IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2018.DOI

Nonnegative tensor completion via low-rank Tucker decomposition: model and algorithm

BILIAN CHEN[1], TING SUN[1], ZHEHAO ZHOU[1], YIFENG ZENG[1,2], AND LANGCAI CAO.[1]
[1]Department of Automation, Xiamen University, Xiamen 361005, China.
[2]School of Computing, Teesside University, UK.

Corresponding authors: Bilian Chen (e-mail: blchen@xmu.edu.cn) and Yifeng Zeng (e-mail: yfzeng@xmu.edu.cn).

This work was supported in part by the National Natural Science Foundation of China (Grants No. 61772442 and 11671335) and the Science Foundation (for Youth) of the Science and Technology Commission of the Central Military Commission (CMC), China.

ABSTRACT We consider the problem of low-rank tensor decomposition of incomplete tensors that has applications in many data analysis problems such as recommender systems, signal processing, machine learning and image inpainting. In this paper, we focus on nonnegative tensor completion via low-rank Tucker decomposition for dealing with it. The speciality of our model is that the ranks of nonnegative Tucker decomposition are no longer constants, while they all become a part of the decisions to be optimized. Our solving approach is based on penalty method and block coordinate descent method with prox-linear updates for regularized multiconvex optimization. We demonstrate the convergence of our algorithm. Numerical results on the three image datasets show that the proposed algorithm offers competitive performance compared with other existing algorithms even though the data is highly sparse.

INDEX TERMS Nonnegative tensor completion, nonnegative Tucker decomposition, adjustable core tensor size, block coordinate descent

I. INTRODUCTION

DATA collected in real life may not be completely accurate due to various factors such as collection difficulties, noise interference, and manually identified unwanted outliers. As such, some data may be missing and should be completed. These can all lead to the completion problem for missing data. Most data can be expressed in the form of matrix, since matrix is a commonly powerful tool to express and store these data. Hence, many practical problems can be transformed to study matrix completion [1]–[3]. Usually when looking for a solution, the matrix is assumed to be low rank [1], [4], [5]. The low rank matrix completion has a wide range of applications. For example in image processing, low resolution input images can be reconstructed into high resolution images by solving image interpolation problem using matrix completion and recovery method [6]. In recommendation systems, after users submit their scores on the corresponding items to the database, the system makes recommendations to other users by analyzing these data. However, in reality, users typically score only several items. At this point, the system must estimate users' preferences for non-scoring items, which can be realized by performing matrix completion [7], [8]. In the medical field, regular medical records aid medical staff in analyzing and monitoring patient health status. However, these records are usually incomplete due to unpunctuality and absence of patients. By using the method of matrix decomposition under some latent factors to obtain the complete medical record data, experimental results indicate that the proposed algorithm can perform well compared with existing methods [9].

With the rapid development of science and technology, the generation of abundant data has exceeded people's imagination, resulting in redundant storage space, increasing computational cost, and other issues. The form of matrix data may not fully demonstrate the essential characteristics of high-dimensional data. Therefore, how to find a concise representation for these data has gained more and more attention. Under this circumstance, using tensors to express high-order data has become a trend in data expression. The tensors are a high-dimensional extension of vectors and matrices. Urgent analysis of high dimensional data now widely appears in signal processing, machine learning and data analysis. In the present study, we consider the problem wherein only part of the data is obtained when observing or collecting. We

also discuss the effectiveness of the process of missing data completion, and we aim to solve this problem via tensor completion. Tensor completion is a powerful tool that is used to estimate or recover missing values in multi-way data, which arises in various research areas, such as recommendation systems [10], multi-class learning [6], data mining [11], computer vision [12], [13], and traffic [14]–[16].

By taking advantage of both low-rankness and nonnegative factors, Xu et al. [17] had shown that superior results can be generally obtained, compared with just using one of the two aspects for nonnegative matrix completion problems. Following by the spirit of [17], we, in this paper, consider the nonnegative tensor completion problem for nonnegative data, and we solve it by performing nonnegative Tucker decomposition, which decomposes the high-order tensor into a low-rank core tensor in every mode multiplied by a matrix, where the core tensor and the matrices are all nonnegative. A Tucker decomposition with a rank of $(r_1, r_2, \ldots, r_d)$ means that the size of the core tensor is $(r_1 \times r_2 \times \cdots \times r_d)$, where the parameters $(r_1, r_2, \ldots, r_d)$ are predetermined traditionally. Nonetheless, this is limited in some applications. Therefore, we consider a new nonnegative Tucker decomposition model with an adjustable core tensor size when the sum of the tensor rank ($\sum_{i=1}^{d} r_i = c$) is fixed. Then, the most reasonable dimension of size $(r_1, r_2, \ldots, r_d)$ is selected automatically through an optimization process. In particular, we resort to a classical block coordinate descent type search method for regularized multiconvex optimization, which suits well for our model. Finally, we report how the block coordinate descent method can be applied to solve our problem, along with a convergence result of our algorithm.

The rest of the paper is organized as follows. In Section II, we discuss some relevant works on the tensor completion problem. In Section III, we introduce some basic operations on tensors, block coordinate descent method, and nonnegative tensor completion problem. In Section IV, we present a new model for the nonnegative tensor completion problem on the basis of the nonnegative Tucker decomposition with unspecified core tensor size. Then, we solve it on the basis of the penalty method and the block coordinate descent method. In Section V, we demonstrate the performance of our new model and method using several image datasets. Finally we conclude our work in Section VI.

II. RELATED WORK

On the task of filling in incomplete data (i.e., data with missing, unknown or unreliable values) for many high-dimensional data, also known as the tensor completion problem, one can transform it as a low rank tensor decomposition (or approximation) problem, since the normal structural assumption on a tensor that makes the completion problem well posed is that the tensor has low rank in every mode [18]. Usually, CANDECOMP/PARAFAC (CP) decomposition and Tucker decomposition are employed. The technique can well recover the incomplete data either on synthetic data or on real-world visual data, see e.g., [13], [18]–[20].

Royer et al. [21] implemented the CP decomposition using the conjugate gradient and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithms for tensor completion problems. To handle the incomplete data, Yokota et al. [20] proposed a new low-rank smooth CP decomposition method that is different from the existing methods. In their proposed method, they considered two types of smoothness constraint. Instead of setting the upper bound of the expected rank of the tensor, their algorithm increased the number of components gradually until the optimal rank was reached. Results indicated that the efficiency of their method on visual data outperformed some other popular algorithms. However, their method does not guarantee global convergence. To handle high-dimensional, large-scale datasets and applications, Karlsson et al. [22] proposed novel parallel algorithms for tensor completion in the CP format. More recently, Kaya and Uçar [23] proposed a novel computational framework for reducing the cost of a core operation in computing the CP decomposition for sparse tensors. However, they do not address the issue on selecting an appropriate rank for tensor completion via CP decomposition.

A similar issue also exists in tensor completion via Tucker decomposition, i.e., determining the rank of the core tensor in the presence of missing entries and noise. This issue is challenging, because it is already difficult to compute a Tucker decomposition of rank $(r_1, r_2, \ldots, r_d)$, where r_k is less than the column rank of mode-k unfolding of a given tensor for one or more k [24]. In this case, the methods based on the Tucker decomposition usually perform poorly. Therefore, many researchers have investigated on this issue. Chen et al. [25] proposed a new computational model for a low-rank Tucker decomposition where the configuration and the size of the core become part of the decisions to be optimized. Their algorithm guaranteed to converge to a stationary point of their model. To determine the multilinear rank of high-dimensional datasets automatically, Yang et al. [26] used a group-based, log-sum penalty functional for placing structural sparsity over the core tensor. Then, they proposed an iterative reweighted algorithm to decompose an incomplete tensor into a concise Tucker decomposition. Meanwhile, Filipović and Jukić [18] proposed a simple algorithm for Tucker decomposition of a sparse tensor and its application to low-n-rank tensor completion. They demonstrated that the proposed algorithm performs well even when the ranks are over estimated. However, no theoretical guarantee has been provided for their proposed method. Recently, Sejoon Oh et al. [27] proposed the P-TUCKER, a scalable Tucker decomposition method for sparse tensors, which performs alternating least squares with a row-wise update rule in a fully parallel way. Hence, it saves much computational time and substantially reduces memory requirements for updating factor matrices.

On the basis of the relevant research, we consider the nonnegative tensor completion problem via Tucker decomposition. In relation to this, we propose a new model for the problem concerned that can determine the size of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

core tensor automatically. Finally, we provide a theoretical guarantee for our algorithm.

III. PRELIMINARIES

A. OVERVIEW OF TENSORS

In this paper, we use calligraphic letters, capital letters, boldface lowercase letters, and non-bold lowercase letters to denote tensors, matrices, vectors, and scalars. For example, tensor $\mathcal{G}$, matric A, vector $\boldsymbol{y}$ and scalar i. We use the subscripts to denote the element of a tensor, a matrix, or a vector, e.g., $\mathcal{G}_{ijk}$ represents the (i,j,k)-th element of the tensor $\mathcal{G}$, A_{ij} represents the (i,j)-th element of matrix A, y_i represents the i-th element of vector $\boldsymbol{y}$. Below we list some tensor operations and properties, most of which follow from the survey paper [24]. For more details, please also refer to [24].

A tensor is a multidimensional array, and the order of a tensor is the number of its dimensions, also known as the ways or the modes of a tensor. Particularly, a vector is a 1st order tensor, and a matrix is a 2nd order tensor. For a d-th order tensor $\mathcal{G} = (\mathcal{G}_{i_1 i_2 \ldots i_d}) \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$, where $d \geq 3$, it has d modes, namely, mode-1, mode-2, $\ldots$, mode-d. Denote the mode-k matricization (or unfolding) of tensor $\mathcal{G}$ to be $G_{(k)}$, then the $(i_1, i_2, \ldots, i_d)$-th entry of tensor $\mathcal{G}$ is mapped to the (i_k, j)-th entry of matrix $G_{(k)} \in \mathbb{R}^{n_k \times \prod_{\ell \neq k} n_\ell}$, where

$$j = 1 + \sum_{1 \leq \ell \leq d, \ell \neq k} (i_\ell - 1) \prod_{1 \leq t \leq \ell - 1, t \neq k} n_t.$$

One important tensor operation is the multiplication of a tensor by a matrix. The k-mode product of tensor $\mathcal{G}$ by a matrix $U \in \mathbb{R}^{m \times n_k}$, denoted by $\mathcal{G} \times_k U$, is a tensor in $\mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{k-1} \times m \times n_{k+1} \times \cdots \times n_d}$, which is defined by

$$(\mathcal{G} \times_k U)_{i_1 i_2 \ldots i_{k-1} \ell \, i_{k+1} \ldots i_d} = \sum_{i_k = 1}^{n_k} \mathcal{G}_{i_1 i_2 \ldots i_{k-1} i_k i_{k+1} \ldots i_d} U_{\ell i_k}.$$

For better understanding this multiplication, we can rewrite the equation in terms of tensor unfolding, i.e.,

$$\mathcal{Y} = \mathcal{G} \times_k U \Longleftrightarrow Y_{(k)} = U G_{(k)}.$$

Moreover, the k-mode product of tensor $\mathcal{G}$ by multiple matrices $U^{(k)} \in \mathbb{R}^{m_k \times n_k}$, $k = 1, 2, \ldots, d$ can also be expressed by the matrix Kronecker product as follows:

$$\begin{aligned} &\mathcal{Y} = \mathcal{G} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_d U^{(d)} \\ \Longleftrightarrow\ &Y_{(k)} = U^{(k)} G_{(k)} \left(U^{(d)} \otimes \cdots \otimes U^{(k+1)} \right. \\ &\left. \otimes U^{(k-1)} \otimes \cdots \otimes U^{(1)} \right)^T, \end{aligned}$$

for any $k = 1, 2, \ldots, d$.

Analogous to the Frobenius norm of a matrix, the Frobenius norm of tensor $\mathcal{G}$ is the usual 2-norm, defined by

$$\|\mathcal{G}\|_F := \sqrt{\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} \mathcal{G}_{i_1 i_2 \ldots i_d}{}^2}.$$

We uniformly denote the 2-norm for vectors, and the Frobenius norm for matrices and tensors, all by notation $\|\cdot\|_F$ throughout this paper. Moreover, we denote the spectral norm for matrices by $\|\cdot\|_2$.

The k-rank of tensor $\mathcal{G}$, denoted by $\text{rank}_k(\mathcal{G})$, is the column rank of mode-k unfolding $\mathcal{G}_{(k)}$, i.e., $\text{rank}_k(\mathcal{G}) = \text{rank}\,(\mathcal{G}_{(k)})$. A d-th order tensor whose $\text{rank}_k(\mathcal{C}) = r_k$ for $k = 1, 2, \ldots, d$, is briefly called a rank-$(r_1, r_2, \ldots, r_d)$ tensor. One important tensor decomposition is Tucker decomposition. It decomposes a tensor $\mathcal{G}$ in the form of $\mathcal{G} = \mathcal{C} \times_1 A^{(1)} \times_2 A^{(2)} \cdots \times_d A^{(d)}$, where $\mathcal{C} \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_d}$ is called the core tensor, and $A^{(i)} \in \mathbb{R}^{n_i \times r_i}, i = 1, 2, \ldots, d$, are called factor matrices.

B. A BLOCK COORDINATE DESCENT ALGORITHM FOR REGULARIZED MULTICONVEX OPTIMIZATION

Let us focus on the regularized multiconvex optimization problem and its corresponding block coordinate descent method discussed in Xu and Yin [19].

$$\min_{\boldsymbol{x} \in \chi} F(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s) \equiv f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s) + \sum_{i=1}^{s} v_i(\boldsymbol{x}_i),$$

where $\boldsymbol{x}$ is decomposed into s block variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s$, χ is a set of feasible points, assuming that it is a closed, block multi-convex subset of $\mathbb{R}^n$, f is a differentiable block multi-convex function, $v_i, i = 1, 2, \ldots, s$, are extended-value convex functions. The idea of Gauss-Seidel iterative block coordinate descent (BCD) method is to minimize F by cycling over each of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s$ while fixing the remaining block variables. Let $\boldsymbol{x}_i^k$ represent the value of $\boldsymbol{x}_i$ after its kth update, and let

$$f_i^k(\boldsymbol{x}_i) \triangleq f(\boldsymbol{x}_1^k, \ldots, \boldsymbol{x}_{i-1}^k, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}^{k-1}, \ldots, \boldsymbol{x}_s^{k-1}),$$

$$\chi_i^k \triangleq \chi(\boldsymbol{x}_1^k, \ldots, \boldsymbol{x}_{i-1}^k, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}^{k-1}, \ldots, \boldsymbol{x}_s^{k-1}),$$

for all i and k. One choice of update schemes for $\boldsymbol{x}_i^k$ is to find the optimal solution for the prox-linear subproblem

$$\begin{aligned} \boldsymbol{x}_i^k = &\arg\min_{\boldsymbol{x}_i \in \chi_i^k} \langle \hat{\boldsymbol{g}}_i^k, \boldsymbol{x}_i - \hat{\boldsymbol{x}}_i^{k-1} \rangle \\ &+ \tfrac{L_i^{k-1}}{2} \left\| \boldsymbol{x}_i - \hat{\boldsymbol{x}}_i^{k-1} \right\|_F^2 + v_i(\boldsymbol{x}_i), \end{aligned} \tag{1}$$

where parameter $L_i^{k-1} > 0$, and

$$\hat{\boldsymbol{x}}_i^{k-1} = \boldsymbol{x}_i^{k-1} + \omega_i^{k-1}(\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_i^{k-2})$$

denotes an extrapolated point, $\omega_i^{k-1} \geq 0$ is the extrapolation weight, and $\hat{\boldsymbol{g}}_i^k = \nabla f_i^k(\hat{\boldsymbol{x}}_i^{k-1})$ is the block-partial gradient of f at $\hat{\boldsymbol{x}}_i^{k-1}$. It has been found that the updating rule (1) on all or some blocks could result in achieving lower objective values, and consuming less computational efforts as well. Moreover, it can be proved that Algorithm 1 has convergence guarantee under some mild conditions. With these benefits, we will choose the prox-linear rule for handling our new model later. One is referred to [19] for more details.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

Algorithm 1: BCD method with prox-linear updates

1: Initial points $(\boldsymbol{x}_1^{-1},\ldots,\boldsymbol{x}_s^{-1})=(\boldsymbol{x}_1^0,\ldots,\boldsymbol{x}_s^0)$.
2: **for** $k=1,2,\ldots$ **do**
3: **for** $i=1,2,\ldots,s$ **do**
4: $\boldsymbol{x}_i^k=\arg\min_{\boldsymbol{x}_i\in X_i^k}\langle \hat{\boldsymbol{g}}_i^k,\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i^{k-1}\rangle + \frac{L_i^{k-1}}{2}\left\|\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i^{k-1}\right\|_F^2+v_i(\boldsymbol{x}_i)$
5: **end for**
6: **if** stopping criterion is satisfied **then**
7: Return $(\boldsymbol{x}_1^k,\ldots,\boldsymbol{x}_s^k)$
8: **end if**
9: **end for**

C. NONNEGATIVE TENSOR COMPLETION PROBLEM

One traditional way for dealing with nonnegative tensor completion problem is through nonnegative Tucker decomposition (NTD), which gives

$$\begin{aligned}\min\quad & \left\|P_\Omega(\mathcal{F}-\mathcal{G}\times_1 A^{(1)}\times_2 A^{(2)}\times_3\cdots\times_d A^{(d)})\right\|_F^2\\ \text{s.t.}\quad & \mathcal{G}\in\mathbb{R}^{r_1\times r_2\cdots\times r_d},\ \mathcal{G}\geq 0,\\ & A^{(i)}\in\mathbb{R}^{n_i\times r_i},\ A^{(i)}\geq 0, i=1,2,\ldots,d,\end{aligned}$$

where $\mathcal{F}\in\mathbb{R}^{n_1\times n_2\times\cdots\times n_d}$ is the observed sparse tensor, $\Omega\subset[n_1]\times[n_2]\times\cdots\times[n_d]$ is the index set of the known data in $\mathcal{F}$, $P_\Omega(\mathcal{X})$ keeps the data of $\mathcal{X}$ in Ω and sets the rest ones to zero. The core tensor $\mathcal{G}$ and factor matrices $A^{(i)}, i=1,2,\ldots,d$ are nonnegative, here each $r_i, i=1,2,\ldots,d$ is a pre-specified integer. In order to solve the problem, we transform it to the equivalent problem:

$$\begin{aligned}\min\quad & \left\|\mathcal{M}-\mathcal{G}\times_1 A^{(1)}\times_2 A^{(2)}\times_3\cdots\times_d A^{(d)}\right\|_F^2\\ \text{s.t.}\quad & P_\Omega(\mathcal{M})=P_\Omega(\mathcal{F})\\ & \mathcal{G}\in\mathbb{R}^{r_1\times r_2\cdots\times r_d},\ \mathcal{G}\geq 0,\\ & A^{(i)}\in\mathbb{R}^{n_i\times r_i},\ A^{(i)}\geq 0, i=1,2,\ldots,d.\end{aligned}$$

where $\mathcal{M}$ is a tensor with the same dimension as $\mathcal{F}$. We may apply BCD method, i.e., alternatively update blocks $\mathcal{G}, A^{(i)}, i=1,2,\ldots,d$ and $\mathcal{M}$. Specifically, we update $\mathcal{M}$ by the following rule:

$$\mathcal{M}=P_\Omega(\mathcal{F})+P_{\Omega^c}(\mathcal{G}\times_1 A^{(1)}\times_2 A^{(2)}\times_3\cdots\times_d A^{(d)}),$$

where Ω^c is the complement of Ω. The updating rule guarantees the constraint $P_\Omega(\mathcal{M})=P_\Omega(\mathcal{F})$ that has been satisfied in the whole iterative process.

IV. NONNEGATIVE TENSOR COMPLETION PROBLEM BASED ON NTD WITH UNSPECIFIED CORE TENSOR SIZE

Xu and Yin [19] discussed the use of the BCD algorithm to solve the nonnegative tensor completion problem, but the core tensor size is given in advanced in their paper. Hence, we generalize the tensor completion model in this section. In fact, we propose a new model for nonnegative tensor completion via NTD without pre-specifying the size of the core tensor. The dimension of each mode for the core is no longer a constant, it becomes a variable that also needs to be optimized, which is the speciality of our new model. Several techniques, including BCD method and penalty method, are proposed to solve the new model as well.

A. THE PROBLEM FORMULATION

Given a dth order nonnegative tensor $\mathcal{F}\in\mathbb{R}^{n_1\times n_2\cdots\times n_d}$, we hope to use a low rank nonnegative core tensor $\mathcal{G}\in\mathbb{R}^{r_1\times r_2\cdots\times r_d}$ and d nonnegative matrices $A^{(i)}$ to represent $\mathcal{F}, 1\leq r_i\leq n_i, i=1,2,\ldots,d$. Our work is going to find the best approximation of $\mathcal{F}$, where all the i-rank of the core tensor $r_i, i=1,2,\ldots,d$ are all variables that need to be optimized. Let c be a given constant integer number. We force that $\sum_{i=1}^d r_i=c$ to prevent r_i from being too large in general. Hence, we would like to consider the nonnegative tensor completion problem based on a low rank NTD, the new optimization problem is:

$$\begin{aligned}\min\quad & \tfrac{1}{2}\left\|\mathcal{M}-\mathcal{G}\times_1 A^{(1)}\times_2 A^{(2)}\times_3\cdots\times_d A^{(d)}\right\|_F^2\\ \text{s.t.}\quad & P_\Omega(\mathcal{M})=P_\Omega(\mathcal{F}),\\ & \mathcal{G}\in\mathbb{R}^{r_1\times r_2\cdots\times r_d},\ \mathcal{G}\geq 0,\\ & A^{(i)}\in\mathbb{R}^{n_i\times r_i},\ A^{(i)}\geq 0, i=1,2,\ldots,d,\\ & r_i\in\mathbb{Z},\ 1\leq r_i\leq n_i,\ i=1,2,\ldots,d,\\ & \textstyle\sum_{i=1}^d r_i=c.\end{aligned}$$

Because the second and third constraints contain both block variables $\mathcal{G}, A^{(i)}$ and rank variables r_i, the problem is difficult to solve directly. In order to separate these variable constraints, we adopt the similar separating technique that developed in [25] by adding d new block variables $T^{(i)}\in\mathbb{R}^{m_i\times m_i}$, where they used it to deal with a specific low-rank Tucker decomposition problem with orthogonal constraints. That is, we enlarge the column dimension of block $A^{(i)}$ from r_i to m_i, and then multiply diagonal matrices $T^{(i)}$ whose diagonal element is 0 or 1 to select only r_i columns, where

$$\begin{aligned}&m_i=\min\{n_i,c\}, i=1,2,\ldots,d,\\ &T^{(i)}=\text{diag}\,(t^{(i)}),\ t^{(i)}\in\{0,1\}^{m_i},\end{aligned}$$

and

$$\sum_{j=1}^{m_i} t_j^{(i)}=r_i \text{ for } i=1,2,\ldots,d.$$

That is, if $T_{j,j}^{(i)}=1$, we select the j-th column in $A^{(i)}$, otherwise, $T_{j,j}^{(i)}=0$ indicates that the column is not selected. After the block variables are added, the objective function becomes:

$$\begin{aligned}(NT)\quad \min\quad & \tfrac{1}{2}\left\|\mathcal{M}-\mathcal{G}\times_1 (A^{(1)}T^{(1)})\times_2 (A^{(2)}T^{(2)})\times_3\right.\\ & \left.\cdots\times_d (A^{(d)}T^{(d)})\right\|_F^2\\ \text{s.t.}\quad & P_\Omega(\mathcal{M})=P_\Omega(\mathcal{F}),\\ & \mathcal{G}\in\mathbb{R}^{m_1\times m_2\cdots\times m_d},\ \mathcal{G}\geq 0,\\ & A^{(i)}\in\mathbb{R}^{n_i\times m_i},\ A^{(i)}\geq 0, i=1,2,\ldots,d,\\ & t^{(i)}\in\{0,1\}^{m_i},\\ & \textstyle\sum_{j=1}^{m_i} t_j^{(i)}\geq 1,\\ & \textstyle\sum_{i=1}^{d}\sum_{j=1}^{m_i} t_j^{(i)}=c.\end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

B. THE PENALTY METHOD

To solve the optimization problem (NT) using the BCD algorithm, we put the undivided constraint $\sum_{i=1}^{d}\sum_{j=1}^{m_i} t_j^{(i)} = c$ to the objective function, resulting in a penalty function:

$$\begin{aligned} &p(\lambda, \mathcal{M}, \mathcal{G}, A^{(1)}, \ldots, A^{(d)}, T^{(1)}, \ldots, T^{(d)}) \\ &= \tfrac{1}{2}\left\|\mathcal{M} - \mathcal{G} \times_1 (A^{(1)}T^{(1)}) \times_2 (A^{(2)}T^{(2)}) \times_3 \cdots \right. \\ &\quad \left. \times_d (A^{(d)}T^{(d)})\right\|_F^2 + \lambda\left(\textstyle\sum_{i=1}^{d}\sum_{j=1}^{m_i} t_j^{(i)} - c\right)^2. \end{aligned}$$

where $\lambda > 0$ is the penalty parameter. Thus, the penalty model becomes

$$\begin{aligned} (PM) \quad \min \quad & \tfrac{1}{2}\left\|\mathcal{M} - \mathcal{G} \times_1 (A^{(1)}T^{(1)}) \times_2 (A^{(2)}T^{(2)}) \right. \\ & \left. \times_3 \cdots \times_d (A^{(d)}T^{(d)})\right\|_F^2 \\ & + \lambda\left(\textstyle\sum_{i=1}^{d}\sum_{j=1}^{m_i} t_j^{(i)} - c\right)^2 \\ \text{s.t.} \quad & P_\Omega(\mathcal{M}) = P_\Omega(\mathcal{F}), \\ & \mathcal{G} \in \mathbb{R}^{m_1 \times m_2 \cdots \times m_d},\ \mathcal{G} \geq 0, \\ & A^{(i)} \in \mathbb{R}^{n_i \times m_i},\ A^{(i)} \geq 0, i = 1, 2, \ldots, d, \\ & t^{(i)} \in \{0,1\}^{m_i}, \\ & \textstyle\sum_{j=1}^{m_i} t_j^{(i)} \geq 1. \end{aligned}$$

The BCD method is called for solving this optimization problem since the block variables are separated now. Let us discuss how to solve the subproblems in implementing the BCD method with prox-linear updates [19] that presented in Algorithm 1.

1) Updating factor matrices

In this part, we wish to optimize $(A^{(i)}, T^{(i)})$ while other block variables $(A^{(j)}, T^{(j)}), j = 1, 2, \cdots, i-1, i+1, \cdots, d$ and the tensors $\mathcal{M}$ and $\mathcal{G}$ are fixed. Consider the block $(A^{(i)}, T^{(i)})$ as a whole and let $Z^{(i)} = A^{(i)}T^{(i)}$, we have

$$\begin{aligned} h(Z^{(i)}) &= \tfrac{1}{2}\left\|\mathcal{M} - \mathcal{G} \times_1 (A^{(1)}T^{(1)}) \times_2 (A^{(2)}T^{(2)}) \times_3 \right. \\ &\quad \left. \cdots \times_d (A^{(d)}T^{(d)})\right\|_F^2 \\ &= \tfrac{1}{2}\left\|\mathcal{G} \times_1 Z^{(1)} \times_2 Z^{(2)} \times_3 \cdots \times_d Z^{(d)} - \mathcal{M}\right\|_F^2, \end{aligned}$$

then its gradient is computed as

$$\begin{aligned} \nabla_{Z^{(i)}} h &= \left(\mathcal{G} \times_1 Z^{(1)} \times_2 Z^{(2)} \times_3 \cdots \times_d Z^{(d)} - \mathcal{M}\right)_{(i)} \\ &\quad \left(\left(\mathcal{G} \times_1 Z^{(1)} \cdots \times_{i-1} Z^{(i-1)} \times_{i+1} Z^{(i+1)} \right.\right. \\ &\quad \left.\left. \cdots \times_d Z^{(d)}\right)_{(i)}\right)^T \\ &= \left(Z^{(i)} G_{(i)} \left(Z^{(d)} \otimes \cdots \otimes Z^{(i+1)} \otimes Z^{(i-1)} \right.\right. \\ &\quad \left.\left. \otimes \cdots \otimes Z^{(1)}\right)^T - M_{(i)}\right) \\ &\quad \left(G_{(i)} \left(Z^{(d)} \otimes \cdots \otimes Z^{(i+1)} \otimes Z^{(i-1)} \right.\right. \\ &\quad \left.\left. \otimes \cdots \otimes Z^{(1)}\right)^T\right)^T. \end{aligned}$$

Let

$$B_i^{k-1} = G_{(i)}^{k-1}\left(Z_{k-1}^{(d)} \otimes \cdots \otimes Z_{k-1}^{(i+1)} \otimes Z_{k-1}^{(i-1)} \otimes \cdots \otimes Z_{k-1}^{(1)}\right)^T.$$

Therefore, we take

$$L_i^{k-1} = \|B_i^{k-1}(B_i^{k-1})^T\|_2,$$

$$\omega_i^{k-1} = \min\left(\hat{\omega}_{k-1}, \delta_\omega \sqrt{\frac{L_i^{k-2}}{L_i^{k-1}}}\right),$$

where $\delta_\omega < 1$ is a pre-selected parameter and $\hat{\omega}_{k-1} = \frac{t_{k-1}-1}{t_k}$ with $t_0 = 1$ and $t_k = \frac{1}{2}\left(1 + \sqrt{1 + 4t_{k-1}^2}\right)$. In addition, let $\hat{Z}_{k-1}^{(i)} = Z_{k-1}^{(i)} + \omega_i^{k-1}\left(Z_{k-1}^{(i)} - Z_{k-2}^{(i)}\right)$, and

$$\hat{H}_i^{k-1} = \left(\hat{Z}_{k-1}^{(i)} B_i^{k-1} - M_{(i)}\right)\left(B_i^{k-1}\right)^T$$

be the gradient.

Then we use the prox-linear iterative rules to derive the update:

$$\begin{aligned} Z_k^{(i)} = \arg\min_{Z^{(i)} \geq 0} & \left\langle \hat{H}_i^{k-1}, Z^{(i)} - \hat{Z}_{k-1}^{(i)} \right\rangle \\ & + \frac{L_i^{k-1}}{2}\left\|Z^{(i)} - \hat{Z}_{k-1}^{(i)}\right\|_F^2, \end{aligned}$$

which can be written in the closed form

$$Z_k^{(i)} = \max\left(0, \hat{Z}_{k-1}^{(i)} - \hat{H}_i^{k-1}/L_i^{k-1}\right). \tag{2}$$

At the end of iteration k, we check whether $h(Z_k) \geq h(Z_{k-1})$. If so, we re-update $Z_k^{(i)}$ by Eq. (2) with $\hat{Z}_{k-1}^{(i)} = Z_{k-1}^{(i)}$, for $i = 1, \ldots, d$.

By the above procedures, we can find the optimal $Z^{(i)}, i = 1, 2, \cdots, d$, which is nonnegative. Since the optimization of $A^{(i)}$ is irrelevant to the penalty term of function p and $T^{(i)}$ is a diagonal matrix whose element is 0 or 1, the columns of the optimal $A^{(i)}$ is selected from that of $Z^{(i)}$, e.g., if the element $(2,2)$ of matrix $T^{(1)}$ is 1, the 2nd column of $A^{(1)}$ is equal to that of $Z^{(1)}$, otherwise, it is a zero column. More specifically, we need the following lemma to guide us to find the optimal $A^{(i)}$ and $T^{(i)}$, which can be easily proved.

Lemma 1 Suppose matrix $U = (u_1, u_2, \cdots, u_m) \in \mathbb{R}^{n \times m}$, where $u_i \in \mathbb{R}^n$, and the number of its columns is equal to the dimension of tensor $\mathcal{F}$ in mode k. Let matrix Q_i be the same size as U, for $i = 1, 2, \cdots, m$. For each $i \in \{1, 2, \cdots, m\}$, the i-th column of Q_i is u_i, and the rest of Q_i's columns are all zeros. Then, the k-mode product of tensor $\mathcal{F}$ and U has the following property:

$$\mathcal{F} \times_k U = \mathcal{F} \times_k Q_1 + \mathcal{F} \times_k Q_2 + \cdots + \mathcal{F} \times_k Q_m.$$

Now we are ready to find the optimal $A^{(i)}$ and $T^{(i)}$ after obtaining $Z^{(i)}$ for a given i. Without loss of generality, we present the case that $i = 1$. The target optimization problem is

$$\begin{aligned} (AT^1) \quad \min \quad & \tfrac{1}{2}\left\|(A^{(1)}T^{(1)})W_1 - M_{(1)}\right\|_F^2 \\ & + \lambda\left(\textstyle\sum_{j=1}^{m_1} t_j^{(1)} + c_1\right)^2 \\ s.t. \quad & A^{(1)} \in \mathbb{R}^{n_1 \times m_1},\ A^{(1)} \geq 0, \\ & t^{(1)} \in \{0,1\}^{m_1}, \\ & \textstyle\sum_{j=1}^{m_1} t_j^{(1)} \geq 1. \end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

where $W_1 = G_{(1)}\left(A^{(d)}T^{(d)} \otimes \cdots \otimes A^{(2)}T^{(2)}\right)^T$ and $c_1 = \sum_{i=2}^{d}\sum_{j=1}^{m_i} t_j^{(i)} - c$. First, we select only one single column of $Z^{(i)}$, that is, the 1st, the 2nd, ..., or the m_1-th column of $Z^{(1)}$, denoted as $z_j, j = 1, 2, \cdots, m_1$. Therefore, we can construct new matrices

$$\bar{A}_j^{(1)} = (0, \cdots, 0, z_j, 0, \cdots, 0) \in \mathbb{R}^{n_1 \times m_1},$$

for $j = 1, 2, \cdots, m_1$, according to Lemma 1 and calculate the corresponding value of function h respectively, denoted as $h_j^{(1)}, j = 1, 2, \cdots, m_1$. Then, we sort these values in ascending order, the smaller the value is, the higher priority we choose. The sum of all $\bar{A}_j^{(1)}$ will results in $A^{(1)}$. Furthermore, if the j-th column of $\bar{A}^{(1)}$ (or $A^{(1)}$) is non-zero, $t_j^{(1)}$ is equal to 1. Next, we need to determine the total number of columns that is chosen in $Z^{(1)}$, i.e., the value of $\sum_{j=1}^{m_1} t_j^{(1)}$. This is a combinatorial problem, but it can be solved in polynomial time. For the optimization problem (AT^1), all the possible values for $\sum_{j=1}^{m_1} t_j^{(1)}$ are $\{1, 2, \ldots, m_1\}$, thus, we only need to try m_1 different values for $\sum_{j=1}^{m_1} t_j^{(1)}$ and pick the best solution. For example, suppose the optimal value $\sum_{j=1}^{m_1} t_j^{(1)} = m^*$, then we pick the first m^* smaller values in $h_j^{(1)}$ $(j = 1, 2, \cdots, m_1)$, and find the corresponding optimal $A^{(1)}$ and also $T^{(1)}$.

2) Updating tensors

For problem (PM), we fix block variables $(A^{(j)}, T^{(j)})$, $j = 1, 2, \cdots, d$ and the tensor $\mathcal{M}$, and then compute the gradient

$$\begin{aligned}\nabla_{\mathcal{G}} p \;=\; & \big(\mathcal{G} \times_1 \left(A^{(1)}T^{(1)}\right) \times_2 \left(A^{(2)}T^{(2)}\right) \times_3 \ldots \\ & \times_d (A^{(d)}T^{(d)}) - \mathcal{M}\big) \times_1 \left(A^{(1)}T^{(1)}\right)^T \\ & \times_2 (A^{(2)}T^{(2)})^T \times_3 \cdots \times_d \left(A^{(d)}T^{(d)}\right)^T.\end{aligned}$$

Let

$$\begin{aligned}B_{k-1} \;=\; & \left(A_{k-1}^{(d)}T_{k-1}^{(d)}\right) \otimes \cdots \otimes \left(A_{k-1}^{(2)}T_{k-1}^{(2)}\right) \\ & \otimes \left(A_{k-1}^{(1)}T_{k-1}^{(1)}\right), \\ S_{k-1} \;=\; & \|(B_{k-1})^T B_{k-1}\|_2, \\ \hat{\mathcal{G}}_{k-1} \;=\; & \mathcal{G}_{k-1} + \omega_{k-1}\left(\mathcal{G}_{k-1} - \mathcal{G}_{k-2}\right),\end{aligned}$$

where parameter ω_{k-1} is updated similarly as that in Section IV-B1. Due to the advantages of the prox-linear iterative rule, we use it again to update the core tensor:

$$\mathcal{G}_k = \max\left(0, \hat{\mathcal{G}}_{k-1} - (\nabla_{\mathcal{G}} p)_{k-1}/S_{k-1}\right). \tag{3}$$

At the end of iteration k, we check whether $p(\mathcal{G}_k) \geq p(\mathcal{G}_{k-1})$. If so, we re-update $\mathcal{G}_k$ by Eq. (3) with $\hat{\mathcal{G}}_{k-1} = \mathcal{G}_{k-1}$. Remark that we compute $(B_{k-1})^T B_{k-1}$ by the following way in order to compute S_{k-1} more efficiently:

$$\begin{aligned}(B_{k-1})^T B_{k-1} \;=\; & \left(A_{k-1}^{(d)}T_{k-1}^{(d)}\right)^T \left(A_{k-1}^{(d)}T_{k-1}^{(d)}\right) \otimes \\ & \cdots \otimes \left(A_{k-1}^{(1)}T_{k-1}^{(1)}\right)^T \left(A_{k-1}^{(1)}T_{k-1}^{(1)}\right).\end{aligned}$$

Finally, we update $\mathcal{M}$ when having $\mathcal{G}_k$ and $A_k^{(i)}, T_k^{(i)}, i = 1, 2, \ldots, d$ as follows:

$$\begin{aligned}\mathcal{M}_k = P_\Omega(\mathcal{F}) + P_{\Omega^c}(\mathcal{G}_k \times_1 (A_k^{(1)}T_k^{(1)}) \\ \times_2 (A_k^{(2)}T_k^{(2)}) \times_3 \cdots \times_d (A_k^{(d)}T_k^{(d)})).\end{aligned} \tag{4}$$

C. OUR ALGORITHM AND CONVERGENCE RESULT

Algorithm 2: BCD method with prox-linear updates for solving (PM)

Input: Nonnegative tensor $\mathcal{F}$ with missing values, index set of the known data Ω, integer $c \geq d$, and parameter α;

Output: Nonngetive tensor $\mathcal{M}$.

1: Initialization: positive parameter λ_0, initial values $(\mathcal{G}_{-1}, A_{-1}^{(1)}, A_{-1}^{(2)}, \ldots, A_{-1}^{(d)}, T_{-1}^{(1)}, T_{-1}^{(2)}, \ldots, T_{-1}^{(d)}) = (\mathcal{G}_0, A_0^{(1)}, A_0^{(2)}, \ldots, A_0^{(d)}, T_0^{(1)}, T_0^{(2)}, \ldots, T_0^{(d)})$.
2: **for** $k = 1, 2, \ldots$ **do**
3: Update $\mathcal{G}_k$ according to (3).
4: **if** $p(\mathcal{G}_k) \geq p(\mathcal{G}_{k-1})$ **then**
5: Reupdate $\mathcal{G}_k$ according to (3) with $\hat{\mathcal{G}}_{k-1} = \mathcal{G}_{k-1}$.
6: **end if**
7: **for** $i = 1, 2, \ldots, d$ **do**
8: Update $Z_k^{(i)}$ according to (2).
9: **if** $h(Z_k) \geq h(Z_{k-1})$ **then**
10: Reupdate $Z_k^{(i)}$ according to (2) with $\hat{Z}_{k-1}^{(i)} = Z_{k-1}^{(i)}$.
11: **end if**
12: Extract $A_k^{(i)}$ and $T_k^{(i)}$ from $Z_k^{(i)}$.
13: $Z_k^{(i)} = A_k^{(i)}T_k^{(i)}$.
14: **end for**
15: Update $\mathcal{M}_k$ according to (4).
16: **if** stopping criterion is satisfied **then**
17: Return $\mathcal{M}_k$.
18: **end if**
19: $\lambda := \alpha\lambda$.
20: **end for**

We solve the nonnegative tensor completion problem (NT) via the penalty method. According to the structure of the penalty model (PM), we apply the BCD method with prox-linear updating rule. Algorithm 2 summarizes the procedures for solving problem (PM). We alternatively update the core tensor (lines 3-6), the factor matrices (lines 7-14), and tensor $\mathcal{M}$ (line 15) until a stopping criterion is satisfied.

We have the following convergence result of Algorithm 2, which can be verified directly from Theorem 3.1 of [19].

Theorem 1 Let $\{\mathcal{G}_k, A_k^{(i)}, T_k^{(i)}, \mathcal{M}_k\}, i = 1, 2, \ldots, d$ be the sequence generated by Algorithm 2. Suppose that $\{\mathcal{G}_k\}, \{A_k^{(i)}\}, \{T_k^{(i)}\}$ are bounded and there exists a positive constant ξ such that $\xi \leq L_i^k$ for all k and i, and a positive constant η such that $\eta \leq S_k$ for all k. Then $\{\mathcal{G}_k, A_k^{(i)}, T_k^{(i)}, \mathcal{M}_k\}, i = 1, 2, \ldots, d$ converges to a critical point.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

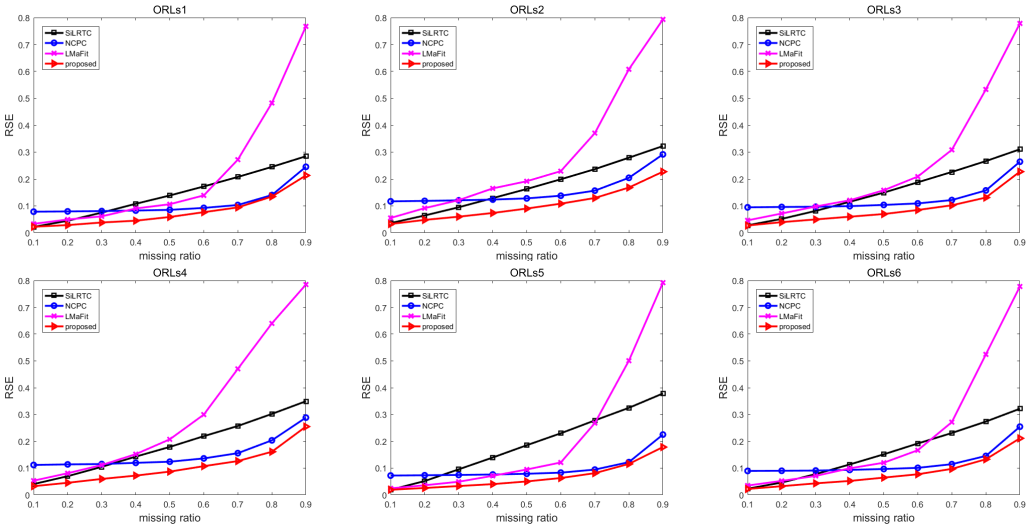

Fig. 1: Comparisons in terms of RSE for the ORL database of faces when missing ratio varies.

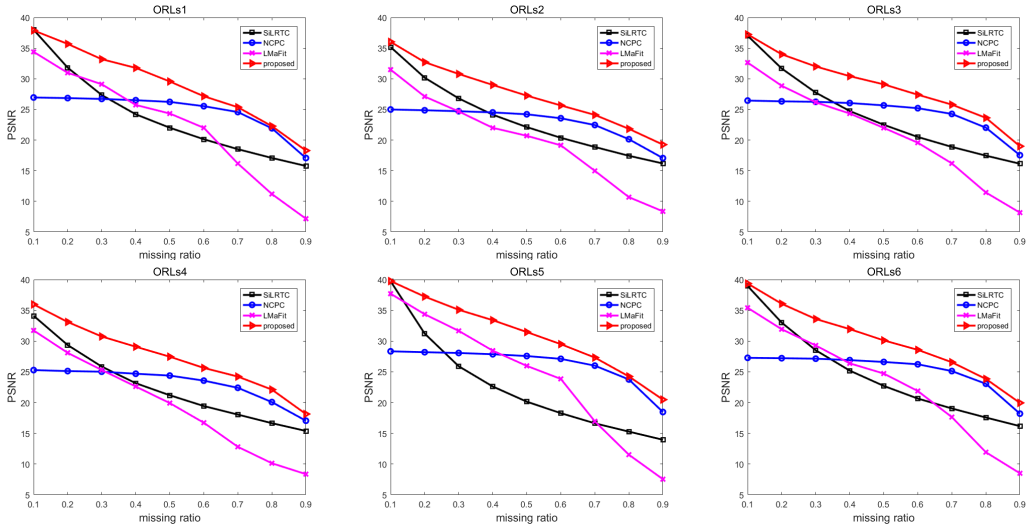

Fig. 2: Comparisons in terms of PSNR for the ORL database of faces when missing ratio varies.

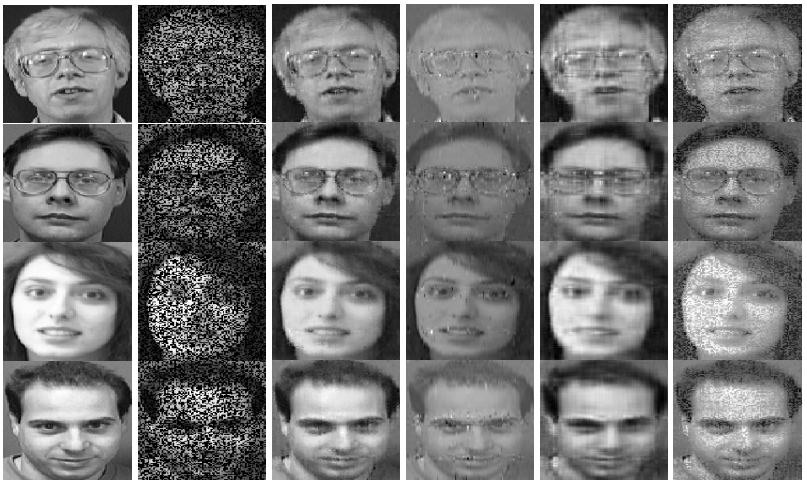

Fig. 3: Completion results of the ORL database. Columns from left to right is the original images, 50% corrupted images and recovered images by our algorithm, LMaFit, NCPC and SiLRTC, respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

V. EXPERIMENTAL RESULTS

In this section, we use three different types of image data to test our algorithm and to demonstrate its effectiveness and efficiency, since the corresponding tensor is always nonnegative that fits our work. All the numerical computations are conducted in an Intel Core CPU 3.30GHz 8GB RAM computer. The supporting software is MATLAB R2016a as a platform. We use MATLAB Tensor Toolbox 2.6 [28] whenever tensor operations are called. The proposed algorithm is compared with the three state-of-the-art algorithms, i.e., low rank matrix fitting (LMaFit) algorithm [29], nonnegative CP decomposition from partial observations (NCPC) algorithm [19] and the simple low rank tensor completion (SiLRTC) algorithm [13].

In implementing all the four algorithms, the termination precision is set to be 10^{-5}, and the maximum iteration is set to be 500. In addition, we set Zfull $= 1$, est rank $= 2$ and max rank $= 50$ and r (rank estimate)$= 20$ for LMaFit, esr $= 50$ for NCPC, and the weights to control the rank in each mode of the tensor are set to be equal (i.e., $\alpha_i = \frac{1}{3}, i = 1, 2, 3$) for SiLRTC. The other parameters of these three algorithms are set as default values. For our algorithm, the larger the constant value c, the better the completion effect in general. Since we consider nonnegative tensor completion via a low rank NTD, we could not set the value c too large. Therefore, we set $c = 100$ in our experiments. Let $\alpha = 2$, and we set the initial λ to be $\lambda_0 = 4\|\mathcal{M}\|_F$. The initial values of $\mathcal{G}_{-1}$ and $A^{(1)}_{-1}, A^{(2)}_{-1}, A^{(3)}_{-1}$ are the maximum value between random generated values drawn from the standard normal distribution and 0. The initial values of $T^{(1)}_{-1}, T^{(2)}_{-1}, T^{(3)}_{-1}$ are all zero, except that their (1,1)-entry are 1.

For the image data, we randomly remove some pixels of each original image whose entries follow from uniform distribution and we use the relative square error (RSE) and the peak signal-to-noise ratio (PSNR) to measure the quality of the algorithms, which are defined as follows:

$$\text{RSE} = \frac{\|\hat{\mathcal{M}} - \mathcal{M}\|_F}{\|\mathcal{M}\|_F}, \quad \text{PSNR} = 10\,log(\frac{255^2}{\text{MSE}})$$

where $\mathcal{M} \in \mathbb{R}^{I\times J\times K}$ is the observed nonnegative tensor and $\hat{\mathcal{M}}$ is the completion result of the algorithms, and

$$\text{MSE} = \frac{1}{IJK}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(\mathcal{M}_{ijk} - \hat{\mathcal{M}}_{ijk})^2.$$

Each algorithm is repeated three times, taking the average of RSE and PSNR. In general, the smaller the RSE and the larger the PSNR, the better the effect of the image completion.

A. THE ORL DATABASE OF FACES

We use the ORL database of faces [30] in AT&T Laboratories Cambridge. In this database, there are 40 different subjects, each of which contains 10 different grayscale images, each 92×112 pixels in size. We use one of these subjects to form a 92×112×10 sized tensor.

We show some comparison results in terms of RSE and PSNR for this database by using the four algorithms when missing ratio varies from 0.1 to 0.9 in Fig. 1 and Fig. 2, respectively. It can be seen from these two figures that when the missing ratio is greater than or equal to 0.2, the proposed algorithm has the smallest RSE and the largest PSNR among the four algorithms, which means our algorithm has the best completion performance. We here focus on the recovered quality of each algorithm, hence, we do not report the computational time of them although our method costs more in general. It is worth mentioning that most of the computational effort of our proposed algorithm is to find a suitable combination of Tucker rank (r_1, r_2, r_3), which means that our rank selecting strategy is better than that of other algorithms. This computational effort in selecting a better rank is worthwhile. Fig. 3 further certificates this phenomenon. We can vividly observe that our algorithm is able to recover the image with 50% missing information and has the best completion effect among the four algorithms.

B. MCGILL CALIBRATED COLOUR IMAGE DATABASE

The second image dataset is from McGill calibrated colour image database [31]. The size of each image is 768×576. In order to save computational time, we resize the image to 192×144 pixels, i.e., a 192×144×3 sized tensor is formed, and a total of six color images (namely Rabbit, Leaves, FallenLeaves, NetTiger, Grass and Flowers) are used.

Fig. 4 and Fig. 5 show the comparison results in terms of RSE and PSNR for the six color images when missing ratio varies from 0.1 to 0.9. We can see that the performance of our algorithm, NCPC and SiLRTC are all better than that of LMaFit when missing value is larger than 0.3, and the former three ones has similar performance when missing ratio is in [0.3, 0.7]. However, when the missing ratio is greater than or equal to 0.8, the proposed algorithm has the smallest RSE and the largest PSNR among the four algorithms, meaning the best completion performance that our algorithm has. Fig. 6 shows the completion results of Rabbit, Leaves and Flowers when missing ratio is 0.5. The recovery effect of all the four algorithms is satisfactory, and SiLRTC performs worse than the other three ones.

C. COLUMBIA UNIVERSITY IMAGE LIBRARY

The last test is on Columbia university image library (COIL-20) [32]. Each object contains 72 grayscale images with a size of 128×128 pixels. For each object, we select the first 10 images and form a tensor of 128×128×10 size. We choose a total of six objects.

Fig. 7 and Fig. 8 show the comparison results in terms of RSE and PSNR for the six objects when missing ratio varies from 0.1 to 0.9. We observe that the proposed algorithm has the smallest RSE and the largest PSNR among the four algorithms when the missing ratio is less than or equal to 0.5. Fig. 9 shows the completion results of four grayscale images when missing ratio is 0.3. It can be seen that our algorithm clearly has the best completion effect.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

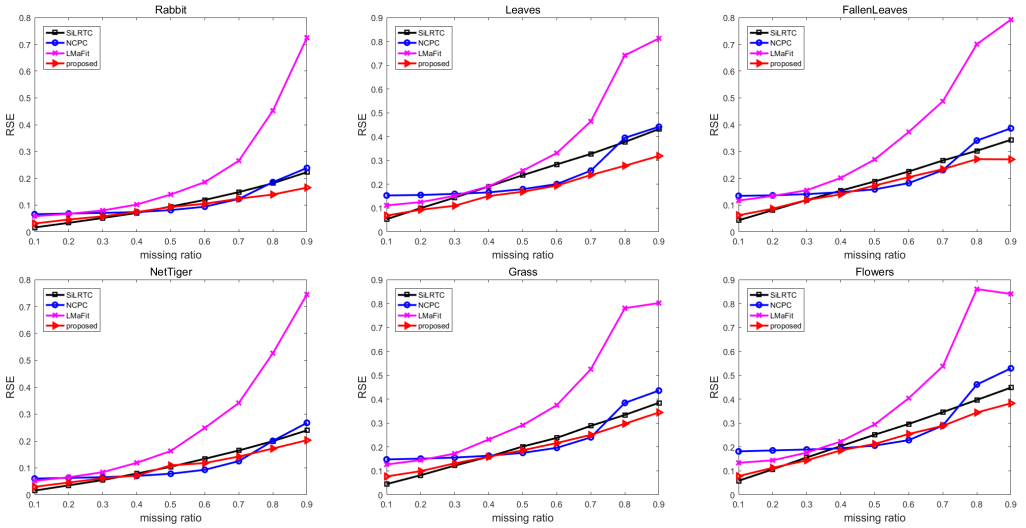

Fig. 4: Comparisons in terms of RSE for the colour image database when missing ratio varies.

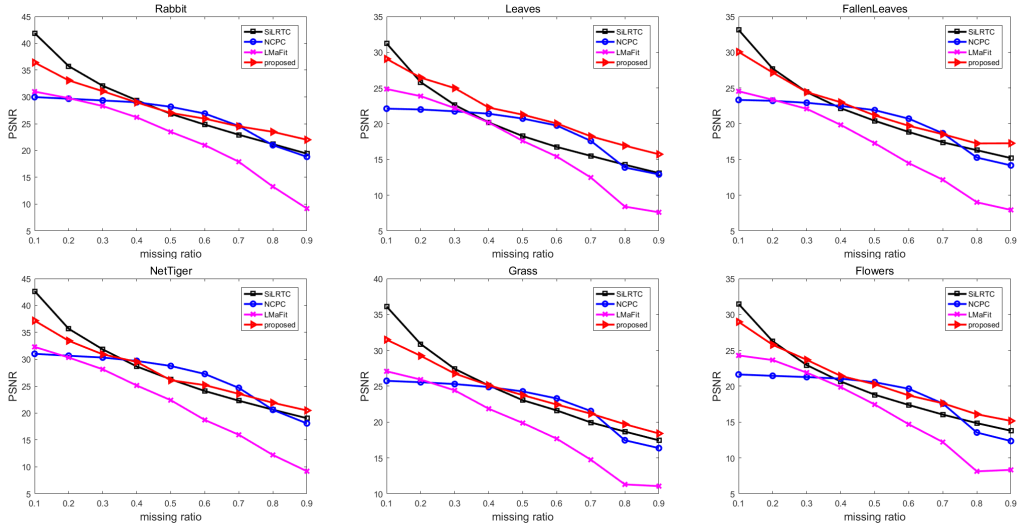

Fig. 5: Comparisons in terms of PSNR for the colour image database when missing ratio varies.

Fig. 6: Completion results of the colour image database. Columns from left to right is the original images, 50% corrupted images and recovered images by our algorithm, LMaFit, NCPC and SiLRTC, respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

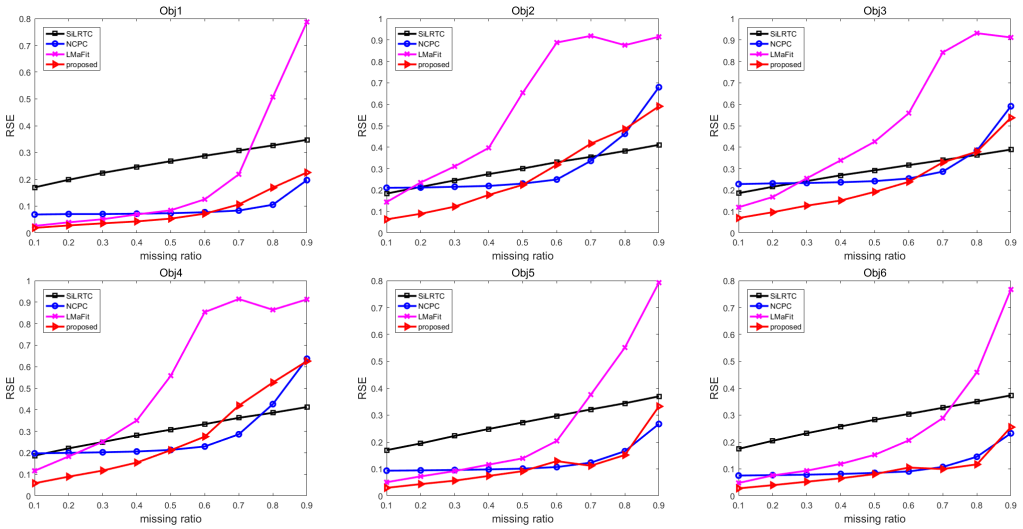

Fig. 7: Comparisons in terms of RSE for COIL-20 database when missing ratio varies.

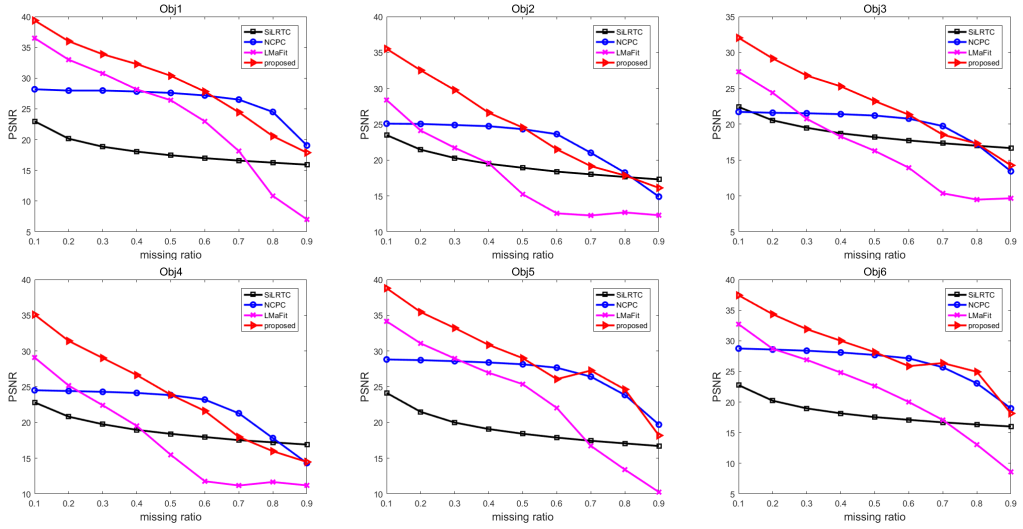

Fig. 8: Comparisons in terms of PSNR for COIL-20 database when missing ratio varies.

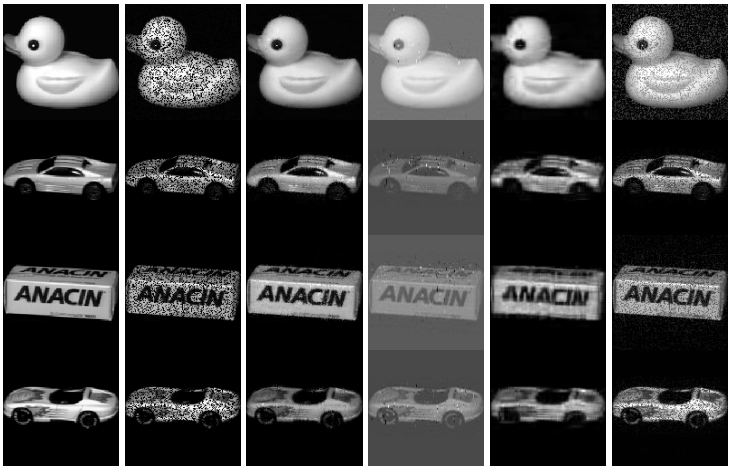

Fig. 9: Completion results of COIL-20 database. Columns from left to right is the original images, 30% corrupted images and recovered images by our algorithm, LMaFit, NCPC and SiLRTC, respectively.

VI. CONCLUSION

This paper considers the problem of nonnegative tensor completion via low rank Tucker decomposition in which the core tensor size can be adjusted by itself, which deals with an important practical issue for real applications. Traditionally, the core tensor size is given in advance. We construct a new model for solving this particular problem. The approach that we present is based on the penalty method and block coordinate descent method with prox-linear updates for regularized multiconvex optimization. We present the convergence of our proposed algorithm. The numerical results on the three image datasets show that our algorithm is competitive compared to other existing algorithms even though the data is very sparse, e.g., the missing ratio is up to 90%.

REFERENCES

[1] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in the 45th Annual ACM Symposium on the Theory of Computing, 2013, pp. 665–674.
[2] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from noisy entries," Journal of Machine Learning Research, vol. 11, no. 3, pp. 2057–2078, 2010.
[3] R. Sun and Z. Q. Luo, "Guaranteed matrix completion via non-convex factorization," IEEE Transactions on Information Theory, vol. 62, no. 11, pp. 6535–6579, 2016.
[4] F. Cao, M. Cai, and Y. Tan, "Image interpolation via low-rank matrix completion and recovery," IEEE Transactions on Circuits and Systems for Video Technology, vol. 25, no. 8, pp. 1261–1270, 2015.
[5] S. Lu, X. Ren, and F. Liu, "Depth enhancement via low-rank matrix completion," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3390–3397.
[6] G. Obozinski, B. Taskar, and M. I. Jordan, "Joint covariate selection and joint subspace selection for multiple classification problems," Statistics and Computing, vol. 20, no. 2, pp. 231–252, 2010.
[7] G. Ramos, J. Saude, C. Caleiro, and S. Kar, "Recommendation via matrix completion using kolmogorov complexity," 2017, arXiv:1707.06055.
[8] D. Shin, S. Cetintas, K. C. Lee, and I. S. Dhillon, "Tumblr blog recommendation with boosted inductive matrix completion," in Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 203–212.
[9] F. Chan, A. Ma, P. Yuen, C. Yip, Y. Tse, W. Wong, and L. Wong, "Temporal matrix completion with locally linear latent factors for medical applications," 2016, arXiv:1611.00800.
[10] N. Boumal and P. A. Absil, "Rtrmc: a riemannian trust-region method for low-rank matrix completion," in International Conference on Neural Information Processing Systems, 2011, pp. 406–414.
[11] Y. Wang, R. Chen, J. Ghosh, J. C. Denny, A. Kho, Y. Chen, B. A. Malin, and J. Sun, "Rubik: knowledge guided tensor factorization and completion for health data analytics," in the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 1265–1274.
[12] X. Li, Y. Ye, and X. Xu, "Low-rank tensor completion with total variation for visual data inpainting," in Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 2210–2216.
[13] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 1, pp. 208–220, 2013.
[14] B. Ran, H. Tan, J. Feng, Y. Liu, and W. Wang, "Traffic speed data imputation method based on tensor completion," Computational Intelligence and Neuroscience, vol. 2015, no. 22, 2015.
[15] H. Tan, Y. Wu, B. Shen, P. J. Jin, and B. Ran, "Short-term traffic prediction based on dynamic tensor completion," IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 8, pp. 2123–2133, 2016.
[16] K. Xie, L. Wang, X. Wang, G. Xie, J. Wen, and G. Zhang, "Accurate recovery of internet traffic data: A tensor completion approach," in INFOCOM 2016 - the IEEE International Conference on Computer Communications, 2016.
[17] Y. Xu, W. Yin, Z. Wen, and Y. Zhang, "An alternating direction algorithm for matrix completion with nonnegative factors," Frontiers of Mathematics in China, vol. 7, no. 2, pp. 365–384, 2012.
[18] M. Filipović and A. Jukić, "Tucker factorization with missing data with application to low-n-rank tensor completion," Multidimensional Systems and Signal Processing, vol. 26, no. 3, pp. 677–692, 2015.
[19] Y. Xu and W. Yin, "A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion," SIAM Journal on Imaging Sciences, vol. 6, no. 3, pp. 1758–1789, 2013.
[20] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth parafac decomposition for tensor completion," IEEE Transactions on Signal Processing, vol. 64, no. 20, pp. 5423–5436, 2016.
[21] J. Royer, N. Thirion-Moreau, and P. Comon, "Nonnegative 3-way tensor factorization taking into account possible missing data," in the 20th European Signal Processing Conference (EUSIPCO), 2012, pp. 71–75.
[22] L. Karlsson, D. Kressner, and A. Uschmajew, "Parallel algorithms for tensor completion in the cp format," Parallel Computing, vol. 57, pp. 222–234, 2016.
[23] O. Kaya and B. Uçar, "Parallel candecomp/parafac decomposition of sparse tensors using dimension trees," SIAM Journal on Scientific Computing, vol. 40, no. 1, pp. C99–C130, 2018.
[24] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," SIAM review, vol. 51, no. 3, pp. 455–500, 2009.
[25] B. Chen, Z. Li, and S. Zhang, "On optimal low rank tucker approximation for tensors: the case for an adjustable core size," Journal of Global Optimization, vol. 62, no. 4, pp. 811–832, 2015.
[26] L. Yang, J. Fang, H. Li, and B. Zeng, "An iterative reweighted method for tucker decomposition of incomplete tensors," IEEE Transactions on Signal Processing, vol. 64, no. 18, pp. 4817–4829, 2016.
[27] S. Oh, N. Park, S. Lee, and U. Kang, "Scalable tucker factorization for sparse tensors-algorithms and discoveries," in 2018 IEEE 34th International Conference on Data Engineering (ICDE), 2018, pp. 1120–1131.
[28] B. W. Bader, T. G. Kolda et al., "Matlab tensor toolbox version 2.6," Available online, 2015. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/
[29] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm," Mathematical Programming Computation, vol. 4, no. 4, pp. 333–361, 2012.
[30] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in IEEE Workshop on Applications of Computer Vision, 1994.
[31] A. Olmos and F. A. A. Kingdom, "A biologically inspired algorithm for the recovery of shading and reflectance images," Perception, vol. 33, no. 12, pp. 1463–1473, 2004, pMID: 15729913. [Online]. Available: https://doi.org/10.1068/p5321
[32] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-20)," Technical Report CUCS-005-96, Tech. Rep., 1996.

BILIAN CHEN received her Ph.D. degree from The Chinese University of Hong Kong in 2012. Now she is an associate professor in Xiamen University. Her research interests include operational research, optimization theory and recommendation system. Her publications appear in SIAM Journal on Optimization, Journal of Global Optimization, Information Sciences, and so on.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2929189, IEEE Access

TING SUN received her Bachelor's degree in department of automation at Xiamen University in 2016. She is currently pursuing her Master's degree in department of automation at Xiamen University. Her research interest is tensor optimization.

ZHEHAO ZHOU received his B.S. degree in automation from Xiamen University in 2018. He is currently pursuing the Master degree in Artificial Intelligence and System Engineering with Xiamen University. His research interests include machine learning and tensor optimization.

YIFENG ZENG received the Ph.D. degree from National University of Singapore, Singapore, in 2006.He is a Reader with the School of Computing, Teesside University, Middlesbrough, U.K. He is also an Affiliate Professor with Xiamen University, Xiamen, China. His research interests include intelligent agents, decision making, social networks, optimization theory and computer games. Most of his publications appear in the most prestigious international academic journals and conferences, including Journal of Artificial Intelligence Research, Journal of Autonomous Agents and Multi-Agent Systems, International Conference on Autonomous Agents and Multi-Agent Systems, International Joint Conference on Artificial Intelligence, and Association for the Advancement of Artificial Intelligence.

LANGCAI CAO received his Ph.D. degree in automation from Xiamen University in 2011. Now he is an associate professor in Xiamen University. His research interests include research and development of information system, process intelligence and machine learning.