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 another. 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.

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.

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
<th>Regular</th>
<th>Bold</th>
<th>Lower</th>
<th>Capital</th>
<th>Roman</th>
<th>Script</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>$K, R$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Index</td>
<td>$i, j, k, r$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Vector</td>
<td>$u$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Matrix</td>
<td>$U$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Tensor</td>
<td>$X$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Entry</td>
<td>$u_i$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

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

II. MODEL AND MAIN RESULTS

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

$$X = \sum_{r=1}^{R} K \otimes \mathbf{u}^{kr}$$

(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 $X$ (up to scaling factors). This is known as the canonical polyadic decomposition (CPD) of $X$. 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 $R \leq D_2$.

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 $X$ is a tensor (as opposed to a vector, or a matrix), and that at least two of its dimensions are as large as $R$. 

A1 essentially requires that $X$ is a generic rank-$R$ tensor.
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 $U^k := [u^{k1} \ u^{k2} \ ... \ u^{kR}] \in \mathbb{R}^{D_k \times R}$. The key insight is that under A1-A2, all the columns in $X$ lie in the $R$-dimensional subspace spanned by $U^1$, and that the coefficients of the slices of $X$ with respect to $U^1$ are copies of one another, scaled by the values in $U^2, U^3, \ldots, U^K$. So one only needs a few of these coefficients to determine $U^2, U^3, \ldots, U^K$ (up to scaling factors). With this in mind, we first find a basis $\tilde{U}^1$ containing scaled copies of the columns in $U^1$. Next, we determine the coefficients of a few carefully selected columns in $X$ with respect to this basis. These coefficients determine scaled copies of $U^2, U^3, \ldots, U^K$. Finally, we determine scaling factors to obtain the desired CPD.

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

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

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

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

$$X^{2i} := \sum_{r=1}^R (u^{1r} \otimes u^{2r}) u_i^{3r} (u_i^{4r} \ldots u_i^{Kr}) \in U^1(U^2D_i)^T,$$

where $D_i \in \mathbb{R}^{R \times R}$ is the diagonal matrix with $\{u_i^{3r} \Pi_{k \neq i} u_k^{kr}\}_{j \neq i}$ as diagonal entries. Intuitively, $X^{2i}$ is the matrix of $X$ in dimensions $D_1$ and $D_2$, located at position $\{i, 1, \ldots, 1\}$ in dimensions $D_3, D_4, \ldots, D_K$. See Figure 4 for some intuition.

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

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

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

Fig. 2: Each $K$-order, rank-$R$ tensor $X$ is determined by a set of vectors $(u^{kr})_{k \neq 1}$. In the left, these vectors are in general position, that is, each $u^{kr}$ 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 $u^{kr}$'s are as in the right, where some $u^{kr}$'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.

Fig. 3: $X^{Kr}$ is the $k$-order, rank-1 tensor given by $u^{1r} \otimes u^{2r} \otimes \ldots \otimes u^{kr}$. $X^K$ is the $k$-order, rank-$R$ tensor given by $\sum_{r=1}^R X^{Kr}$. Notice that $X^K = X$. 
Recall that for any full-rank matrix $\Gamma \in \mathbb{R}^{R \times R}$, $\tilde{U}^1 \Gamma$ is also a basis of the subspace spanned by $\tilde{U}^1$. The coefficients of $X^{2i}$ with respect to the basis $\tilde{U}^1 \Gamma$ are given by $\Gamma^{-1} \tilde{\Theta}^{2i}$. Since $\tilde{U}^1$ spans the same subspace as $U^1$, and the coefficients of $X^{2j}$ with respect to $U^1$ are scaled copies of the coefficients of $X^{2i}$ with respect to $U^1$, we know that there exists a change of basis $\Gamma$ such that the rows of $\Gamma^{-1} \tilde{\Theta}^{2j}$ are scaled copies of the rows in $\Gamma^{-1} \tilde{\Theta}^{2i}$. More precisely, there exists a full-rank matrix $\Gamma \in \mathbb{R}^{R \times R}$ and a diagonal matrix $\Lambda \in \mathbb{R}^{R \times R}$ such that

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

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

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

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

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

$$X^k := \sum_{r=1}^{R} (u^{kr}_1 \otimes u^{kr}_2) (u^{kr}_1 \ldots u^{kr}_{(k-1)r} u^{kr}_{(k+1)r} \ldots u^{kr}_{kr}),$$

where $D^k \in \mathbb{R}^{R \times R}$ is the diagonal matrix with $\{u^{kr}_1 \ldots u^{kr}_{(k-1)r} u^{kr}_{(k+1)r} \ldots u^{kr}_{kr}\}_{r=1}^{R}$ as diagonal entries. Intuitively, $X^k$ is the face of $X$ in dimensions $D_1$ and $D_k$.

Or more precisely, $X^k$ is the matrix of $X$ in dimensions $D_1$ and $D_k$, located at position $\{1, 1, \ldots, 1\}$ in dimensions $D_2, D_3, \ldots, D_K$. See Figure 5 for some intuition. In particular, $X^2$ is equal to $X^{21}$, as defined before.

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

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

$$X^k = U^1 D^k T U^{kT} = U^1 (\Lambda \Pi) (\Lambda \Pi)^{-1} D^k T U^{kT} = (U^1 \Lambda \Pi) (\Lambda \Pi)^{-1} D^k T U^{kT} = \hat{U}^1 (U^k D^k \Lambda^{-1} \Pi)^T.$$  

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

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

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

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

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

Output: $\mathbf{X}$ = $\sum_{r=1}^R \lambda_r \prod_{k=1}^K \hat{u}_1^{kr}$.

III. Computational Complexity

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

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{U}^1 \mathbf{U}^f$</td>
<td>$(R \times D_1) \times (D_1 \times R)$ product</td>
<td>$\mathcal{O}(R^2 D_1)$</td>
</tr>
<tr>
<td>$(\hat{U}^1 \mathbf{U}^f)^{-1}$</td>
<td>$(R \times R)$ inverse</td>
<td>$\mathcal{O}(R^3)$</td>
</tr>
<tr>
<td>$\hat{U}^1 \mathbf{X}^{22}$</td>
<td>$(R \times D_1) \times (D_1 \times R)$ product</td>
<td>$\mathcal{O}(R^2 D_1)$</td>
</tr>
<tr>
<td>Step 5</td>
<td>$(R \times R)$ by $(R \times R)$ product</td>
<td>$\mathcal{O}(R^3)$</td>
</tr>
<tr>
<td>Step 6</td>
<td>$(R \times R)$ eigenvalue decomposition</td>
<td>$\mathcal{O}(R^3)$</td>
</tr>
<tr>
<td>Step 7</td>
<td>$(D_1 \times R)$ by $(R \times R)$ product</td>
<td>$\mathcal{O}(R^2 D_1)$</td>
</tr>
</tbody>
</table>

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 $\{\tilde{\Theta}^{22}_k\}_{k=2}^K$. This requires several matrix multiplications, and one matrix inverse. Since $\hat{U}^1 \in \mathbb{R}^{D_1 \times R}$ and $\mathbf{X}^k \in \mathbb{R}^{D_1 \times D_k}$, the required operations have the following complexities:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{U}^1 \mathbf{U}^f$</td>
<td>$(R \times D_1) \times (D_1 \times R)$ product</td>
<td>$\mathcal{O}(R^2 D_1)$</td>
</tr>
<tr>
<td>$(\hat{U}^1 \mathbf{U}^f)^{-1}$</td>
<td>$(R \times R)$ inverse</td>
<td>$\mathcal{O}(R^3)$</td>
</tr>
<tr>
<td>$\hat{U}^1 \mathbf{X}^{22}$</td>
<td>$(R \times D_1) \times (D_1 \times R)$ product</td>
<td>$\mathcal{O}(R^2 D_1)$</td>
</tr>
<tr>
<td>Step 10</td>
<td>$(R \times R)$ by $(R \times D_k)$ product</td>
<td>$\mathcal{O}(R^2 D_k)$</td>
</tr>
</tbody>
</table>

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 $\mathcal{O}(KD_1^2)$.

In the third part, Algorithm 1 computes the scaling factors $\{\lambda_r\}_{r=1}^R$. This requires to compute $\hat{\theta}$ and some negligible scalar operations. Computing $\hat{\theta}$ requires to compute $\hat{U}^1 \mathbf{x}$, and multiply it by $(\hat{U}^1 \mathbf{U}^f)^{-1}$, 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 $\mathcal{O}(KD_1^2)$.

IV. Handling Noise

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

$$\mathbf{X} = \sum_{r=1}^R \lambda_r \prod_{k=1}^K \hat{u}_1^{kr} + \mathbf{W},$$

where $\{\hat{u}_1^{kr}\}_{k,r=1}^K$ are as before, and $\mathbf{W}$ is a noise tensor.

Fortunately, there is a straightforward way to adapt our ideas to this setting. Notice that our first step is to use $R$ columns in $\mathbf{X}$ to obtain the basis $\hat{U}^1$ that spans the same subspace as $\mathbf{U}^1$ (see Section II and step 3). Instead, we can use $N \geq R$ columns in $\mathbf{X}$ 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 $\mathbf{X} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_K}$, rank-$R$, parameter $N \in \mathbb{N}$.

2. **PART 1:** Obtain $\hat{\mathbf{U}}^1$.

3. $\hat{\mathbf{U}}^1 = R$ leading left-singular vectors of a matrix with $N \geq R$ columns from $\mathbf{X}$.

4. $\mathbf{X}_R^{21} \cdot \mathbf{X}_R^{22} = \text{matrices with the first } R \text{ columns of } \mathbf{X}^{21} \text{ and } \mathbf{X}^{21}$ (see Figure 4).

5. $\tilde{\Theta}_R^{21} \cdot \tilde{\Theta}_R^{22} = \text{coefficients of } \mathbf{X}_R^{21} \text{ and } \mathbf{X}_R^{22}$ w.r.t. $\hat{\mathbf{U}}^1$.

6. $\Gamma = \text{eigenvectors of } \tilde{\Theta}_R^{22} \cdot \tilde{\Theta}_R^{21}^{-1}$.

7. $\check{\mathbf{U}}^1 = \hat{\mathbf{U}}^1 \Gamma$.

8. **PART 2:** Same as in Algorithm 1.

9. **PART 3:** Same as in Algorithm 1.

10. **Output:** $\{\hat{\mathbf{u}}^{kr}\}_{k,r=1}^K$.

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 $\mathbf{X}$ that will be used to estimate the subspace spanned by $\mathbf{U}^1$. For example, if we decide to use all the columns in $\mathbf{X}$, then $\mathbf{X}$ is equivalent to the unfolding of $\mathbf{X}$ in the first dimension.

This time we can take $\check{\mathbf{U}}^1$ 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 $\mathbf{X}^{21}$. Consequently, the matrix $\tilde{\Theta}_R^{21} \cdot \tilde{\Theta}_R^{22}$ containing the coefficients of the first $R$ columns in $\mathbf{X}^{22}$ with respect to $\check{\mathbf{U}}$ will no longer be the identity matrix (as in the noiseless case), so instead of (3) we have that

$$\Gamma^{-1} \tilde{\Theta}_R^{22} \cdot \tilde{\Theta}_R^{21}^{-1} = \Lambda \Gamma^{-1}.$$  

Left-multiplying by $\Gamma$, we obtain $\tilde{\Theta}_R^{22} \cdot \tilde{\Theta}_R^{21}^{-1} = \Gamma \Lambda \Gamma^{-1}$, and from this we can see that this time, $(\Gamma, \Lambda)$ is the eigenvalue decomposition of $\tilde{\Theta}_R^{22} \cdot \tilde{\Theta}_R^{21}^{-1}$. At this point, we can use $\check{\mathbf{U}} \Gamma$ as an estimate of $\hat{\mathbf{U}}^1$, 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 $\mathbf{X}$. In contrast, step 3 in Algorithm 2 requires to compute the leading $R$ singular vectors of a $D_1 \times N$ matrix, which requires $O(NRD_1)$ 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 $O(NRD_1)$, as opposed to the $O(R^2D_1)$ 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 $O(\max\{NRD_1, KR^2D_1^2\})$.

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 $\mathbf{A}_1$, we first generated vectors $\{\mathbf{u}_r^{kr}\}_{k,r=1}^K$ with $N(0, 1)$ i.i.d. entries, where $\mathbf{u}_r^{kr} \in \mathbb{R}^{D_1}$ for every $k \in [K]$ and $r \in [R]$. We then constructed $\mathbf{X}$ as in (1). In all our noiseless experiments, Algorithm 1 was able to perfectly recover the CPD of $\mathbf{X}$, 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 $\mathbf{A}_2$ 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 $D_1$ to be quite small. More precisely, we will set $D_1 = 3$ and $R = 2$ (as to satisfy $\mathbf{A}_2$). 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 $KR^2D_1^2$ values (linear growth in $D_1$). As shown by Theorem 2, Algorithm 1 only requires $O(KRD_1^2)$ operations to compute the CPD (quadratic growth in $D_1$).
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 $O(KRD_1^2)$ 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 $U^1$. To this end, we first generated $X$ as before, with $K = 4$ and $R = 10$ fixed, and $D_1 = D_2 = \cdots = D_K$. We then contaminated it with $N(0, \sigma^2)$ 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 $N \in \{5, 10, 25, 50, 100, 250, 500, 1000\}$, with $D_1 = 40$ fixed. For each $(N, \sigma)$ pair we generated 100 noisy tensors, ran Algorithm 2, and recorded (i) the estimation error of the subspace spanned by $U^1$, 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 $U^1$ 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 $U^1$ decreases with $N$ (consistency). This follows as a simple consequence of the law of large numbers.
Subspace Mean Error as a function of the noise level and $D$, we obtain consistent estimates of $X$ all the columns in the pair we generated $X$ in inaccurate CPD (see the mean error). Our future work will investigate new techniques to obtain more consistent estimators.

However, determining the subspace spanned by $U^1$ 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 $\text{span}\{\tilde{U}^1\}$, Algorithm 2 uses $X_{11}^{21}$ and $X_{21}^{21}$ to estimate the right change of basis $\Gamma$. Unfortunately, if the entries in $X_{11}^{21}$ and $X_{21}^{21}$ are too noisy, our change of basis $\Gamma$ 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. However, in some cases the entries in $X_{11}^{21}$ and $X_{21}^{21}$ are too noisy. This causes that our change of basis $\Gamma$ 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