Tensor Completion for Estimating Missing Values in Visual Data

Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye

Abstract—In this paper we propose an algorithm to estimate missing values in tensors of visual data. The values can be missing due to problems in the acquisition process, or because the user manually identified unwanted outliers. Our algorithm works even with a small amount of samples and it can propagate structure to fill larger missing regions. Our methodology is built on recent studies about matrix completion using the matrix trace norm. The contribution of our paper is to extend the matrix case to the tensor case by proposing the first definition of the trace norm for tensors and then by building a working algorithm. First, we propose a definition for the tensor trace norm, that generalizes the established definition of the matrix trace norm. Second, similar to matrix completion, the tensor completion is formulated as a convex optimization problem. Unfortunately, the straightforward problem extension is significantly harder to solve than the matrix case because of the dependency among multiple constraints. To tackle this problem, we developed three algorithms: SiLRTC, FaLRTC, and HaLRTC. The SiLRTC algorithm is simple to implement and employs a relaxation technique to separate the dependant relationships and uses the block coordinate descent (BCD) method to achieve a globally optimal solution; The FaLRTC algorithm utilizes a smoothing scheme to transform the original nonsmooth problem into a smooth one and can be used to solve a general tensor trace norm minimization problem; The HaLRTC algorithm applies the alternating direction method of multipliers (ADMM) to our problem. Our experiments show potential applications of our algorithms and the quantitative evaluation indicates that our methods are more accurate and robust than heuristic approaches. The efficiency comparison indicates that FaLTRC and HaLRTC are more efficient than SiLRTC and between FaLRTC and HaLRTC the former is more efficient to obtain a low accuracy solution and the latter is preferred if a high accuracy solution is desired.

Index Terms—Tensor completion, trace norm, sparse learning.

1 Introduction

In computer vision and graphics, many problems can be formulated as a missing value estimation problem, e.g. image in-painting [4], [22], video decoding, video in-painting [23], scan completion, and appearance acquisition completion. The core problem of the missing value estimation lies on how to build up the relationship between the known elements and the unknown ones. Some energy methods broadly used in image in-painting, e.g. PDEs [4] and belief propagation [22] mainly focus on the local relationship. The basic (implicit) assumption is that the missing entries mainly depend on their neighbors. The further apart two points are, the smaller their dependance is. However, sometimes the value of the missing entry depends on the entries which are far away. Thus, it is necessary to develop a tool to directly capture the global information in the data.

In the two-dimensional case, i.e. the matrix case, the "rank" is a powerful tool to capture some type of global information. In Fig. 1, we show a texture with 80% of its elements removed randomly on the left and its reconstruction using a low rank constraint on the right. This example illustrates the power of low rank approximation for missing data estimation. However, "rank(·)" is unfortunately not a convex function. Some heuristic algorithms were proposed to estimate the missing values iteratively [13], [24]. However, they are not guaranteed to find a globally optimal solution due to the non-convexity of the rank constraint.

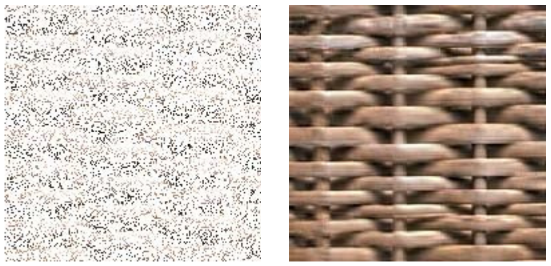

Fig. 1: The left figure contains 80% missing entries shown as white pixels and the right figure shows its reconstruction using the low rank approximation.

Recently, the trace norm of matrices was used to approximate the rank of matrices [30], [7], [37], which leads to a convex optimization problem. The trace norm has been shown to be the tightest convex approximation for the rank of matrices [37], and efficient algorithms for the matrix completion problem using the trace norm constraint were proposed in [30], [7]. Recently, Candès and Recht [9], Recht *et al.* [37], and Candès and Tao [10] showed that under certain conditions, the minimum rank solution can be recovered by solving a convex optimization problem, namely the minimization of the trace norm over the given affine space. Their work theoretically justified the validity

• *Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye are with Arizona State University, Tempe, AZ, 85287.*
E-mail: {Ji.Liu, pmusials, Peter.Wonka, and Jieping.Ye}@asu.edu

of the trace norm to approximate the rank.

Although the low rank approximation problem has been well studied for matrices, there is not much work on tensors, which are a higher-dimensional extension of matrices. One major challenge lies in an appropriate definition of the trace norm for tensors. To the best of our knowledge, this has been not addressed in the literature. In this paper, we make two main contributions: 1) We lay the theoretical foundation of low rank tensor completion and propose the first definition of the trace norm for tensors. 2) We are the first to propose a solution for the low rank completion of tensors.

The challenge of the second part is to build a high quality algorithm. Similar to matrix completion, the tensor completion can be formulated as a convex optimization problem. Unfortunately, the straightforward problem extension is significantly harder to solve than the matrix case because of the dependency among multiple constraints. To tackle this problem, we developed three algorithms: SiLRTC, FaLRTC, and HaLRTC. The SiLRTC algorithm, a pretty simple and intuitive method, employs a relaxation technique to separate the dependant relationships and uses the block coordinate descent (BCD) method to achieve a globally optimal solution. It actually simplifies the LRTC algorithm proposed in our conference paper [29]. The FaLRTC algorithm utilizes a smoothing scheme to transform the original nonsmooth problem into a smooth problem. We also present a theoretical analysis of the convergence rate for the FaLRTC algorithm. The third method applies the alternating direction method of multipliers (ADMM) algorithm [5] to our problem. In addition, we present several heuristic models, which involve non-convex optimization problems. Our experiments show that our method is more accurate and robust than these heuristic approaches. We also give some potential applications in image in-painting, video compression, and BRDF data estimation, using our tensor completion technique. The efficiency comparison indicates that FaLRTC and HaLRTC are more efficient than SiLRTC and between FaLRTC and HaLRTC the former is more efficient to obtain a low accuracy solution and the latter is preferred if a high accuracy solution is desired.

1.1 Notation

We use upper case letters for matrices, e.g. X, and lower case letters for the entries, e.g. x_{ij}. $\Sigma(X)$ is a vector, consisting of the singular values of X in descending order and $\sigma_i(X)$ denotes the i^{th} largest singular value. The Frobenius norm of the matrix X is defined as: $\|X\|_F := (\sum_{i,j} |x_{ij}|^2)^{\frac{1}{2}}$. The spectral norm is denoted as $\|X\| := \sigma_1(X)$ and the trace norm as $\|X\|_{tr} := \sum_i \sigma_i(X)$. Let $X = U\Sigma V^\top$ be the singular value decomposition for X. The "shrinkage" operator $\mathbf{D}_\tau(X)$ is defined as [7]:

$$\mathbf{D}_\tau(X) = U\Sigma_\tau V^\top, \tag{1}$$

where $\Sigma_\tau = diag(\max(\sigma_i - \tau, 0))$. The "truncate" operation $\mathbf{T}_\tau(X)$ is defined as:

$$\mathbf{T}_\tau(X) = U\Sigma_{\bar{\tau}} V^\top, \tag{2}$$

where $\Sigma_{\bar{\tau}} = diag(\min(\sigma_i, \tau))$. It is easy to verify that $X = \mathbf{T}_\tau(X) + \mathbf{D}_\tau(X)$. Let Ω be an index set, then X_Ω denotes the matrix copying the entries from X in the set Ω and letting the remaining entries be "0". A similar definition can be extended to the tensor case. The inner product of the matrix space is defined by $\langle X, Y\rangle = \sum_{i,j} X_{ij}Y_{ij}$.

We follow [11] to define the terminology of tensors used in the paper. An n-mode tensor (or $n-$order tensor) is defined as $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_n}$. Its elements are denoted as $x_{i_1,\cdots,i_n}$, where $1 \le i_k \le I_k, 1 \le k \le n$. For example, a vector is a 1-mode tensor and a matrix is a 2-mode tensor. It is sometimes convenient to unfold a tensor into a matrix. The "unfold" operation along the k-th mode on a tensor $\mathcal{X}$ is defined as $\text{unfold}_k(\mathcal{X}) := X_{(k)} \in \mathbb{R}^{I_k\times(I_1\cdots I_{k-1}I_{k+1}\cdots I_n)}$. The opposite operation "fold" is defined as $\text{fold}_k(X_{(k)}) := \mathcal{X}$. Denote $\|\mathcal{X}\|_F := (\sum_{i_1,i_2,\cdots i_n} |a_{i_1,i_2,\cdots i_n}|^2)^{\frac{1}{2}}$ as the Frobenius norm of a tensor. It is clear that $\|\mathcal{X}\|_F = \|X_{(k)}\|_F$ for any $1 \le k \le n$. Please refer **to** [11] for a more extensive overview of tensors. In addition, we use a nonnegative superscript number to denote the iteration index, e.g., $\mathcal{X}^k$ denotes the value of $\mathcal{X}$ at the k^{th} iteration; the superscript "-2" in K^{-2} denotes the power.

1.2 Organization

We review related work in Section 2, introduce a convex model and three heuristic models for the low rank tensor completion problem in Section 3, present the SiLRTC, FaLRTC, and HaLRTC algorithms to solve the convex model in Section 4, Section 5, and Section 6 respectively, report empirical results in Section 7, and conclude this paper in Section 8. To increase the readability of this paper for the casual reader most technical details can be found in the appendix.

2 Related Work

The low rank or approximately low rank problem broadly occurs in science and engineering, e.g. computer vision [42], machine learning [1], [2], signal processing [26], and bioinformatics [44]. Fazel *et al.* [13], [12] introduced a low rank minimization problem in control system analysis and design. They heuristically used the trace norm to approximate the rank of the matrix. They showed that the trace norm minimization problem can be reformulated as a semidefinite programming (SDP) problem via its dual norm (spectral norm). Srebro *et al.* [39] employed second-order cone programming (SCOP) to formulate a trace norm related problem in matrix factorization. However, many existing optimization methods such as SDPT3 [41] and SeDuMi [40] cannot solve a SDP or SOCP problem when the size of the matrix is much larger than 100×100 [30], [37]. This limitation prevented the usage of the matrix completion technique in computer vision and image processing. Recently, to solve the rank minimization problem for large scale matrices, Ma *et al.* [30] applied the fixed point and Bregman iterative method and Cai *et al.* [7] proposed a singular value thresholding algorithm. In both algorithms, one key building block is the existence of a closed form solution for the following optimization problem:

$$\min_{X\in\mathbb{R}^{p\times q}}: \quad \frac{1}{2}\|X-M\|_F^2+\tau\|X\|_{tr}, \tag{3}$$

where $M\in\mathbb{R}^{p\times q}$, and τ is a constant. Candès and Recht [9], Recht *et al.* [37], and Candès and Tao [10] theoretically justified the validity of the trace norm to approximate the rank of matrices. Recht [36] recently improved their result and also largely simplified the proof by using the golfing scheme from quantum information theory [15]. An alternative singular value based method for matrix completion was recently proposed and justified by Keshavan *et al.* [21].

This journal paper builds on our own previous work [29] where we extended the matrix trace norm to the tensor case and proposed to recover the missing entries in a low rank tensor by solving a tensor trace norm minimization problem. We used a relaxation trick on the objective function such that the block coordinate descent algorithm can be employed to solve this problem [29]. Since this approach is not efficient enough, some recent papers tried to use the alternating direction method of multipliers (ADMM) to efficiently solve the tensor trace norm minimization problem. The ADMM algorithm was developed in the 1970s, but was successful in solving large scale problems and optimization problems with multiple nonsmooth terms in the objective function [28] recently. Signoretto *et al.* [38] and Gandy *et al.* [14] applied the ADMM algorithm to solve the tensor completion problem with Gaussian observation noise, i.e.,

$$\min_{\mathcal{X}}: \quad \frac{\lambda}{2}\|\mathcal{X}_\Omega-\mathcal{T}_\Omega\|_F^2+\|\mathcal{X}\|_*, \tag{4}$$

where $\|\mathcal{X}\|_*$ is the tensor trace norm defined in Eq. (8). The tensor completion problem without observation noise can be solved by optimizing Eq. (4) iteratively with an increasing value of λ [38], [14]. Tomioka *et al.* [43] proposed several slightly different models for the problem Eq. (4) by introducing dummy variables and also applied ADMM to solve them. Out of these three algorithms for tensor completion based on ADMM, we choose to compare to the algorithm by Gandy et al., because the problem statement is identical to ours. Our results will show that our adaption of ADMM and our proposed FaLRTC algorithm are more efficient.

Besides tensor completion, the tensor trace norm proposed in [26] can be applied in various other computer vision problems such as visual saliency detection [47], medical imaging [16], corrupted data correction [26], [27], data compression [25].

3 The Formulation of Tensor Completion

This section presents a convex model and three heuristic models for tensor completion.

3.1 Convex Formulation for Tensor Completion

Before introducing the low rank tensor completion problem, let us start from the well-known optimization problem [24] for low rank matrix completion:

$$\begin{aligned}\min_X:&\quad \text{rank}(X)\\ s.t.:&\quad X_\Omega=M_\Omega,\end{aligned} \tag{5}$$

where $X,M\in\mathbb{R}^{p\times q}$, and the elements of M in the set Ω are given while the remaining elements are missing. The missing elements of X are determined such that the rank of the matrix X is as small as possible. The optimization problem in Eq. (5) is a nonconvex optimization problem since the function rank(X) is nonconvex. One common approach is to use the trace norm $\|.\|_*$ to approximate the rank of matrices. The advantage of the trace norm is that $\|.\|_*$ is the tightest convex envelop for the rank of matrices. This leads to the following convex optimization problem for matrix completion [3], [7], [30]:

$$\begin{aligned}\min_X:&\quad \|X\|_*\\ s.t.:&\quad X_\Omega=M_\Omega.\end{aligned} \tag{6}$$

The tensor is the generalization of the matrix concept. We generalize the completion algorithm for the matrix (i.e., 2-mode or 2-order tensor) case to higher-order tensors by solving the following optimization problem:

$$\begin{aligned}\min_{\mathcal{X}}:&\quad \|\mathcal{X}\|_*\\ s.t.:&\quad \mathcal{X}_\Omega=\mathcal{T}_\Omega\end{aligned} \tag{7}$$

where $\mathcal{X},\mathcal{T}$ are n-mode tensors with identical size in each mode. The first issue is the definition of the trace norm for the general tensor case. We propose the following definition for the tensor trace norm:

$$\|\mathcal{X}\|_* := \sum_{i=1}^n \alpha_i\|\mathcal{X}_{(i)}\|_*. \tag{8}$$

where α_i's are constants satisfying $\alpha_i\geq 0$ and $\sum_{i=1}^n\alpha_i=1$. In essence, the trace norm of a tensor is a convex combination of the trace norms of all matrices unfolded along each mode. Note that when the mode number n is equal to 2 (i.e. the matrix case), the definition of the trace norm of a tensor is consistent with the matrix case, because the trace norm of a matrix is equal to the trace norm of its transpose. Under this definition, the optimization in Eq. (7) can be written as:

$$\begin{aligned}\min_{\mathcal{X}}:&\quad \sum_{i=1}^n \alpha_i\|\mathcal{X}_{(i)}\|_*\\ s.t.:&\quad \mathcal{X}_\Omega=\mathcal{T}_\Omega.\end{aligned} \tag{9}$$

Here one might ask why we do not define the tensor trace norm as the convex envelop of the tensor rank like in the matrix case. Unlike matrices, computing the rank of a general tensor (mode number >2) is an NP hard problem [18]. Therefore, there is no explicit expression for the convex envelop of the tensor rank to the best of our knowledge.

3.2 Three Heuristic Algorithm

We introduce several heuristic models, which, unlike the one in the last section, involve non-convex optimization problems. A goal of introducing the heuristic algorithms is to establish some basic methods that can be used for comparison.

Tucker: One natural approach is to use the Tucker model [46] for tensor factorization to the tensor completion problem as follows:

$$\begin{aligned} \min_{\mathcal{X},C,U_1,\cdots,U_n} :\quad & \frac{1}{2}\|\mathcal{X} - C \times_1 U_1 \times_2 U_2 \times_3 \cdots \times_n U_n\|_F^2 \\ s.t. :\quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega \end{aligned} \tag{10}$$

where $C \times_1 U_1 \times_2 U_2 \times_3 \cdots \times_n U_n$ is the Tucker model based tensor factorization, $U_i \in \mathbb{R}^{I_i \times r_i}$, $\mathcal{C} \in \mathbb{R}^{r_1 \times \cdots \times r_n}$, and $\mathcal{T}, \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$. One can simply use the block coordinate descent method to solve this problem by iteratively optimizing two blocks $\mathcal{X}$ and $C, U_1, \cdots, U_n$ respectively while fixing the other. $\mathcal{X}$ can be computed by letting $\mathcal{X}_\Omega = \mathcal{T}_\Omega$ and $\mathcal{X}_{\bar{\Omega}} = (C \times_1 U_1 \times_2 U_2 \times_3 \cdots \times_n U_n)_{\bar{\Omega}}$. $C, U_1, \cdots, U_n$ can be computed by any existing tensor factorization algorithm based on the Tucker model. The procedure can also be employed to solve the following two heuristic algorithms.

Parafac: Another natural approach is to use the parallel factor analysis (Parafac) model [17], resulting in the following optimization problem:

$$\begin{aligned} \min_{\mathcal{X},U_1,U_2,\cdots,U_n} :\quad & \frac{1}{2}\|\mathcal{X} - U_1 \circ U_2 \circ \cdots \circ U_n\|_F^2 \\ s.t. :\quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega \end{aligned} \tag{11}$$

where $\circ$ denotes the outer product and $U_1 \circ U_2 \circ \cdots \circ U_n$ is the Parafac model based decomposition, $U_i \in \mathbb{R}^{I_i \times r}$, and $\mathcal{T}, \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$.

SVD: The third alternative is to consider the tensor as multiple matrices and force the unfolding matrix along each mode of the tensor to be low rank as follows:

$$\begin{aligned} \min_{\mathcal{X},M_1,M_2,\cdots,M_n} :\quad & \frac{1}{2}\sum_{i=1}^{n} \|\mathcal{X}_{(i)} - M_i\|_F^2 \\ s.t. :\quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega \\ & \text{rank}(M_i) \le r_i \quad i = 1, \cdots, n. \end{aligned} \tag{12}$$

where $M_i \in \mathbb{R}^{I_i \times (\prod_{k \neq i} I_k)}$, and $\mathcal{T}, \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_n}$.

4 A Simple Low Rank Tensor Completion (SiLRTC) Algorithm

In this section we present the SiLRTC algorithm to solve the convex model in Eq. (9), which is simple to understand and to implement. In Section 4.1, we relax the original problem into a simple convex structure which can be solved by block coordinate descent. Section 4.2 presents the details of the proposed algorithm.

4.1 Simplified Formulation

The problem in Eq. (9) is difficult to solve due to the interdependent matrix trace norm terms, i.e., while we optimize the sum of multiple matrix trace norms, the matrices share the same entries and cannot be optimized independently. Hence, the existing result in Eq. (3) cannot be used directly. Our key motivation of simplifying this original problem is how to split these interdependent terms such that they can be solved independently. We introduce additional matrices $M_1, \cdots, M_n$ and obtain the following equivalent formulation:

$$\begin{aligned} \min_{\mathcal{X},M_i} :\quad & \sum_{i=1}^{n} \alpha_i \|M_i\|_* \\ s.t. :\quad & \mathcal{X}_{(i)} = M_i \quad \text{for } i = 1, \cdots, n \\ & \mathcal{X}_\Omega = \mathcal{T}_\Omega \end{aligned} \tag{13}$$

In this formulation, the trace norm terms are still not independent because of the equality constraints $M_i = \mathcal{X}_{(i)}$ which enforces all M_i's to be identical. Thus, we relax the equality constraints $M_i = \mathcal{X}_{(i)}$ by $\|M_i - \mathcal{X}_{(i)}\|_F^2 \le d_i$ as Eq. (14), so that we can independently solve each subproblem later on.

$$\begin{aligned} \min_{\mathcal{X},M_i} :\quad & \sum_{i=1}^{n} \alpha_i \|M_i\|_* \\ s.t. :\quad & \|\mathcal{X}_{(i)} - M_i\|_F^2 \le d_i \quad \text{for } i = 1, \cdots, n \\ & \mathcal{X}_\Omega = \mathcal{T}_\Omega \end{aligned} \tag{14}$$

$d_i (> 0)$ is a threshold that could be defined by the user, but we do not use d_i explicitly in our algorithm. This optimization problem can be converted to an equivalent formulation for certain positive values of β_i's:

$$\begin{aligned} \min_{\mathcal{X},M_i} :\quad & \sum_{i=1}^{n} \alpha_i \|M_i\|_* + \frac{\beta_i}{2}\|\mathcal{X}_{(i)} - M_i\|_F^2 \\ s.t. :\quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega. \end{aligned} \tag{15}$$

This is a convex but nondifferentiable optimization problem. Next, we show how to solve the optimization problem in Eq. (15).

4.2 The Main Algorithm

We propose to employ block coordinate descent (BCD) for the optimization. The basic idea of block coordinate descent is to optimize a group (block) of variables while fixing the other groups. We divide the variables into $n + 1$ blocks: $\mathcal{X}, M_1, M_2, \cdots, M_n$.

Computing $\mathcal{X}$: The optimal $\mathcal{X}$ with all other variables fixed is given by solving the following subproblem:

$$\begin{aligned} \min_{\mathcal{X}} :\quad & \sum_{i=1}^{n} \frac{\beta_i}{2}\|M_i - \mathcal{X}_{(i)}\|_F^2 \\ s.t. :\quad & \mathcal{X}_\Omega = \mathcal{T}_\Omega. \end{aligned} \tag{16}$$

It is easy to check that the solution to Eq. (16) is given by

$$\mathcal{X}_{i_1,\cdots,i_n} = \begin{cases} \left(\frac{\sum_i \beta_i \text{fold}_i(M_i)}{\sum_i \beta_i}\right)_{i_1,\cdots,i_n} & (i_1, \cdots, i_n) \notin \Omega; \\ \mathcal{T}_{i_1,\cdots,i_n} & (i_1, \cdots, i_n) \in \Omega. \end{cases} \tag{17}$$

Computing M_i**:** M_i is the optimal solution of the following problem.

$$\begin{aligned}\min_{M_i}:\ &\frac{\beta_i}{2}\|M_i-\mathcal{X}_{(i)}\|_F^2+\alpha_i\|M_i\|_*\\ \equiv&\frac{1}{2}\|M_i-\mathcal{X}_{(i)}\|_F^2+\frac{\alpha_i}{\beta_i}\|M_i\|_*.\end{aligned} \tag{18}$$

This problem has been proven to lead to a closed form in recent papers like [30], [7]. Thus the optimal M_i can be computed as $\mathbf{D}_\tau(\mathcal{X}_{(i)})$ where $\tau=\frac{\alpha_i}{\beta_i}$.

We call the proposed algorithm "SiLRTC", which stands for Simple Low Rank Tensor Completion algorithm. The pseudo-code of the SiLRTC algorithm is given in **Algorithm 1** below. As convergence criteria we compare the difference of $\mathcal{X}$ in subsequent iterations to a threshold. Since the objective in Eq. (15) is convex and the nonsmooth term is separable, BCD is guaranteed to find the global optimal solution [45]. Note that this SiLRTC algorithm actually simplifies the LRTC algorithm proposed in our conference paper [29] by removing a redundant variable $\mathcal{Y}$. SiLRTC and LRTC produce almost identical results.

Algorithm 1 SiLRTC: Simple Low Rank Tensor Completion

```
Input: $\mathcal{X}$ with $\mathcal{X}_\Omega=\mathcal{T}_\Omega$, $\beta_i$'s, and $K$
Output: $\mathcal{X}$
1: for $k=1$ to $K$ do
2:     for $i=1$ to $n$ do
3:         $M_i=\mathbf{D}_{\frac{\alpha_i}{\beta_i}}(\mathcal{X}_{(i)})$
4:     end for
5:     update $\mathcal{X}$ by Eq. (17).
6: end for
```

5 A Fast Low Rank Tensor Completion (FaLRTC) Algorithm

Although the proposed algorithm in Section 4 is easy to implement, its convergence speed is low in practice. In addition, SiLRTC is hard to extend to any tensor trace norm minimization problem, e.g., the formulation "logistic loss + tensor trace norm" is hard to minimize using the strategy above. In this section we propose a new algorithm to significantly improve the convergence speed of SiLRTC and to solve a general tensor trace norm minimization problem defined below:

$$\min_{\mathcal{X}\in Q}:\ f(\mathcal{X}):=f_0(\mathcal{X})+\sum_{i=1}^{n}\alpha_i\|\mathcal{X}_{(i)}\|_* \tag{19}$$

where $\mathcal{X}\in\mathbb{R}^{I_1\times I_2\times\ldots\times I_n}$, Q is a convex set, and $f_0(\mathcal{X})$ is smooth and convex. One can easily verify that the low rank tensor completion problem in Eq. (9) is just a special case with $f_0(\mathcal{X})=0$ and $Q=\{\mathcal{X}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_n}\mid\mathcal{X}_\Omega=\mathcal{T}_\Omega\}$.

The difficulty to efficiently solve the tensor trace norm related minimization problems lies on that there exist multiple dependent nonsmooth terms in the objective function. Although one can use the subgradient information to replace the gradient information, the convergence rate is $O(K^{-1/2})$ where K is the iteration number [33]. In comparison, the optimal convergence rate for minimizing general smooth functions is $O(K^{-2})$ [33], [31]. Nesterov [34] proposed a general method to solve a nonsmooth optimization problem. The basic idea is to

- first convert the original nonsmooth problem into a smooth one;
- then solve the smooth problem and use its solution to approximate the original problem.

We will follow this procedure to solve the problem in Eq. (19).

Section 5.1 employs the smoothing scheme proposed by Nesterov [34] to convert the original nonsmooth objective function in Eq. (19) into a smooth one. Section 5.2 proposes an efficient updating rule to solve the smooth problem and analyzes the convergence rate of this method. Section 5.3 applies the smoothing scheme and the efficient updating rule to the low rank tensor completion problem.

5.1 Smoothing Scheme

Consider one nonsmooth term in Eq. (19), i.e., the matrix trace norm function $\|X\|_*$. Its dual version can be written as:

$$g(X):=\|X\|_*=\max_{\|Y\|\le1}\langle X,Y\rangle. \tag{20}$$

Its smooth version is

$$g_\mu(X)=\max_{\|Y\|\le1}\langle X,Y\rangle-d_\mu(Y) \tag{21}$$

where $d_\mu(Y)$ is a strongly convex function with the parameter μ. Theorem 1 in [34] proves that $g_\mu(X)$ is smooth and its gradient can be computed by

$$\nabla g_\mu(X)=Y^*(X):=arg\max_{\|Y\|\le1}\langle X,Y\rangle-d_\mu(Y). \tag{22}$$

Although one can arbitrarily choose a strongly convex function $d_\mu(Y)$ to smooth the original function, a good choice can lead to a closed form for the dual variables Y. Otherwise, it involves a complicated min-max optimization problem. Here, we choose $d(Y)=\frac{\mu}{2}\|Y\|_F^2$ where $\mu>0$ as the strongly convex term and the gradient is

$$\begin{aligned}\nabla g_\mu(X)=&Y^*(X)\\ :=&arg\max_{\|Y\|\le1}\langle X,Y\rangle-\frac{\mu}{2}\|Y\|_F^2\\ =&arg\min_{\|Y\|\le1}\|Y-\frac{1}{\mu}X\|_F^2\\ =&\mathbf{T}_1(\frac{1}{\mu}X).\end{aligned} \tag{23}$$

The last equality is due to **Lemma** 1 in the **Supplemental Material**.

We apply this smoothing scheme to all nonsmooth terms in Eq. (19) by introducing n dual variables $\mathcal{Y}_1,\cdots,\mathcal{Y}_n\in$

$\mathbb{R}^{I_1\times I_2\times\cdots\times I_n}$ and n positive constants $\mu_1,\cdots,\mu_n$. The objective function $f(X)$ is converted into:

$$\begin{aligned}&f_\mu(\mathcal{X}) := f_{\mu_1,\cdots,\mu_n}(\mathcal{X})\\ &=f_0(\mathcal{X}) + \sum_{i=1}^n \max_{\|\mathcal{Y}_{i(i)}\|\le 1} \alpha_i\langle \mathcal{X}_{(i)}, \mathcal{Y}_{i(i)}\rangle - \frac{\mu_i}{2}\|\mathcal{Y}_{i(i)}\|_F^2\\ &=f_0(\mathcal{X}) + \sum_{i=1}^n \max_{\|\mathcal{Y}_{i(i)}\|\le 1} \alpha_i\langle \mathcal{X}, \mathcal{Y}_{i}\rangle - \frac{\mu_i}{2}\|\mathcal{Y}_{i}\|_F^2.\end{aligned} \tag{24}$$

Its gradient can be computed by

$$\nabla f_\mu(\mathcal{X}) = \nabla f_0(\mathcal{X}) + \sum_{i=1}^n \alpha_i \mathbf{T}_1\left(\frac{\alpha_i}{\mu_i}\mathcal{X}_{(i)}\right). \tag{25}$$

Finally, we obtain a smooth optimization problem as follows:

$$\min_{\mathcal{X}\in Q\subset\mathbb{R}^{I_1\times\cdots\times I_n}} : \ f_\mu(\mathcal{X}) \tag{26}$$

and will use its optimal solution to approximate the original problem in Eq. (19).

5.2 An Efficient Algorithm to Solve the Smooth Version

In essence, the problem in Eq. (26) is a smooth optimization. Nesterov [34] also proposed an algorithm to solve any smooth problem in a bounded domain and guaranteed two convergence rates $O(K^{-2})$ for the smooth problem and $O(K^{-1})$ for the original problem, i.e.,

$$f_\mu(\mathcal{X}^K) - f_\mu(\mathcal{X}^*_\mu) \le O\left(K^{-2}\right) \tag{27}$$

$$f(\mathcal{X}^K) - f(\mathcal{X}^*) \le O\left(K^{-1}\right), \tag{28}$$

where $\mathcal{X}^*_\mu$ and $\mathcal{X}^*$ are respectively the optimal solutions of $f_\mu(\mathcal{X})$ and $f(\mathcal{X})$, $\mathcal{X}^0$ is the initial point, and $\mathcal{X}^K$ is the output of our updating rule for $f_\mu(\mathcal{X})$. One is not surprised about the first convergence rate, since for a general smooth optimization problem, the rate $O(K^{-2})$ is guaranteed by Nesterov's popular accelerated algorithm [32]. The second rate is quite interesting. Note that $\mathcal{X}^K$ is the output by minimizing $f_\mu(\mathcal{X})$ instead of $f(\mathcal{X})$. The second inequality indeed indicates how far the approximate solution is away from the true solution after K iterations.

However, the constant factor of $O(K^{-1})$ in Eq. (28) is proportional to the size of the domain set Q, see Theorem 3 [34]. For this reason, the domain Q is assumed to be bounded in [34]. Hence, if this algorithm is applied to solve Eq. (26) directly, the inequality in Eq. (28) cannot be guaranteed. Based on the algorithm in [34], we propose an improved efficient algorithm to solve the problem Eq. (26) in **Algorithm** 2 which allows the domain set Q to be unbound and can guarantee the results in Eq. (27) and Eq. (28). A more detailed explanation of the accelerated scheme in **Algorithm 2** is provided in the **Supplemental Material**.

Unsurprisingly, the proposed algorithm can guarantee the convergence rate $O(K^{-2})$ for the smooth problem like many existing accelerated algorithms [33], [34], [19], [31], see **Theorem** 2 in the **Supplemental Material**.

At the same time, the following theorem shows that the convergence rate $O(K^{-1})$ for the original nonsmooth problem is also obtained by the proposed algorithm.

Theorem 1. *Define D as any positive constant satisfying*

$$D \ge \min_{\mathcal{X}^*} : \ \|\mathcal{X}^* - \mathcal{X}^0\|_F, \tag{34}$$

where $\mathcal{X}^*$ is the optimal solution of the problem Eq. (19) and $\mathcal{X}^0$ is the starting point. Set the parameters in the problem (26) as

$$\mu_i = \frac{2\alpha_i D}{K\sqrt{cI_i}}.$$

*After K iterations in **Algorithm** 2, the output $\mathcal{X}^K$ for the problem in Eq. (26) satisfies*

$$f(\mathcal{X}^K) - f(\mathcal{X}^*) \le \frac{2\bar{L}}{c}\left(\frac{D}{K}\right)^2 + \frac{2D}{K}\sum_i \alpha_i\sqrt{\frac{I_i}{c}}. \tag{35}$$

where $\bar{L}$ is the Lipschitz constant of $f_0(\mathcal{X})$.

Theorem 1 and **Theorem** 2 extend the results in [34].

Although Nesterov's popular accelerated algorithm [32], [31], [33] can guarantee the convergence rate $O(K^{-2})$ for the smooth problem, there is no evidence showing that it can achieve $O(K^{-1})$ for the original problem. Ji et al. [19] used this smoothing scheme and Nesterov's popular accelerated method to solve a multiple kernel learning problem, and also claimed the convergence rate $O(K^{-1})$ for the original problem. However, what they really guaranteed is $f(\mathcal{X}^K) - f(\mathcal{X}^*_\mu) \le O(K^{-1})$ which can be obtained from Eq. (27).

Theorem 1 also implies that the reasonable parameters should satisfy

$$\mu_1 \ : \ \mu_2 \ : \ \cdots \ : \mu_n = \frac{\alpha_1}{\sqrt{I_1}} \ : \ \frac{\alpha_2}{\sqrt{I_2}} : \ \cdots \ : \frac{\alpha_n}{\sqrt{I_n}}.$$

5.3 The FaLRTC Algorithm

This section considers the specific tensor completion problem in Eq. (9). The smooth version of this problem is

$$\begin{aligned}\min_{\mathcal{X}} : \ &\sum_{i=1}^n \max_{\|\mathcal{Y}_{i(i)}\|\le 1} : \alpha_i\langle\mathcal{X},\mathcal{Y}_i\rangle - \frac{\mu_i}{2}\|\mathcal{Y}\|_F^2\\ s.t. : \ &\mathcal{X}_\Omega = \mathcal{T}_\Omega.\end{aligned} \tag{36}$$

Now we can use the proposed **Algorithm** 2 to solve this smooth minimization problem above. First it is easy to verify

$$\mathcal{Z}^{k+1} = \mathcal{Z}^k - \frac{\theta^{k+1}}{L^k}\nabla f_\mu(\mathcal{W}^{k+1}), \tag{37}$$

$$\begin{aligned}&\left(\nabla f_\mu(\mathcal{W}^{k+1})\right)_{i_1,\cdots,i_n} =\\ &\begin{cases}\left(\sum_i \frac{(\alpha_i)^2}{\mu_i}\mathbf{T}_{\frac{\mu_i}{\alpha_i}}(\mathcal{W}^{k+1}_{(i)})\right)_{i_1,\cdots,i_n}, & (i_1,i_2,\cdots,i_n)\notin\Omega;\\ 0, & (i_1,i_2,\cdots,i_n)\in\Omega.\end{cases}\end{aligned} \tag{38}$$

The last equation is due to $\mathcal{W}^{k+1}_\Omega = T_\Omega$. For a simple implementation, we require the sequence L^k to be non-decreasing, although the updating rule in **Algorithm** 2

Algorithm 2 An Efficient Algorithm

```
Input: $c \in (0,1)$, $x^0$, $K$, and $L^0$.
Output: $x^K$
1: Initialize $z^0 = w^0 = x^0$ and $B^0 = 0$
2: for $k = 0$ to $K$ do
3:     Find a $L^{k+1}$ as small as possible from $\{\cdots,\ cL^k,\ L^k,\ L^k/c,\ ...\}$, such that Eq. (31) holds. Let
       $$\theta^{k+1} = \frac{L^k}{2L^{k+1}}(1+\sqrt{1+4B^kL^{k+1}}),\ w^{k+1} = \tau^{k+1}z^k + (1-\tau^{k+1})x^k \quad (29)$$
       $$x' = arg\min_{x\in Q}:\ p(x) = f_\mu(w^{k+1}) + \langle \nabla f_\mu(w^{k+1}), x - w^{k+1}\rangle + \frac{L^{k+1}}{2}\|x - w^{k+1}\|^2 \quad (30)$$
       where $\tau^{k+1} = (\frac{\theta^{k+1}}{L^k})/B^{k+1}$ and $B^k = \sum_{i=1}^k \frac{\theta^i}{L^{i-1}}$. Test whether the following inequality holds:
       $$f_\mu(x') \le p(x^{k+1}), \quad (31)$$
4:     Update $x^{k+1}$ and $z^{k+1}$ by
       $$x^{k+1} = arg\min_{x\in\{x', x^k, z^k\}}:\ f_\mu(x), \quad (32)$$
       $$z^{k+1} = arg\min_{z\in Q}:\ h(z) = \frac{1}{2}\|z - x^0\|^2 + \sum_{i=1}^{k+1}\frac{\theta^i}{L^{i-1}}(f_\mu(w^i) + \langle\nabla f_\mu(w^i), z - w^i\rangle) \quad (33)$$
5: end for
```

allows that L^k (L' in **Algorithm** 3) becomes smaller (theoretically, the smaller, the better, see the proof of **Theorem** 2). Because $f(\mathcal{X})$ involves computing SVD (this is a big workload), we remove $\mathcal{Z}^k$ from the candidate pool. Note that these slight changes do not change the properties of the output $\mathcal{X}^K$ in **Theorem** 2 and **Theorem** 1. Finally, **Algorithm** 3 summarizes the fast low rank tensor completion algorithm, namely, FaLRTC. The lines 5 to 7 in **Algorithm** 3 are optional as they merely guarantee that the objective is nonincreasing. The lines 8 to 13 are the line search step. The main cost in this part is evaluating $f_\mu(\mathcal{X}')$, $f_\mu(\mathcal{W})$, and $\nabla f_\mu(\mathcal{W})$. In fact, one of $f_\mu(\mathcal{W})$ and $\nabla f_\mu(\mathcal{W})$ can be obtained without much additional cost while computing the other. Since all gradient algorithms have to compute the gradient, the additional cost in each iteration of the proposed algorithm is to compute $f_\mu(\mathcal{X}')$. One can avoid this extra cost by initialing L as the Lipschitz constant of the objective $f_\mu(\mathcal{W})$, i.e., $\sum_{i=1}^n \frac{1}{\mu_i}$ because it satisfies the condition in line 9. However, in practice a line search often improves the efficiency. In addition, the FaLRTC algorithm can be accelerated by decreasing μ_i iteratively. Specifically, we set $\mu_i^k = ak^{-p} + b$ for $k = 1, ..., K$ at the k^{th} iteration where p is a constant in the range $[1.1, 1.2]$ and a and b are determined by solving $\mu_i^0 = a + b$ and $\mu_i^K = aK^{-p} + b$ ($\mu_i^0 = 0.4\alpha_i\|\mathcal{X}_{(i)}\|$, $\mu_i^K = \mu_i$ is the input of **Algorithm 3**).

Algorithm 3 FaLRTC: Fast Low Rank Tensor Completion

```
Input: $c \in (0,1)$, $\mathcal{X}$ with $\mathcal{X}_\Omega = \mathcal{T}_\Omega$, $K$, $\mu_i$'s, and $L$.
Output: $\mathcal{X}$
1: Initialize $\mathcal{Z} = \mathcal{W} = \mathcal{X}$, $L' = L$, and $B = 0$
2: for $k = 0$ to $K$ do
3:     while true do
4:         $$\begin{aligned}\theta =& \frac{L}{2L'}(1+\sqrt{1+4L'B});\\ \mathcal{W} =& \frac{\theta/L}{B+\theta/L}\mathcal{Z} + \frac{B}{B+\theta/L}\mathcal{X};\end{aligned} \quad (39)$$
5:         if $f_\mu(\mathcal{X}) \le f_\mu(\mathcal{W}) - \|\nabla f_\mu(\mathcal{W})\|_F^2/2L'$ then
6:             break;
7:         end if
8:         $\mathcal{X}' = \mathcal{W} - \nabla f_\mu(\mathcal{W})/L'$;
9:         if $f_\mu(\mathcal{X}') \le f_\mu(\mathcal{W}) - \|\nabla f_\mu(\mathcal{W})\|_F^2/2L'$ then
10:            $\mathcal{X} = \mathcal{X}'$;
11:            break;
12:        end if
13:        $L' = L'/c$;
14:    end while
15:    $$\begin{aligned}L =& L';\\ \mathcal{Z} =& \mathcal{Z} - \frac{\theta}{L}\nabla f_\mu(\mathcal{W});\\ B =& B + \frac{\theta}{L};\end{aligned} \quad (40)$$
16: end for
```

6 A High Accuracy Low Rank Tensor Completion (HaLRTC) Algorithm

The ADMM algorithm was developed in the 1970s, with roots in the 1950s, but received renewed interest due to the fact that it is efficient to tackle large scale problems and solve optimization problems with multiple nonsmooth terms in the objective [28]. This section follows the ADMM algorithm to solve the noiseless case in a direct way. Based on the SiLRTC algorithm, we also give a simple implementation using the ADMM framework. Recall that the formulation in Eq. (13) is an equivalent form of the original problem. We replace the dummy matrices M_i's by their tensor versions:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{M}_1,\cdots,\mathcal{M}_n} : \ & \sum_{i=1}^{n} \alpha_i \|\mathcal{M}_{i(i)}\|_* \\ s.t. : \ & \mathcal{X}_\Omega = \mathcal{T}_\Omega \\ & \mathcal{X} = \mathcal{M}_i, \ i = 1, \cdots, n. \end{aligned} \tag{41}$$

We define the augmented Lagrangian function as follows:

$$\begin{aligned} & L_\rho(\mathcal{X}, \mathcal{M}_1, \cdots, \mathcal{M}_n, \mathcal{Y}_1, \cdots, \mathcal{Y}_n) \\ =& \sum_{i=1}^{n} \alpha_i \|\mathcal{M}_{i(i)}\|_* + \langle \mathcal{X} - \mathcal{M}_i, \mathcal{Y}_i \rangle + \frac{\rho}{2} \|\mathcal{M}_i - \mathcal{X}\|_F^2 \end{aligned} \tag{42}$$

According to the framework of ADMM, one can iteratively update $\mathcal{M}_i$'s, $\mathcal{X}$, and $\mathcal{Y}_i$'s as follows:

1) $\{\mathcal{M}_1^{k+1}, \cdots, \mathcal{M}_n^{k+1}\} = arg \min_{\mathcal{M}_1,\cdots,\mathcal{M}_n} :$ $L_\rho(\mathcal{X}^k, \mathcal{M}_1, \cdots, \mathcal{M}_n, \mathcal{Y}_1^{k+1}, \cdots, \mathcal{Y}_n^{k+1})$
2) $\mathcal{X}^{k+1} = arg \min_{\mathcal{X} \in Q} :$ $L_\rho(\mathcal{X}, \mathcal{M}_1^{k+1}, \cdots, \mathcal{M}_n^{k+1}, \mathcal{Y}_1^k, \cdots, \mathcal{Y}_n^{k+1})$
3) $\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^k - \rho(\mathcal{M}_i^{k+1} - \mathcal{X}^{k+1})$.

One can refer to Eq. (18) and Eq. (16) in the SiLRTC algorithm to obtain the closed form solutions for the first two steps. We summarize the HaLRTC algorithm in **Algorithm 4**. This algorithm can also be accelerated by adaptively changing ρ. An efficient strategy [28] is to let $\rho^0 = \rho$ (the input in **Algorithm 4**) and increase ρ^k iteratively by $\rho^{k+1} = t\rho^k$ where $t \in [1.1, 1.2]$.

Algorithm 4 HaLRTC: High Accuracy Low Rank Tensor Completion

```
Input: $\mathcal{X}$ with $\mathcal{X}_\Omega = \mathcal{T}_\Omega$, $\rho$, and $K$
Output: $\mathcal{X}$
1: Set $\mathcal{X}_\Omega = \mathcal{T}_\Omega$ and $\mathcal{X}_{\bar{\Omega}} = 0$.
2: for $k = 0$ to $K$ do
3:     for $i = 1$ to $n$ do
4:         $\mathcal{M}_i = \text{fold}_i \left[ \mathbf{D}_{\frac{\alpha_i}{\rho}} \left( \mathcal{X}_{(i)} + \frac{1}{\rho} \mathcal{Y}_{i(i)} \right) \right]$
5:     end for
6:     $\mathcal{X}_{\bar{\Omega}} = \frac{1}{n} \left( \sum_{i=1}^{n} \mathcal{M}_i - \frac{1}{\rho} \mathcal{Y}_i \right)_{\bar{\Omega}}$
7:     $\mathcal{Y}_i = \mathcal{Y}_i - \rho(\mathcal{M}_i - \mathcal{X})$
8: end for
```

Note that the proposed ADMM algorithm in this section aims to solve the tensor completion problem without observation noise unlike the previous work in [38], [43], [14]. Signoretto *et al.* [38] and Gandy *et al.* [14] also consider the noiseless case in Eq. (9). They relax the equality constraint in Eq. (9) into the noisy case in Eq. (4) and apply the ADMM framework to solve the relaxed problem with an increasing value of λ. However, our ADMM algorithm handles this equality constraint directly without using any relaxation technique. The comparison in Section 7 will show that our ADMM is more efficient than the ADM-TR Algorithm in [14].

Although the convergence of the general ADMM algorithm is guaranteed [28], the convergence rate may be slow. From the comparison in Section 7.3, we can observe that HaLRTC is comparable to FaLRTC and even more efficient to achieve a higher accuracy.

7 Results

In this section, we first validate the tensor trace norm based model in Eq. (9) by comparing to three heuristic models in Section 3.2 and the matrix completion model for tensor completion. Then the efficiency comparison between SiLRTC, FaLRTC, and HaLRTC is reported. Several applications of tensor completion conclude this section. All experiments were implemented in Matlab (version 7.9.0) and all tests were performed on an Intel Core 2 2.67GHz and 3GB RAM computer.

In the following we will use the 3-mode tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ to explain how we generated the synthetic test data sets used in the following comparison. All synthetic data in Section 7 are generated in this manner. The tensor data follows the tucker model, i.e., $\mathcal{T} = \mathcal{C} \times_1 U_1 \times_2 U_2 \times_3 U_3$ where the core tensor $\mathcal{C}$ is of size $r_1 \times r_2 \times r_3$ and U_i is of size $I_i \times r_i$. The entry of $\mathcal{T}$ is computed by

$$\mathcal{T}(i,j,k) = \sum_{1 \le m,n,l \le r} \mathcal{C}(m,n,l) U_1(i,m) U_2(j,n) U_3(k,l). \tag{43}$$

The entries of U_i are random samples drawn from a uniform distribution in the range $[-0.5, \ 0.5]$ and the entries of $\mathcal{C}$ are from a uniform distribution in the range $[0,1]$. Typically, the ranks of the tensor $\mathcal{T}$ unfolded respectively along three modes are $[r_1, r_2, r_3]$. The data is finally normalized such as $\|\mathcal{T}\|_F =$ the number of entries in $\mathcal{T}$.

7.1 Model Comparison: Tensor Trace Norm Versus Heuristic Models

We compare the proposed tensor completion model in Eq. (9) to three heuristic models in Section 3.2 on both synthetic and real-world data. We use SiLRTC, FaLRTC, and HaLRTC to solve Eq. (9) and compare their results to three heuristic methods. Since three heuristic models are nonconvex, multiple initial points are tested and the average performance is used for comparison. The initial value at each missing entry is generated from the Guassian distribution $N(\mu, 1)$ where μ is the average of other observed entries. All experiments are repeated 10 times. The performance is measured by $RSE = \|\mathcal{X} - \mathcal{T}\|_F / \|\mathcal{T}\|_F$. One can easily verify its connection to the signal-to-noise ratio (SNR) by $SNR = -20 \log_{10} RSE$.

In the SiLRTC algorithm, the value of α_i is set to $1/3$ (3 is the mode number) and we only change the value of β_i. Let $\beta_i = \alpha_i/\gamma_i$. It is easy to see that when the γ_i's go to 0, the optimization problem in Eq. (15) converges to the original problem (7). In the FaLRTC algorithm, the parameters α_i's are set like in the SiLRTC algorithm and the parameters μ_i's are set as $\mu_i = \mu\frac{\alpha_i}{\sqrt{r_i}}$. Typically, μ is in the range $[1,\ 10]$, if the data is normalized as above. The rank parameters of the three heuristic algorithms follow the ranks of the tensor $\mathcal{T}$ unfolded along each mode.

We choose the percentage of randomly sampled elements as 30% and 50% respectively and present the comparison in Fig. 2 and Fig. 3.

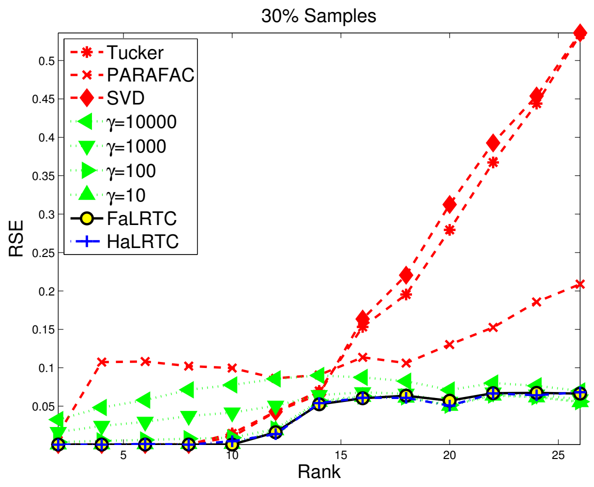

Fig. 2: The RSE comparison on the synthetic data. $I_1 = I_2 = I_3 = 50$. The ranks of the tensor are given by $r_1 = r_2 = r_3 = r$ where r is equivalent to $2, 4, 6, 8, \cdots, 26$ respectively. Tucker: Tucker model heuristic algorithm; Parafac: Parafac model based heuristic algorithm; SVD: the heuristic algorithm based on the SVD; $\gamma = 10000$, $\gamma = 1000$, $\gamma = 100$, and $\gamma = 10$ denote the proposed SiLRTC algorithm with $\beta_i = 1$ and $\gamma_i = \gamma$. We let $\mu = 5$ in FaLRTC algorithm. The sample percentage is 30%.

The brain MRI data is of size $181 \times 217 \times 181$. Although its ranks unfolded along three modes are $[164, 198, 165]$, they can decrease to $[35, 42, 36]$ if removing small singular values less than 0.01 percent of its Frobenius norm. Thus this is approximately an low rank data. We use the same normalization strategy above and report the comparison results in Table 1.

Results from Fig. 2, 3, and Tab. 1 show that the proposed convex formulation outperforms the three heuristic algorithms. The performance of the three heuristic algorithms is poor for high rank problems. We can also observe that the proposed convex formulation is able to recover the missing information using a small number of samples. Next we evaluate the SiLRTC algorithm using different parameters. We observe that the smaller the γ value, the closer the solution of Eq. (15) to the original problem in Eq. (9). The performance of FaLRTC is similar to the SiLRTC algorithm with $\gamma = 10$.

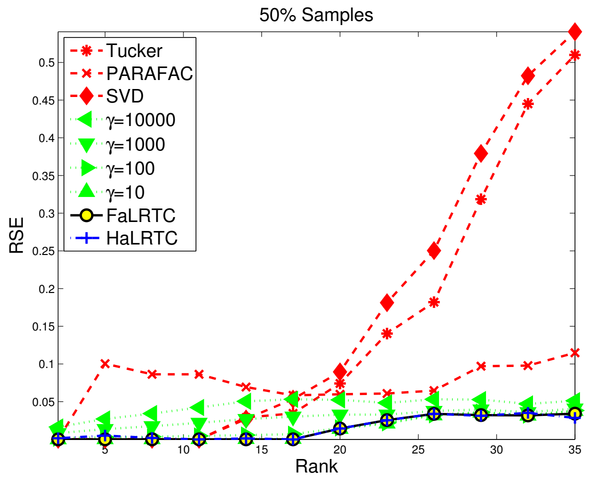

Fig. 3: See the caption of Fig. 2. The only difference is the sample percentage of 50%.

7.2 Model Comparison: Tensor Trace Norm Versus Matrix Trace Norm

In this subsection we compare the behavior of matrix and tensor completion in practical applications. We have not obtained meaningful theoretical bounds for our tensor completion algorithms. Given a tensor $\mathcal{T}$ with missing entries, we can unfold it along the ith mode into a matrix structure $\mathcal{T}_{(i)}$. Then the missing value estimation problem can be formulated as a low rank matrix completion problem:

$$\min_{X_\Omega = \mathcal{T}_{(i)\Omega}} : \ \|X\|_*. \tag{44}$$

We compare tensor completion and matrix completion on both the synthetic data and the real-world data and report results in Table 2 and Table 3. We show an example slice of the MRI data in Fig. 5.

Results from these two tables indicate that the proposed low rank tensor completion algorithm outperforms the low rank matrix completion algorithm, especially when the number of missing entries is large.

Next, we consider a natural idea for using the matrix completion technique for tensor completion. Without the unfolding operation, one can directly apply the matrix completion algorithm on each single slice of the tensor with missing entries. We call this as the slicewise matrix completion (S-MC). The comparisons on synthetic data and real data are reported in Table 4 and Fig. 4, which also show the advantages of tensor completion. In particular, Table 4 indicates that tensor completion can almost perfectly recover the missing values while matrix completion performs rather poorly in this case.

The key reason why the proposed algorithm outperforms the matrix completion based algorithms may be that tensor completion utilizes all information along all dimensions, while matrix completion only considers the constraints along two particular dimension.

7.3 Efficiency Comparison

This experiment concerns the efficiency of SiLRTC, FaLRTC, HaLRTC, and ADM-TR proposed in [14]. We com-

TABLE 1: The RSE comparison on the MRI brain data. Tucker: Tucker model heuristic algorithm; Parafac: Parafac model based heuristic algorithm; SVD: the heuristic algorithm based on the SVD; $\gamma = 10000$, $\gamma = 1000$, $\gamma = 100$, and $\gamma = 10$ correspond to the proposed SiLRTC algorithm with $\beta_i = 1$ and $\gamma_i = \gamma$. In the FaLRTC algorithm, μ is fixed as 5. We try different parameter values for three heuristic algorithms and report the best performance we obtain. The top, middle, and bottom parts of the table correspond to the sample percentage: 20%, 50%, and 80%, respectively.

RSE Comparison (10^{-4})

Samples	Tucker	Parafac	SVD	SiLRTC $\gamma = 10^4$	SiLRTC $\gamma = 10^3$	SiLRTC $\gamma = 10^2$	SiLRTC $\gamma = 10$	FaLRTC	HaLRTC
20%	371	234	274	309	47	22	21	17	16
50%	65	58	62	101	12	2	0	0	0
80%	21	45	16	40	4	1	0	0	0

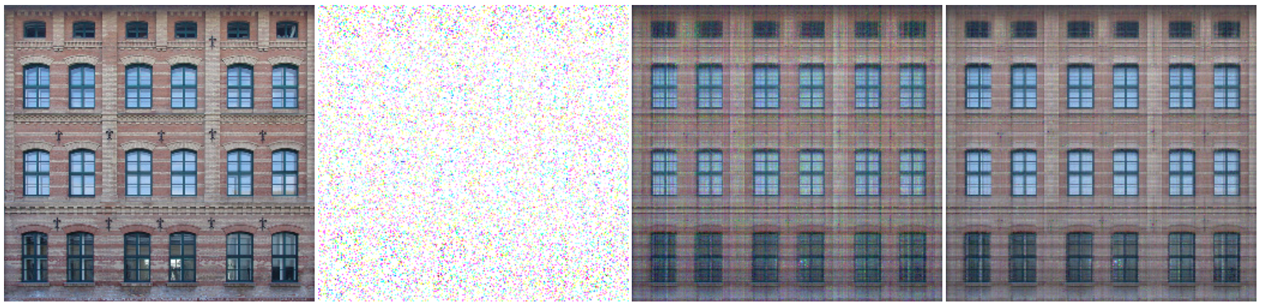
(a) Original (b) With 90% missing entries (c) S-MC: RSE=0.1889 (d) TC: RSE=0.1445

Fig. 4: The RSE comparison on the real image between tensor completion and S-MC. (a) The original image. (b) We randomly remove 90% entries from the original color image (3-mode tensor) and fill the value “255” on them. (c) The recovered result by S-MC which slices the image into red, green, and blue channels. (d) The recovered result by FaLRTC.

TABLE 2: The RSE comparison on the synthetic data with $r_i = 2$. MCk ($k = 1, 2, 3, \cdots$) denotes the performance of the solution of the problem (44) with $i = k$. We use the FaLRTC algorithm to do tensor completion with $\mu = 1$. The top, middle, and bottom parts of the table respond to different sizes of the synthetic data.

RSE Comparison (10^{-4}), Size $20 \times 20 \times 20$

Samples	MC1	MC2	MC3	TC
25%	1663	1782	1685	34
40%	247	258	241	2
60%	0	0	0	0

RSE Comparison (10^{-4}), Size $20 \times 20 \times 20 \times 20$

Samples	MC1	MC2	MC3	MC4	TC
20%	1875	1763	2011	1804	3
40%	92	102	97	88	0
60%	0	1	0	0	0

RSE Comparison (10^{-4}), Size $20 \times 20 \times 20 \times 20 \times 20$

Samples	MC1	MC2	MC3	MC4	MC5	TC
15%	1874	1830	1663	1502	1688	50
40%	125	119	131	114	136	0
60%	0	0	0	0	0	0

TABLE 3: The RSE comparison on the MRI brain data. $\mu = 5$ in the FaLRTC algorithm. MCk ($k = 1, 2, 3, \cdots$) denotes the performance of the solution of the problem (44) with $i = k$.

RSE Comparison (10^{-4})

Samples	MC1	MC2	MC3	TC
20%	670	781	862	**17**
50%	27	36	75	**0**

TABLE 4: The RSE comparison on the synthetic data with the size $60 \times 60 \times 60$. S-MCk ($k = 1, 2, 3, \cdots$) denotes the performance by applying matrix completion method on each slice. We use the FaLRTC algorithm to do tensor completion with $\mu = 1$.

RSE Comparison (10^{-4}), Samples percentage = 20%

	S-MC1	S-MC2	S-MC3	TC
$r_i's = 2$	2224	2317	2365	0
$r_i's = 4$	4887	4756	4441	1
$r_i's = 6$	6479	6789	6247	1

pare them on the synthetic data and the results are summarized in Table 5. We report the running time of four algorithms when achieving the same recovery accuracy measured by RSE from 10^{-1} to 10^{-6} respectively. We apply the efficient implementations of FaLRTC and HaLRTC by iteratively updating μ_i^k and ρ^k. We can observe from this table: 1) the larger the value of γ, the faster the SiLRTC algorithm, but a smaller value of γ can lead to a more accurate solution in the SiLRTC algorithm; 2) when three algorithms are configured to have the same recovery accuracy, the FaLRTC algorithm and the HaLRTC algorithm are much faster than the SiLRTC algorithm and the ADM-TR algorithm; 3) the FaLRTC algorithm is more efficient for obtaining the lower accuracy solutions while the HaLRTC algorithm is much more efficient for obtaining higher accuracy solutions. Fig. 6 shows the RSE curves of the FaLRTC algorithm and the HaLRTC algorithm in terms of the number of iterations and the computation time. We

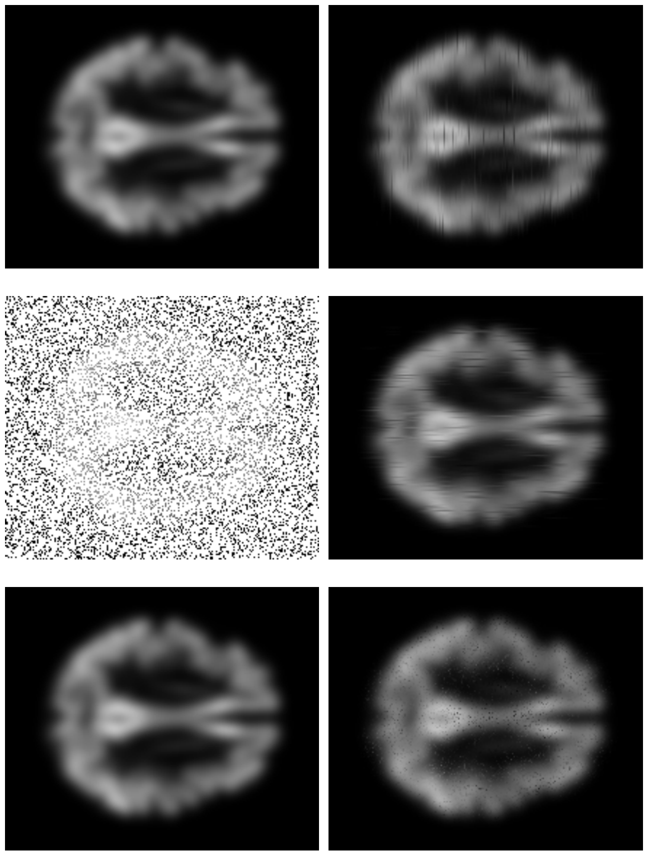
Fig. 5: The comparison of tensor completion and matrix completion. The left up image (one slice of the MRI) is the original; we randomly select pixels for removal shown in white in the left middle image; the left bottom image is the reconstruction by the proposed FaLRTC algorithm with $\mu = 5$; the right up, middle, and bottom images are respectively the results of matrix completion algorithm MC1, MC2, and MC3.

can observe from this figure that the FaLRTC algorithm converges very fast at the beginning but takes some time to obtain a high accuracy. Thus, FaLRTC is a good choice if an accuracy lower than 10^{-2} is acceptable and the HaLRTC algorithm is recommended if a higher accuracy is desired. In practice, one might combine the FaLRTC algorithm and the HaLRTC algorithm to improve the efficiency by running FaLRTC first and then HaLRTC.

7.4 Applications

In this section, we outline three potential application examples with three different types of data: Images, Videos, and reflectance data.

Images: Our algorithm can be used to estimate missing data in images and textures. For example, in Fig. 7 we show how missing pixels can be filled in a facade image. Note how our algorithm can propagate global structure even though a significant amount of information is missing.

Videos: The proposed algorithm may be used for video compression and video in-painting. The core idea of video compression is to remove individual pixels and to use tensor

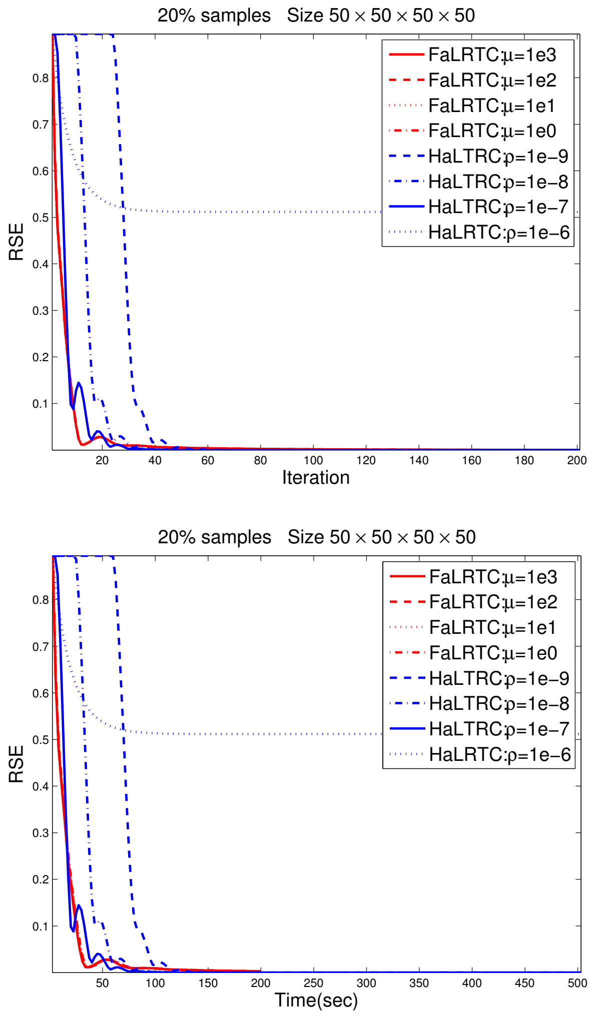

Fig. 6: The RSE comparison on the synthetic data. $I_1 = I_2 = I_3 = I_4 = 50$ and $r_1 = r_2 = r_3 = r_4 = 2$. All parameters are identical to the setting in the button part of Table 5.

completion to recover the missing information. Similarly, a user can eliminate unwanted pixels in the data and use the proposed algorithm to compute alternative values for the removed pixels. See Fig. 8 for an example frame of a video.

Reflectance data: The BRDF is the "Bidirectional Reflectance Distribution Function". The BRDF specifies the reflectance of a target as a function of illumination direction and viewing direction and can be interpreted as a 4 mode tensor. BRDFs of real materials can be acquired by complex appearance acquisition systems that typically require taking photographs of the same object under different lighting conditions. As part of the acquisition process, data can be missing or be unreliable, such as in the MIT BRDF data set. We use tensor completion to estimate the missing entries in reflectance data. See Fig. 9 for an example. More results are shown in the video accompanying this paper.

TABLE 5: The efficiency comparison on the synthetic data. We run the proposed SiLRTC algorithm with $\beta_i = 1$ and three different values of γ_i (assume all γ_i's are identical to γ as before): 10^4, 10^3, and 10^2. ADM-TR denotes the ADMM algorithm proposed in [14]. We report the best result we obtain by tuning the main parameters in this algorithm. The FaLRTC algorithm is run with four different values of μ (recall $\mu_i = \mu\frac{\alpha_i}{\sqrt{I_i}}$): 1000, 100, 10, and 1. μ_i^k is updated iteratively based on the scheme in **Section 5.3**, i.e., $\mu_i^K = \mu\frac{\alpha_i}{\sqrt{I_i}}$, $\mu_i^0 = 0.4\alpha_i\|\mathcal{X}_{(i)}\|$, and $\mu_i^k = ak^{-p} + b$ with $p = 1.15$ and $K = 200$. We run the HaLRTC algorithm with five different inputs for ρ: 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}. ρ is iteratively increased as discussed in **Section 6**, i.e., $\rho^{(0)} = \rho$, $\rho^{k+1} = t\rho^k$ where $t = 1.15$. We use the accuracy measured by RSE from 10^{-1} to 10^{-6} as the stopping criteria and report the running time when the required accuracy is obtained. The top and bottom parts of the table record the running time on two different synthetic data respectively. "-" denotes the running time exceeds 100 seconds in the top table and 600 seconds in the bottom table.

Running Time (sec) $I_1 = I_2 = I_3 = 100$ $r_1 = r_2 = r_3 = 2$ Samples 20%

RSE(SNR)	$10^{-1}(20db)$	$3\times10^{-2}(30db)$	$10^{-2}(40db)$	$10^{-3}(60db)$	$10^{-4}(80db)$	$10^{-5}(100db)$	$10^{-6}(120db)$
SiLRTC: $\gamma = 10^4$	8.4674	-	-	-	-	-	-
SiLRTC: $\gamma = 10^3$	83.7156	90.7864	-	-	-	-	-
SiLRTC: $\gamma = 10^2$	-	-	-	-	-	-	-
ADM-TR	15.9333	28.7841	65.8791	94.1238	-	-	-
FaLRTC: $\mu = 10^3$	**2.8477**	**3.5597**	8.5432	53.3950	-	-	-
FaLRTC: $\mu = 10^2$	2.9097	3.6372	8.7292	37.4629	66.1966	-	-
FaLRTC: $\mu = 10^1$	2.9532	3.6915	8.4905	36.9152	63.4942	68.6623	-
FaLRTC: $\mu = 10^0$	3.0084	3.7605	8.6491	37.6047	64.3041	69.5687	-
HaLRTC: $\rho = 10^{-8}$	9.2625	10.8970	11.9867	15.8007	19.6146	23.4286	27.2425
HaLRTC: $\rho = 10^{-7}$	5.4485	7.3555	8.9900	11.9867	16.0731	19.8871	23.7010
HaLRTC: $\rho = 10^{-6}$	3.5415	5.4485	**7.3555**	**10.8970**	**14.7110**	**18.5249**	**22.3389**
HaLRTC: $\rho = 10^{-5}$	-	-	-	-	-	-	-

Running Time (sec) $I_1 = I_2 = I_3 = I_4 = 50$ $r_1 = r_2 = r_3 = r_4 = 2$ Samples 20%

RSE(SNR)	$10^{-1}(20db)$	$3\times10^{-2}(30db)$	$10^{-2}(40db)$	$10^{-3}(60db)$	$10^{-4}(80db)$	$10^{-5}(100db)$	$10^{-6}(120db)$
SiLRTC: $\gamma = 10^4$	398.7844	-	-	-	-	-	-
SiLRTC: $\gamma = 10^3$	467.3572	554.4987	-	-	-	-	-
SiLRTC: $\gamma = 10^2$	-	-	-	-	-	-	-
ADM-TR	198.9887	226.7895	398.7835	588.9871	-	-	-
FaLRTC: $\mu = 10^3$	**25.3592**	**30.9946**	76.0776	349.3935	-	-	-
FaLRTC: $\mu = 10^2$	25.9213	31.6816	77.7640	336.9775	509.7864	-	-
FaLRTC: $\mu = 10^1$	26.3341	32.1861	79.0023	339.4174	514.9781	544.2382	-
FaLRTC: $\mu = 10^0$	26.4496	32.3273	79.3488	340.9059	517.2366	546.6251	-
HaLRTC: $\rho = 10^{-9}$	82.5930	107.6211	112.6268	150.1690	185.2085	217.7451	252.7845
HaLRTC: $\rho = 10^{-8}$	52.5592	72.5817	77.5873	112.6268	147.6662	182.7056	217.7451
HaLRTC: $\rho = 10^{-7}$	35.0394	37.5423	**67.5761**	**92.6042**	**130.1465**	**167.6887**	**200.2254**
HaLRTC: $\rho = 10^{-6}$	-	-	-	-	-	-	-

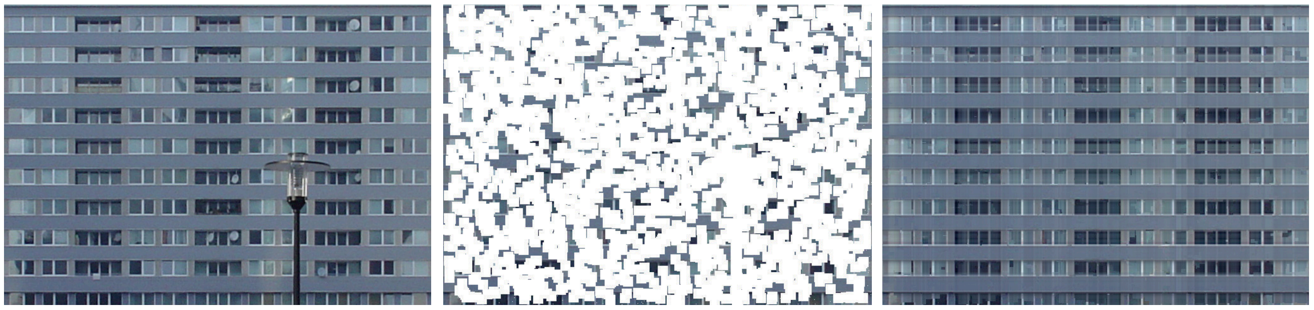

Fig. 7: Facade in-painting. The left image is the original image; we select the lamp and satellite dishes together with a large set of randomly positioned squares as the missing parts shown in white in the middle image; the right image is the result of the proposed completion algorithm.

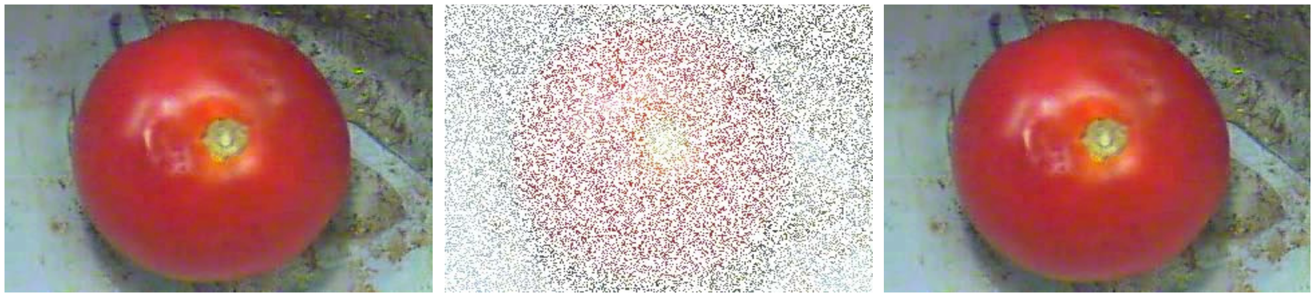

Fig. 8: Video completion. The left image (one frame of the video) is the original; we randomly select pixels for removal shown in white in the middle image; the right image is the result of the proposed LTRC algorithm.

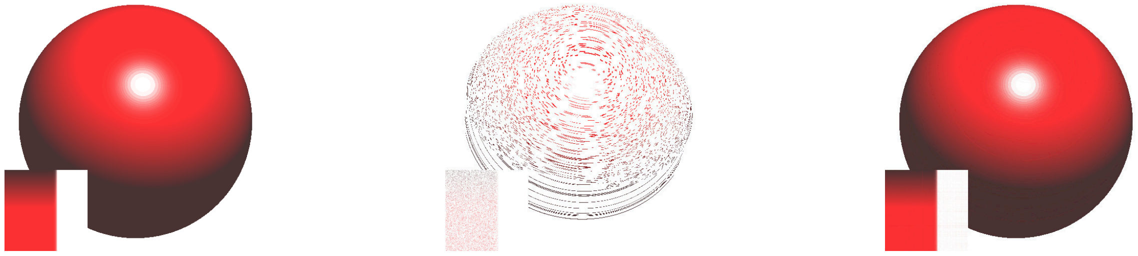

Fig. 9: The left image is a rendering of an original phong BRDF; we randomly select 90% of the pixels for removal shown in white in the middle image; the right image is the result of the proposed SiLRTC algorithm.

8 Conclusion

In this paper, we extend low rank matrix completion to low rank tensor completion. We propose tensor completion based on a novel definition of the trace norm for tensors together with three convex optimization algorithms: SiLRTC, FaLRTC, and HaLRTC to tackle the problem. The first algorithm, SiLRTC, is intuitive to understand and simple to implement. The latter two algorithms, FaLRTC and HaLRTC, are significantly faster than SiLRTC and use more advanced optimization techniques. Additionally, several heuristic algorithms are presented. The experiments show that the proposed algorithms are more stable and accurate in most cases, especially when the sample entries are very limited. Several application examples show the broad applicability of tensor completion in computer vision and graphics.

The proposed tensor completion algorithms assume that the data is of low rank. This may not be the case in certain applications. We plan to extend the theoretical results of Candès and Recht to the tensor case. We also plan to extend the proposed algorithms using techniques recently proposed in [8], [48].

Acknowledgments

This work was supported by NSF IIS-0812551, CCF-0811790, IIS-0953662, CCF-1025177, and NGA HM1582-08-1-0016. We thank anonymous reviewers for pointing out ADMM and recent following work on tensor completion. We thank S. Gandy for sharing the implementation of the ADM-TR algorithm.

References

[1] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. *ICML*, pages 17–24, 2007.
[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. *NIPS*, pages 243–272, 2007.
[3] F. R. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9:1019–1048, 2008.
[4] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. *SIGGRAPH*, pages 414–424, 2000.
[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method and multipliers. *Unpublished*, 2011.
[6] J. Cai. Fast singular value thresholding without singular value decomposition. *UCLA CAM Report*, 2010.
[7] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
[8] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Joural of the ACM*, 58(1):1–37, 2009.
[9] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.
[10] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2009.
[11] L. Elden. *Matrix Methods in Data Mining and Pattern Recognition*. 2007.
[12] M. Fazel. Matrix rank minimization with applications. *PhD thesis, Stanford University*.
[13] M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. *ACC*, pages 4734–4739, 2001.
[14] S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problem*, 27, 2011.
[15] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert. Quantum state tomography via compressed sensing. *http//arxiv.org/abs/0909.3304*, 2009.
[16] L. Guo, Y. Li, J. Yang, and L. Lu. Hole-filling by rank sparsity tensor decomposition for medical imaging. *IEICE*, pages 1–4, 2010.
[17] R. A. Harshman. Foundations of the parafac procedure: models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.
[18] C. J. Hillar and L. heng Lim. Most tensor problems are np hard. *CoRR*, abs/0911.1393, 2009.

[19] S. Ji, L. Sun, R. Jin, and J. Ye. Multi-label multiple kernel learning. *NIPS*, pages 777–784, 2008.

[20] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. *ICML*, pages 457–464, 2009.

[21] R. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE Transaction on Information Theory*, abs/0901.3150, 2010.

[22] N. Komodakis and G. Tziritas. Image completion using global optimization. *CVPR*, pages 417–424, 2006.

[23] T. Korah and C. Rasmussen. Spatiotemporal inpainting for recovering texture maps of occluded building facades. *IEEE Transactions on Image Processing*, 16:2262–2271, 2007.

[24] M. Kurucz, A. A. Benczur, and K. Csalogany. Methods for large scale svd with missing values. *KDD*, pages 31–38, 2007.

[25] N. Li and B. Li. Tensor completion for on-board compression of hyperspectral images. *ICIP*, pages 517–520, 2010.

[26] Y. Li, J. Yan, Y. Zhou, and J. Yang. Optimum subspace learning and error correction for tensors. *ECCV*, 2010.

[27] Y. Li, Y. Zhou, J. Yan, J. Yang, and X. He. Tensor error correction for corrupted values in visual data. *ICIP*, pages 2321–2324, 2010.

[28] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Technical Report UILU-ENG-09-2215, UIUC, (arXiv: 1009.5055)*, 2009.

[29] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *ICCV*, pages 2114–2121, 2009.

[30] S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1):321–353, 2009.

[31] A. Nemirovski. *Efficient methods in convex programming*. 1995.

[32] Y. Nesterov. A method of solving a convex programing problem with convergence rate $o(1/k^2)$. *Soviet Mathematics Doklady*, 27(2), 1983.

[33] Y. Nesterov. Introductory lectures on convex programming. *Lecture Notes*, pages 119–120, 1998.

[34] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathemtaical Programming*, 103(1):127–152, 2005.

[35] T. K. Pong, P. Tseng, S. Ji, and J. Ye. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20(6):3465–3489, 2010.

[36] B. Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 11:2287–2322, 2010.

[37] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

[38] M. Signoretto, L. D. Lathauwer, and J. A. K. Suykens. Nuclear norms for tensors and their use for convex multilinear estimation. *Submitted to Linear Algebra and Its Applications*, 2010.

[39] N. Srebro, J. D. M. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. *NIPS*, pages 1329–1336, 2005.

[40] J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11:623–625, 1998.

[41] K. C. Toh, M. J. Todd, and R. H. Tutuncu. Sdpt3: a matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[42] M. Tomasi and T. Kanade. Shape and motion from image stream under orthography: a factorization method. *International Journal of Computer Vision*, 9:137–154, 1992.

[43] R. Tomioka, K. Hayashi, and H. Kashima. Estimation of low-rank tensors via convex optimization. *arxiv.org/abs/1010.0789*, 2011.

[44] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17:520–525, 2001.

[45] P. Tseng. Convergence of block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory Application*, 109:475–494, 2001.

[46] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966.

[47] J. Yan, J. Liu, Y. Li, Z. Niu, and Y. Liu. Visual saliency detection via rank-sparsity decomposition. *ICIP*, pages 1089–1092, 2010.

[48] Z. Zhou, X. Li, J. Wright, E. J. Candès, and Y. Ma. Stable principal component pursuit. *CoRR*, abs/1001.2363, 2010.

Ji Liu is currently a graduate student of the Department of Computer Sciences at University Wisconsin-Madison. He received his bachelor degree in automation from University of Science and Technology of China in 2005 and master degree in Computer Science from Arizona State University in 2010. His research interests include optimization, machine learning, computer vision, and graphics. He won the KDD best research paper award honorable mention in 2010.

Przemyslaw Musialski received a MSc degree from the Bauhaus University Weimar in 2007 (Germany) and a PhD degree from the Vienna University of Technology in 2010 (Austria). From 2007 till 2010 he was with VRVis Research Center in Vienna, where he was working on image processing and image-based urban modeling. In 2010 he continued this work at the Vienna University of Technology. Since 2011 he is postdoctoral scholar at the Arizona State University, where he is conducting research on image and texture processing.

Peter Wonka received the MS degree in urban planning and the doctorate in computer science from the Technical University of Vienna. He is currently with Arizona State University (ASU). Prior to coming to ASU, he was a postdoctorate researcher at the Georgia Institute of Technology for two years. His research interests include various topics in computer graphics, visualization, and image processing.

Jieping Ye is an Associate Professor of the Department of Computer Science and Engineering at Arizona State University. He received his Ph.D. in Computer Science from University of Minnesota, Twin Cities in 2005. His research interests include machine learning, data mining, and biomedical informatics. He won the outstanding student paper award at ICML in 2004, the SCI Young Investigator of the Year Award at ASU in 2007, the SCI Researcher of the Year Award at ASU in 2009, the NSF CAREER Award in 2010, and the KDD best research paper award honorable mention in 2010 and 2011.