HHS Public Access

Author manuscript

Comput Stat Data Anal. Author manuscript; available in PMC 2021 October 01.

Published in final edited form as:

Comput Stat Data Anal. 2020 October ; 150: . doi:10.1016/j.csda.2020.106989.

Generalized Co-Clustering Analysis via Regularized Alternating Least Squares

Gen Li[a,*]

[a]Department of Biostatistics, Columbia University. New York, NY 10032

Abstract

Biclustering is an important exploratory analysis tool that simultaneously clusters rows (e.g., samples) and columns (e.g., variables) of a data matrix. Checkerboard-like biclusters reveal intrinsic associations between rows and columns. However, most existing methods rely on Gaussian assumptions and only apply to matrix data. In practice, non-Gaussian and/or multi-way tensor data are frequently encountered. A new CO-clustering method via Regularized Alternating Least Squares (CORALS) is proposed, which generalizes biclustering to non-Gaussian data and multi-way tensor arrays. Non-Gaussian data are modeled with single-parameter exponential family distributions and co-clusters are identified in the natural parameter space via sparse CANDECOMP/PARAFAC tensor decomposition. A regularized alternating (iteratively reweighted) least squares algorithm is devised for model fitting and a deflation procedure is exploited to automatically determine the number of co-clusters. Comprehensive simulation studies and three real data examples demonstrate the efficacy of the proposed method. The data and code are publicly available[1].

Keywords

Exponential Family; Biclustering; Generalized Linear Model; Parafac/Candecomp; Tensor

1. Introduction

Biclustering is a powerful analytical tool for matrix data. It simultaneously clusters rows and columns of a matrix to identify checkerboard-like biclusters. Unlike standard clustering which focuses on grouping features only in one dimension, biclustering is flexible enough to identify important subgroup structures in both dimensions of a matrix. It has unique advantages in many applications. For example, in gene expression analysis, a subset of genes (variables) may be co-expressed in a subgroup of samples who share similar traits, but not in others. Such co-expression pattern may not be discovered by standard clustering algorithms because the structure only exists in a subgroup of samples. When leveraging all samples for

[1]The code is available at https://github.com/reagan0323/CORALS

[*]Corresponding author: gl2521@cumc.columbia.edu (Gen Li).

clustering, the signal may be overwhelmed by noise from irrelevant samples. Biclustering, however, has the potential to correctly identify such local structure by performing simultaneous clustering of genes and samples. Biclustering excels at identifying local patterns and has gained popularity in gene expression analysis [1], text mining [2] and beyond [3].

The idea of biclustering dates back to 1970s [4]. Since then, different biclustering algorithms have been proposed. There are mainly two schools of thought: matrix-based approaches [5, 6] and graph-based approaches [7, 8]. The former focuses on the decomposition of a matrix into blocks, while the latter converts biclustering to a graph partitioning problem. Other computational methods also exist, which typically exploit an iterative clustering procedure between rows and columns [9, 10, 11]. A comprehensive review of biclustering methods can be found in [1].

Most existing biclustering methods explicitly or implicitly assume observed data to be continuous with Gaussian errors. For example, [10] proposed a significance-based method where the significance score is derived from a Gaussian likelihood; [12] and [13] developed probabilistic additive models for biclustering under the Gaussian assumption; [11] proposed a sparse and symmetrized version of K-means clustering based on the minimization of a Gaussian likelihood. In addition, most matrix-based methods employ a least squares term in the objective function, closely pertinent to the Gaussian likelihood [14, 6, 5].

In practice, non-Gaussian matrix data are frequently encountered. For instance, in one of the motivating examples, music annotation data [15] are collected from a set of songs to study semantic annotation. Each song has a binary list of annotations. It is of interest to cluster songs and annotations simultaneously from the binary matrix to identify song groups that share similar features. Another example concerns a bag of words study on NIPS conference papers [16]. The data set contains counts of high-frequency words in thousands of NIPS papers published between 1987 and 2015. By investigating biclusters in this count-valued matrix, we aim to study how machine learning topics shifted in the past decades. Other examples such as co-morbidity studies (binary matrix of patientsxdiseases) and RNA-Seq studies (count matrix of samplesxgenes) may also involve biclustering of non-Gaussian data. Despite the prevalence of non-Gaussian matrix data, few biclustering methods are directly applicable.

In this paper, we develop a new method, called generalized Co-clustering via Regularized Alternating Least Squares (CORALS), to conduct simultaneous clustering on non-Gaussian data. We use the term “co-clustering” because the proposed method is general enough to handle multi-way tensor arrays which subsume matrix data (i.e., two-way tensors) as a special case. Hereafter, we will use the term “biclustering” if a method can only be applied to matrix data, and use “co-clustering” only if it is applicable to higher-order tensor data. We model non-Gaussian data with exponential family distributions which cover a range of continuous and discrete data types. The proposed method exploits a likelihood framework and identifies co-clusters in the natural parameter space. The likelihood function better accommodates the discreteness, boundedness and heteroscedasticity of non-Gaussian data than the commonly used squared loss function. The natural parameter of an exponential

family distribution is typically continuous and unbounded, and carries straightforward interpretations. We develop a regularized alternating (iteratively reweighted) least squares algorithm to estimate one co-cluster at a time, and exploit a deflation algorithm to automatically determine the number of co-clusters. The method enjoys great flexibility by allowing co-clusters to overlap and permitting the existence of background entries that do not belong to any co-clusters.

Biclustering for non-Gaussian matrix data has been scarcely studied in the literature. Among them, a class of probabilistic biclustering models termed latent block models has been developed [17, 18, 19]. The model-based methods handle binary, count, and categorical data via a likelihood approach. They connect to stochastic block models under the binary settings [20, 21, 22]. Flynn and Perry [23] proposed a profile likelihood-based approach which accommodates more general exponential family data modalities. However, these methods are exhaustive and require each matrix element to be assigned to one and only one bicluster. The exhaustive biclustering result is very restrictive in practice, as true biclusters may overlap and there may also exist non-belonging entries. More recently, Lee and Huang [24] developed a penalized Bernoulli likelihood approach for biclustering binary matrices, which allow overlapping, non-exhaustive results. However, the delicate Majorize-Minimization algorithm does not generalize to other non-Gaussian distributions. In addition, none of the aforementioned methods can be easily extended to multi-way tensor arrays.

Multi-way tensor arrays, as a generalization of matrices, are frequently encountered in practice as modern data become more heterogeneous and complex. For example, in neuroimaging analysis, multi-dimensional images across multiple subjects or conditions naturally form a higher-way data array; in genomics, gene expression data collected at different times or locations can also be represented as a tensor. A detailed tutorial on tensors can be found in [25]. Co-clustering analyses of multi-way tensors have grown in popularity in recent years. [26] proposed a hyperplane detection method coupled with higher-order singular value decomposition (i.e., Tucker decomposition). [27] converted a non-negative tensor to higher-order Markov chains and devised a spectral partitioning method. [28] and [29] developed low-rank decomposition methods with truncated loadings under Gaussian errors. [30] generalized a convex biclustering approach [14] to nulti-way tensors to achieve exhaustive co-clustering (i.e., each element is assigned to one co-cluster). However, the existing methods have various limitations such as 1) arbitrary numbers of co-clusters that may affect the stability of final results [26]; 2) rigid requirements of data format [27]; 3) ad hoc algorithms for model fitting [28, 29]; 4) unrealistic exhaustive and exclusive co-clustering assumptions [30]. Moreover, no method can adequately handle heterogeneous data types.

Our proposed method addresses the above limitations and provides a useful co-clustering tool for the exploratory analysis of non-Gaussian tensor data. The main contributions of the paper are as outlined below:

- We develop a very general framework for co-clustering analysis which is applicable to multi-way tensor arrays with observations from the exponential family.

- The method identifies realistic local structures by allowing co-clusters to overlap and not requiring every element of a tensor to belong to a co-cluster.
- The model fitting procedure is based on a principled penalized likelihood framework.
- The algorithm automatically selects the number of co-clusters in a tensor and can stop early with nested results.

The rest of the paper is organized as follows. In Section 2, we introduce the model framework of co-clustering for non-Gaussian tensor data. In Section 3, we devise the model fitting algorithm and an adaptive method to choose the number of co-clusters. Comprehensive simulation results are presented in Section 4 and three real application examples are included in Section 5. We discuss remaining questions and further research directions in Section 6. Technical details of the algorithm and additional simulation results can be found in the appendix.

2. Co-clustering Framework in Exponential Family

2.1. Notation

Throughout the paper, we use bold Euler script letters (e.g., $\mathcal{X}$) to represent multi-way (3 or more) tensor arrays, and bold capital and lowercase letters (e.g., $\boldsymbol{X}$ and $\boldsymbol{x}$) to represent matrices (2-way tensors) and vectors (1-way tensors), respectively. Let $\|\boldsymbol{X}\|_{\mathbb{F}}$ denote the Frobenius norm of the matrix $\boldsymbol{X}$, and $\|\boldsymbol{x}\|_1$ denote the ℓ_1 norm of the vector $\boldsymbol{x}$. Define $\mathcal{X}: p_1 \times \cdots \times p_K$ as a K-way tensor, with p_k being the dimension of the kth *mode*. Let $\mathcal{X}[i_1, \cdots, i_K]$ denote the $(i_1, \cdots, i_K)$th element of the tensor. The *matricization* operation transforms a multi-way tensor into a matrix, by unfolding the array along a particular mode. In particular, we denote $\boldsymbol{X}^{(k)}: p_k \times \prod_{j=1, j \neq k}^{K} p_j$ as the matricization of $\mathcal{X}$ along the kth mode. The *vectorization* operator, denoted by vec$(\cdot)$, stacks the columns of a matrix into a vector. A *unit-rank* K-way tensor $\mathcal{X}$ of dimensions $p_1 \times \cdots \times p_K$ can be represented by the *outer product* of unit-norm vectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_K$ of length $p_1, \cdots, p_K$ and a positive constant γ as $\mathcal{X} = \gamma \boldsymbol{v}_1 \circ \cdots \circ \boldsymbol{v}_K$, with entries $\mathcal{X}[i_1, \cdots, i_K] = \prod_{k=1}^{K} \gamma \boldsymbol{v}_k[i_k]$. Correspondingly, the CANDECOMP/PARAFAC (CP) decomposition factorizes a rank-R tensor $\mathcal{X}$ into a sum of R unit-rank tensors

$$\mathcal{X} = \sum_{r=1}^{R} \gamma_r v_{1r} \circ \cdots \circ \boldsymbol{v}_{Kr} := [\boldsymbol{\gamma}; V_1, \cdots, V_K],$$

where $\boldsymbol{v}_{kr}$ is the rth column of $\boldsymbol{V}_k$ with unit norm and $\boldsymbol{\gamma} = (\gamma_1, \cdots, \gamma_R$ is a vector with positive elements. More details about tensor notations and calculations can be found in [25].

2.2. Exponential Family Matrix Data Biclustering

We start from matrix data. Let $\boldsymbol{X}$ be an $n \times p$ matrix of observed data, which may take continuous or discrete values. We assume the data follow some distribution in the single-parameter exponential family. Different entries may not be identically distributed, but they

are from the same distribution. In particular, we assume the distribution is known or chosen beforehand based on domain knowledge. Denote the probability density function of an entry x_{ij} as

$$f(x_{ij} \mid \theta_{ij}) = h(x_{ij}) \exp\{x_{ij}\theta_{ij} - b(\theta_{ij})\},$$

where θ_{ij} is the corresponding natural parameter and $b(\cdot)$ and $h(\cdot)$ are distribution-specific functions. We collect all the natural parameters in $\boldsymbol{\Theta}$ which has the same size as $\boldsymbol{X}$. Although dispersion is not explicitly considered in the likelihood, having entry-wise natural parameters actually compensates the lack of dispersion parameters. A detailed discussion can be found in [31]. The singular value decomposition (SVD) of $\boldsymbol{\Theta}$ can be written as

$$\boldsymbol{\Theta} = \sum_{i=1}^{r} d_i \boldsymbol{u}_i \boldsymbol{v}_i^T, \tag{1}$$

where $r \leq \min(n, p)$ is the rank of $\boldsymbol{\Theta}$ and $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_r)$, $\boldsymbol{U} = (\boldsymbol{u}_1, \cdots, \boldsymbol{u}_r)$ and $\boldsymbol{V} = (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_r)$ are the matrices of positive non-increasing singular values, orthonormal left and right singular vectors, respectively. Without loss of generality, we let $\boldsymbol{U}$ absorb the singular values in $\boldsymbol{D}$.

We further assume $\boldsymbol{U}$ and $\boldsymbol{V}$ are sparse. The sparsity can naturally induce biclustering patterns in the natural parameter space. To see this, let us assume both $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are sparse in the ith layer. Correspondingly, the non-zero entries in $\boldsymbol{u}_i \boldsymbol{v}_i^T$ form a submatrix, which can be viewed as a bicluster. The bicluster is a local pattern that differs from the rest of the entries. Different layers may identify different (potentially overlapping) biclusters. In particular, if only one of $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ is sparse or neither is sparse, the layer is deemed to capture a global pattern. The sum of different layers will give rise to global patterns as well as checkerboard-like biclustering patterns.

To achieve sparse estimation of $\boldsymbol{U}$ and $\boldsymbol{V}$, we propose to solve the following penalized likelihood optimization problem

$$\begin{aligned} &\min_{\boldsymbol{U},\boldsymbol{V}} \; -\ell(\boldsymbol{X} \mid \boldsymbol{\Theta}) + P_u(\boldsymbol{U}) + P_v(\boldsymbol{V}) \\ &s.t. \;\; \boldsymbol{\Theta} = \boldsymbol{U}\boldsymbol{V}^T \end{aligned} \tag{2}$$

where $\ell(\boldsymbol{X} \mid \boldsymbol{\Theta}) \propto \sum_{i=1}^{n} \sum_{j=1}^{p} \{x_{ij}\theta_{ij} - b(\theta_{ij})\}$ is the log likelihood function of $\boldsymbol{X}$, and $P_u(\cdot)$ and $P_v(\cdot)$ are column-wise sparsity-inducing penalty functions for $\boldsymbol{U}$ and $\boldsymbol{V}$. Possible options include the ℓ_1 or LASSO penalty [32], smoothly clipped absolute deviation (SCAD) penalty [33], minimax concave penalty (MCP) [34], among others. The proposed framework associates with many existing methods in the literature. For example, a similar approach called sparse SVD (SSVD) has been developed for continuous data [6]. The method minimizes a penalized least squares function to achieve biclustering of continuous data matrix. Our approach essentially generalizes the squared loss to a negative log likelihood loss to accommodate more general data types. The proposed method also connects to the exponential family principal component analysis (EPCA) model in [31], without the penalty

terms. With structural penalties, it also coincides with the exponential family functional principal component analysis in [35].

The biclusters identified in this way have several properties. First, the biclusters are in the natural parameter space and thus offer straightforward interpretations. A natural parameter is a monotone transformation of the expectation of a random variable. For example, for binary data, the natural parameter is the log odds of success. A bicluster in the natural parameter space captures a subset of samples and variables that manifest higher or lower than average log odds. Second, different biclusters may overlap. Each layer only determines one bicluster and the non-zero entries in one layer may overlap with those in another layer. The overlapping feature is desired because in practice samples and variables may appear in multiple biclusters (e.g., a gene may belong to multiple pathways). Third, the biclustering is non-exhaustive. Namely, there may be samples and variables that are not contained in any bicluster. Since biclusters only capture the most coherent samples and variables, it is reasonable to believe that many are in the background. The proposed method is flexible enough to allow the existence of non-belonging entries which may be captured by the global structure. Finally, with proper treatment (which we shall introduce in Section 3), the leading biclusters are not subject to the total number of biclusters. For many computational methods, biclustering results highly depend on the preset number of biclusters. Adding or subtracting one bicluster may drastically change the structure of all biclusters. Our method, thanks to the nested feature of SVD, is insensitive to the misspecification of the number of biclusters and thus enjoys more stability.

2.3. Extension to Tensor Data

The proposed biclustering framework is readily generalizable to multi-way tensor arrays. For simplicity, we focus on 3-way tensors hereafter. All derivations can be trivially extended to higher-way tensors. Let $\mathcal{X}$ denote a $p_1 \times p_2 \times p_3$ tensor with continuous or discrete values from the exponential family. Let $\mathcal{T}$ denote the underlying natural parameter tensor with the same dimensions. Similar to (1), we assume $\mathcal{T}$ has a low-rank CP decomposition as

$$\mathcal{T} = \sum_{i=1}^{r} d_i \boldsymbol{v}_{1i} \circ \boldsymbol{v}_{2i} \circ \boldsymbol{v}_{3i},$$

where $\boldsymbol{V}_k = (\boldsymbol{v}_{k1}, \cdots, \boldsymbol{v}_{kr})$ contains sparse loading vectors ($k = 1,2,3$). For identifiability, we require each loading vector to have unit norm. Unlike SVD for matrix data, the strict orthogonality between different loading vectors is not required. Instead, a much weaker condition based on the concept of k-rank is sufficient to guarantee the identifiability of the CP decomposition [25]. To identify local structure, if all $\boldsymbol{v}_{1i}$, $\boldsymbol{v}_{2i}$ and $\boldsymbol{v}_{3i}$ are sparse in the ith layer, the non-zero entries in $\boldsymbol{v}_{1i} \circ \boldsymbol{v}_{2i} \circ \boldsymbol{v}_{3i}$ form a sub-tensor, or what we call, a co-cluster. The co-clusters identified by different layers may overlap because the loadings may have overlapping non-sparsity patterns. To induce sparsity, we optimize the following objective function

$$-\ell(\mathcal{X} \mid \mathcal{T}) + P_{v_1}(\boldsymbol{V}_1) + P_{v_2}(\boldsymbol{V}_2) + P_{v_3}(\boldsymbol{V}_3), \tag{3}$$

where $\boldsymbol{P}_{v_k}(\cdot)$s are column-wise sparsity-inducing penalty functions. Again, the negative log likelihood in the objective function is a generalization of the squared loss in standard CP decomposition [36, 37].

The previous biclustering framework (2), other than its additional orthogonality constraints on the loadings, can be viewed as a special case of this co-clustering framework. In fact, in order to obtain sparsity, people rarely enforce strict orthogonality of loadings in matrix decomposition [38, 39, 6]. We hereby follow the convention and relax the orthogonality constraints in biclustering. As a result, (2) exactly becomes a special case of the above co-clustering framework. Without loss of generality, hereafter we just focus on the general co-clustering framework for presentation.

Remark: Sometimes it may be desired to separate positive entries from negative entries in a non-zero sub-tensor and treat them as separate co-clusters [28]. As a result, a non-zero sub-tensor identified in each layer may contain more than one co-cluster. For example, in binary scenarios, large positive values correspond to high odds (success rates greater than 0.5), while large negative values correspond to low odds (success rates lower than 0.5). If one cares more about co-clusters with high odds (i.e., more 1s), one should primarily focus on positive-value sub-sub-tensors within a non-zero sub-tensor. The signs of the values within a sub-tensor are determined by the signs in each loading vector. The total number of same-sign co-clusters within a non-zero sub-tensor is up to 2^K for a K-way tensor. These co-clusters are mutually exclusive. In other words, we may identify ur to 2^K non-overlapping co-clusters within each layer of Model (3).

3. Estimation Algorithm

In this section, we first introduce a regularized alternating least squares (ALS) algorithm for solving (3) of each layer, and then devise a deflation algorithm to sequentially estimate different layers. The deflation algorithm automatically determines the total number of co-clusters and produces nested results.

3.1. Regularized Alternating Least Squares

We focus on the estimation of a unit rank model in this subsection. To induce sparsity in the loading estimates in (3), we particularly focus on the LASSO penalty term [32]. We propose a regularized ALS algorithm which alternates the estimation of $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$. Each step can be formulated as a regularized generalized linear model (GLM) estimation problem. We remark that other sparsity-inducing penalties [33, 34] can also be used if desired and the proposed algorithm largely remains the same.

More specifically, we aim to solve the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{v}_1,\boldsymbol{v}_2,\boldsymbol{v}_3} \; & -\sum_{i_1=1}^{p_1}\sum_{i_2=1}^{p_2}\sum_{i_3=1}^{p_3}\{\mathcal{X}[i_1,i_2,i_3]\mathcal{T}[i_1,i_2,i_3]-b(\mathcal{T}[i_1,i_2,i_3])\} \\ & +\lambda_1\|\boldsymbol{v}_1\|_1+\lambda_2\|\boldsymbol{v}_2\|_1+\lambda_3\|\boldsymbol{v}_3\| \\ s.t. \quad & \mathcal{T}=d\boldsymbol{v}_1\circ\boldsymbol{v}_2\circ\boldsymbol{v}_3 \end{aligned}$$

where λ_1, λ_2, λ_3 are tuning parameters and b(·) is a distribution-specific convex function. For simplicity, we let $\boldsymbol{v}_1$ absorb the constant d. The optimization problem does not have closed-form solution, but the objective function is convex with respect to each loading vector while holding the other two fixed. Thus, it is natural to use an ALS algorithm to solve the problem. Below we discuss the estimation of $\boldsymbol{v}_1$ while holding $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ constant. By symmetry, the estimation of $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ can be obtained similarly.

With fixed $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$, we first matricize $\mathcal{X}$ and $\mathcal{T}$ along the first mode as $\boldsymbol{X}^{(1)}$ and $\boldsymbol{T}^{(1)}$. In particular, given the unit-rank decomposition, we have $\boldsymbol{T}^{(1)} = \boldsymbol{v}_1(\boldsymbol{v}_3 \otimes \boldsymbol{v}_2)^T$. We further vectorize $\boldsymbol{T}^{(1)^T}$ as

$$\mathrm{vec}\left(\boldsymbol{T}^{(1)^T}\right) = \{\boldsymbol{I} \otimes (\boldsymbol{v}_3 \otimes \boldsymbol{v}_2)\}\boldsymbol{v}_1,$$

where $\boldsymbol{I}$ is an $p_1 \times p_1$ identity matrix. Consequently, the original optimization problem has an equivalent form as

$$\min_{\boldsymbol{v}}\left[-\mathrm{vec}\left(\boldsymbol{X}^{(1)^T}\right)^T\{\boldsymbol{I} \otimes (\boldsymbol{v}_3 \otimes \boldsymbol{v}_2)\}\boldsymbol{v} + \mathbf{1}^T b(\{\boldsymbol{I} \otimes (\boldsymbol{v}_3 \otimes \boldsymbol{v}_2)\}\boldsymbol{v})\right] + \lambda_1\|\boldsymbol{v}\|_1, \quad (4)$$

where $\mathbf{1}$ is a length-$(p_1p_2p_3)$ vector of all 1s and $b(\cdot)$ represents an entrywise function with a little abuse of notation. It immediately follows that the first part of the above objective function resembles a GLM problem with $\mathrm{vec}(\boldsymbol{X}^{(1)^T})$ being the response, $\boldsymbol{I}\otimes(\boldsymbol{v}_3\otimes\boldsymbol{v}_2)$ being the (orthogonal) design matrix and $\boldsymbol{v}$ being the coefficient vector to be estimated. With the additional LASSO penalty, the optimization problem becomes a regularized GLM problem, which can be solved via an iteratively reweighted penalized least squares algorithm. A detailed description can be found in the appendix.

Once estimated, we fix $\boldsymbol{v}_1$ and continue to estimate $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ in a similar fashion. We alternate the estimation of different loadings until convergence. With fixed tuning parameters, the algorithm always converges because the objective function decreases in each step. However, since the optimization is not convex, it is not guaranteed to converge to the global optimum. The choice of initial values and the order in which the loadings are estimated may affect the final result. In practice, one may run the algorithm from multiple starting points and in different orders, and compare the likelihood values corresponding to different estimates.

3.2. Deflation Algorithm

To identify multiple non-zero sub-tensors from the observed array, we exploit a deflation algorithm. It is a sequential procedure where we estimate one unit-rank structure at a time given what has already been estimated. In the Gaussian scenario, this can be simplified as applying the unit-rank estimation algorithm to a residual tensor, obtained by subtracting previously estimated sub-tensors from the observed data [6, 38]. However, for non-Gaussian data, the residual tensor is not well defined. One cannot directly subtract a sub-tensor in the natural parameter space from the observed data. Instead, we treat the previously estimated sub-tensors as an offset in the subsequent estimation.

More specifically, suppose $\widetilde{\boldsymbol{V}}_1, \widetilde{\boldsymbol{V}}_2, \widetilde{\boldsymbol{V}}_3$ have been estimated in previous steps. Let $\widetilde{\mathcal{T}} = [\widetilde{\boldsymbol{V}}_1, \widetilde{\boldsymbol{V}}_2, \widetilde{\boldsymbol{V}}_3]$. We have $\mathcal{T} = \widetilde{\mathcal{T}} + \boldsymbol{v}_1 \circ \boldsymbol{v}_2 \circ \boldsymbol{v}_3$, where $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$ are the loading vectors to be estimated. In order to estimate $\boldsymbol{v}_1$, we note that

$$\mathrm{vec}\left(\boldsymbol{T}^{(1)^T}\right) = \mathrm{vec}\left(\widetilde{\boldsymbol{T}^{(1)}}^T\right) + \{\boldsymbol{I} \otimes (\boldsymbol{v}_3 \otimes \boldsymbol{v}_2)\}\boldsymbol{v}_1 .$$

By substituting the above quantity in the optimization problem (4), we obtain a regularized GLM with offset being $\mathrm{vec}\left(\widetilde{\boldsymbol{T}^{(1)}}^T\right)$. Consequently, we can use a similar iteratively reweighted penalized least squares algorithm as before to estimate $\boldsymbol{v}_1$ (similar for $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$). As a result, we can sequentially identify non-zero sub-tensors from the observed data array. An advantage of the deflation algorithm is that the identified structure is nested with respect to different ranks. In other words, regardless of the preset rank, the first few sub-tensors always remain the same.

Although not guaranteed, the order of identified unit-rank structures is usually in agreement with the order of the norm of the loading vectors in $\widetilde{\boldsymbol{V}}_1$. The latter can be viewed as the level of importance of the identified co-clusters. In practice, one could always reorder co-clusters post hoc if desired.

3.3. Tuning Parameter Selection

There are a few tuning parameters to be selected, including the rank r of the natural parameter tensor $\mathcal{T}$ and the sparsity parameters λ_k (k = 1,2,3). The rank determines the upper bound of the total number of co-clusters to be identified (given that some unit-rank layers may actually capture global patterns). Thanks to the deflation algorithm and sparse estimation, the rank can be automatically selected. That is, we terminate the algorithm when one of the loadings in a layer is estimated to be zero. By design, no more local or global patterns can be found given what has been identified. Thus, the number of non-zero layers is a good estimate of r.

Regarding the sparsity parameters, one may particularly design some information criterion that balances the goodness of fit and parsimony, and search over a multi-dimensional grid for the best triplet. However, such a criterion may not be readily available and the computation may be prohibitive for higher-way tensor arrays. Alternatively, we exploit an adaptive approach to select the tuning parameters. Note that there is only one tuning parameter involved in (4). The proposed solution (see appendix) seeks to iteratively optimize a penalized least squares function, which resembles a standard LASSO regression problem. The tuning parameter selection for LASSO regression has been well studied in the literature [40]. Here we use the Bayesian information criterion (BIC) to select the tuning parameter within each step of the iterative algorithm. We remark that the adaptively selected tuning parameters may vary from one iteration to another, but our numerical studies show that they tend to stabilize within a few iterations. Such adaptive selection has been widely used in the literature for its computational advantage [35, 38, 6].

4. Simulation

In this section, we conduct comprehensive simulation studies to compare the proposed method with existing competitors. In particular, we consider EPCA [31], SSVD [6], convex biclustering [14], consistent biclustering [23], binary biclustering [24], Tensor Truncated Power (TTP) method [29], Poisson CP decomposition [37] and Generalized CP (GCP) decomposition [36] under different scenarios. For fair comparison, we set the rank to be the true rank whenever applicable and use the default setting to select tuning parameters for each method. We further explore the rank selection accuracy of the proposed method in Section 4.3. Additional simulation results on model misspecification can be found in the appendix.

4.1. Settings

We consider the following settings in the main paper.

- **Setting 1 (count-valued matrix):** Let $\boldsymbol{X}$ be a 100 × 100 count-valued matrix, where each entry is generated from a Poisson distribution. The underlying natural parameter (logarithmic Poisson rate) matrix $\boldsymbol{\Theta}$ is generated by

$$\boldsymbol{\Theta} = \mu \mathbf{1}\mathbf{1}^T + \boldsymbol{u}_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \boldsymbol{v}_2^T,$$

  where $\mathbf{1}$ is a length-100 vector of 1s and $\mu = 2$. The loading vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ have nonzero values in entries 1–50 and 31–70; the loading vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ have nonzero values in entries 1–40 and 21–60. The nonzero values are generated from a uniform distribution on [−0.5, −0.4] and [0.4, 0.5] (to be bounded away from 0). We subsequently normalize the loading vectors to have unit norms, and further multiply $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ by 15 and 10. We note that $\mu \mathbf{1}\mathbf{1}^T$ implies the global structure and $\boldsymbol{u}_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \boldsymbol{v}_2^T$ captures the local biclustering structure. The biclusters induced by ($\boldsymbol{u}_1$, $\boldsymbol{v}_1$) and ($\boldsymbol{u}_2$, $\boldsymbol{v}_2$) are overlapping and non-exhaustive.

- **Setting 2 (binary matrix):** The setting is very similar to Setting 1, except that data are generated from Bernoulli distributions with the underlying natural parameter matrix $\boldsymbol{\Theta}$ containing log odds instead of log rates. We simulate $\boldsymbol{\Theta} = \boldsymbol{u}_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \boldsymbol{v}_2^T$ where the loadings are generated in the same way as in Setting 1. To enhance signal, we multiply standardized $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ by 100 and 80, respectively.

- **Setting 3 (continuous 3-way tensor):** The elements of a 3-way tensor array $\mathcal{X}$ of dimension 50 × 50 × 50 are simulated independently from univariate Gaussian distributions with the mean tensor $\mathcal{T}$ being

$$\mathcal{T} = \lambda_1 \boldsymbol{v}_{11} \circ \boldsymbol{v}_{12} \circ \boldsymbol{v}_{13} + \lambda_2 \boldsymbol{v}_{21} \circ \boldsymbol{v}_{22} \circ \boldsymbol{v}_{23},$$

where all the loading vectors are sparse. We particularly consider two scenarios where the two co-clusters defined by the two layers are of 1) equal size (i.e., 20 × 20 × 20) and 2) unequal size (i.e, 20 × 20 × 20 and 30 × 30 × 30). The first scenario is in favor of the competing method TTP because it is specifically designed to identify equal-sized co-clusters. In both scenarios, we vary the standard deviation of the Gaussian random variables to allow the signal-to-noise ratio (SNR, defined as the signal variance divided by the noise variance) ranging from 0.1 to 1.

- **Setting 4 (count-valued 3-way tensor):** The setting is similar to Setting 1 in spirit, where data are simulated from a Poisson distribution with the underlying natural parameter tensor $\mathcal{T}$. In particular, $\mathcal{T}$ has the global structure $\mu\mathbf{1} \circ \mathbf{1} \circ 1$ with $\mu = 2$ and the individual structure $\lambda_1 \boldsymbol{v}_{11} \circ \boldsymbol{v}_{12} \circ \boldsymbol{v}_{13} + \lambda_2 \boldsymbol{v}_{21} \circ \boldsymbol{v}_{22} \circ \boldsymbol{v}_{23}$ with $\lambda_1 = 50$ and $\lambda_2 = 40$. The subtensor $\boldsymbol{v}_{11} \circ \boldsymbol{v}_{12} \circ \boldsymbol{v}_{13}$ has nonzero values in $\{1, \cdots, 20\} \times \{1, \cdots, 20\} \times \{1, \cdots, 20\}$ while the subtensor $\boldsymbol{v}_{21} \circ \boldsymbol{v}_{22} \circ \boldsymbol{v}_{23}$ has nonzero values in $\{11, \cdots, 40\} \times \{11, \cdots, 40\} \times \{21, \cdots, 40\}$. The nonzero entries in the loading vectors are generated in the same way as in Setting 1.

Other than the proposed CORALS method, we implement SSVD, EPCA, consistent biclustering and convex biclustering under Setting 1 (SSVD and convex biclustering are applied to the log-transformed data); EPCA, consistent biclustering and binary biclustering are implemented under Setting 2; CP and TTP are implemented under Setting 3; CP, Poisson CP and GCP methods are implemented under Setting 4. Since most methods cannot identify co-clusters or capture overlapping, non-exhaustive patterns, the main comparison criterion is the estimation accuracy of the underlying natural parameters. We evaluate the following Frobenius loss for estimates from different methods

$$\text{Loss} = \|\widehat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}\|_{\mathbb{F}}, \quad \text{or} \quad \|\widehat{\mathcal{T}} - \mathcal{T}\|_{\mathbb{F}}.$$

Moreover, for those methods that can capture overlapping co-clusters (e.g., SSVD and binary biclustering), we also compare the sensitivity and specificity of co-cluster identification, which are defined as follows:

$$\text{Sen} = \frac{\#\text{ of correctly identified co-cluster elements}}{\#\text{ of true co-cluster elements}},$$
$$\text{Spc} = \frac{\#\text{ of correctly specified non-co-cluster elements}}{\#\text{ of true non-co-cluster elements}}.$$

In addition, we also compare the fitting times of different methods. Under each setting, we fix the generative natural parameters and repeat the simulation 100 times.

4.2. Results

The results from Setting 1 are presented in Table 1. The proposed method has the lowest loss of parameter estimation among all methods. It also provides reasonably high sensitivity and specificity rates. We note that both convex biclustering and consistent biclustering are designed for exhaustive biclustering, and thus they cannot adequately identify overlapping biclusters. The fitting time of CORALS is similar to EPCA, and is significantly faster than

convex biclustering. The results from Setting 2 are shown in Table 2. Similar to the results from Setting 1, CORALS outperforms the competing methods in estimation loss and has desirable sensitivity and specificity of bicluster identification.

In Setting 3, we compare CORALS with CP and TTP over a range of SNRs in both scenarios. The results are presented in Figure 1. In the first scenario where co-clusters in different layers are of equal size, TTP has the smallest losses. This is not surprising because the scenario is in favor of TTP and the true co-cluster size is only provided to TTP as a required input. When the signal increases, the loss decreases for all methods and the difference between different methods diminishes. In the second scenario where co-clusters have different sizes, both CORALS and CP significantly outperform TTP, especially when the SNR is high. Moreover, CORALS is computationally more efficient than TTP (the model fitting time of CORALS is over 20 times faster than TTP in this setting).

Table 3 contains the results from Setting 4. Again, CORALS has the lowest estimation loss while achieving satisfactory co-clustering performance. None of the competing methods can achieve co-clustering. The computing time of CORALS depends on the data distribution and dimension, but overall it is computationally efficient.

4.3. Rank Selection

As a proof of concept, we rerun the simulation under Settings 1, 2 and 4 with the proposed CORALS method, and let the method determine the number of co-clusters in each case. In other words, we do not specify the rank, but let the deflation algorithm determines when to terminate. In Settings 1 and 2, the algorithm correctly identifies the true rank in almost all simulation runs ($\geq 99\%$). In Setting 4, the count-valued tensor setting, the deflation algorithm correctly specifies the rank more than two thirds of the simulation runs. For the remaining runs, it mostly underestimates the true rank by one. This may be due to the relatively low signal level. Nonetheless, the first few layers still capture the dominant global and local patterns, and are not subject to the rank misspecification.

5. Real Applications

We apply CORALS to three real data examples and demonstrate its utility. The NIPS Bag of Words data are publicly available from https://archive.ics.uci.edu/ml/datasets/NIPS+Conference+Papers+1987-2015 [16]; the CAL500 data are available from the Mulan library (http://mulan.sourceforge.net/datasets-mlc.html) [15]; the multiple sclerosis (MS) data are available from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE33464 [41]. The Matlab code for analyzing the data sets is available upon request.

5.1. NIPS Bag of Words

The NIPS Bag of Words data consist of 5,811 NIPS conference papers published from 1987 to 2015. The data set contains counts of 11,463 unique words by paper. Analysis of the word counts may reveal interesting evolution of topics in the field of machine learning. We first preprocess the data by aggregating papers from the same year. The number of papers per year is presented in the left panel of Figure 2. We further reduce the number of words by

removing words with zero count in any year. As a result, we end up with 3,556 unique words. We subsequently apply the proposed method to the word×year count-valued matrix.

The method identifies 4 layers of global and local patterns. In particular, the first layer corresponds to the global structure. The loading corresponding to years is shown in the right panel of Figure 2. It apparently coincides with the trend of paper volumes over years, indicating that the global pattern mainly captures the overall volume effect.

Next, we look into the 3 layers of local patterns. In each layer, the non-zero entries in both loadings form a bicluster. Following the remark in Section 2, we specially focus on the two sub-biclusters with positive values within each layer. In particular, one sub-bicluster consists of words and years both with positive loading values, and the other with negative loading values. These sub-biclusters capture more-frequent-than-usual appearance of words in a period of time. They potentially reflect the surge of certain topics in certain years. In total, there are 6 word×year sub-biclusters of size 287 × 19,530 × 10,127 × 18,118 × 4,13 × 21 and 82 × 7 identified from the 3 layers of local patterns. In particular, Bicluster 1 (BC1) lasts from 1987 to 2005 and highlights words such as “network”, “neural” “architecture” and “pattern”, which imply the popularity of neural network and pattern recognition-related topics in the 90s and early 2000. BC3 lasts from 1995 to 2012 and highlights “kernel”, “spike”, “margin”, “label”, “clustering” and “mixture”, which indicate the prevalence of supervised and unsupervised learning during the period. BC4 mainly consists of 2013 to 2015 and highlights “network”, “layer”, “deep” and “net” as a result of the deep-learning fever. BC5 spans 1987–1997 and 2006–2014 and highlights “parallel”, “distributed” and “memory”, which suggest the long-lasting interest of parallel/distributed computing in the field. These findings are in part consistent with those in [16] and provide interesting insights into the evolution of topics in the machine learning field.

5.2. CAL500 Music Annotation

The CAL500 data set contains semantic annotations for 502 western popular songs. The original data contain 174 candidate annotations, including 36 emotion features, 47 genres, 15 usage variables, 33 instrument variables, 27 characteristic features and 16 vocal types. Each song can be tagged with multiple annotations. The annotations of a song were determined by multiple human listeners with consensus. A more detailed description can be found in [15]. We preprocess the data by filtering out annotations that show up in fewer than 30 songs. As a result, we end up with a 502 × 103 binary matrix, with 1 indicating the presence of an annotation in a song and 0 otherwise. A heat map of the matrix is shown in the left panel of Figure 3.

One question of interest is to identify song sets that share similar annotations. Moreover, we are interested in knowing what the common annotations are that distinguish each song set. Once achieved, it will enhance our understanding of the relationship between semantic annotations and songs, and facilitate music retrieval. The process of identifying songs and annotations is essentially a biclustering analysis. Thus, we apply CORALS to the processed binary matrix. The method identify 3 layers of structure, including a global pattern and two local patterns of size 421 × 87 and 386 × 90. The estimated loadings of the local patterns are not as sparse as desired, possibly due to a low signal-to-noise level. To improve

interpretation, we focus on the most dominant entries in both dimensions. In particular, we threshold the normalized loadings for songs by ±0.05 (values between −0.05 and 0.05 are set to 0) and loadings for annotations by ±0.1. Subsequently, we obtain two local patterns of size 144×29 and 133×18. We further restrict our focus on sub-matrices of the local patterns with positive estimated values (see the remark in Section 2). As a result, we get 4 meaningful sub-biclusteis of 79 × 14, 65 × 15, 67 × 11 and 66 × 7. The identified biclusters are shown in the right panel of Figure 3. They are highly accordant with the observed data.

We further look into the identified sub-biclusters. The first sub-bicluster highlights annotations such as “Emotion-Pleasant-Comfortable”, “Emotion-Touching-Loving”, “Not Heavy Beat” and “Usage-Going to sleep”, and includes songs such as “Imagine” by John Lennon, “In the mood” by Glenn Miller and “For you and I” by 10cc. The second sub-bicluster, which belongs to the same local pattern with the first sub-bicluster but is on the other end of the spectrum, highlights annotations such as “Emotion-Angry-Aggressive”, “Emotion-Powerful-Strong”, “Song-Fast-Tempo” and “Song-High-Energy”, and includes songs such as “Six pack” by Black Flag and “Last depression” by Skitzo. The third and fourth sub-biclusters capture the contrast of songs with happy/positive emotions vs sad/negative emotions. In summary, the method proves useful for identifying similar songs based on semantic annotations.

5.3. Multiple Sclerosis

The MS data were collected to study the transcriptional effects of subcutaneous Interferon (IFN)-β treatment in patients with MS. Blood samples were collected from 12 patients at multiple time points (before first and second IFN-β injection as well as after 1 month, 1 year and 2 years). Gene expression profiles were measured by Affymetrix DNA microarrays. We follow the preprocessing steps in [26] and focus on the therapy-related genetic pathways with 56 genes. As a result, we obtain a gene expression array (gene×subject×time) of size 56 × 12 × 5. Our goal is to identify co-clusters that shed light on the effect of IFN-β therapy on gene expressions.

We apply CORALS to the MS data. The method identifies 5 layers of non-zero structures, including 1 global pattern and 4 local patterns. In particular, the 4 co-clusters are very similar with slight differences in different dimensions. The numbers of selected genes in different co-clusters are 16, 12, 13 and 23. We focus on 12 genes (CXCL10, DDX58, EIF2AK2, IFI27, IFIH1, IFIT1, IRF7, ISG15, MX1, OAS1, RSAD2, ZBP1) that are commonly selected by most co-clusters and conduct a gene ontology (GO) enrichment analysis using the analysis tool from the PANTHER Classification System (http://pantherdb.org/). The significant results under the nominal level of 0.05 are presented in Table 4.

We find that the selected genes are highly enriched for molecular function (RNA binding) and biological processes (defense response to virus and negative regulation of viral genome replication). Within the regulation process, the following functional subclasses are also significantly enriched: negative regulation of viral life cycle, regulation of viral process, regulation of symbiosis (encompassing mutualism through parasitism), regulation of viral life cycle, and regulation of viral genome replication. Some of these findings have been

reported in the literature [42, 26]. We further identify that the Interferon Signaling pathway is significantly enriched (1.97 fold difference between the 12 selected genes and 56 reference genes) with a p-value of 0.032. Since most samples were measured after the IFN-β treatment, it is reasonable that the interferon signaling pathway genes are significantly enriched in the selected gene set. Overall, the proposed method replicates existing results and may provide new insights into the transcriptional effects of IFN-β treatment.

6. Discussion

We propose a general framework for co-clustering analysis of multi-way tensor arrays. The method can effectively accommodate continuous, binary or count-valued data by identifying co-clusters in the natural parameter space. The model fitting algorithm is adaptive and efficient. The number of co-clusters can be automatically determined from the estimation procedure. Numerical studies demonstrate the efficacy and utility of the proposed method.

The proposed method can be readily generalized to composite data with different data types. With prior information about the distribution of each data entry, one could simply modify the estimation algorithm in Section 3 by using distribution-specific likelihood functions. However, a co-cluster in composite data may not have a straightforward interpretation because natural parameters in different distributions have different meanings. A related future research direction is to adapt the method to multi-view multi-type data, where multiple data sets are measured on the same set of samples and each data set may have a unique data type [43, 44]. How to simultaneously cluster samples and variables (in different sets) is an intriguing question.

We only consider single-parameter exponential family distributions in this paper. The extension to multi-parameter distributions may also be desired. However, several challenges remain. First, different parameters may have different co-clustering patterns, and it is not clear about how to define consensus co-clusters. Second, the proposed model may be over-saturated for multi-parameter distributions since we treat each entry of a tensor as an observation in the current framework. Third, the computation may be prohibitive because we can no longer leverage the equivalence to a GLM problem in the estimation procedure. More efforts are needed to address these challenges.

Another direction of interest is to generalize the CP decomposition to the Tucker decomposition. The Tucker decomposition has grown in popularity in recent years due to its general form. It subsumes the CP decomposition as a special case. Many tensor models are based on the Tucker decomposition [see 45, 46, for example]. More recently, [47] proposed a tensor block model for multiway clustering, which is a special form of the Tucker decomposition. It can be used to identify exclusive, constant-mean clusters. Nonetheless, it remains an open area to exploit the Tucker decomposition for flexible overlapping co-clustering.

There are a few additional open questions for future research. First, the computational efficiency of CORALS may be further improved for large-scale problems. The proposed estimation algorithm has nested loops of iterations. Within each iteration, all the entries of a

tensor are stacked into a long vector for computation. These may lead to high computational cost and excessive memory utilization, especially for large-scale problems. One could consider a one-iteration variant and a subsampling scheme to simplify the algorithm. However, computational properties such as the convergence rate need to be carefully studied. Second, same-sign co-clusters in a tensor may be identified by directly imposing some sign restrictions on loading vectors. For instance, non-negative loading entries in all the loading vectors in a layer would lead to a positive co-cluster. However, challenges include how to adaptively determine the sign of a loading vector and how to efficiently solve the optimization in (4) with sign restrictions. Third, the generalization to incomplete tensor data is also of interest and calls for more investigation.

Acknowledgements

This work was supported by the National Institutes of Health [R01HG010731-01A1].

Appendix

Appendix A. Iteratively Reweighted Penalized Least Squares

In our model fitting algorithm, a critical piece is to fit a generalized linear model (GLM) with the canonical link and a sparsity-inducing penalty. In a general setting, let $\boldsymbol{y} \in \mathbb{R}^n$ denote a response vector, $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ denote a design matrix and $\boldsymbol{\beta} \in \mathbb{R}^p$ denote a coefficient vector. The optimization problem can be formulated as

$$\min_{\boldsymbol{\beta}} \; -\boldsymbol{y}^T \boldsymbol{X}\boldsymbol{\beta} - \mathbf{1}^T b(\boldsymbol{X}\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1, \tag{A.1}$$

where $b(\cdot)$ is an entrywise convex function corresponding to the data distribution in the exponential family and λ is a tuning parameter. The first two terms come from the exponential family log likelihood and the last term is a LASSO penalty. In particular, the objective function (4) in the main paper is a special case of (A.1).

We note that the objective function in (A.1) is convex. We exploit an iteratively reweighted least squares approach to solve (A.1). We first apply a quadratic approximation to the first two terms of the objective function in (A.1), and convert it to a weighted least squares term. More specifically, let $Q(\boldsymbol{\beta}) = -\boldsymbol{y}^T \boldsymbol{X}\boldsymbol{\beta} - \mathbf{1}^T b(\boldsymbol{X}\boldsymbol{\beta})$. We have

$$\frac{\partial Q(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\boldsymbol{X}^T \boldsymbol{y} + \boldsymbol{X}^T b'(\boldsymbol{X}\boldsymbol{\beta}),$$
$$\frac{\partial^2 Q(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T} = \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X},$$

where $\boldsymbol{W}$ is a diagonal matrix with diagonal values being $b''(\boldsymbol{X}\boldsymbol{\beta})$. Correspondingly, the quadratic approximation of $Q(\boldsymbol{\beta})$ at $\boldsymbol{\beta}_0$ is

$$Q(\boldsymbol{\beta}) \approx Q(\boldsymbol{\beta}_0) + (\boldsymbol{\beta} - \boldsymbol{\beta}_0)^T \frac{\partial Q(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} |_{\boldsymbol{\beta}_0} + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}_0)^T \frac{\partial^2 Q(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T} |_{\boldsymbol{\beta}_0} (\boldsymbol{\beta} - \boldsymbol{\beta}_0)$$
$$\propto \|\boldsymbol{W}^{1/2}\boldsymbol{y}^\star - \boldsymbol{W}^{1/2}\boldsymbol{X}\boldsymbol{\beta}\|^2,$$

where $\boldsymbol{y}^* = \boldsymbol{X}\boldsymbol{\beta}_0 + [\boldsymbol{y} - b'(\boldsymbol{X}\boldsymbol{\beta}_0)] \cdot [b''(\boldsymbol{X}\boldsymbol{\beta}_0)]^{-1}$ is a working response vector and $\cdot$ represents the Hadamard product.

Given the current estimate $\boldsymbol{\beta}_0$, the original problem (A.1) is approximated by

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{W}^{1/2}\boldsymbol{y}^\star - \boldsymbol{W}^{1/2}\boldsymbol{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{A.2}$$

which is a penalized weighted least squares problem. There are many off-the-shelf methods to address this problem [48]. In particular, if the design matrix $\boldsymbol{W}^{1/2}\boldsymbol{X}$ have orthogonal columns, which is the case for the optimization (4) in the main paper, the above problem has a closed-form solution

$$\hat{\boldsymbol{\beta}} = \text{thres}\left(\boldsymbol{\beta}_{WLS}, 2\lambda diag\left[(\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\right]\right),$$

where thres($\boldsymbol{\beta}$, $\boldsymbol{\lambda}$) is an entrywise soft-thresholding function for $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\boldsymbol{\lambda} \in \mathbb{R}^p$ with each entry being thres(β_j, λ_j) = sign(β_j) max($|\beta_j| - \lambda_j$, 0), and $\boldsymbol{\beta}_{WLS} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{y}^*$ is the weighted least squares estimate.

Once obtained, $\hat{\boldsymbol{\beta}}$ becomes the current estimate and is used to calculate the weight matrix $\boldsymbol{W}$ and the working response $\boldsymbol{y}^*$. Then we solve (A.2) again. All in all, the final solution of (A.1) can be obtained by iteratively solving (A.2).

Appendix B. Additional Simulation on Model Misspecification

We conduct additional simulation studies where data are not generated from the proposed model. In particular, we consider the following settings

- **Setting S1 (continuous matrix with block mean):** Following a simulation setting in [14], we generate continuous data from the following block model

$$\boldsymbol{X} = \begin{bmatrix} \mu_a \mathbf{1}_a & \mu_a \mathbf{1}_a & \mu_d \mathbf{1}_d \\ \mu_b \mathbf{1}_b & \mu_c \mathbf{1}_c & \mu_e \mathbf{1}_e \end{bmatrix} + E,$$

  where $\mathbf{1}.$ is a matrix of ones with compatible size, $\boldsymbol{\mu} = (\mu_a, \mu_b, \mu_c, \mu_d, \mu_e)^T = (1, 0, 0.25, -1, 1.25)^T$ is a mean vector, and $\boldsymbol{E}$ contains independent and identically distributed (i.i.d.) noise from $N(0, 0.1)$. The data matrix has dimensions 100×100 and $\mathbf{1}_a$ has dimensions 40×30.

- **Setting S2 (count-valued matrix with block mean):** Following a simulation setting in [23], we generate Poisson count data with the following block-structured mean parameters

$$\mathbf{\Lambda} = \begin{bmatrix} \mu_a \mathbf{1}_a & \mu_c \mathbf{1}_c & \mu_e \mathbf{1}_e \\ \mu_b \mathbf{1}_b & \mu_d \mathbf{1}_d & \mu_f \mathbf{1}_f \end{bmatrix},$$

where $\boldsymbol{\mu} = (\mu_a, \mu_b, \mu_c, \mu_d, \mu_e, \mu_f)^T = 10 * (0.92, 0.17, 0.77, 1.41, 1.66, 1.45)^T$. The dimensions of the blocks are the same as those in **Setting S1**.

- **Setting S3 (count-valued matrix with over-dispersion):** Count-valued data are generated from a negative binomial distribution where the mean structure is the same as in **Setting 1** and the dispersion parameter is randomly generated from a uniform distribution with support [5, 10] (i.e., moderate to large dispersion).

We compare CORALS with both convex biclustering and consistent biclustering under Settings S1 and S2. Since the underlying mean parameters have an exclusive and complete block structure, we also measure the performance of different methods with the Rand Index (RI) [49]. The index captures the similarity between estimated and true partitions. We use an ad hoc partition method for CORALS results, where rows and columns are partitioned by the composite signs of the estimated loading vectors. In Setting S3, we compare CORALS with SSVD, EPCA, convex biclustering and consistent biclustering. In particular, for CORALS and consistent biclustering, we use the Poisson likelihood, which is a misspecified distribution for the data.

The results are presented in Tables B.5 and B.6. In Settings S1, convex biclustering has the best performance with the smallest Frobenius loss and largest RI. This is not surprising since the generative block model in this setting is the probabilistic model for convex biclustering. However, the superior performance comes at a price – the fitting time of convex biclustering is more than 50 times greater than that of CORALS. In Setting S2, where the generative model is from consistent biclustering, it indeed has the best performance followed by convex biclustering and CORALS. In both settings, the lesser performance of the proposed method indicates that the current form of CORALS and the ad hoc partition method may not be suitable for data with a block mean structure. Further investigations are needed to better adapt the method to this scenario. In Setting S3, even with a misspecified distribution, CORALS is still among the best. It has a significantly lower loss than SSVD and convex biclustering, and has the best sensitivity and specificity rates of co-cluster identification. The result shows the proposed method is relatively robust against the distribution misspecification.

Table B.5.

The comparison of different methods under **Settings S1 and S2**. The mean and standard deviation (in parenthesis) of different criteria are calculated based on 100 simulation runs.

		CORALS	CvxBC	CstBC
Setting S1	Loss	7.172(0.442)	1.128(1.085)	11.323(17.942)
	RI	0.918(0.008)	0.956(0.047)	0.933(0.065)
	Time (sec)	0.119(0.335)	6.483(0.398)	3.975(0.262)

		CORALS	CvxBC	CstBC
Setting S2	Loss	12.499(0.475)	8.887(0.907)	0.912(0.342)
	RI	0.870(0.003)	0.972(0.051)	1.000(0.000)
	Time (sec)	0.171(0.009)	9.488(1.156)	4.021(0.087)

Table B.6.

The comparison of different methods under **Setting S3**. The mean and standard deviation (in parenthesis) of different criteria are calculated based on 100 simulation runs.

	CORALS	SSVD	CvxBC	CstBC	EPCA
Loss	15.531(1.512)	22.996(1.006)	24.426(0.540)	13.432(1.270)	14.170(0.422)
Sen	0.568(0.207)	0.549(0.183)	N/A	N/A	1(0)
Spc	0.926(0.041)	0.923(0.042)	N/A	N/A	0(0)
Time (sec)	0.956(2.685)	0.769(1.311)	5.593(1.290)	0.009(0.004)	3.448(0.823)

References

[1]. Pontes B, Giráldez R, Aguilar-Ruiz JS, Biclustering on expression data: A review, Journal of biomedical informatics 57 (2015) 163–180. [PubMed: 26160444]

[2]. Busygin S, Prokopyev O, Pardalos PM, Biclustering in data mining, Computers & Operations Research 35 (2008) 2964–2987.

[3]. Fan N, Boyko N, Pardalos PM, Recent advances of data biclustering with application in computational neuroscience, in: Computational Neuroscience, Springer, 2010, pp. 85–112.

[4]. Hartigan JA, Direct clustering of a data matrix, Journal of the american statistical association 67 (1972) 123–129.

[5]. Sill M, Kaiser S, Benner A, Kopp-Schneider A, Robust biclustering by sparse singular value decomposition incorporating stability selection, Bioinformatics 27 (2011) 2089–2097. [PubMed: 21636597]

[6]. Lee M, Shen H, Huang JZ, Marron J, Biclustering via sparse singular value decomposition, Biometrics 66 (2010) 1087–1095. [PubMed: 20163403]

[7]. Dhillon IS, Co-clustering documents and words using bipartite spectral graph partitioning, in: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2001, pp. 269–274.

[8]. Kluger Y, Basri R, Chang JT, Gerstein M, Spectral biclustering of microarray data: coclustering genes and conditions, Genome research 13 (2003) 703–716. [PubMed: 12671006]

[9]. Cheng Y, Church GM, Biclustering of expression data, in: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, AAAI Press, 2000, pp. 93–103. URL: http://dl.acm.org/citation.cfm?id=645635.660833.

[10]. Shabalin AA, Weigman VJ, Perou CM, Nobel AB, Finding large average submatrices in high dimensional data, The Annals of Applied Statistics 3 (2009) 985–1012.

[11]. Tan KM, Witten DM, Sparse biclustering of transposable data, Journal of Computational and Graphical Statistics 23 (2014) 985–1008. [PubMed: 25364221]

[12]. Segal E, Taskar B, Gasch A, Friedman N, Koller D, Rich probabilistic models for gene expression, Bioinformatics 17 (2001) S243–S252. [PubMed: 11473015]

[13]. Segal E, Battle A, Koller D, Decomposing gene expression into cellular processes, in: Biocomputing 2003, World Scientific, 2002, pp. 89–100.

[14]. Chi EC, Allen GI, Baraniuk RG, Convex biclustering, Biometrics 73 (2017) 10–19. [PubMed: 27163413]

[15]. Turnbull D, Barrington L, Trres D, Lanckriet G, Towards musical query-by-semantic-description using the cal500 data set, in: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, 2007, pp. 439–446.

[16]. Perrone V, Jenkins PA, Spanò D, Teh YW, Poisson random fields for dynamic feature models, The Journal of Machine Learning Research 18 (2017) 4626–4670.

[17]. Govaert G, Nadif M, Block clustering with bernoulli mixture models: Comparison of different approaches, Computational Statistics & Data Analysis 52 (2008) 3233–3245.

[18]. Govaert G, Nadif M, Latent block model for contingency table, Communications in Statistics—Theory and Methods 39 (2010) 416–425.

[19]. Govaert G, Nadif M, Co-clustering: models, algorithms and applications, John Wiley & Sons, 2013.

[20]. Choi D, Wolfe PJ, Co-clustering separately exchangeable network data, The Annals of Statistics 42 (2014) 29–63.

[21]. Gao C, Lu Y, Ma Z, Zhou HH, Optimal estimation and completion of matrices with biclustering structures, The Journal of Machine Learning Research 17 (2016) 5602–5630.

[22]. Keribin C, Brault V, Celeux G, Govaert G, Model selection for the binary latent block model, in: Proceedings of COMPSTAT, volume 2012, 2012.

[23]. Flynn CJ, Perry PO, Consistent biclustering, arXiv preprint arXiv:1206.6927 (2012).

[24]. Lee S, Huang JZ, A biclustering algorithm for binary matrices based on penalized bernoulli likelihood, Statistics and Computing 24 (2014) 429–441.

[25]. Kolda TG, Bader BW, Tensor decompositions and applications, SIAM review 51 (2009) 455–500.

[26]. Zhao H, Wang DD, Chen L, Liu X, Yan H, Identifying multi-dimensional co-clusters in tensors based on hyperplane detection in singular vector spaces, PloS one 11 (2016) e0162293. [PubMed: 27598575]

[27]. Wu T, Benson AR, Gleich DF, General tensor spectral co-clustering for higher-order data, in: Advances in Neural Information Processing Systems, 2016, pp. 2559–2567.

[28]. Wang M, Fischer J, Song YS, Three-way clustering of multi-tissue multi-individual gene expression data using semi-nonnegative tensor decomposition, The Annals of Applied Statistics 13 (2019) 1103–1127.

[29]. Sun WW, Lu J, Liu H, Cheng G, Provable sparse tensor decomposition, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 79 (2017) 899–916.

[30]. Chi EC, Gaines BR, Sun WW, Zhou H, Yang J, Provable convex co-clustering of tensors, arXiv preprint arXiv:1803.06518 (2018).

[31]. Collins M, Dasgupta S, Schapire RE, A generalization of principal components analysis to the exponential family, in: Advances in neural information processing systems, 2001, pp. 617–624.

[32]. Tibshirani RJ, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 58 (1996) 267–288.

[33]. Fan J, Li R, Variable selection via nonconcave penalized likelihood and its oracle properties, Journal of the American statistical Association 96 (2001) 1348–1360.

[34]. Zhang C-H, Nearly unbiased variable selection under minimax concave penalty, The Annals of Statistics 38 (2010) 894–942.

[35]. Li G, Huang JZ, Shen H, Exponential family functional data analysis via a low-rank model, Biometrics 74 (2018) 1301–1310. [PubMed: 29738627]

[36]. Hong D, Kolda TG, Duersch JA, Generalized canonical polyadic tensor decomposition, SIAM Review 62 (2020) 133–163.

[37]. Chi EC, Kolda TG, On tensors, sparsity, and nonnegative factorizations, SIAM Journal on Matrix Analysis and Applications 33 (2012) 1272–1299.

[38]. Shen H, Huang JZ, Sparse principal component analysis via regularized low rank matrix approximation, Journal of multivariate analysis 99 (2008) 1015–1034.

[39]. Zou H, Hastie T, Tibshirani R, Sparse principal component analysis, Journal of computational and graphical statistics 15 (2006) 265–286.

[40]. Zou H, Hastie T, Tibshirani R, On the “degrees of freedom” of the lasso, The Annals of Statistics 35 (2007) 2173–2192.

[41]. Hecker M, Hartmann C, Kandulski O, Paap BK, Koczan D, Thiesen H-J, Zettl UK, Interferon-beta therapy in multiple sclerosis: the short-term and long-term effects on the patients’ individual gene expression in peripheral blood, Molecular neurobiology 48 (2013) 737–756. [PubMed: 23636981]

[42]. Moore JL, Du Z, Bao Z, Systematic quantification of developmental phenotypes at single-cell resolution during embryogenesis, Development 140 (2013) 3266–3274. [PubMed: 23861063]

[43]. Zhu H, Li G, Lock EF, Generalized integrative principal component analysis for multi-type data with block-wise missing structure, Biostatistics 21 (2020) 302–318. [PubMed: 30247540]

[44]. Li G, Gaynanova I, A general framework for association analysis of heterogeneous data, The Annals of Applied Statistics 12 (2018) 1700–1726.

[45]. Li X, Xu D, Zhou H, Li L, Tucker tensor regression and neuroimaging analysis, Statistics in Biosciences 10 (2018) 520–545.

[46]. Zhang X, Li L, Tensor envelope partial least-squares regression, Technometrics 59 (2017) 426–436.

[47]. Wang M, Zeng Y, Multiway clustering via tensor block models, in: Advances in Neural Information Processing Systems, 2019, pp. 713–723.

[48]. Efron B, Hastie T, Johnstone I, Tibshirani RJ, Least angle regression, The Annals of Statistics 32 (2004) 407–499.

[49]. Rand WM, Objective criteria for the evaluation of clustering methods, Journal of the American Statistical association 66 (1971) 846–850.

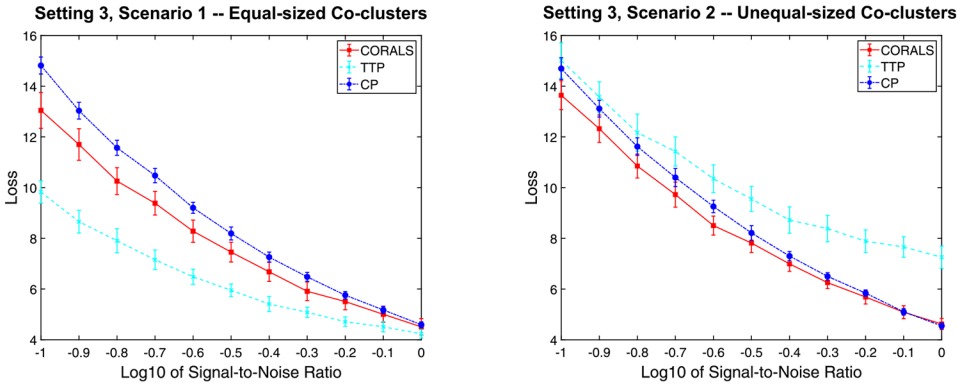

Figure 1.

Simulation results under **Setting 3.** The left panel corresponds to the first scenario where co-clusters in different layers are of equal size; the right panel corresponds to the second scenario where co-clusters have different sizes. The medians and median absolute deviations (i.e., the error bars) of the estimation losses for CORALS, CP and TTP are evaluated over a range SNRs.

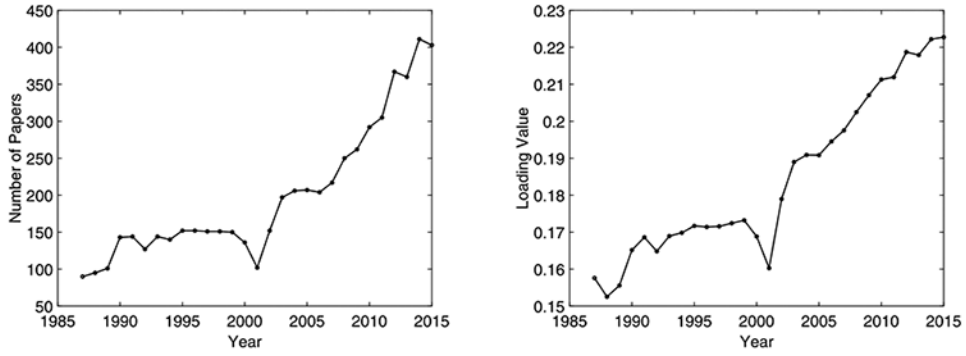

Figure 2.

NIPS Bag of Words Example. The left panel shows the numbers of papers published at NIPS from 1987 to 2015; the right panel contains the loading values correspond to the global structure estimated from CORALS.

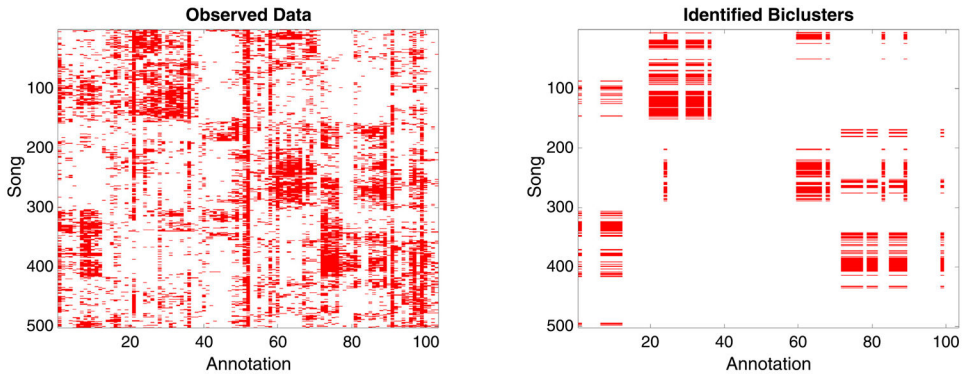

Figure 3.

CAL500 Music Annotation Example. The left panel shows the heat map of the binary annotation matrix where rows represent songs and columns represent annotations (red=presence; white=absence). The right panel shows the heat map of the identified biclusters (red=in a bicluster; white=not in a bicluster). Rows and columns are ordered in the same way in both figures for visualization purpose.

Table 1.

The comparison of different methods under **Setting 1**. CvxBC and CstBC represent convex biclustering and consistent biclustering respectively. The mean and standard deviation (in parenthesis) of different criteria are calculated based on 100 simulation runs.

	CORALS	SSVD	CvxBC	CstBC	EPCA
Loss	9.747(0.496)	13.600(0.945)	19.865(0.229)	10.552(0.256)	10.699(13.130)
Sen	0.980(0.026)	0.907(0.115)	N/A	N/A	1(0)
Spc	0.825(0.044)	0.846(0.047)	N/A	N/A	0(0)
Time (sec)	1.987(5.579)	0.941(1.419)	17.802(7.933)	0.009(0.008)	2.269(0.641)

Table 2.

The comparison of different methods under **Setting 2**. BinBC and CstBC represent binary biclustering and consistent biclustering respectively. The mean and standard deviation (in parenthesis) of different criteria are calculated based on 100 simulation runs.

	CORALS	BinBC	CstBC	EPCA
Loss	73.673(3.713)	94.056(3.881)	102.570(7.165)	129.714(21.453)
Sen	0.749(0.044)	0.556(0.049)	N/A	1(0)
Spc	0.822(0.055)	0.977(0.018)	N/A	0(0)
Time (sec)	3.751(10.323)	1.768(5.624)	0.012(0.007)	0.270(0.067)

Table 3.

The comparison of different methods under **Setting 4.** The CP method is applied to the log-transformed data. The mean and standard deviation (in parenthesis) are calculated based on 100 simulation runs.

	CORALS	CP	Poisson CP	GCP
Loss	25.300(18.147)	44.049(10.605)	65.123(0.073)	52.191(1.533)
Sen	0.748(0.360)	1(0)	1(0)	1(0)
Spc	0.933(0.040)	0(0)	0(0)	0(0)
Time (sec)	0.443(0.716)	0.019(0.010)	0.487(0.048)	0.628(0.115)

Table 4.

GO enrichment analysis of 12 selected genes in the multiple sclerosis example. Enriched biological processes are presented along with the counts and p-values from the Fisher's Exact test. In particular, $N = 56$ and $B = 12$ are the numbers of genes in the whole data and the selected subset, respectively; n and b are the numbers of genes associated with the corresponding GO term within N and B.

GO term	Description	Enrichment Counts (N, B, n, b)	p-value
GO:0051607	Defense response to virus	(56,12,26,11)	0.014
GO:0045071	Negative regulation of viral genome replication	(56,12,9,6)	0.026
GO:0003723	RNA binding	(56,12,13,7)	0.040