CO-CLUSTERING OF HIGH-ORDER DATA VIA REGULARIZED TUCKER DECOMPOSITIONS

Pedro A. Forero, Paul A. Baxley and Matthew Capella

SPAWAR Systems Center Pacific, San Diego, CA, USA

ABSTRACT

Computational methods for identifying hidden structures in high-order data are critical for exploratory data analysis tasks. This work proposes a joint dimensionality reduction and co-clustering algorithm for tensors. A compressed representation of a tensor is obtained via a Tucker-like decomposition model, whose factor matrices capture the tensor co-clustering structure. Factor matrices correspond to the cluster centroids of the tensor fibers per mode, whose entries interact nonlinearly to build the tensor approximation. The algorithm, developed based on the alternating-direction method of multipliers, has computational complexity similar to that of a single Tucker decomposition.

1. INTRODUCTION

Multi-way arrays, often called *tensors*, are becoming pervasive in several scientific, engineering, and social science applications [1]. Understanding the hidden structure that couples the data modes (dimensions) interacting within a tensor without relying on preconceptions about the data structure is a challenging task often faced in exploratory data analysis. *Clustering* is an unsupervised learning technique that seeks to partition a set of data into non-overlapping groups whose elements are 'close' to each other according to a predefined metric, thereby revealing the hidden structure of the data. Traditionally, clustering methods have focused on partitioning objects according to a *single* set of features. Unlike classical clustering, partitional *co-clustering* seeks to partition data into 'blocks' that are similar across, e.g., both row and column features. Co-Clustering techniques have found applications in machine learning, bioinformatics, text document mining, and social network analysis due to their ability to group data features across multiple data modes [2].

Co-clustering approaches for tensors have used ideas that extend classical clustering methods to high-order data. Graph structures derived from the tensor data have been used to develop spectral co-clustering algorithms based on the *spacey* random walks and random sampling [3, 4]. Classical clustering methods, such as K-means, have been applied per tensor mode, thereby yielding per mode clustering assignments whose Cartesian product yields the desired co-clustering [5]. A related line of research has used variations of classical tensor decompositions, such as the CANDECOP/PARAFAC (CP), higher-order singular value (HOSVD) and Tucker decompositions, as the basis for developing tensor co-clustering algorithms [6, 7, 8]. For instance, the rows of the factor matrices obtained via a regularized Tucker decomposition were used as a low-dimensional representation for tensor data in [9]. These low dimensional representations are clustered independently per mode and integrated as in [5] to obtain the tensor co-clustering.

This work was funded by the Naval Innovative Science and Engineering program at SPAWAR Systems Center Pacific.

Motivated by the link between low-dimensional tensor representations and clustering, this paper proposes a joint dimensionality reduction and co-clustering approach for high-order tensor data. It seeks to construct a Tucker decomposition for the tensor such that the columns of the factor matrices in the decomposition correspond to the cluster centers of tensor data for each of its modes. Not only are the clusters required to be 'good' representatives of the tensor data per mode, but they are also required to serve as a 'good' low dimensional representation for the data via the sum of their rank-1 interactions. Our framework enables structural constraints such as sparsity, non-negativity and orthogonality to be embedded in the selection of the core tensor and factor matrices. It also allows for various types of dissimilarity metrics to be used for capturing the per-mode cluster structure. The proposed co-clustering algorithm is developed based on the alternating direction method of multipliers (ADMoM). The resulting ADMoM iterations reduce to familiar block-coordinate descent (BCD) for solving a regularized Tucker decomposition and a single K-means update per tensor mode. The performance of the proposed algorithm is illustrated and compared with other methods via numerical tests on synthetic data.

2. TENSOR PRELIMINARIES

An order-N *tensor* is defined as a multidimensional array $\overline{\mathbf{Y}} \in \mathbb{R}^{D_1 \times \cdots \times D_N}$, where $D_n \in \mathbb{N}$ denotes the dimensionality of its n-th mode (dimension). Tensors are natural generalizations of vectors $\mathbf{y} \in \mathbb{R}^{D_1}$ and matrices $\mathbf{Y} \in \mathbb{R}^{D_1 \times D_2}$, which are order-1 and order-2 tensors, respectively. When working with tensors, it is often useful to consider subsets of their entries $[\overline{\mathbf{Y}}]_{d_{i_1},\ldots,d_{i_N}} := y_{d_{i_1},\ldots,d_{i_N}} \in \mathbb{R}$ and to reorganize their entries into a single vector or matrix. In particular a mode-n *fiber* of $\overline{\mathbf{Y}}$ is defined as a 1-dimensional subtensor comprising the entries of $\overline{\mathbf{Y}}$ obtained by fixing all but the n-th index, and a *slice* of $\overline{\mathbf{Y}}$ is defined as a 2-dimensional subtensor comprising the entries of $\overline{\mathbf{Y}}$ obtained by fixing all but two of its indices. The mode-n matricization of $\overline{\mathbf{Y}}$ arranges all mode-n fibers of $\overline{\mathbf{Y}}$ into the columns of $\mathbf{Y}_{(n)} \in \mathbb{R}^{D_n \times D_{-n}}$, with $D_{-n} := \prod_{n' \neq n} D_n$; see [10]. The mode-$n$ product between $\overline{\mathbf{Y}}$ and a matrix $\mathbf{U} \in \mathbb{R}^{Q \times D_n}$, denoted as $\overline{\mathbf{Y}} \times_n \mathbf{U}$, is defined as $\overline{\mathbf{Y}} \times_n \mathbf{U} = \mathbf{U}\mathbf{Y}_{(n)} \in \mathbb{R}^{Q \times D_{-n}}$, where $(\cdot)'$ denotes the transpose operator.

Similar to the singular value decomposition (SVD) for matrices, it is possible to develop tensor decomposition models that represent a tensor as a weighted sum of rank-one tensors. A rank-one tensor $\mathbf{u}_1 \circ \cdots \circ \mathbf{u}_N \in \mathbb{R}^{D_1 \times \cdots \times D_N}$ is defined as the outer product of vectors $\{\mathbf{u}_n\}_{n=1}^N$, whose $(i_1, \ldots, i_N)$ entry is given by $\prod_{n=1}^N u_{n,i_n}$, with $u_{n,i_n} := [\mathbf{u}_n]_{i_n}$. In particular the Tucker decomposition models $\overline{\mathbf{Y}}$ as

$$\overline{\mathbf{Y}} = \sum_{i_1=1}^{R_1} \cdots \sum_{i_N=1}^{R_N} g_{r_{i_1},\ldots,r_{i_N}} \left(\mathbf{u}_{1,i_1} \circ \cdots \circ \mathbf{u}_{N,i_N}\right) \qquad (1)$$

where the scalars $g_{r_{i_1},\ldots,r_{i_N}}$ denote the entries of the core tensor

$\overline{\mathbf{G}} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$, $R_n \leq D_n \; \forall n$, and $\mathbf{U}_n := [\mathbf{u}_{n,1} \ldots \mathbf{u}_{n,R_n}] \in \mathbb{R}^{D_n \times R_n}$, $n = 1, \ldots, N$, are the factor matrices [10, 11]. In contrast to the well-known CP decomposition in which $g_{r_{i_1},\ldots,r_{i_N}} \neq 0$ if and only if $i_1 = i_2 = \ldots = i_N$, the Tucker decomposition allows weighed interactions across all columns of the $\mathbf{U}_n$'s, with the entries of $\overline{\mathbf{G}}$ defining the weights of the interactions [10]. Note that (1) can be written compactly using the mode-n product notation as $\overline{\mathbf{Y}} = \overline{\mathbf{G}} \times \{\mathbf{U}\}$, where $\overline{\mathbf{G}} \times \{\mathbf{U}\} := \overline{\mathbf{G}} \times_1 \mathbf{U}_1 \times_2 \ldots \times_N \mathbf{U}_N$.

3. CO-CLUSTERING AND TUCKER DECOMPOSITION

Let $\mathbf{y}_{(n),i}$ denote the i-th column of $\mathbf{Y}_{(n)}$. Given an order-N tensor $\overline{\mathbf{Y}}$, we seek to jointly partition each set of mode-n fibers $\{\mathbf{y}_{(n),i}\}_{i=1}^{D_{-n}}$, into K_n non-overlapping sets while finding estimates for $(\overline{\mathbf{G}}, \mathbf{U}_1, \ldots, \mathbf{U}_N)$ to approximate $\overline{\mathbf{Y}}$ as in (1). Per mode n, the cluster centers $\{\mathbf{u}_{n,i}\}_{i=1}^{K_n}$ obtained for the mode-n fibers are used as the columns of the factor matrix $\mathbf{U}_n \in \mathbb{R}^{D_n \times K_n}$ in (1), with $R_n = K_n$. Let $\mathbf{S}_n \in \{0,1\}^{K_n \times D_{-n}}$, whose entries $s^n_{k,j} := [\mathbf{S}_n]_{k,j}$, $\forall k, j, n$, denote cluster membership coefficients. An $s^n_{k,j} = 1$ if the j-th mode-n fiber belongs to cluster k and $s^n_{k,j} = 0$ otherwise.

An estimate for $(\overline{\mathbf{G}}, \{\mathbf{U}_n, \mathbf{S}_n\}_{n=1}^N)$ is given by the solution of

$$\min_{\overline{\mathbf{G}}, \{\mathbf{U}_n, \mathbf{S}_n \in \mathcal{S}_n\}_{n=1}^N} \frac{1}{2} \left\| \overline{\mathbf{Y}} - \overline{\mathbf{G}} \times \{\mathbf{U}\} \right\|_F^2 + \frac{\zeta}{2} \left\| \overline{\mathbf{G}} \right\|_F^2 + \sum_{n=1}^N \frac{\zeta_n}{2} \left\| \mathbf{U}_n \right\|_F^2 + \sum_{n=1}^N \frac{\gamma_n}{2} \left\| \mathbf{Y}_{(n)} - \mathbf{U}_n \mathbf{S}_n \right\|_F^2 \tag{2}$$

where $\mathcal{S}_n := \{\mathbf{S} \in \{0,1\}^{K_n \times D_{-n}} : \mathbf{S}' \mathbf{1}_{K_n} = \mathbf{1}_{D_{-n}}\}$, $\mathbf{1}_m$ denotes an $m \times 1$ vector of ones, $\| \cdot \|_F$ the Frobenious norm, and $(\zeta, \{\zeta_n, \gamma_n\}_{n=1}^N)$ non-negative tuning parameters. The first term in the cost function of (2) seeks to minimize the approximation error for $\overline{\mathbf{Y}}$. Each summand, say the n-th one, in the last term of the cost in (2), together with its corresponding set of constraints, seeks to identify a partition of the mode-n fibers of $\overline{\mathbf{Y}}$. In particular, the constraint set $\mathcal{S}_n$ guarantees that each feasible $\mathbf{S}_n$ assigns every mode-n fiber to only one cluster. The Frobenious-norm regularizers on $\overline{\mathbf{G}}$ and $\{\mathbf{U}_n\}_{n=1}^N$ penalize large value assignments for the entries of $\overline{\mathbf{G}}$ and $\{\mathbf{U}_n\}_{n=1}^N$ and alleviate the scaling issue inherent to the term $\overline{\mathbf{G}} \times \{\mathbf{U}\}$. Note that additional constraints, such as sparsity and orthogonality, can be imposed on the $\mathbf{U}_n$'s in (2) [12].

Problem (2) is non-convex and highly nonlinear. It defines a form of tensor co-clustering in which each entry of $\overline{\mathbf{Y}}$ is assigned to a co-cluster indexed by the labels of the clusters to which each one of the N fibers (one per mode) are assigned. The dependency across co-cluster dimensions in (2) can be tuned via the γ_n's. In particular, when $\gamma_n \to \infty$, $\forall n$, (2) decomposes to N separate clustering problems, one for each set of mode-n fibers of $\overline{\mathbf{Y}}$.

In order to develop a practical solver for (2), it is useful to consider the following related problem

$$\begin{aligned} \min_{\substack{\overline{\mathbf{G}}, \{\mathbf{U}_n\}_{n=1}^N, \\ \{\mathbf{Z}_n, \mathbf{S}_n \in \mathcal{S}_n\}_{n=1}^N}} & \frac{1}{2} \left\| \overline{\mathbf{Y}} - \overline{\mathbf{G}} \times \{\mathbf{U}\} \right\|_F^2 + \frac{\zeta}{2} \left\| \overline{\mathbf{G}} \right\|_F^2 \\ & + \sum_{n=1}^N \frac{\zeta_n}{2} \left\| \mathbf{U}_n \right\|_F^2 + \sum_{n=1}^N \frac{\gamma_n}{2} \left\| \mathbf{Y}_{(n)} - \mathbf{Z}_n \mathbf{S}_n \right\|_F^2 \\ \text{Subj. to} \quad & \mathbf{Z}_n = \mathbf{U}_n, \quad n = 1, \ldots, N \end{aligned} \tag{3}$$

where the auxiliary variables $\mathbf{Z}_n \in \mathbb{R}^{D_n \times K_n}$ and corresponding equality constraints $\mathbf{Z}_n = \mathbf{U}_n$, $n = 1, \ldots, N$ have been introduced. These equality constraints guarantee that any feasible solution for (3) is also a feasible solution for (2). Furthermore, the cost in (3) can now be divided into two parts: (i) one that focuses on constructing a good approximation $\overline{\mathbf{G}} \times \{\mathbf{U}\}$ for $\overline{\mathbf{Y}}$ with appropriate regularizers for the core tensor and the factor matrices; and (ii) one that focuses on partitioning the mode-n fibers of $\overline{\mathbf{Y}}$ for each mode. In the following section, a computationally tractable solver for (2) is developed.

Remark 1 (Partial-Mode Clustering) *Problem* (2) *performs full-mode clustering of* $\overline{\mathbf{Y}}$. *It, however, can be readily adapted to applications where data partitioning for some modes of* $\overline{\mathbf{Y}}$ *is not required. In this case, which is termed partial-mode clustering, one can set* $\gamma_n = 0$, *if* $\overline{\mathbf{Y}}$ *is not to be clustered along mode* n *and remove the constraints on* $\mathbf{S}_n$. *In this case, the* $\mathbf{U}_n$*'s associated with modes not being clustered will be chosen so that they minimize the approximation error for* $\overline{\mathbf{Y}}$ *only.*

4. ADMOM TENSOR CO-CLUSTERING

In this section a numerical solver for (3) based on ADMoM is developed. First, consider the augmented Lagrangian for (3) given by

$$\begin{aligned} \mathcal{L}(\overline{\mathbf{G}}, \{\mathbf{U}_n, \mathbf{Z}_n, \mathbf{S}_n, \mathbf{\Lambda}_n\}_{n=1}^N) &= \frac{1}{2} \left\| \overline{\mathbf{Y}} - \overline{\mathbf{G}} \times \{\mathbf{U}\} \right\|_F^2 \\ &+ \frac{\zeta}{2} \left\| \overline{\mathbf{G}} \right\|_F^2 + \sum_{n=1}^N \frac{\zeta_n}{2} \left\| \mathbf{U}_n \right\|_F^2 + \sum_{n=1}^N \frac{\gamma_n}{2} \left\| \mathbf{Y}_{(n)} - \mathbf{Z}_n \mathbf{S}_n \right\|_F^2 \\ &+ \sum_{n=1}^N \mathrm{Tr}\left[\mathbf{\Lambda}_n' \left(\mathbf{Z}_n - \mathbf{U}_n \right) \right] + \sum_{n=1}^N \frac{\rho_n}{2} \left\| \mathbf{Z}_n - \mathbf{U}_n \right\|_F^2 \end{aligned} \tag{4}$$

where $\mathbf{\Lambda}_n \in \mathbb{R}^{D_n \times K_n}$ is the Lagrange multiplier matrix associated to the equality constraint $\mathbf{Z}_n = \mathbf{U}_n$, $\{\rho_n > 0\}_{n=1}^N$ are positive tuning parameters, and $\mathrm{Tr}(\cdot)$ denotes the trace operator.

With $\tau \in \mathbb{N}$ denoting an iteration index, the ADMoM updates for solving (3) are given in terms of the augmented Lagrangian as

$$\left(\overline{\mathbf{G}}^{[\tau+1]}, \{\mathbf{U}_n^{[\tau+1]}, \mathbf{S}_n^{[\tau+1]}\}_{n=1}^N \right) = \underset{\overline{\mathbf{G}}, \{\mathbf{U}_n, \mathbf{S}_n \in \mathcal{S}_n\}_{n=1}^N}{\arg\min} \mathcal{L}(\overline{\mathbf{G}}, \{\mathbf{U}_n, \mathbf{Z}_n^{[\tau]}, \mathbf{S}_n, \mathbf{\Lambda}_n^{[\tau]}\}_{n=1}^N) \tag{5a}$$

$$\{\mathbf{Z}_n^{[\tau+1]}\}_{n=1}^N = \underset{\{\mathbf{Z}_n\}_{n=1}^N}{\arg\min} \mathcal{L}(\overline{\mathbf{G}}^{[\tau+1]}, \{\mathbf{U}_n^{[\tau+1]}, \mathbf{Z}_n, \mathbf{S}_n^{[\tau+1]}, \mathbf{\Lambda}_n^{[\tau]}\}_{n=1}^N) \tag{5b}$$

$$\mathbf{\Lambda}_n^{[\tau+1]} = \mathbf{\Lambda}_n^{[\tau]} + \rho_n \left(\mathbf{Z}_n^{[\tau+1]} - \mathbf{U}_n^{[\tau+1]} \right), n = 1, \ldots, N \tag{5c}$$

where (5a) updates $\overline{\mathbf{G}}$, $\mathbf{U}_n$'s and $\mathbf{S}_n$'s with all other variables fixed, (5b) updates the $\mathbf{Z}_n$'s with all other variables fixed, and (5c) updates the $\mathbf{\Lambda}_n$'s with all other variables fixed.

Solving (5a) decomposes across $(\overline{\mathbf{G}}, \{\mathbf{U}_n\}_{n=1}^N)$ and $\{\mathbf{S}_n\}_{n=1}^N$. The updates $(\overline{\mathbf{G}}^{[\tau+1]}, \{\mathbf{U}_n^{[\tau+1]}\}_{n=1}^N)$ are given by the solution of

$$\min_{\overline{\mathbf{G}}, \{\mathbf{U}_n\}_{n=1}^N} \frac{1}{2} \left\| \overline{\mathbf{Y}} - \overline{\mathbf{G}} \times \{\mathbf{U}\} \right\|_F^2 + \frac{\zeta}{2} \left\| \overline{\mathbf{G}} \right\|_F^2 + \sum_{n=1}^N \frac{\zeta_n}{2} \left\| \mathbf{U}_n \right\|_F^2 - \sum_{n=1}^N \mathrm{Tr}\left[\mathbf{\Lambda}_n' \mathbf{U}_n \right] + \sum_{n=1}^N \frac{\rho_n}{2} \left\| \mathbf{Z}_n - \mathbf{U}_n \right\|_F^2 \tag{6}$$

which can be interpreted as a regularized Tucker decomposition problem, where the regularizer not only penalizes the size of the

entries of each $\mathbf{U}_n$, but it also penalizes the differences between $\mathbf{U}_n$'s and $\mathbf{Z}_n$'s. Before developing a solver for (6) it is useful to recall the following identities [10, 11]

$$\begin{aligned}\left\|\overline{\mathbf{Y}}-\overline{\mathbf{G}}\times\{\mathbf{U}\}\right\|_F &= \left\|\mathbf{Y}_{(n)}-\mathbf{U}_n\mathbf{G}_{(n)}(\boldsymbol{\Upsilon}_{N,n+1}\otimes\boldsymbol{\Upsilon}_{n-1,1})'\right\|_F \\ &= \left\|\mathbf{y}_{(n)}-[\boldsymbol{\Upsilon}_{N,n+1}\otimes\boldsymbol{\Upsilon}_{n-1,1}\otimes\mathbf{U}_n]\,\mathbf{g}_{(n)}\right\|_F\end{aligned} \tag{7a}$$

where $\boldsymbol{\Upsilon}_{n,m} := \bigotimes_{n'=n}^{m}\mathbf{U}_{n'}$, $\bigotimes_{n'=n}^{m}\mathbf{U}_{n'} := \mathbf{U}_n\otimes\ldots\otimes\mathbf{U}_m$, $\otimes$ denotes the Kronecker product, $\text{vec}(\cdot)$ the vectorization operator, $\mathbf{y}_{(n)} := \text{vec}(\mathbf{Y}_{(n)})$, and $\mathbf{g}_{(n)} := \text{vec}(\mathbf{G}_{(n)})$.

Problem (6) can be solved via a BCD algorithm, whereby each $\{\mathbf{U}_n\}_{n=1}^N$ and then $\overline{\mathbf{G}}$ are updated one at a time with all other variables fixed. With t denoting the iteration index for the BCD updates, each $\mathbf{U}_n$ can be updated in closed form as

$$\begin{aligned}\mathbf{U}_n^{[t+1,\tau]} = &\left[\mathbf{Y}_{(n)}\left(\boldsymbol{\Upsilon}_{N,n+1}^{[t+1,\tau]}\otimes\boldsymbol{\Upsilon}_{n-1,1}^{[t,\tau]}\right)\mathbf{G}_{(n)}^{[t,\tau]\prime}+\boldsymbol{\Psi}_n^{[\tau]}\right]\times \\ &\left[\mathbf{G}_{(n)}^{[t,\tau]}\left(\boldsymbol{\Xi}_{N,n+1}^{[t+1]}\otimes\boldsymbol{\Xi}_{n-1,1}^{[t]}\right)\mathbf{G}_{(n)}^{[t,\tau]\prime}+\theta_n\mathbf{I}_{K_n}\right]^{-1}\end{aligned} \tag{8}$$

where $\boldsymbol{\Upsilon}_{n,m}^{[t,\tau]} := \bigotimes_{n'=n}^{m}\mathbf{U}_{n'}^{[t,\tau]}$, $\boldsymbol{\Xi}_{n,m}^{[t,\tau]} := \bigotimes_{n'=n}^{m}\mathbf{U}_{n'}^{[t,\tau]\prime}\mathbf{U}_{n'}^{[t,\tau]}$, $\boldsymbol{\Psi}_n^{[\tau]} := \boldsymbol{\Lambda}_n^{[\tau]}+\rho_n\mathbf{Z}_n^{[\tau]}$, $\theta_n := \rho_n+\zeta_n$. The inverse matrix in (8) always exists since $\theta_n>0$, $\forall n$, and, thus, (8) is well defined for all factor-matrix updates $\mathbf{U}_n^{[t+1,\tau]}$.

Once all $\mathbf{U}_n$'s are updated, the entries of $\overline{\mathbf{G}}$ are updated as

$$\mathbf{g}_{(1)}^{[t+1,\tau]} = \left(\boldsymbol{\Xi}_{N,1}^{[t+1,\tau]}+\zeta\mathbf{I}_{\overline{K}}\right)^{-1}\boldsymbol{\Upsilon}_{N,1}^{[t+1,\tau]\prime}\mathbf{y}_{(1)} \tag{9}$$

where $\overline{K} := \prod_{n=1}^N K_n$. The core tensor $\overline{\mathbf{G}}^{[t+1,\tau]}$ is obtained by appropriately folding the entries of $\mathbf{g}_{(1)}^{[t+1,\tau]}$ [10]. Note that the inverse matrix in (9) exists for all iterations t, even when $\boldsymbol{\Xi}_{N,1}^{[t+1,\tau]}$ is rank deficient. After $t_{\max}$ iterations, the BCD updates summarized by (8) and (9) yield $\overline{\mathbf{G}}^{[\tau+1]} = \overline{\mathbf{G}}^{[t_{\max},\tau]}$ and $\{\mathbf{U}_n^{[\tau+1]} = \mathbf{U}_n^{[t_{\max},\tau]}\}_{n=1}^N$.

Solving for $\{\mathbf{S}_n\}_{n=1}^N$ decomposes across n. After using the constraint set $\mathcal{S}_n$ to rewrite the cost in (5a), computing (5a) with respect to $\mathbf{S}_n$ reduces to

$$\mathbf{S}_n^{[\tau+1]} = \underset{\mathbf{S}_n\in\mathcal{S}_n}{\arg\min}\,\frac{\gamma_n}{2}\sum_{i=1}^{D_{-n}}\sum_{k=1}^{K_n}s_{k,i}^n\left\|\mathbf{y}_{(n),i}-\mathbf{z}_{n,k}^{[\tau]}\right\|_2^2 \tag{10}$$

where $\mathbf{z}_{n,k}^{[\tau]}\in\mathbb{R}^{D_n}$ denotes the k-th column of $\mathbf{Z}_n^{[\tau]}$. Problem (12) can be solved in closed form for each entry of $\mathbf{S}_n^{[\tau+1]}$ as

$$s_{k,i}^{n,[\tau+1]} = \begin{cases}1 & \left\|\mathbf{y}_{(n),i}-\mathbf{z}_{n,k}^{[\tau]}\right\|_2 \le \left\|\mathbf{y}_{(n),i}-\mathbf{z}_{n,k'}^{[\tau]}\right\|_2\ \forall k'\neq k \\ 0 & \text{Otherwise}\end{cases}. \tag{11}$$

Solving for $\{\mathbf{Z}_n\}_{n=1}^N$ in (5b) decomposes across $\mathbf{Z}_n$. Thus, (5b) can be solved per $\mathbf{Z}_n$ via

$$\min_{\mathbf{Z}_n}\frac{\gamma_n}{2}\left\|\mathbf{Y}_{(n)}-\mathbf{Z}_n\mathbf{S}_n\right\|_F^2+\text{Tr}\left(\boldsymbol{\Lambda}_n'\mathbf{Z}_n\right)+\frac{\rho_n}{2}\left\|\mathbf{Z}_n-\mathbf{U}_n\right\|_F^2. \tag{12}$$

Problem (12) can be solved in closed form per column of $\mathbf{Z}_n$ as

$$\mathbf{z}_{n,k}^{[\tau+1]} = c_k^{n,[\tau+1]}\left(\sum_{i=1}^{D_n}s_{k,i}^{n,[\tau+1]}\mathbf{y}_{(n),i}-\frac{1}{\gamma_n}\mathbf{a}_{n,k}^{[\tau]}\right) \tag{13}$$

Algorithm 1: ADMoM Tensor Co-Clustering Algorithm

Data: $\overline{\mathbf{Y}}$, $\overline{\mathbf{G}}^{[0]}$, $\{\mathbf{U}_n^{[0]}\}_{n=2}^N$, $\{\boldsymbol{\Lambda}_n^{[0]}\}_{n=1}^N$, and the tuple $(\gamma,\zeta,\{\zeta_n,\rho_n,\gamma_n\}_{n=1}^N)$.

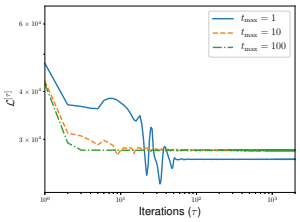
begin
  Construct $\{\mathbf{Y}_{(n)}\}_{n=1}^N$ from $\overline{\mathbf{Y}}$.
  for $\tau=1,\ldots,\tau_{\max}$ **do**
    Set $\overline{\mathbf{G}}^{[1,\tau]}=\overline{\mathbf{G}}^{[\tau+1]}$, $\{\mathbf{U}_n^{[1,\tau]}=\mathbf{U}_n^{[\tau+1]}\}_{n=1}^N$.
    for $t=1,\ldots,t_{\max}$ **do**
      for $n=1,\ldots,N$ **do**
        Construct $\mathbf{G}_{(n)}^{[t,\tau]}$ from $\overline{\mathbf{G}}^{[t,\tau]}$.
        Compute $\mathbf{U}_n^{[t+1,\tau]}$ via (8).
      Compute $\mathbf{g}_{(1)}^{[t+1,\tau]}$ via (9).
      Fold $\mathbf{g}_{(1)}^{[t+1,\tau]}$ into $\overline{\mathbf{G}}^{[t+1,\tau]}$.
    Set $\overline{\mathbf{G}}^{[\tau+1]}=\overline{\mathbf{G}}^{[t_{\max},\tau]}$, $\{\mathbf{U}_n^{[\tau+1]}=\mathbf{U}_n^{[t_{\max},\tau]}\}_{n=1}^N$.
    for $n=1,\ldots,N$ **do**
      Compute the entries of $\mathbf{S}_n^{[\tau+1]}$ via (11).
      Compute each column of $\mathbf{Z}_n^{[\tau+1]}$ via (13).
      Update $\boldsymbol{\Lambda}_n^{[\tau]}$ via (5c).

where $c_k^{n,[\tau+1]} := \left(\sum_{i=1}^{D_n}s_{k,i}^{n,[\tau+1]}+\frac{\rho_n}{\gamma_n}\right)^{-1}$, $\mathbf{a}_{n,k}^{[\tau]} := \boldsymbol{\lambda}_k^{[\tau]}-\rho_n\mathbf{u}_{n,k}^{[\tau+1]}$, $\boldsymbol{\lambda}_k^{[\tau]}$ denotes the k-th column of $\boldsymbol{\Lambda}_n^{[\tau]}$, and $\mathbf{u}_{n,k}^{[\tau+1]}$ the k-th column of $\mathbf{U}_n^{[\tau+1]}$. Note that $\forall n$, $c_k^{n,[\tau+1]}$ always exists since $\rho_n/\gamma_n>0$ and, thus, (13) is well defined.

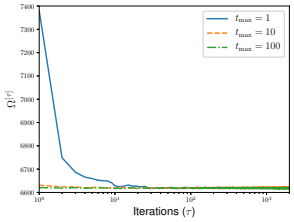
The ADMoM Tensor Co-clustering (ATCO) algorithm for solving (3) is summarized as Algorithm 1. Note that updates (10) and (12) can be carried out independently, in parallel, for each mode $n=1,\ldots,N$. Per iteration τ, small N and with $K_{\max} := \max\{K_1,\ldots,K_N\}$ and $D_{\max} := \max\{D_{-1},\ldots,D_{-N}\}$, the time computational complexity of the BCD updates is dominated by the BCD-iteration computational's complexity given by $O(t_{\max}(NK_{\max}^3+\overline{K}^3+2D_{\max}\overline{K}))$ where the first two terms in the sum correspond to the matrix inverse operations in (8) and (9), and the last term to the computation of the first matrix on the right-hand side of (8).

Remark 2 (On the convergence of the BCD iterations) *Let the cost in* (5a) *be denoted as* $f(\overline{\mathbf{G}},\mathbf{U}_1,\ldots,\mathbf{U}_N)$*, where each element in the argument of* f *corresponds to an optimization block component. Since* f *is differentiable and its partial derivatives per optimization block are continuous,* f *is also* continuously differentiable*. Given that* f *is strictly convex with respect to each optimization block when all other ones are fixed, its minimizers per optimization block, summarized by* (8) *and* (9)*, are unique. Per* τ*, it follows from [13, Prop. 3.7.1] that every limit point of* $\{(\overline{\mathbf{G}}^{[t,\tau]},\{\mathbf{U}_n^{[t,\tau]}\}_{n=1}^N)\}_{t\ge1}$ *is a stationary point for* f*.*

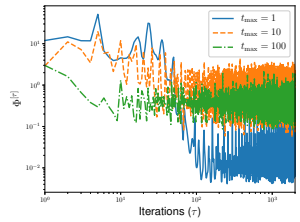
Remark 3 (On the convergence of the ADMoM iterations) *The convergence of a nonconvex ADMoM is an active area of research [14, 15, 16]. Although a full understanding of ADMoM's behavior in the general nonconvex case is still lacking, there are a few theoretical results that can help to analyze the behavior of Algorithm 1, e.g., see [17]. For instance, when* $t_{\max}=1$*, the results in [18] and [17] can be used to show that if the iterations in* (5) *converge, they converge to a Karush-Kuhn-Tucker point of* (3)*.*



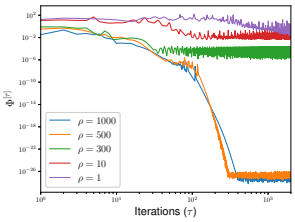
(a) Augmented Lagrangian ($\rho = 10$).



(b) LS approximation for $\overline{\mathbf{Y}}$ ($\rho = 10$).



(c) Equality-constraints violation ($\rho = 10$).



(d) Equality-constraints violation ($t_{max} = 1$).

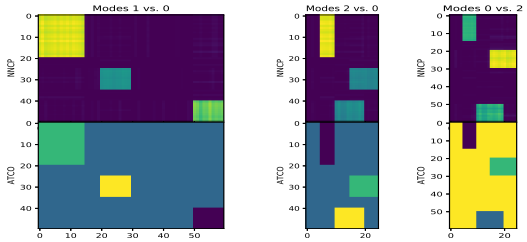
Fig. 1: Numerical evolution of ATCO with $\gamma = 1$.

5. NUMERICAL EXPERIMENTS

In this section the performance of the proposed co-clustering algorithm is illustrated on a synthetic 3rd-order tensor $\overline{\mathbf{Y}} \in \mathbb{R}^{50\times60\times25}$. Tensor $\overline{\mathbf{Y}}$ was constructed by inserting 3 non-overlapping sub-tensors (cubes) whose entries are uniformly distributed in the real-valued intervals $[3, 5]$, $[2, 3]$ and $[1, 4]$. Independent and identically distributed zero-mean Gaussian noise with variance $\sigma^2 = 0.4$ was added to each entry of the resulting tensor to obtain $\overline{\mathbf{Y}}$. ATCO was executed using $\zeta = 100$, $\zeta_1 = \zeta_2 = \zeta_3 = 0.01$, $K_1 = K_2 = K_3 = 4$ and $\rho_1 = \rho_2 = \rho_3 = \rho$, with the additional cluster per mode added to capture all fibers not belonging to any of the clusters induced by the artificial cubes. The HOSVD was used for initializing $\overline{\mathbf{G}}^{[0]}$, $\mathbf{U}_2^{[0]}$ and $\mathbf{U}_3^{[0]}$. All $\mathbf{\Lambda}_n^{[0]}$'s were set at random. The figures of merit used were $\Phi^{[\tau]} := \frac{\rho}{2}\sum_{n=1}^{3} ||\mathbf{Z}_n^{[\tau]} - \mathbf{U}_n^{[\tau]}||_F^2$, $\Omega^{[\tau]} := \frac{1}{2}\left\|\overline{\mathbf{Y}} - \overline{\mathbf{G}}^{[\tau]} \times \left\{\mathbf{U}^{[\tau]}\right\}\right\|_F^2$ and $\mathcal{L}^{[\tau]} := \mathcal{L}(\overline{\mathbf{G}}^{[\tau]}, \{\mathbf{U}_n^{[\tau]}, \mathbf{Z}_n^{[\tau]}, \mathbf{S}_n^{[\tau]}, \mathbf{\Lambda}_n^{[\tau]}\}_{n=1}^3)$, which illustrate the aggregate cost of equality-constraints violation, the quality of the least-squares (LS) approximation for $\overline{\mathbf{Y}}$, and the evolution of the augmented Lagrangian, respectively.

The effect of the number of BCD iterations, $t_{\max}$, executed for solving (5a) on the convergence of the ADMoM iterations was explored numerically. It was observed that $\mathcal{L}^{[\tau]}$ converges as $\tau \to \infty$, even for $t_{\max} = 1$, as illustrated in Fig. 1a. Note that the quality of the solution obtained in terms of co-clustering quality and tensor reconstruction quality changed as a function of $t_{\max}$, especially for small $t_{\max}$ values. The effect of $t_{\max}$ was more evident on $\Omega^{[\tau]}$, which showed that a small value of $t_{\max}$ translated to a larger τ for reaching a similar LS reconstruction error for $\overline{\mathbf{Y}}$ as shown in Fig. 1b. The parameter ρ controls how quickly the equality-constraints violations decrease as shown in Figs. 1c and 1d. Although ρ did not significantly impact how quickly $\mathcal{L}^{[\tau]}$ would converge, its choice affected the convergence value of $\mathcal{L}^{[\tau]}$ and $\Omega^{[\tau]}$. Larger values of ρ yielded larger values for both $\mathcal{L}^{[\tau]}$ and $\Omega^{[\tau]}$ as $\tau \to \infty$.

The co-clustering performance of ATCO, with $t_{\max} = 1$ and $\rho = 400$, was compared with that of a non-negative CP decomposition (NNCP) of rank 3 [6], an approximate co-clustering method (ACC) as suggested in [5], and the dynamic co-clustering (DCC) method proposed in [9]. All the methods considered herein, rely on some form of low-dimensional tensor decomposition that yields an approximation $\hat{\overline{\mathbf{Y}}}$ to $\overline{\mathbf{Y}}$. In ACC, the approximate partitioning of $\overline{\mathbf{Y}}$ per mode-n was obtained by partitioning the rows of the corresponding $\mathbf{Y}_{(n)}$ via the K-means algorithm with $K = 4$. In DCC, the factor matrices obtained via NNCP were clustered across their rows via K-means, with $K = 4$, to obtain the tensor partitioning per mode. For all methods, the co-clustering assignment was computed via the Cartesian product of clustering assignments per mode.



(a) (b) (c)

Fig. 2: Graphical illustration of the co-clustering per mode pairs obtained via the rank-3 approximation yielded by NNCP (top row), and via the fiber memberships yielded by ATCO (bottom row).

Table 1 shows a comparison of the reconstruction error and the Fowlkes-Mallows index (FMI) obtained for the co-clustering methods considered [19]. Since NNCP does not naturally yield co-clustering memberships for $\overline{\mathbf{Y}}$, no FMI score was computed for it. Instead Fig. 2 shows the clustering membership assignments obtained by ATCO and compares them with rank-3 matrix approximations constructed via NNCP for different mode-pairs via the CP decomposition factors. These matrices can be used to construct a co-clustering for $\overline{\mathbf{Y}}$ after using a data-dependent thresholding rule to identify how entries of $\overline{\mathbf{Y}}$ should be assigned to co-clusters. ATCO not only yields factor matrices that illustrate the rank-1 structure of $\overline{\mathbf{Y}}$, but it also yields assignment memberships per entry directly.

Metric	ATCO	NNCP	ACC	DCC
$\frac{1}{2}\|\|\overline{\mathbf{Y}} - \hat{\overline{\mathbf{Y}}}\|\|_F^2$	6675.19	6700.39	23,327.52	6700.39
FMI	0.5656	—	0.1705	0.2110

Table 1: Reconstruction-error and co-clustering quality comparison.

6. CONCLUSIONS

ATCO is a new tensor dimensionality reduction and co-clustering algorithm that combines a tensor-approximation and clustering criteria. The tensor approximation is obtained via a Tucker-like decomposition, which can naturally accommodate various type of structural constraints. ATCO's performance was illustrated and compared with other co-clustering methods via numerical experiments on synthetic data. A detailed analysis of the convergence of ATCO remains as an ongoing research direction. Nevertheless, it was empirically observed that ATCO converges even when only one BCD step across the block optimization variables is used. Further experimentation on real datasets is ongoing.

7. REFERENCES

[1] 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 Processing Magazine*, vol. 32, no. 2, pp. 145–163, March 2015.

[2] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha, "A generalized maximum entropy approach to bregman co-clustering and matrix approximation," *J. Mach. Learn. Res.*, vol. 8, pp. 1919–1986, Dec. 2007.

[3] T. Wu, A. R. Benson, and D. F. Gleich, "General tensor spectral co-clustering for higher-order data," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, USA, 2016, NIPS'16, pp. 2567–2575, Curran Associates Inc.

[4] D. Hatano, T. Fukunaga, T. Maehara, and K. Kawarabayashi, "Scalable algorithm for higher-order co-clustering via random sampling," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, 2017, pp. 1992–1999.

[5] S. Jegelka, S. Sra, and A. Banerjee, "Approximation algorithms for tensor clustering," in *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, Berlin, Heidelberg, 2009, ALT'09, pp. 368–383, Springer-Verlag.

[6] Q. Zhou, G. Xu, and Y. Zong, "Web co-clustering of usage network using tensor decomposition," in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, Sept 2009, vol. 3, pp. 311–314.

[7] E. E. Papalexakis, N. D. Sidiropoulos, and R. Bro, "From k-means to higher-way co-clustering: Multilinear decomposition with sparse latent factors," *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 493–506, Jan 2013.

[8] H. Zhao, D. D. Wang, L. Chen, X. Liu, and H Yan, "Identifying multi-dimensional co-clusters in tensors based on hyperplane detection in singular vector spaces," *PloS One*, vol. 11, no. 9, Sep 2016.

[9] W. W. Sun and L. Li, "Dynamic tensor clustering," *Journal of the American Statistical Association*, (To appear), 2018.

[10] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009.

[11] 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 Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, July 2017.

[12] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley Publishing, 2009.

[13] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 3rd edition, 2016.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[15] S. Magnússon, P. C. Weeraddana, M. G. Rabbat, and C. Fischione, "On the convergence of alternating direction Lagrangian methods for nonconvex structured optimization problems," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 3, pp. 296–309, Sept 2016.

[16] M. Hong, Z. Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[17] 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, Oct 2015.

[18] P. A. Forero, A. Cano, and G. B. Giannakis, "Distributed clustering using wireless sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 707–724, Aug 2011.

[19] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *Journal of the American Statistical Association*, vol. 78, no. 383, pp. 553–569, 1983.