Chinese Journal of Electronics
vol. 33, no. 5, pp. 1308–1316, September 2024
https://doi.org/10.23919/cje.2023.00.035

RESEARCH ARTICLE

Efficient Nonnegative Tensor Decomposition Using Alternating Direction Proximal Method of Multipliers

Deqing WANG[1,2,3] and Guoqiang HU[4]

1. *State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China*
2. *Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China*
3. *Key Laboratory of Marine Robotics, Liaoning Province, Shenyang 110169, China*
4. *College of Artificial Intelligence, Dalian Maritime University, Dalian 116026, China*

Corresponding author: Deqing WANG, Email: deqing.wang@foxmail.com

Manuscript Received February 14, 2023; Accepted September 7, 2023

Abstract — Nonnegative CANDECOMP/PARAFAC (NCP) tensor decomposition is a powerful tool for multiway signal processing. The alternating direction method of multipliers (ADMM) optimization algorithm has become increasingly popular for solving tensor decomposition problems in the block coordinate descent framework. However, the ADMM-based NCP algorithm suffers from rank deficiency and slow convergence for some large-scale and highly sparse tensor data. The proximal algorithm is preferred to enhance optimization algorithms and improve convergence properties. In this study, we propose a novel NCP algorithm using the alternating direction proximal method of multipliers (ADPMM) that consists of the proximal algorithm. The proposed NCP algorithm can guarantee convergence and overcome the rank deficiency. Moreover, we implement the proposed NCP using an inexact scheme that alternatively optimizes the subproblems. Each subproblem is optimized by a finite number of inner iterations yielding fast computation speed. Our NCP algorithm is a hybrid of alternating optimization and ADPMM and is named A^{2}DPMM. The experimental results on synthetic and real-world tensors demonstrate the effectiveness and efficiency of our proposed algorithm.

Keywords — Tensor decomposition, Nonnegative CANDECOMP/PARAFAC, Alternating direction proximal method of multipliers, Proximal algorithm, Sparse regularization.

Citation — Deqing WANG and Guoqiang HU, "Efficient Nonnegative Tensor Decomposition Using Alternating Direction Proximal Method of Multipliers," *Chinese Journal of Electronics*, vol. 33, no. 5, pp. 1308–1316, 2024. doi: 10.23919/cje.2023.00.035.

I. Introduction

Nonnegative CANDECOMP/PARAFAC (NCP) decomposition is an essential tool to process nonnegative tensor in signal processing and has been widely applied to various domains of multiway data analyses [1]–[4]. It can extract intrinsic nonnegative, sparse and low-rank components from the tensor data. NCP can be represented as a constrained nonconvex optimization problem and solved in the block coordinate descent (BCD) framework. The BCD-based NCP algorithm is favourable for decomposing large-scale tensor data, especially those contaminated by considerable noise. To efficiently solve the constrained tensor decomposition, numerous optimization methods have been developed, such as the alternating proximal gradient (APG) [5]–[7], alternating nonnegative quadratic programming (ANQP) [8], [9] and alternating optimization-based alternating direction method of multipliers (AO-ADMM) [10].

Alternating direction method of multipliers (ADMM) is a conventional optimization method and has become increasingly popular in signal processing and machine learning [11], [12]. ADMM has been employed and adapted for many types of tensor decomposition problems. Liavas *et al.* proposed the ADMM-based NCP algorithm and its parallel implementation [13]. Later, Huang *et al.*

Associatet Editor: Prof. Meng WANG, Hefei University of Technology.

proposed the AO-ADMM algorithm [10], which combines the ADMM algorithm and an alternating optimization scheme in the BCD framework. AO-ADMM updates each subproblem of NCP using ADMM with multiple inner iterations and shows outstanding performances for the whole NCP problem. Recently, AO-ADMM has been extended to solve many other constrained tensor decomposition problems, such as PARAFAC2 [14] and coupled matrix and tensor factorizations (CMTF) [15]. Moreover, the ADMM is flexible in coping with many types of regularization items [16], [17] in the objective function and solving structured low-rank factorization [18], [19].

Although the ADMM method is favourable for tensor analysis, it still suffers from ill-posed problems. In practice, when tensor data are highly sparse and the tensor rank is very small, the decomposed factor matrices will not be of full column rank. In such a case, the ADMM-based tensor decomposition will suffer from rank deficiency and, therefore, degenerate [8]. From the perspective of mathematical optimization, many techniques have been proposed to enhance ADMM. The combination of the proximal algorithm and method of multipliers is a significant enhancement of ADMM, termed the alternating direction proximal method of multipliers (ADPMM) [20]. The advantage of the proximal algorithm in ADPMM is that it manufactures strongly convex minimization problems [20], [21]. On the other hand, when the objective function is nonlinear or nonsmooth, the linearized ADMM or proximal linearized ADMM is preferred [22]–[25].

In this study, we first extend the ADPMM to the NCP problem. In the BCD framework [8], NCP is optimized by alternatively solving each subproblem in the outer loop. We employ the proximal algorithm to the subproblem by adding a proximal regularization item to the objective function. In this way, a strongly convex surrogate function is constructed for the subproblem. Afterwards, the subproblem is optimized by alternatively updating the primal variable and dual variable in the inner loop. The ADPMM in the subproblem can overcome the rank deficiency that NCP may encounter when the tensor data are highly sparse. Second, we implement the ADPMM-based NCP using an inexact scheme. Specifically, the subproblem is updated by a finite number of inner iterations. The inner iteration will stop when a predefined maximum number of iterations or tolerance is satisfied. The inexact scheme can significantly improve the efficiency of NCP [7], [8].

The rest of this paper is organized as follows. Section II introduces the basic ideas of ADPMM and NCP. In Section III, we present the proposed NCP algorithm and its extension to sparse NCP. Section IV describes the experiments on synthetic and real-world tensors. Finally, we conclude the paper in Section V.

II. Preliminaries

In this paper, we use normal lowercase letters (e.g., x), boldface lowercase letters (e.g., $\boldsymbol{x}$), boldface uppercase letters (e.g., $\boldsymbol{X}$) and boldface script letters (e.g., $\boldsymbol{\mathcal{X}}$) to denote scalars, vectors, matrices and tensors, respectively. The operator $\circ$ represents the outer product of vectors, $\odot$ represents the Khatri-Rao product, $*$ represents the Hadamard product that is the elementwise matrix product, $\langle\ \rangle$ represents the inner product, and $[\![\ ]\!]$ represents Kruskal operator. $\|\ \|_{\mathrm{F}}$ denotes Frobenius norm, and $\|\ \|_1$ denotes l_1-norm. A nonnegative matrix $\boldsymbol{X}$ is expressed by $\boldsymbol{X} \geq \mathbf{0}$. For a vector $\boldsymbol{u} \in \mathbb{R}^d$ and a symmetric positive semidefinite matrix $\boldsymbol{M} \in \mathbb{S}_+^d := \{\boldsymbol{M} : \boldsymbol{M} = \boldsymbol{M}^{\mathrm{T}}, \boldsymbol{M} \geq \mathbf{0}\}$, the seminorm induced by $\boldsymbol{M}$ is defined by $\|\boldsymbol{u}\|_{\boldsymbol{M}} = \langle \boldsymbol{u}, \boldsymbol{M}\boldsymbol{u}\rangle^{1/2}$. The objective function of tensor decomposition is denoted by the script letter $\mathcal{F}$.

Given a nonnegative Nth-order tensor $\boldsymbol{\mathcal{X}}$, i.e., $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the NCP can be represented by the following minimization problem:

$$\begin{aligned} \min_{\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \ldots, \boldsymbol{A}^{(N)}} & \left\{ \frac{1}{2} \left\| \boldsymbol{\mathcal{X}} - [\![\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \ldots, \boldsymbol{A}^{(N)}]\!] \right\|_{\mathrm{F}}^2 \right. \\ & \left. + \sum_{n=1}^{N} \boldsymbol{r}\left(\boldsymbol{A}^{(n)}\right) \right\} \\ & \text{s.t. } \boldsymbol{A}^{(n)} \geq \mathbf{0} \text{ for } n = 1, 2, \ldots, N \end{aligned} \tag{1}$$

where $\boldsymbol{A}^{(n)} \in \mathbb{R}^{I_n \times R}$ for $n = 1, 2, \ldots, N$ are the estimated factor matrices in different modes, $\boldsymbol{r}\left(\boldsymbol{A}^{(n)}\right)$ is the regularization item, I_n is the size in mode-n, and R is the predefined number of components that can be seen as the selected rank.

Let $\boldsymbol{X}_{(n)}$ represent the mode-n unfolding of original tensor $\boldsymbol{\mathcal{X}}$. The mode-n unfolding of $[\![\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \ldots, \boldsymbol{A}^{(N)}]\!]$ can be written as $\boldsymbol{A}^{(n)}\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}$, where $\boldsymbol{B}^{(n)} = \left(\boldsymbol{A}^{(N)} \odot \cdots \odot \boldsymbol{A}^{(n+1)} \odot \boldsymbol{A}^{(n-1)} \odot \cdots \odot \boldsymbol{A}^{(1)}\right)$. In the BCD framework [5], [8], each factor matrix $\boldsymbol{A}^{(n)}$ is updated alternatively by the following subproblem:

$$\begin{aligned} \min_{\boldsymbol{A}^{(n)}} & \frac{1}{2} \left\| \boldsymbol{X}_{(n)} - \boldsymbol{A}^{(n)} \left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \boldsymbol{r}\left(\boldsymbol{A}^{(n)}\right) \\ & \text{s.t. } \boldsymbol{A}^{(n)} \geq \mathbf{0} \end{aligned} \tag{2}$$

III. Novel NCP Algorithms

In this section, we introduce our novel NCP algorithm. In the BCD framework, we alternatively optimize the subproblem (2) with respect to each factor matrix using the ADPMM. The full name of our NCP algorithm is alternating optimization-based alternating direction proximal method of multipliers, abbreviated to A^{2}DPMM, in which the first "alternating" means that the factor matrices are updated alternatively as subproblems in the outer loop of NCP; In contrast, the second "alternating" means that the primal, auxiliary and dual variables are updated alternatively in the inner loop of each subproblem.

1. A^{2}DPMM for NCP

In this part, we present the details of the A^{2}DPMM algorithm to realize NCP. The computation steps of A^{2}DPMM are similar to that of ADMM for NCP [26]. However, it outperforms ADMM because of the proximal regularization items in ADPMM. We explain how to use ADPMM to solve the subproblem of NCP as follows. Introducing an auxiliary variable $\widetilde{\boldsymbol{A}}^{(n)}\in\mathbb{R}^{I_n\times R}$ for the primal variable $\boldsymbol{A}^{(n)}$, we reform the subproblem (2) as

$$\begin{gathered}\min_{\boldsymbol{A}^{(n)}}\frac{1}{2}\left\|\boldsymbol{X}_{(n)}-\boldsymbol{A}^{(n)}\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}\right\|_{\mathrm{F}}^2+\boldsymbol{r}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)\\ \text{s.t. }\boldsymbol{A}^{(n)}=\widetilde{\boldsymbol{A}}^{(n)},\widetilde{\boldsymbol{A}}^{(n)}\geq 0\end{gathered}\tag{3}$$

Next, we construct two functions for (3):

$$\begin{aligned}&\mathcal{F}\left(\boldsymbol{A}^{(n)}\right)=\frac{1}{2}\left\|\boldsymbol{X}_{(n)}-\boldsymbol{A}^{(n)}\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}\right\|_{\mathrm{F}}^2\\&\mathcal{G}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)=\boldsymbol{r}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)\end{aligned}$$

We employ $\boldsymbol{\Lambda}^{(n)}\in\mathbb{R}^{I_n\times R}$ as the Lagrange multiplier (dual variable). Given a point $\left(\boldsymbol{A}_k^{(n)},\widetilde{\boldsymbol{A}}_k^{(n)},\boldsymbol{\Lambda}_k^{(n)}\right)$ in the $(k+1)$th iteration, we define the proximal augmented Lagrangian function of (3) by

$$\begin{aligned}&\mathcal{P}_k\left(\boldsymbol{A}^{(n)},\widetilde{\boldsymbol{A}}^{(n)},\boldsymbol{\Lambda}^{(n)}\right)\\&=\mathcal{F}\left(\boldsymbol{A}^{(n)}\right)+\mathcal{G}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)\\&\quad+\left\langle\boldsymbol{\Lambda}^{(n)},\boldsymbol{A}^{(n)}-\widetilde{\boldsymbol{A}}^{(n)}\right\rangle+\frac{\rho_n}{2}\left\|\boldsymbol{A}^{(n)}-\widetilde{\boldsymbol{A}}^{(n)}\right\|_{\mathrm{F}}^2\\&\quad+\frac{1}{2}\left\|\boldsymbol{A}^{(n)}-\boldsymbol{A}_k^{(n)}\right\|_{\boldsymbol{M}_1}^2+\frac{1}{2}\left\|\widetilde{\boldsymbol{A}}^{(n)}-\widetilde{\boldsymbol{A}}_k^{(n)}\right\|_{\boldsymbol{M}_2}^2\end{aligned}\tag{4}$$

We select $\rho_n=\operatorname{tr}\left[\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}\boldsymbol{B}^{(n)}\right]/R$ according to the empirical setting in [10].

Afterwards, we update the sequence $\boldsymbol{A}_k^{(n)}$, $\widetilde{\boldsymbol{A}}_k^{(n)}$ and $\boldsymbol{\Lambda}_k^{(n)}$ for the subproblem (3) as follows:

1) Updating the primal variable $\boldsymbol{A}^{(n)}$

$$\begin{aligned}\boldsymbol{A}_{k+1}^{(n)}&=\underset{\boldsymbol{A}^{(n)}}{\operatorname{argmin}}\,\mathcal{P}_k\left(\boldsymbol{A}^{(n)},\widetilde{\boldsymbol{A}}_k^{(n)},\boldsymbol{\Lambda}_k^{(n)}\right)\\&=\underset{\boldsymbol{A}^{(n)}}{\operatorname{argmin}}\left\{\mathcal{F}\left(\boldsymbol{A}^{(n)}\right)+\left\langle\boldsymbol{\Lambda}_k^{(n)},\boldsymbol{A}^{(n)}-\widetilde{\boldsymbol{A}}_k^{(n)}\right\rangle\right.\\&\qquad\left.+\frac{\rho_n}{2}\left\|\boldsymbol{A}^{(n)}-\widetilde{\boldsymbol{A}}_k^{(n)}\right\|_{\mathrm{F}}^2+\frac{1}{2}\left\|\boldsymbol{A}^{(n)}-\boldsymbol{A}_k^{(n)}\right\|_{\boldsymbol{M}_1}^2\right\}\end{aligned}\tag{5}$$

We select $\boldsymbol{M}_1=\alpha_n\boldsymbol{I}_R$, where $\boldsymbol{I}_R\in\mathbb{R}^{R\times R}$ is an identity matrix, and $\alpha_n\in\boldsymbol{\alpha}=[\alpha_1,\dots,\alpha_N]^{\mathrm{T}}$ can be seen as the regularization parameter of the proximal item. Supposing $\boldsymbol{\Phi}^{(n)}=\frac{1}{\rho_n}\boldsymbol{\Lambda}^{(n)}$ and computing the partial derivative $\frac{\partial\mathcal{P}_k}{\partial\boldsymbol{A}^{(n)}}=0$, we obtain the solution

$$\begin{aligned}\boldsymbol{A}_{k+1}^{(n)}=&\left[\boldsymbol{X}_{(n)}\boldsymbol{B}^{(n)}+\alpha_n\boldsymbol{A}_k^{(n)}+\rho_n\left(\widetilde{\boldsymbol{A}}_k^{(n)}-\boldsymbol{\Phi}_k^{(n)}\right)\right]\\&\cdot\left[\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}\boldsymbol{B}^{(n)}+\alpha_n\boldsymbol{I}_R+\rho_n\boldsymbol{I}_R\right]^{-1}\end{aligned}\tag{6}$$

2) Updating the auxiliary variable $\widetilde{\boldsymbol{A}}^{(n)}$

$$\begin{aligned}\widetilde{\boldsymbol{A}}_{k+1}^{(n)}&=\underset{\widetilde{\boldsymbol{A}}^{(n)}\geq 0}{\operatorname{argmin}}\,\mathcal{P}_k\left(\boldsymbol{A}_{k+1}^{(n)},\widetilde{\boldsymbol{A}}^{(n)},\boldsymbol{\Lambda}_k^{(n)}\right)\\&=\underset{\widetilde{\boldsymbol{A}}^{(n)}\geq 0}{\operatorname{argmin}}\left\{\mathcal{G}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)+\left\langle\boldsymbol{\Lambda}_k^{(n)},\boldsymbol{A}_{k+1}^{(n)}-\widetilde{\boldsymbol{A}}^{(n)}\right\rangle\right.\\&\qquad+\frac{\rho_n}{2}\left\|\boldsymbol{A}_{k+1}^{(n)}-\widetilde{\boldsymbol{A}}^{(n)}\right\|_{\mathrm{F}}^2\\&\qquad\left.+\frac{1}{2}\left\|\widetilde{\boldsymbol{A}}^{(n)}-\widetilde{\boldsymbol{A}}_k^{(n)}\right\|_{\boldsymbol{M}_2}^2\right\}\end{aligned}\tag{7}$$

We select $\boldsymbol{M}_2=\boldsymbol{0}$. Using the proximal operator, we obtain the following closed form solution

$$\widetilde{\boldsymbol{A}}_{k+1}^{(n)}=\operatorname{prox}_{\frac{\mathcal{G}}{\rho_n}}\left(\boldsymbol{A}_{k+1}^{(n)}+\tfrac{1}{\rho_n}\boldsymbol{\Lambda}_k^{(n)}\right)\tag{8}$$

For example, if sparsity is expected to impose on the factors, we can set $\mathcal{G}\left(\widetilde{\boldsymbol{A}}^{(n)}\right)=\beta_n\sum_{r=1}^{R}\left\|\widetilde{\boldsymbol{a}}_r^{(n)}\right\|_1$, where $\widetilde{\boldsymbol{a}}_r^{(n)}$ is the rth column of $\widetilde{\boldsymbol{A}}^{(n)}$ and β_n is the positive regularization parameter in vector $\boldsymbol{\beta}\in\mathbb{R}^{N\times 1}$. With both nonnegative constraint and sparse regularization, the updating of $\widetilde{\boldsymbol{A}}^{(n)}$ becomes

$$\widetilde{\boldsymbol{A}}_{k+1}^{(n)}=\max\left\{0,\ \boldsymbol{A}_{k+1}^{(n)}+\tfrac{1}{\rho_n}\boldsymbol{\Lambda}_k^{(n)}-\tfrac{\beta_n}{\rho_n}\boldsymbol{E}\right\}\tag{9}$$

where all the elements in $\boldsymbol{E}\in\mathbb{R}^{I_n\times R}$ equal to one.

3) Updating the Lagrange multiplier (dual variable) $\boldsymbol{\Lambda}^{(n)}$

$$\begin{aligned}\boldsymbol{\Lambda}_{k+1}^{(n)}&=\underset{\boldsymbol{\Lambda}^{(n)}}{\operatorname{argmin}}\left\{-\mathcal{P}_k\left(\boldsymbol{A}_{k+1}^{(n)},\widetilde{\boldsymbol{A}}_{k+1}^{(n)},\boldsymbol{\Lambda}^{(n)}\right)\right.\\&\qquad\left.+\frac{1}{2\rho_n}\left\|\boldsymbol{\Lambda}^{(n)}-\boldsymbol{\Lambda}_k^{(n)}\right\|_{\mathrm{F}}^2\right\}\\&=\underset{\boldsymbol{\Lambda}^{(n)}}{\operatorname{argmin}}\left\{-\left\langle\boldsymbol{\Lambda}^{(n)},\boldsymbol{A}_{k+1}^{(n)}-\widetilde{\boldsymbol{A}}_{k+1}^{(n)}\right\rangle\right.\\&\qquad\left.+\frac{1}{2\rho_n}\left\|\boldsymbol{\Lambda}^{(n)}-\boldsymbol{\Lambda}_k^{(n)}\right\|_{\mathrm{F}}^2\right\}\end{aligned}\tag{10}$$

Afterwards, we obtain the solution

$$\boldsymbol{\Lambda}_{k+1}^{(n)}=\boldsymbol{\Lambda}_k^{(n)}+\rho_n\left(\boldsymbol{A}_{k+1}^{(n)}-\widetilde{\boldsymbol{A}}_{k+1}^{(n)}\right)\tag{11}$$

Equation (11) can be represented equivalently in the scaled form as follows:

$$\boldsymbol{\Phi}_{k+1}^{(n)}=\boldsymbol{\Phi}_k^{(n)}+\boldsymbol{A}_{k+1}^{(n)}-\widetilde{\boldsymbol{A}}_{k+1}^{(n)}\tag{12}$$

The implementation of A^{2}DPMM is summarized in Algorithm 1.

Based on Algorithm 1, we analyze the computation complexity of A^{2}DPMM. In an outer iteration of the A^{2}DPMM algorithm, the computation mainly focuses on $\boldsymbol{X}_{(n)}\boldsymbol{B}^{(n)}$, $\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}}\boldsymbol{B}^{(n)}$ and the inner loop. The computational complexities of these three parts are respectively $O\left(R\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N}I_{\tilde{n}}+R\prod_{\tilde{n}=1}^{N}I_{\tilde{n}}\right)$, $O\left(R^2\sum_{\tilde{n}=1,\tilde{n}\neq n}^{N}I_{\tilde{n}}\right)$ and $O\left(\bar{K}\times\left(I_nR^2+R^3\right)\right)$, where $\bar{K}$ is the predefined inner iteration number.

Algorithm 1 A^{2}DPMM for NCP

```
Input: $\boldsymbol{\mathcal{X}}$, $R$, $\alpha$, $\boldsymbol{\beta}$.
Output: $\boldsymbol{A}^{(n)}$, $n = 1, 2, \ldots, N$.
Initialize $\boldsymbol{A}^{(n)} \in \mathbb{R}^{I_n \times R}$, $n = 1, 2, \ldots, N$, using nonnegative random numbers, and initialize $\boldsymbol{\Phi}^{(n)} \in \mathbb{R}^{I_n \times R}$ using zeros;
// The outer loop starts here
repeat
    for $n = 1$ to $N$ do
        Make mode-$n$ unfolding of $\boldsymbol{\mathcal{X}}$ as $\boldsymbol{X}_{(n)}$ and compute $\boldsymbol{X}_{(n)}\boldsymbol{B}^{(n)}$;
        Compute $\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}} \boldsymbol{B}^{(n)}$;
        $\rho_n = \operatorname{tr}\left[\left(\boldsymbol{B}^{(n)}\right)^{\mathrm{T}} \boldsymbol{B}^{(n)}\right] / R$;
        // The inner loop starts here
        repeat
            Update the primal variable $\boldsymbol{A}^{(n)}$ using (6);
            Update the auxiliary variable $\widetilde{\boldsymbol{A}}^{(n)}$ using (8);
            Update the dual variable $\boldsymbol{\Phi}^{(n)}$ using (12);
        until inner termination criterion is reached
        // The inner loop ends here
        $\boldsymbol{A}^{(n)} = \widetilde{\boldsymbol{A}}^{(n)}$;
    end
until outer termination criterion is reached
// The outer loop ends here
return $\boldsymbol{A}^{(n)}$, $n = 1, 2, \ldots, N$.
```

2. Convergence analysis

The proximal augmented Lagrangian in (4) manufactures strong convexity for the subproblem of NCP and sparse NCP. We set $\boldsymbol{M}_1 = \alpha_n \boldsymbol{I}_R > \boldsymbol{0}$ and $\boldsymbol{M}_2 = \boldsymbol{0}$. The proximal augmented Lagrangian function $\mathcal{P}_k$ in (4) can be represented as

$$\begin{aligned}\mathcal{P}_k =& \frac{1}{2}\left\| \begin{pmatrix} \boldsymbol{X}_{(n)}^{\mathrm{T}} \\ \sqrt{\alpha_n}\left(\boldsymbol{A}_k^{(n)}\right)^{\mathrm{T}} \end{pmatrix} - \begin{pmatrix} \boldsymbol{B}^{(n)} \\ \sqrt{\alpha_n}\boldsymbol{I}_R \end{pmatrix} \left(\boldsymbol{A}^{(n)}\right)^{\mathrm{T}} \right\|_{\mathrm{F}}^2 \\ &+ \left\langle \boldsymbol{\Lambda}^{(n)}, \boldsymbol{A}^{(n)} - \widetilde{\boldsymbol{A}}^{(n)} \right\rangle + \frac{\rho_n}{2}\left\| \boldsymbol{A}^{(n)} - \widetilde{\boldsymbol{A}}^{(n)} \right\|_{\mathrm{F}}^2\end{aligned} \tag{13}$$

The concatenated matrix $\begin{pmatrix} \boldsymbol{B}^{(n)} \\ \sqrt{\alpha_n}\boldsymbol{I}_R \end{pmatrix}$ must be of full column rank. Hence, the selection of $\boldsymbol{M}_1 = \alpha_n \boldsymbol{I}_R > \boldsymbol{0}$ will introduce a full column rank variable in the proximal augmented Lagrangian function of the subproblem. According to Theorem 5.6 in [20], the ADPMM can stably decrease the objective function of the subproblem and guarantee that the subproblem converges to a saddle point. Furthermore, in the BCDframework, according to Proposition 3.7.1 in [27], the A^{2}DPMM can guarantee that the NCP converges to a stationary point.

Furthermore, according to [20], there are many choices of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ that guarantee the convergence of diverse optimization applications.

3. Inexact scheme

The primal variable $\boldsymbol{A}^{(n)}$, auxiliary variable $\widetilde{\boldsymbol{A}}^{(n)}$ and dual variable $\boldsymbol{\Phi}^{(n)}$ are updated alternatively in the inner loop of the subproblem. If the inner loop terminates based on a unique stopping criterion tolerance, the subproblem may iterate tens or hundreds of times. Too many inner iterations will make the tensor decomposition algorithm inefficient. Therefore, we employ an inexact scheme for the subproblem. Specifically, we update the variables in the subproblem by a finite number of inner iterations, e.g., 10 or 20. The inner loop will terminate when the predefined number is reached, yielding an inexact subproblem solution. The inexact scheme will reduce the computational complexity of the inner loop and speed up the tensor decomposition significantly [7], [8].

IV. Experiments and Results

We evaluated the proposed A^{2}DPMM algorithm on both synthetic and real-world tensors. We compared the A^{2}DPMM with three NCP algorithms: PROX-ANQP [8], AO-ADMM [10] and Fast-HALS [28].

For A^{2}DPMM, AO-ADMM and PROX-ANQP, the algorithms were implemented using the outer loop that optimizes the NCP problem and the inner loop that optimizes each subproblem. The outer loop was terminated based on the change of the relative error (see Section 4.3.1 in [8]). In contrast, the termination of the inner loop was based on the relative residual of each factor matrix with a dynamically adjustable threshold (see Section 4.3.2 in [8]). Furthermore, using the inexact scheme, the inner loop will also terminate when the predefined maximum number of inner iterations is reached.

The experiments were conducted on a computer with an Intel Core i7-9750H 2.6 GHz CPU, 32 GB memory, a 64-bit Windows system and MATLAB R2022a. The fundamental tensor computation was based on Tensor Toolbox v3.2.1 [29].

1. Synthetic tensor data

In the first experiment, we constructed a synthetic third-order sparse tensor $\boldsymbol{\mathcal{X}}_{\mathrm{SYN}} = [\![\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \boldsymbol{A}^{(3)}]\!]$ similar to that in Section 5.1.2 in [8]. The tensor size is 500 × 500 × 500, and the actual number of components is 100 (rank = 100). Nonnegative noise with signal-to-noise ratio (SNR) of 10 dB was added.

We set the outer stopping tolerance by 1×10^{-6}, the maximum running time by 600 seconds, the maximum number of inner iterations by 50 and the initial number of components by 200. The true sparsity levels of three factor matrices are 0.95, 0.75 and 0.5 respectively, where the maximum of 1 means a zero matrix and the minimum of 0 means a fully dense matrix [8]. We set α_n in (6) by 1×10^{-4}. The values of $\beta_n = 0, 0.5, 1, 3, 5$ and 7 were selected for all algorithms to evaluate their abilities to impose sparsity. The selection of β_n was based on trying and testing for each tensor. We record the values of the objective function (Obj), relative error (RelErr), running time in seconds (in wall-clock time), outer iteration number (Iter), number of nonzero components (NNC), sparsity level (Spars) and peak-signal-to-noise ratio (PSNR) of all estimated factor matrices. All algorithms were run 30 times. The average values of all criteria are computed and shown in Table 1.

Table 1 Comparison of NCP algorithms on $\boldsymbol{\mathcal{X}}_{\text{SYN}} \in \mathbb{R}^{500\times500\times500}$

Method	β_n	Obj	RelErr	Time (s)	Iter	NNC	Spars$_1$	Spars$_2$	Spars$_3$	PSNR$_1$	PSNR$_2$	PSNR$_3$
A^{2}DPMM	0	2.46×10^6	0.2207	64.6	41.5	200.00	0.260	0.279	0.041	49.26	46.92	46.72
	0.5	2.54×10^6	0.2208	64.2	41.1	190.77	0.679	0.404	0.089	49.22	48.18	48.10
	1	2.62×10^6	0.2209	51.3	33.2	138.37	0.811	0.511	0.097	58.91	48.93	49.72
	3	2.83×10^6	0.2209	54.0	35.2	103.10	0.915	0.683	0.354	69.66	50.90	49.33
	5	2.99×10^6	0.2210	56.9	37.2	101.63	0.930	0.712	0.477	69.03	63.06	56.78
	7	3.11×10^6	0.2210	66.2	43.7	101.27	0.929	0.718	0.499	68.43	66.47	64.66
AO-ADMM	0	2.46×10^6	0.2207	65.0	41.7	200.00	0.257	0.277	0.041	49.16	47.01	46.66
	0.5	2.54×10^6	0.2208	62.6	40.5	189.70	0.681	0.405	0.088	49.46	48.20	48.07
	1	2.62×10^6	0.2209	50.1	32.4	137.87	0.805	0.507	0.091	58.99	48.91	49.80
	3	2.83×10^6	0.2209	53.9	35.0	103.43	0.914	0.679	0.355	69.56	50.35	49.40
	5	3.00×10^6	0.2210	56.0	36.5	101.53	0.928	0.712	0.475	69.14	62.54	56.28
	7	3.08×10^6	0.2210	88.1	59.2	100.97	0.840	0.666	0.499	65.78	63.71	63.56
PROX-ANQP	0	2.46×10^6	0.2207	74.7	33.6	200.00	0.230	0.277	0.065	49.33	47.70	45.79
	0.5	2.54×10^6	0.2208	74.1	35.1	199.27	0.624	0.439	0.203	49.17	47.14	46.13
	1	2.62×10^6	0.2209	57.1	29.1	160.77	0.803	0.623	0.367	54.47	51.85	51.25
	3	2.82×10^6	0.2209	64.3	38.0	101.87	0.915	0.713	0.498	70.01	67.64	66.46
	5	2.95×10^6	0.2209	78.6	48.5	101.47	0.928	0.717	0.500	69.83	67.92	66.50
	7	3.06×10^6	0.2210	81.8	51.1	101.37	0.922	0.716	0.500	68.83	67.52	66.24
Fast-HALS	0	2.46×10^6	0.2208	87.6	66.6	200.00	0.772	0.450	0.013	41.26	52.68	57.60
	0.5	2.94×10^6	0.2212	55.9	42.5	200.00	0.613	0.507	0.109	44.74	48.67	48.44
	1	3.38×10^6	0.2225	88.2	66.8	200.00	0.395	0.475	0.162	47.63	50.85	42.15
	3	5.02×10^6	0.2323	123.3	93.0	200.00	0.206	0.409	0.376	43.54	48.99	36.74
	5	6.45×10^6	0.2468	100.4	76.1	200.00	0.119	0.369	0.462	40.16	47.32	34.74
	7	7.71×10^6	0.2639	88.1	66.9	200.00	0.089	0.332	0.510	38.16	45.87	33.39

Note: Ground truth levels: Spars$_1$=0.95, Spars$_2$=0.75 and Spars$_3$=0.5.
Spars$_n$: Sparsity level of the mode-n estimated factor.
NNC: Number of nonzero components.

From the perspective of running time, it can be seen in Table 1 that the A^{2}DPMM is very fast in comparison with AO-ADMM, PROX-ANQP and Fast-HALS. The Fast-HALS is the slowest among four algorithms with different values of β_n, because Fast-HALS does not employ the inexact scheme. From the perspective of sparsity recovering, when the sparse regularization parameter β_n is set by 5 or 7, the NNC values of A^{2}DPMM, AO-ADMM and PROX-ANQP are close to the true number of components of 100. Moreover, when $\beta_n = 5$ or 7, the PSNR values of A^{2}DPMM and PROX-ANQP are higher than those of AO-ADMM and Fast-HALS, and the factor sparsity levels of A^{2}DPMM and PROX-ANQP are very close to the true values of 0.95, 0.75 and 0.5 respectively. The reason is that the proximal algorithm in A^{2}DPMM and PROX-ANQP can overcome the rank deficiency that may occur in tensor decomposition for recovering sparse components [8]. Fast-HALS fails in recovering the true sparse components and it has high relative error with $\beta_n > 0$. The reason is that Fast-HALS does not use the proximal algorithm and is prone to sparse regularization.

The synthetic experimental results prove that A^{2}DPMM is efficient for NCP decomposition and effective for recovering more accurate sparse components.

2. Real-world video tensor

In the second experiment, we generated a real-world third-order dense tensor using surveillance video data from the UCSD Anomaly Detection Dataset[*1]. The dataset contains 70 video samples, each sample contains 200 video images, and each image has a height of 158

[*1] http://www.svcl.ucsd.edu/projects/anomaly/dataset.html

pixels and a width of 238 pixels. Hence, the size of the video tensor is pixel × pixel × images = 158 × 238 × 14000.

We set the outer stopping tolerance by 1×10^{-6}, the maximum running time by 1200 seconds, the maximum number of inner iterations by 20 and the initial number of components by 20. We recorded only the values of the objective function (Obj), relative error (RelErr), running time in seconds (in wall-clock time) and outer iteration number (Iter). We do not consider sparse regularization for this dense video tensor ($\beta_n = 0$). All algorithms were run 30 times, and the average values are shown in Table 2.

Table 2 Comparison of NCP algorithms on video tensor (β_n=0)

Method	Obj	RelErr	Time (s)	Iter
A^{2}DPMM	1.49×10^6	0.1756	236.4	268.2
AO-ADMM	1.49×10^6	0.1755	271.6	295.6
PROX-ANQP	1.49×10^6	0.1754	245.3	286.3
Fast-HALS	1.49×10^6	0.1758	600.0	749.1

The results in Table 2 show that A^{2}DPMM converges very fast and terminates in much less time for the large-scale dense tensor compared with PROX-ANQP, AO-ADMM and Fast-HALS. We plot the objective function value curves of all four NCP algorithms in Figure 1. It is clear in Figure 1 that the A^{2}DPMM converges faster than the other three algorithms.

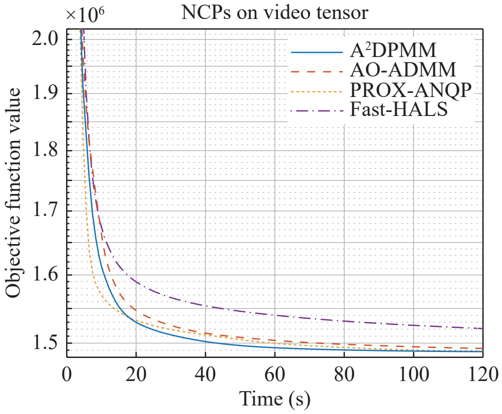

Figure 1 The objective function value curves of NCPs on the third-order video tensor (β_n=0).

3. Real-world fMRI tensor

In the third experiment, we employ a brain signal tensor of resting-state functional magnetic resonance imaging (fMRI) data. The data come from the Tao Wu dataset[*2], which consists of 20 patients with Parkinson's disease (PD) and 20 age-matched controls (healthy subjects) [30]. Two hundred and thirty-nine temporal points were originally recorded in each subject's data. The first ten temporal points were removed after standard fMRI data preprocessing with 229 temporal points left. Next, the average blood oxygenation level dependent (BOLD) signal time series were computed using a published brain parcellation comprising 129 regions of interest (ROIs) [31], [32]. Afterwards, these time series were transformed into the time-frequency domain using continuous wavelet transform (CWT), which has 50 spectral points representing the frequency information between 0 and 0.25 Hz. Finally, a fourth-order nonnegative fMRI tensor was generated, whose size was space × frequency × time × subject = 129 × 50 × 229 × 40. The fMRI data collection and preprocessing details are described in [33].

The settings of the algorithm parameters are the same as those for the video tensor in Section IV.2. We also evaluate the performances using the values of Obj, RelErr, Time and Iter. The average values of 30 runs for each algorithm are shown in Table 3. The objective function value curves of all four algorithms on the fourth-order fMRI tensor are shown in Figure 2. Table 3 and Figure 2 demonstrate that the proposed A^{2}DPMM algorithm has faster convergence compared with the other three algorithms.

Table 3 Comparison of NCP algorithms on fMRI tensor (β_n=0)

Method	Obj	RelErr	Time (s)	Iter
A^{2}DPMM	1.05×10^6	0.4629	41.9	136.4
AO-ADMM	1.05×10^6	0.4629	44.7	145.6
PROX-ANQP	1.05×10^6	0.4628	43.9	143.5
Fast-HALS	1.05×10^6	0.4630	75.7	262.8

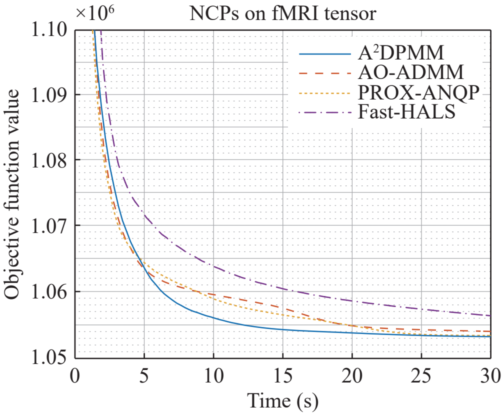

Figure 2 The objective function value curves of NCPs on the fourth-order fMRI tensor (β_n=0).

Furthermore, we tested the ability of the A^{2}DPMM algorithm to impose sparsity on the factor matrices of the fMRI tensor decomposition. The sparse regularization parameters $\beta_n = 0, 10, 50, 100, 150$ and 200 were tested for A^{2}DPMM. We recorded the average values of the NNC and the Spars of all the estimated factor matrices after 30 runs in Table 4. The results demonstrate that

[*2] https://fcon_1000.projects.nitrc.org/indi/retro/parkinsons.html

our A^{2}DPMM algorithm can successfully impose sparsity on the factor matrices and reduce the number of nonzero components by increasing the value of the sparsity regularization parameter β_n.

We also selected four groups of components extracted by the A^{2}DPMM algorithm with $\beta_n = 200$, as shown in Figure 3. Groups (a), (b) and (c) contain many sparse components. By visual observation of the subject components in the first three groups, there are significant differences between the patients and controls. The differences mean that the corresponding sparse brain features in these groups may often appear in healthy controls, and the highlighted positions of the brain shown in the spatial components[*3] might indicate illnesses in the patients. On the other hand, group (d) shows a group of dense components, which are common for both controls and patients. The results in Figure 3 prove that our NCP algorithm of A^{2}DPMM is an efficient and effective tool for analysing fMRI tensor data. Furthermore, statistical analysis of the extracted components is necessary for a rigorous analysis of brain functions from a neuro-

Table 4 Sparse regularization performances of A^{2}DPMM on fMRI tensor

β_n	NNC	Spars$_1$	Spars$_2$	Spars$_3$	Spars$_4$
0	20.00	0.001	0.223	0.110	0.137
10	18.57	0.072	0.290	0.183	0.193
50	15.73	0.215	0.437	0.360	0.341
100	13.13	0.350	0.587	0.555	0.540
150	9.50	0.532	0.681	0.688	0.687
200	5.77	0.712	0.773	0.792	0.788

Note: Spars$_n$ = Sparsity level of the mode-n estimated factor.
NNC = Number of nonzero components.

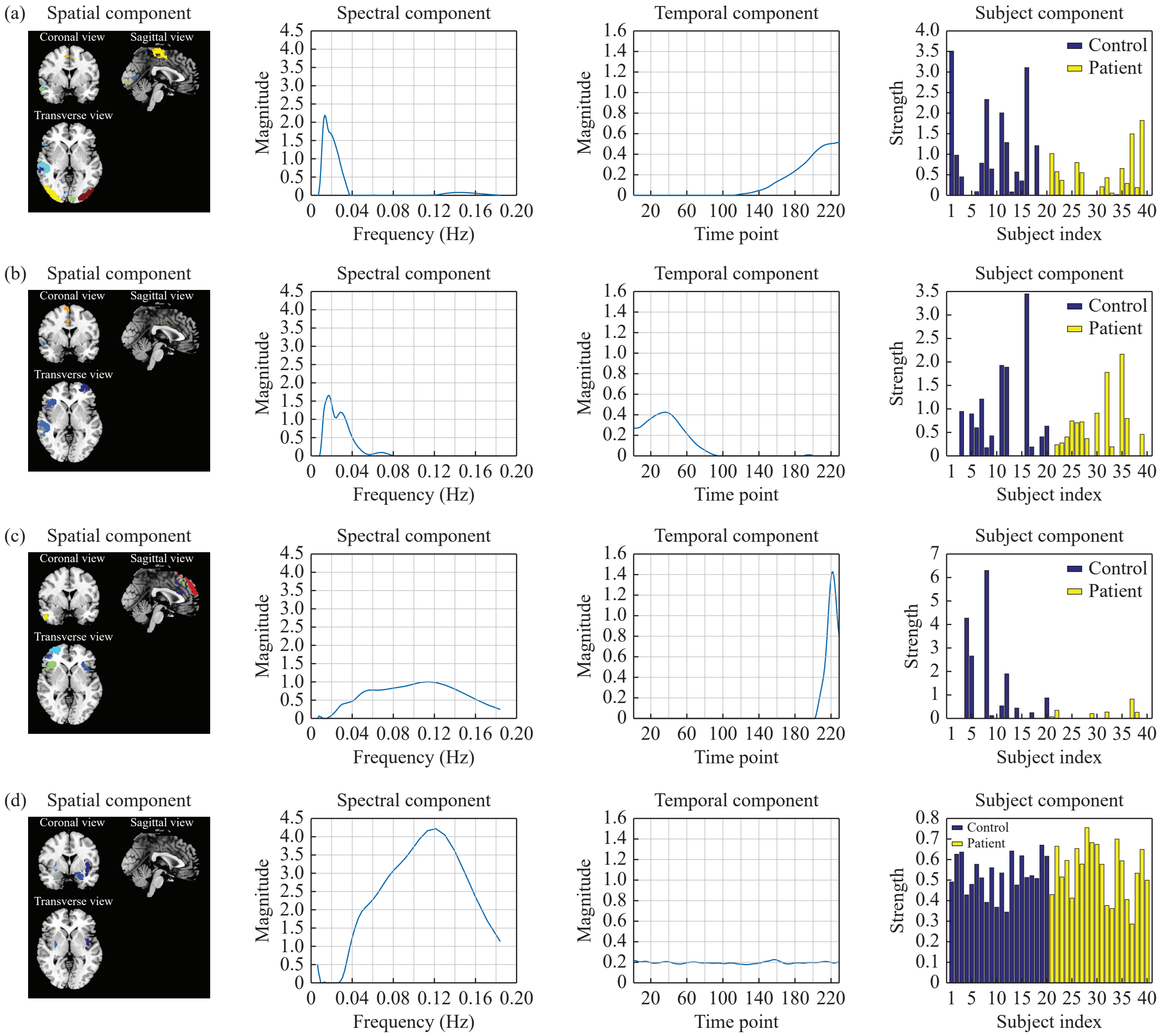

Figure 3 Selected groups of components from the fMRI tensor using the A^{2}DPMM algorithm.

[*3] The spatial components were plotted using the software REST [34]. REST can be downloaded from http://www.rfmri.org/REST.

science viewpoint.

V. Conclusion

In this study, we proposed a novel NCP algorithm using alternating optimization in the BCDframework and the alternating direction proximal method of multipliers, abbreviated to A^{2}DPMM. Our proposed NCP can overcome the rank deficiency problem in conventional ADMM-based methods. We evaluated the performances of the new algorithm on both synthetic and real-world tensors. The experimental results proved that our proposed method has better convergence properties and faster computation speed than the latest NCP methods. The proposed algorithm is suitable for processing large-scale dense and sparse tensor data. It can be a powerful tool for various areas, such as neuroscience, fluorescence and hyperspectral data processing. The A^{2}DPMM algorithm can naturally be extended to other tensor decomposition and fusion models in BCD framework, such as Tucker decomposition, PARAFAC2 and CMTF. Moreover, the proposed A^{2}DPMM algorithm can be further improved using the latest linearization techniques to cope with many other nonlinear and nonsmooth problems in tensor decomposition.

Acknowledgements

This work was supported by the State Key Laboratory of Robotics (Grant No. 2023-Z04) and the Natural Science Foundation of Liaoning Province (Grant No. 2022-BS-029). The authors would like to thank Dalian University of Technology for the support of the experimental environment.

References

[1] A. Cichocki, R. Zdunek, A. H. Phan, *et al.*, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. John Wiley & Sons, Hoboken, NJ, USA, 2009.

[2] A. Cichocki, D. Mandic, L. De Lathauwer, *et al.*, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[3] N. D. Sidiropoulos, L. D. Lathauwer, X. Fu, *et al.*, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[4] Y. P. Liu, *Tensors for Data Processing*. Elsevier, Amsterdam, The Netherlands, 2022.

[5] Y. Y. Xu and W. T. 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.

[6] Y. Zhang, G. X. Zhou, Q. B. Zhao, *et al.*, "Fast nonnegative tensor factorization based on accelerated proximal gradient and low-rank approximation," *Neurocomputing*, vol. 198, pp. 148–154, 2016.

[7] D. Q. Wang and F. Y. Cong, "An inexact alternating proximal gradient algorithm for nonnegative CP tensor decomposition," *Science China Technological Sciences*, vol. 64, no. 9, pp. 1893–1906, 2021.

[8] D. Q. Wang, Z. Chang, and F. Y. Cong, "Sparse nonnegative tensor decomposition using proximal algorithm and inexact block coordinate descent scheme," *Neural Computing and Applications*, vol. 33, no. 24, pp. 17369–17387, 2021.

[9] J. Kim and H. Park, "Fast nonnegative tensor factorization with an active-set-like method," in *High-Performance Scientific Computing*, M. W. Berry, K. A. Gallivan, E. Gallopoulos, *et al.*, Eds. Springer, London, UK, pp. 311–326, 2012.

[10] K. J. Huang, N. D. Sidiropoulos, and A. P. Liavas, "A flexible and efficient algorithmic framework for constrained matrix and tensor factorization," *IEEE Transactions on Signal Processing*, vol. 64, no. 19, pp. 5052–5065, 2016.

[11] S. Boyd, N. Parikh, E. Chu, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[12] Z. C. Lin, H. Li, and C. Fang, *Alternating Direction Method of Multipliers for Machine Learning*. Springer, Singapore, 2022.

[13] A. P. Liavas and N. D. Sidiropoulos, "Parallel algorithms for constrained tensor factorization via alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 63, no. 20, pp. 5450–5463, 2015.

[14] M. Roald, C. Schenker, V. D. Calhoun, *et al.*, "An AO-ADMM approach to constraining PARAFAC2 on all modes," *SIAM Journal on Mathematics of Data Science*, vol. 4, no. 3, pp. 1191–1222, 2022.

[15] C. Schenker, J. E. Cohen, and E. Acar, "A flexible optimization framework for regularized matrix-tensor factorizations with linear couplings," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 3, pp. 506–521, 2021.

[16] Q. M. Yao, Y. Q. Wang, B. Han, *et al.*, "Low-rank tensor learning with nonconvex overlapped nuclear norm regularization," *The Journal of Machine Learning Research*, vol. 23, no. 1, article no. 136, 2022.

[17] B. C. Pan, C. D. Li, and H. J. Che, "Nonconvex low-rank tensor approximation with graph and consistent regularizations for multi-view subspace learning," *Neural Networks*, vol. 161, pp. 638–658, 2023.

[18] Y. Wang, W. J. Zhang, L. Wu, *et al.*, "Iterative views agreement: An iterative low-rank based structured optimization method to multi-view spectral clustering," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, New York, NY, USA, pp. 2153–2159, 2016.

[19] Y. Wang, L. Wu, X. M. Lin, *et al.*, "Multiview spectral clustering via structured low-rank matrix factorization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4833–4843, 2018.

[20] R. Shefi and M. Teboulle, "Rate of convergence analysis of decomposition methods based on the proximal method of multipliers for convex minimization," *SIAM Journal on Optimization*, vol. 24, no. 1, pp. 269–297, 2014.

[21] N. Parikh, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[22] Z. C. Lin, R. S. Liu, and Z. X. Su, "Linearized alternating direction method with adaptive penalty for low-rank representation," in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, Granada, Spain, pp. 612–620, 2011.

[23] Y. Y. Ouyang, Y. M. Chen, G. H. Lan, *et al.*, "An accelerated linearized alternating direction method of multipliers," *SIAM Journal on Imaging Sciences*, vol. 8, no. 1, pp. 644–681, 2015.

[24] C. Y. Lu, J. S. Feng, S. C. Yan, *et al.*, "A unified alternating direction method of multipliers by majorization minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 527–541, 2018.

[25] E. K. Ryu and W. Yin, "ADMM-type methods," in *Large-*

Scale Convex Optimization: Algorithms & Analyses via Monotone Operators, E. K. Ryu and W. Yin, Eds. Cambridge University Press, Cambridge, UK, pp. 160–189, 2022.

[26] D. Q. Wang, "*Extracting meaningful EEG features using constrained tensor decomposition*," *Ph.D. Thesis*, University of Jyväskylä, Jyväskylä, Finland, 2019.

[27] D. P. Bertsekas, *Nonlinear Programming*, 3rd ed., Athena Scientific, Belmont, Massachusetts, 2016.

[28] A. Cichocki and A. H. Phan, "Fast local algorithms for large scale nonnegative matrix and tensor factorizations," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E92-A, no. 3, pp. 708–721, 2009.

[29] B. W. Bader and T. G. Kolda, "Tensor toolbox for MATLAB, version 3.2.1," Available at: *https://www.tensortoolbox.org/*, 2021-4-5.

[30] L. Badea, M. Onu, T. Wu, *et al.*, "Exploring the reproducibility of functional connectivity alterations in parkinson's disease," *PLoS One*, vol. 12, no. 11, article no. e0188196, 2017.

[31] B. T. T. Yeo, F. M. Krienen, J. Sepulcre, *et al.*, "The organization of the human cerebral cortex estimated by intrinsic functional connectivity," *Journal of Neurophysiology*, vol. 106, no. 3, pp. 1125–1165, 2011.

[32] E. Y. Choi, B. T. T. Yeo, and R. L. Buckner, "The organization of the human striatum estimated by intrinsic functional connectivity," *Journal of Neurophysiology*, vol. 108, no. 8, pp. 2242–2263, 2012.

[33] G. Q. Hu, D. Q. Wang, S. W. Luo, *et al.*, "Frequency specific co-activation pattern analysis via sparse nonnegative tensor decomposition," *Journal of Neuroscience Methods*, vol. 362, artilce no. 109299, 2021.

[34] X. W. Song, Z. Y. Dong, X. Y. Long, *et al.*, "REST: A toolkit for resting-state functional magnetic resonance imaging data processing," *PLoS One*, vol. 6, no. 9, artilce no. e25031, 2011.

Deqing WANG received the B.E. degree in automation and the M.E. degree in pattern recognition and intelligent system from Harbin Engineering University, Harbin, China, in 2009 and 2012, respectively. He received the Ph.D. degree from the University of Jyväskylä, Jyväskylä, Finland, in 2019. He was appointed as an Assistant Engineer and an Engineer at Dalian Scientific Test and Control Technology Institute, China Shipbuilding Industry Corporation (CSIC), Dalian, China, in 2012 and 2014, respectively. He is currently an Assistant Professor at Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China. His research interests include signal processing, machine learning, tensor decomposition, robotics and intelligent systems.
(Email: deqing.wang@foxmail.com)

Guoqiang HU received the B.E. degree and the Ph.D. degree in biomedical engineering from Dalian University of Technology, Dalian, China, in 2015 and 2022, respectively. He was a visiting doctoral student at Harvard Medical School, Boston, MA, USA, from 2018 to 2020. He is currently a Lecturer at the College of Artificial Intelligence, Dalian Maritime University, Dalian, China. His research interests include brain signal analysis and processing, independent component analysis, tensor decomposition and artificial intelligence.
(Email: guoqiang.hu@dlmu.edu.cn)