Efficient Low Rank Tensor Ring Completion

Wenqi Wang, Vaneet Aggarwal
Purdue University, West Lafayette, IN
{wang2041, vaneet}@purdue.edu

Shuchin Aeron
Tufts University, Medford, MA
shuchin@ece.tufts.edu

Abstract

Using the matrix product state (MPS) representation of the recently proposed tensor ring (TR) decompositions, in this paper we propose a TR completion algorithm, which is an alternating minimization algorithm that alternates over the factors in the MPS representation. This development is motivated in part by the success of matrix completion algorithms that alternate over the (low-rank) factors. We propose a novel initialization method and analyze the computational complexity of the TR completion algorithm. The numerical comparison between the TR completion algorithm and the existing algorithms that employ a low rank tensor train (TT) approximation for data completion shows that our method outperforms the existing ones for a variety of real computer vision settings, and thus demonstrates the improved expressive power of tensor ring as compared to tensor train.

1. Introduction

Tensor decompositions for representing and storing data have recently attracted considerable attention due to their effectiveness in compressing data for statistical signal processing [11, 5, 17, 13, 3]. In this paper we focus on Tensor Ring (TR) decomposition [18] and in particular its relation to Matrix Product States (MPS) [14] representation for tensor and use it for completing data from missing entries. In this context our algorithm is motivated by recent work in matrix completion where under a suitable initialization an alternating minimization algorithm [10, 8] over the low rank factors is able to accurately predict the missing data.

Recently, tensor networks, considered as the generalization of tensor decompositions, have emerged as the potentially powerful tools for analysis of large-scale tensor data [14]. The most popular tensor network is the Tensor Train (TT) representation, which for an order-d tensor with each dimension of size n requires $O(dnr^2)$ parameters, where r is the rank of each of the factors, and thus allows for the efficient data representation [15]. Tensor train decompositions have been recently considered in [7, 16] and the authors in [7, 16] considered the completion of data via an alternating least square method.

Although TT format has been widely applied in numerical analysis, its applications to image classification and completion are rather limited [13, 7, 16]. As outlined in [18], TT decomposition suffers from the following limitations. Namely, (i) TT model requires rank-1 constraints to the border factors, (ii) TT ranks are typically small for near-border factors and large for the middle factors, and (iii) the multiplications of the TT factors are not permutation invariant. In order to alleviate those drawbacks, a tensor ring (TR) decomposition has been proposed in [18]. TR decomposition removes the unit rank constraints for the boundary tensor factors and utilizes a trace operation in the decomposition. The multilinear products between factors also have no strict ordering and the factors can be circularly shifted due to the properties of the trace operation. This paper provides novel algorithms for data completion when the data is modeled as a TR decomposition.

For data completion using tensor decompositions, one of the key attributes is the notion of the rank. Even though the rank in TR is a vector, we can assume all ranks to be the same, unlike that in TT case where the intermediate ranks are higher, thus providing a single parameter that can be tuned based on the data and the number of samples available. The use of trace operation in the tensor ring structure brings challenges for completion as compared to that for tensor train decomposition. The tensor ring structure is equivalent to a cyclic structure in tensor networks [10], and understanding this structure can help understand completion for more general tensor networks. In this paper, we propose an alternating minimization algorithm for the tensor ring completion. The initialization of the algorithm is an extension of TT approximation algorithm in [15] after zero-filling the missing data. Further, all the sub-problems in alternating minimization are converted to efficient least square problems, thus significantly improving the complexity of each sub-problem. We also analyze the storage and computational complexity of the proposed algorithm.

We note that, to the best of our knowledge, tensor ring completion has never been investigated for tensor comple-

tion, even though tensor ring factorization has been proposed in [18]. The different novelties as compared to [18] include the initialization algorithm, exclusion of tensor factor normalization, conversion of tensor completion problem into different least square sub-problems, and analysis of complexity in storage and computation.

The proposed algorithm is evaluated on a variety of datasets, including Einstein's image, Extended YaleFace Dataset B, and high speed video. The results are compared with the tensor train completion algorithms in [7, 16], and the additional structure in the tensor ring is shown to significantly improve the performance as compared to using the TT structure.

The rest of the paper is organized as follows. In section 2 we introduce the basic notation and preliminaries on the TR decomposition. In section 3 we outline the problem statement and propose the main algorithm. We also describe the computational complexity of the proposed algorithm. Following that we test the algorithm extensively against competing methods on a number of real and synthetic data experiments in section 4. Finally we provide conclusion and future research directions in section 5.

2. Notation & Preliminaries

In this paper, vector and matrices are represented by bold face lower case letters $(\mathbf{x}, \mathbf{y}, \mathbf{z}, \cdots)$ and bold face capital letters $(\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \cdots)$ respectively. A tensor with order more than two is represented by calligraphic letters $(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$. For example, an n^{th} order tensor is represented by $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, where $I_{i:i=1,2,\cdots,n}$ is the tensor dimension along mode i. The tensor dimension along mode i could be an expression, where the expression inside () is evaluated as a scalar, e.g. $\mathcal{X} \in \mathbb{R}^{(I_1 I_2) \times (I_3 I_4) \times (I_5 I_6)}$ represents a 3-mode tensor where dimensions along each mode is $I_1 I_2$, $I_3 I_4$, and $I_5 I_6$ respectively. An entry inside a tensor $\mathcal{X}$ is represented as $\mathcal{X}(i_1, i_2, \cdots, i_n)$, where $i_{k:k=1,2,..,n}$ is the location index along the k^{th} mode. A colon is applied to represent all the elements of a mode in a tensor, e.g. $\mathcal{X}(:, i_2, \cdots, i_n)$ represents the fiber along mode 1 and $\mathcal{X}(:, :, i_3, i_4, \cdots, i_n)$ represents the slice along mode 1 and mode 2 and so forth. Similar to Hadamard product under matrices case, Hadamard product between tensors is the entry-wise product of the two tensors. vec$(\cdot)$ represents the vectorization of the tensor in the argument. The vectorization is carried out lexicographically over the index set, stacking the elements on top of each other in that order. Frobenius norm of a tensor is the same as the vector ℓ_2 norm of the corresponding tensor after vectorization, e.g. $\|\mathcal{X}\|_F = \|\text{vec}(\mathcal{X})\|_{\ell_2}$. $\times$ between matrices is the standard matrix product operation.

Definition 1. *(Mode-i unfolding [4]) Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$ be a n-mode tensor. Mode-i unfolding of $\mathcal{X}$, denoted as $\mathcal{X}_{[i]}$, matrized the tensor $\mathcal{X}$ by putting the i^{th} mode in the matrix rows and remaining modes with the original order in the columns such that*

$$\mathcal{X}_{[i]} \in \mathbb{R}^{I_i \times (I_1 \cdots I_{i-1} I_{i+1} \cdots I_n)}. \tag{1}$$

Definition 2. *(Left Unfolding and Right Unfolding [9]) Let $\mathcal{X} \in \mathbb{R}^{R_{i-1} \times I_i \times R_i}$ be a third order tensor, the left unfolding is the matrix obtained by taking the first two modes indices as rows indices and the third mode indices as column indices such that*

$$\mathbf{L}(\mathcal{X}) = (\mathcal{X}_{[3]})^T \in \mathbb{R}^{(R_{i-1} I_i) \times R_i}. \tag{2}$$

Similarly, the right unfolding gives

$$\mathbf{R}(\mathcal{X}) = \mathcal{X}_{[1]} \in \mathbb{R}^{R_{i-1} \times (I_i R_i)}. \tag{3}$$

Definition 3. *(Mode-i canonical matrization [4]) Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$ be an n^{th} order tensor, the mode-i canonical matrization gives*

$$\mathcal{X}_{<i>} \in \mathbb{R}^{(\prod_{t=1}^{i} I_t) \times (\prod_{t=i+1}^{n} I_t)}, \tag{4}$$

such that any entry in $\mathcal{X}_{<i>}$ satisfies

$$\begin{aligned} &\mathcal{X}_{<i>}(i_1 + (i_2 - 1) I_1 + \cdots + (i_k - 1) \prod_{t=1}^{k-1} I_t, \\ &i_{k+1} + (i_{k+2} - 1) I_{k+1} + \cdots + (i_n - 1) \prod_{t=k+1}^{n-1} I_t) \\ =&\mathcal{X}(i_1, \cdots, i_n). \end{aligned} \tag{5}$$

Definition 4. *(Tensor Ring [18]) Let $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$ be an n-order tensor with I_i-dimension along the i_{th} mode, then any entry inside the tensor, denoted as $\mathcal{X}(i_1, \cdots, i_n)$, is represented by*

$$\begin{aligned} \mathcal{X}(i_1, \cdots, i_n) = \sum_{r_1=1}^{R_1} \cdots \sum_{r_n=1}^{R_n} &\mathcal{U}_1(r_n, i_1, r_1) \cdots \\ &\mathcal{U}_n(r_{n-1}, i_n, r_n), \end{aligned} \tag{6}$$

where $\mathcal{U}_i \in \mathbb{R}^{R_{i-1} \times I_i \times R_i}$ is a set of 3-order tensors, also named matrix product states (MPS), which consist the bases of the tensor ring structures. Note that $\mathcal{U}_j(:, i_j, :) \in \mathbb{R}^{R_{j-1} \times 1 \times R_j}$ can be regarded as a matrix of size $\mathbb{R}^{R_{j-1} \times R_j}$, thus (6) *is equivalent to*

$$\mathcal{X}(i_1, \cdots, i_n) = tr(\mathcal{U}_1(:, i_1, :) \times \cdots \times \mathcal{U}_n(:, i_n, :)). \tag{7}$$

Remark 1. *(Tensor Ring Rank (TR-Rank)) In the formulation of tensor ring, we note that tensor ring rank is the vector $[R_1, \cdots, R_n]$. In general, R_i are not necessary to be the same. In our set-up, we set $R_i = R \; \forall i = 1, \cdots, n$, and* **the scalar R is referred as the tensor ring rank** *in the remainder of this paper.*

Remark 2. *(Tensor Train [15]) Tensor train is a special case of tensor ring when $R_n = 1$.*

Based on the formulation of tensor ring structure, we define a tensor connect product, the operation between the MPSs, to describe the generation of high order tensor $\mathcal{X}$ from the sets of MPSs $\mathcal{U}_{i:i=1,\cdots,n}$. Let $R_0 \triangleq R_n$ for ease of expressions.

Definition 5. *(Tensor Connect Product) Let $\mathcal{U}_i \in \mathbb{R}^{R_{i-1}\times I_i \times R_i}, i = 1, \cdots, n$ be n 3rd-order tensors, the tensor connect product between $\mathcal{U}_j$ and $\mathcal{U}_{j+1}$ is defined as,*

$$\begin{aligned}\mathcal{U}_j\mathcal{U}_{j+1} &\in \mathbb{R}^{R_{j-1}\times(I_jI_{j+1})\times R_{j+1}} \\ &= \textit{reshape}\left(\mathbf{L}(\mathcal{U}_j) \times \mathbf{R}(\mathcal{U}_{j+1})\right).\end{aligned} \tag{8}$$

Thus, the tensor connect product n MPSs is

$$\mathcal{U} = \mathcal{U}_1 \cdots \mathcal{U}_n \in \mathbb{R}^{R_0\times(I_1\cdots I_n)\times R_n}. \tag{9}$$

Tensor connect product gives the product rule for the production between 3-order tensors, just like the matrix product as for 2-order tensor. Under matrix case, $\mathcal{U}_j \in \mathbb{R}^{1\times I_j\times R_j}$, $\mathcal{U}_{j+1} \in \mathbb{R}^{R_j\times I_{j+1}\times 1}$. Thus tensor connect product gives the vectorized solution of matrix product.

We then define an operator f that applies on $\mathcal{U}$. Let $\mathcal{U} \in \mathbb{R}^{R_0\times(I_1\cdots I_n)\times R_n}$ be the 3-order tensor, $R_0 = R_n$, and let f be a reshaping operator function that reshapes a 3-order tensor $\mathcal{U}$ to a tensor of dimension $\mathcal{X}$ of dimension $\mathbb{R}^{I_1\times\cdots\times I_n}$, denoted as

$$\mathcal{X} = f(\mathcal{U}), \tag{10}$$

where $\mathcal{X}(i_1, \cdots, i_n)$ is generated by

$$\mathcal{X}(i_1, \cdots, i_n) = \text{tr}(\mathcal{U}(:, i_1+(i_2-1)I_1+\cdots+(i_n-1)I_{n-1}, :)). \tag{11}$$

Thus a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_n}$ with tensor ring structure is equivalent to

$$\mathcal{X} = f(\mathcal{U}_1 \cdots \mathcal{U}_n). \tag{12}$$

Similar to matrix transpose, which can be regarded as an operation that cyclic swaps the two modes for a 2-order tensor, we define a 'tensor permutation' to describe the cyclic permutation of the tensor modes for a higher order tensor.

Definition 6. *(Tensor Permutation) For any n-order tensor $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_n}$, the i^{th} tensor permutation is defined as $\mathcal{X}^{P_i} \in \mathbb{R}^{I_i\times I_{i+1}\times\cdots\times I_n\times I_1\times I_2\times\cdots\times I_{i-1}}$ such that $\forall_i, j_i \in [1, I_i]$*

$$\mathcal{X}^{P_i}(j_i, \cdots, j_n, j_1, \cdots, j_{i-1}) = \mathcal{X}(j_1, \cdots, j_n). \tag{13}$$

Then we have the following result.

Lemma 1. *If $\mathcal{X} = f(\mathcal{U}_1 \cdots \mathcal{U}_n)$, then $\mathcal{X}^{P_i} = f(\mathcal{U}_i\mathcal{U}_{i+1} \cdots \mathcal{U}_n\mathcal{U}_1 \cdots \mathcal{U}_{i-1})$.*

With this background and basic constructs, we now outline the main problem setup.

3. Formulation and Algorithm for Tensor Ring Completion

3.1. Problem Formulation

Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_n}$ that is partially observed at locations Ω, let $\mathcal{P}_\Omega \in \mathbb{R}^{I_1\times\cdots\times I_n}$ be the corresponding binary tensor in which 1 represents an observed entry and 0 represents a missing entry. The problem is to find a low tensor ring rank (TR-Rank) approximation of the tensor $\mathcal{X}$, denoted as $f(\mathcal{U}_1 \cdots \mathcal{U}_n)$, such that the recovered tensor matches $\mathcal{X}$ at $\mathcal{P}_\Omega$. This problem is referred as the tensor completion problem under tensor ring model, which is equivalent to the following problem

$$\min_{\mathcal{U}_{i:i=1,\cdots,n}} \|\mathcal{P}_\Omega \circ (f(\mathcal{U}_1 \cdots \mathcal{U}_n) - \mathcal{X})\|_F^2. \tag{14}$$

Note that the rank of the tensor ring R is predefined and the dimension of $\mathcal{U}_{i:i=1,\cdots,n}$ is $\mathbb{R}^{R\times I_i\times R}$.

To solve this problem, we propose an algorithm, referred as Tensor Ring completion by Alternating Least Square (TR-ALS) to solve the problem in two steps.

- Choose an initial starting point by using Tensor Ring Approximation (TRA). This initialization algorithm is detailed in Section 3.2.
- Update the solution by applying Alternating Least Square (ALS) that alternatively (in a cyclic order) estimates a factor say $\mathcal{U}_i$ keeping the other factors fixed. This algorithm is detailed in Section 3.3.

3.2. Tensor Ring Approximation (TRA)

A heuristic initialization algorithm, namely TRA, for solving (14) is proposed in this section. The proposed algorithm is a modified version of tensor train decomposition as proposed in [15]. We first perform a tensor train decomposition on the zero-filled data, where the rank is constrained by Singular Value Decomposition (SVD). Then, an approximation for the tensor ring is formed by extending the obtained factors to the desired dimensions by filling the remaining entries with small random numbers. We note that the small entries show faster convergence as compared to zero entries based on our considered small examples, and thus motivates the choice in the algorithm. Further, non-zero random entries help the algorithm initialize with larger ranks since the TT decomposition has the corner ranks as 1. Having non-zero entries can help the algorithm not getting stuck in a local optima of low corner rank. The TRA algorithm is given in Algorithm 1.

3.3. Alternating Least Square

The proposed tensor ring completion by alternating least square method (TR-ALS) solves (14) by solving the following problem for each i iteratively. The factors are initialized

Algorithm 1 Tensor Ring Approximation (TRA)

Input: Missing entry zero filled tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, TR-Rank R, small random variable depicting the standard deviation of the added normal random variable σ

Output: Tensor train decomposition $\mathcal{U}_{i:i=1,\cdots,n} \in \mathbb{R}^{R \times I_i \times R}$

1: Apply mode-1 canonical matricization for $\mathcal{X}$ and get matrix $\mathbf{X}_1 = \mathcal{X}_{<1>} \in \mathbb{R}^{I_1 \times (I_2 I_3 \cdots I_n)}$
2: Apply SVD and threshold the number of singular values to be $T_1 = \min(R, I_1, I_2 \cdots I_n)$, such that $\mathbf{X}_1 = \mathbf{U}_1 \mathbf{S}_1 \mathbf{V}_1^\top, \mathbf{U}_1 \in \mathbb{R}^{I_1 \times T_1}, \mathbf{S}_1 \in \mathbb{R}^{T_1 \times T_1}, \mathbf{V}_1 \in \mathbb{R}^{T_1 \times (I_2 I_3 \cdots I_n)}$. Reshape $\mathbf{U}_1$ to $\mathbb{R}^{1 \times I_1 \times T_1}$ and extend it to $\mathcal{U}_1 \in \mathbb{R}^{R \times I_1 \times R}$ by filling the extended entries by random normal distributed values sampled from $\mathcal{N}(0, \sigma^2)$.
3: Let $\mathbf{M}_1 = \mathbf{S}_1 \mathbf{V}_1^\top \in \mathbb{R}^{T_1 \times (I_2 I_3 \cdots I_n)}$.
4: **for** $i = 2$ to $n-1$ **do**
5: Reshape $\mathbf{M}_{i-1}$ to $\mathbf{X}_i \in \mathbb{R}^{(T_{i-1} I_i) \times (I_{i+1} I_{i+2} \cdots I_n)}$.
6: Compute SVD and threshold the number of singular values to be $T_i = min(R, T_{i-1} I_i, I_{i+1} \cdots I_n)$, such that $\mathbf{X}_i = \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top, \mathbf{U}_i \in \mathbb{R}^{(T_{i-1} I_i) \times T_i}, \mathbf{S}_i \in \mathbb{R}^{T_i \times T_i}, \mathbf{V} \in \mathbb{R}^{T_i \times (I_{i+1} I_{i+2} \cdots I_n)}$. Reshape $\mathbf{U}_i$ to $\mathbb{R}^{T_{i-1} \times I_i \times T_i}$ and extend it to $\mathcal{U}_i \in \mathbb{R}^{R \times I_i \times R}$ by filling the extended entries by random normal distributed values sampled from $\mathcal{N}(0, \sigma^2)$.
7: Set $\mathbf{M}_i = \mathbf{S}_i \mathbf{V}_i^\top \in \mathbb{R}^{T_i \times (I_{i+1} I_{i+2} \cdots I_n)}$
8: **end for**
9: Reshape $\mathbf{M}_{n-1} \in \mathbb{R}^{T_{n-1} \times I_n}$ to $\mathbb{R}^{T_{n-1} \times I_n \times 1}$, and extend it to $\mathcal{U}_n \in \mathbb{R}^{R \times I_n \times R}$ by filling the extended entries by random normal distributed values sampled from $\mathcal{N}(0, \sigma^2)$ to get $\mathcal{U}_n$
10: Return $\mathcal{U}_1, \cdots, \mathcal{U}_n$

from the TRA algorithm presented in the previous section.

$$\mathcal{U}_i = \text{argmin}_{\mathcal{Y}} \|\mathcal{P}_\Omega \circ f(\mathcal{U}_1 \cdots \mathcal{U}_{i-1} \mathcal{Y} \mathcal{U}_{i+1} \cdots \mathcal{U}_n) - \mathcal{X}_\Omega)\|_F^2. \tag{15}$$

Lemma 2. *When $i \neq 1$, solving*

$$\mathcal{U}_i = \textit{argmin}_{\mathcal{Y}} \|\mathcal{P}_\Omega \circ f(\mathcal{U}_1 \cdots \mathcal{U}_{i-1} \mathcal{Y} \mathcal{U}_{i+1} \cdots \mathcal{U}_n) - \mathcal{X}_\Omega)\|_F^2 \tag{16}$$

is equivalent to

$$\mathcal{U}_i = \textit{argmin}_{\mathcal{Y}} \|\mathcal{P}_\Omega^{P_i} \circ f(\mathcal{Y} \mathcal{U}_{i+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{i-1}) - \mathcal{X}_\Omega^{P_i}\|_F^2. \tag{17}$$

Since the format of (17) is exactly the same for each i when the other factors are known, it is enough to describe solving a single $\mathcal{U}_k$ without loss of generality. Based on Lemma 2, we need to solve the following problem.

$$\mathcal{U}_k = \text{argmin}_{\mathcal{Y}} \|\mathcal{P}_\Omega^{P_k} \circ f(\mathcal{Y} \mathcal{U}_{k+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{k-1}) - \mathcal{X}_\Omega^{P_k}\|_F^2. \tag{18}$$

We further apply mode-k unfolding, which gives the equivalent problem

$$\mathcal{U}_k = \text{argmin}_{\mathcal{Y}} \|\mathcal{P}_{\Omega\ [k]}^{P_k} \circ f(\mathcal{Y} \mathcal{U}_{k+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{k-1})_{[k]} - \mathcal{X}_{\Omega\ [k]}^{P_k}\|_F^2, \tag{19}$$

where $\mathcal{P}_{\Omega\ [k]}^{P_k}$, $f(\mathcal{Y} \mathcal{U}_{k+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{k-1})_{[k]}$ and $\mathcal{X}_{\Omega\ [k]}^{P_k}$ are matrices with dimension $\mathbb{R}^{I_k \times (I_{k+1} \cdots I_n I_1 \cdots I_{k-1})}$.

The trick in solving (19) is that each slice of tensor $\mathcal{Y}$, denoted as $\mathcal{Y}(:, i_k, :), i_k \in \{1, \cdots, I_k\}$ which corresponds to each row of $\mathcal{P}_{\Omega\ [k]}^{P_k}$, $f(\mathcal{Y} \mathcal{U}_{k+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{k-1})_{[k]}$ and $\mathcal{X}_{\Omega\ [k]}^{P_k}$, can be solved independently, thus equation (19) can be solved by solving I_k equivalent subproblems

$$\mathcal{U}_k(:, i_k, :) = \text{argmin}_{\mathcal{Z} \in \mathbb{R}^{R \times 1 \times R}} \|\mathcal{P}_{\Omega\ [k]}^{P_k}(i_k, :) \circ f(\mathcal{Z} \mathcal{U}_{k+1} \cdots \mathcal{U}_{k-1}) - \mathcal{X}_{\Omega\ [k]}^{P_k}(i_k, :)\|_F^2. \tag{20}$$

Let $\mathcal{B}^{(k)} = \mathcal{U}_{k+1} \cdots \mathcal{U}_n \mathcal{U}_1 \cdots \mathcal{U}_{k-1} \in \mathbb{R}^{R \times (I_{k+1} \cdots I_n I_1 \cdots I_{k-1}) \times R}$, Ω_{i_k} be the observed entries in vector $\mathcal{X}_{[k]}(i_k, :)$, thus $\mathcal{B}_{\Omega_{i_k}}^{(k)} \in \mathbb{R}^{R \times (I_{k+1} \cdots I_n I_1 \cdots I_{k-1})_{\Omega_{i_k}} \times R}$ are the components in $\mathcal{B}^{(k)}$ such that $\mathcal{P}_{\Omega\ [k]}^{P_k}(i_k, (I_{k+1} \cdots I_n I_1 \cdots I_{k-1})_{\Omega_{i_k}})$ are observed. Thus equation (20) is equivalent to

$$\mathcal{U}_k(:, i_k, :) = \text{argmin}_{\mathcal{Z}} \|f(\mathcal{Z} \mathcal{B}_{\Omega_{i_k}}^{(k)}) - \mathcal{X}_{\Omega\ [k]}^{P_k}(i_k, (I_{k+1} \cdots I_n I_1 \cdots I_{k-1})_{\Omega_{i_k}}))\|_F^2. \tag{21}$$

We regard $\mathcal{Z} \in \mathbb{R}^{R \times 1 \times R}$ as a matrix $\mathbf{Z} \in \mathbb{R}^{R \times R}$. Since the Frobenius norm of a vector in (21) is equivalent to entry-wise square summation of all entries, we rewrite (21) as

$$\mathcal{U}_k(:, i_k, :) = \text{argmin}_{\mathbf{Z} \in \mathbb{R}^{R \times R}} \sum_{j \in \Omega_{i_k}} \|\text{tr}(\mathbf{Z} \times \mathcal{B}_{\Omega_{i_k}}^{(k)}(:, j, :)) - \mathcal{X}_{\Omega\ [k]}^{P_k}(i_k, j)\|_F^2. \tag{22}$$

Lemma 3. *Let $\mathbf{A} \in \mathbb{R}^{r_1 \times r_2}$ and $\mathbf{B} \in \mathbb{R}^{r_2 \times r_1}$ be any two matrices, then*

$$\textit{Trace}(\mathbf{A} \times \mathbf{B}) = vec(\mathbf{B}^\top)^\top vec(\mathbf{A}). \tag{23}$$

Based on Lemma 3, (22) becomes

$$\mathcal{U}_k(:, i_k, :) = \text{argmin}_{\mathbf{Z}} \sum_{j \in \Omega_{i_k}^{(k)}} \|\text{vec}((\mathcal{B}_{\Omega_{i_k}}^{(k)}(:, j, :))^\top)^\top \text{vec}(\mathbf{Z}) - \mathcal{X}_{\Omega\ [k]}^{P_k}(i_k, j)\|_F^2. \tag{24}$$

Algorithm 2 TR-ALS Algorithm

Input: Zero-filled Tensor $\mathcal{X}_\Omega \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n}$, binary observation index tensor $\mathcal{P}_\Omega \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n}$, tensor ring rank R, thresholding parameter tot, maximum iteration $maxiter$

Output: Recovered tensor $\mathcal{X}_R$

1: Apply tensor ring approximation in Algorithm 1 on $\mathcal{X}_\Omega$ to initialize the MPSs $\mathcal{U}_{i:i=1,\cdots,n} \in \mathbb{R}^{R \times I_i \times R}$. Set iteration parameter $\ell = 0$.
2: **while** $\ell \leq maxiter$ **do**
3: $\ell = \ell + 1$
4: **for** $i = 1$ to n **do**
5: **Solve by Least Square Method** $\mathcal{U}_i^{(\ell)} = \text{argmin}_{\mathcal{U}} \|\mathcal{P}_\Omega \circ (\mathcal{U}\mathcal{U}_{i+1}^{(\ell-1)} \ldots \mathcal{U}_n^{(\ell-1)} \mathcal{U}_1^{(\ell)} \ldots \mathcal{U}_{i-1}^{(\ell)} - \mathcal{X})\|_F^2$
6: **end for**
7: **if** $\frac{\|\mathcal{U}_n^{(\ell+1)} - \mathcal{U}_n^{(\ell)}\|_F}{\|\mathcal{U}_n^{(\ell)}\|_F} \leq tot$ **then**
8: Break
9: **end if**
10: **end while**
11: Return $\mathcal{X}_R = \text{reshape}(\mathcal{U}_1^{(\ell)} \mathcal{U}_2^{(\ell)} \ldots \mathcal{U}_{n-1}^{(\ell)} \mathcal{U}_n^{(\ell)})$

Then the problem for solving $\mathcal{U}_k[:, i_k, :]$ becomes a least square problem. Solving I_k least square problem would give the optimal solution for $\mathcal{U}_k$. Since each $\mathcal{U}_{i:i=1,\cdots,n}$ can solved by a least square method, tensor completion under tensor ring model can be solved by taking orders to update $\mathcal{U}_{i:i=1,\cdots,n}$ until convergence. We note the completion algorithm does not require normalization on each MPS, unlike the decomposition algorithm [18] that normalizes all the MPSs to seek a unique factorization. The stopping criteria in TR-ALS is measured via the changes of the last tensor factors $\mathcal{U}_n$ since if the last factor does not change, the other factors are less likely to change. Details of the algorithm are given in Algorithm 2.

3.4. Complexity Analysis

Storage Complexity Given an n-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$, the total amount of parameters to store is $\prod_{i=1}^n I_i$, which increases exponentially with order. Under tensor ring model, we can reduce the storage space by converting each factor (except the last) one by one to being orthonormal and multiply the product with the next factor. Thus, the number of parameters to store the MPSs $\mathcal{U}_{i:i=1,\cdots,n-1}$ with orthonormal property requires storage $\sum_{i=1}^{n-1}(R^2 I_i - R^2)$, and $\mathcal{U}_n$ with parameter $R^2 I_n$. Thus, the total amount of storage is $R^2(\sum_i^n I_i - n + 1)$, where the tensor ring rank R can be adjusted to fit the tensor data at the desired accuracy.

Computational Complexity For each $\mathcal{U}_i$, the least square problem in (19) solved by pseudo-inverse gives a computational complexity $\max(O(PR^4), O(R^6))$, where P is the total number of observations. Within one iteration when n MPSs need to be updated, the overall complexity is $\max(O(nPR^4), O(nR^6))$.

We note that tensor train completion [7] gives the similar complexity as tensor ring completion. However, tensor train rank is a vector and it is hard for tuning to achieve the optimal completion. The intermediate ranks in tensor train are large in general, leading to significantly higher computational complexity of tensor train. This is alleviated in part by the tensor ring structure which can be parametrized by the tensor ring rank which can be smaller than the intermediate ranks of the tensor train in general. In addition, the single parameter in the tensor ring structure leads to an ease in characterizing the performance for different ranks and can be easily tuned for practical applications. The lower ranks lead to lower computational complexity of data completion under the tensor ring structure as compared to the tensor train structure.

4. Numerical Results

In this section, we compare our proposed TR-ALS algorithm with tensor train completion under alternating least square (TT-ALS) algorithm [7], which solves the tensor completion by alternating least squares under tensor train format. SiLRTC algorithm is another tensor train completion algorithm proposed in [16] and the tensor train rank is tuned based on the dimensionality of the tensor. It is selected for comparison as it shows good recovery in image completion [16]. The evaluation merit we consider is Recovery Error (RE). Let $\hat{\mathcal{X}}$ be the recovered tensor and $\mathcal{X}$ be the ground truth of the tensor. Thus, the recovery error is defined as

$$RE = \frac{\|\hat{\mathcal{X}} - \mathcal{X}\|_F}{\|\mathcal{X}\|_F}.$$

Tensor ring completion by alternating least square (TR-ALS) algorithm is an iterative algorithm and the maximum iteration, $maxiter$, is set to be 300. The convergence is captured by the change of the last factorization term $\mathcal{U}_n$, where the error tolerance is set to be 10^{-10}.

In the remaining of the section, we first evaluate the completion results for synthetic data. Then we validate the proposed TR-ALS algorithm on image completion, YaleFace image-sets completion, and video completion.

4.1. Synthetic Data

In this section, we consider a completion problem of a 4-order tensor $\mathcal{X} \in \mathbb{R}^{20 \times 20 \times 20 \times 20}$ with TR-Rank being 8 without loss of generality. The tensor is generated by a sequence of connected 3-rd order tensor $\mathcal{U}_{i:i=1,\cdots,4} \in \mathbb{R}^{8 \times 20 \times 8}$ and every entry in $\mathcal{U}_i$ are sampled independently from a standard normal distribution.

TT-ALS is considered as a comparable to show the difference between tensor train model and tensor ring model.

Two different tensor train ranks are chosen for the comparisons. The first tensor-train ranks are chosen as $[8, 8, 8]$, and the completion with these ranks is called Low rank tensor train (LR-TT) completion. The second tensor-train ranks are chosen as the double of the first ($[16, 16, 16]$), and the completion with these ranks is called High rank tensor train (HR-TT) completion. Another comparable used is the SiLRTC algorithm proposed in [16], where the rank is adjusted according to the dimensionality of the tensor data, and a heuristic factor of $f = 1$ in the proposed algorithm of [16] is selected for testing.

Fig.1a shows the completion error of TR-ALS, LR-TT, HR-TT, and SiLRTC for observation ratio from 10% to 60%. TR-ALS shows the lowest recovery error compared with other algorithms and the recovery error drops to 10^{-10} for observation ratio larger than 14%. The large completion errors of all tensor train algorithm at every observation ratio show that tensor train algorithm can not effectively complete the tensor data generated under tensor ring model. Fig. 1b shows the convergence of TR-ALS under sampling ratios $10\%, 15\%, 20\%, 30\%, 40\%,$ and 50%, and the plot indicates the higher the observation ratios, the faster the algorithm converges. When the observation ratio is lower than 10%, the tensor with missing data can not be completed under the proposed set-up. The fast convergence of the proposed TR-ALS algorithm indicates that alternating least square is effective in tensor ring completion.

4.2. Image Completion

In this section, we consider the completion of RGB Einstein Image [1], treated as a 3-order tensor $\mathcal{X} \in \mathbb{R}^{600\times600\times3}$. A reshaping operation is applied to transform the image into a 7-order tensor of size $\mathbb{R}^{6\times10\times10\times6\times10\times10\times3}$. Reshaping low order tensors into high order tensors is a common practice in literature and has shown improved performance in classification [13] and completion [16].

Fig. 2a shows the recovery error versus rank for TR-ALS and TT-ALS when the percentage of data observed are $5\%, 10\%, 20\%, 30\%$. At any considered ranks, TR-ALS completes the image with a better accuracy than TT-ALS. For any given percentage of observations, the recovery error first decreases as the rank increases which is caused by the increased information being captured by the increased number of parameters in the tensor structure. The recovery error then starts to increase after a thresholding rank, which can be ascribed to over-fitting. *The plot also indicates that higher the observation ratio, larger the thresholding rank, which to the best of our knowledge is reported for the first time*. Fig. 2b shows the recovered image of Einstein image when 10% pixels are randomly observed. TR-ALS with rank 28 gives the best recovery accuracy in the considered ranks.

4.3. YaleFace Dataset Completion

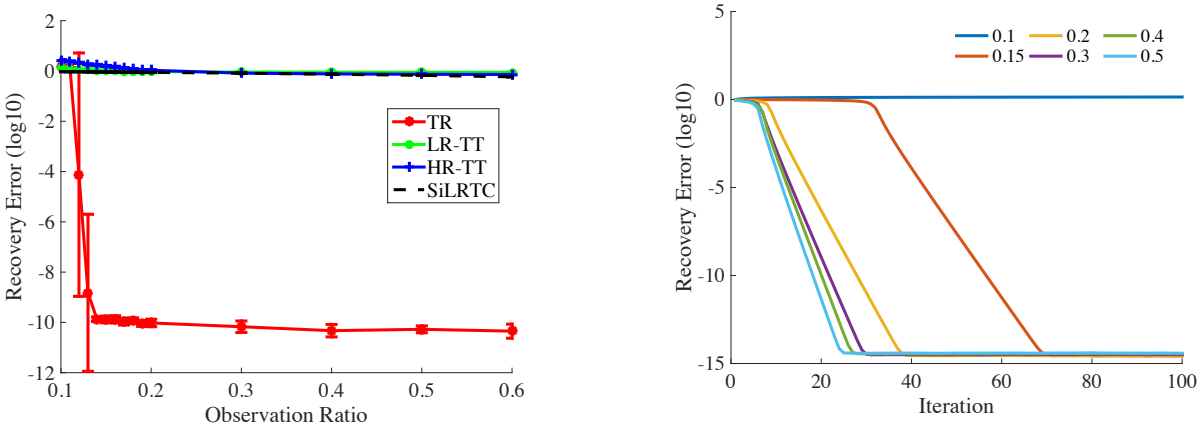
In this section, we consider Extended YaleFace Dataset B [6] that includes 38 people with 9 poses under 64 illumination conditions. Each image has the size of 192×168, where we down-sample the size of each image to 48×42 for ease of computation. We consider the image subsets of 38 people under 64 illumination with 1 pose by formatting the data into a 4-order tensor in $\mathbb{R}^{48\times42\times64\times38}$, which is further reshaped into a 8-order tensor $\mathcal{X} \in \mathbb{R}^{6\times8\times6\times7\times8\times8\times19\times2}$. We consider the case when 10% of pixels are randomly observed. YaleFace sets completion is considered to be harder than an image completion since features under different illumination and across human features are harder to learn than information from the color channels of images. Table 1 shows that for any considered rank, TR-ALS recovers data better than TT-ALS and the best completion result in the given set-up is 16.25% for TR-ALS as compared with 25.55% given by TT-ALS. Further we reshape the data into an 11-order tensor and 4-order tensor to evaluate the effect of reshaped tensor size on tensor completion. The result in Table 1 shows that in the given reshaping set-up, reshaping tensor from 4-order tensor to 7-th order tensor significantly improve the performance of tensor completion by decreasing recovery error from 21.48% to 16.25%. However, further reshaping to 11-order tensor slightly degrades the performance of completion, resulting in an increased recovery error to 16.34%.

Fig. 3 shows the original image, missing images, and recovered images using TR-ALS and TT-ALS algorithms for ranks of $10, 20,$ and 30, where the completion results given by TR-ALS better captures the detail information given from the image and recovers the image with a better resolution.

4.4. Video completion

The video data we used in this section is high speed camera video for gun shooting [2]. It is downloaded from Youtube with 85 frames in total and each frame is consisted by a $100 \times 260 \times 3$ image. Thus the video is a 4-order tensor of size $100 \times 260 \times 3 \times 85$, which is further reshaped into a 11-order tensor of size $5 \times 2 \times 5 \times 2 \times 13 \times 2 \times 5 \times 2 \times 3 \times 5 \times 17$ for completion. Video is a multi-dimensional data with different color channel a time dimension in addition to the 2D image structure.

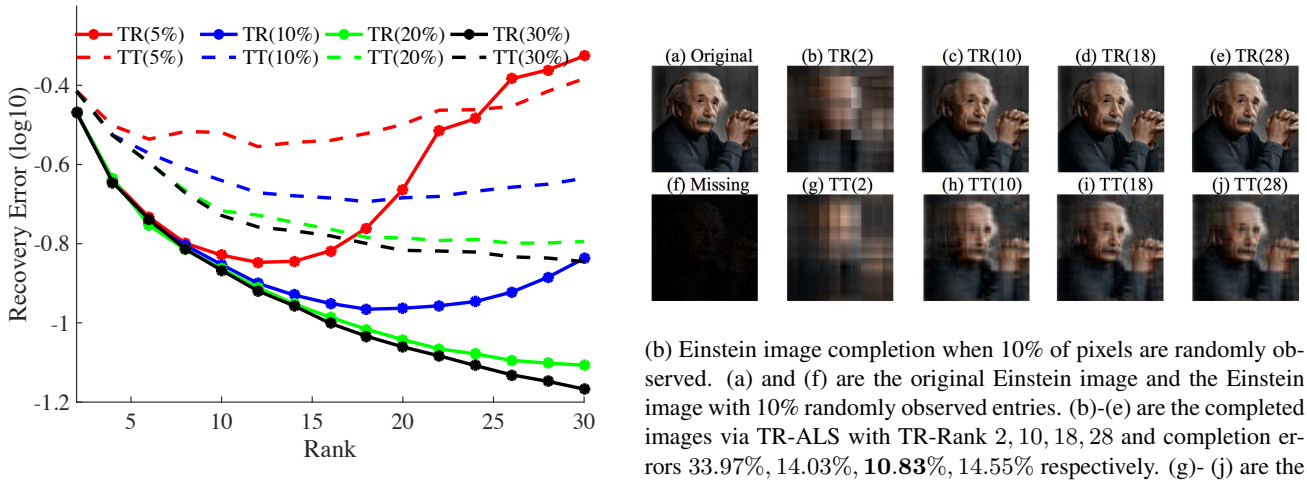
In Table 2, we show that TR-ALS achieves 6.25% recovery error when 10% of the pixels are observed, which is much better than the best recovery error of 14.83% achieved by TT-ALS. The first frame of the video is shown in Fig. 4, where the first row shows the original frame and the completed frames by TR-ALS, and the second row shows the frame with missing entries and the frames completed by TT-ALS. The resolution, and the display of the bullets and the smoke depict that the proposed TR-ALS achieves better



(a) Recovery error (log 10) versus observation Ratio. The plots are the average of 10 experiments and the error bar are marked using one standard deviation.

(b) Convergence plot for TR-ALS under observation ratio from 0.1 to 0.5

Figure 1: Completion for synthetic data. Synthetic data is a 4_{th} order tensor of dimension $20 \times 20 \times 20 \times 20$ with TR-Rank being 8.



(a) The recovery error versus rank for TR-ALS and TT-ALS under observation ratio $5\%, 10\%, 20\%, 30\%$.

(b) Einstein image completion when 10% of pixels are randomly observed. (a) and (f) are the original Einstein image and the Einstein image with 10% randomly observed entries. (b)-(e) are the completed images via TR-ALS with TR-Rank $2, 10, 18, 28$ and completion errors $33.97\%, 14.03\%, \mathbf{10.83}\%, 14.55\%$ respectively. (g)- (j) are the completed images via TT-ALS with TT-Rank $2, 10, 18, 28$ and completion errors $38.51\%, 22.89\%, \mathbf{20.70}\%, 23.19\%$ respectively.

Figure 2: Completion for Einstein image. Einstein image is of size $600 \times 600 \times 3$, and is reshaped into a 7-order tensor of size $6 \times 10 \times 10 \times 6 \times 10 \times 10 \times 3$ tensor for tensor ring completion

Rank	5	10	15	20	25	30
TT-ALS ($\mathbb{R}^{6\times8\times6\times7\times8\times8\times19\times2}$)	37.08%	29.65%	27.91%	26.84%	26.16%	25.55%
TR-ALS ($\mathbb{R}^{6\times8\times6\times7\times8\times8\times19\times2}$)	33.45%	**24.67%**	**20.72%**	**18.47%**	**16.92%**	**16.25%**
TR-ALS ($\mathbb{R}^{2\times3\times2\times4\times2\times3\times7\times8\times8\times19\times2}$)	33.73%	25.08%	21.20%	18.97%	17.34%	16.34%
TR-ALS ($\mathbb{R}^{48\times42\times64\times38}$)	**30.36%**	26.08%	23.74%	22.22%	21.48%	21.57%

Table 1: Completion error of 10% observed Extended YaleFace data via TT-ALS and TR-ALS under rank $5, 10, 15, 20, 25, 30$.

Rank	10	15	20	25	30
TT-ALS	19.16%	14.83%	16.42%	16.86%	16.99%
TR-ALS	13.90%	10.12%	8.13%	6.88%	**6.25%**

Table 2: Completion error of 10% observed Video data via TT-ALS and TR-ALS under rank $10, 15, 20, 25, 30$.

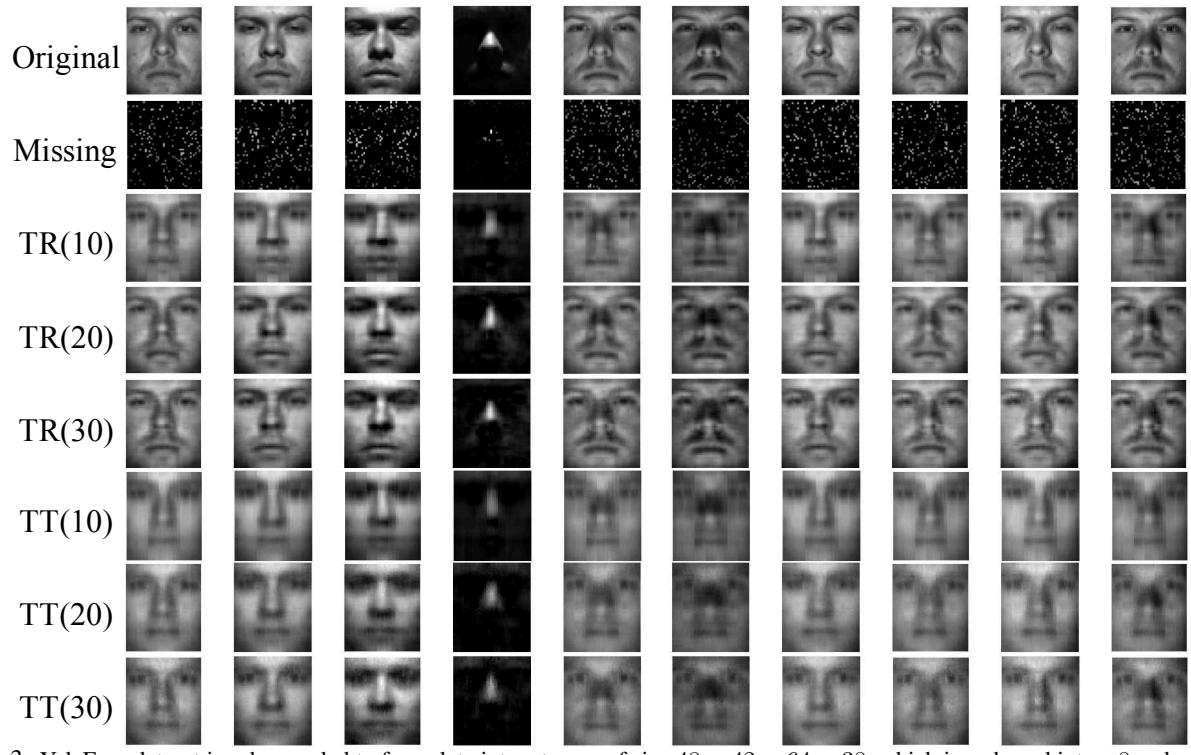


Figure 3: YaleFace dataset is sub-sampled to formulate into a tensor of size $48 \times 42 \times 64 \times 38$, which is reshaped into a 8-order tensor of size $6 \times 8 \times 6 \times 7 \times 8 \times 8 \times 19 \times 2$ for tensor ring completion. 90% of the pixels are assumed to be randomly missing. From top to bottom are original images, missing images, TR-ALS completed images with TR-Ranks $10, 20, 30$, and TT-ALS completed images with TT-Ranks $10, 20, 30$.

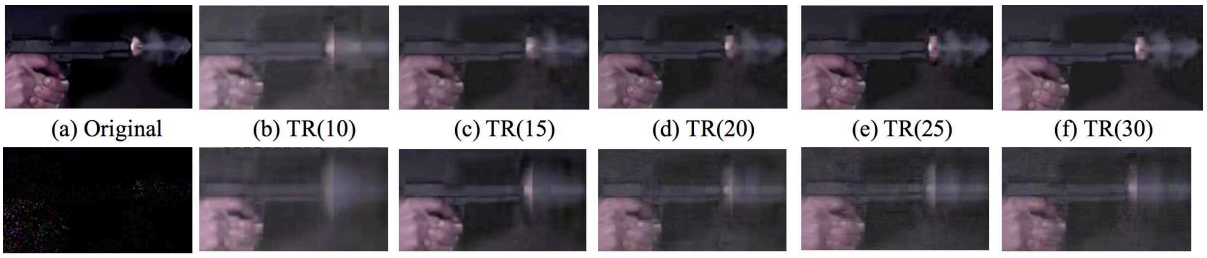


Figure 4: Gun Shot is a video of size $100 \times 260 \times 3 \times 80$ download from Youtube, which is reshaped into a 11-order tensor of size $5 \times 2 \times 5 \times 2 \times 13 \times 2 \times 5 \times 2 \times 3 \times 5 \times 17$ for tensor ring completion. 90% of the pixels are assumed to be randomly missing. (a) and (g) are the first frame of the original video and missing video. (b)-(f) are the completed frame via TR-ALS using TR-Rank $10, 15, 20, 25, 30$. (h)-(l) are the completed frame via TT-ALS using TR-Rank $10, 15, 20, 25, 30$.

completion results as compared to the TT-ALS algorithm.

5. Conclusion

We propose a novel algorithm for data completion using tensor ring decomposition. This is the first paper on data completion exploiting this structure which is a non-trivial extension of the tensor train structure. Our algorithm exploits the matrix product state representation and uses alternating minimization over the low rank factors for completion. The evaluation of the proposed approach on a variety of datasets, including Einstein's image, Extended YaleFace Dataset B, and video completion demonstrates the significant improvement of tensor ring completion as compared to tensor train completion.

Deriving provable performance guarantees on tensor completion using the proposed algorithm is left as further work. In this context, the statistical machinery for proving analogous results for the matrix case [10, 8] can be used.

Acknowledgements. This work was supported in part by the National Science Foundation under grants CCF 1527486, CCF 1553075, and CCF 1319653.

References

[1] Albert Einstein image. http://orig03.deviantart.net/7d28/f/2012/361/1/6/albert_einstein_by_zuzahin-d5pcbug.jpg.

[2] Pistol shot recorded at 73,000 frames per second. https://youtu.be/7y9apnbI6GA. Published by Discovery on 2015-08-15.

[3] M. Ashraphijuo, V. Aggarwal, and X. Wang. Deterministic and probabilistic conditions for finite completability of low rank tensor. *arXiv preprint arXiv:1612.01597*, 2016.

[4] A. Cichocki. Tensor networks for big data analytics and large-scale optimization problems. *arXiv preprint arXiv:1407.3124*, 2014.

[5] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, 2015.

[6] A. S. Georghiades and P. N. Belhumeur. Illumination cone models for faces recognition under variable lighting. In *Proceedings of CVPR*, 1998.

[7] L. Grasedyck, M. Kluge, and S. Kramer. Variants of alternating least squares tensor completion in the tensor train format. *SIAM Journal on Scientific Computing*, 37(5):A2424–A2450, 2015.

[8] M. Hardt. On the provable convergence of alternating minimization for matrix completion. *CoRR*, abs/1312.0925, 2013.

[9] S. Holtz, T. Rohwedder, and R. Schneider. On manifolds of tensors of fixed tt-rank. *Numerische Mathematik*, 120(4):701–731, 2012.

[10] P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674. ACM, 2013.

[11] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[12] J. Li, J. Choi, I. Perros, J. Sun, and R. Vuduc. Model-driven sparse cp decomposition for higher-order tensors. In *Parallel and Distributed Processing Symposium (IPDPS), 2017 IEEE International*, pages 1048–1057. IEEE, 2017.

[13] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. Tensorizing neural networks. In *Advances in Neural Information Processing Systems*, pages 442–450, 2015.

[14] R. Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[15] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[16] H. N. Phien, H. D. Tuan, J. A. Bengua, and M. N. Do. Efficient tensor completion: Low-rank tensor train. *arXiv preprint arXiv:1601.01083*, 2016.

[17] M. Vasilescu and D. Terzopoulos. Multilinear image analysis for face recognition. *Proceedings of the International Conference on Pattern Recognition ICPR 2002*, 2:511–514, 2002. Quebec City, Canada.

[18] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535*, 2016.