langle \mathcal{U},\hat{\mathcal{X}}\rangle$.

If $\alpha \ge 0$, then (1) is obvious. If $\alpha < 0$, then $\langle \mathcal{U},\hat{\mathcal{X}}\rangle = -\alpha\sigma$ and changing the sign of any single $\hat{\mathbf{x}}_j$, $j = 1,\ldots,d$, we obtain (1). For any $\mathcal{U} \neq 0$, there exists at least one rank-one tensor $\mathcal{X}$ such that $\langle \mathcal{U},\mathcal{X}\rangle > 0$, implying that $\|\mathcal{U}\|_s > 0$. In fact, we can take $\mathcal{X} = \mathcal{U}^{(l')}$, where $\mathcal{U}^{(l')}$ is one of the rank-one terms of $\mathcal{U}$. Indeed, assuming that $\langle \mathcal{U},\mathcal{U}^{(l)}\rangle \le 0$ for all $l = 1,\ldots,r_u$, and writing the Frobenius norm of $\mathcal{U}$ as

$$\|\mathcal{U}\|_F^2 = \sum_{l} \sigma_l \langle \mathcal{U},\mathcal{U}^{(l)}\rangle \le 0, \tag{4.6}$$

we conclude that $\mathcal{U} = 0$. Thus, there is at least one rank-one term of $\mathcal{U}$ such that $\langle \mathcal{U},\mathcal{U}^{(l)}\rangle > 0$. Finally, the triangle inequality follows since

$$\begin{aligned} \|\mathcal{U}+\mathcal{V}\|_s &= \sup_{\mathcal{X}:\|\mathcal{X}\|_F=1} \langle \mathcal{U}+\mathcal{V},\mathcal{X}\rangle \\ &\le \sup_{\mathcal{Y}:\|\mathcal{Y}\|_F=1} \langle \mathcal{U},\mathcal{Y}\rangle + \sup_{\mathcal{Z}:\|\mathcal{Z}\|_F=1} \langle \mathcal{V},\mathcal{Z}\rangle \\ &= \|\mathcal{U}\|_s + \|\mathcal{V}\|_s. \end{aligned} \tag{4.7}$$

where $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ are rank-one tensors. □

Eqs. (4.2) are solved by an iteration equivalent to ALS for approximating via rank-one tensors. However, in this case, there is no linear system to solve and, thus, the iteration resembles the power method for matrices. Specifically, for each direction $j$, the iteration proceeds by updating the left hand side, vector $\mathbf{x}_{j'}$, by evaluating the right hand side with the currently available vectors $\mathbf{x}_j$. Re-normalizing $\mathbf{x}_{j'}$ to obtain the normalization factor $\sigma$, and sweeping through the directions, the iteration terminates when the change in $\sigma$ is small. However, unlike the power method for matrices, this iteration may have more than one stationary point, meaning that the answer may depend on the initialization.

**Remark 4.2.** Although the definition of the *s-norm* parallels the matrix 2-norm and can be useful as a way of estimating errors, computing the $s$-norm exactly for arbitrary dense tensors is claimed to be an NP-hard problem in [36]. We note that for symmetric tensors, the global convergence of the iteration described above has been claimed in [40].

We discuss our approach to initialization below but first consider the cost of estimating the $s$-norm and its relation to the Frobenius norm. Let us assume that $M_j = M$ for $j = 1, \ldots, d$, and let $n_{it}$ denote the number of iterations required for the rank one iteration to converge ($n_{it}$ is usually small). For each direction, $(d-1)\cdot r$ inner products are computed at a cost of $\mathcal{O}(M)$ operations each. Since only one inner product must be updated at a time, the total computational cost is estimated as

$$n_{it} \cdot d \cdot r \cdot M. \tag{4.8}$$

For comparison, the cost of computing the Frobenius norm via (1.7) and (1.8) is $\mathcal{O}(d \cdot r^2 \cdot M)$, so that estimating the $s$-norm is faster if $n_{it} < r$. Another advantage of using the $s$-norm is that there is no loss of significant digits in computing it via (4.2). We have

**Lemma 4.3.** *The Frobenius and s-norms satisfy*

$$\frac{1}{\sum_{l=1}^{r} \sigma_l} \|\mathcal{U}\|_F^2 \le \|\mathcal{U}\|_s \le \|U\|_F, \tag{4.9}$$

*where $\mathcal{U}$ is defined in (1.4), $U$ in (1.11) and $\|U\|_F = (\operatorname{tr} U^* U)^{1/2} = (\sum_{l=1}^{r} \sigma_l^2)^{1/2}$.*

**Proof.** By definition of the matrix 2-norm, we have

$$\|U\|_2 = \left\|U^*\right\|_2 = \max_{\|\mathbf{x}\|_2=1} \left\|U^*\mathbf{x}\right\|_2. \tag{4.10}$$

Here $\mathbf{x}$ is a vector in $\mathbb{R}^N$ corresponding to a dense tensor. Thus, taking $\mathbf{x}$ corresponding to the rank-one tensor achieving the best approximation to $\mathcal{U}$ and satisfying (4.2) and (4.3), we obtain $\|\mathcal{U}\|_s \le \|U\|_2 \le \|U\|_F$. Starting from $\|\mathcal{U}\|_F^2 = \sum_{l=1}^{r} \sigma_l \langle \mathcal{U}^{(l)}, \mathcal{U} \rangle$ and replacing the rank one terms $\mathcal{U}^{(l)}$ with the best rank one approximation $\mathcal{X}$ of the tensor $\mathcal{U}$, we obtain

$$\sum_{l=1}^{r} \sigma_l \langle \mathcal{U}^{(l)}, \mathcal{U} \rangle \le \sum_{l=1}^{r} \sigma_l \langle \mathcal{X}, \mathcal{U} \rangle = \|\mathcal{U}\|_s \sum_{l=1}^{r} \sigma_l. \qquad \square \tag{4.11}$$

### 4.1. Approximating the s-norm

The proof of Lemma 4.1 assumes that the best rank-one approximation can be obtained. However, in general, the iteration in (4.2) may converge to a local maximum and, thus, the result may depend on the initialization. Therefore, we need a systematic approach to initializing (4.2). Several approaches to initialization have been suggested previously (see, e.g., [23, 38]).

We initialize the iteration by using components of the given tensor $\mathcal{U}$ (rather than a random initialization). In applications we are interested in, a small number of terms typically dominate the representation so that it is often sufficient to initialize using the term with the largest $s$-value. Alternatively, we generated the initial guess using component matrices in each direction by either averaging their columns or computing the singular vector corresponding to their largest singular value. Obviously, these are heuristic choices but they appear to work well in the computational environment we are interested in. For any of these initialization methods, there is no guarantee that they provide the globally optimal rank-one approximation for an arbitrary CTD.

In the environment of computing the tensor ID, a weaker form of the definition for the $s$-norm may be appropriate. Instead of demanding that the s-norm be the global maximum of (4.3), we can define it to be the maximally computed $\sigma$ using a specific initialization method. However, this is only permitted if this definition of the s-norm satisfies the triangle inequality in Lemma 4.1. In other words, if we denote $\sigma^u$ and $\sigma^v$ to be the maximal computed (not necessarily global) $s$-values for tensors $\mathcal{U}$ and $\mathcal{V}$, it is necessary that $\sigma^{(u+v)}$ computed using the same initialization method satisfies $\sigma^u + \sigma^v \le \sigma^{(u+v)}$. When computing tensor ID under this definition, we can always verify the triangle inequality *a posteriori*.

## 5. Examples

All of the numerical examples were implemented in MATLAB [46]. We used the CTD data structure and basic routines available through the Sandia Tensor Toolbox 2.5 [5] and implemented ALS, tensor ID, and all additional routines with no special effort to optimize or parallelize the codes. All experiments were performed on a PC laptop with a 2.20 GHz Intel i7 chipset and 8 GB of RAM.

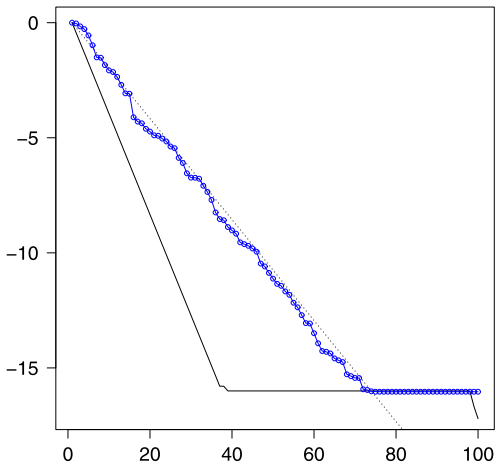


**Fig. 1.** Decay of singular values of the matrices in Example 5.1. The logarithm ($\log_{10}$) of the singular values are displayed as a function of their index. The singular values of $G$ are plotted with a solid line. The singular values of $Y$ are displayed using symbol "∘", while the original $s$-values of the tensor $\mathcal{U}$ are displayed using fine dots and nearly coincide with the singular values of $Y$.

### 5.1. Comparison of CTD rank reduction algorithms

We first illustrate the loss of significant digits in constructing the tensor ID using the Gram matrix in (1.12) by comparing its numerical rank with that of the matrix generated via the randomized approach in Section 3.3.

For the comparison, we construct a tensor $\mathcal{U}$ with terms that are nearly orthogonal and with $s$-values decaying exponentially fast. In this case, a tensor ID for a user-selected accuracy $\epsilon$ can be obtained by simply dropping terms with small $s$-values. Therefore, we can directly compare the number of the skeleton terms simply by estimating the numerical rank of the associated matrices (without actually computing new tensor ID coefficients).

Consider a random tensor in dimension $d = 20$, with $M_1 = \cdots = M_{20} = 128$ and $r = 100$, and generated as

$$\mathcal{U} = \sum_{l=1}^{r} \sigma_l \mathcal{U}^{(l)} \quad \text{with} \quad \mathcal{U}^{(l)} = \bigotimes_{j=1}^{d} \mathbf{u}_j^{(l)}, \; u_{i_j}^{(l)} \sim N(0,1), \tag{5.1}$$

where $N(0,1)$ denotes the normal distribution with zero mean and unit variance and where the vectors $\mathbf{u}_j^{(l)}$ are normalized to have unit Frobenius norm. The $s$-values assigned to the terms are exponentially decaying,

$$\sigma_l = \exp(-l/2), \quad l = 1, \ldots, r. \tag{5.2}$$

By construction, the terms of $\mathcal{U}$ are nearly orthogonal so that the truncation error incurred by removing small terms is approximately

$$\epsilon_{l'} = \left( \sum_{l>l'} \sigma_l^2 \right)^{1/2}. \tag{5.3}$$

Therefore, using $\epsilon_{l'} \leq \epsilon$, the tensor ID for accuracy $\epsilon$ should select the first $l'$ terms. In order for the tensor ID to succeed in choosing these terms, the matrices for its construction must have numerical rank greater than $l'$.

We compute the Gram matrix $G$ via (1.12). For the random projection method, we generate the tensors $\mathcal{R}^{(l)}$ for $l = 1, \ldots, r$ and form the $r \times r$ projection matrix $Y$ via (1.15). We then compute the singular values of the matrices $G$ and $Y$.

The singular values of the matrices $G$ and $Y$ are shown in Fig. 1. For reference, we also plot the $s$-values of $\mathcal{U}$. Notice that for accuracy $\epsilon \approx 10^{-16}$ the numerical rank of $G$ is only ∼35 since, as expected, the singular values of $G = U^*U$ decay twice as fast logarithmically as those of matrix $U$. This implies that the tensor ID using $G$, computed in double precision with ≈ 16 accurate digits, loses its ability to distinguish significant terms for requested accuracies smaller than $\approx 10^{-8}$. On the other hand, for accuracies $\ll 10^{-8}$, the numerical rank of $Y$ allows us to select for up to 75–80 terms. The displayed results do not depend in any significant way on the choices of distributions in the random projection method described in Section 3.3.

#### 5.1.1. A trivial example of reduction of separation rank

Continuing with the previous example, we impose additional structure on the terms of $\mathcal{U}$ and choose the last 30 terms (at random) from the first 70 terms and give them the exponentially decaying weights in (5.3). Since by the original construction the terms were nearly mutually orthogonal, for double precision accuracy the algorithms should produce ∼70 terms (cf. Fig. 2), i.e., choose all the linearly independent terms of the tensor.

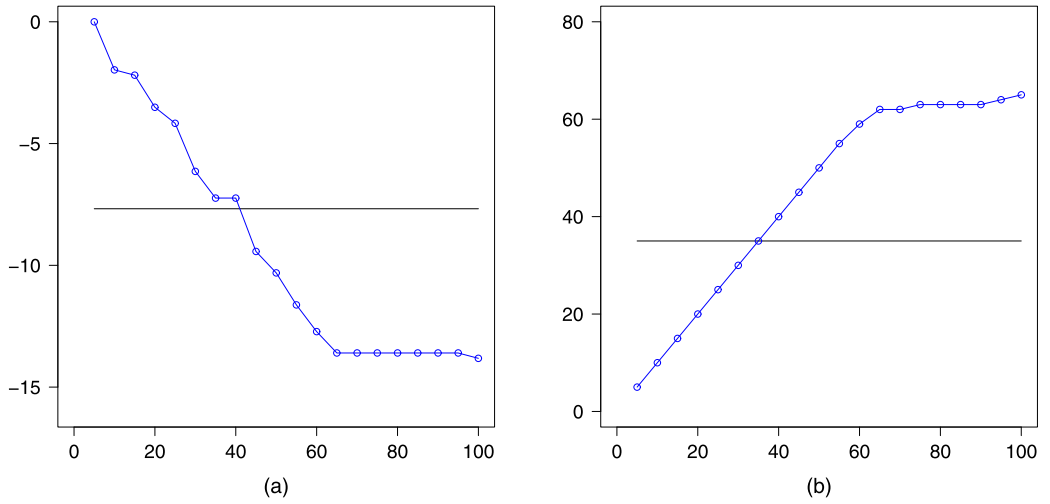


**Fig. 2.** (a) Logarithm ($\log_{10}$) of relative accuracy of computing tensor ID using the random projection method and (b) the resulting separation rank as a function of the number of projections, $\ell$, for Example 5.1. Results for the Gram matrix approach (which do not depend on the parameter $\ell$) are shown as a horizontal solid line. The errors are computed using the $s$-norm.

Results for the Gram and randomized tensor ID algorithms are shown in Fig. 2. The left plot shows the relative $s$-norm error plotted against the separation rank, $\ell$, of the tensor IDs computed via the matrices $G$ and $Y$. The right plot shows the separation rank of the tensor ID approximation as a function of $\ell$. The underlying separation rank of the CTD is known to be $r \sim 70$ for $\epsilon \sim \epsilon_{machine}$, and the error for the randomized methods levels off when $k$ approaches this value. The Gram method, however, is only able to identify the first ∼35 terms due to inherent loss of accuracy.

The Frobenius error for the randomized tensor ID approximately matches the $s$-norm error until the cutoff of $\sqrt{\epsilon_{machine}}$ is attained, at which point it stays constant with respect to $k$ (not shown). Hence, this example also demonstrates the usefulness of the $s$-norm when high accuracy is sought.

### 5.1.2. Comparison of CTD-ID and ALS

In order to illustrate the use of the CTD-ID, we construct CTDs that cannot be represented exactly by subsets of their terms. Then, for a fixed relative accuracy $\epsilon = 10^{-4}$, we construct low-rank approximations using both ALS and CTD-ID and compare the resulting separation ranks and CPU times. We consider CTDs of the form

$$\mathcal{U} = \sum_{l=1}^{r} \bigotimes_{j=1}^{d} \mathbf{u}_j^{(l)}\left(\boldsymbol{\theta}_j^{(l)}(\sigma)\right), \tag{5.4}$$

where $\mathbf{u}_j^{(l)}$ are the spherical coordinates of a unit vector in $\mathbb{R}^M$ with $M = 5$, parameterized by the $M - 1 = 4$ angles $\boldsymbol{\theta}_j^{(l)}$ drawn at random from $U(\sigma[-1, 1])$ for all $j, l$. Here $\sigma$ is a scalar satisfying $0 \le \sigma \le \pi$ which controls the cross-correlation between the $r$ terms of the CTD. Letting $\rho_{ik}$ denote the cross-correlation between terms $i, i'$ in the CTD, we note that $\min_{ii'} |\rho_{ii'}| \to 1$ as $\sigma \to 0$ and $\min_{ii'} |\rho_{ii'}| \to 0$ as $\sigma \to \pi$ in expectation.

Results for dimension $d = 10$ and $\sigma = 0.05$, 0.10, and 0.20 are presented in Table 2. For representations with high cross-correlation between terms, the CTD-ID computes an approximation with only slightly more terms than ALS but in far less time. As $\sigma$ increases and the terms become less cross-correlated, CTD-ID requires more terms than ALS to reach the desired accuracy but, again, in far less time. Thus (in some applications) it may be advantageous to compute the CTD-ID first and then apply ALS to the result.

### 5.1.3. Comparison of CTD-ID and Gram matrix reduction algorithms

To emphasize the advantage of using the randomized projection over the Gram matrix based method as discussed in Section 1.2, we provide a comparison in Table 3. We consider the same CTD as above with $\sigma = 0.20$ and $d = 5$. In order to compare the methods directly, we compute first the best possible approximation using the Gram-based method and report its separation rank, $k_G$, and accuracy in the $s$-norm, $\|\epsilon_G\|_s$. We then apply CTD-ID to obtain separation ranks $k_{ID}^{(1)} = k_G$, $k_{ID}^{(2)} = k_G + 25$, and $k_{ID}^{(3)} = k_G + 50$ with corresponding accuracies $\|\epsilon_{ID}^{(1)}\|_s$, $\|\epsilon_{ID}^{(2)}\|_s$ and $\|\epsilon_{ID}^{(3)}\|_s$. We note that such a procedure of incrementally increasing the separation rank beyond $k_G$ is not possible using the Gram matrix based method since the condition number of the Gram matrix is $\sim 10^{16}$.

**Table 2**
Results for Example 5.1.2. The parameter $\sigma$ controls the cross-correlation of the terms in (5.4), which become more correlated as $\sigma \to 0$. The separation rank of the original CTD is given by $r$, and the minimum cross-correlation between terms by $\min_{ii'} |\rho_{ii'}|$. The parameters $k_{ID}$ and $k_{ALS}$ indicate the separation rank for CTD-ID and ALS approximations, respectively, while $t_{ID}$ and $t_{ALS}$ indicate CPU times (in seconds) to construct them using MATLAB on a PC laptop with a 2.20 GHz Intel i7 chipset and 8 GB of RAM.

| | $r$ | $\min_{ii'} \lvert\rho_{ii'}\rvert$ | $k_{ID}$ | $k_{ALS}$ | $t_{ID}$ (s) | $t_{ALS}$ (s) |
|---|---|---|---|---|---|---|
| | 200 | 0.952 | 7 | 4 | 0.210 | 2.94 |
| | 400 | 0.952 | 7 | 5 | 0.208 | 4.76 |
| | 800 | 0.952 | 7 | 4 | 0.238 | 9.54 |
| (a) $\sigma = 0.05$ | 1600 | 0.951 | 7 | 5 | 0.265 | 50.3 |
| | 3200 | 0.951 | 8 | 4 | 0.513 | 178 |
| | 6400 | 0.951 | 8 | – | 1.25 | – |
| | 12 800 | 0.951 | 7 | – | 3.98 | – |
| | $r$ | $\min_{ii'} \lvert\rho_{ii'}\rvert$ | $k_{ID}$ | $k_{ALS}$ | $t_{ID}$ (s) | $t_{ALS}$ (s) |
| | 200 | 0.820 | 16 | 5 | 0.271 | 3.58 |
| | 400 | 0.818 | 16 | 5 | 0.291 | 4.69 |
| | 800 | 0.819 | 15 | 5 | 0.301 | 12.4 |
| (b) $\sigma = 0.10$ | 1600 | 0.818 | 16 | 5 | 0.353 | 49.2 |
| | 3200 | 0.818 | 16 | 5 | 0.659 | 255 |
| | 6400 | 0.818 | 16 | – | 1.65 | – |
| | 12 800 | 0.818 | 15 | – | 5.39 | – |
| | $r$ | $\min_{ii'} \lvert\rho_{ii'}\rvert$ | $k_{ID}$ | $k_{ALS}$ | $t_{ID}$ (s) | $t_{ALS}$ (s) |
| | 200 | 0.446 | 31 | 8 | 0.340 | 7.76 |
| | 400 | 0.442 | 32 | 9 | 0.408 | 10.6 |
| | 800 | 0.448 | 30 | 12 | 0.416 | 23.2 |
| (c) $\sigma = 0.20$ | 1600 | 0.441 | 60 | 11 | 0.818 | 71.9 |
| | 3200 | 0.440 | 60 | 12 | 1.30 | 455 |
| | 6400 | 0.440 | 61 | – | 2.98 | – |
| | 12 800 | 0.440 | 60 | – | 6.65 | – |

**Table 3**
Achievable accuracies of Gram matrix-based and randomized CTD-ID methods for CTD (5.4) with $\sigma = 0.20$ and $d = 5$.

| $r$ | $k_G$ | $\|\epsilon_G\|_s$ | $k_{ID}^{(1)}$ | $\|\epsilon_{ID}^{(1)}\|_s$ | $k_{ID}^{(2)}$ | $\|\epsilon_{ID}^{(2)}\|_s$ | $k_{ID}^{(3)}$ | $\|\epsilon_{ID}^{(3)}\|_s$ |
|---|---|---|---|---|---|---|---|---|
| 200 | 50 | 0.7e−8 | 50 | 0.5e−8 | 75 | 0.2e−9 | 100 | 0.5e−11 |
| 400 | 50 | 0.9e−8 | 50 | 2.1e−8 | 75 | 0.3e−9 | 100 | 0.7e−11 |
| 800 | 52 | 2.3e−8 | 52 | 2.3e−8 | 77 | 1.5e−9 | 102 | 3.1e−11 |
| 1600 | 54 | 0.7e−8 | 54 | 4.1e−8 | 79 | 0.7e−9 | 104 | 2.8e−11 |
| 3200 | 55 | 0.3e−8 | 55 | 0.8e−8 | 80 | 0.6e−9 | 105 | 4.4e−11 |

### 5.2. The tensor ID within convergent, self-correcting Schulz iteration

We present three examples of using the tensor ID within the SGTI approach in order to accelerate reduction of separation rank. These examples were originally presented in [16]. We consider the quadratically convergent, self-correcting Schulz iteration [55], given by

$$\begin{aligned} \mathbb{X}_{n+1} &= 2\mathbb{X}_n - \mathbb{X}_n\mathbb{B}\mathbb{X}_n, \\ \mathbb{X}_0 &= \alpha\mathbb{B}^*, \end{aligned} \tag{5.5}$$

where $\alpha$ is chosen so that the initial error $\mathbb{E}_0$ satisfies $\|\mathbb{E}_0\| = \|\mathbb{I} - \mathbb{X}_0\mathbb{B}\| < 1$. In these examples, the operator $\mathbb{B}$ is a preconditioned elliptic operator whose inverse corresponds to the Green's function of a Poisson equation (see more details below). Within each iteration, we first form the quantity $2\mathbb{I} - \mathbb{B}\mathbb{X}_n$ which we then left-multiply by $\mathbb{X}_n$ to obtain $\mathbb{X}_{n+1}$. Both of these operations significantly increase the separation rank and require a reduction step.

The reduction step, which would typically be performed using ALS, is instead performed with the randomized projection tensor ID Algorithm 3. Only after each complete iteration, when the separation rank has been reduced as much as possible via the tensor ID, is ALS invoked to further refine the approximation. In doing so, we avoid using ALS in the usual manner, i.e., to achieve a certain accuracy of approximation. Instead, its role is limited to reducing the dynamic range of the $s$-values (i.e., avoiding near cancellation of terms with large $s$-values) by running the algorithm for only a fixed (small) number of iterations. The reduction errors of the tensor ID and several ALS iterations are then corrected by the next Schulz iteration.

#### 5.2.1. Inverse operator for the Poisson equation

As the first example, we consider the periodic, constant coefficient Poisson equation. In this case we know that the Green's function has an efficient separated representation, see e.g., [11], and we want to demonstrate that we can approximate the Green's function starting with the differential operators in

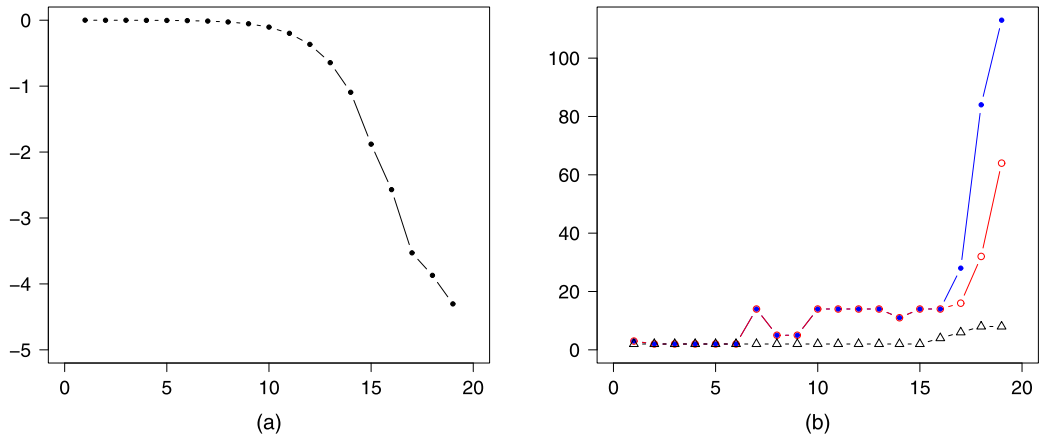


**Fig. 3.** Results for Example 5.2.1. Schulz error $\|\mathbb{E}_n\| = \frac{1}{2}(\|\mathbb{I} - \mathbb{X}_n\mathbb{B}\| + \|\mathbb{I} - \mathbb{B}\mathbb{X}_n\|)/\|I\|$ per iteration $n = 1, 2, 3, \ldots$ for constructing the Green's function for (5.6) and separation rank of Schulz iterate $\mathbb{X}_n$ before (dots) and after (circles) reduction by tensor ID. The triangles show the separation rank after applying ALS (with a fixed number of iterations) to the result of tensor ID.

$$\begin{aligned} -\Delta u(\mathbf{x}) &= f(\mathbf{x}), \quad \mathbf{x} \in (0,1)^3, \\ u(0,y,z) &= u(0,y,z), \\ u(x,0,z) &= u(x,1,z), \\ u(x,y,0) &= u(x,y,1), \end{aligned} \tag{5.6}$$

where

$$\Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \frac{\partial^2}{\partial x_3^2}.$$

We use eighth-order finite differences to discretize the second derivative in each direction to obtain the $512 \times 512$ matrix $A$, leading to the operator with separation rank $r = 3$,

$$\mathbb{A} = A \otimes I \otimes I + I \otimes A \otimes I + I \otimes I \otimes A$$

where $I$ denotes the identity matrix. We represent the matrix $A$ in a wavelet basis to ensure that both the operator and its inverse are sparse [9,10,7,8]. In a wavelet basis, the second derivative operator has a diagonal preconditioner, $P$ (see, e.g., [8]), such that the condition number of $PAP$ is $\mathcal{O}(1)$. Applying such preconditioner in dimension $d = 3$ results in the well-conditioned operator

$$\mathbb{B} = (PAP) \otimes I \otimes I + I \otimes (PAP) \otimes I + I \otimes I \otimes (PAP).$$

Since the problem (5.6) is periodic, the operators $\mathbb{A}$ and $\mathbb{B}$ have a one-dimensional null space spanned by a constant. This necessitates the use of the one dimensional projector within the Schulz iteration in order to avoid accumulation of the error in the null space. The results of this computation are shown in Fig. 3.

*5.2.2. Variable coefficient elliptic operator in dimension $d = 10$*

Next we consider the 10-dimensional PDE on the unit cube,

$$-\nabla \cdot \big(a(\mathbf{x})\nabla u(\mathbf{x})\big) = f(\mathbf{x}), \quad \mathbf{x} \in (0,1)^{10}, \tag{5.7}$$

with periodic boundary conditions and where

$$a(\mathbf{x}) = 1 - 0.9\exp\big(-3 \times 10^3 \cdot (\mathbf{x} - 0.5)^2\big). \tag{5.8}$$

In this case we reformulate the problem by using the constant coefficient Green's function to convert (5.7) into an integral equation, i.e., the constant coefficient Green's function is used as a preconditioner. By separating the constant from the variable terms in (5.7), the discretized elliptic operator $\mathbb{A}$ can be split into constant and variable parts,

$$\mathbb{A} = \mathbb{A}_c + \mathbb{A}_v. \tag{5.9}$$

Applying the discretized, constant coefficient Green's function $\mathbb{G}_c$ to $\mathbb{A}$, we obtain

$$\mathbb{B} = \mathbb{G}_c(\mathbb{A}_c + \mathbb{A}_v) = \mathbb{I} + \mathbb{G}_c\mathbb{A}_v. \tag{5.10}$$

The variable coefficient Green's function $\mathbb{G}$ of Eq. (5.7) can now be constructed by computing $\mathbb{B}^{-1}$ via Schulz iteration and setting $\mathbb{G} = \mathbb{B}^{-1}\mathbb{G}_c$.

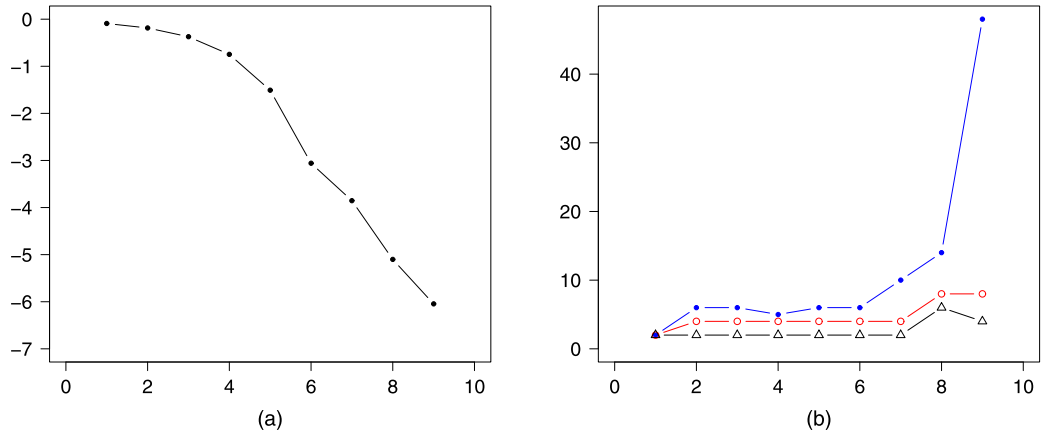


**Fig. 4.** Results for Example 5.2.2. Schulz error $\|\mathbb{E}_n\| = \frac{1}{2}(\|\mathbb{I}-\mathbb{X}_n\mathbb{B}\| + \|\mathbb{I}-\mathbb{B}\mathbb{X}_n\|)/\|I\|$ per iteration $n = 1, 2, 3, \ldots$ on $\log_{10}$ scale and separation rank of Schulz iterate before reduction (dots) and after (circles) tensor ID. The triangles show the separation rank after applying ALS (with a fixed number of iterations) to the result of tensor ID.

In this example, second order staggered finite differences are used to discretize the derivative operators in each direction at 128 equispaced points, leading to the elliptical operator $\mathbb{A}$ with separation rank 20. Upon applying the constant coefficient Green's function $\mathbb{G}_c$ and truncating terms with small $s$-values, the preconditioned operator $\mathbb{B}$ has nominal separation rank 271. To 6 digits of relative accuracy, however, the separation rank of $\mathbb{B}$ may be dramatically reduced (via, e.g., ALS iteration), and thus the iteration proceeds on an operator of separation rank only 5. As before, we represent all operators in a wavelet basis to ensure the sparsity of both $\mathbb{B}$ and $\mathbb{B}^{-1}$. Results are shown in Fig. 4.

*5.2.3. Stochastic PDE in dimension $d = 8$*

In our last example, we consider the 8-dimensional stochastic PDE,

$$-\nabla \cdot \big(a(\mathbf{x}, \omega)\nabla u(\mathbf{x}, \omega)\big) = f(\mathbf{x}, \omega), \quad \mathbf{x} \in (0, 1)^3,$$

where $\omega \in \Omega$ corresponds to probability space $(\Omega, \mathcal{F}, P)$ and the (spatially asymmetric) variable coefficient is given by

$$a(\mathbf{x}, \omega) = 1 + \sum_{l=1}^{5} 2^{-l} a_l(\omega) \sin(2l\pi x) \sin(2l\pi y) \sin\big(2(l+1)\pi z\big).$$

In other words, the variable coefficient is understood to have a deterministic spatial part with random coefficients, which we take in this example to be uniformly distributed, $a_l \sim U([-1, 1])$. The function $a$ may be thought of as a Karhunen–Loeve (KL) expansion of a random field with some (here, unspecified) covariance function. The resulting operator is thus 8-dimensional, with three spatial and five stochastic dimensions, the latter of which are discretized at Clenshaw–Curtis quadrature nodes,

$$a_l(\omega_n) = -\cos\left(\frac{\pi m}{M_{stoch} - 1}\right), \quad l = 1, \ldots, 5,\ m = 1, \ldots, M_{stoch} - 1.$$

As before, the spatial operator is discretized on a staggered grid at 128 points using second order finite differences, and we use $M_{stoch} = 16$ nodes in the stochastic directions. For the preconditioner, we use the constant coefficient Green's function in the spatial directions and identity matrices in the stochastic directions, leading to a preconditioned operator $\mathbb{B}$ with nominal separation rank of 1188. This number is reduced to 24 by truncating terms with small $s$-values and then applying ALS to the result. Results are shown in Fig. 5.

*5.3. A limitation of tensor ID: orthogonal decompositions*

Finally, we illustrate a limitation of using the tensor ID by constructing an example for which it is not expected to work at all. Consider the $d$-dimensional tensor

$$\mathcal{U} = \bigotimes_{j=1}^{d} \left( \sum_{l=1}^{L} \sigma_l \mathbf{u}_j^{(l)} \right) \tag{5.11}$$

where $\sigma_l = 1$ for all $l = 1, \ldots, L$, and

$$\left\langle \mathbf{u}_j^{(l)}, \mathbf{u}_j^{(m)} \right\rangle = \delta_{lm} \tag{5.12}$$

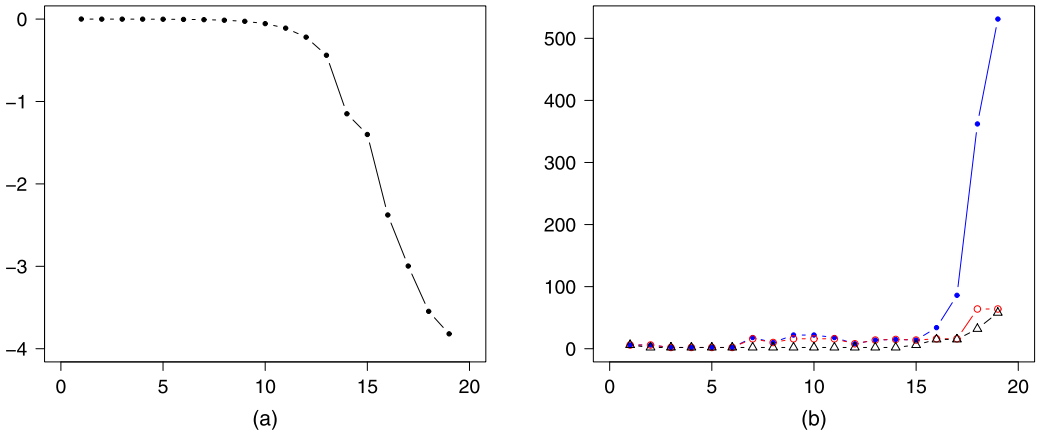


**Fig. 5.** Results for Example 5.2.3. Schulz error $\|\mathbb{E}_n\| = \frac{1}{2}(\|\mathbb{I}-\mathbb{X}_n\mathbb{B}\| + \|\mathbb{I}-\mathbb{B}\mathbb{X}_n\|)/\|I\|$ per iteration $n = 1, 2, 3, \ldots$ on $\log_{10}$ scale and separation rank of Schulz iterate before reduction (dots) and after (circles) tensor ID. The triangles show the separation rank after applying ALS (with a fixed number of iterations) to the result of tensor ID.

for $j = 1, \ldots, d$ and $l, m = 1, \ldots L$. Objects such as (5.11) appear quite frequently in various fields. For example, if we take the vectors $\mathbf{u}_j^{(l)}$ to correspond to orthogonal polynomials with respect to a specified probability measure, discretized at properly chosen quadrature nodes, $\mathcal{U}$ may be interpreted as a tensor order polynomial chaos expansion (PCE) [60].

Expanding the tensor product (5.11), the result is seen to have nominal separation rank $L^d$. By (5.12), all of the terms are mutually orthogonal and, thus, the columns of corresponding matrix $U$ in (1.11) are orthonormal. Consequently, the tensor ID cannot provide a reduction of the separation rank. However, if the $s$-values decay rapidly, we can truncate with controlled error by using Parseval's identity (cf. Example 5.1). In fact, truncation may in this case be viewed as a special instance of the tensor ID, with skeleton indices corresponding to the $k$ terms with $s$-values above the accuracy threshold.

## Acknowledgements

We would like to thank Martin Mohlenkamp (Ohio University) and Terry Haut (LANL) for making many useful suggestions to improve the manuscript.

## Appendix A. Supplementary material

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.jcp.2014.10.009.

## References

[1] E. Acar, B. Yener, Unsupervised multiway data analysis: a literature survey, IEEE Trans. Knowl. Data Eng. 21 (1) (2009) 6–20.
[2] D. Achlioptas, Database-friendly random projections: Johnson–Lindenstrauss with binary coins, J. Comput. Syst. Sci. 66 (4) (June 2003) 671–687.
[3] D. Achlioptas, F. McSherry, Fast computation of low-rank matrix approximations, J. ACM 54 (2) (2007) 9.
[4] N. Ailon, B. Chazelle, Approximate nearest neighbors and the fast Johnson–Lindenstrauss transform, in: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, ACM, 2006, pp. 557–563.
[5] B.W. Bader, T.G. Kolda, MATLAB Tensor Toolbox Version 2.5, available online, January 2012.
[6] J. Ballani, L. Grasedyck, M. Kluge, Black box approximation of tensors in hierarchical tucker format, Linear Algebra Appl. 438 (2) (2013) 639–657.
[7] G. Beylkin, On the representation of operators in bases of compactly supported wavelets, SIAM J. Numer. Anal. 29 (6) (1992) 1716–1740.
[8] G. Beylkin, On wavelet-based algorithms for solving differential equations, in: J.J. Benedetto, M.W. Frazier (Eds.), Wavelets: Mathematics and Applications, in: Stud. Adv. Math., CRC, Boca Raton, FL, 1994, pp. 449–466.
[9] G. Beylkin, R. Coifman, V. Rokhlin, Fast wavelet transforms and numerical algorithms, I, Commun. Pure Appl. Math. 44 (2) (August 1991) 141–183, Yale Univ. Technical Report YALEU/DCS/RR-696, August 1989.
[10] G. Beylkin, R. Coifman, V. Rokhlin, Wavelets in numerical analysis, in: Wavelets and Their Applications, Jones and Bartlett, Boston, MA, 1992, pp. 181–210.
[11] G. Beylkin, G. Fann, R.J. Harrison, C. Kurcz, L. Monzón, Multiresolution representation of operators with boundary conditions on simple domains, Appl. Comput. Harmon. Anal. 33 (2012) 109–139, http://dx.doi.org/10.1016/j.acha.2011.10.001.
[12] G. Beylkin, J. Garcke, M.J. Mohlenkamp, Multivariate regression and machine learning with sums of separable functions, SIAM J. Sci. Comput. 31 (3) (2009) 1840–1857.
[13] G. Beylkin, M.J. Mohlenkamp, Numerical operator calculus in higher dimensions, Proc. Natl. Acad. Sci. USA 99 (16) (August 2002) 10246–10251.
[14] G. Beylkin, M.J. Mohlenkamp, Algorithms for numerical analysis in high dimensions, SIAM J. Sci. Comput. 26 (6) (July 2005) 2133–2159.
[15] G. Beylkin, M.J. Mohlenkamp, F. Pérez, Approximating a wavefunction as an unconstrained sum of Slater determinants, J. Math. Phys. 49 (3) (2008) 032107.
[16] D. Biagioni, Numerical construction of Green's functions in high-dimensional elliptic problems with variable coefficients and analysis of renewable energy data via sparse and separable approximations, PhD thesis, University of Colorado, 2012.
[17] R. Bro, Parafac. Tutorial & applications, in: Special Issue 2nd Internet Conf. in Chemometrics (INCINC'96), Chemom. Intell. Lab. Syst. 38 (1997) 149–171, http://www.models.kvl.dk/users/rasmus/presentations/parafac_tutorial/paraf.htm.

[18] E.J. Candès, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies?, IEEE Trans. Inf. Theory 52 (12) (2006) 5406–5425.
[19] E.J. Candès, J. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, IEEE Trans. Inf. Theory 52 (2) (Feb. 2006) 489–509.
[20] J.D. Carroll, J.J. Chang, Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart–Young decomposition, Psychometrika 35 (1970) 283–320.
[21] H. Cheng, Z. Gimbutas, P.-G. Martinsson, V. Rokhlin, On the compression of low-rank matrices, SIAM J. Sci. Comput. 205 (1) (2005) 1389–1404.
[22] W.F. de la Vega, M. Karpinski, R. Kannan, S. Vempala, Tensor decomposition and approximation schemes for constraint satisfaction problems, in: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, STOC '05, ACM, New York, NY, USA, 2005, pp. 747–754.
[23] L. De Lathauwer, B. De Moor, J. Vandewalle, On the best rank-1 and rank-(r1, r2, . . . , rn) approximation of higher-order tensors, SIAM J. Matrix Anal. Appl. 21 (4) (2000) 1324–1342.
[24] V. de Silva, L.-H. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem, SIAM J. Matrix Anal. Appl. 30 (3) (2008) 1084–1127.
[25] J. Demmel, Accurate singular value decompositions of structured matrices, SIAM J. Matrix Anal. Appl. 21 (2) (1999) 562–580.
[26] P. Drineas, R. Kannan, Pass efficient algorithms for approximating large matrices, in: Proceedings of the Fourteenth Annual ACM–SIAM Symposium on Discrete Algorithms, SODA '03, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003, pp. 223–232.
[27] P. Drineas, M.W. Mahoney, A randomized algorithm for a tensor-based generalization of the singular value decomposition, Linear Algebra Appl. 420 (2–3) (2007) 553–571.
[28] A. Edelman, Eigenvalues and condition numbers of random matrices, PhD thesis, Massachusetts Institute of Technology, 1989.
[29] A. Frieze, R. Kannan, S. Vempala, Fast Monte-Carlo algorithms for finding low-rank approximations, J. ACM 51 (6) (November 2004) 1025–1041.
[30] S.A. Goreinov, E.E. Tyrtyshnikov, N.L. Zamarashkin, A theory of pseudoskeleton approximations, Linear Algebra Appl. 261 (1) (1997) 1–21.
[31] S.A. Goreinov, N.L. Zamarashkin, E.E. Tyrtyshnikov, Pseudo-skeleton approximations by matrices of maximal volume, Math. Notes 62 (4) (1997) 515–519.
[32] L. Grasedyck, D. Kressner, C. Tobler, A literature survey of low-rank tensor approximation techniques, CoRR arXiv:1302.7121, 2013.
[33] M. Gu, S.C. Eisenstat, Efficient algorithms for computing a strong rank-revealing qr factorization, SIAM J. Sci. Comput. 17 (4) (1996) 848–869.
[34] N. Halko, P.-G. Martinsson, J.A. Tropp, Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions, SIAM Rev. 53 (2) (2011) 217–288.
[35] R.A. Harshman, Foundations of the Parafac procedure: model and conditions for an "explanatory" multi-mode factor analysis, Working Papers in Phonetics 16, UCLA, 1970, http://publish.uwo.ca/~harshman/wpppfac0.pdf.
[36] C. Hillar, L.-H. Lim, Most tensor problems are NP hard, Technical report, arXiv:0911.1393, Nov. 2009.
[37] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, in: Conference in Modern Analysis and Probability, New Haven, CT, 1982, in: Contemp. Math., vol. 26, Amer. Math. Soc., Providence, RI, 1984, pp. 189–206.
[38] E. Kofidis, P.A. Regalia, On the best rank-1 approximation of higher-order supersymmetric tensors, SIAM J. Matrix Anal. Appl. 23 (3) (2001) 863–884.
[39] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500.
[40] T.G. Kolda, J.R. Mayo, Shifted power method for computing tensor eigenpairs, SIAM J. Matrix Anal. Appl. 32 (4) (October 2011) 1095–1124.
[41] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, M. Tygert, Randomized algorithms for the low-rank approximation of matrices, Proc. Natl. Acad. Sci. USA 104 (51) (2007) 20167–20172.
[42] L.-H. Lim, Singular values and eigenvalues of tensors: a variational approach, in: 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, 2005, pp. 129–132.
[43] M.W. Mahoney, Tensor-cur decompositions for tensor-based data, in: Proceedings of the 12th Annual ACM SIGKDD Conference, 2006, pp. 327–336.
[44] P.-G. Martinsson, V. Rokhlin, M. Tygert, A randomized algorithm for the approximation of matrices, Technical report, Yale CS research report YALEU/DCS/RR-1361, 2006.
[45] P.-G. Martinsson, V. Rokhlin, M. Tygert, A randomized algorithm for the approximation of matrices, Appl. Comput. Harmon. Anal. 30 (1) (2011) 47–68.
[46] MATLAB, Version 8.0.0 (R2012b), The MathWorks Inc., Natick, Massachusetts, 2012.
[47] Martin J. Mohlenkamp, Musings on multilinear fitting, Linear Algebra Appl. 438 (2011) 834–852.
[48] N.H. Nguyen, P. Drineas, T.D. Tran, Tensor sparsification via a bound on the spectral norm of random tensors, arXiv:1005.4732, 2010.
[49] I. Oseledets, E. Tyrtyshnikov, TT-cross approximation for multidimensional arrays, Linear Algebra Appl. 432 (1) (2010) 70–88.
[50] C.H. Papadimitriou, H. Tamaki, P. Raghavan, S. Vempala, Latent semantic indexing: a probabilistic analysis, in: Proceedings of the Seventeenth ACM SIGACT–SIGMOD–SIGART Symposium on Principles of Database Systems, PODS '98, ACM, New York, NY, USA, 1998, pp. 159–168.
[51] V.V. Prasolov, Problems and Theorems in Linear Algebra, Transl. Math. Monogr., vol. 134, Amer. Math. Soc., Providence, RI, 1994.
[52] H. Rauhut, R. Schneider, Z. Stojanac, Low-rank tensor recovery via iterative hard thresholding, in: Proceedings of the International Conference on Sampling Theory and Applications, 2013.
[53] M. Rudelson, R. Vershynin, Sampling from large matrices: an approach through geometric functional analysis, J. ACM 54 (4) (July 2007).
[54] T. Sarlos, Improved approximation algorithms for large matrices via random projections, in: 47th Annual IEEE Symposium on Foundations of Computer Science, FOCS '06, 2006, pp. 143–152.
[55] G. Schulz, Iterative Berechnung der reziproken Matrix, Z. Angew. Math. Mech. 13 (1933) 57–59.
[56] G. Tomasi, R. Bro, A comparison of algorithms for fitting the PARAFAC model, Comput. Stat. Data Anal. 50 (7) (2006) 1700–1734.
[57] C.E. Tsourakakis, Mach: fast randomized tensor decompositions, CoRR arXiv:0909.4969, 2009.
[58] E. Tyrtyshnikov, Incomplete cross approximation in the mosaic-skeleton method, Computing 64 (4) (2000) 367–380.
[59] E.E. Tyrtyshnikov, Mosaic-skeleton approximations, Calcolo 33 (1–2) (1996) 47–57.
[60] N. Wiener, The homogeneous chaos, Am. J. Math. 60 (4) (1938) 897–936.
[61] F. Woolfe, E. Liberty, V. Rokhlin, M. Tygert, A fast randomized algorithm for the approximation of matrices, Appl. Comput. Harmon. Anal. 25 (3) (2008) 335–366.
[62] T. Zhang, G.H. Golub, Rank-one approximation to high order tensors, SIAM J. Matrix Anal. Appl. 23 (2) (2001) 534–550.