SIAM J. MATRIX ANAL. APPL.
Vol. 42, No. 2, pp. 832–858

INCREMENTAL CP TENSOR DECOMPOSITION BY ALTERNATING MINIMIZATION METHOD*

CHAO ZENG† AND MICHAEL K. NG†

Abstract. In practical applications, incremental tensors are very common: only a portion of tensor data is available, and new data are arriving in the next time step or continuously over time. To handle this type of tensors time-saving algorithms are required for online computation. In this paper, we consider incremental CP (CANDECOMP/PARAFAC) decomposition, which requires one to update the CP decomposition after new tensor data, together with the exiting tensor, are ready for analysis. There exist several incremental CP decomposition algorithms, but almost all of these algorithms assume that the number of CP decomposition components remains fixed in the incremental process. The main contribution of this paper is the study of how to add components in the incremental CP decomposition. We derive the coordinate representation of the incremental CP decomposition with respect to a special basis and show related properties of tensor rank and the uniqueness of such incremental CP decomposition. Under the framework of this representation, the proposed method can be solved by using an alternating minimization algorithm. Numerical examples are presented to show the good performance of the proposed algorithms in terms of computational time and data fitting compared with existing methods.

Key words. incremental tensor, CP decomposition, alternating minimization

AMS subject classifications. 15A69, 90C25, 90C30, 65K10

DOI. 10.1137/20M1319097

1. Introduction.

The CANDECOMP/PARAFAC (CP) decomposition [10, 20] is an important tool for high-order data analysis. It decomposes a tensor as a sum of rank-one tensors and can be regarded as a generalization of matrix singular value decomposition (SVD). Unlike matrix SVD, CP decomposition is essentially unique under mild conditions [13, 25, 28, 38, 44]. CP decomposition is widely used in signal processing [39, 54], data mining [31], machine learning [40], and chemometrics [41]. The interested reader can refer to [11, 12, 24] for a thorough review. In general, it is NP-hard to compute the tensor rank, which is the smallest number of components (see section 2.2 for the definition) in the CP decomposition [21, 22]. The standard computational method is to prescribe a fixed number of components in the CP decomposition, and then to determine the decomposition via the alternating least squares (ALS) method; see [10, 20, 24]. This method has been extended to randomized versions [3, 35]. In the literature, many other algorithms [45, 49] have been proposed and developed to improve the CP decomposition results.

Tensor decompositions are usually studied and analyzed under batch mode, i.e., whole tensor data are available. In practical applications, we encounter many cases [47] where only partial data sets are known at first, and the new data sets are available in the next time step or are arriving continuously over time. Examples include live video stream, surveillance video, network flow, and social media data. Such tensors are called tensor streams or incremental tensors; see, for instance, [46, 47]. It is

*Received by the editors February 14, 2020; accepted for publication (in revised form) by B. Hashemi February 22, 2021; published electronically June 16, 2021.
https://doi.org/10.1137/20M1319097
Funding: The work of the second author was partially supported by the HKRGC GRF grants 12306616, 12200317, 12300218, 12300519, and 17201020.
†Department of Mathematics, The University of Hong Kong, Pokfulam, Hong Kong (zengchao@nankai.edu.cn, mng@maths.hku.hk).

interesting—and necessary—to develop incremental algorithms for decompositions of such incremental tensors. The main reason is because when the initial decomposition is already known, the incremental decomposition can be updated more efficiently than recalculation of the whole decomposition from scratch.

Incremental algorithms for CP decompositions have been studied in several papers. We will give a detailed review of these works in section 6. Almost all existing works assume that the number of components is fixed in the incremental process. However, when new data are different from old data, the fixed number of components may not be sufficient to cope with the variation between the old and new data. Therefore, the error between the whole tensor data and the updated CP decomposition may be large. For example, in the driving recorder dataset in Figure 1, objects disappear or appear, and the background is also changing in the new image frame. We may set a large enough number of components at the initialization stage, but there is no guarantee of a good data fitting for the incremental tensor. Also the computational cost would be increased. In section 5.3, we test some incremental tensors with rapidly increasing rank and find that OnlineCP [55] (one of the state-of-the-art algorithms) perform poorly in terms of the fitting error even when the initial number of components is large.

FIG. 1. *Two frames of a driving recorder dataset.*

The main contribution of this paper is to design an incremental CP decomposition algorithm for the case where the number of components is not fixed. We view the CP decomposition of a tensor as the coordinate representation of its $(N-1)$st-order subtensors with respect to a special rank-one basis (see Definition 3.5). By using such rank-one basis, we discuss the rank and the uniqueness of the CP decomposition of the new tensor. Moreover, by using these developed results and properties, we see that the incremental algorithm can be designed by updating the rank-one basis in the representation. New rank-one components are added by extending the rank-one basis. Numerical examples are presented to demonstrate that the proposed algorithm can yield results comparable to the standard algorithm ALS, while being computationally much more efficient, and can obtain much better results than state-of-the-art algorithm, such as OnlineCP.

The rest of this paper is organized as follows. In section 2, we introduce some basic definitions and review the CP decomposition, ALS and incremental ALS. In section 3, we define the rank-one basis and present some properties of the incremental CP decomposition. The algorithm based on the rank-one basis is proposed in section 4. Experimental results are given in section 5. In section 6, we review related works in detail. Conclusions are presented in section 7.

2. Preliminaries.

2.1. Notation and definitions. We use boldface lowercase letters $(\mathbf{a}, \mathbf{b}, \ldots)$ to denote vectors, boldface uppercase letters $(\mathbf{A}, \mathbf{B}, \ldots)$ to denote matrices, and calligraphic letters $(\mathcal{X}, \mathcal{Y}, \ldots)$ to denote higher-order tensors (order three or higher). The $(i_1, i_2, \ldots, i_N)$th entry of an Nth-order tensor $\mathcal{X}$ is denoted by $x_{i_1 i_2 \cdots i_N}$.

The *mode-n fibers* of a tensor are the higher-order analogue of matrix column and row vectors. The n-rank of a tensor $\mathcal{X}$, denoted by $\text{rank}_n(\mathcal{X})$, is the dimension of the vector space spanned by all mode-n fibers. A tensor can be vectorized into a vector with specific ordering. We adopt the following simple ordering in MATLAB:

$$\text{vec}(\mathcal{X}) := \mathcal{X}(:).$$

A tensor can also be unfolded into a matrix. The mode-n unfolding matrix of $\mathcal{X}$ is denoted by $\mathbf{X}_{(n)}$ and arranges the mode-n fibers as columns of the resulting matrix, where tensor entry $x_{i_1 i_2 \cdots i_N}$ maps to entry (i_n, j) of $\mathbf{X}_{(n)}$ via the relation

$$j = 1 + \sum_{k=1, k\neq n}^{N} (i_k - 1) J_k, \text{ where } J_k = \prod_{m=1, m\neq n}^{k-1} I_m.$$

Given matrices $\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_J \end{bmatrix} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_L \end{bmatrix} \in \mathbb{R}^{K \times L}$, if $J = L$, their *Khatri–Rao product* is defined by

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_J \otimes \mathbf{b}_J \end{bmatrix},$$

where "$\otimes$" denotes the Kronecker product: $\mathbf{a} \otimes \mathbf{b} = [a_1 b_1 \cdots a_1 b_K \cdots a_I b_1 \cdots a_I b_K]^T$. If $I = K$ and $J = L$, their *Hadamard product* is $\mathbf{A} \circledast \mathbf{B} \in \mathbb{R}^{I \times J}$, the entrywise product of the matrices. The following identity is useful in our discussions:

$$(\mathbf{A} \odot \mathbf{B})^T (\mathbf{C} \odot \mathbf{D}) = \mathbf{A}^T \mathbf{C} \circledast \mathbf{B}^T \mathbf{D}. \tag{1}$$

The Frobenius norm of a matrix $\mathbf{A}$ is denoted by $\|\mathbf{A}\|$, and the Frobenius norm of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is given by

$$\|\mathcal{X}\| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N}^2}.$$

2.2. CANDECOMP/PARAFAC decompositions. We follow the terminology from [24]. The outer product of N vectors $\mathbf{a}^{(n)} \in \mathbb{R}^{I_n}$ is an Nth-order tensor

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \cdots \circ \mathbf{a}^{(N)} \in \mathbb{R}^{I_1 \times \cdots \times I_N},$$

where

$$x_{i_1, i_2 \cdots i_N} = a_{i_1}^{(1)} a_{i_2}^{(2)} \cdots a_{i_N}^{(N)} \quad \text{for all } 1 \leq i_n \leq I_n.$$

If an Nth-order tensor can be written as the outer product of N vectors, we say that it is *rank-one*.

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The CP decomposition aims to factorize a tensor into a sum of component rank-one tensors,

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)}, \tag{2}$$

where $\mathbf{a}_r^{(n)} \in \mathbb{R}^{I_n}$ is called a *factor vector*. Each rank-one tensor is called a *component*. The following collection of all factor vectors for a given mode is called a *factor matrix*:

$$\mathbf{A}^{(n)} = \begin{bmatrix} \mathbf{a}_1^{(n)} & \mathbf{a}_2^{(n)} \cdots & \mathbf{a}_R^{(n)} \end{bmatrix}.$$

Model (2) can also be expressed as

$$\mathcal{X} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]. \tag{3}$$

We can alternatively represent (2) and (3) by multiplying each component by a scalar weight,

$$\mathcal{X} = [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] = \sum_{r=1}^{R} \lambda_r \ \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)}, \tag{4}$$

where $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_R]^T$. Usually, this form is obtained by normalizing all the factor vectors in (2) to unit length and expressing the product of the normalization factors as a scalar weight λ_r for each component, i.e., $\|\mathbf{a}_r^{(n)}\| = 1$ for $n = 1, \ldots, N, r = 1, \ldots, R$ in (4).

Clearly $\mathcal{X}$ has many CP decompositions. The *rank* of $\mathcal{X}$ is defined as the smallest R existing in (2). It is well known that the calculation of tensor rank is NP-hard [21, 22]. For (2), if $R = \text{rank}(\mathcal{X})$, then it is called the *rank decomposition* [24].

DEFINITION 2.1 (see [44]). *The decomposition* $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ *is called unique up to permutation and scaling if any alternative decomposition* $[\![\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(N)}]\!]$ *satisfies* $\mathbf{B}^{(n)} = \mathbf{A}^{(n)}\boldsymbol{\Pi}\boldsymbol{\Lambda}^{(n)}, n = 1, \ldots, N$, *with* $\boldsymbol{\Pi}$ *an* $R \times R$ *permutation matrix, and* $\boldsymbol{\Lambda}^{(n)}$ *nonsingular diagonal matrices such that* $\Pi_{n=1}^{N}\boldsymbol{\Lambda}^{(n)} = \mathbf{I}$.

If a CP decomposition is unique, then it must be a rank decomposition. This is because when $R > \text{rank}(\mathcal{X})$, we can generate many CP decompositions with R terms from a rank decomposition by decomposing one factor vector into $R - \text{rank}(\mathcal{X}) + 1$ terms.

The most famous results [25, 38] on the uniqueness condition depend on the concept of k-rank. The k-rank of a matrix $\mathbf{A}$, denoted by $k_{\mathbf{A}}$, is the largest integer such that every set containing $k_{\mathbf{A}}$ columns of $\mathbf{A}$ is linearly independent. For the CP decomposition (3), its uniqueness condition presented in [38] is

$$\sum_{n=1}^{N} k_{\mathbf{A}^{(n)}} \geq 2R + N - 1. \tag{5}$$

2.3. Alternating least squares. Given $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, fitting the CP decomposition of $\mathcal{X}$ with R components can be formulated as

$$\min_{\{\mathbf{A}^{(n)}\}_{n=1}^{N}} \left\| \mathcal{X} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] \right\|.$$

The ALS method is a block nonlinear Gauss–Seidel approach. For each iteration, $\mathbf{A}^{(n)}$ is found by solving the linear least squares problem given by

$$\min_{\mathbf{A}^{(n)}} \left\| \mathcal{X} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] \right\|,$$

whose matrix form is

$$\min_{\mathbf{A}^{(n)}} \left\| \mathbf{X}_{(n)} - \mathbf{A}^{(n)}\mathbf{H}^{(n)^T} \right\|, \tag{6}$$

where $\mathbf{H}^{(n)} = \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}$. The normal equation for (6) is

$$\mathbf{X}_{(n)}\mathbf{H}^{(n)} = \mathbf{A}^{(n)}(\mathbf{H}^{(n)^T}\mathbf{H}^{(n)}),$$

where $\mathbf{H}^{(n)^T}\mathbf{H}^{(n)} = \circledast_{m \neq n}\mathbf{A}^{(m)^T}\mathbf{A}^{(m)}$ by (1).

2.4. Incremental ALS. In numerical computation, the term "decomposition" is often used when "approximation" is meant instead. That is, (2) and (3) have the form

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!], \tag{7}$$

where R is a prescribed integer. By appending a new *slice* $\mathcal{Z} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{N-1}}$, we obtain a new tensor $\hat{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times (I_N+1)}$ with $\hat{\mathcal{X}}(:, \ldots, :, 1 : I_N) = \mathcal{X}$ and $\hat{\mathcal{X}}(:, \ldots, :, I_N + 1) = \mathcal{Z}$. We want to obtain a CP decomposition of $\hat{\mathcal{X}}$ based on the CP decomposition of $\mathcal{X}$. The resulting decomposition is called an *incremental CP decomposition* of $\hat{\mathcal{X}}$.

Suppose the incremental CP decomposition of $\hat{\mathcal{X}}$ has $\hat{R}$ components, and denote by $[\![\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N)}]\!]$ the decomposition that we want to find, where $\hat{\mathbf{A}}^{(n)} \in \mathbb{R}^{I_n \times \hat{R}}$ for $n = 1, \ldots, N-1$ and $\hat{\mathbf{A}}^{(N)} = \left[\begin{smallmatrix} \mathbf{C} \\ \mathbf{v}^T \end{smallmatrix}\right]$ with $\mathbf{C} \in \mathbb{R}^{I_N \times \hat{R}}$ and $\mathbf{v} \in \mathbb{R}^{\hat{R}}$. In [42, 51], $\mathcal{X}$ is replaced by $\tilde{\mathcal{X}}$ as follows for computing the incremental CP decomposition of $\hat{\mathcal{X}}$:

$$\begin{aligned} \min_{\{\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}} \mathscr{F}(\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N)}) := & \left\| \mathcal{Z} - [\![\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N-1)}, \mathbf{v}^T]\!] \right\|^2 \\ & + \left\| [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N-1)}, \mathbf{A}^{(N)}]\!] - [\![\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N-1)}, \mathbf{C}]\!] \right\|^2. \end{aligned} \tag{8}$$

This replacement avoids storage of the old tensor $\mathcal{X}$ and thus save space cost. In addition, such a replacement can save time cost when using ALS, which will be shown in the following. Actually, the analogous replacement has been used in incremental matrix SVD [6].

Define

$$\begin{aligned} \mathbf{W}^{(n)} =& \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}, \\ \hat{\mathbf{W}}^{(n)} =& \hat{\mathbf{A}}^{(N-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(n+1)} \odot \hat{\mathbf{A}}^{(n-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(1)}. \end{aligned}$$

By (6), when we use ALS to solve (8), $A^{(n)} (n = 1, \ldots, N-1)$ is solved by

$$\min_{\hat{\mathbf{A}}^{(n)}} \left\| \left[\mathbf{A}^{(n)} \left(\mathbf{A}^{(N)} \odot \mathbf{W}^{(n)} \right)^T \quad \mathbf{Z}_{(n)} \right] - \hat{\mathbf{A}}^{(n)} \left(\begin{bmatrix} \mathbf{C} \\ \mathbf{v}^T \end{bmatrix} \odot \hat{\mathbf{W}}^{(n)} \right)^T \right\|,$$

whose normal equation is

$$\begin{aligned} & \hat{\mathbf{A}}^{(n)} \left(\hat{\mathbf{A}}^{(N)^T} \hat{\mathbf{A}}^{(N)} \circledast \hat{\mathbf{W}}^{(n)^T} \hat{\mathbf{W}}^{(n)} \right) = \hat{\mathbf{A}}^{(n)} \left(\left(\mathbf{C}^T \mathbf{C} + \mathbf{v}\mathbf{v}^T \right) \circledast \hat{\mathbf{W}}^{(n)^T} \hat{\mathbf{W}}^{(n)} \right) \\ & = \mathbf{A}^{(n)} \left(\mathbf{A}^{(N)^T} \mathbf{C} \circledast \mathbf{W}^{(n)^T} \hat{\mathbf{W}}^{(n)} \right) + \mathbf{Z}_{(n)} \left(\mathbf{v}^T \odot \hat{\mathbf{W}}^{(n)} \right); \end{aligned} \tag{9}$$

for $n = N$, $\hat{\mathbf{A}}^{(N)}$ is solved by

$$\min_{\hat{\mathbf{A}}^{(n)}} \left\| \begin{bmatrix} \mathbf{A}^{(N)} \mathbf{H}^{(N)^T} \\ \mathrm{vec}(\mathcal{Z})^T \end{bmatrix} - \begin{bmatrix} \mathbf{C}\hat{\mathbf{H}}^T \\ \mathbf{v}^T \hat{\mathbf{H}}^T \end{bmatrix} \right\|,$$

where $\mathbf{H}^{(N)}$ is defined as in (6) and $\hat{\mathbf{H}} = \hat{\mathbf{A}}^{(N-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(1)}$. The normal equation of the above problem is

$$\hat{\mathbf{A}}^{(N)} (\hat{\mathbf{H}}^T \hat{\mathbf{H}}) = \begin{bmatrix} \mathbf{C}(\hat{\mathbf{H}}^T \hat{\mathbf{H}}) \\ \mathbf{v}^T (\hat{\mathbf{H}}^T \hat{\mathbf{H}}) \end{bmatrix} = \begin{bmatrix} \mathbf{A}^{(N)} (\mathbf{H}^{(N)^T} \hat{\mathbf{H}}) \\ \mathrm{vec}(\mathcal{Z})^T \hat{\mathbf{H}} \end{bmatrix}. \tag{10}$$

By (1), we have

$$\hat{\mathbf{W}}^{(n)^T}\hat{\mathbf{W}}^{(n)} = \circledast_{m=1,m\neq n}^{N-1}\hat{\mathbf{A}}^{(m)^T}\hat{\mathbf{A}}^{(m)}, \quad \mathbf{W}^{(n)^T}\hat{\mathbf{W}}^{(n)} = \circledast_{m=1,m\neq n}^{N-1}\mathbf{A}^{(m)^T}\hat{\mathbf{A}}^{(m)},$$
$$\hat{\mathbf{H}}^T\hat{\mathbf{H}} = \circledast_{m=1}^{N-1}\hat{\mathbf{A}}^{(m)^T}\hat{\mathbf{A}}^{(m)}, \quad \mathbf{H}^{(N)^T}\hat{\mathbf{H}} = \circledast_{m=1}^{N-1}\mathbf{A}^{(m)^T}\hat{\mathbf{A}}^{(m)}.$$

In the later discussion of this part, we define

$$S = \hat{R} - R.$$

Since $\mathcal{X}$ is a subtensor of $\hat{\mathcal{X}}$, we have $\text{rank}(\mathcal{X}) \leq \text{rank}(\hat{\mathcal{X}})$, and S needs to be nonnegative. We will introduce how to use ALS to solve (8) for $S = 0$ and $S > 0$, respectively, in the following cases.

2.4.1. The case $S = 0$. The idea of using ALS to solve (8) for $S = 0$ can be found in CP-stream [42]. CP-stream is presented in a more general setting; see [42] for details. We introduce the algorithm for our setting.

We know that the starting point is very important for ALS. For (8), a good starting point for $\hat{\mathbf{A}}^{(n)}$ is $\mathbf{A}^{(n)}$, $n = 1, \ldots, N-1$. By using this initial guess for (10), the solution of $\mathbf{C}$ is just $\mathbf{A}^{(N)}$, and $\mathbf{v}$ is solved by

$$(\mathbf{H}^{(N)^T}\mathbf{H}^{(N)})\mathbf{v} = \mathbf{H}^{(N)^T}\text{vec}(\mathcal{Z}). \tag{11}$$

After obtaining $\mathbf{C}$ and $\mathbf{v}$, we update $\hat{\mathbf{A}}^{(N)^T}\hat{\mathbf{A}}^{(N)}$ as $\mathbf{A}^{(N)^T}\mathbf{A}^{(N)} + \mathbf{v}\mathbf{v}^T$, and $\hat{\mathbf{A}}^{(n)}$ for $n = 1, \ldots, N-1$ can be computed by (9) as in standard ALS.

For our setting, only one iteration of ALS is needed. This means that after obtaining $\hat{\mathbf{A}}^{(n)}$ for $n = 1, \ldots, N-1$ by (9), we do not update $\hat{\mathbf{A}}^{(N)}$ by (10) again. The main reason is because of the time cost. If more iterations are run, the computation cost of some terms in (9) and (10) incorporating $\mathbf{A}^{(N)}$ or $\hat{\mathbf{A}}^{(N)}$ would increase as more slices are appending. This is not feasible for long-term computation. It was demonstrated by numerical results [42, 55] that if the rank of an incremental tensor does not change significantly, one iteration of ALS is adequate for good data fitting.

We call this algorithm *incremental ALS* for $S = 0$ (iALS($S = 0$) for short).

2.4.2. The case $S > 0$. We can extend the above idea to the case $S > 0$. We call this algorithm *incremental ALS* for $S > 0$ (iALS($S > 0$) for short). We make use of the following lemma to generate a reasonable initial guess.

LEMMA 2.2. *Let* $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times \hat{R}}$ *satisfy* $\mathbf{U}^{(n)}(:, 1:R) = \mathbf{A}^{(n)}$ *for* $n = 1, \ldots, N-1$. *Then for* $\mathscr{F}$ *defined in* (8), *one has*

$$\min_{\mathbf{U}} \mathscr{F}(\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}, \mathbf{U}) \leq \min_{\mathbf{A}} \mathscr{F}(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N-1)}, \mathbf{A}).$$

Proof. For any $\bar{\mathbf{A}} \in \arg\min_{\mathbf{A}\in\mathbb{R}^{(I_N+1)\times R}} \mathscr{F}(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N-1)}, \mathbf{A})$, define a matrix $\bar{\mathbf{U}}$ satisfying $\bar{\mathbf{U}}(:, 1:R) = \bar{\mathbf{A}}$ and $\bar{\mathbf{U}}(:, R+1:\hat{R}) = 0$. It can be verified that

$$\begin{aligned}&\mathscr{F}(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N-1)}, \bar{\mathbf{A}}) = \mathscr{F}(\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}, \bar{\mathbf{U}})\\ &\geq \min_{\mathbf{U}} \mathscr{F}(\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}, \mathbf{U}).\end{aligned}$$

∎

By the above lemma, we can set a semihot starting point for ALS. That is, we set $\hat{\mathbf{A}}^{(n)}(:, 1:R) = \mathbf{A}^{(n)}$, generate $\hat{\mathbf{A}}^{(n)}(:, R+1:\hat{R})$ randomly for $n = 1, \ldots, N-1$, and use them as the starting point of ALS. Then, $\hat{\mathbf{A}}^{(N)}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N-1)}$ are updated by

Algorithm 1: Incremental ALS for $S > 0$ (iALS($S > 0$)).

Input: $\mathcal{Z}, \{\mathbf{A}^{(n)}\}_{n=1}^{N}, \{\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}\}_{n=1}^{N}, S$
Output: $\{\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}$
1 **for** $n = 1, \ldots, N-1$ **do**
2 $\quad \hat{\mathbf{A}}^{(n)}(:, 1:R) \leftarrow \mathbf{A}^{(n)}$ and generate $\hat{\mathbf{A}}^{(n)}(:, R+1:R+S)$ randomly
3 **end**
4 **repeat**
5 $\quad \mathbf{D}_1 \leftarrow \circledast_{m=1}^{N-1}\mathbf{A}^{(m)^T}\hat{\mathbf{A}}^{(m)}$
6 $\quad \mathbf{D}_1 \leftarrow \mathbf{A}^{(N)}\mathbf{D}_1$
7 $\quad \hat{\mathbf{V}} = \hat{\mathbf{A}}^{(N-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(1)}$
8 $\quad \mathbf{D}_2 \leftarrow \mathrm{vec}(\mathcal{Z})^T\hat{\mathbf{V}}$
9 $\quad \mathbf{G} \leftarrow \circledast_{m=1}^{N-1}\hat{\mathbf{A}}^{(m)^T}\hat{\mathbf{A}}^{(m)}$
10 $\quad$ Solve $\hat{\mathbf{A}}^{(N)}\mathbf{G} = \begin{bmatrix}\mathbf{D}_1\\ \mathbf{D}_2\end{bmatrix}$ for $\hat{\mathbf{A}}^{(N)}$
11 $\quad \mathbf{C} \leftarrow \hat{\mathbf{A}}^{(N)}(1:I_N, :), \mathbf{v}^T \leftarrow \hat{\mathbf{A}}^{(N)}(I_N+1, :)$
12 $\quad$ **for** $n = 1, \ldots, N-1$ **do**
13 $\quad\quad \mathbf{D}_1 \leftarrow \mathbf{A}^{(N)^T}\mathbf{C} \circledast \left(\circledast_{m=1,m\neq n}^{N-1}\mathbf{A}^{(m)^T}\hat{\mathbf{A}}^{(m)}\right)$
14 $\quad\quad \mathbf{D}_1 \leftarrow \mathbf{A}^{(n)}\mathbf{D}_1$
15 $\quad\quad \hat{\mathbf{W}} \leftarrow \hat{\mathbf{A}}^{(N-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(n+1)} \odot \hat{\mathbf{A}}^{(n-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(1)}$
16 $\quad\quad \mathbf{D}_2 \leftarrow \mathbf{Z}_{(n)}\left(\mathbf{v}^T \odot \hat{\mathbf{W}}\right)$
17 $\quad\quad \mathbf{G} \leftarrow \hat{\mathbf{A}}^{(N)^T}\hat{\mathbf{A}}^{(N)} \circledast \left(\circledast_{m=1,m\neq n}^{N-1}\hat{\mathbf{A}}^{(m)^T}\hat{\mathbf{A}}^{(m)}\right)$
18 $\quad\quad$ Solve $\hat{\mathbf{A}}^{(n)}\mathbf{G} = \mathbf{D}_1 + \mathbf{D}_2$ for $\hat{\mathbf{A}}^{(n)}$
19 $\quad$ **end**
20 **until** termination criteria met

(10) and (9). Unlike the case $S = 0$, iALS($S > 0$) needs several iterations to converge. The summary of the algorithm is presented in Algorithm 1.

However, we cannot guarantee that the performance corresponding to semihot starting points is always good. The reason is that the Eckart–Young theorem [16] cannot be extended to higher-order tensors [15]. That is, there is no relationship between $\mathbf{A}^{(n)}$ and the final result $\hat{\mathbf{A}}^{(n)}$ in general. Therefore, iALS needs several iterations to converge, and the approximation error is influenced by the starting point.

Finally, we discuss the time cost of Algorithm 1. For convenience, we denote $I = \prod_{n=1}^{N-1} I_n$. From line 5 to line 11, the main cost lies in lines 6–8, whose costs are $\mathcal{O}(R(R+S)I_N), \mathcal{O}((R+S)I), \mathcal{O}((R+S)I)$, respectively. The total cost of lines 13 and 14 is $\mathcal{O}(R(R+S)\sum_n I_n)$, the total cost of lines 15 and 16 is $\mathcal{O}((R+S)I)$, and the cost of line 17 is $\mathcal{O}((R+S)^2\sum_n I_n)$. Therefore, the total time cost of one iteration is $\mathcal{O}(N(R+S)I + NR(R+S)I_N)$. We can see that the time cost of iALS($S > 0$) is relevant to I_N. Therefore, this method is not feasible for long-term computation.

3. Rank-one basis and incremental CP decompositions. Suppose $\mathcal{X}$ has a CP decomposition in (2) and (3). The new tensor $\hat{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times (I_N+1)}$ satisfies $\hat{\mathcal{X}}(:, \ldots, :, 1:I_N) = \mathcal{X}$ and $\hat{\mathcal{X}}(:, \ldots, :, I_N+1) = \mathcal{Z}$. We discuss the relationship between $\mathrm{rank}(\mathcal{X})$ and $\mathrm{rank}(\hat{\mathcal{X}})$ and consider how to obtain an incremental CP decomposition of $\hat{\mathcal{X}}$ in this section. Since we are interested in the CP decomposition in the incremental style, we first review the CP decomposition from a new perspective

corresponding to this style.

The tensor $\mathcal{X}$ can be regarded as a "vector" with length I_N, and each entry of this vector is an $(N-1)$st-order tensor $\mathcal{Z}^t := \mathcal{X}(:,\ldots,:,t)$, $1 \le t \le I_N$. Each $\mathcal{Z}^t$ is called a *slice* of $\mathcal{X}$. From (2) and (3), we can also define some $(N-1)$st-order tensors,

$$\mathcal{B}_r = \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N-1)}, \quad 1 \le r \le R.$$

Then

$$\left(\mathcal{Z}^1, \ldots, \mathcal{Z}^{I_N}\right) = \left(\mathcal{B}_1, \ldots, \mathcal{B}_R\right) \left(\mathbf{A}^{(N)}\right)^T. \tag{12}$$

This is analogous to a decomposition of a matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$:

$$\mathbf{W} = \mathbf{U}\mathbf{V}^T \quad \text{with } \mathbf{U} \in \mathbb{R}^{m\times r}, \mathbf{V} \in \mathbb{R}^{n\times r}. \tag{13}$$

One has $\operatorname{rank}(\mathbf{W}) \le \operatorname{rank}(\mathbf{U})$; if $\operatorname{rank}(\mathbf{U}) = \operatorname{rank}(\mathbf{V}) = r$, then $\operatorname{rank}(\mathbf{W}) = r$. We have similar results for (12).

LEMMA 3.1. *If $\mathcal{B}_1, \ldots, \mathcal{B}_R$ are linearly dependent, then* $\operatorname{rank}(\mathcal{X}) \le R-1$.

Proof. Without loss of generality, we assume that $\mathcal{B}_R$ can be expressed as a linear combination of $\mathcal{B}_1, \ldots, \mathcal{B}_{R-1}$,

$$\exists \mathbf{P} \in \mathbb{R}^{(R-1)\times R} \text{ such that } \left(\mathcal{B}_1, \ldots, \mathcal{B}_R\right) = \left(\mathcal{B}_1, \ldots, \mathcal{B}_{R-1}\right)\mathbf{P}.$$

Then (12) becomes

$$\left(\mathcal{Z}^1, \ldots, \mathcal{Z}^{I_N}\right) = \left(\mathcal{B}_1, \ldots, \mathcal{B}_{R-1}\right)\mathbf{Q}^T,$$

where $\mathbf{Q} = \mathbf{A}^{(N)}\mathbf{P}^T \in \mathbb{R}^{I_N\times(R-1)}$. It follows that

$$\mathcal{X} = \sum_{r=1}^{R-1} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N-1)} \circ \mathbf{Q}(:,r),$$

implying that $\operatorname{rank}(\mathcal{X}) \le R-1$. ∎

REMARK 3.2. *Lemma* 3.1 *is actually the same as* [28, Theorem 2] *and* [44, Lemma 3.1]*, where the results are with respect to the rank of* $\odot_{n=1}^{N-1}\mathbf{A}^{(n)}$*. Note that* $\odot_{n=1}^{N-1}\mathbf{A}^{(n)}$ *is just the matrix with columns being the vectorized* $\mathcal{B}_r, 1 \le r \le R$*,*

$$\left[\operatorname{vec}(\mathcal{B}_1) \ \cdots \ \operatorname{vec}(\mathcal{B}_R)\right] = \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}. \tag{14}$$

LEMMA 3.3. *If $\mathcal{B}_1, \ldots, \mathcal{B}_R$ are linearly independent and* $\operatorname{rank}(\mathbf{A}^{(N)}) = R$*, then* $\operatorname{rank}(\mathcal{X}) = R$.

Proof. First, we have that

$$\dim \operatorname{span}\{\mathcal{Z}^1, \ldots, \mathcal{Z}^{I_N}\} = \operatorname{rank}(\mathbf{X}_{(N)}) = \operatorname{rank}_{I_N}(\mathcal{X}).$$

This can be observed by the fact that the matrix with rows being the vectorized $\mathcal{Z}^t, 1 \le t \le I_N$, is just the mode-$I_N$ unfolding matrix $\mathbf{X}_{(N)}$. If $\mathcal{B}_1, \ldots, \mathcal{B}_R$ are linearly independent, by (12), one has

$$\dim \operatorname{span}\{\mathcal{Z}^1, \ldots, \mathcal{Z}^{I_N}\} = \operatorname{rank}(\mathbf{A}^{(N)}).$$

Combining the above two equations and [15, eq. (2.16)] yields

$$\operatorname{rank}(\mathcal{X}) \ge \operatorname{rank}_{I_N}(\mathcal{X}) = \operatorname{rank}(\mathbf{A}^{(N)}) = R.$$

On the other hand, it is clear that $\operatorname{rank}(\mathcal{X}) \le R$. ∎

REMARK 3.4. *For* (13)*, if* $\operatorname{rank}(\mathbf{U}) = r$*, one has* $\operatorname{rank}(\mathbf{M}) = \operatorname{rank}(\mathbf{V})$*. However, that* $\mathcal{B}_1, \ldots, \mathcal{B}_R$ *are linearly independent cannot yield that* $\operatorname{rank}(\mathcal{X}) = \operatorname{rank}(\mathbf{A}^{(N)})$*. There is an example in* [15, Lemma 4.7]*. A tensor* $\mathcal{X} \in \mathbb{R}^{2\times 2\times 2}$ *given by*

$$\mathcal{X}(:,:,1) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathcal{X}(:,:,2) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

has a rank decomposition

$$\mathcal{X} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

It can be checked that

$$\mathcal{B}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathcal{B}_2 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \mathcal{B}_3 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

are linearly independent. However, $\operatorname{rank}(\mathcal{X}) = 3 > \operatorname{rank}(\mathbf{A}^{(3)}) = 2$.

3.1. Rank-one basis.

DEFINITION 3.5. *Given an* N*th-order tensor* $\mathcal{X}$*, suppose a linearly independent set* $\mathscr{B} := \{\mathcal{B}_1, \ldots, \mathcal{B}_R\}$ *satisfies that*

1. $\mathcal{B}_r$ *is a rank-one tensor of order* $N-1$ *for all* $r = 1, \ldots, R$*; and*
2. $\mathcal{X}(:, \ldots, :, t) \in \operatorname{span}\mathscr{B}$ *for all* $t = 1, \ldots, I_N$.

Then $\mathscr{B}$ *is called a* rank-one basis *of* $\mathcal{X}$.

For a rank decomposition of $\mathcal{X}$: $\mathcal{X} = \sum_{r=1}^{\operatorname{rank}(\mathcal{X})} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)}$, by Lemma 3.1, $\mathcal{B}_1, \ldots, \mathcal{B}_{\operatorname{rank}(\mathcal{X})}$ defined by

$$\mathcal{B}_r = \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N-1)}, \quad 1 \le r \le \operatorname{rank}(\mathcal{X}),$$

are linearly independent, and hence $\{\mathcal{B}_1, \ldots, \mathcal{B}_{\operatorname{rank}(\mathcal{X})}\}$ is a rank-one basis of $\mathcal{X}$.

If $\mathscr{B}$ is a rank-one basis of $\mathcal{X}$, by (12), $\left(\mathbf{A}^{(N)}(t,:)\right)^T$ is the coordinate representation of $\mathcal{Z}^t$ with respect to the basis $\mathscr{B}$. The CP decomposition is determined by the rank-one basis. To find a CP decomposition of $\mathcal{X}$, we just need to find a rank-one basis of $\mathcal{X}$. For $N = 2$, i.e., the matrix case, since a nonzero first-order tensor, which is a vector, is inherently rank-one, every basis of the column space is a rank-one basis. However, for $N \ge 3$, to find such a special basis is difficult.

PROPOSITION 3.6. *For any tensor* $\mathcal{X}$*, one has*

$$\operatorname{rank}(\mathcal{X}) = \min\{\#\mathscr{B} : \mathscr{B} \text{ is a rank-one basis of } \mathcal{X}\},$$

where # *stands for cardinality.*

Proof. By definition, for any rank-one basis $\mathscr{B}$ of $\mathcal{X}$, we have $\operatorname{rank}(\mathcal{X}) \le \#\mathscr{B}$.

On the other hand, if $\mathscr{B}$ is the rank-one basis corresponding to a rank decomposition of $\mathcal{X}$, then $\#\mathscr{B} = \operatorname{rank}(\mathcal{X})$. ∎

3.2. Incremental CP decompositions. To obtain a CP decomposition of $\hat{\mathcal{X}}$, we only need to find a rank-one basis for $\mathcal{Z}^1, \ldots, \mathcal{Z}^{I_N}, \mathcal{Z}$. To reflect the incremental style, it is reasonable to consider a rank-one basis $\hat{\mathscr{B}}$ involving $\mathscr{B}$. Denote

$$S_{\min} = \min_{\mathbf{v}\in\mathbb{R}^R} \operatorname{rank}\left(\mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}\right). \tag{15}$$

If $S_{\min} = 0$, i.e., $\mathcal{Z} \in \operatorname{span}\mathscr{B}$, we have the following proposition.

PROPOSITION 3.7. *Let* $\mathscr{B} := \{\mathcal{B}_1, \ldots, \mathcal{B}_R\}$ *be a rank-one basis of* $\mathcal{X}$. *If* $\mathcal{Z} \in \text{span}\mathscr{B}$, *then* $\mathscr{B}$ *is also a rank-one basis of* $\hat{\mathcal{X}}$. *Moreover,*

1. *if the CP decomposition of* $\mathcal{X}$ *under* $\mathscr{B}$ *is unique, then the CP decomposition of* $\hat{\mathcal{X}}$ *under* $\mathscr{B}$ *is unique; and*
2. *if* $\text{rank}(\mathcal{X}) = R$, *then* $\text{rank}(\hat{\mathcal{X}}) = R$.

Proof. Because $\mathcal{Z} \in \text{span}\mathscr{B}$, by definition, $\mathscr{B}$ is a rank-one basis of $\hat{\mathcal{X}}$.

If the CP decomposition of $\mathcal{X}$ under $\mathscr{B}$ is unique, we prove the uniqueness of the CP decomposition of $\hat{\mathcal{X}}$ under $\mathscr{B}$ by contradiction. Suppose that there is a CP decomposition of $\hat{\mathcal{X}}$ under another basis $\mathscr{B}'$ with $\#\mathscr{B}' = R$. Note that $\mathscr{B}'$ is also a rank-one basis of $\mathcal{X}$. This implies that the CP decomposition of $\mathcal{X}$ is not unique, which would be a contradiction.

If $\text{rank}(\mathcal{X}) = R$, since $\mathcal{X}$ is a subtensor of $\hat{\mathcal{X}}$, it follows that $\text{rank}(\hat{\mathcal{X}}) \geq \text{rank}(\mathcal{X}) = R$. On the other hand, $\text{rank}(\hat{\mathcal{X}}) \leq \#\mathscr{B} = R$. ∎

If $S_{\min} > 0$, by the subadditivity of the tensor rank, we have

$$R \leq \text{rank}(\hat{\mathcal{X}}) \leq R + S_{\min}.$$

Note that the case $\text{rank}(\hat{\mathcal{X}}) = R$ exists. Here is an example.

EXAMPLE 3.8. *Consider a* $2 \times 2 \times 1$ *tensor* $\mathcal{X}$,

$$\mathcal{X} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

We can choose

$$\mathscr{B} = \left\{ \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

and $\text{rank}(\mathcal{X}) = \#\mathscr{B} = 2$. *Let the new tensor* $\hat{\mathcal{X}}$ *satisfy*

$$\hat{\mathcal{X}}(:,:,1) = \mathcal{X}, \quad \hat{\mathcal{X}}(:,:,2) = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

We can check that $\hat{\mathcal{X}}(:,:,2) \notin \text{span}\mathscr{B}$. *However,* $\text{rank}(\hat{\mathcal{X}}) = 2 \neq \text{rank}(\mathcal{X}) + 1$, *because*

$$\hat{\mathcal{X}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The following proposition gives a sufficient condition for $\text{rank}(\hat{\mathcal{X}}) > R$.

PROPOSITION 3.9. *Let* $\mathscr{B} := \{\mathcal{B}_1, \ldots, \mathcal{B}_R\}$ *be a rank-one basis of* $\mathcal{X}$. *If the CP decomposition of* $\mathcal{X}$ *under* $\mathscr{B}$ *is unique and* $S_{\min} > 0$ *for* (15), *then*

$$\text{rank}(\hat{\mathcal{X}}) \geq R + 1.$$

Proof. If $\text{rank}(\hat{\mathcal{X}}) \leq R$, then $\hat{\mathcal{X}}$ has a rank-one basis $\mathscr{B}'$ with $\#\mathscr{B}' = R$. We have $\mathscr{B} \neq \mathscr{B}'$. Otherwise, $\mathscr{B}$ is also a rank-one basis of $\hat{\mathcal{X}}$, which contradicts $\mathcal{Z} \notin \text{span}\mathscr{B}$. Note that $\mathscr{B}'$ is also a rank-one basis of $\mathcal{X}$. This implies that the CP decomposition of $\mathcal{X}$ is not unique, which would be a contradiction. ∎

If $S_{\min} = 1$, we have the following corollary.

COROLLARY 3.10. *Let* $\mathscr{B} := \{\mathcal{B}_1, \ldots, \mathcal{B}_R\}$ *be a rank-one basis of* $\mathcal{X}$. *If* $\mathcal{Z} \notin \text{span}\mathscr{B}$ *and there exists a vector* $\mathbf{v} \in \mathbb{R}^R$ *such that* $\hat{\mathcal{B}} := \mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}$ *is a rank-one tensor, then* $\hat{\mathscr{B}} := \mathscr{B} \cup \{\hat{\mathcal{B}}\}$ *is a rank-one basis of* $\hat{\mathcal{X}}$. *Moreover, if the CP decomposition of* $\mathcal{X}$ *under* $\mathscr{B}$ *is unique, then* $\text{rank}(\hat{\mathcal{X}}) = R + 1$.

In Corollary 3.10, the uniqueness of the CP decomposition of $\mathcal{X}$ does not guarantee the uniqueness of the CP decomposition of $\hat{\mathcal{X}}$.

EXAMPLE 3.11. *Consider a* $3 \times 3 \times 2$ *tensor* $\mathcal{X}$ *with*

$$\mathcal{X}(:,:,1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{X}(:,:,2) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then $\mathcal{X}$ *has a CP decomposition*

$$\mathcal{X} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Since

$$k_{\mathbf{A}^{(1)}} + k_{\mathbf{A}^{(2)}} + k_{\mathbf{A}^{(3)}} = 3 + 3 + 2 = 2R + 2,$$

by (5)*, this decomposition is unique. The rank-one basis of* $\mathcal{X}$ *is*

$$\mathscr{B} = \left\{ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right\}. \tag{16}$$

Let the new tensor $\hat{\mathcal{X}}$ *satisfy*

$$\hat{\mathcal{X}}(:,:,1:2) = \mathcal{X}, \quad \hat{\mathcal{X}}(:,:,3) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

We can check that both

$$\hat{\mathcal{X}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

and

$$\hat{\mathcal{X}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \circ \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

are CP decompositions.

If $S_{\min} \geq 2$ for (15), then even if the CP decomposition of $\mathcal{X}$ under $\mathscr{B}$ is unique, one does not have $\operatorname{rank}(\hat{\mathcal{X}}) = R + S_{\min}$, as shown in the next example.

EXAMPLE 3.12. *Let* $\mathcal{X} \in \mathbb{R}^{3\times3\times2}$ *satisfy*

$$\mathcal{X}(:,:,1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{X}(:,:,2) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

and let $\hat{\mathcal{X}}$ *satisfy*

$$\hat{\mathcal{X}}(:,:,1:2) = \mathcal{X}, \quad \hat{\mathcal{X}}(:,:,3) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

By Example 3.11*, under the rank-one basis* $\mathscr{B}$ *defined in* (16)*, the CP decomposition of* $\mathcal{X}$ *is unique and* $\operatorname{rank}(\mathcal{X})=3$*. We can check* $\min_{\mathbf{v}\in\mathbb{R}^3}\operatorname{rank}(\hat{\mathcal{X}}(:,:,3)-(\mathcal{B}_1,\mathcal{B}_2,\mathcal{B}_3)\mathbf{v})=2$ *and*

$$\hat{\mathcal{X}}=\begin{bmatrix}1\\1\\0\end{bmatrix}\circ\begin{bmatrix}1\\1\\0\end{bmatrix}\circ\begin{bmatrix}\frac{1}{2}\\0\\\frac{1}{2}\end{bmatrix}+\begin{bmatrix}1\\-1\\0\end{bmatrix}\circ\begin{bmatrix}1\\-1\\0\end{bmatrix}\circ\begin{bmatrix}\frac{1}{2}\\0\\-\frac{1}{2}\end{bmatrix}+\begin{bmatrix}0\\1\\1\end{bmatrix}\circ\begin{bmatrix}0\\1\\1\end{bmatrix}\circ\begin{bmatrix}0\\\frac{1}{2}\\\frac{1}{2}\end{bmatrix}+\begin{bmatrix}0\\1\\-1\end{bmatrix}\circ\begin{bmatrix}0\\1\\-1\end{bmatrix}\circ\begin{bmatrix}0\\\frac{1}{2}\\-\frac{1}{2}\end{bmatrix}.$$

Combining this decomposition with Proposition 3.9 *leads to* $\operatorname{rank}(\hat{\mathcal{X}})=4<R+S_{\min}$.

Now we discuss how to construct a rank-one basis for $\hat{\mathcal{X}}$ with the help of $\mathscr{B}$ when $S_{\min}\geq 1$ for (15). Suppose

$$\mathbf{v}^*\in\arg\min_{\mathbf{v}\in\mathbb{R}^R}\ \operatorname{rank}\left(\mathcal{Z}-(\mathcal{B}_1,\ldots,\mathcal{B}_R)\mathbf{v}\right),$$

and let

$$\hat{\mathcal{B}}:=\mathcal{Z}-(\mathcal{B}_1,\ldots,\mathcal{B}_R)\mathbf{v}^*.$$

Then $\operatorname{rank}(\hat{\mathcal{B}})=S_{\min}$. Suppose $\hat{\mathcal{B}}$ has the rank decomposition

$$\hat{\mathcal{B}}=\sum_{r=1}^{S_{\min}}\hat{\mathcal{B}}_r, \tag{17}$$

where $\hat{\mathcal{B}}_r$ is rank-one for all $r=1,\ldots,S_{\min}$.

PROPOSITION 3.13. *Let* $\mathscr{B}:=\{\mathcal{B}_1,\ldots,\mathcal{B}_R\}$ *be a rank-one basis of* $\mathcal{X}$*. Suppose* $S_{\min}\geq 1$ *for* (15) *and* $\hat{\mathcal{B}}_1,\ldots,\hat{\mathcal{B}}_{S_{\min}}$ *are defined as* (17)*. Then the set* $\hat{\mathscr{B}}:=\mathscr{B}\cup\{\hat{\mathcal{B}}_1,\ldots,\hat{\mathcal{B}}_{S_{\min}}\}$ *is a rank-one basis of* $\hat{\mathcal{X}}$*.*

Proof. We only need to prove that $\mathcal{B}_1,\ldots,\mathcal{B}_R,\hat{\mathcal{B}}_1,\ldots,\hat{\mathcal{B}}_{S_{\min}}$ are linearly independent. Suppose there exist $c_i,1\leq i\leq R$, and $d_j,1\leq j\leq S_{\min}$, not all zero, such that

$$\sum_{i=1}^{R}c_i\mathcal{B}_i+\sum_{j=1}^{S_{\min}}d_j\hat{\mathcal{B}}_j=0.$$

If $d_j=0$ for all j, since $\mathcal{B}_1,\ldots,\mathcal{B}_R$ are linearly independent, it follows that $c_i=0$ for all i. Therefore, there exists at least one j such that $d_j\neq 0$. Without loss of generality, we assume that $d_{S_{\min}}\neq 0$. That is, $\hat{\mathcal{B}}_{S_{\min}}$ can be represented as a linear combination of $\mathcal{B}_1,\ldots,\mathcal{B}_R,\hat{\mathcal{B}}_1,\ldots,\hat{\mathcal{B}}_{S_{\min}-1}$. There exist $\alpha_i,1\leq i\leq R$, and $\beta_j,1\leq j\leq S_{\min}-1$, such that

$$\mathcal{Z}=\sum_{i=1}^{R}\alpha_i\mathcal{B}_i+\sum_{j=1}^{S_{\min}-1}\beta_j\hat{\mathcal{B}}_j.$$

Let $\mathbf{u}=[\alpha_1,\ldots,\alpha_R]^T\in\mathbb{R}^R$. Then

$$\operatorname{rank}\left(\mathcal{Z}-(\mathcal{B}_1,\ldots,\mathcal{B}_R)\mathbf{u}\right)=\operatorname{rank}\left(\sum_{j=1}^{S_{\min}-1}\beta_j\hat{\mathcal{B}}_j\right)\leq S_{\min}-1,$$

which contradicts $\operatorname{rank}(\hat{\mathcal{B}})=S_{\min}$. ∎

However, Proposition 3.13 is only a theoretical result. When $N=3$, (15) is an NP-hard problem [8, 9, 34]. When $N>3$, (15) cannot be solved directly, because the calculation of tensor rank is NP-hard; see [24, section 3.4] for details. We will find some alternative methods for (15) in the next section.

4. The algorithm. We use the notation in section 2.4. For (7), we assume that $\mathscr{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_R\}$ is a rank-one basis of $\tilde{\mathcal{X}}$, where

$$\mathcal{B}_r = \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N-1)}, \quad 1 \le r \le R.$$

We want to compute a CP decomposition of $\hat{\mathcal{X}}$ with $\hat{R} = R + S$ components.

4.1. Extend rank-one basis. We focus on the case $S > 0$ in this subsection. First we consider how to obtain $\hat{\mathscr{B}}$ from $\mathscr{B}$. As mentioned in the last paragraph of section 3, we cannot obtain the rank-one basis based on (15). As in the strategy of fitting CP decompositions, we can find $\hat{\mathscr{B}}$ with the number of components fixed. Define

$$M_S = \{\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{N-1}} : \operatorname{rank}(\mathcal{T}) \le S,\ S > 0\}.$$

The model for finding $\hat{\mathscr{B}}$ is

$$\min_{\mathcal{T} \in M_S, \mathbf{v} \in \mathbb{R}^R} \mathscr{L}(\mathcal{T}, \mathbf{v}) := \|\mathcal{Z} - \mathcal{T} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}\|, \tag{18}$$

where $\mathcal{Z} = \hat{\mathcal{X}}(:, \ldots, :, I_N + 1)$. For the case $N = 3$, $\mathcal{Z}, \mathcal{T}$ and $\mathcal{B}_r$ in (18) are actually matrices. For ease of writing a unified algorithm, we do not change notation.

Model (18) can be solved by the following alternating minimization scheme:

$$\begin{cases} \mathbf{v}^{(k)} \in \arg\min_{\mathbf{v} \in \mathbb{R}^R} \mathscr{L}(\mathcal{T}^{(k)}, \mathbf{v}); & (19) \\ \mathcal{T}^{(k+1)} \in \arg\min_{\mathcal{T} \in M_S} \mathscr{L}(\mathcal{T}, \mathbf{v}^{(k)}). & (20) \end{cases}$$

The subproblem (19) is a least squares problem,

$$\min_{\mathbf{v} \in \mathbb{R}^R} \left\| \mathcal{Z} - \mathcal{T}^{(k)} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v} \right\|, \tag{21}$$

and the subproblem (20) is a low-rank approximation problem,

$$\min_{\mathcal{T} \in M_S} \left\| \mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}^{(k)} - \mathcal{T} \right\|. \tag{22}$$

For the case $N = 3$, (22) can be solved easily by computing a partial SVD of $\mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}^{(k)}$ with S components. When the order of a tensor is greater than or equal to three, the best rank-r approximation does not exist in general [15]. Therefore, unlike the case $N = 3$, (22) is ill-posed for the case $N \ge 4$. Suppose the result of ALS for fitting the CP decomposition of a tensor $\mathcal{M}$ with r components is $\texttt{ALS}(\mathcal{M}, r)$. We can use $\texttt{ALS}\big(\mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}^{(k)}, S\big)$ to solve (22).

For the case $N = 3$, the convergence analysis of this alternating minimization can follow the framework of [4], in which case the two subproblems (19) and (20) have unique minimizers and we have $\mathscr{L}(\mathcal{T}^{(k+1)}, \mathbf{v}^{(k+1)}) \le \mathscr{L}(\mathcal{T}^{(k)}, \mathbf{v}^{(k)})$. We cannot obtain the global convergence of $\{\mathcal{T}^{(k)}, \mathbf{v}^{(k)}\}$ in theory. The detailed convergence analysis of the alternating minimization is involved and beyond the scope of this paper. We refer the interested reader to [5, 50, 53] and references therein for detailed discussions. Actually, for our algorithm, the convergence behavior of $\{\mathscr{L}(\mathcal{T}^{(k)}, \mathbf{v}^{(k)})\}$ is more meaningful, which follows from $0 \le \mathscr{L}(\mathcal{T}^{(k+1)}, \mathbf{v}^{(k+1)}) \le \mathscr{L}(\mathcal{T}^{(k)}, \mathbf{v}^{(k)})$. For the case $N \ge 4$, since (22) is ill-posed, even the convergence of $\{\mathscr{L}(\mathcal{T}^{(k)}, \mathbf{v}^{(k)})\}$ cannot be guaranteed.

After obtaining $\hat{\mathscr{B}}$, we can update factor matrices. Suppose we obtain a numerical solution $\{\mathcal{T}^*, \mathbf{v}^*\}$ for (18). For the case $N = 3$, let the SVD of $\mathcal{T}^*$ be

$$\mathcal{T}^* = \sum_{r=1}^{S} \sigma_r \ \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)},$$

and denote $\mathbf{U}^{(n)} = [\mathbf{u}_1^{(n)} \ \cdots \ \mathbf{u}_S^{(n)}]$ for $n = 1, 2$, $\boldsymbol{\sigma} = [\sigma_1, \ldots, \sigma_S]^T$. For the case $N \geq 4$, let the CP decomposition of $\mathcal{T}^*$ obtained by ALS be

$$\mathcal{T}^* = [\![\boldsymbol{\sigma}; \mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}]\!],$$

where $\boldsymbol{\sigma} \in \mathbb{R}^S, \mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times S}$ for $n = 1, \ldots, N-1$. Then $\hat{\mathscr{B}}$ yields the updated factor matrices,

$$\hat{\mathbf{A}}^{(n)} = \begin{bmatrix} \mathbf{A}^{(n)} & \mathbf{U}^{(n)} \end{bmatrix}, \quad 1 \leq n \leq N-1. \tag{23}$$

The coordinate of $\mathcal{Z}$ with respect to $\hat{\mathscr{B}}$ is $\begin{bmatrix} \mathbf{v}^{*T} & \boldsymbol{\sigma}^T \end{bmatrix}$, and then

$$\hat{\mathbf{A}}^{(N)} = \begin{bmatrix} \mathbf{A}^{(N)} & 0 \\ \mathbf{v}^{*T} & \boldsymbol{\sigma}^T \end{bmatrix}. \tag{24}$$

Finally, we can update $\hat{\mathbf{A}}^{(n)}$ again with the same method as that for updating $\hat{\mathbf{A}}^{(n)}$ in iALS($S = 0$), where $n = 1, \ldots, N-1$. In fact, this algorithm can be combined with iALS($S = 0$) perfectly for the general case. We will show a systematic algorithm in the following.

4.2. The systematic algorithm. We set $\mathcal{T}^{(1)}$ as a zero tensor for the first step of (19). By (14), the matrix form of (21) is

$$\min_{\mathbf{v} \in \mathbb{R}^R} \left\| \text{vec}(\mathcal{Z} - \mathcal{T}^{(k)}) - \mathbf{H}^{(N)} \mathbf{v} \right\|,$$

where $\mathbf{H}^{(N)}$ is defined as in (6). As in (11), the normal equation of this problem is

$$(\mathbf{H}^{(N)^T} \mathbf{H}^{(N)}) \mathbf{v} = \mathbf{H}^{(N)^T} \text{vec}(\mathcal{Z} - \mathcal{T}^{(k)}).$$

The starting point is important for ALS. When $N \geq 4$, for the first time computing (22), i.e., the computation of $\mathcal{T}^{(2)}$ of (20), we choose a random starting point; for $k \geq 2$, a good starting point for solving (22) with ALS is $\mathcal{T}^{(k)}$.

We give the algorithm for adding components by alternating minimization (ACAM) in Algorithm 2, where $\mathbf{v}^{(1)}$ for the first step of (19) is input for ease of presentation of the systematic algorithm.

After obtaining $\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N)}$ by (23) and (24), we can update $\hat{\mathbf{A}}^{(1)}, \ldots, \hat{\mathbf{A}}^{(N-1)}$ again with the same method of iALS($S = 0$). Noting the special form of (23) and (24), we can simplify the computation of some intermediate results. We give the detailed computation in Algorithm 3. Because both obtaining new components and updating factor matrices by ALS are done by alternating minimization, we call the systematic algorithm *incremental CP decomposition by alternating minimization* (iCP-AM for short). The algorithm is presented in Algorithm 4.

Algorithm 2: Adding components by alternating minimization (ACAM).

1 **function** $[\mathbf{v}, \boldsymbol{\sigma}, \{\mathbf{U}^{(n)}\}_{n=1}^{N}] = \texttt{ACAM}\ (\mathcal{Z}, \mathbf{H}^{(N)}, {\mathbf{H}^{(N)}}^T\mathbf{H}^{(N)}, \mathbf{v}^{(1)}, S)$
2 $\quad \mathcal{E} \leftarrow \mathcal{Z} - \text{reshape}\left(\mathbf{H}^{(N)}\mathbf{v}^{(1)}, [I_1, \ldots, I_{N-1}]\right)$
3 $\quad$ **repeat**
4 $\quad\quad$ **if** $N = 3$ **then**
5 $\quad\quad\quad$ $[\mathbf{U}^{(1)}, \boldsymbol{\Sigma}, \mathbf{U}^{(2)}] \leftarrow \text{partialSVD}(\mathcal{E}, S)$
6 $\quad\quad\quad$ $\boldsymbol{\sigma} \leftarrow \text{diag}(\boldsymbol{\Sigma})$
7 $\quad\quad\quad$ $\mathcal{T} \leftarrow \mathbf{U}^{(1)}\boldsymbol{\Sigma}{\mathbf{U}^{(2)}}^T$
8 $\quad\quad$ **else**
9 $\quad\quad\quad$ $[\![\boldsymbol{\sigma}; \mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}]\!] \leftarrow \texttt{ALS}(\mathcal{E}, S)$
10 $\quad\quad\quad$ $\mathcal{T} \leftarrow [\![\boldsymbol{\sigma}; \mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N-1)}]\!]$
11 $\quad\quad$ **end**
12 $\quad\quad$ $\mathbf{b} \leftarrow {\mathbf{H}^{(N)}}^T \text{vec}\,(\mathcal{Z} - \mathcal{T})$
13 $\quad\quad$ Solve $({\mathbf{H}^{(N)}}^T\mathbf{H}^{(N)})\mathbf{v} = \mathbf{b}$ for $\mathbf{v}$
14 $\quad\quad$ $\mathcal{E} \leftarrow \mathcal{Z} - \text{reshape}\left(\mathbf{H}^{(N)}\mathbf{v}, [I_1, \ldots, I_{N-1}]\right)$
15 $\quad$ **until** termination criteria met
16 **end function**

4.3. The estimation of S. An important issue is how to determine S. If we have a priori knowledge of the data, S can be set based on this knowledge. Otherwise, we can set S adaptively based on the data.

A simple strategy is to set S based on the residual of the first step of the alternating minimization, i.e., the residual of (19) for $k = 1$,

$$\mathcal{E} = \mathcal{Z} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}^{(1)},$$

where $\mathbf{v}^{(1)} \in \arg\min_{\mathbf{v}\in\mathbb{R}^R} \mathscr{L}(0, \mathbf{v})$. In Algorithm 2, this residual is obtained in line 2. The relative error corresponding to this residual is

$$\varepsilon = \frac{\|\mathcal{E}\|}{\|\mathcal{Z}\|}.$$

Then S can be set as

$$S = \max\left\{\left\lfloor \theta\ \frac{\varepsilon - \varepsilon_0}{\varepsilon_0} \right\rfloor, 0\right\}, \tag{25}$$

where $\theta > 0$ is a parameter, ε_0 is a given threshold, and $\frac{\varepsilon-\varepsilon_0}{\varepsilon_0}$ is the relative change between ε and ε_0. The choice of ε_0 depends on the requirement on the approximation error, and θ is data dependent.

4.4. The computational cost. We consider the cost of iCP-AM in Algorithm 4. As in section 2.4.2, we denote $I = \prod_{n=1}^{N-1} I_n$.

Like Algorithm 1, the cost of the noniteration part of iCP-AM is $\mathcal{O}(N(R+S)I)$. Now we discuss the iteration part of iCP-AM, i.e., ACAM in Algorithm 2. In Algorithm 2, the total cost from line 12 to 14 is $\mathcal{O}(RI)$. If $N = 3$, the total cost from line 5 to line 7 is $\mathcal{O}(SI)$; if $N \geq 4$, the implementation of ALS for an $(N-1)$st-order tensor in line 9 is also done by an iterative procedure, whose cost is $\mathcal{O}((N-1)SI)$ for each iteration; see [3]. Hence, the total cost of one iteration of ACAM is $\mathcal{O}((R+S)I)$

Algorithm 3: Compute intermediate results.

Input: $\{\mathbf{A}^{(n)}\}_{n=1}^{N}, \{\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}\}_{n=1}^{N}, \{\mathbf{U}^{(n)}\}_{n=1}^{N-1}, \mathbf{v}, \boldsymbol{\sigma}, S$

Output: $\{\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N-1}, \mathbf{A}^{(N)^T}\mathbf{C}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}$

if $S = 0$ **then**
  for $n = 1, \ldots, N-1$ **do**
    $\hat{\mathbf{A}}^{(n)} \leftarrow \mathbf{A}^{(n)}, \mathbf{A}^{(n)^T}\hat{\mathbf{A}}^{(n)} \leftarrow \mathbf{A}^{(n)^T}\mathbf{A}^{(n)}, \hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)} \leftarrow \mathbf{A}^{(n)^T}\mathbf{A}^{(n)}$
  end
  $\hat{\mathbf{A}}^{(N)} \leftarrow \begin{bmatrix} \mathbf{A}^{(N)} \\ \mathbf{v}^T \end{bmatrix}$
  $\mathbf{A}^{(N)^T}\mathbf{C} \leftarrow \mathbf{A}^{(N)^T}\mathbf{A}^{(N)}$
  $\hat{\mathbf{A}}^{(N)^T}\hat{\mathbf{A}}^{(N)} \leftarrow \mathbf{A}^{(N)^T}\mathbf{A}^{(N)} + \mathbf{v}\mathbf{v}^T$
else
  for $n = 1, \ldots, N-1$ **do**
    $\hat{\mathbf{A}}^{(n)} \leftarrow \begin{bmatrix} \mathbf{A}^{(n)} & \mathbf{U}^{(n)} \end{bmatrix}$
    $\mathbf{M} \leftarrow \mathbf{A}^{(n)^T}\mathbf{U}^{(n)}, \mathbf{L} \leftarrow \mathbf{U}^{(n)^T}\mathbf{U}^{(n)}$
    $\mathbf{A}^{(n)^T}\hat{\mathbf{A}}^{(n)} \leftarrow \begin{bmatrix} \mathbf{A}^{(n)^T}\mathbf{A}^{(n)} & \mathbf{M} \end{bmatrix}$
    $\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)} \leftarrow \begin{bmatrix} \mathbf{A}^{(n)^T}\mathbf{A}^{(n)} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{L} \end{bmatrix}$
  end
  $\hat{\mathbf{A}}^{(N)} \leftarrow \begin{bmatrix} \mathbf{A}^{(N)} & 0 \\ \mathbf{v}^T & \boldsymbol{\sigma}^T \end{bmatrix}$
  $\mathbf{A}^{(N)^T}\mathbf{C} \leftarrow \begin{bmatrix} \mathbf{A}^{(N)^T}\mathbf{A}^{(N)} & 0 \end{bmatrix}$ ▷ $\mathbf{A}^{(N)^T}\mathbf{C} \in \mathbb{R}^{R\times(R+S)}$
  $\hat{\mathbf{A}}^{(N)^T}\hat{\mathbf{A}}^{(N)} \leftarrow \begin{bmatrix} \mathbf{A}^{(N)^T}\mathbf{A}^{(N)} + \mathbf{v}\mathbf{v}^T & \mathbf{v}\boldsymbol{\sigma}^T \\ \boldsymbol{\sigma}\mathbf{v}^T & \boldsymbol{\sigma}\boldsymbol{\sigma}^T \end{bmatrix}$
end

for $N = 3$. Suppose the iteration number of ALS used in ACAM is M; then the total cost of one iteration of ACAM is $\mathcal{O}((R + (N-1)MS)I)$ for $N \geq 4$. We can find that the cost of iCP-AM is constant with respect to the length I_N of processed data.

In Table 1, we summarize the time cost of iCP-AM, along with iALS($S > 0$) and ALS, both of which can handle the case $S > 0$.

5. Experiments. In this section, we evaluate the performance of iCP-AM. As in [3], we use the *fit* to evaluate results. For a CP decomposition $\tilde{\mathcal{X}} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ of $\mathcal{X}$, the fit is defined as

$$F = 1 - \frac{\|\mathcal{X} - \tilde{\mathcal{X}}\|}{\|\mathcal{X}\|}.$$

All experiments are performed on MATLAB R2016a with Tensor Toolbox, version 3.0 [2] on a laptop (Intel Core i5-6300HQ CPU @ 2.30GHz, 8.00G RAM). The running time is the user-observed wall-clock time, measured by the combination of MATLAB functions `tic` and `toc`. We employ the influential PROPACK [26] to compute the partial SVD. There are many possible ways to do the initialization of ALS. We will use the following two different ways for the initialization of `ALS`$(\mathcal{X}, R)$:

1. Higher-order SVD (HOSVD) initialization: Initialize $\mathbf{A}^{(n)}$ to be the leading R left singular vectors of the mode-n unfolding, $\mathbf{X}_{(n)}$.
2. Random initialization: Initialize $\mathbf{A}^{(n)}$ randomly.

Algorithm 4: Incremental CP decomposition by alternating minimization (iCP-AM).

Input: $\mathcal{Z}, \{\mathbf{A}^{(n)}\}_{n=1}^{N}, \{\mathbf{A}^{(n)^T}\mathbf{A}^{(n)}\}_{n=1}^{N}, S$
Output: $\{\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}$
$\mathbf{H} \leftarrow \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}$
$\mathbf{G} \leftarrow \mathbf{A}^{(N-1)^T}\mathbf{A}^{(N-1)} \circledast \cdots \circledast \mathbf{A}^{(1)^T}\mathbf{A}^{(1)}$
$\mathbf{b} \leftarrow \mathbf{H}^T \mathrm{vec}(\mathcal{Z})$
Solve $\mathbf{G}\mathbf{v} = \mathbf{b}$ for $\mathbf{v}$
if $S > 0$ **then**
  $[\mathbf{v}, \boldsymbol{\sigma}, \{\mathbf{U}^{(n)}\}_{n=1}^{N}] \leftarrow \texttt{ACAM}(\mathcal{Z}, \mathbf{H}, \mathbf{G}, \mathbf{v}, S)$
end
Get $\{\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N-1}, \mathbf{A}^{(N)^T}\mathbf{C}, \{\hat{\mathbf{A}}^{(n)^T}\hat{\mathbf{A}}^{(n)}\}_{n=1}^{N}$ by Algorithm 3
for $n = 1, \ldots, N-1$ **do**
  $\mathbf{D}_1 \leftarrow \mathbf{A}^{(N)^T}\mathbf{C} \circledast \left(\circledast_{m=1,m\neq n}^{N-1} \mathbf{A}^{(m)^T}\hat{\mathbf{A}}^{(m)}\right)$
  $\mathbf{D}_1 \leftarrow \mathbf{A}^{(n)}\mathbf{D}_1$
  $\hat{\mathbf{W}} \leftarrow \hat{\mathbf{A}}^{(N-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(n+1)} \odot \hat{\mathbf{A}}^{(n-1)} \odot \cdots \odot \hat{\mathbf{A}}^{(1)}$
  $\mathbf{D}_2 \leftarrow \mathbf{Z}_{(n)}\left(\mathbf{v}^T \odot \hat{\mathbf{W}}\right)$
  $\mathbf{G} \leftarrow \hat{\mathbf{A}}^{(N)^T}\hat{\mathbf{A}}^{(N)} \circledast \left(\circledast_{m=1,m\neq n}^{N-1} \hat{\mathbf{A}}^{(m)^T}\hat{\mathbf{A}}^{(m)}\right)$
  Solve $\hat{\mathbf{A}}^{(n)}\mathbf{G} = \mathbf{D}_1 + \mathbf{D}_2$ for $\hat{\mathbf{A}}^{(n)}$
end

TABLE 1
Cost comparison, where M is the iteration number of ALS used in ACAM.

Algorithm	Noniteration part	Iteration part (per iteration)
ALS	-	$\mathcal{O}(NRII_N)$
iALS($S > 0$)	-	$\mathcal{O}(N(R+S)I + NR(R+S)I_N)$
iCP-AM ($N = 3$)	$\mathcal{O}(N(R+S)I)$	$\mathcal{O}((R+S)I)$
iCP-AM ($N \geq 4$)	$\mathcal{O}(N(R+S)I)$	$\mathcal{O}((R+(N-1)MS)I)$

The methods of ALS and iALS terminate when the change in fit goes below 10^{-5} or the number of iterations exceeds 100.

For real-world data, we test the videos "Hall," "News," and "Drive."[1] The original data are color video, i.e., fourth-order tensors (height × width × channels × frames): Hall-4D ($144 \times 176 \times 3 \times 300$), News-4D ($144 \times 176 \times 3 \times 300$), and Drive-4D ($1392 \times 512 \times 3 \times 90$). Some frames of Drive-4D have been shown in Figure 1. To test third-order tensors, we convert these data into black-and-white videos (height × width × frames): Hall-3D ($144 \times 176 \times 300$), News-3D ($144 \times 176 \times 300$) and Drive-3D ($1392 \times 512 \times 90$). We show one frame of "Hall" and "News" in Figure 2. "Hall" and "News" are obtained by a fixed camera, and the background is unchanged, but there are objects disappearing and reappearing. "Drive" is obtained by a driving recorder, and its background is also changing. We expect "Hall" and "News" are incremental tensors with stable rank, and "Drive" is an incremental tensor with rapidly increasing rank.

[1] "Hall" and "News" are from the video trace library [37] and available at http://trace.eas.asu.edu/yuv/. "Drive" is from the kitti dataset [17] and available at http://www.cvlibs.net/datasets/kitti/raw_data.php?type=residential.

We also test two types of synthetic tensors: incremental tensors with stable rank and incremental tensors with rapidly increasing rank. To generate an incremental tensor with stable rank, first we randomly generate N factor matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times 6}$. Then $\mathcal{H}_{\text{true}}$ is generated by $\mathcal{H}_{\text{true}} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$. The test tensor is

$$\mathcal{H} = \mathcal{H}_{\text{true}} + \rho\mathcal{N},$$

where the Gaussian noise tensor $\mathcal{N}$ has normally distributed entries, and $\rho = 0.1\|\mathcal{H}_{\text{true}}\|/\|\mathcal{N}\|$. To generate an incremental tensor with rapidly increasing rank, we first generate 10 rank-5 tensors $\mathcal{G}^j, 1 \le j \le 10$, randomly with the same size $I_1 \times \cdots \times I_{N-1} \times 50$. Then we combine these 10 small tensors to form a large tensor $\mathcal{G}_{\text{true}}$,

$$\mathcal{G}_{\text{true}}(:, \ldots, :, 50(j-1)+1 : 50j)) = \mathcal{G}^j, \quad j = 1, \ldots, 10.$$

The test tensor is

$$\mathcal{G} = \mathcal{G}_{\text{true}} + \rho\mathcal{N},$$

where the Gaussian noise tensor $\mathcal{N}$ has normally distributed entries, and $\rho = 0.1\|\mathcal{G}_{\text{true}}\|/\|\mathcal{N}\|$. The information about the synthetic tensors is shown in Table 2.

FIG. 2. *One slice of test videos. From left to right: Hall-4D, Hall-3D, News-4D, and News-3D.*

TABLE 2
The synthetic tensors.

Tensor	Size	Notes
$\mathcal{H}_3$	$150 \times 150 \times 500$	stable rank
$\mathcal{G}_3$	$150 \times 150 \times 500$	rapidly increasing rank
$\mathcal{H}_4$	$50 \times 50 \times 50 \times 500$	stable rank
$\mathcal{G}_4$	$50 \times 50 \times 50 \times 500$	rapidly increasing rank

5.1. Convergence behavior of iCP-AM for $S > 0$. Here we focus on the convergence of iCP-AM for $S > 0$, i.e., ACAM in Algorithm 2. After obtaining $\{\mathcal{T}^{(k)}, \mathbf{v}^{(k)}\}$ for (18), we see that the fit for fitting $\mathcal{Z}$ is

$$F^{(k)} = 1 - \frac{\left\|\mathcal{Z} - \mathcal{T}^{(k)} - (\mathcal{B}_1, \ldots, \mathcal{B}_R)\mathbf{v}^{(k)}\right\|}{\|\mathcal{Z}\|}.$$

Denote by $\mathcal{M}$ the whole tensor of the test data. We choose the first 150 slices, $\mathcal{X} := \mathcal{M}(:, \ldots, :, 1 : 150)$, as the initialization data and compute $\texttt{ALS}(\mathcal{X}, 5)$ with HOSVD initialization. Then we compute the incremental CP decomposition of $\hat{\mathcal{X}} := \mathcal{M}(:, \ldots, :, 1 : 151)$ by iCP-AM with $S = 1$. We show $F^{(k)}$ and $|F^{(k+1)} - F^{(k)}|$ versus the iteration number for some tensors in Figure 3.

As shown in Figure 3, the sequence $\{F^{(k)}\}$ is monotonically increasing, and the greatest change between successive elements is between $F^{(1)}$ and $F^{(2)}$. This is easy

to understand, because there is a new component added for $F^{(2)}$. ACAM converges very quickly. For the two third-order tensors, the gap between $F^{(2)}$ and $F^{(3)}$ is less than 10^{-5}, and for the two fourth-order tensors, the gap between $F^{(k)}$ and $F^{(k+1)}$ is less than 10^{-5} for some $k < 15$.

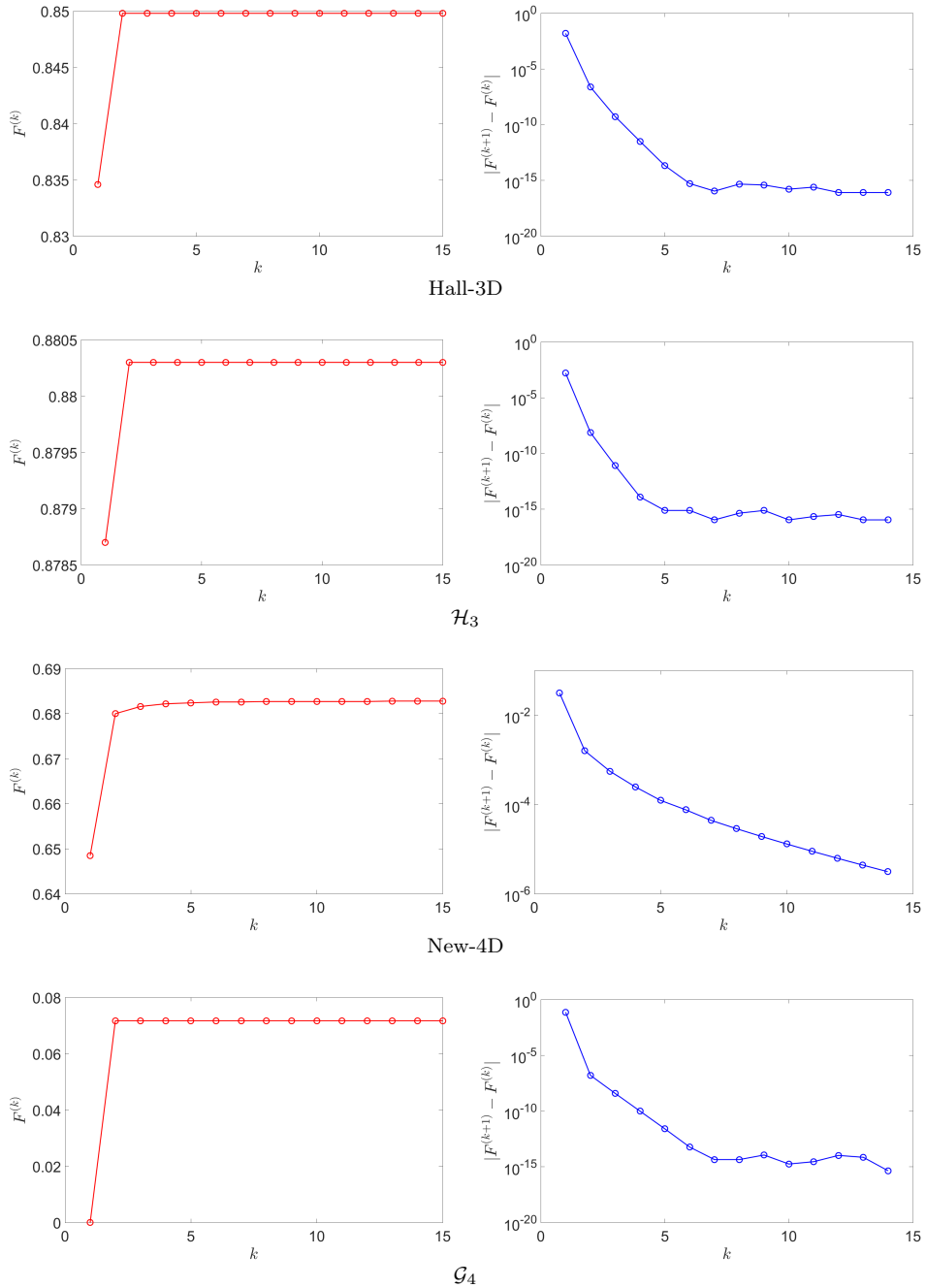

FIG. 3. *Convergence behavior of iCP-AM for $S > 0$: The values of $F^{(k)}$ and $|F^{(k+1)} - F^{(k)}|$ versus the iteration number k.*

In all the following experiments, we terminate the alternating minimization procedure of ACAM if the change in relative error goes below 10^{-5} ($|F^{(k+1)} - F^{(k)}| < 10^{-5}$) or the number of iterations exceeds 50.

5.2. Results on S. Throughout the remaining experiments of section 5, we use the whole data of each tensor. For each test tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times \cdots \times I_{N-1} \times K}$, suppose $\mathcal{M}_0 := \mathcal{M}(:, \ldots, :, 1 : I_N)$ is the initialization data. We define a tensor sequence $\mathcal{M}_0, \mathcal{M}_1, \ldots$, where

$$\mathcal{M}_t := \mathcal{M}(:, \ldots, :, 1 : I_N + t), \quad t = 0, 1, \ldots.$$

Denote by R_t the number of components corresponding to $\mathcal{M}_t$ and $S_t = R_{t+1} - R_t$.

In section 4.3, we discussed how to set S. In this subsection, we present the results on S with the adaptive method defined by (25).

For each test tensor $\mathcal{M}$, we choose the first 10% of the data, i.e., $\mathcal{M}_0 := \mathcal{M}(:, \ldots, :, 1 : \frac{K}{10})$ as the initialization. The result of $\texttt{ALS}(\mathcal{M}_0, 5)$ with HOSVD initialization is used to initialize iCP-AM. For convenience, we set ε_0 to be the relative error of the initialization. After initialization and setting, the remaining 90% of data are appended to the existing tensor one slice at a time. At the tth step, we compute the CP decomposition of $\mathcal{M}_t$ by iCP-AM and record S_t. For the real-world tensors, we test $\theta = 5, 10, 30, 100$; for the synthetic tensors with stable rank, we test $\theta = 1, 2, 3, 4$; and for the synthetic tensors with rapidly increasing rank, we test $\theta = 0.10, 0.15, 0.20, 0.25$.

Table 3 shows the results. We find that the influence of θ on S_t differs greatly on different tensors. This suggests the difficulty in setting θ for practical applications. One strategy is to tune θ empirically based on the existing knowledge from previous data obtained by the same device.

5.3. Comparison results. We will run iCP-AM on a whole tensor and compare with iALS, ALS, and OnlineCP [55]. For each test tensor $\mathcal{M} \in \mathbb{R}^{I_1 \times \cdots \times I_{N-1} \times K}$, we choose the first 10% of slices as the initialization data, i.e., $\mathcal{M}_0 := \mathcal{M}(:, \ldots, :, 1 : \frac{K}{10})$. The result of $\texttt{ALS}(\mathcal{M}_0, 5)$ with HOSVD initialization is used to initialize all methods.

First, we test "Hall" and "News" and the synthetic tensors with stable rank. After initialization, the remaining 90% of data are appended to the existing tensor one slice at a time.

(i) The first setting is $S_t = 0$ for all t. At the tth step, we compute the CP decomposition of $\mathcal{M}_t$ by OnlineCP, ALS, and iALS, respectively. Because iCP-AM for $S = 0$ is the same as iALS for $S = 0$, we do not test iCP-AM.

(ii) The second setting is to set a suitable θ in (25) such that the final number of components $\bar{R} = 6$ and $\bar{R} = 7$. At the tth step, we compute the CP decomposition of $\mathcal{M}_t$ by iCP-AM, iALS, and ALS, respectively, with the same number of components R_t.

For all cases, if $S_t = 0$, the results of $\texttt{ALS}(\mathcal{M}_{t-1}, R_{t-1})$ are used as the initialization of $\texttt{ALS}(\mathcal{M}_t, R_t)$ to save cost and obtain a better fit. Otherwise, $\texttt{ALS}(\mathcal{M}_t, R_t)$ is with random initialization. After processing each step, we record the running time and compute the fit of the updated CP decomposition. After all data are processed, in Table 4, we record the fit and the running time of the initialization and compute the mean fit and the mean running time for one step corresponding to different methods under different settings. The experiments are repeated 10 times, and Table 4 reports the average results.

As shown in Table 4, when $\bar{R} = 5$, OnlineCP and iALS have almost the same performance; when $\bar{R} > 5$, iALS and iCP-AM have almost the same performance.

TABLE 3

Results on S for different θ's with the adaptive method defined by (25), where ε_0 is set to be the relative error of the initialization. Here $\bar{R}$ is the final number of components, and $\max S := \max_t\{S_t\}$.

Data	ε_0		$\theta = 5$	$\theta = 10$	$\theta = 30$	$\theta = 100$
Hall-3D	0.1516	$\bar{R}$	6	8	10	17
		$\max S$	1	3	5	12
News-3D	0.2588	$\bar{R}$	7	9	19	40
		$\max S$	2	2	10	29
Drive-3D	0.3518	$\bar{R}$	10	16	41	65
		$\max S$	2	3	12	37
Hall-4D	0.1669	$\bar{R}$	5	7	9	16
		$\max S$	0	2	4	11
News-4D	0.2942	$\bar{R}$	8	11	23	48
		$\max S$	3	5	16	38
Drive-4D	0.3591	$\bar{R}$	11	17	41	63
		$\max S$	2	3	12	36
Data	ε_0		$\theta = 1$	$\theta = 2$	$\theta = 3$	$\theta = 4$
$\mathcal{H}_3$	0.1081	$\bar{R}$	38	132	247	353
		$\max S$	3	6	10	13
$\mathcal{H}_4$	0.1676	$\bar{R}$	9	19	26	40
		$\max S$	1	3	4	6
Data	ε_0		$\theta = 0.10$	$\theta = 0.15$	$\theta = 0.20$	$\theta = 0.25$
$\mathcal{G}_3$	0.0961	$\bar{R}$	5	36	39	43
		$\max S$	0	1	1	2
$\mathcal{G}_4$	0.1130	$\bar{R}$	5	43	26	40
		$\max S$	0	1	4	6

For most tensors, if we do not change the numbers of components, the mean fits of all methods would be lower than the initialization fits. By increasing the number of components, iALS and iCP-AM can maintain the fits and even obtain better results than the initialization fits. The gap between the iCP-AM (iALS) fit and the ALS fit is less than 0.02 for most cases. As for running time, iALS and iCP-AM also perform very well. Note that the ALS has a good initialization from the results of the last step, and the ALS running time is already very short. Compared to ALS, iALS and iCP-AM can still save more than 50 $\times$ running time on average.

Now we test "Drive" and the synthetic tensors with rapidly increasing rank. The test procedure is similar to that for the real-world tensors. The first setting is $R_0 = 5$ for all methods, S_t is obtained by (25), with $\theta = 10$ for Drive-3D, and Drive-4D, and with $\theta = 0.2$ for $\mathcal{G}_3$ and $\mathcal{G}_4$; the number of components of OnlineCP is fixed at five for all steps, and iALS, iCP-AM, and ALS have the same number of components R_t for the tth step. Suppose the final number of components corresponding to iCP-AM is $\bar{R}_1$. The second setting is that $R_0 = \bar{R}_1$ and $S_t = 0$ for all t. Because iCP-AM for $S = 0$ is the same as iALS for $S = 0$, we only test OnlineCP and iALS for this setting.

As shown in Table 5, gradually increasing the number of components can obtain a good performance for iALS and iCP-AM. iCP-AM outperforms iALS in terms of both the fit and the running time. The iCP-AM fit is worse than the ALS fit, but iCP-AM can save more than 100 $\times$ running time. For this type of data, setting a large enough number of components at the initialization stage cannot yield a good

TABLE 4

Results of different methods on Hall, News, and synthetic tensors with stable rank. Here "ini" is the initialization, and $\bar{R}$ is the final number of components. The fit is shown in percentage, and the running time is measured in seconds.

Data		ini	$\bar{R}=5$			$\bar{R}=6$			$\bar{R}=7$		
			OnlineCP	ALS	iALS	ALS	iALS	iCP-AM	ALS	iALS	iCP-AM
Hall-3D	fit	84.84	83.56	83.64	83.56	84.57	84.11	84.10	85.42	84.68	84.68
	time	1.1e+0	1.0e-3	4.7e-2	1.0e-3	7.2e-2	1.3e-3	1.4e-3	8.0e-2	1.4e-3	1.4e-3
News-3D	fit	74.12	70.46	71.62	70.63	72.86	72.58	72.58	73.72	72.81	72.83
	time	1.1e+0	1.0e-3	6.3e-2	1.0e-3	6.9e-2	1.2e-3	1.3e-3	1.5e-1	1.3e-3	1.5e-3
$\mathcal{H}_3$	fit	89.19	88.48	88.48	88.48	89.96	89.33	89.36	90.14	89.44	89.44
	time	5.3e-2	1.0e-3	3.5e-2	1.0e-3	4.4e-2	1.1e-3	1.2e-3	5.2e-2	1.1e-3	1.2e-3
Hall-4D	fit	83.31	82.19	82.25	82.19	83.23	82.76	82.76	84.16	83.48	83.48
	time	3.0e+0	2.8e-3	1.2e-1	3.0e-3	2.6e-1	4.0e-3	4.0e-3	2.8e-1	4.4e-3	4.4e-3
News-4D	fit	70.58	66.85	67.49	66.89	69.21	69.14	69.14	71.02	69.58	69.58
	time	3.0e+0	3.0e-3	2.7e-1	3.1e-3	2.9e-1	4.0e-3	4.0e-3	7.4e-1	4.1e-3	4.1e-3
$\mathcal{H}_4$	fit	83.24	84.86	84.86	84.86	89.33	86.86	86.86	89.33	87.24	87.24
	time	2.0e-1	3.4e-3	2.1e-1	3.5e-3	2.7e-1	4.0e-3	4.0e-3	4.1e-1	4.2e-3	4.1e-3

TABLE 5

Results of different methods on Drive and synthetic tensors with rapidly increasing rank. Here R_0 is the number of components corresponding to the initialization, and $S_t = R_{t+1} - R_t$. Boldface indicates the best result among the incremental methods.

		$R_0=5, S_t\geq 0$					$R_0=16, S_t=0$		
		ini	OnlineCP	iALS	iCP-AM	ALS	ini	OnlineCP	iALS
Drive-3D	fit	64.82	51.91	66.85	**67.10**	68.91	70.93	56.28	**59.71**
	time	1.3e+1	**1.7e-2**	5.5e-2	4.4e-2	1.4e+1	3.4e+1	**3.6e-2**	4.0e-2
	$\bar{R}$	-	5	16	16	16	-	16	16
		$R_0=5, S_t\geq 0$					$R_0=39, S_t=0$		
		ini	OnlineCP	iALS	iCP-AM	ALS	ini	OnlineCP	iALS
$\mathcal{G}_3$	fit	90.39	14.55	83.33	**83.47**	89.15	90.51	57.94	**61.33**
	time	4.3e-2	**1.0e-3**	4.1e-3	2.7e-3	2.7e-1	1.5e+0	**3.2e-3**	3.7e-3
	$\bar{R}$	-	5	39	39	39	-	39	39
		$R_0=5, S_t\geq 0$					$R_0=17, S_t=0$		
		ini	OnlineCP	iALS	iCP-AM	ALS	ini	OnlineCP	iALS
Drive-4D	fit	64.09	50.76	65.01	**66.32**	68.37	69.12	60.64	**61.09**
	time	4.9e+1	**6.4e-2**	3.0e-1	2.7e-1	5.0e+1	1.4e+2	**1.6e-1**	1.7e-1
	$\bar{R}$	-	5	17	17	17	-	17	17
		$R_0=5, S_t\geq 0$					$R_0=48, S_t=0$		
		ini	OnlineCP	iALS	iCP-AM	ALS	ini	OnlineCP	iALS
$\mathcal{G}_4$	fit	88.70	9.09	71.47	**77.02**	82.05	88.72	55.67	**59.18**
	time	2.2e-1	**3.4e-3**	1.8e-2	1.5e-2	2.5e+0	6.0e+0	**1.8e-2**	2.1e-2
	$\bar{R}$	-	5	48	48	48	-	48	48

result for OnlineCP and iALS. This is shown in the second setting of the test.

The fits of "Drive" corresponding to ALS, iCP-AM, and iALS in the first setting are shown in Figure 4. We can see that the iCP-AM fit is higher than the iALS fit at all timestamps.

5.4. Scalability evaluation. In Table 1, we showed the time costs of ALS, iALS($S>0$), and iCP-AM. Because the time cost of the batch method ALS is much longer than that of the two incremental algorithms, we only test the scalability of iALS($S>0$) and iCP-AM.

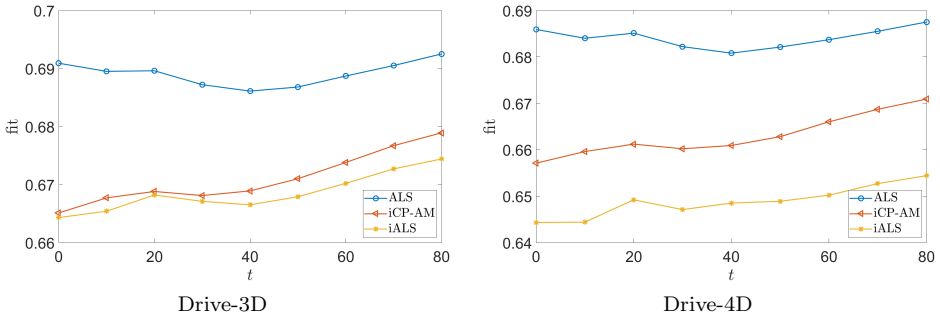

FIG. 4. *The fits at the tth step for Drive-3D and Drive-4D.*

We generate a third-order tensor with size $20 \times 20 \times 10^5$ and a fourth-order tensor with size $7 \times 7 \times 7 \times 10^5$. For each tensor $\mathcal{M}$, define a sequence of tensors $\mathcal{M}_t := \mathcal{M}(:,:,10^4+t), t = 1,\ldots,9\times 10^4$. We run iALS and iCP-AM with $S_t = 1$ on $\mathcal{M}_t, t = 1,\ldots,9\times 10^4$. Both methods are forced to execute one iteration, and the ALS in iCP-AM for $N \geq 4$ is also forced to execute one iteration.

We show the running time for each t in Figure 5. Although the time cost of iCP-AM is constant with respect to the length of the data, the assignment of $\hat{\mathbf{A}}^{(N)}$ in line 15 of Algorithm 3 would take a longer time as the length of the data increases. Therefore, we can find that the running time of iCP-AM increases slightly as t increases. In contrast to iCP-AM, the running time of iALS increases much faster as t increases.

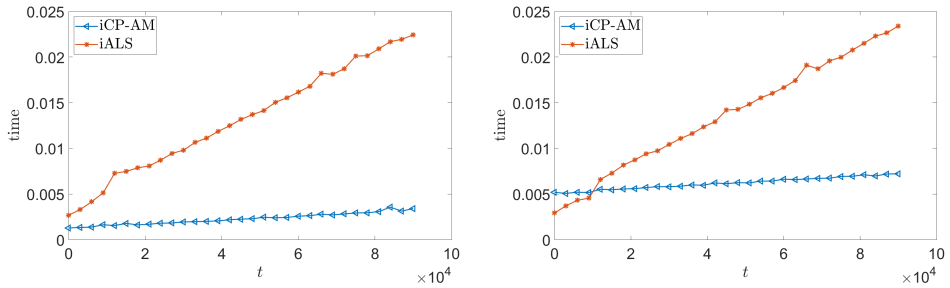

FIG. 5. *Scalability comparison between iALS and iCP-AM. The left figure is the running time (in seconds) for adding one slice to a* $20 \times 20 \times (10^4+t-1)$ *tensor at time t, and the right figure is the running time for adding one slice to a* $7 \times 7 \times 7 \times (10^4+t-1)$ *tensor at time t, where* $S_t = 1$ *for all cases.*

6. Related work. The earliest work on incremental CP decompositions is by Nion and Sidiropoulos [32]. Their approach concerns third-order tensors only. They first update the third mode factor matrix by the least squares method, and then update the first two factor matrices simultaneously by using matrix SVD. The major drawback of this work is that it cannot be extended for higher-order tensors. In [55], Zhou et al. propose OnlineCP. Unlike [32], OnlineCP can handle tensors with arbitrary orders. They first update the last factor matrix by the least squares method, and then update the first $N-1$ factor matrices one by one through the Jacobi iteration process. In [42], Smith et al. propose CP-stream. We have already introduced this algorithm in section 2.4.1 in detail. All of these three methods assume that the number of components is fixed in the incremental process.

In [51], Vandecappelle, Verliet, and De Lathauwer extend the nonlinear least squares (NLS) approach to the incremental case, with the framework for the efficient representation of structured tensors described in [52]. For details of NLS for the batch case, we refer the reader to [1, 43, 49] and references therein. This approach aims to solve (8) by NLS. If the number of components does not change, this method benefits from the fact that the solution of $\hat{\mathbf{A}}^{(n)}$ is close to $\mathbf{A}^{(n)}$ for $n = 1, \ldots, N-1$, and even one iteration of the Gauss–Newton method can achieve a good result. The case where the number of components changes is handled by setting a semihot starting point, such as iALS($S > 0$). Because NLS is slower than ALS [1, 49] in general, this method is slower than iALS; compare the results of [51] and those of [55].

In [33], Pasricha, Gurjal, and Papalexakis propose a method for adding new components. In their method, several slices are added simultaneously. That is, the new tensor is $\mathcal{W} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times (I_N + I_M)}$ with $\mathcal{W}(:, \ldots, :, 1 : I_N) = \mathcal{X}$ and $\mathcal{Y} = \mathcal{W}(:, \ldots, :, I_N + 1 : I_M)$ being the new data. First, they compute the CP decomposition of $\mathcal{Y}$ with R' components, where R' is user-provided. Suppose $\mathscr{B}' = \{\mathcal{B}'_1, \ldots, \mathcal{B}'_{R'}\}$ is a rank-one basis of this decomposition of $\mathcal{Y}$. Then they measure the similarity between $\mathcal{B}'_r$ with each $\mathcal{B}_s, 1 \le s \le R$ by computing the inner product between $\mathcal{B}'_r$ and $\mathcal{B}_s$. With some prescribed threshold, if $\mathcal{B}'_r$ is not similar to any $\mathcal{B}_s$, then $\mathcal{B}'_r$ is added into $\mathscr{B}$. This method has some theoretical issues. Unlike the matrix low-rank approximation, the tensor low-rank approximation problem has no solution in general. Even for the same tensor, the results yielded by ALS with different starting points can be totally different, resulting in totally different rank-one bases. This introduces difficulties when comparing $\mathscr{B}$ and $\mathscr{B}'$. In fact, the numerical results in [33] show that this method is even worse than OnlineCP on some data. In [27], Letourneau et al. refine the method of [33] by combining it with other approaches, e.g., ALS. As shown in the numerical part of [27], there is a moderate gap between the fits of this method and those of the corresponding batch method.

In addition to CP decomposition, HOSVD [14] is also an important type of tensor decomposition. In the literature, there are some results and applications on incremental HOSVD; see [23, 29, 36, 47]. Their basic idea is to study SVDs of unfolding matrices by using the incremental matrix SVD [6, 7, 18, 30]. Recently, a randomized algorithm for truncated HOSVD of a streaming tensor is proposed in [48]. However, this method forms an approximation of the tensor only after all the data have been observed and hence is not an incremental algorithm in the setting of this paper.

7. Conclusions. We consider how to add components for incremental CP decomposition. By viewing the CP decomposition of a tensor as the coordinate representation of its subtensors with respect to a special rank-one basis, we discuss some properties of the incremental CP decomposition, such as the uniqueness and the variation of rank. An alternating minimization algorithm is proposed for adding components and a systematic scheme is designed by combining it with other incremental algorithms. Increasing the number of components can maintain a nonincreasing level of relative error. The experiments show that the algorithms have good performance in time cost and yield comparable fits compared with the standard algorithm ALS.

In the future, we will work on other types of datasets, e.g., network flow and social media data, including sparse tensors. The underlying structures of these data need to be exploited in designing incremental algorithms. Another future study is to consider the downdating algorithm [19] for finding the CP decomposition of a tensor by deleting a slice from an original tensor.

Acknowledgments. We are extremely grateful to the associate editor and three anonymous referees for their valuable feedback, which improved this paper significantly.

REFERENCES

[1] E. Acar, D. M. Dunlavy, and T. G. Kolda, *A scalable optimization approach for fitting canonical tensor decompositions*, J. Chemometrics, 25 (2011), pp. 67–86.
[2] B. W. Bader, T. G. Kolda, et al., *MATLAB Tensor Toolbox Version* 3.0-*dev.*, https://www.tensortoolbox.org, 2017.
[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, https://doi.org/10.1137/17M1112303.
[4] J. C. Bezdek and R. J. Hathaway, *Some notes on alternating optimization*, in Proceedings of the AFSS International Conference on Fuzzy Systems, Springer, New York, 2002, pp. 288–300.
[5] J. Bolte, S. Sabach, and M. Teboulle, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Math. Programming, 146 (2014), pp. 459–494.
[6] M. Brand, *Fast low-rank modifications of the thin singular value decomposition*, Linear Algebra Appl., 415 (2006), pp. 20–30.
[7] J. R. Bunch and C. P. Nielsen, *Updating the singular value decomposition*, Numer. Math., 31 (1978), pp. 111–129.
[8] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math., 9 (2009), 717.
[9] E. J. Candès and T. Tao, *The power of convex relaxation: Near-optimal matrix completion*, IEEE Trans. Inform. Theory, 56 (2010), pp. 2053–2080.
[10] J. D. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition*, Psychometrika, 35 (1970), pp. 283–319.
[11] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and D. P. Mandic, *Tensor networks for dimensionality reduction and large-scale optimization: Part* 1*: Low-rank tensor decompositions*, Found. Trends Mach. Learn., 9 (2016), pp. 249–429.
[12] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, *Tensor decompositions for signal processing applications: From two-way to multiway component analysis*, IEEE Signal Process. Mag., 32 (2015), pp. 145–163.
[13] L. De Lathauwer, *A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 642–666, https://doi.org/10.1137/040608830.
[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] 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, https://doi.org/10.1137/06066518X.
[16] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika, 1 (1936), pp. 211–218.
[17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, *Vision meets robotics: The kitti dataset*, Internat. J. Robotics Res., 32 (2013), pp. 1231–1237.
[18] M. Gu and S. C. Eisenstat, *A Stable and Fast Algorithm for Updating the Singular Value Decomposition.* Research report YALEU/DCS/RR-966, Department of Computer Science, Yale University, New Haven, CT, 1993.
[19] M. Gu and S. C. Eisenstat, *Downdating the singular value decomposition*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 793–810, https://doi.org/10.1137/S0895479893251472.
[20] R. A. Harshman, *Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis*, UCLA Working Papers in Phonetics, 16 (1970), pp. 1–84.
[21] J. Håstad, *Tensor rank is NP-complete*, J. Algorithms, 11 (1990), pp. 644–654.
[22] C. J. Hillar and L.-H. Lim, *Most tensor problems are NP-hard*, J. ACM (JACM), 60 (2013), 45.
[23] W. Hu, X. Li, X. Zhang, X. Shi, S. Maybank, and Z. Zhang, *Incremental tensor subspace learning and its applications to foreground segmentation and tracking*, Int. J. Comput. Vis., 91 (2011), pp. 303–327.

[24] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500, https://doi.org/10.1137/07070111X.
[25] J. B. Kruskal, *Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics*, Linear Algebra Appl., 18 (1977), pp. 95–138.
[26] R. M. Larsen, *PROPACK—Software for Large and Sparse SVD Calculations*, http://sun.stanford.edu/~rmunk/PROPACK/.
[27] P.-D. Letourneau, M. Baskaran, T. Henretty, J. Ezick, and R. Lethin, *Computationally efficient CP tensor decomposition update framework for emerging component discovery in streaming data*, in Proceedings of the 2018 IEEE High Performance Extreme Computing Conference (HPEC), IEEE, Piscataway, NJ, 2018, pp. 1–8.
[28] X. Liu and N. D. Sidiropoulos, *Cramér-Rao lower bounds for low-rank decomposition of multidimensional arrays*, IEEE Trans. Signal Process., 49 (2001), pp. 2074–2086.
[29] X. Ma, D. Schonfeld, and A. Khokhar, *Dynamic updating and downdating matrix SVD and tensor HOSVD for adaptive indexing and retrieval of motion trajectories*, in Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, Piscataway, NJ, 2009, pp. 1129–1132.
[30] M. Moonen, P. Van Dooren, and J. Vandewalle, *A singular value decomposition updating algorithm for subspace tracking*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 1015–1038, https://doi.org/10.1137/0613061.
[31] M. Mørup, *Applications of tensor (multiway array) factorizations and decompositions in data mining*, WIREs Data Mining Knowl. Discov., 1 (2011), pp. 24–40.
[32] D. Nion and N. D. Sidiropoulos, *Adaptive algorithms to track the PARAFAC decomposition of a third-order tensor*, IEEE Trans. Signal Process., 57 (2009), pp. 2299–2310.
[33] R. Pasricha, E. Gujral, and E. E. Papalexakis, *Identifying and alleviating concept drift in streaming tensor decomposition*, in Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, New York, 2018, pp. 327–343.
[34] B. Recht, M. Fazel, and P. A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM Rev., 52 (2010), pp. 471–501, https://doi.org/10.1137/070697835.
[35] M. J. Reynolds, A. Doostan, and G. Beylkin, *Randomized alternating least squares for canonical tensor decompositions: Application to a PDE with random data*, SIAM J. Sci. Comput., 38 (2016), pp. A2634–A2664, https://doi.org/10.1137/15M1042802.
[36] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, *Incremental learning for robust visual tracking*, Int. J. Comput. Vis., 77 (2008), pp. 125–141.
[37] P. Seeling and M. Reisslein, *Video transport evaluation with H.264 video traces*, IEEE Commun. Surveys Tutorials, 14 (2013), pp. 1142–1165.
[38] N. D. Sidiropoulos and R. Bro, *On the uniqueness of multilinear decomposition of N-way arrays*, J. Chemometrics, 14 (2000), pp. 229–239.
[39] N. D. Sidiropoulos, G. B. Giannakis, and R. Bro, *Blind PARAFAC receivers for DS-CDMA systems*, IEEE Trans. Signal Process., 48 (2000), pp. 810–823.
[40] M. Signoretto, Q. T. Dinh, L. De Lathauwer, and J. A. Suykens, *Learning with tensors: A framework based on convex optimization and spectral regularization*, Mach. Learn., 94 (2014), pp. 303–351.
[41] A. Smilde, R. Bro, and P. Geladi, *Multi-way Analysis: Applications in the Chemical Sciences*, John Wiley & Sons, Chichester, 2005.
[42] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, *Streaming tensor factorization for infinite data sources*, in Proceedings of the 2018 SIAM International Conference on Data Mining, SIAM, Philadelphia, 2018, pp. 81–89, https://doi.org/10.1137/1.9781611975321.10.
[43] L. Sorber, M. Van Barel, and L. De Lathauwer, *Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-$(L_r, L_r, 1)$ terms, and a new generalization*, SIAM J. Optim., 23 (2013), pp. 695–720, https://doi.org/10.1137/120868323.
[44] A. Stegeman, *On uniqueness of the nth order tensor decomposition into rank-1 terms with linear independence in one mode*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2498–2516, https://doi.org/10.1137/090779632.
[45] H. De Sterck, *A nonlinear GMRES optimization algorithm for canonical tensor decomposition*, SIAM J. Sci. Comput., 34 (2012), pp. A1351–A1379, https://doi.org/10.1137/110835530.
[46] J. Sun, D. Tao, and C. Faloutsos, *Beyond streams and graphs: Dynamic tensor analysis*, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, 2006, pp. 374–383.
[47] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, *Incremental tensor analysis: Theory and applications*, ACM Trans. Knowl. Discov. Data (TKDD), 2 (2008), 11.

[48] Y. Sun, Y. Guo, C. Luo, J. Tropp, and M. Udell, *Low-rank Tucker approximation of a tensor from streaming data*, SIAM J. Math. Data Sci., 2 (2020), pp. 1123–1150, https://doi.org/10.1137/19M1257718.
[49] G. Tomasi and R. Bro, *A comparison of algorithms for fitting the PARAFAC model*, Comput. Statist. Data Anal., 50 (2006), pp. 1700–1734.
[50] P. Tseng, *Convergence of a block coordinate descent method for nondifferentiable minimization*, J. Optim. Theory Appl., 109 (2001), pp. 475–494.
[51] M. Vandecappelle, N. Vervliet, and L. De Lathauwer, *Nonlinear least squares updating of the canonical polyadic decomposition*, in Proceedings of the 2017 25th European Signal Processing Conference (EUSIPCO), IEEE, Piscataway, NJ, 2017, pp. 663–667.
[52] N. Vervliet, O. Debals, and L. De Lathauwer, *Exploiting efficient representations in large-scale tensor decompositions*, SIAM J. Sci. Comput., 41 (2019), pp. A789–A815, https://doi.org/10.1137/17M1152371.
[53] S. J. Wright, *Coordinate descent algorithms*, Math. Programming, 151 (2015), pp. 3–34.
[54] Q. Zhao, L. Zhang, and A. Cichocki, *Bayesian CP factorization of incomplete tensors with automatic rank determination*, IEEE Trans. Pattern Anal. Mach. Intell., 37 (2015), pp. 1751–1763.
[55] S. Zhou, N. X. Vinh, J. Bailey, Y. Jia, and I. Davidson, *Accelerating online CP decompositions for higher order tensors*, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, 2016, pp. 1375–1384.