Tensor Completion via the CP Decomposition

Fatoumata Sanogo
sanogof1@uab.edu

Department of Mathematics
University of Alabama at Birmingham
Birmingham, Alabama 35294-1170

Carmeliza Navasca
cnavasca@uab.edu

Department of Mathematics
University of Alabama at Birmingham
Birmingham, Alabama 35294-1170

January 27, 2019

Abstract

We propose a new algorithm for tensor completion. The tensor completion problem is about finding the unknown tensor from a given a tensor with partially observed data. While most tensor completion methods use the Tucker model, our new approach uses the canonical polyadic decomposition model to reconstruct the unknown tensor. The unknown tensor is reconstructed by finding the optimal factors through linear least squares and the singular vectors through a proximal algorithm of soft thresholding.

1 Introduction

The matrix (tensor) completion [5] problem is about filling in missing entries from the partially observed entries of the matrix (tensor). The matrix/tensor completion has received much attention in big data analysis, e.g. in collaborative filtering algorithms. However, the tensor completion problem dates back as early as in 2000. The matrix (tensor) completion problem has been solved using different approaches. Several papers have applied the Canonical Polyadic (CP) and Tucker decomposition to find the missing entries of a tensor. Bro [2] had one of the earliest work on demonstrating two ways to handle missing data using canonical polyadic decomposition. The first way is to alternatively estimate the model parameters while imputing the missing data [3]. It was called the Missing-Skipping (MS) approach; it skips the missing value and builds up the model based only on the observed part [9]. The second way is the weighted least square PARAFAC decomposition [10], [11]. The Tucker decomposition is the popular model for tensor completion [7]. Tucker decomposition and Higher Order Orthogonal Iteration were combined to deal with missing data [12]. Specifically, the trace norm method has been used to predict missing values; see, e.g., [6] [15]. In the paper [5], it shown that the trace norm minimization (a.k.a. matrix rank minimization) problem recovers the unknown matrix. Then in [6], the authors have used the matricized Tucker models and applied the trace norm minimization iteratively.

2 Current Results

In this paper, we propose a method for solving tensor completion. While most algorithms for tensor completion applies the matrix rank minimization approach, we take an alternative approach by formulating a sparse optimization problem for recovering the CP decomposition of a given tensor with partial observed entries. Let us define the CP decomposition of a tensor $\mathcal{S}$ to be $\mathcal{S} = \sum_r^R \sigma_r a_r \circ b_r \circ c_r \in \mathbb{R}^{I\times J\times K}$ where $a_r \in \mathbb{R}^I, b_r \in \mathbb{R}^J, c_r \in \mathbb{R}^K$ are rth column vectors of what we call the factor matrices $A \in \mathbb{R}^{I\times R}, B \in \mathbb{R}^{J\times R}$ and $C \in \mathbb{R}^{K\times R}$, respectively, and σ_r is the rth entry in the column vector $\sigma \in \mathbb{R}^R$. Note that $a_r \circ b_r \circ c_r$ is rank one tensor, derived by the outer products of the three vectors, a_r, b_r and c_r. Given a tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$ with partially observed entries on the index set Ω, the tensor (matrix) completion problem formulation is

$$\begin{aligned} &\min_{A,B,C,\sigma} \|\mathcal{T} - \mathcal{S}\|_F + \lambda\|\sigma\|_{\ell_1} \\ &\text{subject to } \mathcal{S}(\Omega) = \mathcal{T}(\Omega) \end{aligned} \tag{1}$$

where γ is a constant regularization parameter. Here the tensor Frobenious norm is defined as $\|\mathcal{T}\|_F = (\sum_{ijk} \mathcal{T}_{ijk}^2)^{\frac{1}{2}}$ and the vector ℓ_1 norm is $\|\sigma\|_{\ell_1} = \sum_r^R |\sigma_r|$.

The tensor completion problem (1) is a minimization of a sum of a smooth term and a nonsmooth term, we then consider the following optimization problem with smooth and nonsmooth terms:

$$\mathbf{x}^{n+1} = \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}^n) + \langle \mathbf{x} - \mathbf{x}^n, \nabla_{\mathbf{x}} f(\mathbf{x}^n)\rangle + \tfrac{t}{2}\|\mathbf{x} - \mathbf{x}^n\|^2 + g(\mathbf{x}) \right\} \tag{2}$$

where f and g are the smooth and nonsmooth functions, respectively. Here f is approximated at a given point $\mathbf{x}^k$. Furthermore, problem (1) finds the factor matrices A, B, C and vector σ which construct a low rank tensor $\mathcal{S}$, filling in the missing entries of the original

tensor $\mathcal{T}$. Thus, we solve the linear least squares subproblems using gradient descent and then implement a soft thresholding iteratively until the residual vector is within our stopping criteria. This optimization formulation has been applied in analyzing surveillance video in the foreground/background extraction [13]

2.1 Numerical Approach

Our strategy for solving problem (1) is to solve five optimization subproblems iteratively and then impose the constraint. Let $f(A,B,C,\sigma) = \|\mathcal{T} - \mathcal{S}\|_F = \|\mathcal{T} - \sum_r^R \sigma_r a_r \circ b_r \circ c_r\|_F$ and $g(\sigma) = \gamma\|\sigma\|_{\ell_1}$ where R is an upper bound of the rank of $\mathcal{S}$. We rescaled the columns of the matrices $\mathbf{A},\mathbf{B},\mathbf{C}$ [1, 4] to prevent the norm of the approximated matrices from blowing up to infinity when implementing the canonical polyadic decomposition. We then introduce an indicator function, to switch the optimization problem into the following form:

$$\min_{A,B,C,\boldsymbol{\sigma}} \left\| \mathcal{T} - \sum_r^R \boldsymbol{\sigma}_r a_r \circ b_r \circ c_r \right\|_F + \lambda\|\boldsymbol{\sigma}\|_{\ell_1} + \delta_{S_1}(\mathbf{A}) + \delta_{S_2}(\mathbf{B}) + \delta_{S_3}(\mathbf{C})$$
$$\text{subject to } \mathcal{S}(\Omega) = \mathcal{T}(\Omega)$$

where $S_1 = \{\mathbf{A} | \|\mathbf{a}_r\| = 1, r = 1,\cdots,R\}, S_2 = \{\mathbf{B} | \|\mathbf{b}_r\| = 1, r = 1,\cdots,R\}$ and $S_3 = \{\mathbf{C} | \|\mathbf{c}_r\| = 1, r = 1,\cdots,R\}$.

Observe that $f(A,B,C,\sigma)$ is a nonlinear least squares objective function. However, if $\mathcal{T} \approx \sum_r^R \sigma_r a_r \circ b_r \circ c_r$ is matricized, then we have the following:

$$T_{(1)} \approx AD(C \odot B)^T,\ T_{(2)} \approx BD(C \odot A)^T, \quad \text{and } T_{(3)} \approx CD(B \odot A)^T \tag{3}$$

where $T_{(i)}$ are the concatenation of the *sliced* matrices in $\mathcal{T}$ and D is an $r \times r$ diagonal matrix with entries σ_r. The Khatri-Rao product $\odot$ is the column-wise Kronecker product; i.e. $C \odot B = [c_1 \otimes b_1 \ \ldots c_R \otimes b_R]$. Also, another matricization is

$$t \approx \sigma S \tag{4}$$

where t is the vectorization of $\mathcal{T}$, the columns s_r are the vectorization of the rank one tensor $a_r \circ b_r \circ c_r$. Moreover, using the vectorization of tensors methods [14], we turn every rank-one tensor of outer product $\mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$ into a row vector $\mathbf{q}_r$ for $1 \le r \le R$. Thus function $f(A,B,C,\boldsymbol{\sigma})$ can be written the following equivalent forms:

$$\tfrac{1}{2}\|\mathbf{T}_{(1)} - \mathbf{AU}\|_F^2 = \tfrac{1}{2}\|\mathbf{T}_{(2)} - \mathbf{BV}\|_F^2 = \tfrac{1}{2}\|\mathbf{T}_{(3)} - \mathbf{CW}\|_F^2 = \tfrac{1}{2}\|\mathbf{t} - \boldsymbol{\sigma}\mathbf{Q}\|_F^2$$

where,

$$\mathbf{U} = \mathbf{D}(\mathbf{C} \odot \mathbf{B})^T, \mathbf{V} = \mathbf{D}(\mathbf{C} \odot \mathbf{A})^T, \mathbf{W} = \mathbf{D}(\mathbf{B} \odot \mathbf{A})^T, \mathbf{Q} = (\mathbf{q}_1^T, \cdots, \mathbf{q}_R^T)^T \tag{5}$$

and t is a vectorization for tensor $\mathcal{T}$

Then the gradients of $f(\bullet)$ on $\mathbf{A},\mathbf{B},\mathbf{C},\sigma$ are:

$$\begin{aligned} \nabla_{\mathbf{A}} f(\mathbf{A},\mathbf{B},\mathbf{C},\boldsymbol{\sigma}) &= (\mathbf{AU} - \mathbf{T}_{(1)})\mathbf{U}^T, \\ \nabla_{\mathbf{B}} f(\mathbf{A},\mathbf{B},\mathbf{C},\boldsymbol{\sigma}) &= (\mathbf{BV} - \mathbf{T}_{(2)})\mathbf{V}^T, \\ \nabla_{\mathbf{C}} f(\mathbf{A},\mathbf{B},\mathbf{C},\boldsymbol{\sigma}) &= (\mathbf{CW} - \mathbf{T}_{(3)})\mathbf{W}^T. \\ \nabla_{\boldsymbol{\sigma}} f(\mathbf{A},\mathbf{B},\mathbf{C},\boldsymbol{\sigma}) &= (\boldsymbol{\sigma}\mathbf{Q} - \mathbf{t})\mathbf{Q}^T. \end{aligned} \tag{6}$$

Then, with equations, (3-4), we can formulate linear least squares problems with respect to A, B, C and σ.

In the algorithm we starts from $(\mathbf{A}^n,\mathbf{B}^n,\mathbf{C}^n,\boldsymbol{\sigma}^n)$ and iteratively update variables $\mathbf{A},\mathbf{B},\mathbf{C}$ and then $\boldsymbol{\sigma}$ in each loop. Using the equation (2) above, we update $\mathbf{A}$ following the constraint optimization problem below:

$$\arg\min_{\mathbf{A}} \{\langle \mathbf{A} - \mathbf{A}^n, \nabla_{\mathbf{A}} f(\mathbf{A}^n,\mathbf{B}^n,\mathbf{C}^n,\boldsymbol{\sigma}^n)\rangle + \frac{sd_n}{2}\|\mathbf{A} - \mathbf{A}^n\|_F^2\}$$
$$\text{s.t. } \|\mathbf{a}_i\| = 1, i = 1,\cdots,R,$$

where $\mathbf{A} = (\mathbf{a}_1,\cdots,\mathbf{a}_R) \in \mathbb{R}^{I \times R}$, $d_n = \max\{\|\mathbf{U}^n\mathbf{U}^{nT}\|_F, 1\}$ and $\mathbf{U}^n$ is computed from $\boldsymbol{\sigma}^n,\mathbf{B}^n,\mathbf{C}^n$ by using (5). This problem is equivalent to:

$$\arg\min_{\mathbf{A}} \{\|\mathbf{A} - \mathbf{D}^n\|_F^2\} \quad \text{s.t. } \|\mathbf{a}_i\| = 1, i = 1,\cdots,R.$$

where $\mathbf{D}^n = \mathbf{A}^n - \frac{1}{sd_n}\nabla_{\mathbf{A}} f(\mathbf{A}^n,\mathbf{B}^n,\mathbf{C}^n,\boldsymbol{\sigma}^n)$. So we obtain the update of A:

$$\mathbf{a}_i^{n+1} = \mathbf{d}_i^n / \|\mathbf{d}_i^n\|, i = 1,\cdots,R,$$

where $\mathbf{a}_i^{n+1}$ and $\mathbf{d}_i^n$ are the i-th columns of $\mathbf{A}^{n+1}$ and $\mathbf{D}^n$.

Similarly, we update $\mathbf{B}$, $\mathbf{C}$ and $\boldsymbol{\sigma}$ following the same process as $\mathbf{A}$. Now to deal with $\boldsymbol{\sigma}$ we have the following optimization problem:

$$\arg\min_{\boldsymbol{\sigma}} \{\langle \boldsymbol{\sigma} - \boldsymbol{\sigma}^n, \nabla_{\boldsymbol{\sigma}} f(\mathbf{A}^{n+1},\mathbf{B}^{n+1},\mathbf{C}^{n+1},\boldsymbol{\sigma}^n)\rangle + \tfrac{s\eta_n}{2}\|\boldsymbol{\sigma} - \boldsymbol{\sigma}^n\|^2 + \lambda\|\boldsymbol{\sigma}\|_1\}$$

where $\eta_n = \max\{\|\mathbf{Q}^{n+1}\mathbf{Q}^{n+1^T}\|_F, 1\}$. This optimization problem is equivalent to:

$$\arg\min_{\boldsymbol{\sigma}} \tfrac{1}{2}\|\boldsymbol{\sigma} - \boldsymbol{\sigma}^n + \tfrac{1}{s\eta_n}\nabla_{\boldsymbol{\sigma}} f(\mathbf{A}^{n+1},\mathbf{B}^{n+1},\mathbf{C}^{n+1},\boldsymbol{\sigma}^n)\|^2 + \tfrac{\lambda}{s\eta_n}\|\boldsymbol{\sigma}\|_1.$$

So we can obtain the update form for $\boldsymbol{\alpha}$ in Algorithm 1 by using the separate soft thresholding (proof in the appendix):

$$\begin{aligned} \boldsymbol{\sigma}^{n+1} &= prox_{\frac{\lambda}{s\eta_n}}(\boldsymbol{\beta}^{n+1}) \\ &= \begin{cases} 0, & |\boldsymbol{\beta}^{n+1}| \le \frac{\lambda}{s\eta_n} \\ \boldsymbol{\beta}^{n+1} - \frac{\lambda}{s\eta_n}\text{sign}(\boldsymbol{\beta}^{n+1}), & |\boldsymbol{\beta}^{n+1}| > \frac{\lambda}{s\eta_n} \end{cases} \end{aligned} \tag{7}$$

where

$$\boldsymbol{\beta}^{n+1} = \boldsymbol{\sigma}^n - \frac{1}{s\eta_n}\nabla_{\boldsymbol{\alpha}} f(\mathbf{A}^{n+1},\mathbf{B}^{n+1},\mathbf{C}^{n+1},\boldsymbol{\alpha}^n). \tag{8}$$

Each iteration of (7) can be computed efficiently but it could suffer from slow convergence. So we implemented FISTA (fast iterative shrinkage-thresholding algorithm) [18] for our algorithm to converge faster. FISTA is the accelerated variant of ISTA [20]. Each iteration of FISTA has the following format:

$$\sigma^{n+1} = prox_{\frac{\lambda}{s\eta_n}}\left(\boldsymbol{\beta}^{n+1}\right) \tag{9}$$

$$t^{n+1} = \frac{1 + \sqrt{1 + 4t^{n^2}}}{2} \tag{10}$$

$$y^{n+1} = \sigma^n + \left(\frac{t^n - 1}{t^{n+1}}\right)\left(\sigma^{n+1} - \sigma^n\right) \tag{11}$$

See these optimization problems implemented in Algorithm 1.

Algorithm 1 Low-Rank Approximation Of Tensors (LRAT)

Input: A third order tensor with missing entries $\mathcal{A}$, an upper bound R of rank($\mathcal{A}$), a penalty parameter λ and a scale $s > 1$;

Output: An approximated tensor $\hat{\mathcal{B}}$ with an estimated rank $\hat{R}$;

1: Give an initial tensor $\mathcal{B}^0 = [\boldsymbol{\sigma}^0; \mathbf{A}^0, \mathbf{B}^0, \mathbf{C}^0]_R$.

2: Update step:

b. Update matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$:

Compute $\mathbf{U}^n$ by (5) and let $d_n = \max\{\|\mathbf{U}^n\mathbf{U}^{nT}\|_F, 1\}$.

Compute $\mathbf{D}^n$ and $\mathbf{A}^{n+1}$ by

$$\mathbf{D}^n = \mathbf{A}^n - \frac{1}{sd_n}\nabla_{\mathbf{A}} f(\mathbf{A}^n, \mathbf{B}^n, \mathbf{C}^n, \boldsymbol{\sigma}^n),$$
$$\mathbf{A}^{n+1} = \mathbf{D}^n \text{diag}(\|\mathbf{d}_1^n\|, \cdots, \|\mathbf{d}_R^n\|)^{-1}$$

where $\mathbf{d}_i^n$ is the i-th column of $\mathbf{D}^n$ for $i = 1, \cdots, R$.

Compute $\mathbf{V}^n$ by (5) and let $e_n = \max\{\|\mathbf{V}^n\mathbf{V}^{nT}\|_F, 1\}$.

Compute $\mathbf{E}^n$ and $\mathbf{B}^{n+1}$ by

$$\mathbf{E}^n = \mathbf{B}^n - \frac{1}{se_n}\nabla_{\mathbf{B}} f(\mathbf{A}^{n+1}, \mathbf{B}^n, \mathbf{C}^n, \boldsymbol{\sigma}^n),$$
$$\mathbf{B}^{n+1} = \mathbf{E}^n \text{diag}(\|\mathbf{e}_1^n\|, \cdots, \|\mathbf{e}_R^n\|)^{-1}$$

where $\mathbf{e}_i^n$ is the i-th column of $\mathbf{E}^n$ for $i = 1, \cdots, R$.

Compute $\mathbf{W}^n$ by (5) and let $f_n = \max\{\|\mathbf{W}^n\mathbf{W}^{nT}\|_F, 1\}$.

Compute $\mathbf{F}^n$ and $\mathbf{C}^{n+1}$ by

$$\mathbf{F}^n = \mathbf{C}^n - \frac{1}{sf_n}\nabla_{\mathbf{C}} f(\mathbf{A}^{n+1}, \mathbf{B}^{n+1}, \mathbf{C}^n, \boldsymbol{\sigma}^n),$$
$$\mathbf{C}^{n+1} = \mathbf{F}^n \text{diag}(\|\mathbf{f}_1^n\|, \cdots, \|\mathbf{f}_R^n\|)^{-1}$$

where $\mathbf{f}_i^n$ is the i-th column of $\mathbf{F}^n$ for $i = 1, \cdots, R$.

c. Update the row vector $\boldsymbol{\sigma}$:

Compute $\mathbf{Q}^{n+1}$ by (5) and let $\eta_n = \max\{\|\mathbf{Q}^{n+1}\mathbf{Q}^{n+1T}\|_F, 1\}$.

Compute $\boldsymbol{\beta}^{n+1}$ by (8) and use the soft thresholding:

$$\boldsymbol{\sigma}^{n+1} = prox_{\frac{\lambda}{s\eta_n}}\left(\boldsymbol{\beta}^{n+1}\right).$$

d. Update for FISTA using equations (9, 10, 11)

3: Denote the limitations by $\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}, \hat{\boldsymbol{\sigma}}$, compute $\hat{\mathcal{B}} = [\hat{\boldsymbol{\sigma}}; \hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}]_R$ and count the number $\hat{R}$ of nonzero entries in $\hat{\boldsymbol{\sigma}}$.

4: Impose constraints $\mathcal{A}(\Omega) = \mathcal{B}(\Omega)$.

5: **return** The tensor $\hat{\mathcal{B}}$ with the estimated rank $\hat{R}$.

2.2 Numerical Experiments

We implemented our algorithm on a color image of size $246 \times 257 \times 3$; see Figure 1. We recovered an estimated color image after removing 20 percent of the entries from the original color image; see Figures $2-3$.The algorithm without the FISTA implementation took 2.128045 seconds to reach convergence with a relative error of 9.5e-3. Using FISTA it reaches convergence after 2.045318 seconds with a relative error 9.6e-3.

Further, we implemented our algorithm on a color image of size $218 \times 215 \times 3$; see Figure 4. We recovered an estimated color image after removing 20 percent of the entries from the original color image; see Figures $5-6$.The algorithm without the FISTA implementation took 2.998823 seconds to reach convergence with a relative error of 9.4e-3. Using FISTA it reaches convergence after 2.882080 seconds with a relative error 9.7e-3.

3 Conclusion

We propose an alternating optimization algorithm for tensor completion. The implementation is based on four linear least squares problems and a soft thresholding algorithm. Our algorithm is fast and it does not rely on matrix SVD (or matrix singular value thresholding). In our future outlook, we would like to include some sampling rates for our algorithm as well as a regularized version of the optimization problem.

Appendix

Recall $g(\sigma) = \lambda \parallel \sigma \parallel_1$ is convex and non-differentiable. The function g can be turn into a proximal operator to find its minimum using the definition [8] below:

Definition 3.1 *Given a proper closed convex function f:* $\mathbb{R}^n \longrightarrow \mathbb{R} \bigcup \infty$*, the proximal operator scaled by* $\delta > 0$*, is a mapping from* $\mathbb{R}^n \longrightarrow \mathbb{R}$ *defined by*

$$\boldsymbol{prox}_{\delta g}(v) := \underset{y \in \mathbb{R}^n}{argmin} \quad (g(y) + \frac{1}{2\delta} \parallel y - v \parallel^2).$$

Then the proximal operator for $g(\sigma)$ is,

$$\begin{aligned} prox_g(v) &= \underset{\sigma}{\text{argmin}}\ (g(\sigma) + \frac{1}{2\delta} \parallel \sigma - v \parallel_2^2) \\ &= \underset{\sigma}{\text{argmin}}\ (\lambda \parallel \sigma \parallel_1 + \frac{1}{2\delta} \parallel \sigma - v \parallel_2^2) \\ &= \underset{\sigma}{\text{argmin}}\ (\lambda \sum_{i=1}^{n} \mid \sigma_i \mid + \frac{1}{2\delta} \sum_{i=1}^{n} (\sigma_i - v)^2) \end{aligned}$$

For $\sigma = (\sigma_1, ..., \sigma_n)$

$$\begin{aligned} (prox_{\delta g}(v)_i)_i &= \underset{\sigma_i}{\text{argmin}}\ (\lambda \mid \sigma_i \mid + \frac{1}{2\delta}(\sigma_i - v_i)^2) \\ &= \underset{\sigma_i}{\text{argmin}} \begin{cases} (\lambda - \frac{v_i}{\delta})\sigma_i + \frac{1}{2\delta}\sigma_i^2, \sigma_i > 0 \\ -(\lambda + \frac{v_i}{\delta})\sigma_i + \frac{1}{2\delta}\sigma_i^2, \sigma_i < 0 \end{cases} \end{aligned}$$

Figure 1: Original color image data

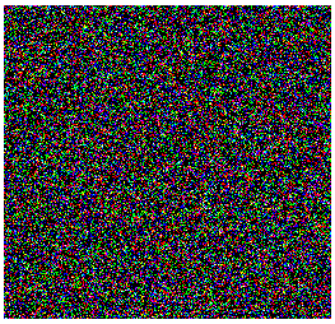

Figure 2: 20% missing entries

Figure 3: Recovered color image data

Figure 4: Original color image data

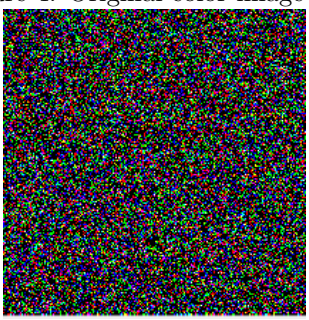

Figure 5: 20% missing entries

Figure 6: Recovered color image data

Hence,

$$prox_g(v)_i = \begin{cases} 0, & \mid v_i \mid \leq \lambda \\ v_i - \lambda sign(v_i), & \mid v_i \mid > \lambda \end{cases}$$

with $\delta = 1$

References

[1] A. Uschmajew, *Local convergence of the alternating least squares algorithm for canonical tensor approximation*, SIAM J. Matrix Anal. Appl. 33(2):639-652, 2012.

[2] R Bro, "Multi-way analysis in the food industry: models, algorithms, and applications. MRI, EPG and EMA,Ó *Proc ICSLP*, 2000.

[3] R Bro, "PARAFAC. Tutorial and applications." *Chemometrics and intelligent laboratory systems*, 1997.

[4] E. Acar, D.M. Dunlavy, and T. Kolda, *A scalable optimization approach for fitting canonical tensor decompositions*, Journal of Chemometrics 25 (2):67-86, (2011).

[5] E. Candès and T. Tao. "The Power of Convex Relaxation: Near-Optimal Matrix Completion," *IEEE Trans on Information Theory,* vol. 56, no. 5, pp. 2053-2080, (2010).

[6] J Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *ICCV*, 2009.

[7] H. Rauhut, R. Schneider, and Z. Stojanac, "Tensor completion in hierarchical tensor representations," Compressed Sensing and its Applications 2015.

[8] N. Parikh and S. Boyd, "Proximal Algorithms," Foundations and Trends in Optimization, vol. 1, no. 3, (2013), pp. 123-231.

[9] G. Tomasi and R. Bro, "PARAFAC and missing values," *Chemometrics and Intelligent Laboratory Systems*, 2005.

[10] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, 2011.

[11] P. Paatero, "A weighted non-negative least squares algorithm for three-way 'PARAFAC'factor analysis," *Chemometrics and Intelligent Laboratory Systems*, 1997.

[12] X. Geng and K. Smith-Miles, "Facial age estimation by multilinear subspace analysis," *ICASSP*, 2009.

[13] X. Wang and C. Navasca, "Low-rank approximation of tensors via sparse optimization, Numer Linear Algebra Appl. vol. 25, no. 2, (2018), pp.

[14] G. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press; 4th edition, 2013

[15] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization" Inverse Problems (2011).

[16] Y. Xu and W. Yin, *A block coordinate descent method for regularized multi-convex optimization with applications to nonnegative tensor factorization and completion*, SIAM J. Imaging Sciences, 6 (2013), pp. 1758-1789.

[17] J. Bolte, S. Sabach and M. Teboulle, *Proximal alternating linearized minimization nonconvex and nonsmooth problems*, Mathematical Programming, 146 (2014), pp. 459-494.

[18] A. Beck and M. Teboulle, *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems*, SIAM J. Imaging Sci., 2(2009), pp. 183-202.

[19] K. Bredies and D. A. Lorenz, *Linear convergence of iterative soft-thresholding*, J. Fourier Anal. Appl., 14(2008), pp. 813-837.

[20] Y. E. Nesterov, *Smooth minimization of nonsmooth functions*, Math. Program., 103(2005), pp. 127-152.