A Simpler Approach to Low-Rank Tensor Canonical Polyadic Decomposition

Daniel L. Pimentel-Alarcón
University of Wisconsin-Madison

Abstract— **In this paper we present a simple and efficient method to compute the canonical polyadic decomposition (CPD) of generic low-rank tensors using elementary linear algebra. The key insight is that all the columns in a low-rank tensor lie in a low-dimensional subspace, and that the coefficients of the columns in each *slice* with respect to *the right* basis are scaled copies of one an other. The basis, together with the coefficients of a few carefully selected columns determine the CPD. The computational complexity of our method scales linearly in the order and the rank of the tensor, and at most quadratically in its largest dimension. Furthermore, our approach can be easily adapted to noisy settings. We complement our theoretical analysis with experiments that support our findings.**

I. Introduction

The canonical polyadic decomposition (CPD) problem consists on writing a tensor as a minimal sum of rank-1 tensors [1]–[3]. This is ubiquitous in many modern applications of chemical sciences, data analysis, signal processing, chemometrics and psychometrics, to name a few [4]–[9]. Existing methods that compute the CPD range from alternating minimization [2], [10] to simultaneous diagonalization [11]–[13], line search [14] and generalized eigenvalue decomposition [15]–[17], among others [18]–[20].

In this paper we present a simple and efficient method to compute the CPD of generic low-rank tensors using elementary linear algebra. The key idea behind our approach is that all the columns in a low-rank tensor lie in a low-dimensional subspace, and that the coefficients of the columns in each *slice* with respect to *the right* basis are scaled copies of one an other. So one only needs to compute a few of these coefficients to determine the whole tensor. The basis, together with the coefficients of a few carefully selected columns determine the CPD. We show that the computational complexity of our method scales linearly in the order and the rank of the tensor, and at most quadratically in its largest dimension. Furthermore, our approach can be easily extended to noisy settings. We complement our theoretical analysis with experiments that support our findings.

Organization of the paper

In Section II we formally state the CPD problem, our CPD algorithm, and our main result, showing that our method will yield the CPD of almost every low-rank tensor. In Section III we study the computational complexity of this algorithm. In Section IV we explain how to easily extend our results to noisy settings. In Section V we present synthetic experiments that support our theoretical results.

Notation

When dealing with tensors, notation can easily get out of hand. To avoid clutter and confusion, we will use the following notations.

	Examples	Regular	Bold	Lower	Capital	Roman	Script
Scalar	K, R	✓			✓		✓
Index	i, j, k, r	✓		✓			✓
Vector	$\mathbf{u}$		✓	✓		✓	
Matrix	$\mathbf{U}$		✓		✓	✓	
Tensor	$\mathfrak{X}$		✓		✓		✓
Entry	u_i	✓		✓		✓	

We will also use $[\cdot]$ as shorthand for $\{1,2,\dots,\cdot\}$. For example, $[K]$ is shorthand for $\{1,2,\dots,K\}$. Throughout the document, k will be an index in $[K]$ and r will be an index in $[R]$.

II. Model and Main Results

Let $\mathfrak{X}$ be a K-order, rank-R tensor of dimensions $D_1 \geq D_2 \geq \cdots \geq D_K$. By definition, $\mathfrak{X}$ can be written as

$$\mathfrak{X} = \sum_{r=1}^{R} \bigotimes_{k=1}^{K} \mathbf{u}^{kr} \tag{1}$$

for some $\{\mathbf{u}^{kr}\}_{k,r=1}^{K,R}$, where $\mathbf{u}^{kr} \in \mathbb{R}^{D_k}$ for every $k \in [K]$ and $r \in [R]$. Our goal is to recover $\{\mathbf{u}^{kr}\}_{k,r=1}^{K,R}$ from $\mathfrak{X}$ (up to scaling factors). This is known as the canonical polyadic decomposition (CPD) of $\mathfrak{X}$. See Figure 1 for some intuition.

Throughout the paper, we will assume that

A1 For each k and r, $\mathbf{u}^{kr}$ is drawn independently according to an absolutely continuous distribution with respect to the Lebesgue measure on $\mathbb{R}^{D_k}$.

A2 $K > 2$ and $R \leq D_2$.

A1 essentially requires that $\mathfrak{X}$ is a generic rank-R tensor. Intuitively, a rank-R tensor randomly selected with respect to a density will satisfy **A1** with probability 1. This is similar or equivalent to other standard assumptions from the literature [8], [10]–[13], [21]–[25]. See Figure 2 to build some intuition. **A2** essentially requires that $\mathfrak{X}$ is a tensor (as opposed to a vector, or a matrix), and that at least two of its dimensions are as large as R.

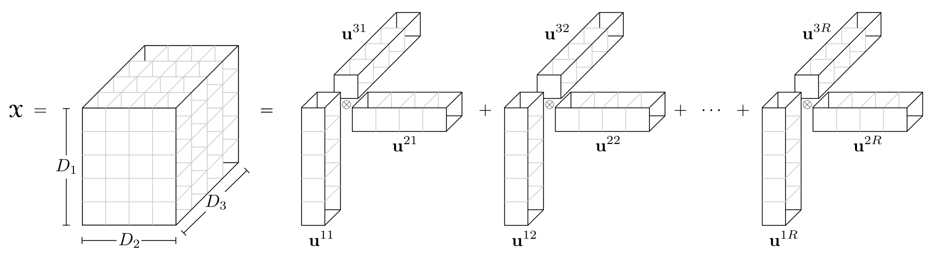


Fig. 1: A rank-R tensor $\mathcal{X}$ can be written as the sum of R rank-1 tensors. In this figure, each rank-1 tensor is given by $\mathbf{u}^{1r} \otimes \mathbf{u}^{2r} \otimes \mathbf{u}^{3r}$.

The main contribution of the paper is a simple and efficient method to compute the CPD of generic low-rank tensors using elementary linear algebra (Algorithm 1 below). Let $\mathbf{U}^k := [\mathbf{u}^{k1}\ \mathbf{u}^{k2}\ \cdots\ \mathbf{u}^{kR}] \in \mathbb{R}^{D_k \times R}$. The key insight is that under **A1**-**A2**, all the columns in $\mathcal{X}$ lie in the R-dimensional subspace spanned by $\mathbf{U}^1$, and that the coefficients of the *slices* of $\mathcal{X}$ with respect to $\mathbf{U}^1$ are copies of one an other, scaled by the values in $\mathbf{U}^2, \mathbf{U}^3, \dots, \mathbf{U}^K$. So one only needs a few of these coefficients to determine $\mathbf{U}^2, \mathbf{U}^3, \dots, \mathbf{U}^K$ (up to scaling factors). With this in mind, we first find a basis $\hat{\mathbf{U}}^1$ containing scaled copies of the columns in $\mathbf{U}^1$. Next, we determine the coefficients of a few carefully selected columns in $\mathcal{X}$ with respect to this basis. These coefficients determine scaled copies of $\mathbf{U}^2, \mathbf{U}^3, \dots, \mathbf{U}^K$. Finally, we determine scaling factors to obtain the desired CPD.

More precisely, let $\mathcal{X}^{kr}$ be the k-order, rank-1 tensor given by $\mathbf{u}^{1r} \otimes \mathbf{u}^{2r} \otimes \cdots \otimes \mathbf{u}^{kr}$, and let $\mathcal{X}^k$ be the k-order, rank-R tensor given by $\sum_{r=1}^R \mathcal{X}^{kr}$. See Figure 3 to build some intuition. It is easy to see that $\mathcal{X}^K = \mathcal{X}$.

Next notice that $\mathcal{X}^{2r}$ contains D_2 copies of $\mathcal{X}^{1r} = \mathbf{u}^{1r}$, scaled by the entries in $\mathbf{u}^{2r}$. Similarly, $\mathcal{X}^{3r}$ contains D_3 copies of $\mathcal{X}^{2r}$, scaled by the entries in $\mathbf{u}^{3r}$. Equivalently, $\mathcal{X}^{3r}$ contains $D_2 D_3$ copies of $\mathbf{u}^{1r}$, each scaled by the product of a combination of the entries in $\mathbf{u}^{2r}$ and $\mathbf{u}^{3r}$. Proceeding iteratively, we see that $\mathcal{X}^{kr}$ contains D_k copies of $\mathcal{X}^{(k-1)r}$, scaled by the combinations of the entries in $\mathbf{u}^{kr}$. Equivalently, $\mathcal{X}^{kr}$ contains $D_2 D_3 \cdots D_k$ copies of $\mathbf{u}^{1r}$, each scaled by the product of a combination of the entries in $\mathbf{u}^{2r}, \mathbf{u}^{3r}, \dots, \mathbf{u}^{kr}$.

This implies that $\mathcal{X}^{Kr}$ contains $D_2 D_3 \cdots D_K$ scaled copies of $\mathbf{u}^{1r}$. Since $\mathcal{X} = \mathcal{X}^K = \sum_{r=1}^R \mathcal{X}^{Kr}$, it follows that $\mathcal{X}$ contains $D_2 D_3 \cdots D_K$ linear combinations of the columns in $\mathbf{U}^1$. Equivalently, all the columns in $\mathcal{X}$ lie in the subspace spanned by $\mathbf{U}^1$.

We will now show how to recover scaled copies of the columns in $\mathbf{U}^1$. For any $i \in [D_3]$, define the matrix $\mathbf{X}^{2i} \in \mathbb{R}^{D_1 \times D_2}$ as

$$\mathbf{X}^{2i} := \sum_{r=1}^{R} \left(\mathbf{u}^{1r} \otimes \mathbf{u}^{2r}\right) \mathrm{u}_i^{3r} \left(\mathrm{u}_1^{4r} \cdots \mathrm{u}_1^{Kr}\right) = \mathbf{U}^1 (\mathbf{U}^2 \mathbf{D}^i)^\mathsf{T},$$

where $\mathbf{D}^i \in \mathbb{R}^{R \times R}$ is the diagonal matrix with $\{\mathrm{u}_i^{3r} \prod_{k=4}^K \mathrm{u}_1^{kr}\}_{r=1}^R$ as diagonal entries. Intuitively, $\mathbf{X}^{2i}$ is the matrix of $\mathcal{X}$ in dimensions D_1 and D_2, located at position $\{i, 1, 1, \dots, 1\}$ in dimensions $D_3, D_4, \dots, D_K$. See Figure 4 for some intuition.

The matrix $\mathbf{\Theta}^{2i} := (\mathbf{U}^2 \mathbf{D}^i)^\mathsf{T} \in \mathbb{R}^{R \times D_2}$ contains the coefficients of $\mathbf{X}^{2i}$ with respect to the basis $\mathbf{U}^1$. Notice that for any $j \in [D_3]$, the rows of $\mathbf{\Theta}^{2j}$ are scaled copies of the rows in $\mathbf{\Theta}^{2i}$. In other words, the coefficients $\mathbf{\Theta}^{2j}$ of the columns in $\mathbf{X}^{2j}$ (with respect to the basis $\mathbf{U}^1$) are scaled copies of the coefficients $\mathbf{\Theta}^{2i}$ of the columns in $\mathbf{X}^{2i}$ (with respect to the basis $\mathbf{U}^1$).

Under **A2**, $D_1 \geq R$, which together with **A1** implies that any R columns of $\mathbf{X}^{2i}$ will be linearly independent with probability 1. By **A2**, $D_2 \geq R$, whence $\mathbf{X}^{2i}$ has at least R columns. We can thus set $\tilde{\mathbf{U}}^1$ to be the $D_1 \times R$ matrix formed with the first R columns of $\mathbf{X}^{2i}$, such that $\tilde{\mathbf{U}}^1$ spans the same subspace as $\mathbf{U}^1$. Now take $\mathbf{X}^{2j}$, with $j \neq i$. It follows that the columns in $\mathbf{X}^{2j}$ lie in the subspace spanned by $\tilde{\mathbf{U}}^1$. We can obtain the coefficients of $\mathbf{X}^{2j}$ in this basis, given by $\tilde{\mathbf{\Theta}}^{2j} = (\tilde{\mathbf{U}}^{1\mathsf{T}} \tilde{\mathbf{U}}^1)^{-1} \tilde{\mathbf{U}}^{1\mathsf{T}} \mathbf{X}^{2j}$, such that $\mathbf{X}^{2j} = \tilde{\mathbf{U}}^1 \tilde{\mathbf{\Theta}}^{2j}$. In general, $\tilde{\mathbf{U}}^1 \neq \mathbf{U}^1$, and so the coefficients $\tilde{\mathbf{\Theta}}^{2j}$ of the columns in $\mathbf{X}^{2j}$ (with respect to the basis $\tilde{\mathbf{U}}^1$) are not scaled copies of the coefficients $\tilde{\mathbf{\Theta}}^{2i}$ of the columns in $\mathbf{X}^{2i}$ (with respect to the basis $\tilde{\mathbf{U}}^1$). In other words, the rows of $\tilde{\mathbf{\Theta}}^{2j}$ will not be scaled copies of the rows in $\tilde{\mathbf{\Theta}}^{2i}$.

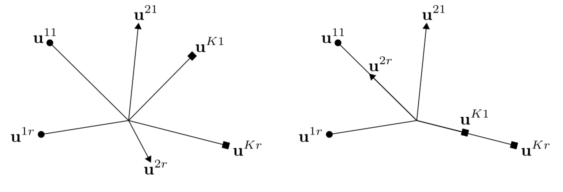
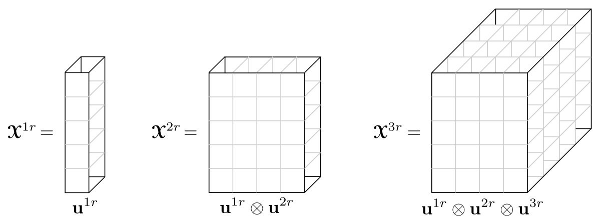


Fig. 2: Each K-order, rank-R tensor $\mathcal{X}$ is determined by a set of vectors $\{\mathbf{u}^{kr}\}_{k,r=1}^{K,R}$. In the **left**, these vectors are in general position, that is, each $\mathbf{u}^{kr}$ is drawn according to an absolutely continuous distribution with respect to the Lebesgue measure on $\mathbb{R}^{D_k}$. For example, according to a gaussian distribution. In this case, the probability that the $\mathbf{u}^{kr}$'s are as in the **right**, where some $\mathbf{u}^{kr}$'s are perfectly aligned with others, is zero. Our results hold for all low-rank tensors, except for a set of measure zero of pathological cases as in the right.



$\mathcal{X}^{1r} =$
$\mathbf{u}^{1r}$
$\mathcal{X}^{2r} =$
$\mathbf{u}^{1r} \otimes \mathbf{u}^{2r}$
$\mathcal{X}^{3r} =$
$\mathbf{u}^{1r} \otimes \mathbf{u}^{2r} \otimes \mathbf{u}^{3r}$

Fig. 3: $\mathcal{X}^{kr}$ is the k-order, rank-1 tensor given by $\mathbf{u}^{1r} \otimes \mathbf{u}^{2r} \otimes \cdots \otimes \mathbf{u}^{kr}$. $\mathcal{X}^k$ is the k-order, rank-R tensor given by $\sum_{r=1}^R \mathcal{X}^{kr}$. Notice that $\mathcal{X}^K = \mathcal{X}$.

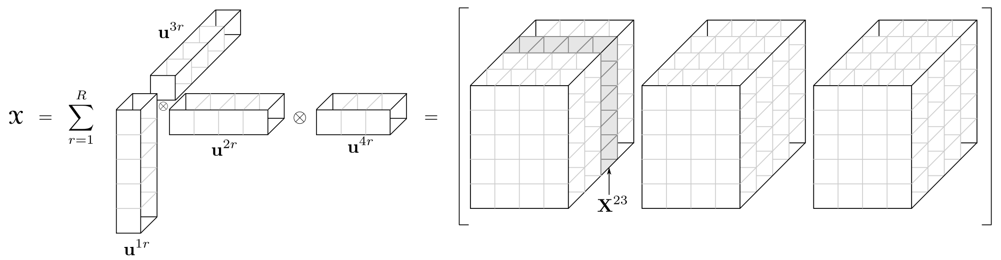


Fig. 4: Intuitively, $\mathbf{X}^{2i}$ is the i^{th} matrix of $\mathcal{X}$ in dimensions D_1 and D_2.

Recall that for any full-rank matrix $\mathbf{\Gamma} \in \mathbb{R}^{R\times R}$, $\tilde{\mathbf{U}}^1\mathbf{\Gamma}$ is also a basis of the subspace spanned by $\tilde{\mathbf{U}}^1$. The coefficients of $\mathbf{X}^{2i}$ with respect to the basis $\tilde{\mathbf{U}}^1\mathbf{\Gamma}$ are given by $\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2i}$. Since $\tilde{\mathbf{U}}^1$ spans the same subspace as $\mathbf{U}^1$, and the coefficients of $\mathbf{X}^{2j}$ with respect to $\mathbf{U}^1$ are scaled copies of the coefficients of $\mathbf{X}^{2i}$ with respect to $\mathbf{U}^1$, we know that there exists a change of basis $\mathbf{\Gamma}$ such that the rows of $\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2j}$ are scaled copies of the rows in $\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2i}$. More precisely, there exists a full-rank matrix $\mathbf{\Gamma} \in \mathbb{R}^{R\times R}$ and a diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{R\times R}$, such that

$$\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2j} = \mathbf{\Lambda}\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2i}. \tag{2}$$

Let $\tilde{\mathbf{\Theta}}^{2i}_R$ denote the $R\times R$ matrix with the first R columns of $\tilde{\mathbf{\Theta}}^{2i}$, and similarly for $\tilde{\mathbf{\Theta}}^{2j}_R$. Since we defined $\tilde{\mathbf{U}}^1$ to be the first R columns of $\mathbf{X}^{2i}$, it follows that $\tilde{\mathbf{\Theta}}^{2i}_R$ is the identity matrix $\mathbf{I}$. Restricting (2) to these first R columns, we have that

$$\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}^{2j}_R = \mathbf{\Lambda}\mathbf{\Gamma}^{-1}. \tag{3}$$

Left-multiplying by $\mathbf{\Gamma}$, we can rewrite this as $\tilde{\mathbf{\Theta}}^{2j}_R = \mathbf{\Gamma}\mathbf{\Lambda}\mathbf{\Gamma}^{-1}$, and from this we can see that $(\mathbf{\Gamma},\mathbf{\Lambda})$ is the eigenvalue decomposition of $\tilde{\mathbf{\Theta}}^{2j}_R$. It follows that the columns in $\hat{\mathbf{U}}^1 := \tilde{\mathbf{U}}^1\mathbf{\Gamma}$ are scaled copies of the columns in $\mathbf{U}^1$.

We will now show that for $k>1$ we can also recover scaled copies of $\mathbf{U}^k$. For any $k>1$, define the matrix $\mathbf{X}^k \in \mathbb{R}^{D_1\times D_k}$ as

$$\begin{aligned}\mathbf{X}^k &:= \sum_{r=1}^{R}\left(\mathbf{u}^{1r}\otimes\mathbf{u}^{kr}\right)\left(\mathrm{u}_1^{2r}\cdots\mathrm{u}_1^{(k-1)r}\cdot\mathrm{u}_2^{(k+1)r}\cdots\mathrm{u}_1^{Kr}\right)\\ &= \mathbf{U}^1(\mathbf{U}^k\mathbf{D}^k)^{\mathsf{T}},\end{aligned}$$

where $\mathbf{D}^k \in \mathbb{R}^{R\times R}$ is the diagonal matrix with $\{\mathrm{u}_1^{2r}\cdots\mathrm{u}_1^{(k-1)r}\cdot\mathrm{u}_2^{(k+1)r}\cdots\mathrm{u}_1^{Kr}\}_{r=1}^R$ as diagonal entries. Intuitively, $\mathbf{X}^k$ is the *face* of $\mathcal{X}$ in dimensions D_1 and D_k. Or more precisely, $\mathbf{X}^k$ is the matrix of $\mathcal{X}$ in dimensions D_1 and D_k, located at position $\{1,1,\dots,1\}$ in dimensions $D_2,D_3,\dots,D_K$. See Figure 5 for some intuition. In particular, $\mathbf{X}^2$ is equal to $\mathbf{X}^{21}$, as defined before.

Let $\hat{\mathbf{U}}^k \in \mathbb{R}^{D_k\times R}$ be the matrix containing the coefficients of $\mathbf{X}^k$ with respect to the basis $\hat{\mathbf{U}}^1$, such that $\mathbf{X}^k = \hat{\mathbf{U}}^1\hat{\mathbf{U}}^{k\mathsf{T}}$. That is, $\hat{\mathbf{U}}^{k\mathsf{T}} := (\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1)^{-1}\hat{\mathbf{U}}^{1\mathsf{T}}\mathbf{X}^k$.

Since the columns in $\hat{\mathbf{U}}^1$ are scaled copies of the columns in $\mathbf{U}^1$, it follows that the columns in $\hat{\mathbf{U}}^k$ are scaled copies of the columns in $\mathbf{U}^k$. To see this, observe that if $\hat{\mathbf{U}}^1 = \mathbf{U}^1\mathbf{\Lambda}\mathbf{\Pi}$ for some diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{R\times R}$ and some permutation matrix $\mathbf{\Pi} \in \mathbb{R}^{R\times R}$, then

$$\begin{aligned}\mathbf{X}^k &= \mathbf{U}^1\mathbf{D}^{k\mathsf{T}}\mathbf{U}^{k\mathsf{T}} = \mathbf{U}^1(\mathbf{\Lambda}\mathbf{\Pi})(\mathbf{\Lambda}\mathbf{\Pi})^{-1}\mathbf{D}^{k\mathsf{T}}\mathbf{U}^{k\mathsf{T}}\\ &= (\mathbf{U}^1\mathbf{\Lambda}\mathbf{\Pi})(\mathbf{\Pi}^{-1}\mathbf{\Lambda}^{-1}\mathbf{D}^{k\mathsf{T}}\mathbf{U}^{k\mathsf{T}}) = \hat{\mathbf{U}}^1(\mathbf{U}^k\mathbf{D}^k\mathbf{\Lambda}^{-1}\mathbf{\Pi})^{\mathsf{T}}.\end{aligned}$$

Since $\hat{\mathbf{U}}^k$ is defined as the coefficient matrix of $\mathbf{X}^k$ with respect to the basis $\hat{\mathbf{U}}^1$, it follows that $\hat{\mathbf{U}}^k$ must be equal to $\mathbf{U}^k\mathbf{D}^k\mathbf{\Lambda}^{-1}\mathbf{\Pi}$. This implies that the columns in $\hat{\mathbf{U}}^k$ are scaled copies of the columns in $\mathbf{U}^k$. Equivalently, the vectors in $\{\hat{\mathbf{u}}^{kr}\}_{r=1}^R$ are scaled copies of the vectors in $\{\mathbf{u}^{kr}\}_{r=1}^R$.

At this point, we know that the elements in $\{\hat{\mathcal{X}}^{Kr} := \bigotimes_{k=1}^K\hat{\mathbf{u}}^{kr}\}_{r=1}^R$ are scaled copies of the elements in $\{\mathcal{X}^{Kr} = \bigotimes_{k=1}^K\mathbf{u}^{kr}\}_{r=1}^R$. It follows that

$$\mathcal{X} = \sum_{r=1}^{R}\mathcal{X}^{Kr} = \sum_{r=1}^{R}\lambda_r\hat{\mathcal{X}}^{Kr}.$$

for some constants $\{\lambda_r\}_{r=1}^R$. In other words, we have determined the rank-1 components of $\mathcal{X}$. It just remains to obtain their weights. This can be easily done using a single vector in $\mathcal{X}$. To see this, let $\mathbf{x} \in \mathbb{R}^{D_1}$ be the column in the first *corner* of $\mathcal{X}$. More precisely, $\mathbf{x}$ is the first column in $\mathbf{X}^2$ (which is actually also the first column in $\mathbf{X}^3,\mathbf{X}^4,\dots,\mathbf{X}^K$). See Figure 5 for some intuition.

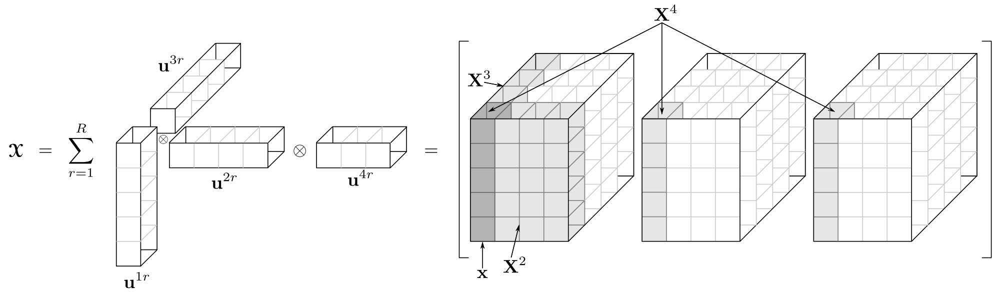


Fig. 5: Intuitively, $\mathbf{X}^k$ is the *face* of $\mathcal{X}$ in dimensions D_1 and D_k. $\mathbf{x} \in \mathbb{R}^{D_1}$ is the column in the first *corner* of $\mathcal{X}$.

Algorithm 1: Low-Rank CPD

```
1  Input: $K$-order, rank-$R$ tensor $\mathcal{X} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_K}$.
2  PART 1: Obtain $\hat{\mathbf{U}}^1$.
3      $\tilde{\mathbf{U}}^1$ = matrix with the first $R$ columns of $\mathbf{X}^{21}$
           (see Figure 4).
4      $\mathbf{X}^{22}_R$ = matrix with the first $R$ columns of $\mathbf{X}^{22}$.
5      $\tilde{\mathbf{\Theta}}^{22}_R$ = coefficients of $\mathbf{X}^{22}_R$ w.r.t. $\tilde{\mathbf{U}}^1$
           $= (\tilde{\mathbf{U}}^{1\mathsf{T}}\tilde{\mathbf{U}}^1)^{-1}\tilde{\mathbf{U}}^{1\mathsf{T}}\mathbf{X}^{22}_R$.
6      $\mathbf{\Gamma}$ = eigenvectors of $\tilde{\mathbf{\Theta}}^{22}_R$.
7      $\hat{\mathbf{U}}^1 = \tilde{\mathbf{U}}^1\mathbf{\Gamma}$.
8  PART 2: Obtain $\{\hat{\mathbf{U}}^k\}_{k=2}^K$.
9      $\mathbf{X}^k$ = face of $\mathcal{X}$ in dimensions $D_1$ and $D_k$
           (see Figure 5).
10     $\hat{\mathbf{U}}^k$ = coefficients of $\mathbf{X}^k$ w.r.t. $\hat{\mathbf{U}}^1$
           $= (\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1)^{-1}\hat{\mathbf{U}}^{1\mathsf{T}}\mathbf{X}^k$.
11 PART 3: Obtain scaling factors $\{\lambda_r\}_{r=1}^R$.
12     $\mathbf{x}$ = first column in $\mathcal{X}$ (see Figure 5).
13     $\hat{\boldsymbol{\theta}}$ = coefficients of $\mathbf{x}$ w.r.t. $\hat{\mathbf{U}}^1$
           $= (\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1)^{-1}\hat{\mathbf{U}}^{1\mathsf{T}}\mathbf{x}$.
14     $\lambda_r = \hat{\theta}_r / \prod_{k=2}^K \hat{\mathrm{u}}_1^{kr}$.
15     $\hat{\mathbf{u}}^{1r} = \lambda_r \hat{\mathbf{u}}^{1r}$.
16 Output: $\{\hat{\mathbf{u}}^{kr}\}_{k,r=1}^{K,R}$.
```

Let $\hat{\boldsymbol{\theta}} = [\hat{\theta}_1 \ \hat{\theta}_2 \ \cdots \ \hat{\theta}_R]^\mathsf{T}$ be the vector with the coefficients of $\mathbf{x}$ with respect to the basis $\hat{\mathbf{U}}^1$. Then

$$\mathbf{x} \ = \ \sum_{r=1}^R \hat{\theta}_r \hat{\mathbf{u}}^{1r}.$$

On the other hand, we want to find constants $\{\lambda_r\}_{r=1}^R$ such that

$$\mathbf{x} \ = \ \sum_{r=1}^R \lambda_r \left(\hat{\mathbf{u}}^{1r} \cdot \prod_{k=2}^K \hat{\mathrm{u}}_1^{kr} \right).$$

Putting the last two equations together, we have that $\lambda_r = \hat{\theta}_r / \prod_{k=2}^K \hat{\mathrm{u}}_1^{kr}$. We thus conclude that

$$\mathcal{X} \ = \ \sum_{r=1}^R \lambda_r \bigotimes_{k=1}^K \hat{\mathbf{u}}^{kr},$$

as desired. Notice that each scaling factor λ_r can be incorporated in one of the columns of $\hat{\mathbf{U}}^1$ (or any other $\hat{\mathbf{U}}^k$) to obtain a CPD of the form in (1), i.e., without scaling factors.

We have thus shown the following theorem, which states that the output of Algorithm 1 will yield the CPD of almost every low-rank tensor.

Theorem 1. *Let* **A1**-**A2** *hold. Let* $\{\hat{\mathbf{u}}^{kr}\}_{k,r=1}^{K,R}$ *be the output of Algorithm 1. Then with probability* 1,

$$\mathcal{X} \ = \ \sum_{r=1}^R \bigotimes_{k=1}^K \hat{\mathbf{u}}^{kr}.$$

III. Computational Complexity

In this section we analyze the computational complexity of Algorithm 1. In the first part, Algorithm 1 computes $\hat{\mathbf{U}}^1$. This requires to compute several matrix multiplications, one matrix inverse, and one eigenvalue decomposition (steps 5 – 7). Since $\tilde{\mathbf{U}}^1, \mathbf{X}^{22}_R \in \mathbb{R}^{D_1 \times R}$, using schoolbook matrix algebra, the required operations have the following complexities:

Operation	Description	Complexity
$\tilde{\mathbf{U}}^{1\mathsf{T}}\tilde{\mathbf{U}}^1$	$(R \times D_1)$ by $(D_1 \times R)$ product	$\mathcal{O}(R^2 D_1)$
$(\tilde{\mathbf{U}}^{1\mathsf{T}}\tilde{\mathbf{U}}^1)^{-1}$	$(R \times R)$ inverse	$\mathcal{O}(R^3)$
$\tilde{\mathbf{U}}^{1\mathsf{T}}\mathbf{X}^{22}_R$	$(R \times D_1)$ by $(D_1 \times R)$ product	$\mathcal{O}(R^2 D_1)$
Step 5	$(R \times R)$ by $(R \times R)$ product	$\mathcal{O}(R^3)$
Step 6	$(R \times R)$ eigenvalue decomposition	$\mathcal{O}(R^3)$
Step 7	$(D_1 \times R)$ by $(R \times R)$ product	$\mathcal{O}(R^2 D_1)$

Since $R \leq D_1$, the computational complexity of the first part of Algorithm 1 is upper bounded by $\mathcal{O}(R^2 D_1)$.

In the second part, Algorithm 1 computes $\{\hat{\mathbf{U}}^k\}_{k=2}^K$. This requires several matrix multiplications, and one matrix inverse. Since $\hat{\mathbf{U}}^1 \in \mathbb{R}^{D_1 \times R}$ and $\mathbf{X}^k \in \mathbb{R}^{D_1 \times D_k}$, the required operations have the following complexities:

Operation	Description	Complexity
$\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1$	$(R \times D_1)$ by $(D_1 \times R)$ product	$\mathcal{O}(R^2 D_1)$
$(\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1)^{-1}$	$(R \times R)$ inverse	$\mathcal{O}(R^3)$
$\hat{\mathbf{U}}^{1\mathsf{T}}\mathbf{X}^k$	$(R \times D_1)$ by $(D_1 \times D_k)$ product	$\mathcal{O}(R D_1 D_k)$
Step 10	$(R \times R)$ by $(R \times D_k)$ product	$\mathcal{O}(R^2 D_k)$

Since step 10 is repeated $K-1$ times, and $D_1 \geq D_2 \geq \cdots \geq D_K$, the computational complexity of the second part of Algorithm 1 is upper bounded by $\mathcal{O}(KRD_1^2)$.

In the third part, Algorithm 1 computes the scaling factors $\{\lambda_r\}_{r=1}^R$. This requires to compute $\hat{\boldsymbol{\theta}}$ and some negligible scalar operations. Computing $\hat{\boldsymbol{\theta}}$ requires to compute $\hat{\mathbf{U}}^{1\mathsf{T}}\mathbf{x}$, and multiply it by $(\hat{\mathbf{U}}^{1\mathsf{T}}\hat{\mathbf{U}}^1)^{-1}$, which was already computed in the second part. It follows that the third part of Algorithm 1 has a computational complexity of $\mathcal{O}(RD)$. Putting everything together, we obtain the following theorem.

Theorem 2. *The computational complexity of Algorithm 1 is* $\mathcal{O}(KRD_1^2)$.

IV. Handling Noise

In practice, measurements are hardly noiseless. So instead of (1), we can model $\mathcal{X}$ as

$$\mathcal{X} \ = \ \sum_{r=1}^R \bigotimes_{k=1}^K \mathbf{u}^{kr} + \mathcal{W}, \tag{4}$$

where $\{\mathbf{u}^{kr}\}_{k,r=1}^{K,R}$ are as before, and $\mathcal{W}$ is a noise tensor.

Fortunately, there is a straight forward way to adapt our ideas to this setting. Notice that our first step is to use R columns in $\mathcal{X}$ to obtain the basis $\tilde{\mathbf{U}}^1$ that spans the same subspace as $\mathbf{U}^1$ (see Section II and step 3). Instead, we can use $N \geq R$ columns in $\mathcal{X}$ to obtain an estimate of this subspace. Under reasonable assumptions on the noise (e.g., independence and finite fourth moment), the accuracy and computational complexity of this procedure will depend on

Algorithm 2: Low-Rank CPD, noisy variant

```
1  Input: Noisy tensor $\mathcal{X} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_K}$,
          rank-$R$, parameter $N \in \mathbb{N}$.
2  PART 1: Obtain $\hat{\mathbf{U}}^1$.
3      $\tilde{\mathbf{U}}^1$ = $R$ leading left-singular vectors of a matrix
           with $N \geq R$ columns from $\mathcal{X}$.
4      $\mathbf{X}_R^{21}, \mathbf{X}_R^{22}$ = matrices with the first $R$ columns
           of $\mathbf{X}^{21}$ and $\mathbf{X}^{21}$ (see Figure 4).
5      $\tilde{\mathbf{\Theta}}_R^{21}, \tilde{\mathbf{\Theta}}_R^{22}$ = coefficients of $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ w.r.t. $\tilde{\mathbf{U}}^1$.
6      $\mathbf{\Gamma}$ = eigenvectors of $\tilde{\mathbf{\Theta}}_R^{22}(\tilde{\mathbf{\Theta}}_R^{21})^{-1}$.
7      $\hat{\mathbf{U}}^1 = \tilde{\mathbf{U}}^1\mathbf{\Gamma}$.
8  PART 2: Same as in Algorithm 1.
9  PART 3: Same as in Algorithm 1.
10 Output: $\{\hat{\mathbf{u}}^{kr}\}_{k,r=1}^{K,R}$.
```

the number of columns that we use. The more columns, the higher precision, and the higher computational complexity. Let $\mathbf{X}$ be the matrix formed with the N columns of $\mathcal{X}$ that will be used to estimate the subspace spanned by $\mathbf{U}^1$. For example, if we decide to use all the columns in $\mathcal{X}$, then $\mathbf{X}$ is equivalent to the *unfolding* of $\mathcal{X}$ in the first dimension.

This time we can take $\tilde{\mathbf{U}}^1$ to be the matrix with the leading left-singular vectors of $\mathbf{X}$, as opposed to being the matrix formed with the first R columns of $\mathbf{X}^{2i}$. Consequently, the matrix $\tilde{\mathbf{\Theta}}_R^{2i}$ containing the coefficients of the first R columns in $\mathbf{X}^{2i}$ with respect to $\tilde{\mathbf{U}}$ will no longer be the identity matrix (as in the noiseless case), so instead of (3) we have that

$$\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}_R^{2j}(\tilde{\mathbf{\Theta}}_R^{2i})^{-1} \;=\; \mathbf{\Lambda}\mathbf{\Gamma}^{-1}.$$

Left-multiplying by $\mathbf{\Gamma}$, we obtain $\tilde{\mathbf{\Theta}}_R^{2j}(\tilde{\mathbf{\Theta}}_R^{2i})^{-1} = \mathbf{\Gamma}\mathbf{\Lambda}\mathbf{\Gamma}^{-1}$, and from this we can see that this time, $(\mathbf{\Gamma}, \mathbf{\Lambda})$ is the eigenvalue decomposition of $\tilde{\mathbf{\Theta}}_R^{2j}(\tilde{\mathbf{\Theta}}_R^{2i})^{-1}$. At this point, we can use $\tilde{\mathbf{U}}\mathbf{\Gamma}$ as an estimate of $\mathbf{U}^1$, and continue with the rest of the procedure exactly as before. This is summarized in Algorithm 2.

Notice that the main difference between Algorithms 1 and 2 lies in step 3. In Algorithm 1, step 3 simply requires to access entries in $\mathcal{X}$. In contrast, step 3 in Algorithm 2 requires to compute the leading R singular vectors of a $D_1 \times N$ matrix, which requires $\mathcal{O}(NRD_1)$ operations. Since $R \leq N, D_1$, by the same arguments as in Section III we conclude that the first part of Algorithm 2 has a computational complexity of $\mathcal{O}(NRD_1)$, as opposed to the $\mathcal{O}(R^2D_1)$ complexity of the first part of Algorithm 1. Since the rest of the procedure is the same for Algorithms 1 and 2, we have the following corollary.

Corollary 1. *The computational complexity of Algorithm 2 is* $\mathcal{O}(\max\{NRD_1, KRD_1^2\})$

Corollary 1 implies that noise will only affect the computational complexity of our approach if we decide to use more than KD_1 columns in step 3 of Algorithm 2 to estimate the subspace spanned by $\mathbf{U}^1$.

V. Experiments

In this section we present a series of experiments to support our theoretical findings.

Noiseless Setting

In our first set of experiments we will study the behavior of Algorithm 1 as a function of K, R and $D_1, D_2, \ldots, D_K$. To keep things simple, we will set $D_1 = D_2 = \cdots = D_K$. To generate K-order, rank-R tensors satisfying **A1**, we first generated vectors $\{\mathbf{u}^{kr}\}_{k,r=1}^{K,R}$ with $\mathcal{N}(0,1)$ i.i.d. entries, where $\mathbf{u}^{kr} \in \mathbb{R}^{D_1}$ for every $k \in [K]$ and $r \in [R]$. We then constructed $\mathcal{X}$ as in (1). In all our noiseless experiments, Algorithm 1 was able to perfectly recover the CPD of $\mathcal{X}$, as stated in Theorem 1, and so on this first set of experiments, we will only focus on the computation time.

In our first experiment we study the behavior of Algorithm 1 as a function of D_1, with $K = 4$, and $R = 10$ fixed. For each value of D_1, we generated 100 tensors as described before, ran Algorithm 1 to obtain its CPD, and recorded the elapsed time in each trial. The results, summarized in Figure 6 show that, consistent with Theorem 2, the computational complexity of Algorithm 1 grows quadratically in D_1.

In our second experiment we study the behavior of Algorithm 1 as a function of R. Since **A2** requires $R \leq D_2$, to allow a wider range of R we set $K = 5$ and $D = 50$. For each value of R, we generated 100 tensors as described before, ran Algorithm 1 to obtain its CPD, and recorded the elapsed time in each trial. The results, summarized in Figure 7, show that, consistent with Theorem 2, the computational complexity of Algorithm 1 grows linearly in R.

In our third experiment we will study the behavior of Algorithm 1 as a function of K. Notice that explicitly storing a tensor requires room for D_1^K values (exponential growth in K). If D_1 is large, as K grows one quickly runs out of memory. So to be able to do this experiment, we will choose D_k to be quite small. More precisely, we will set $D_1 = 3$ and $R = 2$ (as to satisfy **A2**). For each value of K, we generated

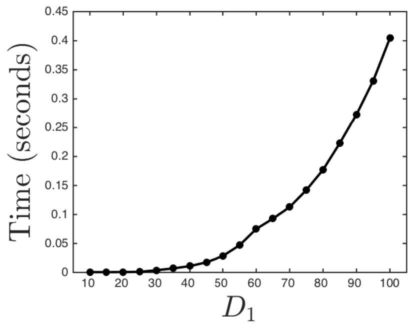


Fig. 6: Average time (over 100 trials) required by Algorithm 1 to compute the CPD of a K-order, rank-R tensor of dimensions $D_1 \times D_1 \times \cdots \times D_1$, as a function of D_1, with $K = 4$ and $R = 10$ fixed. Notice that explicitly storing such tensor requires room for D_1^K values (polynomial growth in D_1). In contrast, storing the CPD only requires room for KRD_1 values (linear growth in D_1). As shown by Theorem 2, Algorithm 1 only requires $\mathcal{O}(KRD_1^2)$ operations to compute the CPD (quadratic growth in D_1).

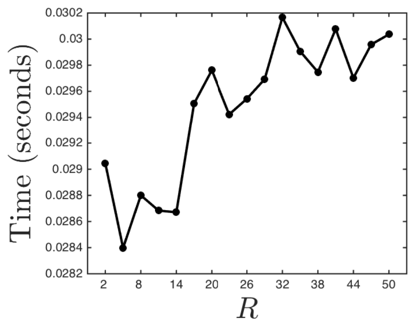


Fig. 7: Average time (over 100 trials) required by Algorithm 1 to compute the CPD of a K-order, rank-R tensor of dimensions $D_1 \times D_1 \times \cdots \times D_1$, as a function of R, with $K = 4$ and $D_1 = 50$ fixed. Theorem 2 shows that Algorithm 1 only requires $\mathcal{O}(KRD_1^2)$ operations (linear growth in R) to compute such CPD.

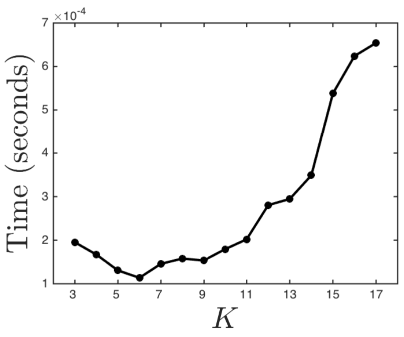


Fig. 8: Average time (over 100 trials) required by Algorithm 1 to compute the CPD of a K-order, rank-R tensor of dimensions $D_1 \times D_1 \times \cdots \times D_1$, as a function of K, with $D_1 = 3$ and $R = 2$ fixed. Notice that explicitly storing such tensor requires room for D_1^K values (exponential growth in K). In contrast, storing the CPD only requires room for KRD_1 values (linear growth in K). Theorem 2 shows that Algorithm 1 only requires $\mathcal{O}(KRD_1^2)$ operations (linear growth in K) to compute such CPD.

100 tensors as described before, ran Algorithm 1 to obtain its CPD, and recorded the elapsed time in each trial. The results are summarized in Figure 8. One of the main advantages of the CPD is that it only requires to store KRD_1 values (linear growth in K). Theorem 2 states that computing the CPD only requires $\mathcal{O}(KRD_1^2)$ operations (linear growth in K).

Noisy Setting

In our second set of experiments we will study the behavior of Algorithm 2 as a function of its dimensions, the noise level and the parameter N indicating the number of columns used to estimate the subspace spanned by $\mathbf{U}^1$. To this end, we first generated $\mathcal{X}$ as before, with $K = 4$ and $R = 10$ fixed, and $D_1 = D_2 = \cdots D_K$. We then contaminated it with $\mathcal{N}(0, \sigma^2)$ i.i.d. entries, to obtain observations according to (4).

First we study the behavior of Algorithm 2 as a function of the noise level and the parameter $N \in [R, D_1^{K-1}] = [10, 64,000]$, with $D_1 = 40$ fixed. For each (N, σ) pair we generated 100 noisy tensors, ran Algorithm 2, and recorded (i) the estimation error of the subspace spanned by $\mathbf{U}^1$, measured as the Frobenius norm of the difference between the projection operators (ii) the normalized error between the original uncontaminated tensor and the tensor obtained by the estimated CPD, and (iii) the elapsed time of the algorithm. The results are summarized in Figure 9.

As discussed in Section IV, there is a tradeoff between the accuracy and the computational complexity of Algorithm 2. This is regulated by the parameter N indicating the number of columns used to obtain $\tilde{\mathbf{U}}^1$ in step **3**. The larger N, the higher precision, and the higher computational complexity. In the rightmost image of Figure 9 we can see that the computational complexity grows with N. Conversely, in the leftmost image of Figure 9 we can see that the estimation error of the subspace spanned by $\mathbf{U}^1$ decreases with N (consistency). This follows as a simple consequence of the law of large numbers.

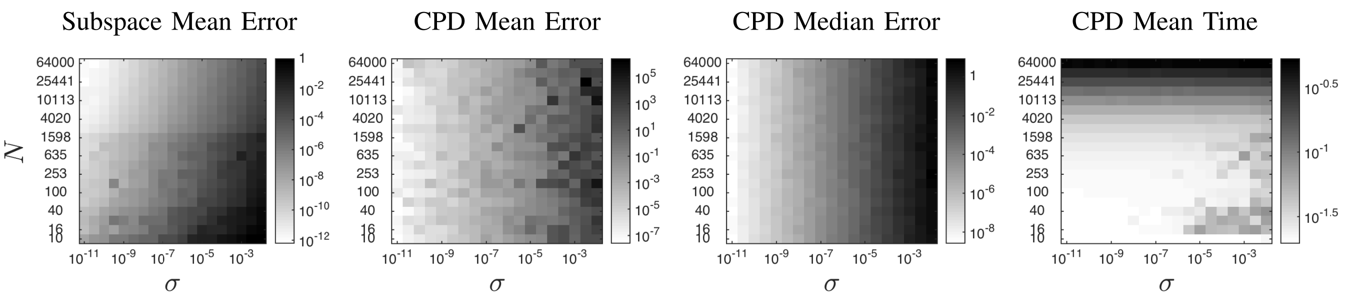


Fig. 9: Transition diagram of estimation errors (first three figures) and elapsed time (rightmost figure) of Algorithm 2 as a function of the noise level σ and the parameter N indicating the number of columns used to estimate the subspace spanned by $\mathbf{U}^1$. In these experiments we fixed $D_1 = 40$, $K = 5$ and $R = 10$. The color of each (N, σ) pixel indicates the average over 100 trials (the lighter the better). As discussed in Section IV, there is a tradeoff between the accuracy and the computational complexity of Algorithm 2, which is regulated by the parameter N. The larger N, the higher precision (see leftmost figure), and the higher computational complexity (see rightmost figure). The median error shows that most of the time, Algorithm 2 can efficiently and accurately (within the noise level) estimate the CPD of contaminated low-rank tensors. However, in some cases the entries in $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ are too noisy. This causes that our change of basis $\mathbf{\Gamma}$ is inaccurate, which leads to a poor estimation of the CPD (see mean error). We noticed that in some noisy cases, computing a subset of R leading singular values leads to numerical problems, which lead to the inconsistent pattern in the bottom-right corner of the rightmost figure.

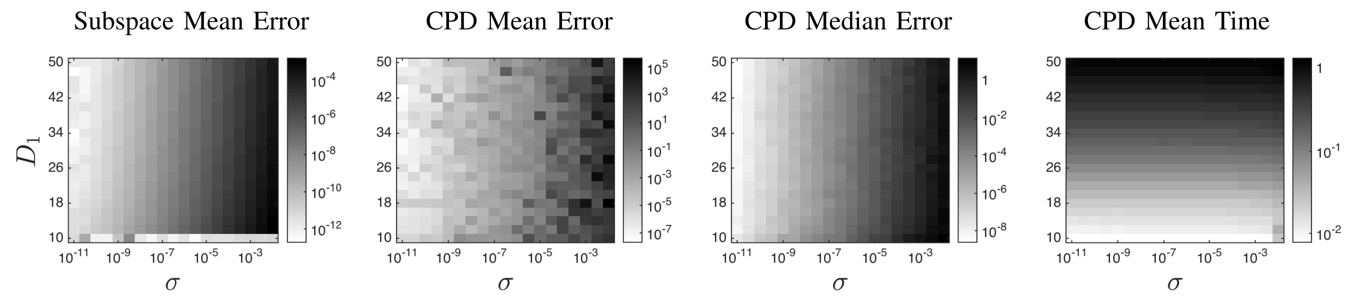


Fig. 10: Transition diagram of estimation errors (first three figures) and elapsed time (rightmost figure) of Algorithm 2 as a function of the dimension D_1 and the noise level σ, with $K = 5$, $R = 10$ and $N = D_1^{K-1}$ fixed. The color of each (D_1, σ) pixel indicates the average over 100 trials (the lighter the better). We point out that the subspace estimation of the case $D_1 = R = 10$ (bottom row of the leftmost figure) is mostly zero because in this case, the subspace spanned by $\mathbf{U}^1$ is the whole space, whence any $D_1 = R = 10$ linearly independent columns would span the same subspace. The leftmost figure shows that Algorithm 2 can consistently estimate this subspace (within the noise level), regardless of the dimension of the tensor. The median error shows that most of the time, Algorithm 2 can efficiently and accurately (within the noise level) estimate the CPD of contaminated low-rank tensors. However, as in the experiments of Figure 9, in some cases the entries in $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ are too noisy. This causes that our change of basis $\mathbf{\Gamma}$ is inaccurate, which leads to a poor estimation of the CPD (see mean error).

However, determining the subspace spanned by $\mathbf{U}^1$ is not enough for the CPD; we need to estimate *the right* basis of this subspace. Once we have an estimate of this subspace, given by $\operatorname{span}\{\tilde{\mathbf{U}}^1\}$, Algorithm 2 uses $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ to estimate *the right* change of basis $\mathbf{\Gamma}$. Unfortunately, if the entries in $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ are too noisy, our change of basis $\mathbf{\Gamma}$ can be inaccurate, which leads to a poor estimation of the CPD. In our experiments, we can see that most of the time Algorithm 2 can efficiently and accurately (within the noise level) estimate the CPD of contaminated low-rank tensors (see the median error). However, there are some cases where our estimate of the basis $\mathbf{U}^1$ is poor, which leads to an inaccurate CPD (see the mean error). Our future work will investigate better techniques to use more entries in $\mathfrak{X}$ to obtain consistent estimates of $\mathbf{\Gamma}$, so that the noise cancels out as we use more entries.

In our final experiment we study the behavior of Algorithm 2 as a function of the noise level and D_1. For each (D_1, σ) pair we generated 100 noisy tensors, ran Algorithm 2 using all the columns in $\mathfrak{X}$ in step **3** (i.e., $N = D_1^{K-1} = 64,000$), and recorded the same three values as in our previous experiment. The results, summarized in Figure 10 show that most of the time, Algorithm 2 can efficiently and accurately (within the noise level) estimate the CPD of contaminated low-rank tensors (see median error). However, as in our previous experiment, in some cases the entries in $\mathbf{X}_R^{21}$ and $\mathbf{X}_R^{22}$ are too noisy. This causes that our change of basis $\mathbf{\Gamma}$ is inaccurate, which leads to a poor estimation of the CPD (see mean error).

VI. Conclusions

In this paper we present a simple and efficient method to compute the CPD of generic low-rank tensors using elementary linear algebra. The key insight is that all the columns in a low-rank tensor lie in a low-dimensional subspace, and the coefficients of the columns in each *slice* with respect to *the right* basis are scaled copies of one an other. The basis, together with the coefficients of a few carefully selected columns determine the CPD. The computational complexity of our method scales linearly in the order and the rank of the tensor, and at most quadratically in its largest dimension, which is confirmed by our experiments. A straightforward extension to noisy settings produces reasonable results on median. However, in some cases, the few entries used for our estimates can be very noisy, which results in a poor CPD. Our future work will investigate new techniques to obtain more consistent estimators.

References

[1] F. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*, Journal of Mathematics and Physics, 1927.
[2] J. Carroll and J. Chang, *Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition*, Psychometrika, 1970.
[3] R. Harshman, *Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis*, UCLA Working Papers in Phonetics, 1970.
[4] A. Smilde, R. Bro and P. Geladi, *Multi-way analysis: Applications in the chemical sciences*, Wiley, Chichester, UK, 2004.
[5] P. Kroonenberg, *Applied multiway data analysis*, Wiley, Hoboken, NJ, 2008.
[6] T. Kolda and B. Bader, *Tensor decompositions and applications*, SIAM Review, 2009.
[7] P. Comon, X. Luciani and A. de Almeida, *Tensor decompositions, alternating least squares and other tales*, Journal of Chemometrics, 2009.
[8] L. De Lathauwer, *A short introduction to tensor-based methods for factor analysis and blind source separation*, International Symposium on Image and Signal Processing and Analysis, 2011.
[9] A. Cichocki, D. Mandic, A-H. Phan, C. Caiafa, G. Zhou, Q. Zhao, and L. De Lathauwer, *Tensor decompositions for signal processing applications from two-way to multiway component analysis*, IEEE Signal Processing Magazine, 2015.
[10] R. Sands and F. Young, *Component models for three-way data: An alternating least squares algorithm with optimal scaling features*, Psychometrika, 1980.
[11] L. De Lathauwer, B. De. Moor and J. Vandewalle, *Computation of the canonical decomposition by means of a simultaneous generalized Schur Decomposition*, SIAM Journal on Matrix Analysis and Applications, 2004.
[12] L. De Lathauwer, *A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization*, SIAM Journal on Matrix Analysis and Applications, 2006.
[13] N. Colombo and N. Vlassis, *Tensor decomposition via joint matrix Schur decomposition*, International Conference on Machine Learning, 2016.

[14] M. Rajih, P. Comon, and R. Harshman, *Enhanced line search: A novel method to accelerate PARAFAC*, SIAM Journal of Matrix Analysis and Applications, 2008.

[15] L. De Lathauwer, J. Castaing and J. Cardoso, *Fourth-order cumulant-based identification of underdetermined mixtures*, IEEE Transactions on Signal Processing, 2007.

[16] I. Domanov and L. De Lathauwer, *Canonical polyadic decomposition of third-order tensors: reduction to generalized eigenvalue decomposition*, SIAM Journal on Matrix Analysis and Applications, 2014.

[17] I. Domanov and L. De Lathauwer, *Canonical polyadic decomposition of third-order tensors: relaxed uniqueness conditions and algebraic algorithm*, available at `http://arxiv.org/abs/1501.07251`, 2015.

[18] G. Zhou, Z. He, Y. Zhang, Q. Zhao and A. Cichocki, *Canonical polyadic decomposition: from 3-way to N-way*, IEEE International Conference on Computational Intelligence and Security, 2012.

[19] L. Grasedyck, D. Kressner and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitteilungen, 2013.

[20] P. Comon, *Tensors: a brief introduction*, IEEE Signal Processing Magazine, 2014.

[21] R. Harshman, *Determination and proof of minimum uniqueness conditions for PARAFAC*, UCLA Working Papers in Phonetics, 1972.

[22] E. Sanchez and B. Kowalski, *Tensorial resolution: a direct trilinear decomposition*, Journal of Chemometrics, 1990.

[23] S. Leurgans, R. Ross and R. Abel, *A decomposition for three-way arrays*, SIAM Journal of Matrix Analysis and Applications, 1993.

[24] N. Faber, L. Buydens and G. Kateman, *Generalized rank annihilation method: I. derivation of eigenvalue problems*, Journal of Chemometrics, 1994.

[25] J. Berge and J. Tendeiro, *The link between sufficient conditions by Harshman and by Kruskal for uniqueness in Candecomp/Parafac*, Journal of Chemometrics, 2009.