Structured Alternating Minimization For Union of Nested Low-Rank Subspaces Data Completion

Morteza Ashraphijuo, Xiaodong Wang, *Fellow, IEEE*

***Abstract*—In this paper, we consider a particular data structure consisting of a union of several nested low-rank subspaces with missing data entries. Given the rank of each subspace, we treat the data completion problem, i.e., to estimate the missing entries. Starting from the case of two-dimensional data, i.e., matrices, we show that the union of nested subspaces data structure leads to a structured decomposition $\mathrm{U} = \mathrm{XY}$ where the factor Y has blocks of zeros that are determined by the rank values. Moreover, for high-dimensional data, i.e., tensors, we show that a similar structured CP decomposition also exists, $\mathcal{U} = \sum_{l=1}^{r} \mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \ldots \otimes \mathbf{a}_d^l$, where $\mathbf{A}_d = [\mathbf{a}_d^1 \ldots \mathbf{a}_d^r]$ contains blocks of zeros determined by the rank values. Based on such structured decompositions, we develop efficient alternating minimization algorithms for both matrix and tensor completions, by enforcing the above structures in each iteration including the initialization. Compared with naive approaches where either the additional rank constraints are ignored, or data completion is performed part by part, the proposed structured alternating minimization approaches exhibit faster convergence and higher recovery accuracy.**

***Index Terms*—Low-rank, matrix completion, tensor completion, union of nested subspaces, structured decomposition, alternating minimization.**

I. Introduction

GIVEN the ubiquitousness of multi-perspective, multi-dimensional big data in our day-to-day lives, a common feature shared by such datasets is the inherent sparsity or low-rank property. On the other hand, missing and faulty data are the norm rather than the exception. Hence a fundamental task in many big data applications is data completion, i.e., to recover the missing data points by exploiting the underlying sparsity structure. In particular, the low-rank matrix completion problem [1] is a classical problem that finds applications in various areas including compressed sensing [2–4], image inpainting [5], network coding [6], image processing [7, 8], data mining [9], etc. The low-rank tensor completion problem has received more attention in the past decade and plays a vital role in multilinear data analysis [10, 11], 3D image reconstruction [12], state estimation [13], color image inpainting [14], video inpainting [15], hyperspectral data recovery [16], higher-order web link analysis [17], etc.

In general, low-rank data completion techniques can be classified into convex and non-convex approaches, and a recent survey can be found in [18]. Specifically, convex approaches to matrix completion are typically based on nuclear norm minimization with theoretical optimality [19, 20]. Moreover, non-convex approaches such as alternating minimization are much faster than convex methods and empirically observed to always converge to the optimum, which has also been shown theoretically [21, 22]. Similarly, for the low-rank tensor completion, various nuclear norm minimization methods with theoretical performance guarantees have been introduced [23–

This work was supported in part by the U.S. National Science Foundation under Grant CCF-1814803 and in part by the U.S. Office of Naval Research under Grant N000141712827.

The authors are with the Department of Electrical Engineering, Columbia University, NY, email: {ashraphijuo,wangx}@ee.columbia.edu.

26], as well as the non-convex approaches such as alternating minimization [27–29], that make use of various tensor decompositions and are much faster than convex methods. For many non-convex approaches knowing the exact or estimated rank of sampled matrices and tensors is a requirement and this problem is studied in [30].

Related to data completion is the problem of data clustering in a union of subspaces with missing data. For example, let $\mathcal{S}_k$ be a subspace of $\mathbb{R}^N$ with rank r_k, $k = 1, \ldots, K$. Given a data matrix $\mathbf{U} \in \mathbb{R}^{N\times T}$ possibly with missing entries, the problem is to assign each column $\mathbf{u}_t$ of $\mathbf{U}$ to a particular subspace $\mathcal{S}_k$. For example, in face recognition, the K subspaces represent K persons and each vector $\mathbf{u}_t$ corresponds to the photo of a person. The clustering problem is then to assign each photo to one of the K persons [31]. Other applications of clustering a union of low-rank data structures include motion recognition [32], texture analysis [33], MIMO channel estimation [34], image analysis [35], etc. Classical approaches to subspace clustering include maximum likelihood methods [36, 37], algebraic algorithms [38–40] and their iterative implementations [41], and spectral clustering of high-dimensional data based on low-rank representation [42]. Moreover, most of these techniques can be extended to handle missing data. For example, nuclear norm minimization is employed in dictionary learning for spectral clustering in [43]; and in algebraic methods, subspaces where the sampled columns belong to are identified by analyzing a set of homogeneous polynomials [44].

In the union of subspace data structure mentioned above, the subspaces $\mathcal{S}_1, \ldots, \mathcal{S}_K$ are unrelated to each other. In this paper, we consider a union of nested low-rank subspaces, i.e., we assume that the subspaces are related according to $\mathcal{S}_1 \subset \mathcal{S}_2 \subset \ldots \subset \mathcal{S}_K$ which reflects the hierarchical data structure. For example, $\mathcal{S}_1$ can correspond to pictures of German Shepherd dogs, $\mathcal{S}_2$ to pictures of dogs, and $\mathcal{S}_3$ to pictures of animals. Or consider the scenario where $\mathcal{S}_1$ can correspond to the news about Apple stock, $\mathcal{S}_2$ to news about all technology stocks, and $\mathcal{S}_3$ to the news about American stock market. In such examples, the data of the first set is a subset of the second set and therefor the spanned space by the basis of the first set is a subset of the spanned space by the basis of the second set. Note that in practice, the assumption that we can find an exact low rank basis for the mentioned datasets may not hold but we can find low rank approximations for them. Theoretical aspects of clustering and completion of such union of nested subspaces data are studied in [45–47]. In particular, the fundamental conditions on the sampling patterns for correctly clustering are characterized for matrices and tensors in [45] and [46], respectively. Moreover, conditions on the sampling patterns for unique completability of the correctly clustered data are given for matrices and tensors in [47] and [46], respectively. However, to date there is no algorithmic study on union of nested subspaces data.

In this paper, we develop efficient alternating minimization based completion algorithms for union of nested subspaces two-dimensional (matrix) and higher-dimensional (tensor) data. For both matrix and tensor cases, first, we show that the union of nested subspaces structure and the corresponding rank constraints lead to a structured decomposition where certain factor has blocks of zeros. Then we develop an alternating minimization algorithm that alternatively updates each factor in the structured decomposition. Since initialization plays an important role in non-convex optimization, we also propose two structured initialization methods, one is based on random initialization and the other is based on solving several smaller least-squares problems. Extensive simulation results are provided to compare our proposed structured approaches to several naive methods that are also based on alternating minimization. Extensive simulation results show that the proposed structured approaches offer both faster convergence speed and higher data recovery accuracy, for both matrix and tensor data, with or without noise.

The remainder of the paper is organized as follows. In Section II, we formulate the problem of union of nested subspaces data completion for the matrix case, outline three naive methods for solving it based on alternating minimization, and then propose our structured alternating minimization algorithm. In Section III, we consider the same problem for the tensor case and develop the corresponding structured alternating minimization algorithm. Simulation results are presented in Section IV. And finally conclusions are drawn in Section V.

II. Completion of Union of Nested Low-Rank Matrices

A. Problem Statement

Assume that $K \geq 2$, $n_1 < n_2 < \cdots < n_K$ and m are given integers. Let $\mathbf{U} \in \mathbb{R}^{m \times n_K}$ be a sampled matrix and denote the matrix consisting of the first n_k columns of $\mathbf{U}$ by $\mathbf{U}_k \in \mathbb{R}^{m \times n_k}$ and also define $\mathbf{M}_k \in \mathbb{R}^{m \times (n_k - n_{k-1})}$ as the matrix consisting of the $(n_k - n_{k-1})$ columns of $\mathbf{U}_k$ that does not belong to $\mathbf{U}_{k-1}$, $k = 1, \ldots, K$. This is shown in Fig. 1 and note that $\mathbf{U} = \mathbf{U}_K$. Moreover, assume that rank$(\mathbf{U}_k) = r_k$, $k = 1, \ldots, K$. Hence, we have $r_1 \leq r_2 \leq \ldots \leq r_K$.

Let Ω denote the set of indices corresponding to the sampled entries, i.e., $\Omega = \{(i,j) : \mathbf{U}(i,j) \text{ is sampled}\}$. Moreover, define $\mathbf{U}_\Omega$ as the matrix obtained from sampling $\mathbf{U}$ according to Ω, i.e.,

$$\mathbf{U}_\Omega(i,j) = \begin{cases} \mathbf{U}(i,j) & \text{if } (i,j) \in \Omega, \\ 0 & \text{if } (i,j) \notin \Omega. \end{cases} \tag{1}$$

We are interested in retrieving the missing entries of $\mathbf{U}$ using an efficient alternating minimization-based method. The challenge is to take advantage of all K rank constraints simultaneously.

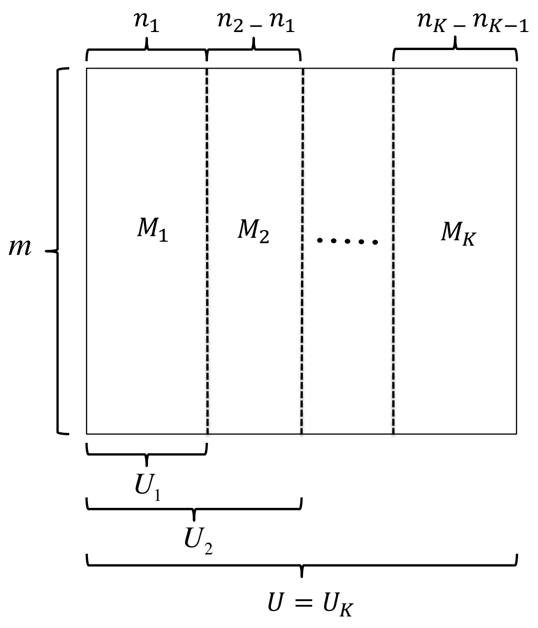


Fig. 1: The union of nested subspaces data structure, rank$(\mathbf{U}_k) = r_k$.

B. Naive Approaches

1) Naive Initialization Methods: Our goal is to find $\mathbf{X} \in \mathbb{R}^{m \times r_K}$ and $\mathbf{Y} \in \mathbb{R}^{r_K \times n}$ such that $\mathbf{U} = \mathbf{XY}$. We consider two simple methods for setting the initial values of $\mathbf{X}$ and $\mathbf{Y}$ – SVD-based initialization and random initialization. In the SVD-based method, we first compute the singular value decomposition (SVD) of $\mathbf{U}_\Omega$ and pick the r_K largest eigenvalues and their corresponding eigenvectors to construct the initial matrices $\mathbf{X}_0$ and $\mathbf{Y}_0$. In particular, if $\mathbf{U}_\Omega = \mathbf{U}_0 \mathbf{S}_0 \mathbf{V}_0^\top$, where the number of nonzero diagonal entries of $\mathbf{S}_0$ can be more than r_K. Then, we define a decomposition corresponding to the r_K largest singular values, i.e., $\mathbf{U}_0(:, 1 : r_K)\mathbf{S}_0(1 : r_K, 1 : r_K)\mathbf{V}_0(:, 1 : r_K)^\top = \mathbf{X}_0\mathbf{Y}_0$, where $\mathbf{X}_0 = \mathbf{U}_0(:, 1 : r_K)\mathbf{S}_0(1 : r_K, 1 : r_K) \in \mathbb{R}^{m \times r_K}$ and $\mathbf{Y}_0 = \mathbf{V}_0(:, 1 : r_K)^\top \in \mathbb{R}^{r_K \times n_K}$.

On the other hand, for random initialization, we simply set $\mathbf{X}_0$ and $\mathbf{Y}_0$ as matrices that contain i.i.d. $\mathcal{N}(0, 1)$ samples. Note that the initial matrix satisfies only one rank constraint r_K for both methods.

2) Naive Alternating Minimization Methods:

(i) Naive approach 1: In this approach we simply discard the rank constraints $r_1, \ldots, r_{K-1}$ and complete $\mathbf{U}$ using only the constraint rank($\mathbf{U}$) $= r_K$. Starting from the above initial $\mathbf{X}_0$ and $\mathbf{Y}_0$, in each iteration, we alternatively optimize $\mathbf{X}_i$ and $\mathbf{Y}_i$ until convergence.

In particular, at the i-th iteration, given $\mathbf{X}_{i-1}$ and $\mathbf{Y}_{i-1}$, we first update $\mathbf{X}_i$ by solving the following regularized least-squares problem

$$\text{minimize}_{\mathbf{X}_i \in \mathbb{R}^{m \times r_K}} \quad \|\mathbf{U}_\Omega - (\mathbf{X}_i \mathbf{Y}_{i-1})_\Omega\|_\mathcal{F} + \lambda \|\mathbf{X}_i\|_\mathcal{F} \ , \tag{2}$$

and then update $\mathbf{Y}_i$ by solving

$$\text{minimize}_{\mathbf{Y}_i \in \mathbb{R}^{r_K \times n_K}} \quad \|\mathbf{U}_\Omega - (\mathbf{X}_i \mathbf{Y}_i)_\Omega\|_\mathcal{F} + \lambda \|\mathbf{Y}_i\|_\mathcal{F} \tag{3}$$

where $\|\cdot\|_\mathcal{F}$ denotes the Frobenius norm and λ is a small constant. The purpose of the regularization term is to avoid singularity in solving the least-squares problems. The iteration continues until it reaches convergence or until the algorithm diverges. The solutions to (2) and (3) can be obtained row by row and column by column, respectively. In particular, denote $\mathbf{E}$ as an all-one $m \times n_K$ matrix, then (2) can be solved as

$$\begin{aligned} \mathbf{X}_i(j,:) = & \underset{\mathbf{X}_i(j,:) \in \mathbb{R}^{1 \times r_K}}{\text{argmin}} \\ & \|\mathbf{U}_\Omega(j,:) - \mathbf{X}_i(j\,:)\mathbf{Y}_{i-1}\text{Diag}\left[\mathbf{E}_\Omega(j,:)\right]\|_\mathcal{F} + \lambda\|\mathbf{X}_i(j,:)\|_\mathcal{F}, \\ & = \mathbf{U}_\Omega(j,:)\text{Diag}\left[\mathbf{E}_\Omega(j,:)\right]\mathbf{Y}_{i-1}^\top \\ & \left(\mathbf{Y}_{i-1}\text{Diag}\left[\mathbf{E}_\Omega(j,:)\right]\mathbf{Y}_{i-1}^\top + \lambda\mathbf{I}\right)^{-1}, \qquad j = 1, \dots, m, \end{aligned} \tag{4}$$

where $\text{Diag}\left[\mathbf{v}\right]$ denotes a diagonal matrix with the diagonal entries being the entries of $\mathbf{v}$. Similarly, (3) can be solved as

$$\begin{aligned} \mathbf{Y}_i(:,j) = & \underset{\mathbf{Y}_i(:,j) \in \mathbb{R}^{r_K \times 1}}{\text{argmin}} \\ & \|\mathbf{U}_\Omega(:,j) - \text{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{X}_i\mathbf{Y}_i(:,j)\|_\mathcal{F} + \lambda\|\mathbf{Y}_i(:,j)\|_\mathcal{F}, \\ & = \left(\mathbf{X}_i^\top\text{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{X}_i + \lambda\mathbf{I}\right)^{-1} \\ & \mathbf{X}_i^\top\text{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{U}_\Omega(:,j), \qquad j = 1, \dots, n_K. \end{aligned} \tag{5}$$

Note that the output of this simple approach, $\hat{\mathbf{U}} = \mathbf{X}_N\mathbf{Y}_N$ for some N, satisfies only the rank constraint rank($\hat{\mathbf{U}}$) $= r_K$, but $\hat{\mathbf{U}}$ may not satisfy other $K-1$ rank constraints.

(ii) Naive approach 2: In this approach, we break the original problem into K independent completion problems, i.e., completing $\mathbf{M}_k \in \mathbb{R}^{m \times (n_k - n_{k-1})}$ with rank($\mathbf{M}_k$) $= r_k$. This method may be fast as each subproblem has a smaller dimension. However, it may result in a solution that does not satisfy any of the rank constraints except for the first one (for $\mathbf{M}_1$), since rank$\left(\mathbf{M}_k\right) = r_k$ does not necessarily result in rank$\left(\mathbf{U}_k\right) = r_k$ (except for $k = 1$).

(iii) Naive approach 3: In this approach, we first complete $\mathbf{U}_1$ with constraint rank($\mathbf{U}_1$) $= r_1$ using the above alternating minimization method. Then, we complete $\mathbf{U}_2 = [\mathbf{U}_1|\mathbf{M}_2]$ with the constraint rank($\mathbf{U}_2$) $= r_2$. Note that the $\mathbf{U}_1$ part of $\mathbf{U}_2$ is already complete and all missing entries are in the $\mathbf{M}_2$ part of $\mathbf{U}_2$. This is repeated and in the k-th step, we complete the $\mathbf{M}_k$ part of $\mathbf{U}_k = [\mathbf{U}_{k-1}|\mathbf{M}_k]$ with the constraint rank($\mathbf{U}_k$) $= r_k$. One important issue with this method is the error propagation when the sampling rate is low, i.e., the erroneously recovered entries at any step will lead to further errors in subsequent steps. However, the output of this method satisfies all rank constraints.

C. Structured Decomposition

In this paper, we propose a structured alternating minimization method for completing $\mathbf{U}$ such that: (1) all K rank constraints are satisfied at each iteration and, (2) it converges faster than the conventional alternating minimization for matrix completion with a single constraint rank($\mathbf{U}$) $= r_K$, by exploiting the additional $K-1$ rank constraints. To this end, we make use of a structured decomposition of $\mathbf{U}$ that is determined by the K rank constraints.

Definition 1. *Consider a decomposition* $\mathbf{U} = \mathbf{X}\mathbf{Y}$ *such that* $\mathbf{X} \in \mathbb{R}^{m \times r_K}$, $\mathbf{Y} \in \mathbb{R}^{r_K \times n_K}$, *and* $\mathbf{Y}(r_1+1 : r_K, 1 : n_1) = \mathbf{0}_{(r_K - r_1) \times n_1}$, $\mathbf{Y}(r_2+1 : r_K, n_1+1 : n_2) = \mathbf{0}_{(r_K - r_2) \times (n_2 - n_1)}$, $\dots$ *and* $\mathbf{Y}(r_{K-1}+1 : r_K, n_{K-2}+1 :$

2641-8770 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Auckland University of Technology. Downloaded on December 24,2020 at 23:42:27 UTC from IEEE Xplore. Restrictions apply.

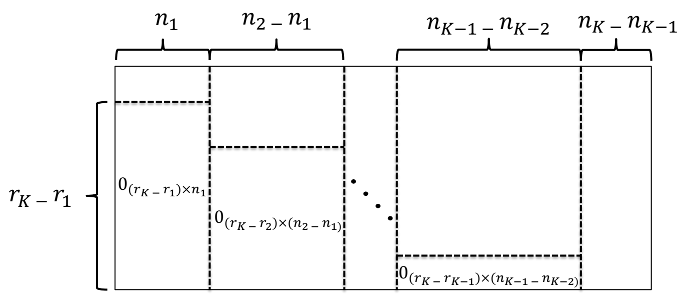

n_1
$n_2 - n_1$
$n_{K-1} - n_{K-2}$
$n_K - n_{K-1}$
$r_K - r_1$
$0_{(r_K - r_1)\times n_1}$
$0_{(r_K - r_2)\times(n_2 - n_1)}$
$0_{(r_K - r_{K-1})\times(n_{K-1} - n_{K-2})}$

Fig. 2: A matrix $\mathbf{Y}$ that satisfies the properties of a structured decomposition given in Definition 1.

$n_{K-1}) = \mathbf{0}_{(r_K - r_{K-1})\times(n_{K-1}-n_{K-2})}$. *This structure is shown in Fig. 2 and we call such decomposition* $\mathbf{U} = \mathbf{XY}$ *a structured decomposition.*

Lemma 1. *Consider a matrix* $\mathbf{U} \in \mathbb{R}^{m\times n_K}$ *that has a structured decomposition* $\mathbf{U} = \mathbf{XY}$. *Then, rank*$(\mathbf{U}(:,1 : n_k)) \le r_k$, $k = 1,\dots,K$.

Proof. Note that $\mathbf{U}(:,1 : n_k) = \mathbf{XY}(:,1 : n_k)$. Hence, under the structured decomposition, we conclude that $\mathbf{U}(:,1 : n_k) = \mathbf{X}(:,1 : r_k)\mathbf{Y}(1 : r_k, 1 : n_k)$ (because $\mathbf{Y}(r_k + 1 : r_K, 1 : n_k) = \mathbf{0}_{(r_K - r_k)\times n_k}$), $k = 1,\dots,K$. Then, $\mathbf{U}(:,1 : n_k) = \mathbf{X}(:,1 : r_k)\mathbf{Y}(1 : r_k, 1 : n_k)$ results that rank$(\mathbf{U}(:,1 : n_k)) \le r_k$, $k = 1,\dots,K$. □

Lemma 2. *If the matrix* $\mathbf{U} \in \mathbb{R}^{m\times n_K}$ *has the union of nested subspaces structure shown in Fig. 1, then there exists a structured decomposition* $\mathbf{U} = \mathbf{XY}$.

Proof. We need to show that there exists a basis $\mathbf{X}$ for $\mathbf{U}$ such that the first n_k columns of $\mathbf{U}$ belong to the subspace span of the first r_k columns of $\mathbf{X}$, $k = 1,\dots,K$. Note that it is easily verified that this statement is equivalent with the existence of a decomposition $\mathbf{U} = \mathbf{XY}$ such that $\mathbf{Y}$ satisfies the structure given in Definition 1. We show the mentioned statement by induction on k. In the k-th step, we construct $\mathbf{X}^k$ such that $\mathbf{U}_{k'}$ belongs to the column span of the first $r_{k'}$ columns of $\mathbf{X}^k$, $k' = 1,\dots,k$. Note that for $k = 1$ it is straightforward to construct $\mathbf{X}^1$, which is simply a basis for $\mathbf{U}_1$. Induction hypothesis results in the matrix $\mathbf{X}^k$ with the mentioned properties and in order to complete the induction, we need to show the existence of a matrix $\mathbf{X}^{k+1}$ such that $\mathbf{U}_{k'}$ belongs to the column span of the first $r_{k'}$ columns of $\mathbf{X}^k$, $k' = 1,\dots,k+1$.

We first note that $\mathbf{X}^k$ belongs to the column span of $\mathbf{U}_{k+1}$, because according to the induction hypothesis, $\mathbf{X}^k$ is a basis for $\mathbf{U}_k$, and $\mathbf{U}_k$ is a subset of columns of $\mathbf{U}_{k+1}$. Let $\mathcal{S}_k$ denote the column span of $\mathbf{X}^k$, which is an r_k-dimensional space and $\mathcal{S}'_{k+1}$ denote the column span of $\mathbf{U}_{k+1}$, which is an r_{k+1}-dimensional space. As a result of our earlier claim, $\mathcal{S}_k$ is a subspace of $\mathcal{S}'_{k+1}$. Let $\mathcal{S}''_k$ denote the $(r_{k+1} - r_k)$-dimensional subspace of $\mathcal{S}'_{k+1}$ such that the union of $\mathcal{S}_k$ and $\mathcal{S}''_k$ is $\mathcal{S}'_{k+1}$.

Consider an arbitrary basis $\mathbf{X}^{k'} \in \mathbb{R}^{m\times(r_{k+1}-r_k)}$ for the space $\mathcal{S}''_k$. Observe that by putting together the columns of $\mathbf{X}^k$ and $\mathbf{X}^{k'}$, i.e., $\mathbf{X}^{k+1} = [\mathbf{X}^k\ \mathbf{X}^{k'}]$, the new matrix $\mathbf{X}^{k+1} \in \mathbb{R}^{m\times r_{k+1}}$ is a basis for the space $\mathcal{S}'_{k+1}$. Therefore, $\mathbf{U}_{k+1}$ belongs to the column span of the first r_{k+1} columns of $\mathbf{X}^{k+1}$ since $\mathbf{X}^k$ has exactly r_{k+1} columns. Given the induction hypothesis, the proof is complete as $\mathbf{U}_{k'}$ belongs to the column span of the first $r_{k'}$ columns of $\mathbf{X}^{k+1}$, $k' = 1,\dots,k+1$. □

D. Proposed Structured Alternating Minimization Algorithm

We are interested in imposing the structured decomposition in the alternating minimization procedure for two reasons: (i) we know that according to Lemma 2 there exists such a decomposition and also, according to Lemma 1 the K rank constraints on the original data will hold in such decomposition and therefore, it is more likely that such decomposition results in the recovery of the original data. (ii) A structured decomposition has many zeros and convergence may be much faster than an unstructured decomposition.

There are two main challenges to impose a structured decomposition in alternating minimization: (i) an efficient initialization, and (ii) an efficient update of $\mathbf{Y}$ at each iteration.

1) Structured Initialization Methods:

SVD-based structured initialization: We first discuss the SVD-based method.

Note that $\mathbf{M}_{k_\Omega}$ represents the matrix obtained from sampling $\mathbf{M}_k$ according to column number $n_{k-1}+1$ to column number n_k of Ω, i.e.,

$$\mathbf{M}_{k_\Omega}(i,j) = \begin{cases} \mathbf{M}_k(i,j) & \text{if } \ (i, j+n_{k-1}) \in \Omega, \\ 0 & \text{if } \ (i, j+n_{k-1}) \notin \Omega. \end{cases} \tag{6}$$

In order to obtain an initialization $\mathbf{X}_0 \in \mathbb{R}^{m\times r_K}$ and $\mathbf{Y}_0 \in \mathbb{R}^{r_K\times n_K}$, we will obtain $\mathbf{X}^k \in \mathbb{R}^{m\times(r_k-r_{k-1})}$ and $\mathbf{Y}^k \in \mathbb{R}^{r_K\times(n_k-n_{k-1})}$ for $k=1,\dots,K$, where $n_0=r_0=0$. Then $\mathbf{X}_0 = \left[\mathbf{X}^1 \dots \mathbf{X}^K\right]$ and $\mathbf{Y}_0 = \left[\mathbf{Y}^1 \dots \mathbf{Y}^K\right]$.

In other words, in the k-th step, we obtain $r_k - r_{k-1}$ columns of the basis, i.e., $\mathbf{X}^k$, and the corresponding coefficients of these $r_k - r_{k-1}$ columns of the basis in $\mathbf{M}_{k'}$'s, i.e., $\mathbf{Y}^k$. Note that we set the coefficients corresponding to $\mathbf{X}^k$ in $\mathbf{M}_{k'}$ for $k' \geq k$ as zeros to meet the structured decomposition.

As the first step, we compute the SVD of $\mathbf{M}_{1_\Omega} \in \mathbb{R}^{m\times n_1}$ and pick the r_1 largest eigenvalues and their corresponding eigenvectors to construct matrices $\mathbf{X}^1 \in \mathbb{R}^{m\times r_1}$ and $\mathbf{Z}^1 \in \mathbb{R}^{r_1\times n_1}$, similar to the initialization explained in Sec. II-A for the naive approaches. Then, we define $\mathbf{Y}^1 = \left[\mathbf{Z}^{1^\top} \ \mathbf{0}_{n_1\times(r_K-r_1)}\right]^\top \in \mathbb{R}^{r_K\times n_1}$, as shown in Fig. 3.

In the second step, we first obtain the SVD of $\mathbf{M}_{2_\Omega} \in \mathbb{R}^{m\times n_2}$ and pick the $r_2 - r_1$ largest eigenvalues and their corresponding eigenvectors to construct matrices $\mathbf{X}^2 \in \mathbb{R}^{m\times(r_2-r_1)}$ and $\mathbf{Z}^2 \in \mathbb{R}^{(r_2-r_1)\times(n_2-n_1)}$. Next, we want to obtain $\mathbf{Y}^2 = \left[\mathbf{K}^{2^\top} \ \mathbf{Z}^{2^\top} \ \mathbf{0}_{(n_2-n_1)\times(r_K-r_2)}\right]^\top \in \mathbb{R}^{r_K\times(n_2-n_1)}$, as shown in Fig. 3, where $\mathbf{K}^2 \in \mathbb{R}^{r_1\times(n_2-n_1)}$ represents the coefficients of $\mathbf{X}^2$ in $\mathbf{M}_1$, which is based on the projection of matrix $(\mathbf{M}_2 - \mathbf{X}^2\mathbf{Z}^2)_\Omega$ on $\mathbf{X}^1$. Specifically, we have

$$\mathbf{K}^2 = \underset{\mathbf{K}^2\in\mathbb{R}^{r_1\times(n_2-n_1)}}{\operatorname{argmin}} \|\left((\mathbf{M}_2 - \mathbf{X}^2\mathbf{Z}^2) - \mathbf{X}^1\mathbf{K}^2\right)_\Omega\|_{\mathcal{F}} + \lambda\|\mathbf{K}^2\|_{\mathcal{F}}, \tag{7}$$

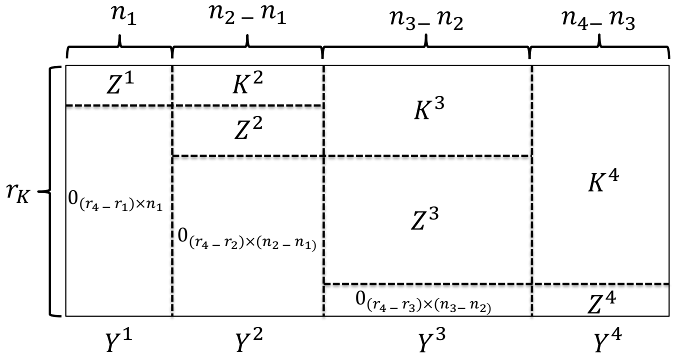


Fig. 3: Structure of $\mathbf{Y}_0$ in the SVD-based structured initialization.

which can be solved column by column similar to (5)

$$\mathbf{K}^2(:,j) = \left(\mathbf{X}^{1^\top}\operatorname{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{X}^1 + \lambda\mathbf{I}\right)^{-1} \mathbf{X}^{1^\top}\operatorname{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{T}^2_\Omega(:,j), \quad j=1,\dots,n_2-n_1, \tag{8}$$

where $\mathbf{T}^2 = \mathbf{M}_2 - \mathbf{X}^2\mathbf{Z}^2$.

Similarly, in the k-th step, we first obtain $\mathbf{X}^k \in \mathbb{R}^{m\times(r_k-r_{k-1})}$ and $\mathbf{Z}^k \in \mathbb{R}^{(r_k-r_{k-1})\times(n_k-n_{k-1})}$ from the SVD of $\mathbf{M}_{k_\Omega} \in \mathbb{R}^{m\times n_k}$. Then, we construct

$$\mathbf{Y}^k = \left[\mathbf{K}^{k^\top} \ \mathbf{Z}^{k^\top} \ \mathbf{0}_{(n_k-n_{k-1})\times(r_K-r_k)}\right]^\top \in \mathbb{R}^{r_K\times(n_k-n_{k-1})}, \tag{9}$$

as shown in Fig. 3, where $\mathbf{K}^k \in \mathbb{R}^{r_{k-1}\times(n_k-n_{k-1})}$ is the coefficient of $\bar{\mathbf{X}}^{k-1} = [\mathbf{X}^1 \dots \mathbf{X}^{k-1}]$ in $\mathbf{M}_k$, which is obtained based on the projection of $(\mathbf{M}_k - \mathbf{X}^k\mathbf{Z}^k)_\Omega$ on $\bar{\mathbf{X}}^k$, i.e.,

$$\mathbf{K}^k = \underset{\mathbf{K}^k\in\mathbb{R}^{r_{k-1}\times(n_k-n_{k-1})}}{\operatorname{argmin}} \|\left((\mathbf{M}_k - \mathbf{X}^k\mathbf{Z}^k) - \bar{\mathbf{X}}^{k-1}\mathbf{K}^k\right)_\Omega\|_{\mathcal{F}} + \lambda\|\mathbf{K}^k\|_{\mathcal{F}}. \tag{10}$$

The solution is given by

$$\mathbf{K}^k(:,j) = \left(\bar{\mathbf{X}}^{k-1^\top}\operatorname{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\bar{\mathbf{X}}^{k-1} + \lambda\mathbf{I}\right)^{-1} \bar{\mathbf{X}}^{k-1^\top}\operatorname{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{T}^k_\Omega(:,j), \quad j=1,\dots,n_k-n_{k-1}, \tag{11}$$

where $\mathbf{T}^k = \mathbf{M}_k - \mathbf{X}^k\mathbf{Z}^k$.

Remark 1. *In the above SVD-based structured initialization the choice of* $\mathbf{K}^k$ *in* (10) *plays a critical role. In* *particular, numerical experiments show that simply setting* $\mathbf{K}^k = \mathbf{0}_{r_K\times(n_k-n_{k-1})}$*, for* $k = 2,\dots,K$*, will result in a poor initialization which significantly reduces the convergence speed of the algorithm.*

Random structured initialization: We also consider random initialization for the proposed structured alternating minimization where $\mathbf{X}_0$ contains i.i.d. $\mathcal{N}(0,1)$ samples and the non-zero entries of $\mathbf{Y}_0$ in Definition 1 are also i.i.d. $\mathcal{N}(0,1)$ samples.

2) Structured Alternating Minimization:

Note that the initialization satisfies the structured decomposition. Now, we need to make sure that at each iteration of the algorithm this property still holds. In particular, in the i-th iteration of the structured alternating minimization procedure, given $\mathbf{X}_{i-1}$ and $\mathbf{Y}_{i-1}$, we first update $\mathbf{X}_i$ according to (4). Then, in (5) we only need to update the non-zero entries of $\mathbf{Y}_i(:,j)$ in the structured decomposition. That is, for $1 \le k \le K$ and $n_{k-1}+1 \le j \le n_k$ we have

$$\begin{aligned}&\mathbf{Y}_i(1:r_k,j) = \\ &\Big(\mathbf{X}_i(:,1:r_k)^\top \mathrm{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{X}_i(:,1:r_k) + \lambda\mathbf{I}\Big)^{-1} \\ &\mathbf{X}_i(:,1:r_k)^\top \mathrm{Diag}\left[\mathbf{E}_\Omega(:,j)\right]\mathbf{U}_\Omega(:,j). \qquad (12)\end{aligned}$$

Finally, we summarize the proposed structured alternating minimization algorithm for union of nested low-rank matrices completion in Algorithm 1. Note that at each iteration of this algorithm, including the initialization, the structured decomposition holds and therefore all K rank constraints hold.

Algorithm 1 Structured Alternating Minimization - Matrix Case

```
 1: Input $\mathbf{U}_\Omega$, $r_1,\dots,r_K$ and $n_1,\dots,n_K$.
 2: Initializing $\mathbf{X}_0 \in \mathbb{R}^{m\times r_K}$ and $\mathbf{Y}_0 \in \mathbb{R}^{r_K\times n_K}$ using either
    the SVD-based or random structured initialization.
 3: repeat
 4:     for $j = 1 : m$ do
 5:         Compute Eq. (4).
 6:     end for
 7:     for $k = 1 : K$ do
 8:         for $j = n_{k-1}+1 : n_k$ do
 9:             Compute Eq. (12).
10:         end for
11:     end for
12: until convergence/divergence
```

III. Completion of Union of Nested Low-Rank Tensors

In this section, we generalize the structured alternating minimization approach to a union of nested low-rank tensor spaces.

A. Background

Recall that the CP-rank of a tensor $\mathcal{U} \in \mathbb{R}^{m_1\times m_2\times\dots m_{d-1}\times m_d}$ is the minimum number r such that there exist $\mathbf{a}_j^l \in \mathbb{R}^{m_j}$ for $1 \le j \le d$ and $1 \le l \le r$ and

$$\mathcal{U} = \sum_{l=1}^{r} \mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \dots \otimes \mathbf{a}_d^l, \qquad (13)$$

or equivalently,

$$\mathcal{U}(x_1,x_2,\dots,x_d) = \sum_{l=1}^{r} \mathbf{a}_1^l(x_1)\mathbf{a}_2^l(x_2)\dots\mathbf{a}_d^l(x_d), \qquad (14)$$

where $\otimes$ denotes the tensor product (outer product) and $\mathcal{U}(x_1,x_2,\dots,x_d)$ denotes the entry of tensor $\mathcal{U}$ with coordinate $\vec{x} = (x_1,x_2,\dots,x_d)$ and $\mathbf{a}_j^l(x_j)$ denotes the x_j-th entry of vector $\mathbf{a}_j^l$. In other words, the CP-rank of a tensor $\mathcal{U}$ is the minimum number of rank-1 tensors that $\mathcal{U}$ can be decomposed to.

For notational convenience, define $M_{d-1} \triangleq m_1m_2\dots m_{d-1}$. Moreover, define the matrix $\widetilde{\mathbf{U}} \in \mathbb{R}^{M_{d-1}\times m_d}$ as the $(d-1)$-th unfolding of tensor $\mathcal{U}$, such that $\mathcal{U}(\vec{x}) = \widetilde{\mathbf{U}}(v(x_1,\dots,x_{d-1}),x_d)$, where $v : (x_1,\dots,x_{d-1}) \to \{1,2,\dots,M_{d-1}\}$ is a bijective mapping. Note that this is a vectorization mapping that merges the first $(d-1)$ dimensions and therefore, there is a corresponding inverse mapping $v^{-1} : \{1,2,\dots,M_{d-1}\} \to (x_1,\dots,x_{d-1})$. Moreover, for a $(d-1)$-dimensional tensor $\mathcal{V} \in \mathbb{R}^{m_1 \times \ldots \times m_{d-1}}$ we can define a vectorization operator vec : $\mathbb{R}^{m_1 \times \ldots \times m_{d-1}} \to \mathbb{R}^{M_{d-1}}$ using the mapping $v(\cdot)$ such that $\mathcal{V}(x_1, ..., x_{d-1}) = \text{vec}(\mathcal{V})(v(x_1, ..., x_{d-1}))$. We call a vector $\mathbf{u} \in \mathbb{R}^{M_{d-1}}$ a "structured column" if $\text{vec}^{-1}(\mathbf{u}) \in \mathbb{R}^{m_1 \times \ldots \times m_{d-1}}$ is a rank-1 tensor, i.e., there exist $\mathbf{u}_j \in \mathbb{R}^{m_j}$ for $j = 1, \ldots, d-1$, such that $\mathbf{u} = \text{vec}(\mathbf{u}_1 \otimes \ldots \otimes \mathbf{u}_{d-1})$.

Lemma 3. *The CP-rank of a tensor $\mathcal{U}$ is equal to the* **minimum number of structured columns** *that span all columns of $\widetilde{\mathbf{U}}$.*

Proof. First we show that there exist $\mathbf{b}_1^l \in \mathbb{R}^{M_{d-1}}$ and $\mathbf{b}_2^l \in \mathbb{R}^{m_d}$ for $1 \leq l \leq r$ such that

$$\widetilde{\mathbf{U}} = \sum_{l=1}^{r} \mathbf{b}_1^l \otimes \mathbf{b}_2^l. \tag{15}$$

Recall the CP decomposition in (13). Then, we define $\mathcal{A}_1^l = \mathbf{a}_1^l \otimes \ldots \otimes \mathbf{a}_{d-1}^l$ and $\mathbf{b}_2^l = \mathbf{a}_d^l$ for $1 \leq l \leq l$ and define $\mathbf{b}_1^l = \text{vec}(\mathcal{A}_1^l)$. Hence, there exist $\mathbf{b}_1^l \in \mathbb{R}^{M_{d-1}}$ and $\mathbf{b}_2^l \in \mathbb{R}^{m_d}$ for $1 \leq l \leq r$ such that (15) holds. Therefore, there exist r structured columns that span all columns of $\widetilde{\mathbf{U}}$. Similarly, if there exists $(r-1)$ structured columns that span all columns of $\widetilde{\mathbf{U}}$ we can use vec^{-1} and obtain a CP-decomposition of rank $(r-1)$ for $\mathcal{U}$. Therefore, $\text{rank}(\mathcal{U}) = r$ means that r is the minimum number of structured columns that span all columns of $\widetilde{\mathbf{U}}$. □

Definition 2. *According to the above lemma, rank($\mathcal{U}$) $= r$ concludes that there exists a set $\mathcal{S}$ consisting of r structured columns whose column span (denoted by $\mathcal{T}$) includes any column of $\widetilde{\mathbf{U}}$. In other words, the column span of these r structured columns, i.e., $\mathcal{T}$, is an unfolded tensor space of rank r. We call such r structured columns a tensor basis for $\mathcal{U}$.*

B. Problem Statement

Consider a fixed number $K \geq 2$ and partially sampled d-way tensors $\mathcal{M}_k \in \mathbb{R}^{m_1 \times m_2 \times \ldots m_{d-1} \times c_k}$, $k = 1, 2, \ldots, K$. Define $n_k = c_1 + \cdots + c_k$ for $k = 1, \ldots, K$, and $c_0 = n_0 = 0$. Let $\mathcal{U}_k \in \mathbb{R}^{m_1 \times m_2 \times \ldots, m_{d-1} \times n_k}$, be the concatenation of $\mathcal{M}_1, \ldots, \mathcal{M}_k$ along the d-th dimension, and r_k denote the CP-rank of $\mathcal{U}_k$, $k = 1, 2, \ldots, K$. Let Ω denote the sampled index set, i.e., $\Omega = \{\vec{x} = (x_1, \ldots, x_d) : \mathcal{U}(\vec{x}) \text{ is sampled}\}$. Moreover, define $\mathcal{U}_\Omega$ as the tensor obtained from sampling $\mathcal{U} = \mathcal{U}_K$ according to Ω, i.e.,

$$\mathcal{U}_\Omega(\vec{x}) = \begin{cases} \mathcal{U}(\vec{x}) & \text{if } \ \vec{x} \in \Omega, \\ 0 & \text{if } \ \vec{x} \notin \Omega. \end{cases} \tag{16}$$

Moreover, we assume a union of nested tensor subspaces structure similar to the matrix case. Specifically, assume that there exist structured columns $\mathbf{u}_l \in \mathbb{R}^{M_{d-1}}$, $l = 1, \ldots, r_K$, such that $\mathcal{S}_k = \{\mathbf{u}_1, \ldots, \mathbf{u}_{r_k}\}$ is a tensor basis for $\mathcal{U}_k$, $k = 1, \ldots, K$. Note that we have $\text{rank}(\mathcal{U}_k) = \text{rank}(\mathcal{M}_k) = r_k$, $k = 1, \ldots, K$. The problem then is to complete the tensor $\mathcal{U}_\Omega$ given the above mentioned union of nested tensor subspaces structure, and the rank values $r_1, ..., r_K$.

C. Alternating Minimization For Tensor Completion

Recall the CP decomposition $\mathcal{U} = \sum_{l=1}^{r_K} \mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \ldots \otimes \mathbf{a}_d^l$, where $\mathbf{a}_j^l \in \mathbb{R}^{m_j}$ for $1 \leq j \leq d$ and $1 \leq l \leq r_K$. Define $\mathbf{A}_j = [\mathbf{a}_j^1 | \ldots | \mathbf{a}_j^{r_K}] \in \mathbb{R}^{m_j \times r_K}$, $j = 1, \ldots, d$. In alternating minimization, given the result of the $(i-1)$-th iteration $\mathbf{A}_j^{(i-1)} \in \mathbb{R}^{m_j \times r_K}$, $j = 1, \ldots, d$, at the i-th iteration, we update all $\mathbf{A}_j$'s one by one in d steps. In particular, in the j-th step, we solve for $\mathbf{A}_j^{(i)}$ using the latest values $\mathbf{A}_1^{(i)}, \ldots \mathbf{A}_{j-1}^{(i)}$, and $\mathbf{A}_{j+1}^{(i-1)}, \ldots \mathbf{A}_d^{(i-1)}$ by solving the following regularized least squares problem

$$\min_{\mathbf{A}_j^{(i)} \in \mathbb{R}^{m_j \times r_K}} \Bigg\| \mathcal{U}_\Omega - \Bigg(\sum_{l=1}^{r_K} \mathbf{A}_1^{(i)}(:,l) \otimes \ldots \otimes \mathbf{A}_{j-1}^{(i)}(:,\ell) \otimes \mathbf{A}_j^{(i)}(:,l) \otimes \mathbf{A}_{j+1}^{(i-1)}(:,l) \otimes \ldots \otimes \mathbf{A}_d^{(i-1)}(:,l) \Bigg)_\Omega \Bigg\|_{\mathcal{F}} + \lambda \|\mathbf{A}_j^{(i)}\|_{\mathcal{F}}, j = 1, \ldots, d. \tag{17}$$

To solve (17), we first write it in matrix form. To do this, we define an operator that reorders the dimensions of a tensor. Consider the tensor in (13) and another tensor

$$\mathcal{U}' = \sum_{l=1}^{r_K} \mathbf{a}_1^l \otimes \ldots \otimes \mathbf{a}_{j-1}^l \otimes \mathbf{a}_{j+1}^l \otimes \ldots \otimes \mathbf{a}_d^l \otimes \mathbf{a}_j^l. \tag{18}$$

Then, it is clear that the only difference between these two tensors is that the order of dimensions has changed from $1, 2, \ldots, d$ in $\mathcal{U}$ to $1, 2, \ldots, j-1, j+1, \ldots, d, j$ in $\mathcal{U}'$. Denote such a dimension reordering operation by $\mathcal{U}' = \sigma_j(\mathcal{U}) \in \mathbb{R}^{m_1 \times \ldots \times m_{j-1} \times m_{j+1} \times \ldots \times m_d \times m_j}$ such that

$$\mathcal{U}(x_1, x_2, \ldots, x_d) = \sigma_j(\mathcal{U})(x_1, x_2, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d, x_j). \tag{19}$$

Then, (17) can be rewritten as

$$\min_{\mathbf{A}_j^{(i)} \in \mathbb{R}^{m_j \times r_K}} \Bigg\| \sigma_j(\mathcal{U})_{\sigma_j(\Omega)} - \Bigg(\sum_{l=1}^{r_K} \mathbf{A}_1^{(i)}(:,l) \otimes \ldots \otimes \mathbf{A}_{j-1}^{(i)}(:,l) \otimes \mathbf{A}_{j+1}^{(i-1)}(:,l) \otimes \ldots \otimes \mathbf{A}_d^{(i-1)}(:,l) \otimes \mathbf{A}_j^{(i)}(:,l) \Bigg)_{\sigma_j(\Omega)} \Bigg\|_{\mathcal{F}} + \lambda \|\mathbf{A}_j^{(i)}\|_{\mathcal{F}}, j = 1, \ldots, d. \tag{20}$$

Now we define $\mathbf{\Gamma}_j^{(i)} \in \mathbb{R}^{m_1 \ldots m_{j-1} m_{j+1} \ldots m_d \times r_K}$ such that

$$\mathbf{\Gamma}_j^{(i)}(:,\ell) \triangleq \text{vec}(\mathbf{A}_1^{(i)}(:,\ell) \otimes \ldots \otimes \mathbf{A}_{j-1}^{(i)}(:,\ell) \otimes \mathbf{A}_{j+1}^{(i-1)}(:,\ell) \otimes \ldots \otimes \mathbf{A}_d^{(i-1)}(:,\ell)) \in \mathbb{R}^{m_1 \ldots m_{j-1} m_{j+1} \ldots m_d}, \ \ell = 1, \ldots, r_K. \tag{21}$$

Then, we can rewrite (20) using its $(d-1)$-th unfolding as

$$\min_{\mathbf{A}_j^{(i)} \in \mathbb{R}^{m_j \times r_K}} \Bigg\| \widetilde{\sigma_j(\mathcal{U})}_{\widetilde{\sigma_j(\Omega)}} - \left(\mathbf{\Gamma}_j^{(i)} \mathbf{A}_j^{(i)^\top} \right)_{\widetilde{\sigma_j(\Omega)}} \Bigg\|_{\mathcal{F}} + \lambda \|\mathbf{A}_j^{(i)}\|_{\mathcal{F}}, \tag{22}$$

where $\widetilde{\sigma_j(\mathcal{U})}$ and $\widetilde{\sigma_j(\Omega)}$ denote the $(d-1)$-th unfolding of $\sigma_j(\mathcal{U})$ and $\sigma_j(\Omega)$, respectively. Note that (22) is of the same form as (3) and hence similar to (5), we can write

$$\mathbf{A}_j^{(i)}(l,:) = \widetilde{\sigma_j(\mathcal{U})}_{\widetilde{\sigma_j(\Omega)}}^{\top}{}^{\top}(l,:) \text{Diag}\left[\mathbf{E}_{\widetilde{\sigma_j(\Omega)}}(:,l) \right] \mathbf{\Gamma}_j^{(i)} \left((\mathbf{\Gamma}_j^{(i)})^\top \text{Diag}\left[\mathbf{E}_{\widetilde{\sigma_j(\Omega)}}(:,l) \right] \mathbf{\Gamma}_j^{(i)} + \lambda \mathbf{I} \right)^{-1}, l = 1, \ldots, m_j, \tag{23}$$

where $\mathbf{E}$ denotes an all-one $(m_1 \ldots m_{j-1} m_{j+1} \ldots m_d) \times m_j$ matrix.

D. Naive Approaches

We can generalize the three naive approaches for the matrix case to the tensor case as follows.

(i) Naive approach 1: We apply the alternating minimization procedure described in Sec. III-C to tensor $\mathcal{U}$ with the only constraint rank$(\mathcal{U}) = r_K$.

(ii) Naive approach 2: We break the original problem into K independent completion problems, i.e., completing $\mathcal{M}_k \in \mathbb{R}^{m_1 \times \ldots \times m_{d-1} \times (n_k - n_{k-1})}$ with rank$(\mathcal{M}_k) = r_k$, $k = 1, \ldots, K$.

(iii) Naive approach 3: We first complete $\mathcal{U}_1$ with constraint rank$(\mathcal{U}_1) = r_1$ using the above alternating minimization method. Then, we complete $\mathcal{U}_2$ with the constraint rank$(\mathcal{U}_2) = r_2$. Note that the $\mathcal{U}_1$ part of $\mathcal{U}_2$ is already complete and all missing entries are in the $\mathcal{M}_2$ part of $\mathcal{U}_2$. This is repeated and in the k-th step, we complete the $\mathcal{M}_k$ part of $\mathcal{U}_k$ with the constraint rank$(\mathcal{U}_k) = r_k$, $k = 1, \ldots, K$.

Similarly to the matrix case, for each of the above naive methods, either CP-based or random initialization can be employed. Specifically, for CP-based initialization, we first calculate the CP decomposition of $\mathcal{U}_\Omega$. Then, we normalize each vector $\mathbf{a}_j^l$ in (13) to have unit norm so that the l-th out-product has a weight of $\|\mathbf{a}_1^l\| ... \|\mathbf{a}_d^l\|$. Then, we choose the leading r_K rank-1 components, sorted according to the weights, to obtain a rank-r_K initialization. And for random

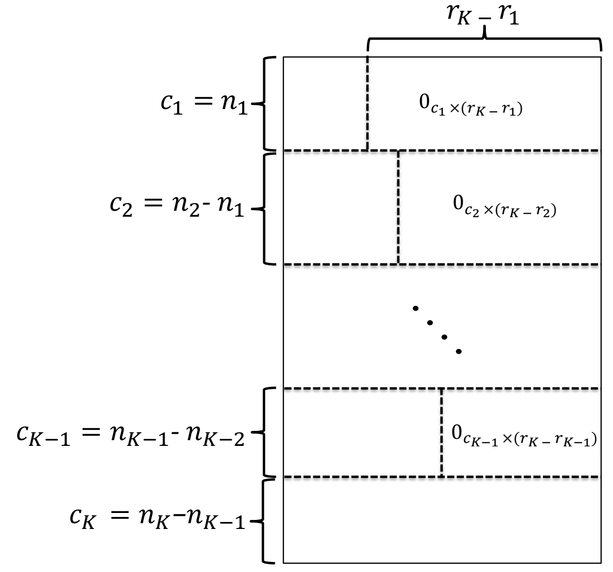


Fig. 4: A matrix $\mathbf{A}_d$ that satisfies the properties of a structured decomposition given in Lemma 4.

initialization, we simply set entries of $\mathbf{A}_j^{(0)}$, $j = 1, ..., d$ as i.i.d $\mathcal{N}(0,1)$ samples.

E. Structured Decomposition

Similar to the matrix case, we develop a tensor completion method based on alternating minimization that takes into account all K rank constraints. First, similar to Definition 1 and Lemma 2 for the matrix case, we have the following lemma on the existence of a structured CP decomposition for tensor $\mathcal{U}$ that has a union of nested subspaces structure defined in Sec. III.B. Recall that $n_k = c_1 + \cdots + c_k$.

Lemma 4. *If the tensor $\mathcal{U} \in \mathbb{R}^{m_1\times\ldots\times m_{d-1}\times m_d}$ has the union of nested tensor subspaces structure, then there exist $\mathbf{a}_d^l \in \mathbb{R}^{m_d}$ for $l = 1,\ldots,r_K$ such that $\mathcal{U} = \sum_{l=1}^{r_K} \mathbf{a}_1^l \otimes \ldots \otimes \mathbf{a}_{d-1}^l \otimes \mathbf{a}_d^l$ and for any $k = 1,\ldots,K$, $x = n_{k-1}+1,\ldots,n_k$ and $l = r_k+1,\ldots,r_K$ we have $\mathbf{a}_d^l(x) = 0$. In other words, $\mathbf{A}_d(n_{k-1}{+}1 : n_k, 1 : r_K{-}r_k) = \mathbf{0}_{c_k\times(r_K-r_k)}$, $k = 1,\ldots,K$. We call such CP-decomposition of $\mathcal{U}$ a structured decomposition (shown in Fig. 4).*

Proof. Note that since each column of $\widetilde{\mathbf{M}}_k$ (the $(d-1)$-th unfolding of $\mathcal{M}_k$) is chosen from the column span of $\mathcal{S}_k$, there exist $\mathbf{B}_k \in \mathbb{R}^{r_k\times c_k}$ such that $\widetilde{\mathbf{M}}_k = [\mathbf{u}_1 \ldots \mathbf{u}_{r_k}]\mathbf{B}_k$, $k = 1,\ldots,K$. Recall that $\widetilde{\mathbf{U}} = [\widetilde{\mathbf{M}}_1 \ldots \widetilde{\mathbf{M}}_K]$. Therefore, we can write

$$\widetilde{\mathbf{U}} = [\mathbf{u}_1 \ldots \mathbf{u}_{r_K}]\underbrace{[\mathbf{C}_1 \ldots \mathbf{C}_K]}_{[\mathbf{a}_d^1 \ldots \mathbf{a}_d^{r_K}]^\top}, \tag{24}$$

where $\mathbf{C}_k = [\mathbf{B}_k^\top \;\; \mathbf{0}_{c_k\times(r_K-r_k)}]^\top \in \mathbb{R}^{r_K\times c_k}$ and $[\mathbf{a}_d^1 \ldots \mathbf{a}_d^{r_K}]^\top = [\mathbf{C}_1 \ldots \mathbf{C}_K]$. Hence, for any $k = 1,\ldots,K$ and $l = r_k+1,\ldots,r_K$ we have $\mathbf{a}_d^l(x) = 0$ if $n_{k-1}+1 \le x \le n_k$.

Since $\mathbf{u}_l = \text{vec}(\mathbf{a}_1^l \otimes \ldots \otimes \mathbf{a}_{d-1}^l)$ for $l = 1,\ldots,r_K$, (24) can be written as

$$\mathcal{U} = \sum_{l=1}^{r_K} \mathbf{a}_1^l \otimes \ldots \otimes \mathbf{a}_{d-1}^l \otimes \mathbf{a}_d^l, \tag{25}$$

and hence, the proof is complete. □

Remark 2. *Note that the structure in Fig. 4 is the transposed structure in Fig. 3.*

F. Proposed Structured Alternating Minimization for Union of Nested Tensor Subspaces

1) Structured Initialization Methods:

CP-based structured initialization: We obtain such structured initialization in K steps: in the k-th step, $k = 1,\ldots,K$, we obtain $\mathbf{B}_j^k \in \mathbb{R}^{m_j\times(r_k-r_{k-1})}$ for $j = 1,\ldots,d-1$, and $\mathbf{B}_d^k \in \mathbb{R}^{c_k\times r_K}$, and the initialization is $\mathbf{A}_j^{(0)} = [\mathbf{B}_j^1 \ldots \mathbf{B}_j^K]$, $j = 1,\ldots,d-1$ and $\mathbf{A}_d^{(0)} = \left[\mathbf{B}_d^{1^\top} \ldots \mathbf{B}_j^{K^\top}\right]^\top$. Note that here $\mathbf{A}_j^{(0)}$ for $j = 1,\ldots,d-1$ and $\mathbf{A}_d^{(0)}$ correspond to $\mathbf{X}_0$ and $\mathbf{Y}_0$ in Sec. II-D, respectively.

We first perform the CP-decomposition of $\mathcal{M}_{1_\Omega} \in \mathbb{R}^{m_1\times\ldots m_{d-1}\times c_1}$ and retain the r_1 leading rank-1 components, to obtain $\mathbf{B}_j^1 \in \mathbb{R}^{m_j\times r_1}$ for $j = 1,\ldots,d-1$, and $\mathbf{C}_d^1 \in \mathbb{R}^{c_1\times r_1}$, i.e.,

$$\mathcal{M}_{1_\Omega} \approx \sum_{l=1}^{r_1} \mathbf{B}_1^1(:,l) \otimes \ldots \otimes \mathbf{B}_{d-1}^1(:,l) \otimes \mathbf{C}_d^1(:,l) \tag{26}$$

Then we define $\mathbf{B}_d^1 = \left[\mathbf{C}_d^1 \ \mathbf{0}_{c_1 \times (r_K - r_1)}\right] \in \mathbb{R}^{c_1 \times r_K}$ that meets the structure of the top block row in Fig. 4.

In the k-th step, $k = 2, ..., K$, we perform the CP-decomposition of $\mathcal{M}_{k_\Omega} \in \mathbb{R}^{m_1 \times ... m_{d-1} \times c_k}$ and retain the $r_k - r_{k-1}$ leading rank-1 components denoted by $\mathbf{B}_j^k \in \mathbb{R}^{m_j \times (r_k - r_{k-1})}$ for $j = 1, \ldots, d-1$, and $\mathbf{C}_d^k \in \mathbb{R}^{c_k \times (r_k - r_{k-1})}$. Then, we define $\mathbf{B}_d^k = \left[\mathbf{K}^k \ \mathbf{C}_d^k \ \mathbf{0}_{c_k \times (r_K - r_k)}\right] \in \mathbb{R}^{c_k \times r_K}$ that meets the structure of the k-th block row in Fig. 4, where $\mathbf{K}^k \in \mathbb{R}^{c_k \times r_{k-1}}$ represents the coefficients of the structured columns 1 to r_{k-1} in $\mathcal{M}_k$, that is calculated as follows.

Let $\bar{\mathcal{M}}_k \in \mathbb{R}^{m_1 \times ... m_{d-1} \times c_k}$ denote the rank-$(r_k - r_{k-1})$ approximation of $\mathcal{M}_{k_\Omega}$, i.e.,

$$\bar{\mathcal{M}}_k \triangleq \sum_{l=1}^{r_k - r_{k-1}} \mathbf{B}_1^k(:,l) \otimes \ldots \otimes \mathbf{B}_{d-1}^k(:,l) \otimes \mathbf{C}_d^k(:,l) \quad (27)$$

Define $\bar{\mathbf{B}}_j^{k-1} = [\mathbf{B}_j^1 \ldots \mathbf{B}_j^{k-1}] \in \mathbb{R}^{m_j \times r_{k-1}}$, $j = 1, \ldots, d-1$. Then $\mathbf{K}^k$ is the projection of tensor $(\mathcal{M}_k - \bar{\mathcal{M}}_k)$ on the structured columns 1 to r_{k-1}, i.e.,

$$\mathbf{K}^k = \underset{\mathbf{K}^k \in \mathbb{R}^{c_k \times r_{k-1}}}{\operatorname{argmin}} \left\| \left(\mathcal{M}_k - \bar{\mathcal{M}}_k\right)_\Omega - \left(\sum_{l=1}^{r_{k-1}} \bar{\mathbf{B}}_1^{k-1}(:,l) \right.\right.$$
$$\left.\left. \otimes \ldots \otimes \bar{\mathbf{B}}_{d-1}^{k-1}(:,l) \otimes \mathbf{K}^k(:,l)\right)_\Omega \right\|_{\mathcal{F}} + \lambda \|\mathbf{K}^k\|_{\mathcal{F}}, \quad (28)$$

which is similar to (20) and can be rewritten using the $(d-1)$-th unfoldings of the corresponding tensors as

$$\mathbf{K}^k = \underset{\mathbf{K}^k \in \mathbb{R}^{c_k \times r_{k-1}}}{\operatorname{argmin}} \left\| \left(\widetilde{\mathbf{M}}_k - \widetilde{\bar{\mathbf{M}}}_k\right)_{\widetilde{\mathbf{\Omega}}} - \left(\widetilde{\mathbf{B}}_k \mathbf{K}^{k^\top}\right)_{\widetilde{\mathbf{\Omega}}} \right\|_{\mathcal{F}}$$
$$+ \lambda \|\mathbf{K}^k\|_{\mathcal{F}}, \quad (29)$$

where $\widetilde{\mathbf{B}}_k \in \mathbb{R}^{m_1 \ldots m_{d-1} \times r_{k-1}}$ is defined as

$$\widetilde{\mathbf{B}}_k(:,\ell) \triangleq \operatorname{vec}(\bar{\mathbf{B}}_1^{k-1}(:,\ell) \otimes \ldots \otimes \bar{\mathbf{B}}_{d-1}^{k-1}(:,\ell))$$
$$\in \mathbb{R}^{m_1 \ldots m_{d-1}}, \quad \ell = 1, \ldots, r_{k-1}. \quad (30)$$

And the solution is

$$\mathbf{K}^k(l,:) = \left(\widetilde{\mathbf{M}}_k - \widetilde{\bar{\mathbf{M}}}_k\right)^\top_{\widetilde{\mathbf{\Omega}}^\top}(l,:) \operatorname{Diag}\left[\mathbf{E}_{\widetilde{\mathbf{\Omega}}}(:,l)\right]$$
$$\widetilde{\mathbf{B}}_k \left(\widetilde{\mathbf{B}}_k^\top \operatorname{Diag}\left[\mathbf{E}_{\widetilde{\mathbf{\Omega}}}(:,l)\right] \widetilde{\mathbf{B}}_k + \lambda \mathbf{I}\right)^{-1}, l = 1, \ldots, c_k, \quad (31)$$

where $\mathbf{E}$ denotes an all-one $(m_1 \ldots m_{d-1}) \times m_d$ matrix.

Random structured initialization: We also consider the random initialization for the proposed structured alternating minimization where $\mathbf{A}_j^{(0)}$ contains i.i.d. $\mathcal{N}(0,1)$ samples for $j = 1, \ldots, d-1$, and the non-zero entries of $\mathbf{A}_d^{(0)}$ in Lemma 4 are also i.i.d. $\mathcal{N}(0,1)$ samples.

2) Structured Alternating Minimization:

Note that the initialization satisfies the structured decomposition. Now, we need to make sure that at each iteration of the algorithm this property still holds. In particular, in the i-th iteration of the structured alternating minimization procedure, given $\mathbf{A}_j^{(i-1)} \in \mathbb{R}^{m_j \times r_K}$ for $j = 1, \ldots, d$, we first update $\mathbf{A}_j^{(i)} \in \mathbb{R}^{m_j \times r_K}$ for $j = 1, \ldots, d-1$ according to (23). Then, to update $\mathbf{A}_d^{(i)} \in \mathbb{R}^{m_d \times r_K}$ we only need to update the non-zero entries in the structured decomposition. That is in (23), for $1 \leq k \leq K$ and $n_{k-1} + 1 \leq l \leq n_k$ we have

$$\mathbf{A}_d^{(i)}(l, 1:r_k) = \widetilde{\mathbf{U}}^\top_{\widetilde{\mathbf{\Omega}}^\top}(l,:) \operatorname{Diag}\left[\mathbf{E}_{\widetilde{\mathbf{\Omega}}}(:,l)\right] \widetilde{\bar{\mathbf{A}}}_d^{(i)}(:, 1:r_k)$$
$$\left((\widetilde{\bar{\mathbf{A}}}_d^{(i)})^\top(1:r_k,:) \operatorname{Diag}\left[\mathbf{E}_{\widetilde{\mathbf{\Omega}}}(:,l)\right] \widetilde{\bar{\mathbf{A}}}_d^{(i)}(:, 1:r_k) + \lambda \mathbf{I}\right)^{-1},$$
$$(32)$$

where $\widetilde{\bar{\mathbf{A}}}_d^{(i)} \in \mathbb{R}^{m_1 \ldots m_{d-1} \times r_K}$ denotes the $(d-1)$-th unfolding of $\bar{\mathcal{A}}_d^{(i)} = \mathbf{A}_1^{(i)} \otimes \ldots \otimes \mathbf{A}_{d-1}^{(i)} \in \mathbb{R}^{m_1 \times \ldots \times m_{d-1} \times r_K}$ and $\mathbf{E}$ denotes an all-one $(m_1 \ldots m_{d-1}) \times m_d$ matrix.

Finally, we summarize the proposed structured alternating minimization algorithm for union of nested low-rank tensor subspaces completion in Algorithm 2. Note that at each iteration of this algorithm, including the initialization, the structured decomposition holds and therefore all K rank constraints hold.

IV. Simulation Results

A. Matrix Case

We consider an example where $K = 4$, $m = 1000$, $n_1 = 300$, $n_2 = 500$, $n_3 = 700$, $n_4 = 900$, $r_1 = 50$, $r_2 = 60$, $r_3 = 70$ and $r_4 = 80$. In order to generate a matrix that

Algorithm 2 Structured Alternating Minimization - Tensor Case

```
1:  Input $\mathcal{U}_\Omega$, $r_1, \dots, r_K$ and $n_1, \dots, n_K$.
2:  Initializing $\mathbf{A}_j^{(0)} \in \mathbb{R}^{m_j \times r_K}$ for $j = 1, \dots, d$ using either
    the CP-based or random structured initialization.
3:  repeat
4:      for $j = 1 : d-1$ do
5:          for $l = 1 : m_j$ do
6:              Compute Eq. (23).
7:          end for
8:      end for
9:      for $k = 1 : K$ do
10:         for $l = n_{k-1}+1 : n_k$ do
11:             Compute Eq. (32).
12:         end for
13:     end for
14: until convergence/divergence
```

is randomly chosen from the manifold corresponding to the given rank constraints, we first generate $\mathbf{X} \in \mathbb{R}^{1000 \times 80}$ ($r_4 = 80$ basis columns) with entries being i.i.d. $\mathcal{N}(0,1)$ samples. Then, we generate $\mathbf{Y} \in \mathbb{R}^{80 \times 900}$ such that it satisfies the structured decomposition given in Definition 1, i.e., $\mathbf{Y}(51:80, 1:300) = \mathbf{0}_{30\times 300}$, $\mathbf{Y}(61:80, 301:500) = \mathbf{0}_{20\times 200}$ and $\mathbf{Y}(71:80, 501:700) = \mathbf{0}_{10\times 200}$ and the rest of the entries are i.i.d. $\mathcal{N}(0,1)$ samples. Then, the matrix $\mathbf{U} = \mathbf{XY}$ satisfies all the rank constraints. We sample the entries of $\mathbf{U}$ independently with probability $0 < p < 1$. The regularization weight is set as $\lambda = 0.01$. We define the convergence metric as $\epsilon_i = \frac{\|\mathbf{X}_i\mathbf{Y}_i\|_{\mathcal{F}} - \|\mathbf{X}_{i-1}\mathbf{Y}_{i-1}\|_{\mathcal{F}}}{\|\mathbf{X}_i\mathbf{Y}_i\|_{\mathcal{F}}}$ and convergence is reached if $\epsilon_i < 10^{-3}$. On the other hand, divergence is declared if $\|\mathbf{X}_i\|_{\mathcal{F}} > 10^6 \|\mathbf{X}_0\|_{\mathcal{F}}$ or $\|\mathbf{Y}_i\|_{\mathcal{F}} > 10^6 \|\mathbf{Y}_0\|_{\mathcal{F}}$.

1) Noiseless Matrix: We say the sampled matrix $\mathbf{U}$ is recovered if the algorithm converges and the normalized error satisfies $\frac{\|\hat{\mathbf{U}} - \mathbf{U}\|_{\mathcal{F}}}{\|\mathbf{U}\|_{\mathcal{F}}} < 0.01$, where $\hat{\mathbf{U}}$ denotes the completed matrix. We consider different number of rank constraints: for $K = 4$, we include all rank constraints r_1, r_2, r_3 and r_4; for $K = 3$, we include rank constraints r_2, r_3 and r_4; and for $K = 2$, we include rank constraints r_3 and r_4. For each case, we generate 100 random matrices from the corresponding manifold. Then for each value of the sampling probability p, we run different completion algorithms on these sampled matrices and calculate the recovery rates.

First, to see the impact of multiple rank constraints on the convergence, in Fig. 5 we illustrate the convergence behaviors of the Naive method 1 and the structured approach, for $K = 4$, $p = 0.3$ and a particular sampled matrix. It is seen that it takes 8 and 10 iterations for the structured approach and the Naive 1 method, respectively, to reach the convergence condition $\epsilon_i < 10^{-3}$.

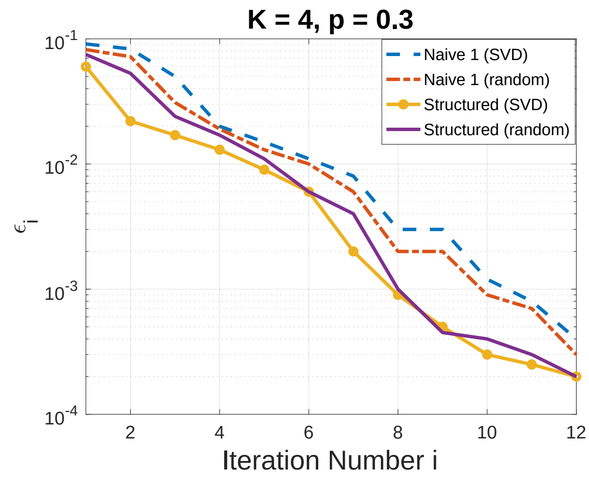
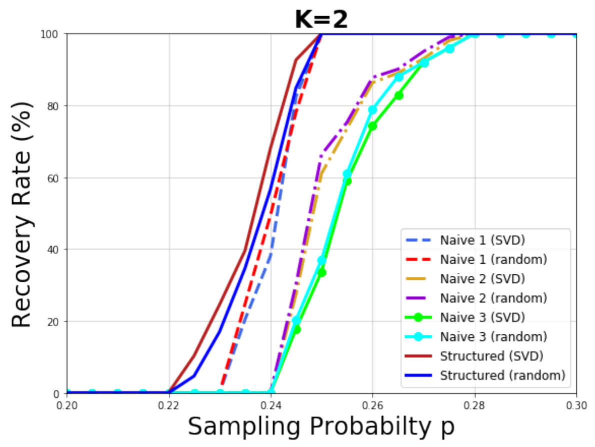


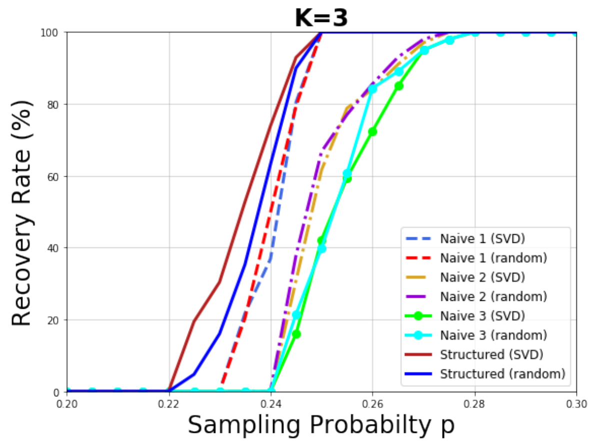
Fig. 5: Convergence comparison for noiseless matrices with $K = 4$ and $p = 0.3$.

Next the recovery rate performances of different algorithms are compared in Figs. 6(a), 6(b), and 6(c) for $K = 2, 3$, and 4, respectively. A number of observations are in order. First, for all three values of K, for both Naive methods 2 and 3, the recovery rate is 1 for $p \geq 0.28$ and it is 0 when $p \leq 0.24$; for both Naive method 1 and the structured approach, the recovery rate is 1 for $p \geq 0.25$; and the recovery rate is 0 for $p \leq 0.23$ for Naive 1. Hence among the three naive methods, Naive 1 has the best recovery performance even though it ignores all additional rank constraints. Second, the structured approach mainly improves the region where the recovery rate is below 1. In particular, the recovery rate is 0 for $p \leq 0.22$ when $K = 2, 3$, whereas it becomes $p \leq 0.21$ when $K = 4$. Moreover, in the region where the recovery rate is below 1, i.e., $p \in (0.22, 0.25)$, its recovery rate is higher than that of Naive

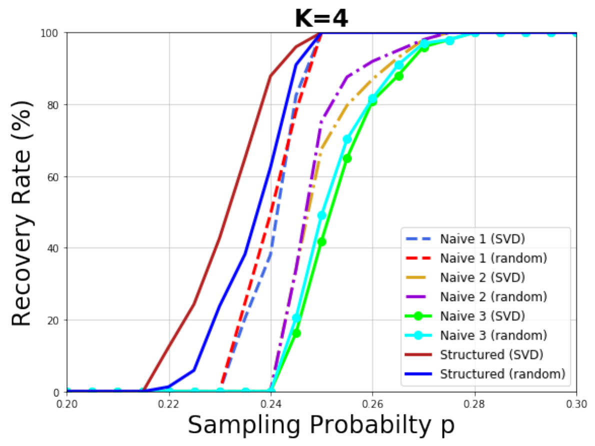
1. Thirdly, for all three naive methods, random initialization leads to better performance than the SVD-based initialization; whereas for the structured approach, SVD-based initialization performs better.



(a) $K = 2$.



(b) $K = 3$.



(c) $K = 4$.

Fig. 6: Recovery rate performances for noiseless matrices with $K = 2, 3, 4$.

Finally, we show the average running time comparisons among different algorithms in Fig. 7 for $K = 4$ and $p = 0.3$. It is seen that the Naive method 3 is much slower than the other methods, since the matrix it processes has more and more samples over the later stages. Moreover, the Naive method 2 is the fastest due to the smaller sizes of the matrices it processes. The structured approach takes only slightly longer than the Naive method 1.

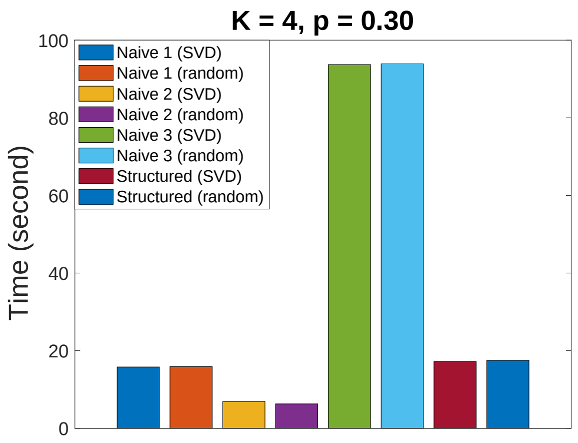


Fig. 7: Running time comparisons for noiseless matrices with $K = 4$ and $p = 0.3$.

2) Noisy Matrix: We now consider the case that the matrix to be completed is noisy, i.e., $\mathbf{Z} = \mathbf{U} + \mathbf{N} = \mathbf{XY} + \mathbf{N}$, where $\mathbf{X}$ and $\mathbf{Y}$ are generated the same way as described in Sec. IV.A; and the entries of $\mathbf{N}$ are i.i.d. $\mathcal{N}(0, \sigma^2)$ samples. We define the signal-to-noise-ratio as SNR $= 10\log_{10}\left(\frac{\frac{1}{mn_K}\sum_{i=1}^{m}\sum_{j=1}^{n_K}\mathbf{U}(i,j)^2}{\sigma^2}\right)$. Moreover, we define the signal-to-error-ratio for the recovered matrix $\hat{\mathbf{U}}$ as SER $= 10\log_{10}\left(\frac{\frac{1}{mn_K}\sum_{i=1}^{m}\sum_{j=1}^{n_K}\mathbf{U}(i,j)^2}{\frac{1}{mn_K}\sum_{i=1}^{m}\sum_{j=1}^{n_K}\left(\hat{\mathbf{U}}(i,j)-\mathbf{U}(i,j)\right)^2}\right)$. Each result of (SNR, SER) is the average of 100 realizations of $\mathbf{Z}$.

First, for $K = 4$, $p = 0.3$, SNR $=$ 10dB and a particular sampled matrix, we show the convergence behaviors of the structured approach and the Naive method 1 in Fig. 8. By comparing Fig. 8 and Fig. 5, we observe that for all methods, it takes more iterations to converge in the noisy case, but still the structured approach converges faster than the Naive method 1. Moreover, for the structured approach, the SVD initialization leads to faster convergence, whereas for the Naive method 1, random initialization converges faster.

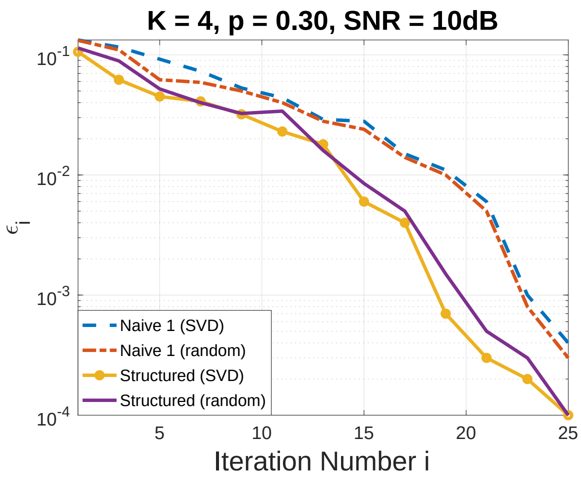
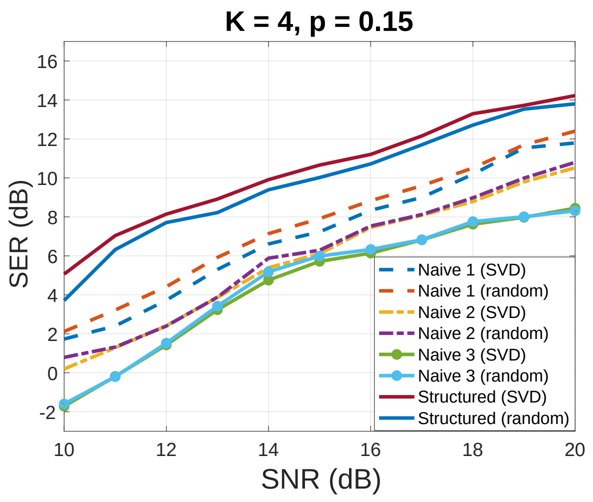


Fig. 8: Convergence comparison for noisy matrices with $K = 4$, $p = 0.15$, and SNR = 10dB.

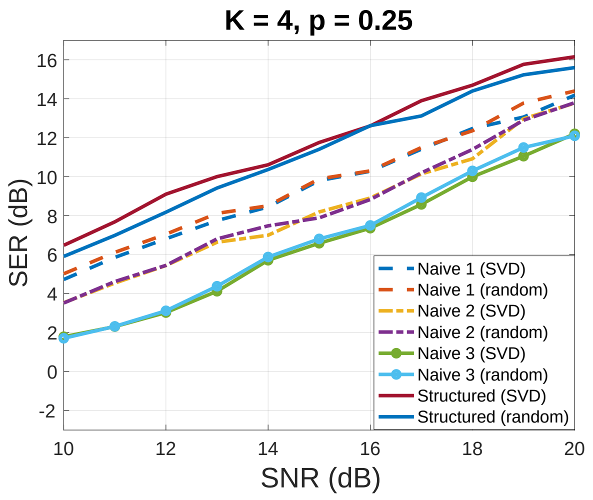
Next, the SER performance results are shown in Figs. 9(a) and 9(b), for $p = 0.15$ and $p = 0.25$, respectively. It is seen that among the naive methods, the Naive method 1 still performs the best in the noisy case. But now there is a significant gain in SER by the proposed structured approach over the naive methods. For example, at SNR = 12dB, for $p = 0.15$ and $p = 0.25$, the SER gains over the Naive 1 method is 3.8dB and 1.9dB, respectively. Moreover, similar to the noiseless case, the SVD-based initialization performs better for the structured approach whereas random initialization performs better for the naive methods.

B. *Tensor Case*

For the tensor case, we consider an example where $d = 4$, $K = 4$, $m_1 = m_2 = m_3 = 40$, $n_1 = 25$, $n_2 = 30$, $n_3 = 35$, $n_4 = 40$, $r_1 = 50$, $r_2 = 60$, $r_3 = 70$ and $r_4 = 80$. In order to generate a tensor that is randomly chosen from the manifold corresponding to the given rank constraints, we first generate $\mathbf{a}_j^l \in \mathbb{R}^{40}$ (structured columns) with entries being i.i.d. $\mathcal{N}(0,1)$ samples for $1 \le j \le (d-1)$ and $1 \le l \le r_K$. Then, we generate $\mathbf{a}_d^l \in \mathbb{R}^{40}$ such that it satisfies the structured decomposition given in Lemma 4,



(a) $p = 0.15$.



(b) $p = 0.25$.

Fig. 9: SER performances for noisy matrices with $K = 4$.

i.e., for any $k = 1,\ldots,K$, $x = n_{k-1}+1,\ldots,n_k$ and $l = r_k+1,\ldots,r_K$ we have $\mathbf{a}_d^l(x) = 0$, and the rest of the entries are i.i.d. $\mathcal{N}(0,1)$ samples. Therefore, the tensor $\mathcal{U} = \sum_{l=1}^{r} \mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \ldots \otimes \mathbf{a}_d^l$ satisfies all the rank constraints. Then, we sample the entries of $\mathcal{U}$ independently with probability $0 < p < 1$. The regularization weight is set as $\lambda = 0.01$. We define the convergence metric as $\epsilon_i = \frac{\|\mathcal{U}_i\|_{\mathcal{F}} - \|\mathcal{U}_{i-1}\|_{\mathcal{F}}}{\|\mathcal{U}_i\|_{\mathcal{F}}}$ (where $\mathcal{U}_i = \sum_{l=1}^{r_K} \mathbf{A}_1^{(i)}(:,l) \otimes \ldots \otimes \mathbf{A}_d^{(i)}(:,l)$) and convergence is reached if $\epsilon_i < 10^{-3}$. On the other hand, divergence is declared if $\|\mathbf{A}_j^{(i)}\|_{\mathcal{F}} > 10^6 \|\mathbf{A}_j^{(0)}\|_{\mathcal{F}}$, for any $j \in \{1,\ldots,d\}$.

1) Noiseless Tensor: We say the sampled tensor $\mathcal{U}$ is recovered if the algorithm converges and the normalized error satisfies $\frac{\|\hat{\mathcal{U}}-\mathcal{U}\|_{\mathcal{F}}}{\|\mathcal{U}\|_{\mathcal{F}}} < 0.01$, where $\hat{\mathcal{U}}$ denotes the completed tensor. Similar to the matrix case, we consider different number of rank constraints: $K = 2, 3$ and 4. For each case and a given sampling probability p, we run different completion algorithms on 100 random tensors from the corresponding manifold and calculate the recovery rates.

In Fig. 10 we illustrate the convergence behaviors of the Naive method 1 and the structured approach, for $K = 4$, $p = 0.2$ and a particular sampled tensor. It is seen that the convergence condition $\epsilon_i < 10^{-3}$ is reached after 17 and 21 iterations for the structured approach and the Naive 1 method, respectively.

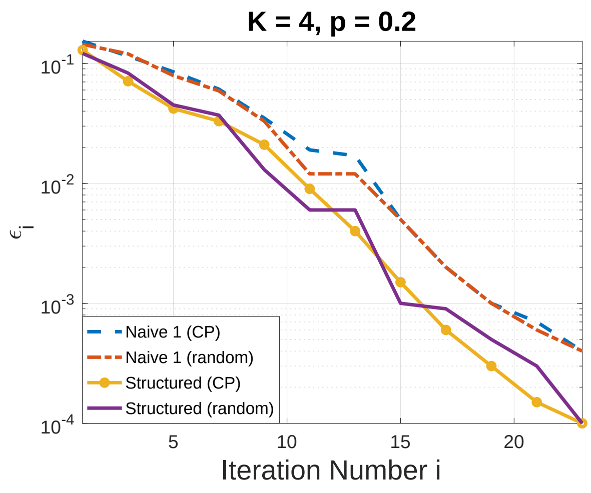
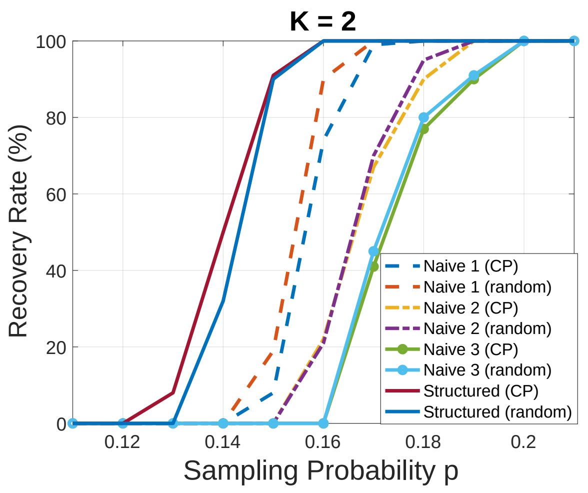


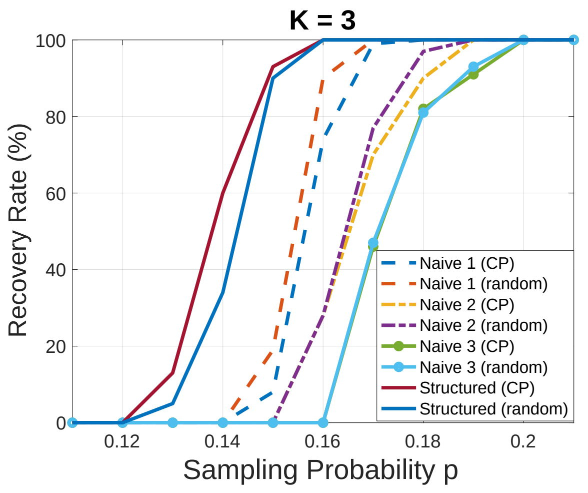
Fig. 10: Convergence comparison for noiseless tensors with $K = 4$ and $p = 0.2$.

Figs. 11(a), 11(b), and 11(c) show the recovery rate performances of different algorithms for $K = 2, 3,$ and 4, respectively. Similarly as in the matrix case, Naive 1 has the best recovery performance among the three naive methods. Compared with Naive 1, the structured approach mainly improves the region where the recovery rate is below 1, i.e., $p \in (0.12, 0.15)$. And, for all three naive methods, random initialization perform better than the CP-based initialization; whereas for the structured approach, CP-based initialization is better.

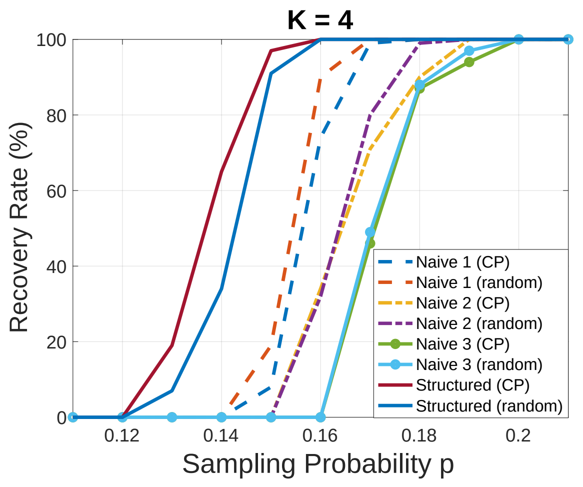
Fig. 12 shows the average running time comparisons among



(a) $K = 2$.



(b) $K = 3$.



(c) $K = 4$.

Fig. 11: Recovery rate performances for noiseless tensors with $K = 2, 3, 4$.

different algorithms for $K = 4$ and $p = 0.2$. Similar to the matrix case, the Naive method 3 is the slowest and the Naive method 2 is the fastest. The structured approach is slightly slower than the Naive method 1.

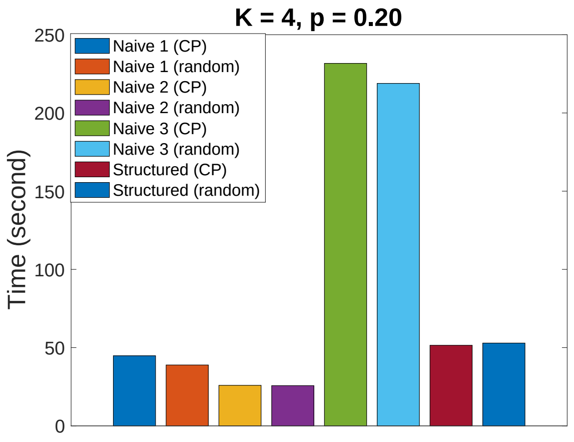


Fig. 12: Running time comparisons for noiseless tensors with $K = 4$ and $p = 0.2$.

2) Noisy Tensor: We now consider the noisy case, i.e., $\mathcal{Z} = \mathcal{U} + \mathcal{N}$, where $\mathcal{U}$ is generated the same way as described in Sec. IV.B; and the entries of $\mathcal{N}$ are i.i.d. $\mathcal{N}(0, \sigma^2)$ samples. We define the signal-to-noise-ratio as SNR $= 10\log_{10}\left(\frac{\frac{1}{m_1\cdots n_d}\sum_{x_1=1}^{m_1}\cdots\sum_{x_d=1}^{n_d}\mathcal{U}(x_1,\dots,x_d)^2}{\sigma^2}\right)$. Moreover, we define the signal-to-error-ratio for the recovered tensor $\hat{\mathcal{U}}$ as SER $= 10\log_{10}\left(\frac{\frac{1}{m_1\cdots n_d}\sum_{x_1=1}^{m_1}\cdots\sum_{x_d=1}^{n_d}\mathcal{U}(x_1,\dots,x_d)^2}{\frac{1}{m_1\cdots n_d}\sum_{x_1=1}^{m_1}\cdots\sum_{x_d=1}^{n_d}\left(\hat{\mathcal{U}}(x_1,\dots,x_d)-\mathcal{U}(x_1,\dots,x_d)\right)^2}\right)$. Each result of (SNR, SER) is the average of 100 realizations of $\mathcal{Z}$.

Fig. 13 shows the convergence behaviors of the structured approach and the Naive method 1 for $K = 4$, $p = 0.2$, SNR $=$ 10dB and a particular sampled tensor. Similar to the matrix case, it takes more iterations to converge in the noisy case for all methods, and the structured approach converges faster. Moreover, for the structured approach, the CP initialization leads to faster convergence, whereas for the Naive method 1, random initialization converges faster.

Figs. 14(a) and 14(b) show the SER performances for $p = 0.05$ and $p = 0.15$, respectively. It is seen that there is a significant gain in SER by the proposed structured approach

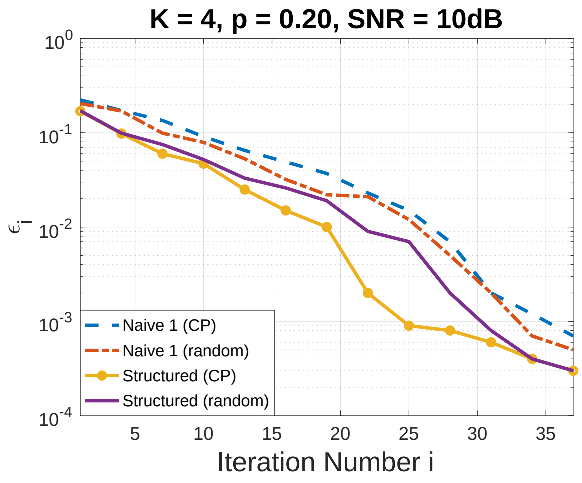
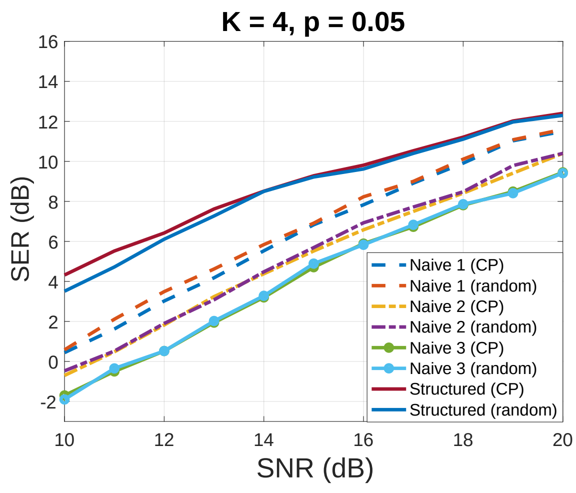


Fig. 13: Convergence comparison for noisy tensors with $K = 4$, $p = 0.15$, and SNR = 10dB.

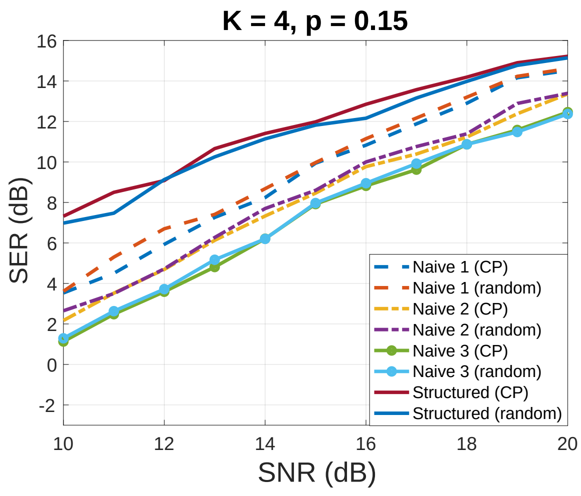
over the naive methods. For example, at SNR $=$ 12dB, for $p = 0.05$ and $p = 0.15$, the SER gains over the Naive (which performs the best among naive methods) method 1 is 2.9dB and 2dB, respectively. Moreover, similar to the noiseless case, the CP-based initialization performs better for the structured approach whereas random initialization performs better for the naive methods.

V. Conclusions

In this paper, we have developed a structured alternating minimization approach to data completion where the data has a union of nested subspaces structure with multiple known rank constraints. Both matrix and tensor cases are studied. Our key observation is that the union of nested subspaces structure leads to a structured decomposition where some factors ($\mathbf{Y}$ for matrix case and $\mathbf{A}_d$ for tensor case) contain blocks of zeros determined by the rank values. The proposed structured alternating minimization algorithms for both matrix and tensor completion enforce such structures in each iteration including the initialization. Simulation results show that compared with naive methods, the proposed structured approaches achieve faster convergence and higher recovery accuracy, especially for noisy data completion.



(a) $p = 0.05$.



(b) $p = 0.15$.

Fig. 14: SER performances for noisy tensors with $K = 4$.

REFERENCES

[1] L. T. Nguyen, J. Kim, and B. Shim, "Low-rank matrix completion: A contemporary survey," *IEEE Access*, vol. 7, pp. 215–237, 2019.

[2] L.-H. Lim and P. Comon, "Multiarray signal processing: Tensor decomposition meets compressed sensing," *Comptes Rendus Mecanique*, vol. 338, no. 6, pp. 311–320, 2010.

[3] N. D. Sidiropoulos and A. Kyrillidis, "Multi-way compressed sensing for sparse low-rank tensors," *IEEE Signal Processing Letters*, vol. 19, no. 11, pp. 757–760, 2012.

[4] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, pp. 1–19, 2011.

[5] Y. Yu, J. Peng, and S. Yue, "A new nonconvex approach to low-rank matrix completion with application to image inpainting," *Multidimensional Systems and Signal Processing*, vol. 30, no. 1, pp. 145–174, 2019.

[6] N. J. Harvey, D. R. Karger, and K. Murota, "Deterministic network coding by matrix completion," in *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2005, pp. 489–498.

[7] E. Candès, Y. C. Eldar, T. Strohmer, and V. Voroninski, "Phase retrieval via matrix completion," *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 199–225, 2013.

[8] H. Ji, C. Liu, Z. Shen, and Y. Xu, "Robust video denoising using low rank matrix completion." in *Proceedings of 2010 Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1791–1798.

[9] L. Eldén, *Matrix Methods in Data Mining and Pattern Recognition*. SIAM, 2007, vol. 4.

[10] D. Kressner, M. Steinlechner, and B. Vandereycken, "Low-rank tensor completion by Riemannian optimization," *BIT Numerical Mathematics*, vol. 54, no. 2, pp. 447–468, 2014.

[11] C. Zhang, H. Fu, S. Liu, G. Liu, and X. Cao, "Low-rank tensor constrained multiview subspace clustering," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1582–1590.

[12] A. C. Sauve, A. O. Hero, W. L. Rogers, S. J. Wilderman, and N. H. Clinthorne, "3D image reconstruction for a Compton SPECT camera model," *IEEE Transactions on Nuclear Science*, vol. 46, no. 6, pp. 2075–2084, 1999.

[13] R. Madbhavi, H. S. Karimi, B. Natarajan, and B. Srinivasan, "Tensor completion based state estimation in distribution systems," *researchgate.net*, 2020.

[14] M. Qin, Z. Li, S. Chen, Q. Guan, and J. Zheng, "Low-rank tensor completion and total variation minimization for color image inpainting," *IEEE Access*, vol. 8, pp. 53 049–53 061, 2020.

[15] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video inpainting under constrained camera motion," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 545–553, 2007.

[16] N. Li and B. Li, "Tensor completion for on-board compression of hyperspectral images," in *Proceedings of 2010 IEEE International Conference on Image Processing*. IEEE, 2010, pp. 517–520.

[17] T. G. Kolda, B. W. Bader, and J. P. Kenny, "Higher-order web link analysis using multilinear algebra," in *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05)*, 2005.

[18] Q. Song, H. Ge, J. Caverlee, and X. Hu, "Tensor completion algorithms in big data analytics," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 1, pp. 1–48, 2019.

[19] E. Candès and B. Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 55, no. 6, pp. 111–119, 2012.

[20] ——, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[21] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion

using alternating minimization," in *Proceedings of 2013 Annual Symposium on the Theory of Computing*, 2013, pp. 665–674.

[22] R. Sun and Z.-Q. Luo, "Guaranteed matrix completion via non-convex factorization," *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 6535–6579, 2016.

[23] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[24] Y. Song, J. Li, X. Chen, D. Zhang, Q. Tang, and K. Yang, "An efficient tensor completion method via truncated nuclear norm," *Journal of Visual Communication and Image Representation*, 2020.

[25] C. Liu, H. Shan, and C. Chen, "Tensor p-shrinkage nuclear norm for low-rank tensor completion," *Neurocomputing*, vol. 387, pp. 255–267, 2020.

[26] Z. Zhang and S. Aeron, "Exact tensor completion using t-SVD," *IEEE Transactions on Signal Processing*, vol. 65, no. 6, pp. 1511–1526, 2016.

[27] Y.-B. Zheng, T.-Z. Huang, T.-Y. Ji, X.-L. Zhao, T.-X. Jiang, and T.-H. Ma, "Low-rank tensor completion via smooth matrix factorization," *Applied Mathematical Modelling*, vol. 70, pp. 677–695, 2019.

[28] X.-Y. Liu, S. Aeron, V. Aggarwal, and X. Wang, "Low-tubal-rank tensor completion using alternating minimization," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1714–1737, 2019.

[29] W. Wang, V. Aggarwal, and S. Aeron, "Tensor completion by alternating minimization under the tensor train (TT) model," *arXiv preprint arXiv:1609.05587*, 2016.

[30] M. Ashraphijuo, X. Wang, and V. Aggarwal, "Rank determination for low-rank data completion," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1–29, 2017.

[31] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *Proceedings of 2011 IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 1615–1622.

[32] S. Rao, R. Tron, R. Vidal, and Y. Ma, "Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1832–1845, 2009.

[33] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy data coding and compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 9, pp. 1546–1562, 2007.

[34] C. Zhang and R. R. Bitmead, "Subspace system identification for training-based MIMO channel estimation," *Automatica*, vol. 41, no. 9, pp. 1623–1632, 2005.

[35] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, "Multi-task low-rank affinity pursuit for image segmentation," in *Proceedings of 2011 IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 2439–2446.

[36] S. R. Rao, R. Tron, R. Vidal, and Y. Ma, "Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[37] A. Y. Yang, S. R. Rao, and Y. Ma, "Robust statistical estimation and segmentation of multiple subspaces," in *Proceedings of 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*. IEEE, 2006.

[38] J. P. Costeira and T. Kanade, "A multibody factorization method for independently moving objects," *International Journal of Computer Vision*, vol. 29, no. 3, pp. 159–179, 1998.

[39] C. W. Gear, "Multibody grouping from motion images," *International Journal of Computer Vision*, vol. 29, no. 2, pp. 133–150, 1998.

[40] Y. Wu, Z. Zhang, T. S. Huang, and J. Y. Lin, "Multibody grouping via orthogonal subspace decomposition," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 2. IEEE, 2001.

[41] T. Zhang, A. Szlam, and G. Lerman, "Median k-flats for hybrid linear modeling with many outliers," in *Proceedings of 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. IEEE, 2009, pp. 234–241.

[42] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 663–670.

[43] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *Proceedings of 2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1615–1622.

[44] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (GPCA)," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1945–1959, 2005.

[45] M. Ashraphijuo and X. Wang, "Clustering a union of low-rank subspaces of different dimensions with missing data," *Pattern Recognition Letters*, vol. 120, pp. 31–35, 2019.

[46] ——, "Union of low-rank tensor spaces: Clustering and completion," *Journal of Machine Learning Research*, vol. 21 (69), pp. 1–36, 2020.

[47] ——, "Fundamental conditions on the sampling pattern for union of low-rank subspaces retrieval," *Annals of Mathematics and Artificial Intelligence*, vol. 87, no. 4, pp. 373–393, 2019.