SIAM J. MATRIX ANAL. APPL.
Vol. 41, No. 2, pp. 605–636

THE COMPUTATION OF LOW MULTILINEAR RANK APPROXIMATIONS OF TENSORS VIA POWER SCHEME AND RANDOM PROJECTION*

MAOLIN CHE†, YIMIN WEI‡, AND HONG YAN§

Abstract. This paper is devoted to the computation of low multilinear rank approximations of tensors. Combining the stretegy of power scheme, random projection, and singular value decomposition, we derive a three-stage randomized algorithm for the low multilinear rank approximation. Based on the singular values of sub-Gaussian matrices, we derive the error bound of the proposed algorithm with high probability. We illustrate the proposed algorithms via several numerical examples.

Key words. randomized algorithms, random projection, low multilinear rank approximation, random sub-Gaussian matrices, power scheme, singular values, singular value decomposition

AMS subject classifications. 15A18, 15A69, 65F15, 65F10

DOI. 10.1137/19M1237016

1. Introduction.

A tensor is an N-dimensional array of numbers denoted by script notation $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ with entries given by $a_{i_1i_2\dots i_N} \in \mathbb{R}$ for $i_n = 1,2,\dots,I_n$ and $n = 1,2,\dots,N$. Without loss of generality, we assume that $N = 3$. Our results using third-order tensors can be straightforwardly generalized to higher-order cases. Subtle differences will be mentioned when they exist. In this paper, we consider the low multilinear rank approximation of a third-order tensor $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$, which is defined as follows.

PROBLEM 1.1. *Suppose that $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$. For given three integers μ_1, μ_2, and μ_3, the goal is to require three columnwise orthogonal matrices $\mathbf{Q}_n \in \mathbb{R}^{I_n\times\mu_n}$ with $\mu_n \le I_n$, such that*

$$a_{i_1i_2i_3} \approx \sum_{j_1,j_2,j_3=1}^{I_1,I_2,I_3} a_{j_1j_2j_3} p_{1,i_1j_1} p_{2,i_2j_2} p_{3,i_3j_3},$$

where $\mathbf{P}_n = \mathbf{Q}_n\mathbf{Q}_n^\top \in \mathbb{R}^{I_n\times I_n}$ is an orthogonal projection.

For the solution of Problem 1.1, we have the following statements.

(a) Problem 1.1 is equivalent to the low multilinear rank approximation of tensors, which can be solved by a number of recently developed algorithms, such

*Received by the editors January 4, 2019; accepted for publication (in revised form) by M. Tygert February 28, 2020; published electronically April 29, 2020.
https://doi.org/10.1137/19M1237016
Funding: The work of the first author was supported by the National Natural Science Foundation of China grant 11901471. The work of the second author was supported by the National Natural Science Foundation of China grant 11771099 and the Innovation Program of Shanghai Municipal Education Committee. The work of the third author was supported by the Hong Kong Research Grants Council grant C1007-15G and the City University of Hong Kong grant 9610460.
†School of Economic Mathematics, Southwestern University of Finance and Economics, Chengdu, 611130, People's Republic of China (chncml@outlook.com, cheml@swufe.edu.cn).
‡Corresponding author. School of Mathematical Sciences and Key Laboratory of Mathematics for Nonlinear Sciences, Fudan University, Shanghai, 200433, People's Republic of China (ymwei@fudan.edu.cn).
§Department of Electrical Engineering, City University of Hong Kong, 83, Tat Chee Avenue, Kowloon, Hong Kong (h.yan@cityu.edu.hk).

as higher-order orthogonal iteration [15], the Newton–Grassmann method [20], the Riemannian trust-region method [30], the quasi-Newton method [53], semidefinite programming [42], and Lanczos-type iteration [23, 52].

(b) When the columns of each $\mathbf{Q}_n$ are extracted from the mode-n unfolding matrix $\mathbf{A}_{(n)}$, then, the solution of Problem 1.1 is called a *CUR*-type decomposition [5, 17, 24, 39, 44, 45] of tensor $\mathcal{A}$, which can be obtained using different versions of the cross approximation method.

(c) When the entries of $\mathcal{A}$ and $\mathbf{Q}_n$ are nonnegative, then Problem 1.1 is sometimes called a nonnegative Tucker decomposition [21, 70, 71, 73].

Randomized algorithms have recently been applied to matrix and tensor decompositions. Interested readers are referred to three surveys [18, 38, 68] for more details about the randomized algorithms for low rank matrix approximations. Some researchers focus on the computation of the CANDECOMP/PARAFAC decomposition and the CUR-type decomposition via the random projection strategy; e.g., see [3, 4, 17, 65].

Zhou, Cichocki, and Xie [72] proposed a distributed randomized Tucker decomposition for arbitrarily large tensors with relatively low multilinear ranks. Navasca and Pompey [43] directly applied the randomized algorithms for low rank matrix approximations [38] to compute the low multilinear rank approximation. Che and Wei [8, 9] designed adaptive randomized algorithms for computing the low multilinear rank approximation of tensors and approximate tensor train decomposition. More recently, Che, Wei, and Yan [10] presented an efficient randomized algorithm for the low multilinear rank approximation based on the random projection strategy and basic matrix decompositions.

In terms of CPU time, Tucker-SVD in [10] is faster than existing algorithms, such as tucker_ALS, mlsvd, lmlra_aca, Adap-Tucker, ran-Tucker, and mlsvd_rsi, for the computation of low multilinear rank approximations. However, in terms of relative error, Tucker-SVD is worse than these algorithms in some senses. Hence, the main aim of our work is to design a numerical algorithm for computing low multilinear rank approximations via random projection and power scheme, which is denoted by Tucker-pSVD. Tucker-pSVD needs more time than Tucker-SVD, but in terms of relative error, Tucker-pSVD is better than Tucker-SVD and competitive with tucker_ALS, mlsvd, and mlsvd_rsi. Tucker-pSVD requires $O(KI^4)$ operations, where K is the number of the power scheme. When $K = 1$, both Tucker-pSVD and mlsvd need $O(I^4)$ operations. However, when a faster but possibly less accurate eigenvalue decomposition to compute the factor matrices is used by applying mlsvd, mlsvd is faster than Tucker-pSVD, and slower than Tucker-SVD. In terms of CPU time, Tucker-pSVD is faster than mlsvd with the MATLAB SVD routine.

To be specific, the proposed algorithm can be divided into three stages. First, we obtain three temporary tensors $\widetilde{\mathcal{A}}_n$, given in (3.3), from $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$. Then, we compute three product tensors

$$\mathcal{B}_1 = \widetilde{\mathcal{A}}_1 \times_2 \mathbf{G}_{1,2} \times_3 \mathbf{G}_{1,3} \in \mathbb{R}^{I_1\times I_2\times I_3}, \quad \mathcal{B}_2 = \widetilde{\mathcal{A}}_2 \times_2 \mathbf{G}_{2,1} \times_3 \mathbf{G}_{2,3} \in \mathbb{R}^{I_2\times I_1\times I_3},$$
$$\mathcal{B}_3 = \widetilde{\mathcal{A}}_3 \times_2 \mathbf{G}_{3,1} \times_3 \mathbf{G}_{3,2} \in \mathbb{R}^{I_3\times I_1\times I_2},$$

where the entries of $\mathbf{G}_{n,m} \in \mathbb{R}^{L_m\times I_m}$ are independent and identically distributed (i.i.d.) Gaussian random variables of zero mean and unit variance, with $m, n = 1, 2, 3$ and $m \neq n$. Finally, for each n, an approximate basis of mode-n unfolding of $\mathcal{A}$ is computed by applying the SVD to mode-1 unfolding of $\mathcal{B}_n$. This algorithm can be viewed as an improvement of Tucker-SVD in [10], and a generalization of the work in [48] from matrices to tensors. Numerical examples illustrate that the proposed algorithm is efficient for the low multilinear rank approximation of a tensor with any given multilinear rank.

1.1. Notations and organizations. Throughout this paper, I, J, and N will be reserved to denote the index upper bounds, unless stated otherwise. We use lowercase letters $x, u, v, \dots$ for scalars, lowercase bold letters $\mathbf{x}, \mathbf{u}, \mathbf{v}, \dots$ for vectors, bold capital letters $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$ for matrices, and calligraphic letters $\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$ for higher-order tensors. This notation is consistently used for lower-order parts of a given structure. For example, the entry with row index i and column index j in a matrix $\mathbf{A}$, i.e., $(\mathbf{A})_{ij}$, is symbolized by a_{ij} (also $(\mathbf{x})_i = x_i$ and $(\mathcal{A})_{i_1 i_2 i_3} = a_{i_1 i_2 i_3}$).

We use $\|\mathbf{x}\|_2$ and $\mathbf{x}^\top$ to denote the 2-norm and the transpose of the vector $\mathbf{x} \in \mathbb{R}^I$, respectively. $\mathbf{0}$ denotes the zero vector in $\mathbb{R}^I$. We use $\mathbf{A} \otimes \mathbf{B}$ to denote the Kronecker product [22] of matrices $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{K \times L}$. $\mathbf{A}^\dagger$ represents the Moore–Penrose inverse [22, 67] of $\mathbf{A} \in \mathbb{R}^{I \times J}$. $\mathbb{S}_I$ is a symmetric group of degree I on the set $\{1, 2, \dots, I\}$.

The rest of our paper is organized as follows. Some basic results are introduced in section 2, such as, basic operations and higher-order SVDs of tensors and singular values of random matrices. Two three-stage randomized algorithms for the low multilinear rank approximation are presented in section 3. In this section, we also analyze probabilistic error bounds and computational complexity of these two algorithms. The probabilistic error bounds are proved in section 4. We illustrate our algorithms via numerical examples in section 5. We conclude this paper and discuss future research topics in section 6.

2. Preliminaries. We review the basic notations and concepts involving tensors used in this paper. The *mode-n product* [13, 27, 31] of a real tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ by a matrix $\mathbf{B} \in \mathbb{R}^{J_n \times I_n}$, denoted by $\mathcal{C} = \mathcal{A} \times_n \mathbf{B}$, is

$$
\begin{aligned}
n = 1:\ c_{j_1 i_2 i_3} &= \sum_{i_1=1}^{I_1} a_{i_1 i_2 i_3} b_{j_1 i_1},\\
n = 2:\ c_{i_1 j_2 i_3} &= \sum_{i_2=1}^{I_2} a_{i_1 i_2 i_3} b_{j_2 i_2},\\
n = 3:\ c_{i_1 i_2 j_3} &= \sum_{i_3=1}^{I_3} a_{i_1 i_2 i_3} b_{j_3 i_3}.
\end{aligned}
$$

For any given tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and three matrices $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{G} \in \mathbb{R}^{J_m \times I_m}$, and $\mathbf{H} \in \mathbb{R}^{J_n' \times J_n}$, one has [31]

$$
(\mathcal{A} \times_n \mathbf{F}) \times_m \mathbf{G} = (\mathcal{A} \times_m \mathbf{G}) \times_n \mathbf{F} = \mathcal{A} \times_n \mathbf{F} \times_m \mathbf{G}, \quad (\mathcal{A} \times_n \mathbf{F}) \times_n \mathbf{H} = \mathcal{A} \times_n (\mathbf{H} \cdot \mathbf{F})
$$

with $m \neq n = 1, 2, 3$, where "$\cdot$" represents the multiplication of two matrices with appropriate sizes.

Scalar products and the Frobenius norm of a tensor are extensions of the well-known definitions, from a matrix to a tensor of an arbitrary order [14, 31]. Suppose that two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the *Frobenius norm* of a tensor $\mathcal{A}$ is given by $\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$, and the scalar product $\langle \mathcal{A}, \mathcal{B} \rangle$ is defined as [14]

$$
\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1, i_2, i_3 = 1}^{I_1, I_2, I_3} a_{i_1 i_2 i_3} b_{i_1 i_2 i_3}.
$$

Generally speaking, the mode-n unfolding matrix of a third-order tensor can be understood as the process of the construction of a matrix containing all the mode-n vectors of the tensor. The order of the columns is not unique and the unfolding matrix of $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, denoted by $\mathbf{A}_{(n)}$, arranges the mode-n fibers into columns of this matrix. More specifically, a tensor element (i_1, i_2, i_3) maps on a matrix element (i_n, j), where

$$n = 1: \ j = i_2 + (i_3 - 1)I_2; \quad n = 2: \ j = i_1 + (i_3 - 1)I_1; \quad n = 3: \ j = i_1 + (i_2 - 1)I_1.$$

We recapitulate the mode-$(1,2)$ product [1, 12] (called *tensor-tensor product*) of two tensors $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and $\mathcal{B} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$ with common modes $I_1 = J_2$ that produces an order-4 tensor $\mathcal{C} = \mathcal{A} \times_1^2 \mathcal{B} \in \mathbb{R}^{I_2 \times I_3 \times J_1 \times J_3}$, where its entries are given by

$$c_{i_2 i_3 j_1 j_3} = \sum_{i_1=1}^{I_1} a_{i_1 i_2 i_3} b_{j_1 i_1 j_3} = \sum_{j_2=1}^{J_2} a_{j_2 i_2 i_3} b_{j_1 j_2 j_3}.$$

Similarly, for given $m \in \{1,2,3\}$ and $n \in \{1,2,3\}$, we can also define the mode-(m,n) product of $\mathcal{A}$ and $\mathcal{B}$ with $I_m = J_n$, which is denoted as $\mathcal{A} \times_m^n \mathcal{B}$.

2.1. Tucker decomposition and low multilinear rank approximations. A *Tucker decomposition* [60] of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is defined as

$$\mathcal{A} \approx \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}, \tag{2.1}$$

where $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ are called the *mode-n factor matrices* and $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ is called the *core tensor* of the decomposition.

The Tucker decomposition is closely related to the mode-n unfolding matrix $\mathbf{A}_{(n)}$ with $n = 1, 2, 3$. In particular, the relation (2.1) implies

$$\begin{aligned}
\mathbf{A}_{(1)} &\approx \mathbf{U}^{(1)} \mathbf{G}_{(1)} (\mathbf{U}^{(3)} \otimes \mathbf{U}^{(2)})^\top, \\
\mathbf{A}_{(2)} &\approx \mathbf{U}^{(2)} \mathbf{G}_{(2)} (\mathbf{U}^{(3)} \otimes \mathbf{U}^{(1)})^\top, \\
\mathbf{A}_{(3)} &\approx \mathbf{U}^{(3)} \mathbf{G}_{(3)} (\mathbf{U}^{(2)} \otimes \mathbf{U}^{(1)})^\top.
\end{aligned}$$

It follows that the rank of the approximations to $\mathbf{A}_{(n)}$ in the above equations is less than or equal to R_n, as the mode-n factor $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ at most has rank R_n. This motivates us to define the multilinear rank of $\mathcal{A}$ as the tuple

$$\{R_1, R_2, R_3\}, \quad \text{where the rank of } \mathbf{A}_{(n)} \text{ is equal to } R_n.$$

By applying the SVD to $\mathbf{A}_{(n)}$ with $n = 1, 2, 3$, we obtain a special form of the Tucker decomposition of a given tensor, which is referred to as the higher-order SVD (HOSVD) [14]. The HOSVD has the following properties: all factor matrices are columnwise orthogonal and the core tensor has the properties of all-orthogonality and ordering. As shown in [14], the suboptimality of HOSVD can be measured by the small singular values of all mode unfoldings. The HOSVD has many applications, such as, handwritten digit classification [51], multilinear subspace analysis [61, 62, 63, 64], item recommendation [58], compression of aerodynamic databases [36, 37].

When $R_n < \text{rank}(\mathbf{A}_{(n)})$ for one or more n, the decomposition is called the *truncated HOSVD* (T-HOSVD). The T-HOSVD is not optimal in terms of giving the best fitting as measured by the Frobenius norm of the difference [15, 16], i.e., the matrix Eckart–Young–Mirsky theorem does not hold for tensors with $N \geq 3$, but it is a good starting point for an iterative ALS (alternating least squares) algorithm. Note that comparison with the T-HOSVD, the sequentially truncated HOSVD [61] (ST-HOSVD) requires fewer flops and satisfies a better error bound (see [27]). With respect to the Frobenius norm of tensors, the low multilinear rank approximation of $\mathcal{A}$ can be rewritten as the optimization problem

$$\begin{aligned} \min \quad & \|\mathcal{A} - \mathcal{G} \times_1 \mathbf{Q}_1 \times_2 \mathbf{Q}_2 \times_3 \mathbf{Q}_3\|_F^2, \\ \text{subject to} \quad & \mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}, \\ & \mathbf{Q}_n \in \mathbb{R}^{I_n \times R_n} \text{ is columnwise orthogonal,} \end{aligned}$$

which is equivalent to the following maximization problem

$$\begin{aligned} \max \quad & \left\|\mathcal{A} \times_1 \mathbf{Q}_1^\top \times_2 \mathbf{Q}_2^\top \times_3 \mathbf{Q}_3^\top\right\|_F^2, \\ \text{subject to} \quad & \mathbf{Q}_n \in \mathbb{R}^{I_n \times R_n} \text{ is columnwise orthogonal.} \end{aligned}$$

If $\mathbf{Q}_n^*$ is a solution of the above maximization problem, then we call $\mathcal{A}\times_1\mathbf{P}_1\times_2\mathbf{P}_2\times_3\mathbf{P}_3$ a *low multilinear rank approximation* of $\mathcal{A}$, where $\mathbf{P}_n = \mathbf{Q}_n^*(\mathbf{Q}_n^*)^\top$.

2.2. Singular values of random matrices. We first introduce the definition of the sub-Gaussian random variable. Sub-Gaussian random variables are an important class of random variables that have strong tail decay properties.

DEFINITION 2.1 (see [54, Definition 3.2]). *A real valued random variable X is called sub-Gaussian if there exists a $b > 0$ such that for all $t > 0$ we have $\mathbf{E}(e^{tX}) \leq e^{b^2t^2/2}$. A random variable X is centered if $\mathbf{E}(X) = 0$.*

We review several results adapted from [34, 49] about random matrices whose entries are sub-Gaussian random variables. We focus on the case where $\mathbf{A}$ is an $I \times J$ matrix with $J > (1 + 1/\ln(I))I$. Similar results can be found in [35] for the square and almost square matrices.

DEFINITION 2.2. *Assume that $\mu \geq 1$, $a_1 > 0$, and $a_2 > 0$. The set $\mathbb{A}(\mu, a_1, a_2, I, J)$ consists of all $I \times J$ random matrices $\mathbf{A}$ whose entries are the centered i.i.d. real valued random variables satisfying the following conditions:* (a) *moments:* $\mathbf{E}(|a_{ij}|^3) \leq \mu^3$*;* (b) *norm:* $\mathbf{P}(\|\mathbf{A}\|_2 > a_1\sqrt{J}) \leq e^{-a_2 J}$*; and* (c) *variance:* $\mathbf{E}(|a_{ij}|^2) \leq 1$.

It is shown in [34] that if $\mathbf{A}$ is sub-Gaussian, then $\mathbf{A} \in \mathbb{A}(\mu, a_1, a_2, I, J)$. For a Gaussian matrix with zero mean and unit variance, we have $\mu = (4/\sqrt{2\pi})^{1/3}$. Theorems 2.3 and 2.4 are taken from [34, section 2].

THEOREM 2.3 (see [34]). *Suppose that $\mathbf{A} \in \mathbb{R}^{I \times J}$ is sub-Gaussian with $I \leq J$, $\mu \geq 1$, and $a_2 > 0$. Then for $a_1 = 6\mu\sqrt{a_2 + 4}$, we have $\mathbf{P}(\|\mathbf{A}\|_2 > a_1\sqrt{J}) \leq e^{-a_2 J}$.*

Theorem 2.3 provides an upper bound for the largest singular value that depends on the desired probability. Theorem 2.4 is used to bound from the upper below the smallest singular value of a random sub-Gaussian matrix.

THEOREM 2.4 (see [34]). *Let $\mu \geq 1$, $a_1 > 0$, and $a_2 > 0$. Suppose that $\mathbf{A} \in \mathbb{A}(\mu, a_1, a_2, I, J)$ with $J > (1 + 1/\ln(I))I$. Then, there exist positive constants c_1 and c_2 such that $\mathbf{P}(\sigma_I(\mathbf{A}) \leq c_1\sqrt{J}) \leq e^{-c_2 J}$.*

3. The proposed algorithm and its modification. In this section, by using the power scheme, random projection, and SVD, we present a randomized algorithm for the low multilinear rank approximations of tensors, summarized in Algorithm 3.1. By means of the strategy discussed in [61], a modification of Algorithm 3.1 is summarized in Algorithm 3.2.

3.1. Framework of the algorithm. By the means of the Frobenius norm of tensors, Problem 1.1 can be represented as the following problem.

PROBLEM 3.1. *Suppose that* $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ *and* ε *is a prescribed tolerance. For given three integers* μ_1, μ_2, *and* μ_3, *the goal is to require three columnwise orthogonal matrices* $\mathbf{Q}_n \in \mathbb{R}^{I_n \times \mu_n}$ *with* $\mu_n \le I_n$, *such that*

$$\left\|\mathcal{A} - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) \times_3 \left(\mathbf{Q}_3\mathbf{Q}_3^\top\right)\right\|_F \le \sqrt{3}\varepsilon.$$

Suppose that all the matrices $\mathbf{Q}_n \in \mathbb{R}^{I_n \times \mu_n}$ are the solution for Problem 3.1, As shown in [10, 50, 61], we have

$$\begin{aligned}
&\mathcal{A} - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) \times_3 \left(\mathbf{Q}_3\mathbf{Q}_3^\top\right)\\
&\quad = \mathcal{A} - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) + \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right)\\
&\qquad + \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) \times_3 \left(\mathbf{Q}_3\mathbf{Q}_3^\top\right).
\end{aligned} \tag{3.1}$$

According to (3.1), we have

$$\left\|\mathcal{A} - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) \times_3 \left(\mathbf{Q}_3\mathbf{Q}_3^\top\right)\right\|_F^2 \le \sum_{n=1}^{3} \left\|\mathcal{A} - \mathcal{A} \times_n \left(\mathbf{Q}_n\mathbf{Q}_n^\top\right)\right\|_F^2.$$

The result relies on the orthogonal projector in the Frobenius norm [61], i.e., for any $n = 1, 2, 3$,

$$\|\mathcal{A}\|_F^2 = \left\|\mathcal{A} \times_n \left(\mathbf{Q}_n\mathbf{Q}_n^\top\right)\right\|_F^2 + \left\|\mathcal{A} \times_n \left(\mathbf{I}_{I_n} - \mathbf{Q}_n\mathbf{Q}_n^\top\right)\right\|_F^2,$$

and the fact that $\|\mathbf{AP}\|_F \le \|\mathbf{A}\|_F\|\mathbf{P}\|_2 = \|\mathbf{A}\|_F$ with $\mathbf{A} \in \mathbb{R}^{I \times J}$, where the *orthogonal projection* $\mathbf{P}$ satisfies [22]

$$\mathbf{P}^2 = \mathbf{P}, \quad \mathbf{P}^\top = \mathbf{P}, \quad \mathbf{P} \in \mathbb{R}^{J \times J}.$$

In order to solve Problem 3.1, for each n, we need to find a columnwise orthogonal matrix $Q_n \in \mathbb{R}^{I_n \times \mu_n}$ such that

$$\left\|\mathcal{A} - \mathcal{A} \times_n \left(\mathbf{Q}_n\mathbf{Q}_n^\top\right)\right\|_F \le \varepsilon.$$

Without loss of generality, suppose that $n = 1$. The first stage of Algorithm 3.1 is to generate another tensor $\widetilde{\mathcal{A}}_1$ as

$$\begin{aligned}
\widetilde{\mathcal{A}}_1 &= \mathcal{A} \times_{2,3}^{1,2} \Bigg[\underbrace{(\mathcal{A} \times_1^1 \mathcal{A}) \times_{3,4}^{1,2} (\mathcal{A} \times_1^1 \mathcal{A}) \times_{3,4}^{1,2} \cdots \times_{3,4}^{1,2} (\mathcal{A} \times_1^1 \mathcal{A})}_{K}\Bigg]\\
&= \mathcal{A} \times_{2,3}^{1,2} \underbrace{(\mathcal{A} \times_1^1 \mathcal{A}) \times_{2,3}^{1,2} (\mathcal{A} \times_1^1 \mathcal{A}) \times_{2,3}^{1,2} \cdots \times_{2,3}^{1,2} (\mathcal{A} \times_1^1 \mathcal{A})}_{K}\\
&= \underbrace{(\mathcal{A} \times_{2,3}^{2,3} \mathcal{A}) \times_1^1 (\mathcal{A} \times_{2,3}^{2,3} \mathcal{A}) \times_1^1 \cdots \times_1^1 (\mathcal{A} \times_{2,3}^{2,3} \mathcal{A})}_{K} \times_1^1 \mathcal{A}
\end{aligned} \tag{3.2}$$

with a fixed positive integer K. We have the following remarks for (3.2).

Remark 3.1. Suppose that $K = 1$. For the case of $n = 1$, there exist two ways to generate $\widetilde{\mathcal{A}}_1$ as follows:

(a) We set $\mathcal{B} = \mathcal{A} \times_1^1 \mathcal{A} \in \mathbb{R}^{I_2 \times I_3 \times I_2 \times I_3}$ and the entries of $\widetilde{\mathcal{A}}_1 = \mathcal{A} \times_{2,3}^{1,2} \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ are given by

$$\widetilde{\mathcal{A}}_1(i_1, i_2, i_3) = \sum_{j_2, j_3 = 1}^{I_2, I_3} a_{i_1 j_2 j_3} b_{j_2 j_3 i_2 i_3}.$$

For this way, to compute $\widetilde{\mathcal{A}}_1$ requires $O(I^5)$ operations with $I_1 = I_2 = I_3 = I$.

(b) We set $\mathcal{B} = \mathcal{A} \times_{2,3}^{2,3} \mathcal{A} \in \mathbb{R}^{I_1 \times I_1}$ and the entries of $\widetilde{\mathcal{A}}_1 = \mathcal{B} \times_1^1 \mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ are given by

$$\widetilde{\mathcal{A}}_1(i_1, i_2, i_3) = \sum_{j_1 = 1}^{I_1} b_{i_1 j_1} a_{j_1 i_2 i_3}.$$

For this way, to compute $\widetilde{\mathcal{A}}_1$ requires $O(I^4)$ operations with $I_1 = I_2 = I_3 = I$.

Hence, for a given positive integer $K \geq 1$, we use the second way to generate $\widetilde{\mathcal{A}}_n$ for all $n = 1, 2, 3$.

In the second stage, we project the mode-1 unfolding of $\widetilde{\mathcal{A}}_1$ on the Kronecker product of standard Gaussian matrices. The result matrix captures most of the range of the mode-1 unfolding of $\widetilde{\mathcal{A}}_1$. The third stage is to compute a basis of this matrix by the SVD. In this paper, we set $T = 10$ and $K = 1$.

Remark 3.2. At step 4 in Algorithm 3.1, the matrices $\mathbf{Q}$ and $\mathbf{S}_n$ can be constructed from the SVD of $\mathbf{B}_{n,(1)}$ [41, Lemma 3.5].

In Algorithm 3.1, we use the interpretation of $O(\cdot)$ to refer to the class of functions whose growth is bounded above and below up to a constant. The symbol $\Delta_{\mu_n+1}(\mathbf{A}_{(n)})$ is defined as

$$\Delta_{\mu_n+1}(\mathbf{A}_{(n)}) = \sqrt{\sum_{i=\mu_n+1}^{I_n} \sigma_i(\mathbf{A}_{(n)})^2},$$

where $\sigma_i(\mathbf{A}_{(n)})$ is the ith singular value of the mode-n unfolding of $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ with $n = 1, 2, 3$.

Remark 3.3. Note that $\|\mathcal{A} - \mathcal{A} \times_n (\mathbf{Q}_n \mathbf{Q}_n^\top)\|_F = \|\mathbf{A}_{(n)} - \mathbf{Q}_n \mathbf{Q}_n^\top \mathbf{A}_{(n)}\|_F$ with $n = 1, 2, 3$. Then from (3.2), the mode-1 unfolding of $\widetilde{\mathcal{A}}_1$ can be represented as

$$\widetilde{\mathbf{A}}_{1,(1)} = \mathbf{A}_{(1)} (\mathbf{A}_{(1)}^\top \mathbf{A}_{(1)})^K.$$

The matrix $\widetilde{\mathbf{A}}_{1,(1)}$ has the same singular vectors as the matrix $\mathbf{A}_{(1)}$, but its singular values satisfy

$$\sigma_i(\widetilde{\mathbf{A}}_{1,(1)}) = \sigma_i(\mathbf{A}_{(1)})^{2K+1}, \quad i = 1, 2, \ldots, \min\{I_1, I_2 I_3\}.$$

We can obtain the columnwise orthogonal matrix $\mathbf{Q}_1$ by directly using [38, Algorithm 4.3] to $\widetilde{\mathbf{A}}_{1,(1)}$. However, as shown in [8], this strategy requires a large amount of storage for generating the Gaussian random matrix.

Remark 3.4. As shown in Algorithm 3.1, for each n, we use the SVD to find a columnwise orthogonal matrix $\mathbf{Q}_n$ from $\mathbf{B}_{n,(1)}$ such that $\mathbf{Q}_n$ captures most of the range of $\mathbf{B}_{n,(1)}$. Hence our goal is to approximate the dominant subspace of $\mathbf{B}_{n,(1)}$ with $n = 1, 2, 3$. As shown in [40, 69], the approximate dominant subspace of $\mathbf{A} \in \mathbb{R}^{I \times J}$ with $I > J$ can be obtained by

$$\mathbf{A} \approx \mathbf{Q}\mathbf{B}, \quad \mathbf{B} = \mathbf{Q}^\dagger \mathbf{A} \in \mathbb{R}^{K \times J},$$

where $\mathbf{Q} \in \mathbb{R}^{I \times K}$ with $K < \min(I, J)$ has full column rank.

Algorithm 3.1 Power scheme + random projection + SVD.

Input: A tensor $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ to decompose, the desired multilinear rank $\{\mu_1,\mu_2,\mu_3\}$, the desired multilinear rank $\{\mu_1,\mu_2,\mu_3\}$, $L_1L_2 \geq \mu_3+T$, $L_1L_3 \geq \mu_2+T$, $L_2L_3 \geq \mu_1+T$ number of columns to use, and two positive integers K and T, where T is an oversampling parameter.

Output: Three columnwise orthogonal matrices $\mathbf{Q}_n \in \mathbb{R}^{I_n\times\mu_n}$ such that $\|\mathcal{A} \times_1 (\mathbf{Q}_1\mathbf{Q}_1^\top) \times_2 (\mathbf{Q}_2\mathbf{Q}_2^\top) \times_3 (\mathbf{Q}_3\mathbf{Q}_3^\top) - \mathcal{A}\|_F \leq \sum_{n=1}^3 O\left(\Delta_{\mu_n+1}(\mathbf{A}_{(n)})^{2K+1} + \Delta_{\mu_n+1}(\mathbf{A}_{(n)})\right)$.

1: Form six real matrices $\mathbf{G}_{n,m} \in \mathbb{R}^{L_m\times I_m}$ whose entries are i.i.d. Gaussian random variables of zero mean and unit variance, where $m,n=1,2,3$ and $m\neq n$.

2: Compute three product tensors

$$\mathcal{B}_1 = \widetilde{\mathcal{A}}_1 \times_2 \mathbf{G}_{1,2} \times_3 \mathbf{G}_{1,3}, \quad \mathcal{B}_2 = \widetilde{\mathcal{A}}_2 \times_2 \mathbf{G}_{2,1} \times_3 \mathbf{G}_{2,3}, \quad \mathcal{B}_3 = \widetilde{\mathcal{A}}_3 \times_2 \mathbf{G}_{3,1} \times_3 \mathbf{G}_{3,2}$$

with

$$\begin{aligned}
\widetilde{\mathcal{A}}_1 &= \mathcal{A} \times_{2,3}^{1,2} \underbrace{(\mathcal{A}\times_1^1\mathcal{A}) \times_{2,3}^{1,2} (\mathcal{A}\times_1^1\mathcal{A}) \times_{2,3}^{1,2} \cdots \times_{2,3}^{1,2} (\mathcal{A}\times_1^1\mathcal{A})}_{K} \in \mathbb{R}^{I_1\times I_2\times I_3},\\
\widetilde{\mathcal{A}}_2 &= \mathcal{A} \times_{1,3}^{1,2} \underbrace{(\mathcal{A}\times_2^2\mathcal{A}) \times_{2,3}^{1,2} (\mathcal{A}\times_2^2\mathcal{A}) \times_{2,3}^{1,2} \cdots \times_{2,3}^{1,2} (\mathcal{A}\times_2^2\mathcal{A})}_{K} \in \mathbb{R}^{I_2\times I_1\times I_3},\\
\widetilde{\mathcal{A}}_3 &= \mathcal{A} \times_{1,2}^{1,2} \underbrace{(\mathcal{A}\times_3^3\mathcal{A}) \times_{2,3}^{1,2} (\mathcal{A}\times_3^3\mathcal{A}) \times_{2,3}^{1,2} \cdots \times_{2,3}^{1,2} (\mathcal{A}\times_3^3\mathcal{A})}_{K} \in \mathbb{R}^{I_3\times I_1\times I_2}.
\end{aligned} \tag{3.3}$$

3: Form the mode-1 unfolding $\mathbf{B}_{n,(1)}$ of each tensor $\mathcal{B}_n$.

4: For each $\mathbf{B}_{n,(1)}$, find a real $I_n\times\mu_n$ matrix $\mathbf{Q}_n$, which is columnwise orthogonal, such that there exists a real $\mu_n \times \prod_{m=1,m\neq n}^3 L_m$ matrix $\mathbf{S}_n$ for which

$$\|\mathbf{Q}_n\mathbf{S}_n - \mathbf{B}_{n,(1)}\|_2 \leq \sigma_{\mu_n+1}(\mathbf{B}_{n,(1)}),$$

where $\sigma_{\mu_n+1}(\mathbf{B}_{n,(1)})$ is the (μ_n+1)st greatest singular value of $\mathbf{B}_{n,(1)}$.

5: Set $\mathbf{Q}_n := \mathbf{Q}_n(:,1:\mu_n)$ for all $n=1,2,3$.

When $\mathbf{A}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, then $\mathbf{Q}=\mathbf{U}(:,1:K)$ and $\mathbf{B}=(\mathbf{\Sigma}(1:K,1:K)\mathbf{V}(:,1:K))^\top$. When $\mathbf{AP}=\widetilde{\mathbf{Q}}\mathbf{R}$, then $\mathbf{Q}=\widetilde{\mathbf{Q}}(:,1:K)$ and $\mathbf{B}=(\mathbf{RP}^\top)(1:K,:)$. When $\mathbf{P}_1\mathbf{A}\mathbf{P}_2=\mathbf{LU}$, then $\mathbf{Q}=(\mathbf{P}_1^\top\mathbf{L})(:,1:K)$ and $\mathbf{B}=(\mathbf{U}\mathbf{P}_2^\top)(1:K,:)$. Hence, we can replace SVD used in Algorithm 3.1 by any algorithm, which can approximate the dominant subspace, such as the rank-revealing QR [6, 7, 26, 29, 47, 56] and rank-revealing LU [46]. This remark is also suitable for Algorithm 3.2.

Hence, when obtaining the error bound of $\|\mathcal{A}-\mathcal{A}\times_n(\mathbf{Q}_n\mathbf{Q}_n^\top)\|_F^2$, we present an error bound for Algorithm 3.1, summarized in the following theorem.

THEOREM 3.1. *Suppose that $I_1\leq I_2I_3$, $I_2\leq I_1I_3$, and $I_3\leq I_1I_2$. Let μ_1, L_2, and L_3 be integers such that $(1+1/\ln(\sqrt{\mu_1}))\sqrt{\mu_1} < L_2, L_3$ and $L_2L_3<\min(I_1,I_2I_3)$. Let μ_2, L_1, and L_3 be integers such that $(1+1/\ln(\sqrt{\mu_2}))\sqrt{\mu_2} < L_1, L_3$ and $L_1L_3<\min(I_1,I_2I_3)$. Let μ_3, L_1, and L_2 be integers such that $(1+1/\ln(\sqrt{\mu_1}))\sqrt{\mu_1} < L_1, L_2$ and $L_1L_2<\min(I_3,I_1I_2)$. Let $\sqrt{\mu_1}$, $\sqrt{\mu_2}$, and $\sqrt{\mu_3}$ be positive integers. For each n, we define a_n, a_n', c_{nm}, and c_{nm}' as in Theorems* 2.3 *and* 2.4 *with $m=1,2,3$ and $m\neq n$.*

For a given tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, columnwise orthogonal matrices $\mathbf{Q}_n$ are obtained by Algorithm 3.1. *Then*

$$\left\|\mathcal{A} - \mathcal{A} \times_1 \left(\mathbf{Q}_1\mathbf{Q}_1^\top\right) \times_2 \left(\mathbf{Q}_2\mathbf{Q}_2^\top\right) \times_3 \left(\mathbf{Q}_3\mathbf{Q}_3^\top\right)\right\|_F \leq 2\sum_{n=1}^{3}\left(C_n \Delta_{\mu_n+1}(\mathbf{A}_{(n)})^{2K+1} + \Delta_{\mu_n+1}(\mathbf{A}_{(n)})\right)$$

with probability at least

$$1 - \Big(e^{-c'_{12}L_2} + e^{-c'_{13}L_3} + e^{-c'_{21}L_1} + e^{-c'_{23}L_3} + e^{-c'_{31}L_1} + e^{-c'_{32}L_2} + e^{-a'_1 I_2 I_3} + e^{-a'_2 I_1 I_3} + e^{-a'_3 I_1 I_2}\Big),$$

where C_1, C_2, and C_3 are given by

$$C_1 = \frac{2}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\sqrt{\frac{a_1^2 I_2 I_3}{c_{12}^2 c_{13}^2 L_2 L_3}},$$

$$C_2 = \frac{2}{\sigma_{\mu_2}(\mathbf{A}_{(2)})^{2K}}\sqrt{\frac{a_2^2 I_1 I_3}{c_{21}^2 c_{23}^2 L_1 L_3}},$$

$$C_3 = \frac{2}{\sigma_{\mu_3}(\mathbf{A}_{(3)})^{2K}}\sqrt{\frac{a_3^2 I_1 I_2}{c_{31}^2 c_{32}^2 L_1 L_2}}.$$

Here K is a given positive integer.

For the case of $n = 1$, the assumptions $(1 + 1/\ln(\sqrt{\mu_1}))\sqrt{\mu_1} < L_2, L_3$ and $L_2L_3 < \min(I_1, I_2I_3)$ in Theorem 3.1 imply $\min(I_1, I_2I_3) > L_2L_3 > (1+1/\ln(\mu_1))\mu_1$. Hence, we set L_2L_3 as the smallest positive integer such that $L_2L_3 \geq \mu_1 + T$ and $\min(I_1, I_2I_3) > L_2L_3 > (1 + 1/\ln(\mu_1))\mu_1$. Let $M = \max(\mu_1 + T, (1 + 1/\ln(\mu_1))\mu_1)$. In practice, we set $L_2 = \text{ceil}(\sqrt{M})$ and $L_2 = \text{round}(\sqrt{M})$, where for $x \in \mathbb{R}$, $\text{ceil}(x)$ rounds the value of x to the nearest integer towards plus infinity and $\text{round}(x)$ rounds the value of x to the nearest integer.

3.2. Modification of Algorithm 3.1. By the strategy discussed in [61], a slight modification of Algorithm 3.1 is given in Algorithm 3.2. Based on (3.1) and the fact $\|\mathbf{A}\mathbf{Q}\|_F \leq \|\mathbf{A}\|_F$ for $A \in \mathbb{R}^{I\times J}$ and any columnwise orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{J\times R}$ $(R \leq J)$, the temporary tensor $\mathcal{C}$ in Algorithm 3.2 is updated for each n.

Remark 3.5. In Algorithms 3.1 and 3.2, we only need the μ_n leading left singular vectors of $\mathbf{B}_{n,(1)}$ with $n = 1, 2, 3$.

Note that the main difference between Algorithms 3.1 and 3.2 is that the temporary tensor $\mathcal{C}$ is updated after each n. We illustrate the difference via an example. The test tensor $\mathcal{A} \in \mathbb{R}^{I\times I\times I}$ is given in the Tucker form: $\mathcal{A} = \mathcal{S} \times_1 \mathbf{G}_1 \times_2 \mathbf{G}_2 \times_3 \mathbf{G}_3$, where the entries of $\mathcal{S} \in \mathbb{R}^{100\times100\times100}$ and $\mathbf{G}_n \in \mathbb{R}^{400\times100}$ $(n = 1, 2, 3)$ are i.i.d. Gaussian variables with zero mean and unit variance. Figure 1 shows that Algorithm 3.2 is more effective than Algorithm 3.1 for computing low multilinear rank approximations. Hence, Algorithm 3.2 is denoted by Tucker-pSVD.

Algorithm 3.2 A slight modification of Algorithm 3.1.

Input: A tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ to decompose, the desired multilinear rank $\{\mu_1, \mu_2, \mu_3\}$, the desired multilinear rank $\{\mu_1, \mu_2, \mu_3\}$, $L_1 L_2 \geq \mu_3 + T$, $L_1 L_3 \geq \mu_2 + T$, $L_2 L_3 \geq \mu_1 + T$ number of columns to use, and two positive integers K and T, where T is an oversampling parameter.

Output: Three columnwise orthogonal matrices $\mathbf{Q}_n \in \mathbb{R}^{I_n \times \mu_n}$ such that $\|\mathcal{A} \times_1 (\mathbf{Q}_1 \mathbf{Q}_1^\top) \times_2 (\mathbf{Q}_2 \mathbf{Q}_2^\top) \times_3 (\mathbf{Q}_3 \mathbf{Q}_3^\top) - \mathcal{A}\|_F \leq \sum_{n=1}^3 O(\Delta_{\mu_n+1}(\mathbf{A}_{(n)})^{2K+1} + \Delta_{\mu_n+1}(\mathbf{A}_{(n)}))$.

Set the temporary tensor: $\mathcal{C} = \mathcal{A}$.
for $n = 1, 2, 3$ **do**
  Form two real matrices $\mathbf{G}_{n,m} \in \mathbb{R}^{L_m \times I_m}$ whose entries are i.i.d. Gaussian random variables of zero mean and unit variance, where $m = 1, 2, 3$ and $m \neq n$.
  Compute the product tensor

$$\mathcal{B}_n \in \left\{ \widetilde{\mathcal{A}}_1 \times_2 \mathbf{G}_{1,2} \times_3 \mathbf{G}_{1,3}, \widetilde{\mathcal{A}}_2 \times_2 \mathbf{G}_{2,1} \times_3 \mathbf{G}_{2,3}, \widetilde{\mathcal{A}}_3 \times_2 \mathbf{G}_{3,1} \times_3 \mathbf{G}_{3,2} \right\}$$

  with

$$\widetilde{\mathcal{A}}_n = \mathcal{C} \times_{\{1,2,3\}-n}^{1,2} \underbrace{(\mathcal{C} \times_n^n \mathcal{C}) \times_{2,3}^{1,2} (\mathcal{C} \times_n^n \mathcal{C}) \times_{2,3}^{1,2} \cdots \times_{2,3}^{1,2} (\mathcal{C} \times_n^n \mathcal{C})}_{K}. \tag{3.4}$$

  Form the mode-1 unfolding $\mathbf{B}_{n,(1)}$ of each tensor $\mathcal{B}_n$.
  For each $\mathbf{B}_{n,(1)}$, find a real $I_n \times \mu_n$ matrix $\mathbf{Q}_n$, which is columnwise orthogonal, such that there exists a real $\mu_n \times \prod_{m=1, m\neq n}^3 L_m$ matrix $\mathbf{S}_n$ for which

$$\|\mathbf{Q}_n \mathbf{S}_n - \mathbf{B}_{n,(1)}\|_2 \leq \sigma_{\mu_n+1}(\mathbf{B}_{n,(1)}),$$

  where $\sigma_{\mu_n+1}(\mathbf{B}_{n,(1)})$ is the $(\mu_n + 1)$st greatest singular value of $\mathbf{B}_{n,(1)}$.
  Set $I_n = \mu_n$ and compute $\mathcal{C} = \mathcal{C} \times_n \mathbf{Q}_n^\top$.
end for

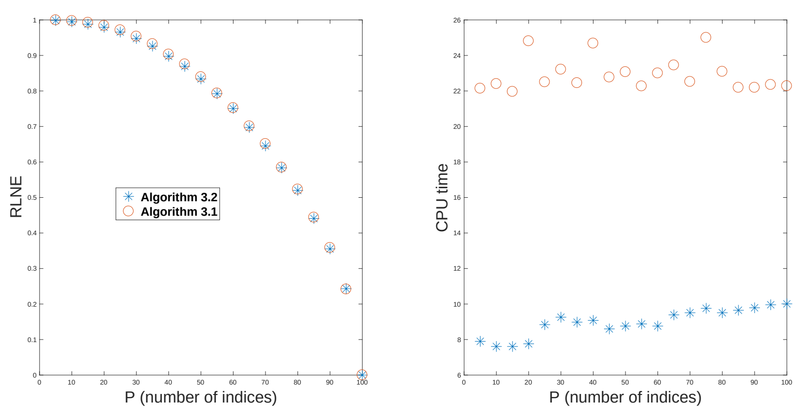

FIG. 1. *Numerical simulation results of applying Algorithms* 3.1 *and* 3.2 *to the tensor* $\mathcal{A}$ *with* $P = 5, 10, \ldots, 100$ *and* $I = 400$. *Note that RLNE in the left part is defined in* (5.1).

3.3. Computational complexity analysis. In this paper, for clarity, we assume that $I_1 = I_2 = I_3 = I$, $L_1 = L_2 = L_3 = L$, and $\mu_1 = \mu_2 = \mu_3 = \mu$ in complexity estimates, where μ_n is the number of columns of $\mathbf{Q}_n$. We can also assume that $I_1 \sim I_2 \sim I_3 \sim I$, $L_1 \sim L_2 \sim L_3 \sim L$, and $\mu_1 \sim \mu_2 \sim \mu_3 \sim \mu$ in complexity estimates [23, p. A2], where $I_n \sim I$ means $I_n = \alpha_n I$ for some constant α_n.

From part (b) in Remark 3.1, to compute the number of floating points operations in Algorithm 3.1, we evaluate the complexity of each step:

S1: Generating six standard Gaussian matrices requires $6IL$ operations.

S2: For all $n = 1, 2, 3$, computing three product tensors $\mathcal{B}_n$ needs $6KI^4 + 6(K - 1)I^3 + 6I^3L + 12I^2L^2$ operations.[1]

S3: Forming the mode-n unfolding $\mathbf{B}_{n,(1)}$ requires $O(IL^2)$ operations.

S4: Computing $\mathbf{Q}_n$ requires $O(IL^2\mu)$ operations with $n = 1, 2, 3$.

By summing up the complexities of all the steps above, then Algorithm 3.1 requires

$$6[KI^4 + (K - 1)I^3 + I^3L + 2I^2L^2 + IL] + O(IL^2 + IL^2\mu)$$

operations for the tensor $\mathcal{A}$. Thus, Algorithm 3.1 requires $O(KI^4)$ operations for the low multilinear rank approximation of $\mathcal{A}$, where $\mu \leq L^2 + K < I$ and $K > 0$ is the oversampling parameter.

Remark 3.6. Note that for a given matrix $\mathbf{A} \in \mathbb{R}^{I\times I}$, computing $\mathbf{A}\mathbf{A}^\top$ needs $I(I + 1)(2I - 1)/2$ operations. Then in Algorithm 3.1, computing $\mathcal{B}_n$ needs

$$3K\frac{I(I + 1)}{2}(2I^2 - 1) + 3(K - 1)\frac{I(I + 1)}{2}(2I - 1) + 6(I^3L + 2I^2L^2)$$

operations.

Similarly to Algorithm 3.1, we can also analyze the computational complexity of Algorithm 3.2. In order to compute the number of floating points operations in Algorithm 3.2, we set $p_1 = 1$, $p_2 = 2$, and $p_3 = 3$.

1. For the case of $n = 1$, generating two standard Gaussian matrices requires $2IL$ operations, and computing $\mathcal{B}_1$ and $\mathcal{C}$ needs $2KI^4 + 2(K - 1)I^3 + 2I^3L + 4I^2L^2$ and $2I^3\mu$ operations, respectively.[2]
2. For the case of $n = 2$, generating two standard Gaussian matrices requires $I(L + \mu)$ operations, and computing $\mathcal{B}_2$ and $\mathcal{C}$ needs $2KI^3\mu + 2(K - 1)I^3 + 2I^2L\mu + 2I^2L^2 + 2IL^2\mu$ and $2I^2\mu^2$ operations, respectively.[3]
3. For the case of $n = 3$, generating two standard Gaussian matrices requires $2\mu L$ operations, and computing $\mathcal{B}_3$ needs $2KI^2\mu^2 + 2(K - 1)I^3 + 2IL^2\mu + 2IL\mu^2 + 2I^2L^2$ operations.[4]

Note that for each n, the number of entries of $\mathcal{B}_n$ in Algorithm 3.2 is IL^2; then, for each n, we have

(i) forming the mode-n unfolding $\mathbf{B}_{n,(1)}$ requires $O(IL^2)$ operations;

(ii) for each $n = 1, 2, 3$, computing $\mathbf{Q}_n$ requires $O(IL^2\mu)$ operations.

[1]There exists another way to generate $\mathcal{B}_n$: for all $n = 1, 2, 3$, it needs $6(K+1)I^4+6(K-1)I^3$ and $6I^3L$ operations to compute three temporary tensors $\widetilde{\mathcal{A}}_n$ and three product tensors $\mathcal{B}_n$, respectively.

[2]There exists another way to compute $\mathcal{B}_1$: it needs $(2K + 2)I^4 + 2(K - 1)I^3$ and $2I^3L + 2I^2L^2$ operations to compute $\widetilde{\mathcal{A}}_1$ and $\mathcal{B}_1$, respectively.

[3]There exists another way to compute $\mathcal{B}_2$: computing $\widetilde{A}_2$, and $\mathcal{B}_2$ needs $(2K+2)I^3\mu+2(K-1)I^3$ and $2I^2L\mu + 2I^2L^2$ operations, respectively.

[4]There exists another way to compute $\mathcal{B}_3$: computing $\widetilde{A}_3$ and $\mathcal{B}_3$ needs $(2K+2)I^2\mu^2+2(K-1)I^3$ and $2LI\mu^2 + 2IL^2\mu$ operations, respectively.

By summing up the complexities of all the steps above, Algorithm 3.2 then requires

$$2K(I^4 + I^3\mu + I^2\mu^2) + 6(K-1)I^3 + (3I + 3\mu)L \\ + 2(I^3L + I^3\mu + 4I^2L^2 + I^2\mu^2 + I^2L\mu + 2IL^2\mu + IL\mu^2) + O(IL^2 + IL^2\mu)$$

operations for the tensor $\mathcal{A}$. Thus, Algorithm 3.2 requires $O(KI^4)$ operations for the low multilinear rank approximation of $\mathcal{A}$, where $\mu \leq L^2 + T < I$ and $T > 0$ is the oversampling parameter.

Following from the above discussion, Algorithm 3.2 is faster than Algorithm 3.1 for computing the low multilinear rank approximation of a tensor under the same conditions.

3.4. Comparison with existing randomized algorithms. With the case of either given multilinear rank or given RLNE, given in (5.1), Che and Wei [8] presented a randomized algorithm for the low multilinear rank approximation of $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$.

Suppose that the multlilinear rank of $\mathcal{A}$ is given as $\{\mu_1, \mu_2, \mu_3\}$, then [8, Algorithm 3.2] can be represented as follows:

Set $L'_1 \geq \mu_1 + K$, $L'_2 \geq \mu_2 + K$, and $L'_3 \geq \mu_3 + K$, where K is an oversampling parameter.
Set the temporary tensor: $\mathcal{C} = \mathcal{A}$.
for $n = 1, 2, 3$ **do**
    Compute $\mathbf{B}_{n,(n)} = \mathbf{A}_{(n)}\mathbf{\Omega}_{(n)}$, where $\mathbf{\Omega}_{(n)} = \mathbf{\Omega}'_1 \odot \cdots \odot \mathbf{\Omega}'_{n-1} \odot \mathbf{\Omega}'_{n+1} \odot \cdots \odot \mathbf{\Omega}'_3$ and $\mathbf{\Omega}'_m \in \mathbb{R}^{I_m \times L'_n}$ is a standard Gaussian matrix with $m \neq n$ and $m = 1, 2, 3$.
    Compute $\mathbf{Q}_n$ as a columnwise orthogonal basis of $\mathbf{B}_{(n)}$ by using SVD and let $\mathbf{Q}_n = \mathbf{Q}_n(:, 1:\mu_n)$.
    Set $\mathcal{C} = \mathcal{C} \times \mathbf{Q}_n^\top$ and let $I_n = \mu_n$.
end for

We also list the randomized Tucker decomposition [72, Algorithm 2] as follows:

Set $L'_1 \geq \mu_1 + K$, $L'_2 \geq \mu_2 + K$, and $L'_3 \geq \mu_3 + K$, where K is an oversampling parameter.
Set the temporary tensor: $\mathcal{C} = \mathcal{A}$.
for $n = 1, 2, 3$ **do**
    Compute $\mathbf{B}_{n,(n)} = \mathbf{A}_{(n)}\mathbf{\Omega}_{(n)}$, where $\mathbf{\Omega}_{(n)}$ is an $(\prod_{k \neq n}^{3} I_k)$-by-$L'_n$ standard Gaussian matrix.
    Compute $\mathbf{Q}_n$ as a columnwise orthogonal basis of $\mathbf{B}_{(n)}$ by using the SVD and let $\mathbf{Q}_n = \mathbf{Q}_n(:, 1:\mu_n)$.
    Set $\mathcal{C} = \mathcal{C} \times \mathbf{Q}_n^\top$ and let $I_n = \mu_n$.
end for

Note that in [8, 72], for each n, the unpivoted QR factorization [22] is used to obtain a columnwise orthogonal basis of $\mathbf{B}_{(n)}$. Tucker-SVD in [10] can be rewritten as follows:

Set $L_2L_3 \geq \mu_1 + K$, $L_1L_3 \geq \mu_2 + K$, and $L_1L_2 \geq \mu_3 + K$, where K is an oversampling parameter.
Set the temporary tensor: $\mathcal{C} = \mathcal{A}$.
for $n = 1, 2, 3$ **do**
    Compute $\mathbf{B}_{n,(n)} = \mathbf{A}_{(n)}\mathbf{\Omega}_{(n)}$, where $\mathbf{\Omega}_{(n)} = \mathbf{\Omega}'_1 \otimes \cdots \otimes \mathbf{\Omega}'_{n-1} \times \mathbf{\Omega}'_{n+1} \otimes \cdots \otimes \mathbf{\Omega}'_3$ and $\mathbf{\Omega}'_m \in \mathbb{R}^{I_m \times L_m}$ is a standard Gaussian matrix with $m \neq n$ and $m = 1, 2, 3$.
    Compute $\mathbf{Q}_n$ as a columnwise orthogonal basis of $\mathbf{B}_{(n)}$ by using SVD and let $\mathbf{Q}_n = \mathbf{Q}_n(:, 1:\mu_n)$.

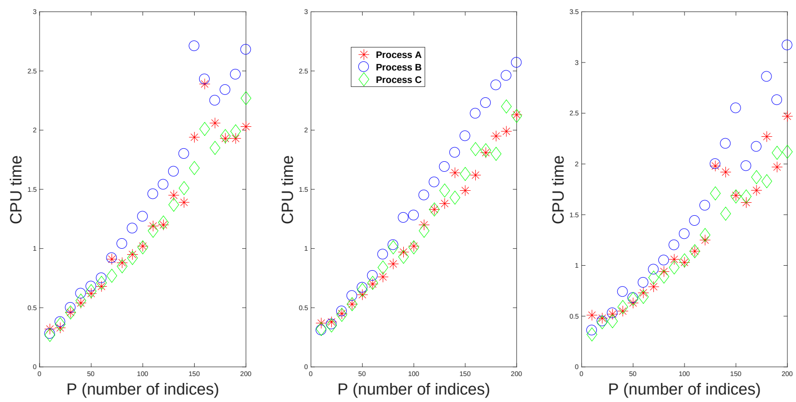

FIG. 2. *Comparison of the computation of* $\mathbf{B}_{n,(n)}$ *in Tucker-SVD,* [8, Algorithm 3.2] *and* [72, Algorithm 2] *with* $P = 10, 20, \ldots, 200$. *Process* A, *Process* B, *and Process* C *are used in Tucker-SVD,* [8, Algorithm 3.2] *and* [72, Algorithm 2], *respectively. Hint: left for* $n = 1$, *middle for* $n = 2$, *and right for* $n = 3$.

  Set $\mathcal{C} = \mathcal{C} \times \mathbf{Q}_n^\top$ and let $I_n = \mu_n$.
end for

Tucker-pSVD, the same as Algorithm 3.2, is rewritten as follows:

Set $L_2 L_3 \geq \mu_1 + K$, $L_1 L_3 \geq \mu_2 + K$ and $L_1 L_2 \geq \mu_3 + K$, where K is an oversampling parameter.
Set the temporary tensor: $\mathcal{C} = \mathcal{A}$.
for $n = 1, 2, 3$ **do**
  Compute $\mathbf{B}_{n,(1)} = \widetilde{\mathbf{A}}_{n,(1)} \mathbf{\Omega}_{(n)}$, where $\mathbf{\Omega}_{(n)} = \mathbf{\Omega}'_1 \otimes \cdots \otimes \mathbf{\Omega}'_{n-1} \times \mathbf{\Omega}'_{n+1} \otimes \cdots \otimes \mathbf{\Omega}'_3$ and $\mathbf{\Omega}'_m \in \mathbb{R}^{I_m \times L_m}$ is a standard Gaussian matrix with $m \neq n$ and $m = 1, 2, 3$. Here $\widetilde{\mathcal{A}}_n$ is given in (3.4).
  Compute $\mathbf{Q}_n$ as a columnwise orthogonal basis of $\mathbf{B}_{(n)}$ by using SVD and let $\mathbf{Q}_n = \mathbf{Q}_n(:, 1 : \mu_n)$.
  Set $\mathcal{C} = \mathcal{C} \times \mathbf{Q}_n^\top$ and let $I_n = \mu_n$.
end for

As shown in Remark 3.3, when the error of Tucker-SVD is large for the tensor whose mode-n singular value delays slowly, Tucker-pSVD has been proposed to relieve this weakness. The main difference betweeen Tucker-SVD, [8, Algorithm 3.2] and [72, Algorithm 2] is to generate the matrix $\mathbf{B}_{n,(n)}$ for each n. For all n, generating six standard Gaussian matrices requires $3I(L+\mu)$ operations for Tucker-SVD, $3I(L'+\mu)$ operations for [8, Algorithm 3.2], and $I^2L' + IL'\mu + L'\mu^2$ for [72, Algorithm 2], where we assume that $L'_1 = L'_2 = L'_3 = L' > L$.

For Tucker-SVD, [8, Algorithm 3.2] and [72, Algorithm 2], CPU times for computing $\mathbf{B}_{n,(n)}$ with $I_1 = I_2 = I_3 = 500$ and $\mu_1 = \mu_2 = \mu_3 = P$ are shown in Figure 2. Here the entries of $\mathcal{A} \in \mathbb{R}^{500 \times 500 \times 500}$ are Gaussian random variables with unit variance and zero mean.

4. Proof of the main results.

In this section, we provide the proof for our main theorem.

4.1. Basic lemmas.

In this section, we obtain some prerequisite results for proving Theorem 3.1.

LEMMA 4.1 (see [22, 67]). *Suppose that* $\mathbf{A} \in \mathbb{R}^{I\times J}$ *such that* $\mathbf{A}$ *is of full column rank, with* $I \geq J$*. Then*

$$\left\|\mathbf{A}^{\dagger}\right\|_2 = \left\|(\mathbf{A}^{\top}\mathbf{A})^{-1}\mathbf{A}^{\top}\right\|_2 = 1/\sigma_J,$$

where σ_J *is the least (that is, the* J*th greatest) singular value of* $\mathbf{A}$*.*

LEMMA 4.2 (see [10, Lemma 5.2]). *Let* I*,* J*, and* K *be three positive integers such that* $K < J < I$*. Suppose that* $\mathbf{Q} \in \mathbb{R}^{I\times K}$ *is columnwise orthogonal. For a given* $\mathbf{A} \in \mathbb{R}^{I\times J}$*, we have*

$$\sigma_{\max}(\mathbf{Q}^{\top}\mathbf{A}) \leq \sigma_{\max}(\mathbf{A}), \quad \sigma_{\min}(\mathbf{Q}^{\top}\mathbf{A}) \geq \sigma_{\min}(\mathbf{A}).$$

For given $\mathbf{A} \in \mathbb{R}^{I\times J}$ and $\mathbf{G} \in \mathbb{R}^{J\times K}$, the following lemma states the singular value of the product $\mathbf{AG}$ is at most $\|\mathbf{G}\|_2$ times greater than the corresponding singular values of $\mathbf{A}$.

LEMMA 4.3 (see [10, Corollary 5.1]). *Suppose that* $\mathbf{A} \in \mathbb{R}^{I\times J}$ *and* $\mathbf{G} \in \mathbb{R}^{J\times S}$ *with* $S \leq \min(I, J)$*. Then for all* $k = 1, 2, \dots, \min(I, J, S) - 1, \min(I, J, S)$*, we have*

$$\sum_{i=k}^{S} \sigma_i(\mathbf{AG})^2 \leq \|\mathbf{G}\|_2^2 \sum_{j=k}^{\min(I,J)} \sigma_j(\mathbf{A})^2.$$

The following classical lemma provides an approximation $\mathbf{QS}$ to $\mathbf{A} \in \mathbb{R}^{I\times J}$ via a columnwise orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{I\times R}$ and $\mathbf{S} \in \mathbb{R}^{R\times J}$.

LEMMA 4.4 (see [10, Lemma 5.3]). *Suppose that* R*,* I*, and* J *are positive integers with* $R < J$ *and* $J \leq I$*. Let* $\mathbf{A} \in \mathbb{R}^{I\times J}$*. Then there exist a columnwise orthogonal matrix* $\mathbf{Q} \in \mathbb{R}^{I\times R}$ *and* $\mathbf{S} \in \mathbb{R}^{R\times J}$ *such that*

$$\|\mathbf{QS} - \mathbf{A}\|_F \leq \Delta_{R+1}(\mathbf{A})$$

with $\Delta_{R+1}(\mathbf{A}) := (\sum_{i=R+1}^{J} \sigma_i(\mathbf{A})^2)^{1/2}$*, where* $\sigma_i(\mathbf{A})$ *is the* i*th greatest singular value of* $\mathbf{A}$ *for all* $i = 1, 2, \dots, J$*.*

Without loss of generality, we assume that $n = 1$. The following lemma states that the product $\mathcal{A} \times_1 (\mathbf{Q}_1\mathbf{Q}_1^{\top})$ is a good approximation to $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$, provided that there exist matrices $\mathbf{G}_m \in \mathbb{R}^{L_m\times I_m}$ ($m = 2, 3$) and $\mathbf{R}_1 \in \mathbb{R}^{\mu_1\times L_2L_3}$ such that (a) $\mathbf{Q}_1$ is columnwise orthogonal; (b) $\mathbf{Q}_1\mathbf{R}_1$ is a good approximation to $(\widetilde{\mathcal{A}}_1 \times_2 \mathbf{G}_2 \times_3 \mathbf{G}_3)_{(1)}$, and (c) there exist a matrix $\mathbf{F} \in \mathbb{R}^{L_2L_3\times I_2I_3}$ such that $\|\mathbf{F}\|_2$ is medium, and $\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^{\top}\mathbf{F}$ is a good approximation to $\mathbf{A}_{(1)}$, where $\widetilde{\mathcal{A}}_1 \in \mathbb{R}^{I_1\times I_2\times I_3}$ is given in (3.2).

LEMMA 4.5. *Suppose that* $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$*,* $\mathbf{Q}_1 \in \mathbb{R}^{I_1\times \mu_1}$ *is columnwise orthogonal with* $\mu_1 \leq I_1$*,* $\mathbf{S}_1$ *is a real* $\mu_1 \times L_2L_3$ *matrix,* $\mathbf{F}$ *is a real* $L_2L_3 \times I_2I_3$ *matrix, and* $\mathbf{G}_m$ *is a real* $L_m \times I_m$ *matrix with* $m = 2, 3$*. Then*

$$\begin{aligned} \left\|\mathcal{A} - \mathcal{A} \times_1 (\mathbf{Q}_1\mathbf{Q}_1^{\top})\right\|_F^2 &\leq 2\left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^{\top}\mathbf{F} - \mathbf{A}_{(1)}\right\|_F^2 \\ &\quad + 2\|\mathbf{F}\|_2^2 \left\|\mathcal{S}_1 \times_1 \mathbf{Q}_1 - \widetilde{\mathcal{A}}_1 \times_2 \mathbf{G}_2 \times_3 \mathbf{G}_3\right\|_F^2, \end{aligned} \tag{4.1}$$

where $\widetilde{\mathcal{A}}_1$ *is given in* (3.2)*,* $\mathcal{S}_1 \in \mathbb{R}^{\mu_1\times L_2\times L_3}$ *is given by* $\mathcal{S}_1 = \mathrm{reshape}(\mathbf{S}_1, [\mu_1, L_2, L_3])$*. Here* $\mathrm{reshape}(\cdot)$ *is a MATLAB function.*

Proof. The proof is straightforward, but tedious, as follows. By using the triangle inequality, we have

$$\begin{aligned}\|\mathcal{A} - \mathcal{A} \times_1 (\mathbf{Q}_1\mathbf{Q}_1^\top)\|_F^2 \le{}& \left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - (\mathbf{Q}_1\mathbf{Q}_1^\top)\,\mathbf{A}_{(1)}\right\|_F^2 \\ &+ \left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F}\right\|_F^2 \\ &+ \left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\right\|_F^2.\end{aligned} \tag{4.2}$$

For the first term in the right-hand side of (4.2), we have

$$\begin{aligned}&\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - (\mathbf{Q}_1\mathbf{Q}_1^\top)\,\mathbf{A}_{(1)}\right\|_F^2 \\ &\qquad \le \left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\right\|_F^2 \|\mathbf{Q}_1\mathbf{Q}_1^\top\|_2^2.\end{aligned} \tag{4.3}$$

Since $\|\mathbf{Q}_1\mathbf{Q}_1^\top\|_2 = 1$, then

$$\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - (\mathbf{Q}_1\mathbf{Q}_1^\top)\,\mathbf{A}_{(1)}\right\|_F^2 \le \left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\right\|_F^2.$$

Now, we provide a bound for the second term in the right-hand side of (4.2). Clearly, we have

$$\begin{aligned}&\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F}\right\|_F^2 \\ &\quad \le \left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top\right\|_F^2 \|\mathbf{F}\|_2^2.\end{aligned}$$

It follows from the triangle inequality that

$$\begin{aligned}&\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top\right\|_F^2 \\ &\quad \le \left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top - \mathbf{Q}_1\mathbf{Q}_1^\top\mathbf{Q}_1\mathbf{S}_1\right\|_F^2 \\ &\qquad + \left\|\mathbf{Q}_1\mathbf{Q}_1^\top\mathbf{Q}_1\mathbf{S}_1 - \mathbf{Q}_1\mathbf{S}_1\right\|_F^2 + \left\|\mathbf{Q}_1\mathbf{S}_1 - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top\right\|_F^2.\end{aligned}$$

Since $\mathbf{Q}_1^\top\mathbf{Q}_1 = \mathbf{I}_{\mu_1}$, then

$$\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\mathbf{Q}_1\mathbf{S}_1 - \mathbf{Q}_1\mathbf{S}_1\right\|_F^2 = 0.$$

Since $\|\mathbf{Q}_1\mathbf{Q}_1^\top\|_2 = 1$, then

$$\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top - \mathbf{Q}_1\mathbf{Q}_1^\top\mathbf{Q}_1\mathbf{S}_1\right\|_F^2 \le \left\|\mathbf{Q}_1\mathbf{S}_1 - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top\right\|_F^2.$$

Hence we have

$$\begin{aligned}&\left\|(\mathbf{Q}_1\mathbf{Q}_1^\top)\,\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F}\right\|_F^2 \\ &\quad \le 2\|\mathbf{F}\|_2^2 \left\|\mathbf{Q}_1\mathbf{S}_1 - \widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top\right\|_F^2.\end{aligned} \tag{4.4}$$

Combining (4.2), (4.3), and (4.4) yields (4.1). ∎

LEMMA 4.6 (see [54, Lemma 4.6]). *Let* $\mathbf{A}$ *be a real* $I \times J$ *matrix with* $I \leq J$ *and* $\mathbf{G}$ *be a real* $J \times L$ *matrix whose entries are i.i.d. Gaussian random variables with zero mean and unit variance. Let* R *and* L *be integers such that* $L < I$ *and* $L > (1+1/\ln(R))R$*. We define* a_1*,* a_2*,* c_1*, and* c_2 *as in Theorems* 2.3 *and* 2.4*. Then, there exists a matrix* $\mathbf{F} \in \mathbb{R}^{L \times J}$ *such that*

$$\|\mathbf{AGF} - \mathbf{A}\|_2 \leq \sqrt{\frac{a_1^2 J}{c_1^2 L} + 1}\sigma_{R+1}(\mathbf{A}),$$

and

$$\|\mathbf{F}\|_2 \leq \frac{1}{c_1\sqrt{L}}$$

with probability at least $1 - e^{-c_2 L} - e^{-a_2 J}$*.*

When $N = 2$, Algorithm 3.1 is similar to the randomized algorithm in [48] for the low rank approximation of matrices that produces accuracy very close to the best possible, for matrices with arbitrary sizes.

The following lemma states that the product $\mathbf{AQQ}^\top$ is a good approximation to $\mathbf{A} \in \mathbb{R}^{I\times J}$, provided that there exist matrices $\mathbf{G} \in \mathbb{R}^{L\times I}$ and $\mathbf{R} \in \mathbb{R}^{R\times L}$ such that: (a) $\mathbf{Q} \in \mathbb{R}^{I\times R}$ is columnwise orthogonal, (b) $\mathbf{QR}$ is a good approximation to $(\mathbf{GB})^\top$, and (c) there exists a matrix $\mathbf{F} \in \mathbb{R}^{I\times L}$ such that $\|\mathbf{F}\|_2$ is medium, and $\mathbf{FGB}$ is a good approximation to $\mathbf{A}$, where $\mathbf{B} = (\mathbf{AA}^\top)^K\mathbf{A}$ with a given positive integer K.

LEMMA 4.7 (see [48, Lemma 3.1]). *Suppose that* K*,* R*,* L*,* I*, and* J *are positive integers with* $R \leq L \leq I \leq J$*. Suppose further that* $\mathbf{A}$ *is a real* $I \times J$ *matrix,* $\mathbf{Q}$ *is a real* $J \times R$ *matrix whose columns are columnwise orthogonal,* $\mathbf{R}$ *is a real* $R \times L$ *upper triangular matrix,* $\mathbf{F}$ *is a real* $I \times L$ *matrix, and* $\mathbf{G}$ *is a real* $L \times I$ *matrix. Then for* $\mathbf{B} = (\mathbf{AA}^\top)^K\mathbf{A}$*, we have*

$$\left\|\mathbf{A} - \mathbf{AQQ}^\top\right\|_2 = 2\|\mathbf{FGB} - \mathbf{A}\|_2 + 2\|\mathbf{F}\|_2 \left\|\mathbf{QR} - (\mathbf{GB})^\top\right\|_2. \tag{4.5}$$

Rokhlin, Szlam, and Tygert [48] considered estimating the second part in the right-hand side of (4.5). In order to estimate the first part in the right-hand side of (4.5), we need the following theorem, which is similar to Lemma 4.6.

THEOREM 4.8. *Let* $\mathbf{A}$ *be a real* $I \times J$ *matrix with* $I \leq J$ *and* $\mathbf{G}$ *be a real* $J \times L$ *matrix whose entries are i.i.d. Gaussian random variables with zero mean and unit variance. Let* R *and* L *be integers such that* $L < I$ *and* $L > (1 + 1/\ln(R))R$*. We define* a_1*,* a_2*,* c_1*, and* c_2 *as in Theorems* 2.3 *and* 2.4*. Then, there exists a matrix* $\mathbf{F} \in \mathbb{R}^{L\times J}$ *such that*

$$\left\|\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{GF} - \mathbf{A}\right\|_2 \leq \Delta_{R+1}(\mathbf{A})^{2K+1}\frac{a_1\sqrt{J}}{c_1\sigma_R^{2K}\sqrt{L}} + \Delta_{R+1}(\mathbf{A}),$$

and

$$\|\mathbf{F}\|_2 \leq \frac{1}{c_1\sigma_R^{2K}\sqrt{L}}$$

with probability at least $1 - e^{-c_2 L} - e^{-a_2 J}$*, where* K *is a given positive integer.*

Remark 4.1. Theorem 4.8 is similar to [48, Lemma 3.2], in which one estimates the upper bounds of $\|\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{GF} - \mathbf{A}\|_2$ and $\|\mathbf{F}\|_2$ by the highly probable bounds on the singular values of a rectangular matrix whose entries are i.i.d. Gaussian random variables of zero mean and unit variance [11, 32].

Proof. We begin by the application of the SVD to $\mathbf{A}$ such that $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{I\times J}$ is columnwise orthogonal, $\mathbf{\Sigma} \in \mathbb{R}^{J\times J}$ is diagonal with nonnegative entries and, $\mathbf{V} \in \mathbb{R}^{J\times J}$ is orthogonal. Hence, we have

$$\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K = \mathbf{U}\mathbf{\Sigma}^{2K+1}\mathbf{V}^\top. \tag{4.6}$$

Assume that given $\mathbf{V}^\top$ and $\mathbf{G}$, suppose that

$$\mathbf{V}^\top\mathbf{G} = \begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix},$$

where $\mathbf{H}$ is an $R\times L$ matrix and $\mathbf{R}$ is a $(J-R)\times L$ matrix. Since $\mathbf{G}$ is a standard Gaussian matrix, and $\mathbf{V}$ is an orthogonal matrix, then $\mathbf{V}^\top\mathbf{G}$ is also a standard Gaussian matrix. Therefore, $\mathbf{H}$ and $\mathbf{R}$ are also standard Gaussian matrices. Define $\mathbf{F} = \mathbf{P}\mathbf{V}^\top$, where $\mathbf{P}$ is a matrix of size $L\times J$ such that

$$\mathbf{P} = \begin{pmatrix}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & \mathbf{0}_{L\times(J-R)}\end{pmatrix}, \quad \mathbf{\Sigma}_1 = \mathbf{\Sigma}(1:R, 1:R).$$

By computing $\|\mathbf{F}\|_2$ using Theorem 2.4, we get

$$\|\mathbf{F}\|_2 = \left\|\mathbf{P}\mathbf{V}^\top\right\|_2 = \left\|\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K}\right\|_2 \le \frac{\left\|\mathbf{H}^\dagger\right\|_2}{\sigma_R^{2K}} = \frac{1}{\sigma_R^{2K}}\frac{1}{\sigma_{\min}(\mathbf{H})} \le \frac{1}{c_1\sqrt{L}\sigma_R^{2K}}$$

with probability not less than $1 - e^{-c_2 L}$. Now, we can bound $\|\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{G}\mathbf{F} - \mathbf{A}\|_2$. By (4.6), we get

$$\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{G}\mathbf{F} - \mathbf{A} = \mathbf{U}\mathbf{\Sigma}\left(\mathbf{\Sigma}^{2K}\begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix}\begin{pmatrix}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & \mathbf{0}_{L\times(J-R)}\end{pmatrix} - \mathbf{I}_J\right)\mathbf{V}^\top. \tag{4.7}$$

We define $\mathbf{\Sigma}_2$ to be the $(J-R)\times(J-R)$ lower-right block of $\mathbf{\Sigma}$. Then

$$\begin{aligned}
&\mathbf{\Sigma}\left(\mathbf{\Sigma}^{2K}\begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix}\begin{pmatrix}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & \mathbf{0}_{L\times(J-R)}\end{pmatrix} - \mathbf{I}_J\right)\\
&\quad = \mathbf{\Sigma}\begin{pmatrix}\mathbf{\Sigma}_1^{2K} & \mathbf{0}_{R\times(J-R)}\\ \mathbf{0}_{(J-R)\times R} & \mathbf{\Sigma}_2^{2K}\end{pmatrix}\begin{pmatrix}\mathbf{0}_{R\times R} & \mathbf{0}_{R\times(J-R)}\\ \mathbf{R}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & -\mathbf{I}_{J-R}\end{pmatrix}\\
&\quad = \begin{pmatrix}\mathbf{0}_{R\times R} & \mathbf{0}_{R\times(J-R)}\\ \mathbf{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & -\mathbf{\Sigma}_2\end{pmatrix}.
\end{aligned}$$

The norm of the last term is:

$$\left\|\begin{pmatrix}\mathbf{0}_{R\times R} & \mathbf{0}_{R\times(J-R)}\\ \mathbf{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K} & -\mathbf{\Sigma}_2\end{pmatrix}\right\|_F \le \left\|\mathbf{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K}\right\|_F + \|\mathbf{\Sigma}_2\|_F.$$

Moreover,

$$\begin{aligned}
\left\|\mathbf{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\mathbf{\Sigma}_1^{-2K}\right\|_F &\le \left\|\mathbf{\Sigma}_1^{-2K}\right\|_2\left\|\mathbf{H}^\dagger\right\|_2\|\mathbf{R}\|_2\left\|\mathbf{\Sigma}_2^{2K+1}\right\|_F\\
&\le \left\|\mathbf{\Sigma}_1^{-1}\right\|_2^{2K}\left\|\mathbf{H}^\dagger\right\|_2\|\mathbf{R}\|_2\|\mathbf{\Sigma}_2\|_F^{2K+1}.
\end{aligned} \tag{4.8}$$

Therefore, by using (4.7), (4.8), and the fact that $\|\mathbf{\Sigma}_2\|_F = \Delta_{R+1}(\mathbf{A})$, we get

$$\begin{aligned}
\left\|\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{G}\mathbf{F} - \mathbf{A}\right\|_F &\le \|\mathbf{\Sigma}_2\|_F^{2K+1}\|\mathbf{R}\|_2\left\|\mathbf{H}^\dagger\right\|_2\sigma_R^{-2K} + \|\mathbf{\Sigma}_2\|_F\\
&= \Delta_{R+1}(\mathbf{A})^{2K+1}\|\mathbf{R}\|_2\left\|\mathbf{H}^\dagger\right\|_2\sigma_R^{-2K} + \Delta_{R+1}(\mathbf{A}).
\end{aligned} \tag{4.9}$$

We also know that

$$\|\mathbf{R}\|_2 \leq \left\|\mathbf{V}^\top \mathbf{G}\right\|_2 = \|\mathbf{G}\|_2 \leq a_1\sqrt{J}$$

with probability not less than $1-e^{-a_2 L}$. Combining (4.9) with the fact that $\|\mathbf{H}^\dagger\|_2 \leq \frac{1}{c_1\sqrt{L}}$ and $\|\mathbf{R}\|_2 \leq a_1\sqrt{J}$ gives

$$\left\|\mathbf{A}(\mathbf{A}^\top\mathbf{A})^K\mathbf{G}\mathbf{F} - \mathbf{A}\right\|_F = \frac{a_1\sqrt{J}}{c_1\sigma_R^{2K}\sqrt{L}}\Delta_{R+1}(\mathbf{A})^{2K+1} + \Delta_{R+1}(\mathbf{A}).$$

Hence this theorem is completely proved. ∎

We now estimate the first term in the right-hand side of (4.1), which is given in the following theorem.

THEOREM 4.9. *Suppose that* $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ *and* K *is a given positive integer. Let* $\mathbf{G}_m$ *be a real* $L_m \times I_m$ *matrix whose entries are i.i.d. Gaussian random variables with zero mean and unit variance for* $m = 2, 3$. *Let* μ_1, L_2, *and* L_3 *be integers such that* $(1 + 1/\ln(\sqrt{\mu_1}))\sqrt{\mu_1} < L_2, L_3$ *and* $L_2L_3 < \min(I_1, I_2I_3)$. *We define* a_1, a_1', c_{12}, c_{12}', c_{13}, *and* c_{13}' *as in Theorems* 2.3 *and* 2.4. *Then there exist a matrix* $\mathbf{F} \in \mathbb{R}^{L_2L_3\times I_2I_3}$ *such that*

$$\left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3\otimes\mathbf{G}_2)^\top\mathbf{F} - \mathbf{A}_{(1)}\right\|_F \leq \frac{1}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\sqrt{\frac{a_1^2 I_2 I_3}{c_{12}^2 c_{13}^2 L_2 L_3}}\Delta_{\mu_1+1}(\mathbf{A}_{(1)})^{2K+1} + \Delta_{\mu_1+1}(\mathbf{A}_{(1)}),$$

and

$$\|\mathbf{F}\|_2 \leq \frac{1}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\frac{1}{c_{12}c_{13}\sqrt{L_2L_3}}$$

with probability at least $1 - e^{-c_{12}'L_2} - e^{-c_{13}'L_3} - e^{-a_1' I_2 I_3}$, *where* $\widetilde{\mathcal{A}}_1$ *is given in* (3.2).

Proof. Note that $\widetilde{\mathbf{A}}_{1,(1)} = \mathbf{A}_{(1)}(\mathbf{A}_{(1)}^\top\mathbf{A}_{(1)})^K$. Suppose that $\mathbf{A}_{(1)} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{I_1\times I_1}$ is orthogonal, $\boldsymbol{\Sigma} \in \mathbb{R}^{I_1\times I_1}$ is diagonal with nonnegative entries, and $\mathbf{V} \in \mathbb{R}^{I_2I_3\times I_1}$ is columnwise orthogonal. Then $\widetilde{\mathbf{A}}_{1,(1)} = \mathbf{U}\boldsymbol{\Sigma}^{2K+1}\mathbf{V}^\top$. Assume that given $\mathbf{V}^\top$ and $\mathbf{G}_3\otimes\mathbf{G}_2$, suppose that

$$\mathbf{V}^\top(\mathbf{G}_3\otimes\mathbf{G}_2) = \begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix},$$

where $\mathbf{H}$ is a $\mu_1 \times L_2L_3$ matrix and $\mathbf{R}$ is an $(I_1-\mu_1)\times L_2L_3$ matrix. Since $\mathbf{G}_3\otimes\mathbf{G}_2$ is a random sub-Gaussian matrix, and $\mathbf{V}$ is a columnwise orthogonal matrix, then $\mathbf{V}^\top(\mathbf{G}_3\otimes\mathbf{G}_2)$ is also a random sub-Gaussian matrix. Therefore, $\mathbf{H}$ and $\mathbf{R}$ are random sub-Gaussian matrices. Define $\mathbf{F}_3\otimes\mathbf{F}_2 = \mathbf{P}\mathbf{V}^\top$, where $\mathbf{P}$ is a matrix of size $L_2L_3\times I_1$ such that

$$\mathbf{P} = \begin{pmatrix}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & \mathbf{0}_{L_2L_3\times(I_1-R)}\end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}(1:\mu_1, 1:\mu_1).$$

According to Lemma 4.2 and Theorem 2.3, we get

$$\|\mathbf{F}\|_2 = \left\|\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K}\right\|_2 \leq \frac{\left\|\mathbf{H}^\dagger\right\|_2}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}} \leq \frac{1}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\frac{1}{\sigma_{\min}(\mathbf{H})} \leq \frac{1}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\frac{1}{\sqrt{c_{12}^2c_{13}^2L_2L_3}}$$

with probability not less than $1 - e^{-c_{12}'L_2} - e^{-c_{13}'L_3}$.

Now, we can bound $\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\|_F$. By using (4.8), we get

$$\begin{aligned}&\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\\ &\qquad = \mathbf{U}\boldsymbol{\Sigma}\left(\boldsymbol{\Sigma}^{2K}\begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix}\begin{pmatrix}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & \mathbf{0}_{L_2L_3\times(I_1-\mu_1)}\end{pmatrix} - \mathbf{I}_{I_1}\right)\mathbf{V}^\top.\end{aligned}$$

We define $\boldsymbol{\Sigma}_2$ to be the $(I_1-\mu_1)\times(I_1-\mu_1)$ lower-right block of $\boldsymbol{\Sigma}$. Then

$$\begin{aligned}&\boldsymbol{\Sigma}\left(\boldsymbol{\Sigma}^{2K}\begin{pmatrix}\mathbf{H}\\ \mathbf{R}\end{pmatrix}\begin{pmatrix}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & \mathbf{0}_{L_2L_3\times(I_1-\mu_1)}\end{pmatrix} - \mathbf{I}_{I_1}\right)\\ &\quad = \boldsymbol{\Sigma}\begin{pmatrix}\boldsymbol{\Sigma}_1^{2K} & \mathbf{0}_{\mu_1\times(I_1-\mu_1)}\\ \mathbf{0}_{(I_1-\mu_1)\times\mu_1} & \boldsymbol{\Sigma}_2^{2K}\end{pmatrix}\begin{pmatrix}\mathbf{0}_{\mu_1\times\mu_1} & \mathbf{0}_{\mu_1\times(I_1-\mu_1)}\\ \mathbf{R}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & -\mathbf{I}_{I_1-\mu_1}\end{pmatrix}\\ &\quad = \begin{pmatrix}\mathbf{0}_{\mu_1\times\mu_1} & \mathbf{0}_{\mu_1\times(I_1-\mu_1)}\\ \boldsymbol{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & -\boldsymbol{\Sigma}_2\end{pmatrix}.\end{aligned}$$

The norm of the last term is

$$\left\|\begin{pmatrix}\mathbf{0}_{\mu_1\times\mu_1} & \mathbf{0}_{\mu_1\times(I_1-\mu_1)}\\ \boldsymbol{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K} & -\boldsymbol{\Sigma}_2\end{pmatrix}\right\|_F \leq \left\|\boldsymbol{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K}\right\|_F + \|\boldsymbol{\Sigma}_2\|_F. \tag{4.10}$$

Moreover,

$$\begin{aligned}\left\|\boldsymbol{\Sigma}_2^{2K+1}\mathbf{R}\mathbf{H}^\dagger\boldsymbol{\Sigma}_1^{-2K}\right\|_F &\leq \left\|\boldsymbol{\Sigma}_1^{-2K}\right\|_2\left\|\mathbf{H}^\dagger\right\|_2\|\mathbf{R}\|_2\|\boldsymbol{\Sigma}_2^{2K+1}\|_F\\ &\leq \left\|\boldsymbol{\Sigma}_1^{-1}\right\|_2^{2K}\left\|\mathbf{H}^\dagger\right\|_2\|\mathbf{R}\|_2\|\boldsymbol{\Sigma}_2\|_F^{2K+1}.\end{aligned} \tag{4.11}$$

Therefore, by using (4.10), (4.11), and the fact that $\|\boldsymbol{\Sigma}_2\|_F = \Delta_{\mu_1+1}(\mathbf{A})$, we get

$$\begin{aligned}&\left\|\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3 \otimes \mathbf{G}_2)^\top \mathbf{F} - \mathbf{A}_{(1)}\right\|_F\\ &\quad \leq \|\boldsymbol{\Sigma}_2\|_F^{2K+1}\|\mathbf{R}\|_2\left\|\mathbf{H}^\dagger\right\|_2\sigma_{\mu_1}^{-2K} + \|\boldsymbol{\Sigma}_2\|_F\\ &\quad = \Delta_{\mu_1+1}(\mathbf{A})^{2K+1}\|\mathbf{R}\|_2\left\|\mathbf{H}^\dagger\right\|_2\sigma_{\mu_1}^{-2K} + \Delta_{\mu_1+1}(\mathbf{A})\\ &\quad \leq \Delta_{\mu_1+1}(\mathbf{A})^{2K+1}\|\mathbf{G}_3\otimes\mathbf{G}_2\|_2\|\mathbf{H}^\dagger\|_2\sigma_{\mu_1}^{-2K} + \Delta_{\mu_1+1}(\mathbf{A}).\end{aligned}$$

By Theorem 2.3, we know

$$\|\mathbf{G}_3\otimes\mathbf{G}_2\|_2 \leq a_1\sqrt{I_2I_3}$$

with probability not less than $1-e^{-a_1' I_2 I_3}$. Hence this theorem is completely proved. ∎

4.2. Proving Theorem 3.1. In this section, we assume that $\mathbf{Q}_1$ in Lemma 4.5 is derived from Algorithm 3.1. The main goal is to estimate the upper bound of $\|\mathcal{A} - \mathcal{A}\times_1(\mathbf{Q}_1\mathbf{Q}_1^\top)\|_F$. As shown in Lemma 4.5 and Theorem 4.9, we need only to give an upper bound for the second part in the right-hand side of (4.1).

For a given $\mathbf{A}\in\mathbb{R}^{I\times J}$, suppose that the entries of $\mathbf{G}\in\mathbb{R}^{J\times L}$ are i.i.d. Gaussian variables of zero mean and unit variance, the following theorem provides a highly probable upper bound on the singular values of the product $\mathbf{AG}$ in term of the singular values of $\mathbf{A}$.

THEOREM 4.10 (see [10, Theorem 5.2]). *Let $\mathbf{A}$ be a real $I\times J$ matrix with $I\leq J$. Let R and L be integers such that $R < L < I$. Suppose that $\mu\geq 1$, and the entries of*

$\mathbf{G} \in \mathbb{R}^{J\times R}$ *are i.i.d. sub-Gaussian random variables with zero mean and unit variance. We define* a_1 *and* a_2 *as in Theorems* 2.3 *and* 2.4*. Then*

$$\Delta_{R+1}(\mathbf{AG}) \leq a_1\sqrt{J}\Delta_{R+1}(\mathbf{A})$$

with probability at least $1-e^{-a_2 J}$*, where* $a_1 = 6\mu\sqrt{a_2+4}$*.*

COROLLARY 4.11. *Suppose that* $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ *and* K *is a given positive integer. Let* $\mathbf{G}_m$ *be a real* $L_m \times I_m$ *matrix whose entries are i.i.d. Gaussian random variables with zero mean and unit variance for* $m = 2, 3$*. We define* a_1 *and* a_2 *as in Theorems* 2.3 *and* 2.4*. Then*

$$\Delta_{\mu_1+1}(\widetilde{\mathbf{A}}_{1,(1)}(\mathbf{G}_3\otimes\mathbf{G}_2)^\top) \leq a_1\sqrt{I_2I_3}\Delta_{\mu_1+1}(\mathbf{A}_{(1)})^{2K+1}$$

with probability at least $1-e^{-a_2L_2L_3}$*, where* $a_1 = 6\mu\sqrt{a_2+4}$ *and* $\widetilde{\mathcal{A}}_1$ *is given in* (3.2)*.*

Proof. Applying Theorem 4.10 and combining Theorem 2.3, it is easy to prove this corollary. □

Combining Theorem 4.9 and Corollary 4.11, it is easy to obtain the following theorem.

THEOREM 4.12. *Suppose that* K *is a given positive integer. Let* $\mathbf{G}_m$ *be a real* $L_m\times I_m$ *matrix whose entries are i.i.d. Gaussian random variables with zero mean and unit variance for* $m = 2, 3$*. Let* μ_1*,* L_2*, and* L_3 *be integers such that* $(1 + 1/\ln(\sqrt{\mu_1}))\sqrt{\mu_1} < L_2, L_3$*, and* $L_2L_3 < \min(I_1, I_2I_3)$*. Let* $\sqrt{\mu_1}$ *be a positive integer. We define* a_1*,* a_2*,* c_{12}*,* c'_{12}*,* c_{13}*, and* c'_{13} *as in Theorems* 2.3 *and* 2.4*.*

For a given $\mathcal{A}\in\mathbb{R}^{I_1\times I_2\times I_3}$ $\mathbf{Q}_1$ *is derived from Algorithm* 3.1 *with* $n = 1$*. Then*

$$\left\|\mathcal{A} - \mathcal{A}\times_1\left(\mathbf{Q}_1\mathbf{Q}_1^\top\right)\right\|_F \leq 2C_1\Delta_{\mu_1+1}(\mathbf{A}_{(1)})^{2K+1} + 2\Delta_{\mu_1+1}(\mathbf{A}_{(1)})$$

with probability at least $1 - e^{-c'_{12}L_2} - e^{-c'_{13}L_3} - e^{-a'_1I_2I_3}$*, where the expression of* C_1 *is given by*

$$C_1 = \frac{2}{\sigma_{\mu_1}(\mathbf{A}_{(1)})^{2K}}\sqrt{\frac{a_1^2I_2I_3}{c_{12}^2c_{13}^2L_2L_3}}.$$

Now, we give a rigorous proof for Theorem 3.1 based on the above discussions.

Proof. Theorem 3.1 is derived from (3.1) and Theorem 4.12. □

5. Numerical examples. In this section, computer programs are developed using numerical computation software MATLAB and the MATLAB Tensor Toolbox [2] and the calculations are implemented on a laptop with Intel Core i5-4200M CPU (2.50 GHz) and 8.00 GB RAM. Floating point numbers in each example have four decimal digits. We use these three functions "ttv," "ttm," and "ttt" in [2] to implement the tensor-vector product, the tensor-matrix product, and the tensor-tensor product, respectively.

For clarity, we assume that $I_1 = I_2 = I_3 := I$ and $L_1 = L_2 = L_3 := L$. The order of computing factor matrices is $\{1, 2, 3\}$. In this section, we will compare our algorithms with the existing numerical algorithms for computing the low multilinear rank approximation of testing tensors $\mathcal{A}\in\mathbb{R}^{I\times I\times I}$ under the case of different P's with fixed I.

The relative error for the low multilinear rank approximation of $\mathcal{A}\in\mathbb{R}^{I\times I\times I}$ is defined as

$$\text{RLNE} = \left\|\mathcal{A} - \widehat{\mathcal{A}}\right\|_F / \|\mathcal{A}\|_F, \tag{5.1}$$

where $\widehat{\mathcal{A}} = \mathcal{A} \times_1 (\mathbf{S}_1\mathbf{S}_1^\dagger) \times_2 (\mathbf{S}_2\mathbf{S}_2^\dagger) \times_3 (\mathbf{S}_3\mathbf{S}_3^\dagger)$ and the matrices $\mathbf{S}_n \in \mathbb{R}^{I \times R_n}$ are derived form the desired numerical algorithms.

In this section, we compare Algorithm 3.2 with the existing algorithms for computing low multilinear rank approximations of tensors via several experiments using both synthetic and real-world data. These algorithms are given by the following:

- Tucker_ALS: higher-order orthogonal iteration [2] (the maximum number of iterations is set to 50, the order to loop through dimensions is $\{1, 2, 3\}$, the entries of initial values are i.i.d. standard Gaussian variables, and the tolerance on difference in fit is set to 0.0001);
- mlsvd: truncated multilinear SVD [61] (the order to loop through dimensions is $\{1, 2, 3\}$ and a faster but possibly less accurate eigenvalue decomposition is used to compute the factor matrices);
- lmlra_aca: low multilinear rank approximation by adaptive cross approximation [5, 66] (the relative singular value tolerance in determining the factor matrices is set to $1e$–12 and the factor matrices are columnwise orthogonal);
- Adap-Tucker: low multilinear rank approximation by the adaptive randomized algorithm [8];
- ran-Tucker: the randomized Tucker decomposition [72];
- mlsvd_rsi: truncated multilinear SVD [61] by a randomized SVD algorithm based on randomized subspace iteration [28] (the oversampling parameter is 10, the number of subspace iterations to be performed is 2, and we remove the parts of the factor matrices and core tensor corresponding due to the oversampling);
- Tucker-SVD: low multilinear rank approximation by combining random projection and the SVD [10].

Remark 5.1. In this section, we set MATLAB maxNumCompThreads to 1 and use "tic" and "toc" to measure running time when applying all algorithms to the test tensors.

Since tucker_ALS is an iterative algorithm, then each step of tucker_ALS requires $6(I^3\mu + I^2\mu^2) + O(I\mu^3)$ operations, whose initial values are chosen by the truncated HOSVD; mlsvd requires $O(I^4 + I^3\mu + I^2\mu^2 + I^2\mu + I\mu^2 + \mu^3)$ or $O(I^4)$ operations; Tucker-SVD requires $3(I + \mu)L + 2(I^3L + I^3\mu + I^2L^2 + I^2\mu^2 + I^2L\mu + 2IL^2\mu + IL\mu^2) + O(IL^2 + IL^2\mu)$ or $O(I^3)$ operations; Adap-Tucker requires $2(I^3L' + I^2L'\mu + IL'\mu^2 + I^3\mu + I^2\mu^2) + 3(I + L')\mu + O(IL'\mu)$ operations; ran-Tucker requires $2(I^3L' + I^2L'\mu + IL'\mu^2 + I^3\mu + I^2\mu^2) + I^2L' + IL'\mu + L'\mu^2 + O(IL'\mu)$ operations; mlsvd_rsi requires $(2K + 2)(I^4 + I^3\mu + I^2\mu^2) + 6(K - 1)I^3 + 2(I^3L' + I^2L'\mu + IL'\mu^2 + I^3\mu + I^2\mu^2) + I^2L' + IL'\mu + L'\mu^2 + O(IL'\mu)$ operations and lmlra_aca requires $O(I\mu^4 + \mu^5)$ operations, where L, I, L', and μ are given in section 3.3.

Remark 5.2. We have three statements for the computational complexity of the proposed algorithms used in this section:

(a) In comparison with Tucker-SVD, Tucker-pSVD takes more than $2K(I^4 + I^3\mu + I^2\mu^2) + 6(K - 1)I^3 + 6I^2L^2$ operations to generate $\mathcal{B}_n$ for all $n = 1, 2, 3$.
(b) Both Tucker-pSVD with $K = 1$ and mlsvd require $O(I^4)$ operations. However, for Tucker-pSVD with $K = 1$, the coefficient of I^4 is 2; and for mlsvd, the coefficient of I^4 is 6 or 14. As shown in [22, p. 493], Golub–Reinsch SVD and R-SVD require $14IJ^2 + 8J^4$ and $6IJ^2 + 20J^3$ operations, respectively, for the SVD of $\mathbf{A} \in \mathbb{R}^{I \times J}$ with $I \geq J$.
(c) Both Tucker-pSVD and mlsvd_rsi require $O(I^4)$ operations. However, for Tucker-pSVD, the number of coefficients of I^4 is $2K$; and for mlsvd_rsi, the

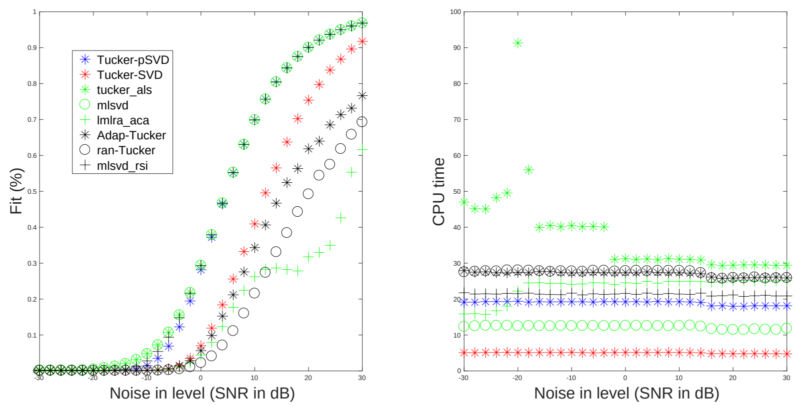

FIG. 3. *For Example* 5.1, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi to the tensor $\mathcal{A}$ with different noise levels from* -30 *to* 30.

number of coefficients of I^4 is $2(K+2)$. When we analyze the computational complexity of Tucker-pSVD, we cite [22] for a leading flop constant of 6 using the R-SVD algorithm. However, when we apply Tucker-SVD and Tucker-pSVD to the test tensors, for each n, we only need the columnwise orthogonal matrix consisting of μ_n leading left singular vectors.

5.1. Simulations using synthetic data. In this section, we present three synthetic tensors to illustrate the proposed algorithms are efficient for the low mutlilinear approximation of a tensor.

Example 5.1. Let $\mathcal{B} \in \mathbb{R}^{I\times I\times I}$ be given in the Tucker form:

$$\mathcal{B} = \mathcal{G} \times_1 \mathbf{Q}_1 \times_2 \mathbf{Q}_2 \times_3 \mathbf{Q}_3,$$

where the entries of $\mathcal{G} \in \mathbb{R}^{R\times R\times R}$ and $\mathbf{Q}_n \in \mathbb{R}^{I\times R}$ $(n = 1, 2, 3)$ are i.i.d. Gaussian variables with zero mean and unit variance. This type of $\mathcal{B}$ is also used in [5].

We now compare the proposed randomized algorithms with Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi for the low multilinear rank approximation of $\mathcal{A} \in \mathbb{R}^{I\times I\times I}$. The form of $\mathcal{A}$ is given as $\mathcal{A} = \mathcal{B} + \beta\mathcal{N}$, and $\mathcal{N} \in \mathbb{R}^{I\times I\times I}$ is an unstructured perturbation tensor with different noise level β. The following signal-to-noise ratio (SNR) measure will be used:

$$\text{SNR [dB]} = 10 \log \left(\frac{\|\mathcal{B}\|_F^2}{\|\beta\mathcal{N}\|_F^2} \right).$$

The FIT value for approximating the tensor $\mathcal{A}$ is defined by

$$\text{FIT} = 1 - \text{ERR},$$

where ERR are given in (5.1). We assume that $I = 400$ and $R = 50$. The results of Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, applied to the tensor $\mathcal{A}$ with different noise level SNRs are shown in Figure 3.

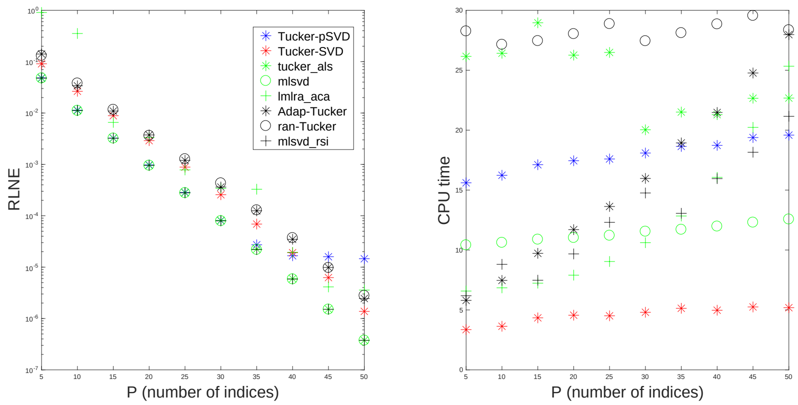

FIG. 4. *For Example* 5.2, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi to the tensor* $\mathcal{A}$ *with* $P = 5, 10, \ldots, 50$ *and* $s = 10$.

From Figure 3, in terms of CPU time, Tucker-SVD is the fastest one, Tucker-pSVD is competitive with mlsvd_rsi and faster than mlsvd, lmlra_aca, tucker_ALS, ran-Tucker, and Adap-Tucker. In terms of RLNE, Tucker-pSVD, tucker_ALS, mlsvd, and mlsvd_rsi are the best ones.

Remark 5.3. As shown in Figure 4, for each algorithm, the CPU time of different SNRs is not very different. The reason is that the size of $\mathcal{C}$ is $400 \times 400 \times 400$ and $P = 50$.

Example 5.2. We will compare the proposed randomized algorithms with Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi for the low multilinear rank approximation of $\mathcal{A} \in \mathbb{R}^{I\times I\times I}$, where the entries of $\mathcal{A}$ are generated by sampling some smooth functions. The entries of $\mathcal{A}$ are given by

$$a_{ijk} = \frac{1}{(i^s + j^s + k^s)^{1/s}},$$

where s is a positive integer. The type of tensor $\mathcal{A}$ is chosen from [5].

We assume $I = 400$. The results of Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, applied to the tensor $\mathcal{A}$ with different multilinear ranks $\{P, P, P\}$ and $s = 10$, are shown in Figure 4.

From Figure 4, in terms of CPU time, Tucker-SVD is the fastest one, and Tucker-pSVD is competitive with mlsvd_rsi and lmlra_aca; in terms of RLNE, Tucker-pSVD is competitive with lmlra_aca.

Example 5.3. We test on a sparse tensor $\mathcal{A} \in \mathbb{R}^{I\times I\times I}$ of the following format [50, 55]:

$$\mathcal{A} = \sum_{j=1}^{10} \frac{1000}{j} \mathbf{x}_j \circ \mathbf{y}_j \circ \mathbf{z}_j + \sum_{j=11}^{I} \frac{1}{j} \mathbf{x}_j \circ \mathbf{y}_j \circ \mathbf{z}_j,$$

where $\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j \in \mathbb{R}^I$ are sparse vectors with nonzero entries (in MATLAB, $\mathbf{x}_j = \text{sprand}(I, 1, 0.015)$, $\mathbf{y}_j = \text{sprand}(I, 1, 0.025)$, and $\mathbf{z}_j = \text{sprand}(I, 1, 0.035)$). We compute a rank-$(R, R, R)$ Tucker decomposition of $\mathcal{A}$ by Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, respectively.

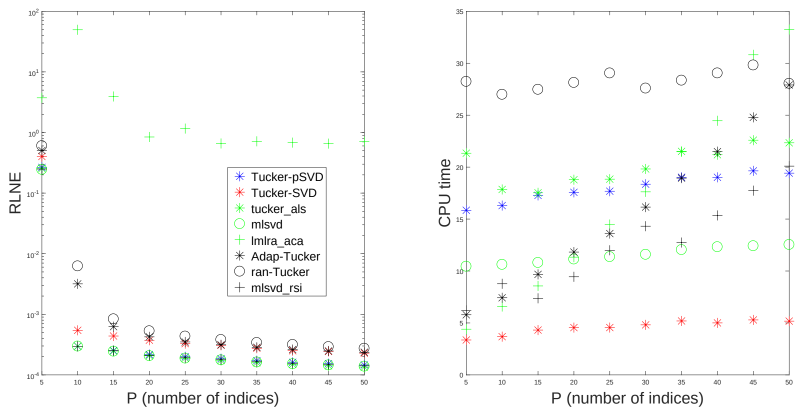

Fig. 5. *For Example* 5.3, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi to the tensor $\mathcal{A}$ with* $P = 5, 10, \ldots, 50$.

We assume that $I = 400$. The results of Tucker-pSVD, Tucker-SVD, tucker_ALS, lmlra_aca, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, applied to the tensor $\mathcal{A}$ with different multilinear ranks $\{P, P, P\}$, are shown in Figure 5. From this figure, in terms of CPU time, Tucker-SVD is the fastest one, and Tucker-pSVD is competitive with mlsvd_rsi and tucker_ALS. In terms of RLNE, Tucker-pSVD is competitive with other algorithms.

Example 5.4. For a given $p > 0$, let $\mathbf{v} \in \mathbb{R}^I$ satisfy

$$v_i = \begin{cases} 1, & i = 1, 2, \ldots, 50; \\ (i-49)^{-p}, & i = 51, 52, \ldots, I. \end{cases}$$

We construct the input tensor $\mathcal{A} \in \mathbb{R}^{I \times I \times I}$ as $\mathcal{A} = \text{tendiag}(\mathbf{v}, [I, I, I])$, where tendiag is a function in the MATLAB Tensor Toolbox [2] which converts $\mathbf{v}$ to a diagonal tensor in $\mathbb{R}^{I \times I \times I}$. Here p controls the rate of decay and we consider two cases: slow polynomial decay $p = 1$ and fast polynomial decay $p = 2$. The type of $\mathcal{A}$ comes from [57, 59].

We assume that $I = 400$. The results of Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, applied to the tensor $\mathcal{A}$ with different multilinear ranks $\{P, P, P\}$, are shown in Figures 6 and 7. From these two figures, in terms of CPU time, Tucker-SVD is the fastest one, and Tucker-pSVD is competitive with mlsvd_rsi and tucker_ALS. In terms of RLNE, Tucker-pSVD is competitive with other algorithms.

Example 5.5. Let $\mathbf{v} \in \mathbb{R}^I$ satisfy $\mathbf{v}(1:50) = \text{logspace}(0, t, 50)$ and $v_i = (i-49)^{-1}$ for all $i = 51, 52, \ldots, I$, where logspace$(x_1, x_2, 50)$ generates a row vector of 50 logarithmically equally spaced points between decades 10^{x_1} and 10^{x_2}. We construct the input tensor $\mathcal{A} \in \mathbb{R}^{I \times I \times I}$ as $\mathcal{A} = \text{tendiag}(\mathbf{v}, [I, I, I])$.

We assume that $I = 400$ and the multilinear rank is $\{50, 50, 50\}$. The results of Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi, applied to the tensor $\mathcal{A}$ with different t's, are shown in Figure 8, which shows that Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi suitable for tensors like $\mathcal{A}$.

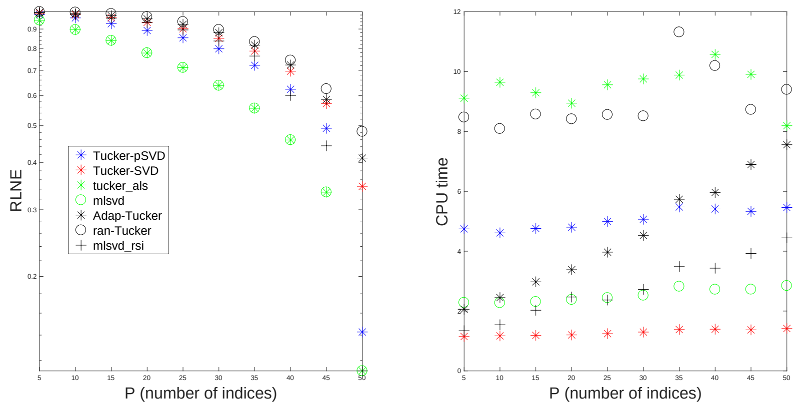

FIG. 6. *For Example* 5.4, *with slow polynomial decay* $p = 1$, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi to the tensor* $\mathcal{A}$ *with* $P = 5, 10, \ldots, 50$.

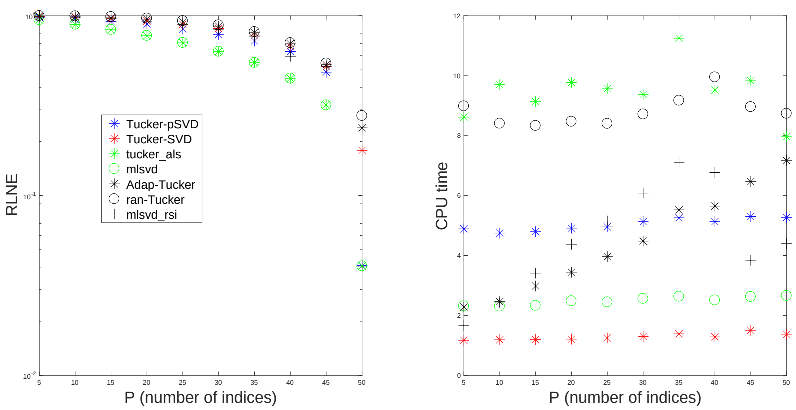

FIG. 7. *For Example* 5.4, *with fast polynomial decay* $p = 2$, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, Adap-Tucker, ran-Tucker, and mlsvd_rsi to the tensor* $\mathcal{A}$ *with* $P = 5, 10, \ldots, 50$.

5.2. Analysis of handwritten digit classification. In handwritten digits classification, we train a classification model to classify new unlabeled images. Savas and Eldén [51] presented two algorithms for handwritten digit classification based on HOSVD. To reduce the training time, a more efficient ST-HOSVD algorithm is presented in [61]. In this section, we compare the performance of Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, ran-Tucker, and mlsvd_rsi on the MNIST database [33],[5] which contains 60,000 training images and 10,000 test images. The 28×28 images have 8-bit grayscales. The digit distribution is given in Table 1. As seen in Table 1, the training images are unequally distributed over the ten classes. Therefore, we restricted the number of training images in every class to less than or equal to 5421.

[5]The database can be obtained from http://yann.lecun.com/exdb/mnist/.

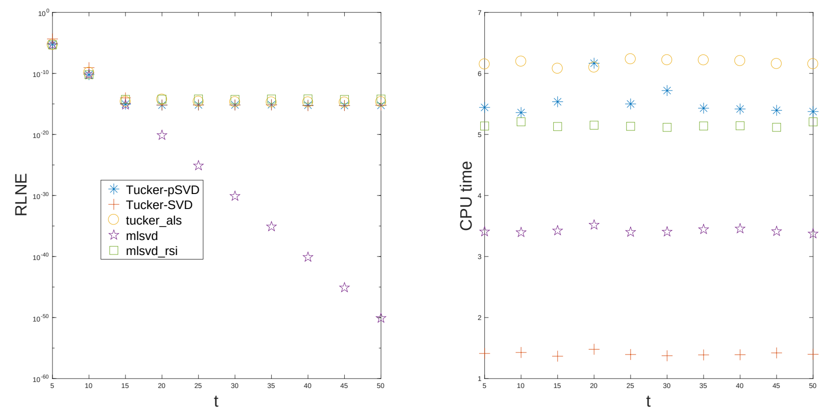

FIG. 8. *For Example* 5.5, *with given multilinear rank* $\{50, 50, 50\}$, *numerical simulation results of applying Tucker-pSVD, Tucker-SVD, tucker_ALS, mlsvd, and mlsvd_rsi to the tensor* $\mathcal{A}$ *with* $t = 5, 10, \ldots, 50$.

TABLE 1
The digit distribution in the MNIST data set.

	0	1	2	3	4	5	6	7	8	9	Total
Train	5923	6742	5958	6131	5842	5421	5918	6265	5851	5949	60000
Test	940	1135	1032	1010	982	892	958	1028	974	1009	10000

The classification data can be preprocessed in several ways [19]. Some kind of blurring and normalization are usual preprocessing techniques. Blurring is important for the identification process, at least when classifying handwritten digits. The blurring can be described as smoothing the pattern or making sharp edges and corners softer. Different kinds of blurring can be obtained depending on the function using in the operation. One of the usual functions is given by

$$g(x, y) = e^{-(x^2+y^2)/(2\sigma^2)},$$

where the standard deviation σ is used to control the amount of blurring. Some examples are given in Figure 9.

Denote the number of training images in every class by K with $K \leq 5421$. The training set is represented by a tensor $\mathcal{A}$ of size $786 \times K \times 10$. The first mode is the texel mode. The second mode corresponds to the training images. The third mode corresponds to different classes. The vector $\mathcal{A}(:, 10, 2)$ thus corresponds to the tenth image representing a two. The classification relies on [51, Algorithm 2]. We use various algorithms to obtain an approximation $\mathcal{A} \approx \mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}$, where the core tensor $\mathcal{G}$ has size $65 \times 142 \times 10$. For $K = 5000$, the results are summarized in Table 2.

From Table 2, in terms of CPU time, Tucker-SVD is the fastest one and tucker_ALS is the most expensive one; and in terms of classification accuracy, Tucker-pSVD, tucker_ALS, mlsvd, and mlsvd_rsi are comparable while Tucker-SVD is the least accurate one.

For different values of K, the results are summarized in Figure 10. From this figure, in terms of CPU time, tucker_ALS is the most expensive one. In terms of classification accuracy, Tucker-pSVD, mlsvd, and mlsvd_rsi are comparable.

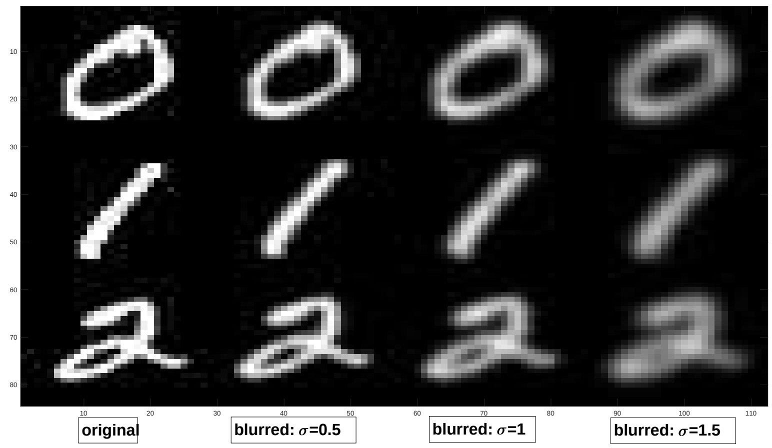

FIG. 9. *Examples of the effect of the Gaussian blurring operator with different deviations* σ.

TABLE 2
Comparison of handwritten digits classification.

	Training times [sec]	RLNE	Classification accuracy (%)
Tucker-pSVD	8.4780	0.3549	92.7080
Tucker-SVD	1.3210	0.4617	91.1400
tucker_ALS	59.2480	0.3205	93.3190
mlsvd	6.9450	0.3218	93.3600
Adap-Tucker	2.8090	0.4732	92.7160
ran-Tucker	3.3740	0.4682	92.7670
mlsvd_rsi	6.2560	0.4682	93.2820

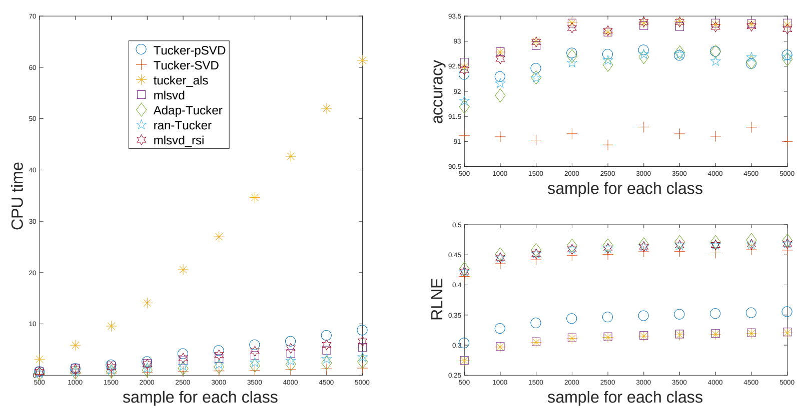

FIG. 10. *Comparison of handwritten digits classification with* $K = 500, 1000, \ldots, 5000$.

5.3. One more example. For mlsvd_rsi, we use mlsvd_rsi ($q = 0$), mlsvd_rsi ($q = 1$), and mlsvd_rsi ($q = 2$) to denote the case where the number of subspace iterations is 0, 1, and 2, respectively. In Tensorlab, mlsvd with "LargeScale" being true is denoted by mlsvd-LargeScale, and mlsvd with the MATLAB SVD routine is

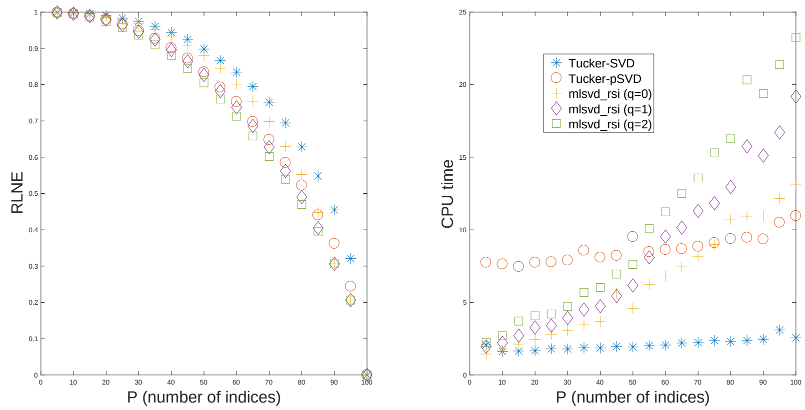

FIG. 11. *Numerical simulation results of applying Tucker-SVD, Tucker-pSVD, mlsvd_rsi* ($q = 0$)*, mlsvd_rsi* ($q = 1$)*, and mlsvd_rsi* ($q = 2$) *to the tensor* $\mathcal{A}$ *with* $P = 5, 10, \ldots, 100$.

denoted by mlsvd-SVD. Note that mlsvd-LargeScale is the same as mlsvd in the above two sections.

Example 5.6. Let $\mathcal{A} \in \mathbb{R}^{400\times400\times400}$ be given in the Tucker form $\mathcal{A} = \mathcal{G} \times_1 \mathbf{Q}_1 \times_2 \mathbf{Q}_2 \times_3 \mathbf{Q}_3$, where the entries of $\mathcal{G} \in \mathbb{R}^{100\times100\times100}$ and $\mathbf{Q}_n \in \mathbb{R}^{400\times100}$ $(n = 1, 2, 3)$ are i.i.d. Gaussian variables with zero mean and unit variance.

When we apply Tucker-SVD, Tucker-pSVD, mlsvd_rsi ($q = 0$), mlsvd_rsi ($q = 1$) and mlsvd_rsi ($q = 2$) to $\mathcal{A}$ with different $\{P, P, P\}$, RLNE and CPU time are shown in Figure 11. From this figure, in terms of RLNE, Tucker-pSVD is better than Tucker-SVD, and worse than mlsvd_rsi ($q = 0$), mlsvd_rsi ($q = 1$), and mlsvd_rsi ($q = 2$). In terms of CPU time, Tucker-SVD is the fastest, and when $P > 50$, Tucker-pSVD is faster than mlsvd_rsi ($q = 1$) and mlsvd_rsi ($q = 2$), and comparable to mlsvd_rsi ($q = 0$). Overall, Tucker-pSVD has a balanced good accuracy and high speed for large P values.

In the following, we give a simpler MATLAB implementation of ST-HOSVD, which is denoted by matlab_sthosvd.

```
function [G,Q1,Q2,Q3] = matlab_sthosvd(X,r1,r2,r3)
[I,J,K] = size(X); X1 = reshape(X,[I J*K]); S1 = X1*X1';
[Q1,E1] = eig(S1);
E1 = diag(E1); [~,ind] = sort(E1,'desc'); Q1 = Q1(:,ind(1:r1));
Y1 = Q1'*X1; Y1 = reshape(Y1,[r1 J K]);
Y2 = reshape(permute(Y1,[2 1 3]),[J r1*K]);
S2 = Y2*Y2'; [Q2,E2] = eig(S2); E2 = diag(E2);
[~,ind] = sort(E2,'desc'); Q2 = Q2(:,ind(1:r2)); Y2 = Q2'*Y2;
Y2 = permute(reshape(Y2,[r2 r1 K]),[2 1 3]);
Y3 = reshape(permute(Y2,[3 2 1]),[K r1*r2]);
S3 = Y3*Y3'; [Q3,E3] = eig(S3); E3 = diag(E3);
[~,ind] = sort(E3,'desc'); Q3 = Q3(:,ind(1:r3)); G3 = Q3'*Y3;
G = permute(reshape(G3,[r3 r2 r1]),[3 2 1]);
end
```

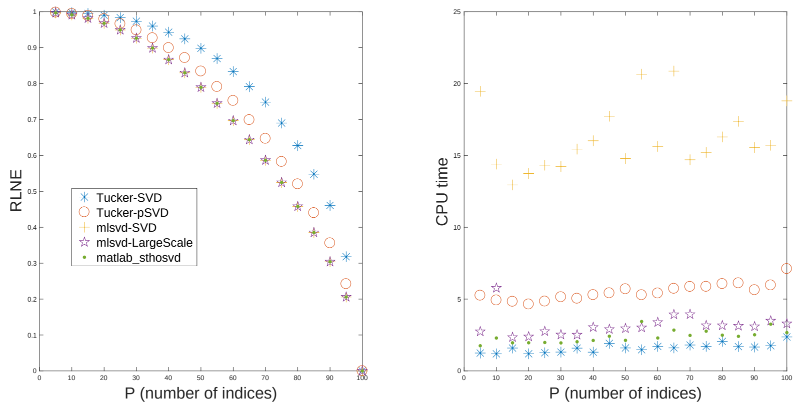

FIG. 12. *Numerical simulation results of applying Tucker-SVD, Tucker-pSVD, mlsvd-SVD, mlsvd-LargeScale, and matlab_sthosvd to the tensor $\mathcal{A}$ with $P = 5, 10, \ldots, 100$.*

When we apply Tucker-SVD, Tucker-pSVD, mlsvd-SVD, mlsvd-LargeScale, and matlab_sthosvd to $\mathcal{A}$ with different $\{P, P, P\}$'s, RLNE and CPU time are shown in Figure 12. This figure shows that in terms of CPU time, Tucker-SVD is the fastest, and Tucker-pSVD is slower than Tucker-SVD, mlsvd-LargeScale, and matlab_sthosvd, and faster than mlsvd-SVD. In terms of RLNE, Tucker-pSVD is better than Tucker-SVD, and worse than mlsvd-SVD, mlsvd-LargeScale, and matlab_sthosvd.

6. Conclusions and further considerations. In this paper, we proposed a randomized algorithm for the low multilinear rank approximation of a tensor with a given multilinear rank, which is based on power scheme, random projection, and SVD. Numerical examples illustrated that this algorithm is competitive with other algorithms, such as tucker_als, mlsvd, and mlsvd_rsi in terms of RLNE.

Our method can be further developed: (a) by combining our results and the work of [25], we can develop concentration results that give insight into the tail bounds; (b) by replacing standard Gaussian matrices by Rademacher random matrices, sparse Rademacher random matrices, subsampled randomized Fourier transforms, and so on, it is possible to combine the analysis with the probabilistic bounds for low multilinear rank approximations; and (c) in this paper, the code for Tucker-pSVD is only used to compute low multilinear rank approximation. We will explore whether there is a more efficient code for Tucker-pSVD in our future research.

Acknowledgment. The authors would like to thank the handling editor and two anonymous reviewers for their careful and very detailed comments on our paper.

REFERENCES

[1] B. W. BADER AND T. G. KOLDA, *Algorithm* 862: *MATLAB tensor classes for fast algorithm prototyping*, ACM Trans. Math. Software, 32 (2006), pp. 635–653.

[2] B. W. BADER, ET AL., *Tensor Toolbox for MATLAB Version,* 3.0-dev, https://www.tensortoolbox.org.

[3] C. BATTAGLINO, G. BALLARD, AND T. G. KOLDA, *A practical randomized CP tensor decomposition*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 876–901.

[4] D. BIAGIONI, D. J. BEYLKIN, AND G. BEYLKIN, *Randomized interpolative decomposition of separated representations*, J. Comput. Phys., 281 (2015), pp. 116–134.

[5] C. F. Caiafa and A. Cichocki, *Generalizing the column-row matrix decomposition to multi-way arrays*, Linear Algebra Appl., 433 (2010), pp. 557–573, https://doi.org/10.1016/j.laa.2010.03.020.

[6] T. F. Chan, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82, https://doi.org/10.1016/0024-3795(87)90103-0.

[7] S. Chandrasekaran and I. Ipsen, *On rank revealing QR factorizations*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 592–622.

[8] M. Che and Y. Wei, *Randomized algorithms for the approximations of Tucker and the tensor train decompositions*, Adv. Comput. Math., 45 (2019), pp. 395–428.

[9] M. Che and Y. Wei, *Theory and Computation of Complex Tensors and its Applications*, Springer, Singapore, 2020.

[10] M. Che, Y. Wei, and H. Yan, *Randomized Algorithms for the Low Multilinear Rank Approximations of Tensors*, preprint, arXiv:1908.11031, 2019.

[11] Z. Chen and J. J. Dongarra, *Condition numbers of Gaussian random matrices*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 603–620.

[12] A. Cichocki, *Tensor Networks for Big Data Analytics and Large-Scale Optimization Problems*, preprint, arXiv:1407.3124, 2014.

[13] A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley, Chichester, UK, 2009.

[14] L. De Lathauwer, B. De Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1253–1278, https://doi.org/10.1137/S0895479896305696.

[15] L. De Lathauwer, B. De Moor, and J. Vandewalle, *On the best rank-1 and rank-$(R_1, R_2, \ldots, R_n)$ approximation of higher-order tensors*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1324–1342.

[16] V. De Silva and L.-H. Lim, *Tensor rank and the ill-posedness of the best low-rank approximation problem*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1084–1127.

[17] P. Drineas and M. W. Mahoney, *A randomized algorithm for a tensor-based generalization of the singular value decomposition*, Linear Algebra Appl., 420 (2007), pp. 553–571, https://doi.org/10.1016/j.laa.2006.08.023.

[18] P. Drineas and M. W. Mahoney, *RandNLA: Randomized numerical linear algebra*, Commun. ACM, 59 (2016), pp. 80–90.

[19] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed., Wiley-Interscience, New York, 2012.

[20] L. Eldén and B. Savas, *A Newton–Grassmann method for computing the best multilinear rank-(r_1, r_2, r_3) approximation of a tensor*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 248–271.

[21] M. P. Friedlander and K. Hatz, *Computing non-negative tensor factorizations*, Optim. Methods Softw., 23 (2008), pp. 631–647, https://doi.org/10.1080/10556780801996244.

[22] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed., Johns Hopkins University Press, Baltimore, MD, 2013.

[23] S. A. Goreinov, I. V. Oseledets, and D. V. Savostyanov, *Wedderburn rank reduction and Krylov subspace method for tensor approximation. Part 1: Tucker case*, SIAM J. Sci. Comput., 34 (2012), pp. A1–A27, https://doi.org/10.1137/100792056.

[24] S. A. Goreinov and E. E. Tyrtyshnikov, *The maximal-volume concept in approximation by low-rank matrices*, in Structured Matrices in Mathematics Computer Science and Engineering I, AMS, Providence, RI, 2001, pp. 47–51.

[25] M. Gu, *Subspace iteration randomization and singular value problems*, SIAM J. Sci. Comput., 37 (2015), pp. A1139–A1173.

[26] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.

[27] W. Hackbusch, *Numerical tensor calculus*, Acta Numer., 23 (2014), pp. 651–742.

[28] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288, https://doi.org/10.1137/090771806.

[29] Y. P. Hong and C. T. Pan, *Rank revealing QR factorizations and the singular value decompositions*, Math. Comp., 58 (1992), pp. 213–232.

[30] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer, *Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 115–135, https://doi.org/10.1137/090764827.

[31] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500, https://doi.org/10.1137/07070111X.

[32] J. Kuczyński and H. Woźniakowski, *Estimating the largest eigenvalues by the power and Lanczos algorithms with a random start*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 1094–1122.
[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proc. IEEE, 86 (1998), pp. 2278–2324.
[34] A. E. Litvak, A. Pajor, M. Rudelson, and N. Tomczak-Jaegermann, *Smallest singular value of random matrices and geometry of random polytopes*, Adv. Math., 195 (2005), pp. 491–523.
[35] A. E. Litvak and O. Rivasplata, *Smallest singular value of sparse random matrices*, Stud. Math., 212 (2012), pp. 195–218.
[36] L. S. Lorente, J. M. Vega, and A. Velazquez, *Compression of aerodynamic databases using high-order singular value decomposition*, Aerosp. Sci. Technol., 14 (2010), pp. 168–177.
[37] L. S. Lorente, J. M. Vega, and A. Velazquez, *Generation of aerodynamic databases using high-order singular value decomposition*, J. Aircraft, 45 (2010), pp. 1779–1788.
[38] M. W. Mahoney, *Randomized algorithms for matrices and data*, Found. Trends Mach. Learn., 3 (2011), pp. 123–224.
[39] M. W. Mahoney, M. Maggioni, and P. Drineas, *Tensor-CUR decompositions for tensor-based data*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 957–987, https://doi.org/10.1137/060665336.
[40] P.-G. Martinsson and S. Voronin, *A randomized blocked algorithm for efficiently computing rank-revealing factorizations of matrices*, SIAM J. Sci. Comput., 38 (2016), pp. S485–S507.
[41] P.-G. Martinsson, V. Rokhlin, and M. Tygert, *A randomized algorithm for the decomposition of matrices*, Appl. Comput. Harmon. Anal., 30 (2011), pp. 47–68.
[42] C. Navasca and L. De Lathauwer, *Low multilinear rank tensor approximation via semidefinite programming*, in IEEE 17th European Signal Processing Conference, IEEE, Piscataway, NJ, 2009, pp. 520–524.
[43] C. Navasca and D. N. Pompey, *Random projections for low multilinear rank tensors*, in Visualization and Processing of Higher Order Descriptors for Multi-Valued Data, I. Hotz and T. Schultz, eds., Springer, Cham, Switzerland, 2015, pp. 93–106.
[44] 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, https://doi.org/10.1137/060655894.
[45] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov, *Cross approximation in tensor electron density computations*, Numer. Linear Algebra Appl., 17 (2010), pp. 935–952, https://doi.org/10.1002/nla.682.
[46] C. T. Pan, *On the existence and computation of rank-revealing LU factorizations*, Linear Algebra Appl., 316 (2000), pp. 199–222.
[47] C. T. Pan and P. T. Tang, *Bounds on singular values revealed by QR factorization*, BIT, 39 (1999), pp. 740–756.
[48] V. Rokhlin, A. Szlam, and M. Tygert, *A randomized algorithm for principal component analysis*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1100–1124.
[49] M. Rudelson and R. Vershynin, *Smallest singular value of a random rectangular matrix*, Commun. Pure Appl. Math., 62 (2009), pp. 1707–1739.
[50] A. K. Saibaba, *HOID: Higher order interpolatory decomposition for tensors based on Tucker representation*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 1223–1249, https://doi.org/10.1137/15M1048628.
[51] B. Savas and L. Eldén, *Handwritten digit classification using higher order singular value decomposition*, Pattern Recogn., 40 (2007), pp. 993–1003.
[52] B. Savas and L. Eldén, *Krylov-type methods for tensor computations* I, Linear Algebra Appl., 438 (2013), pp. 891–918, https://doi.org/10.1016/j.laa.2011.12.007.
[53] B. Savas and L.-H. Lim, *Quasi-Newton methods on Grassmannians and multilinear approximations of tensors*, SIAM J. Sci. Comput., 32 (2010), pp. 3352–3393, https://doi.org/10.1137/090763172.
[54] G. Shabat, Y. Shmueli, Y. Aizenbud, and A. Averbuch, *Randomized LU decomposition*, Appl. Comput. Harmon. Anal., 44 (2016), pp. 246–272.
[55] D. C. Sorensen and M. Embree, *A DEIM induced CUR factorization*, SIAM J. Sci. Comput., 38 (2016), pp. A1454–A1482.
[56] G. W. Stewart, *Four algorithms for the the [sic] efficient computation of truncated pivoted QR approximations to a sparse matrix*, Numer. Math., 83 (1999), pp. 313–323.
[57] Y. Sun, Y. Guo, C. Luo, J. A. Tropp, and M. Udell, *Low-Rank Tucker Approximation of a Tensor from Streaming Data*, preprint, arXiv:1904.10951v1, 2019.

[58] P. Symeonidis, *ClustHOSVD: Item recommendation by combining semantically enhanced tag clustering with tensor HOSVD*, IEEE Trans. Syst. Man Cybern. Syst., 46 (2016), pp. 1240–1251.

[59] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Practical sketching algorithms for low-rank matrix approximation*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1454–1485.

[60] L. R. Tucker, *Some mathematical notes on three-mode factor analysis*, Psychometrika, 31 (1966), pp. 279–311.

[61] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, *A new truncation strategy for the higher-order singular value decomposition*, SIAM J. Sci. Comput., 34 (2012), pp. A1027–A1052, https://doi.org/10.1137/110836067.

[62] M. Vasilescu and D. Terzopoulos, *Multilinear image analysis for facial recognition*, in 16th International Conference on Pattern Recognition, Vol. 2, IEEE Computer Society, Los Alamitos, CA, 2002, pp. 511–514.

[63] M. Vasilescu and D. Terzopoulos, *Multilinear subspace analysis of image ensembles*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, IEEE Computer Society, Los Alamitos, CA, 2003, pp. 93–99.

[64] M. Vasilescu and D. Terzopoulos, *Tensor textures: Multilinear image-based rendering*, ACM Trans. Graph., 23 (2004), pp. 336–342.

[65] N. Vervliet and L. De Lathauwer, *A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors*, IEEE J. Sel. Topics Signal Process., 10 (2016), pp. 284–295.

[66] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer, *Tensorlab* 3.0, http://tensorlab.net.

[67] G. Wang, Y. Wei, and S. Qiao, *Generalized Inverses: Theory and Computations.*, Dev. Math. 53, Springer, Singapore, 2018, https://doi.org/10.1007/978-981-13-0146-9.

[68] D. P. Woodruff, *Sketching as a tool for numerical linear algebra*, Found. Trends Theoret. Comput. Sci., 10 (2014), pp. 1–157.

[69] W. Yu, Y. Gu, and Y. Li, *Efficient randomized algorithms for the fixed-precision low-rank matrix approximation*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 1339–1359.

[70] Y. Zhang, G. Zhou, Q. Zhao, A. Cichocki, and X. Wang, *Fast nonnegative tensor factorization based on accelerated proximal gradient and low-rank approximation*, Neurocomputing, 198 (2016), pp. 148–154.

[71] G. Zhou, A. Cichocki, and S. Xie, *Fast nonnegative matrix/tensor factorization based on low-rank approximation*, IEEE Trans. Signal Process., 60 (2012), pp. 2928–2940.

[72] G. Zhou, A. Cichocki, and S. Xie, *Decomposition of Big Tensors with Low Multilinear Rank*, preprint, arXiv:1412.1885v1, 2014.

[73] G. Zhou, A. Cichocki, Q. Zhao, and S. Xie, *Efficient nonnegative Tucker decompositions: Algorithms and uniqueness*, IEEE Trans. Image Process., 24 (2015), pp. 4990–5003.