EFFICIENT CONSTRAINED TENSOR FACTORIZATION BY ALTERNATING OPTIMIZATION WITH PRIMAL-DUAL SPLITTING

Shunsuke Ono[†] and Takuma Kasai[††]

[†]Tokyo Institute of Technology   [††]RIKEN

ABSTRACT

Tensor factorization with hard and/or soft constraints has played an important role in signal processing and data analysis. However, existing algorithms for constrained tensor factorization have two drawbacks: (i) they require matrix-inversion; and (ii) they cannot (or at least is very difficult to) handle *structured* regularizations. We propose a new tensor factorization algorithm that circumvents these drawbacks. The proposed method is built upon alternating optimization, and each subproblem is solved by a primal-dual splitting algorithm, yielding an efficient and flexible algorithmic framework to constrained tensor factorization. The advantages of the proposed method over a state-of-the-art constrained tensor factorization algorithm, called AO-ADMM, are demonstrated on regularized nonnegative tensor factorization.

Index Terms— alternating optimization, constrained tensor factorization, nonconvex optimization, proximal splitting

1. INTRODUCTION

Tensor factorization techniques have been extensively studied and applied not only to signal processing and machine learning problems, including signal analysis and blind source separation [1], dimensionality reduction and learning latent variable models [2, 3], but also to scientific problems in chemometrics [4] and neuroscience [5]. Recent comprehensive reviews on tensor factorization can be found in [6, 7]

In the so-called *canonical polyadic decomposition* (CPD) model, also known as the *parallel factor analysis* (PARAFAC) model [8, 9], a tensor is decomposed into a sum of the lowest possible number of rank-1 tensors, where a rank-one tensor consists of an outer product of vectors. Although CPD is essentially unique under relatively mild conditions [10], hard and/or soft constraints on factors, such as nonnegativity, sparsity, smoothness and so on, are very useful for restoring identifiability, improving estimation accuracy, ensuring interpretability of the results, and fixing ill-posedness [7]. On the other hand, although there exist a bunch of tensor factorization algorithms, most of them are designed for unconstrained

cases or customized for handling a specific constraint, as pointed out in [11].

Very recently, a tensor factorization algorithm that can easily and naturally incorporate various types of constraints has been proposed [11]. This method is based on the *alternating optimization*, i.e., updating each variable (factor matrix) by solving the corresponding subproblem in a cyclic fashion, which is the standard approach adopted in many other tensor factorization algorithms. What is different is how to solve each subproblem: the method adopts the *alternating direction method of multipliers* (ADMM) [12–14]. ADMM can efficiently solve nonsmooth convex optimization problems with the help of proximal splitting techniques [15]. Thereby, this method, which the authors named AO-ADMM, deals with involved subproblems that have no closed-form solutions.

However, there are two things to be improved. One is that AO-ADMM requires matrix-inversion at each iteration of ADMM. The authors of [11] suggest to alleviate this computational difficulty by Cholesky decomposition and back-substitution, but it is still a bottleneck. The other is that a class of *structured* regularization, i.e., the composition of a simple regularization function and a linear operator, cannot be (or at least is very difficult to be) used as a soft constraint. Representative examples include the overlapping group lasso [16] and the total variation [17], which would be useful in many applications, as the lasso and the quadratic variation having been commonly used [18–20]. Note that these drawbacks are common to other constrained tensor factorization algorithms.

To circumvent these drawbacks, we propose a new algorithmic framework for constrained tensor factorization. The proposed method is built upon alternating optimization as well as AO-ADMM, but the essential difference is that each subproblem is solved by a *primal-dual splitting algorithm* [21, 22]. It can solve nonsmooth convex optimization problems involving linear operators without inversion and has been applied to signal and image processing problems, e.g., [23–28]. Thus, incorporating it into alternating optimization yields a more efficient and flexible algorithm for constrained tensor factorization than AO-ADMM. The advantages of the proposed method are demonstrated on regularized nonnegative tensor factorization, where aside from its efficiency, we empirically show that our method achieves better factorization in terms of mean squared error.

2. PRELIMINARIES

2.1. Canonical Polyadic Decomposition (CPD)

For brevity, we focus on third-order tensors in this paper but everything naturally generalizes to higher-order tensors. We denote a tensor by bold calligraphic letters like $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, a matrix by bold capital letters like $\mathbf{X} \in \mathbb{R}^{N_1 \times N_2}$, and a vector by bold letters like $\mathbf{x} \in \mathbb{R}^N$.

We denote the Khatri-Rao product (column-wise Kronecker product) of $\mathbf{X}$ and $\mathbf{Y}$ with the same number of columns (i.e.. $M_2 = N_2$) by $\mathbf{X} \odot \mathbf{Y} \in \mathbb{R}^{M_1 N_1 \times N_2}$, and the tensor product (outer product) of $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{y} \in \mathbb{R}^M$ by $\mathbf{x} \circledcirc \mathbf{y} \in \mathbb{R}^{N \times M}$. Note that the tensor product of three vectors yields a rank-1 third-order tensor. The mode-d matricization of $\mathcal{X}$ is a matrix of size $\prod N_{i \neq d} \times N_d$, denoted by $\mathbf{X}_{(d)}$. Each row of $\mathbf{X}_{(d)}$ is a vector obtained by fixing all the indices of $\mathcal{X}$ except the d-th one, and the matrix is formed by stacking these row vectors by traversing the rest of the indices from 3 back to 1.

CPD of $\mathcal{X}$ is given by

$$\mathcal{X} = \sum_{r=1}^{R} \circledcirc_{d=1}^{3} \mathbf{f}_d^{(r)}, \tag{1}$$

where $\mathbf{f}_d^{(r)} \in \mathbb{R}^{N_d}$ ($d = 1, 2, 3, r = 1, \ldots, R$) are the factors of $\mathcal{X}$, and $R > 0$ is the rank of $\mathcal{X}$ that is the minimum number of rank-1 tensors required to represent $\mathcal{X}$ as their sum. Note that a predetermined $0 < \tilde{R} \leq \min_d \{N_d\}$ is used in practice instead of the true tensor rank R since computing R in CPD is NP-hard [29]. We denote the above relation by

$$\mathcal{X} = [\mathbf{F}_d]_{d=1}^{3}, \tag{2}$$

where $\mathbf{F}_d := (\mathbf{f}_d^{(1)} \cdots \mathbf{f}_d^{(R)}) \in \mathbb{R}^{N_d \times R}$ is the factor matrix of the d-th mode. With the help of this notation, we can express CPD in a matricized form:

$$\mathbf{X}_{(d)} = (\circledcirc_{i \neq d} \mathbf{F}_i) \mathbf{F}_d^{\top}. \tag{3}$$

2.2. Primal-Dual Splitting Algorithm

Let $\Gamma_0(\mathbb{R}^N)$ be the set of all proper lower semicontinuous convex functions on $\mathbb{R}^N$. Consider convex optimization problems of the form:

$$\min_{\mathbf{x} \in \mathbb{R}^N} f(\mathbf{x}) + g(\mathbf{x}) + h(\mathcal{L}\mathbf{x}), \tag{4}$$

where $f, g \in \Gamma_0(\mathbb{R}^N)$ (f is β-Lipschitz differentiable with $\beta > 0$), $h \in \Gamma_0(\mathbb{R}^K)$, and $\mathcal{L} : \mathbb{R}^N \to \mathbb{R}^K$ is a linear operator. Also, let us introduce the notion of the *proximity operator* of index $\gamma > 0$ of $f \in \Gamma_0(\mathbb{R}^N)$ as follows:

$$\operatorname{prox}_{\gamma f} : \mathbb{R}^N \to \mathbb{R}^N : \mathbf{x} \mapsto \underset{\mathbf{y}}{\operatorname{argmin}} f(\mathbf{y}) + \frac{1}{2\gamma} \|\mathbf{y} - \mathbf{x}\|^2. \tag{5}$$

A primal-dual splitting (PDS) algorithm [21] solves Prob. (4) by the following iterative procedure: for any $\mathbf{x}^{(0)} \in \mathbb{R}^N$, $\mathbf{y}^{(0)} \in \mathbb{R}^K$, and $\gamma_1, \gamma_2 > 0$ satisfying $\gamma_1(\frac{\beta}{2} + \gamma_2 \|\mathcal{L}^* \mathcal{L}\|) < 1$ ($\mathcal{L}^*$ is the adjoint operator of $\mathcal{L}$, and $\| \cdot \|$ denotes the operator norm), iterate

$$\left| \begin{array}{l} \mathbf{x}^{(n+1)} := \operatorname{prox}_{\gamma_1 g}(\mathbf{x}^{(n)} - \gamma_1(\nabla f(\mathbf{x}^{(n)}) + \mathcal{L}^*(\mathbf{y}^{(n)}))), \\ \mathbf{y}^{(n+1)} := \operatorname{prox}_{\gamma_2 h^*}(\mathbf{y}^{(n)} + \gamma_2 \mathcal{L}(2\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)})), \end{array} \right. \tag{6}$$

where h^* is the convex conjugate function of h, and γ_1 and γ_2 can be seen as the stepsizes. We note that the proximity operator of h^* is available via that of h as

$$\operatorname{prox}_{\gamma h^*}(\mathbf{x}) = \mathbf{x} - \gamma \operatorname{prox}_{\gamma^{-1} h}(\gamma^{-1} \mathbf{x}) \tag{7}$$

(see, e.g., [30, Theorem 14.3(ii)]). Under some mild condition on g, h, and $\mathcal{L}$, the sequence $(\mathbf{x}^{(n)})_{k \in \mathbb{N}}$ converges to an optimal solution of Prob. (4).

3. PROPOSED METHOD

3.1. Problem Formulation

Consider the following data observation model:

$$\mathcal{Y} = \mathcal{M}(\mathcal{X} + \mathcal{E}), \tag{8}$$

where $\mathcal{Y}$ is an observed data stored as a tensor possibly with missing data, $\mathcal{X}$ is a true tensor data, $\mathcal{M}$ is a self-adjoint idempotent linear operator that specifies the observed entries in $\mathcal{Y}$, i.e., zeroing out the entries whose indices correspond to missing data, and $\mathcal{E}$ is an additive noise.

Then, wth the notation in (2), we formulate constrained tensor factorization as a generic optimization problem:

$$\min_{\mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3} \frac{1}{2} \|\mathcal{Y} - \mathcal{M}([\mathbf{F}_d]_{d=1}^3)\|_F^2 + \sum_{d=1}^{3} h_d(\mathcal{L}_d(\mathbf{F}_d^{\top}))$$

$$\text{s.t. } \mathbf{F}_d^{\top} \in C_d \ (d = 1, 2, 3), \tag{9}$$

where $\| \cdot \|_F$ is the Frobenius norm of a tensor, $h_d \circ \mathcal{L}_d$ is a regularization function (soft constraint) for the d-th factor matrix consisting of a (possibly nonsmooth) convex function h_d and a linear operator $\mathcal{L}_d$, and C_d is a closed convex set representing a hard constraint on the d-th factor matrix. Here we assume that the proximity operator of h_d and the metric projection[1] onto C_d are efficiently computable.

This formulation covers various existing constrained tensor factorization problems. A typical example would be non-negative tensor factorization [31, 32], which is recovered by setting $h_d := 0$ and $C_d := \mathbb{R}_+^{N_d \times R}$ ($\mathbb{R}_+$ denotes the set

---

[1]The *metric projection* onto a closed convex set C is given by

$$P_C : \mathbb{R}^N \to \mathbb{R}^N : \mathbf{x} \mapsto \underset{\mathbf{y} \in C}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{x}\|^2.$$

of all nonnegative real numbers). Another example is ℓ_1-regularized tensor factorization [18, 19] promoting the sparsity of factors, which corresponds to penalizing factors by the ℓ_1 norm, i.e., $h_d := \|\cdot\|_1$. These constraints have shown to be very useful for the reasons mentioned in Sec. 1.

3.2. Optimization

Since Prob. (9) is nonconvex due to the multi-linearity of CPD, it is very difficult to solve the problem directly. As a remedy, the alternating optimization is commonly used: each factor matrix $\mathbf{F}_d$ is updated in a cyclic fashion. Specifically, for each $\mathbf{F}_d$, we solve the following subproblem:

$$\min_{\mathbf{F}_d} \frac{1}{2}\|\mathbf{Y}_{(d)} - \mathcal{M}((\odot_{i\neq d}\widetilde{\mathbf{F}}_i)\mathbf{F}_d^\top)\|_F^2 + h_d(\mathcal{L}_d(\mathbf{F}_d^\top))$$
$$\text{s.t. } \mathbf{F}_d^\top \in C_d, \quad (10)$$

where we use the matricized form in (3), and $\widetilde{\mathbf{F}}_i$ $(i \neq d)$ are the fixed factor matrices except the mode d.

Now we can see that Prob (10) is convex but is still a tough problem because of the nonsmoothness. Thus, we propose to approximately solve the problem by few iterations of the primal-dual splittihg algorithm in (6). To this end, first, we introduce the indicator function of C_d, defined by $\iota_{C_d}(\mathbf{x}) := 0$, if $\mathbf{x} \in C$; $\iota_{C_d}(\mathbf{x}) := \infty$, otherwise. It should be noted that the proximity operator of the indicator function of C_d is equivalent to the metric projection onto C_d. Second, by letting $\mathbf{W} := \odot_{i\neq d}\widehat{\mathbf{F}}_i$ and $\mathbf{F} := \mathbf{F}_d^\top$, Prob (10) can be rewritten as

$$\min_{\mathbf{F}} \frac{1}{2}\|\mathbf{Y}_{(d)} - \mathcal{M}(\mathbf{W}\mathbf{F})\|_F^2 + \iota_{C_d}(\mathbf{F}) + h_d(\mathcal{L}_d(\mathbf{F})).$$

Let us define

$$f(\mathbf{F}) := \frac{1}{2}\|\mathbf{Y}_{(d)} - \mathcal{M}(\mathbf{W}\mathbf{F})\|_F^2,$$
$$g(\mathbf{F}) := \iota_{C_d}(\mathbf{F}),$$
$$h(\mathbf{G}) := h_d(\mathbf{G}) \text{ and } \mathcal{L} := \mathcal{L}_d.$$

Since the squared loss term is β-Lipschitz differentiable with

$$\beta = \|\mathbf{W}^\top \mathcal{M}^*(\mathcal{M}(\mathbf{W}))\| = \|\mathbf{W}^\top \mathcal{M}(\mathbf{W})\|$$
$$\leq \|\mathbf{W}^\top\|\|\mathcal{M}\|\|\mathbf{W}\| = \|\mathbf{W}^\top\mathbf{W}\| \ (\because \|\mathcal{M}\| = 1),$$

and the proximity operators of ι_{C_d} and h_d are available from the assumptions on C_d and h_d, we can derive an iterative algorithm for solving Prob. (10) based on the primal-dual splitting algorithm in (6) as follows: for any $\mathbf{F}^{(0)}$, $\mathbf{G}^{(0)}$, and $\gamma_1, \gamma_2 > 0$ satisfying

$$\gamma_1\left(\frac{\|\mathbf{W}^\top\mathbf{W}\|}{2} + \gamma_2\|\mathcal{L}_d^*\mathcal{L}_d\|\right) < 1, \quad (11)$$

set $\mathbf{A} := \mathbf{W}^\top \mathcal{M}(\mathbf{W})$ and $\mathbf{B} := \mathbf{W}^\top \mathbf{Y}_{(d)}$, and iterate

$$\begin{vmatrix} \mathbf{F}^{(n+1)} := P_{C_d}(\mathbf{F}^{(n)} - \gamma_1(\mathbf{A}\mathbf{F}^{(n)} - \mathbf{B} + \mathcal{L}_d^*(\mathbf{G}^{(n)}))), \\ \mathbf{G}^{(n)} \leftarrow \mathbf{G}^{(n)} + \gamma_2\mathcal{L}_d(2\mathbf{F}^{(n+1)} - \mathbf{F}^{(n)}), \\ \mathbf{G}^{(n+1)} := \mathbf{G}^{(n)} - \gamma_2 \text{prox}_{\frac{1}{\gamma_2}h_d}(\frac{\mathbf{G}^{(n)}}{\gamma_2}). \end{vmatrix}$$
$$(12)$$

Remark 1 (Comparison with AO-ADMM [11]). Clearly, there is no matrix inversion in (12). Specifically, whereas AO-ADMM requires the inversion of $\mathbf{A} + \mathbf{I}$ (see Algorithm 1 in [11]), our algorithm only needs to compute the multiplication of $\mathbf{A}$ and $\mathbf{F}^{(n)}$. The authors of [11] suggest to use Cholesky decomposition and back-substitution for efficiently computing this inversion, but we will see in the next section that our algorithm outperforms AO-ADMM in terms of CPU time. We also remark that at the update of $\mathbf{G}$, our algorithm just computes the proximity operator of h_d, i.e., the linear operator $\mathcal{L}_d$ is decoupled. This is another big difference from AO-ADMM: it requires to compute the proximity operator of $h_d \circ \mathcal{L}_d$, which does *not* have a closed-form solution in general even if that of h_d does. Indeed, such a situation arises in the following cases: the overlapping group lasso, i.e., $h_d := \|\cdot\|_1$ and $\mathcal{L}_d$ is an operator that replicates overlapping variables, and the total variation, i.e., $h_d := \|\cdot\|_1$ and $\mathcal{L}_d$ is a discrete difference operator.

It should be noted that AO-ADMM can be used not only for the squared loss function (the first term in (10)) but also other loss functions such as ℓ_1 norm and Kullback-Leibler divergence. Although we only describe the squared loss case in this paper, our approach can also handle other cases by letting $f(\mathbf{F}) := 0$, $h(\mathbf{G}_1, \mathbf{G}_2) := l(\mathbf{G}_1) + h_d(\mathbf{G}_2)$ and $\mathcal{L} := (\mathcal{M}\mathbf{W}, \mathcal{L}_d)$ in (4), where l is a loss function.

Finally, by incorporating (12) into alternating optimization, we obtain a new algorithm for solving the generic constrained tensor factorization problem formulated in (9). The whole algorithm is shown in Algorithm 1.

Algorithm 1: Proposed method for solving (9) (AO-PDS)

input : $\mathbf{F}_d^{(0)}, \mathbf{G}_d^{(0)}, \mathbf{Y}_{(d)}$ and $\|\mathcal{L}_d^*\mathcal{L}_d\|$ $(d = 1, 2, 3)$
1 **while** *A stopping criterion is not satisfied* **do**
2   **for** $d = 1$ *to* 3 **do**
3      $\mathbf{W}^{(k)} := \odot_{i\neq d}\mathbf{F}_i^{(k)}$;
4      $\mathbf{A}^{(k)} := \mathbf{W}^{(k)\top}\mathcal{M}(\mathbf{W}^{(k)})$;
5      $\mathbf{B}^{(k)} := \mathbf{W}^{(k)\top}\mathbf{Y}_{(d)}$;
6   Compute γ_1 and γ_2 by (13) and (14);
7   Update $\mathbf{F}_d^{(k)}$ and $\mathbf{G}_d^{(k)}$ using (12) initialized with the previous $\mathbf{F}_d^{(k)}$ and $\mathbf{G}_d^{(k)}$;
8      $\mathbf{F}_d^{(k+1)} \leftarrow \mathbf{F}_d^{(k)}$;
9      $\mathbf{G}_d^{(k+1)} \leftarrow \mathbf{G}_d^{(k)}$;
10    $k \leftarrow k + 1$;
output: $\mathbf{F}_d^{(k)}$ $(d = 1, 2, 3)$

Remark 2 (Iteration number and stepsizes of (12)). As in the case of AO-ADMM, each factor matrix $\mathbf{F}_d$ and dual variable $\mathbf{G}_d$ are updated in a cyclic fashion in Alg. 1, so that we can expect that after a number of iterations, these variables
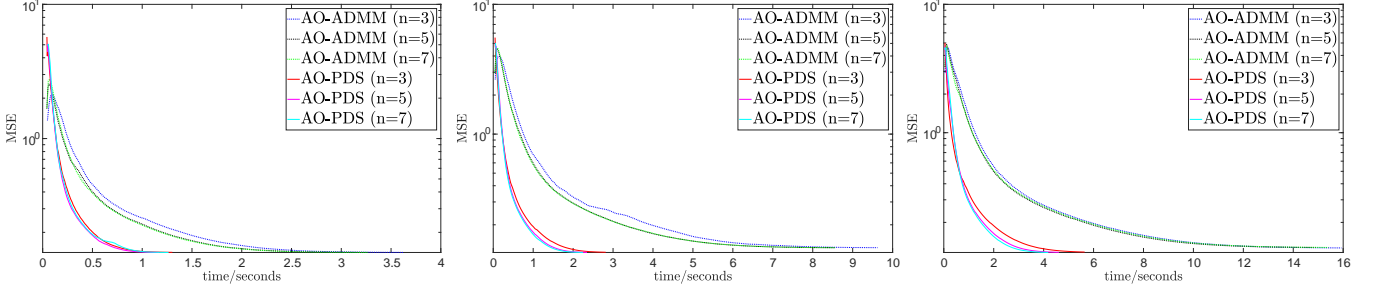
Fig. 1. Evolution of MSE (logarithmic scale) versus time in seconds on the regularized nonnegative tensor factorization: $R = 5$ (left), $R = 10$ (center) and $R = 15$ (right).

obtained in the previous iteration of (12) are good initial variables to the current iteration. This means that few iterations of (12) would be enough for updating $\mathbf{F}_d$ and $\mathbf{G}_d$, which keeps our method computationally efficient. Indeed, the numerical experiments in the next section show that a very few number of iterations (3 to 7) are sufficient for empirical convergence with reasonable precision.

In (12), the stepsizes $\gamma_1, \gamma_2 > 0$ are set to

$$\gamma_1 := 0.99 \frac{2}{\mathrm{trace}(\mathbf{W}^\top \mathbf{W})}, \quad (13)$$

$$\gamma_2 := \frac{1}{\gamma_1 \|\mathcal{L}_d^* \mathcal{L}_d\|} - \frac{\mathrm{trace}(\mathbf{W}^\top \mathbf{W})}{2\|\mathcal{L}_d^* \mathcal{L}_d\|}, \quad (14)$$

respectively. This setting satisfies the inequality in (11) since the trace of $\mathbf{W}^\top \mathbf{W}$ is an upper bound of $\|\mathbf{W}^\top \mathbf{W}\|$, and it can efficiently be computed at each iteration of Alg. 1. Note that $\|\mathcal{L}_d^* \mathcal{L}_d\|$ can be estimated in advance, and once determined, it can be used for every iteration because $\mathcal{L}_d$ does not change.

4. NUMERICAL EXPERIMENTS

We examined our method on regularized nonnegative tensor factorization. Our method was compared with AO-ADMM [11], where we used the MATLAB code distributed by the authors of [11]. All experiments were performed using MATLAB (R2017a), on a Windows 8.1 (64bit) laptop computer with an Intel Core i7 2.6 GHz processor and 16 GB of RAM.

We tested these algorithms on synthetic data. Specifically, synthetic true tensor data were generated as follows: for $N_1 = N_2 = N_3 = 100$ and $R = 5, 10$ or 15, the true factor matrices, denoted by $\mathbf{F}_d^{true}$ ($d = 1, 2, 3$), are obtained by drawing their elements from an i.i.d. uniform distribution on the interval $(0, 1)$, and then 80% of the elements of $\mathbf{F}_1^{true}$ are randomly set to 0, i.e., only $\mathbf{F}_1^{true}$ is sparse. The observed tensor data $\mathcal{Y}$ is then obtained by (8), where the elements of $\mathcal{E}$ are drawn from an i.i.d. Gaussian distribution with standard deviation 0.1. In the experiments, we did not consider missing data, i.e., $\mathcal{M}$ equals to an identity operator.

For soft constraints (regularization), we adopted the ℓ_1 norm for $\mathbf{F}_1$ and the squared Frobenius norm for $\mathbf{F}_2$ and $\mathbf{F}_3$, where their hyperparameters were set to 5 and 2, respectively. The nonnegativity constraint is also imposed

on each factor matrix. Then we measured the evolution of mean squared error (MSE) versus time in seconds on the regularized nonnegative tensor factorization problem solved by AO-ADMM and our algorithm, where MSE is defined by $\frac{1}{R(N_1+N_2+N_3)} \sum_{d=1}^{3} \|\mathbf{F}_d^{true} - \mathbf{F}_d\|_F^2$. Note that the above hyperparameters were hand-optimized in the MSE sense.

For the parameters of AO-ADMM, we used the settings recommended by the authors of [11]. The stopping criterion of the outer loop of each algorithm is set to $|\mathrm{MSE}_k - \mathrm{MSE}_{k-1}| < 1.0 \times 10^{-5}$ (k is the number of iterations of the outer loop). Note that this criterion can only be used for synthetic data since it uses MSE. For practical situations, one may use the value of the objective function in (9) as a stopping criterion.

The results are shown in Figure 1, where $n = 3, 5, 7$ are the number of the inner loop of each algorithm (ADMM or primal-dual splitting). One can see that our method (AO-PDS) is about *three times faster* than AO-ADMM. In addition, the best MSE values of AO-PDS are 0.142, 0.122 and 0.117; and those of AO-ADMM are 0.142, 0.133 and 0.127, respectively for $R = 5, 10, 15$, i.e., our method results in *more accurate* factorization in terms of MSE for $R = 10, 15$. As expected, we observe that very small number of outer iterations are enough for empirical convergence of AO-PDS. This would be thanks to the warm-start nature of alternating optimization, as in the case of AO-ADMM.

5. CONCLUSION

We have proposed an efficient and flexible tensor factorization algorithm based on alternating optimization combined with primal-dual splitting. Our algorithm has two advantages over AO-ADMM, a state-of-the-art tensor factorization algorithm, as follows: (i) it is free from matrix-inversion; and (ii) it can efficiently handle structured regularizations. Our experimental results on regularized nonnegative tensor factorization not only support our claim that the proposed method is more computationally efficient than AO-ADMM but also revealed that the proposed method achieves better factorization than AO-ADMM in the sense of MSE.

6. REFERENCES

[1] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Non-negative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*, John Wiley & Sons, 2009.

[2] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "Tag recommendations based on tensor dimensionality reduction," in *Proc. ACM Conf. Rec. Syst.*, 2008, pp. 43–50.

[3] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models.," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2773–2832, 2014.

[4] A. Smilde, R. Bro, and P. Geladi, *Multi-way analysis: applications in the chemical sciences*, John Wiley & Sons, 2005.

[5] M. Mørup, L. K. Hansen, C. S. Herrmann, J. Parnas, and S. M. Arnfred, "Parallel factor analysis as an exploratory tool for wavelet transformed event-related eeg," *NeuroImage*, vol. 29, no. 3, pp. 938–947, 2006.

[6] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[7] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, 2017.

[8] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[9] R. A. Harshman, "Foundations of the parafac procedure: Models and conditions for an explanatory multimodal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, pp. 1–84, 1970.

[10] N. D. Sidiropoulos and R. Bro, "On the uniqueness of multilinear decomposition of n-way arrays," *J. Chemometrics*, vol. 14, no. 3, pp. 229–239, 2000.

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

[12] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite elements approximations," *Comput. Math. Appl.*, vol. 2, pp. 17–40, 1976.

[13] J. Eckstein and D. P. Bertsekas, "On the Douglas-Rachford splitting method and proximal point algorithm for maximal monotone operators," *Math. Program.*, vol. 55, pp. 293–318, 1992.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "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.

[15] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke *et al*, Ed., pp. 185–212. Springer-Verlag, 2011.

[16] L. Jacob, G. Obozinski, and J.-P. Vert, "Group lasso with overlap and graph lasso," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2009.

[17] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, no. 1-4, pp. 259–268, 1992.

[18] G. Allen, "Sparse higher-order principal components analysis," in *Proc. Int. Conf. Art. Int. Stat. (AISTATS)*, 2012, pp. 27–36.

[19] Ji Liu, J. Liu, P. Wonka, and J. Ye, "Sparse non-negative tensor factorization using columnwise coordinate descent," *Patt. Recognit.*, vol. 45, no. 1, pp. 649–656, 2012.

[20] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth parafac decomposition for tensor completion," *IEEE Trans. Signal Process.*, vol. 64, no. 20, pp. 5423–5436, 2016.

[21] L. Condat, "A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *J. Opt. Theory Appl.*, vol. 158, no. 2, pp. 460–479, 2013.

[22] B. C. Vu, "A splitting algorithm for dual monotone inclusions involving cocoercive operators," *Adv. Comput. Math.*, vol. 38, pp. 667–681, 2013.

[23] L. Condat, "A generic proximal algorithm for convex optimization: Application to total variation minimization," *IEEE Signal Process. Lett.*, vol. 21, no. 8, pp. 985–989, 2014.

[24] S. Ono and I. Yamada, "Decorrelated vectorial total variation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2014, pp. 4090–4097.

[25] F. Andersson, M. Carlsson, J.-Y. Tourneret, and H. Wendt, "A new frequency estimation method for equally and unequally spaced data," *IEEE Trans. Signal Process.*, vol. 62, no. 21, pp. 5761–5774, 2014.

[26] P. L. Combettes, L. Condat, J. C. Pesquet, and B. C. Vu, "A forward-backward view of some primal-dual optimization methods in image recovery," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2014, pp. 4141–4145.

[27] S. Ono and I. Yamada, "Hierarchical convex optimization with primal-dual splitting," *IEEE Trans. Signal Process.*, vol. 63, no. 2, pp. 373–388, 2015.

[28] S. Ono, "Primal-dual plug-and-play image restoration," *IEEE Signal Process. Lett.*, vol. 24, no. 8, pp. 1108–1112, 2017.

[29] C. J. Hillar and L.-H. Lim, "Most tensor problems are np-hard," *Journal of the ACM*, vol. 60, no. 6, 2013.

[30] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, Springer, New York, 2011.

[31] R. Bro and S. De Jong, "A fast non-negativity-constrained least squares algorithm," *J. Chemometrics*, vol. 11, no. 5, pp. 393–401, 1997.

[32] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 792–799.