A Doubly-Enhanced EM Algorithm for Model-Based Tensor Clustering*

Qing Mai, Xin Zhang, Yuqing Pan and Kai Deng

Florida State University

Abstract

Modern scientific studies often collect data sets in the form of tensors. These datasets call for innovative statistical analysis methods. In particular, there is a pressing need for tensor clustering methods to understand the heterogeneity in the data. We propose a tensor normal mixture model approach to enable probabilistic interpretation and computational tractability. Our statistical model leverages the tensor covariance structure to reduce the number of parameters for parsimonious modeling, and at the same time explicitly exploits the correlations for better variable selection and clustering. We propose a doubly-enhanced expectation-maximization (DEEM) algorithm to perform clustering under this model. Both the Expectation-step and the Maximization-step are carefully tailored for tensor data in order to maximize statistical accuracy and minimize computational costs in high dimensions. Theoretical studies confirm that DEEM achieves consistent clustering even when the dimension of each mode of the tensors grows at an exponential rate of the sample size. Numerical studies demonstrate favorable performance of DEEM in comparison to existing methods.

Keywords: Clustering; the EM Algorithm; Gaussian Mixture Models; Kronecker Product Covariance; Minimax; Tensor.

*Corresponding author: Xin Zhang (xzhang8@fsu.edu). The authors would like to thank the Co-Editors, Associate Editor and reviewers for helpful comments. Research for this paper was supported in part by grants CCF-1617691 and CCF-1908969 from the National Science Foundation.

1 Introduction

Tensor data are increasingly popular in modern scientific studies. Research in brain image analysis, personalized recommendation and multi-tissue multi-omics studies often collect data in the form of matrices (i.e, 2-way tensors) or higher-order tensors for each observation. The tensor structure brings challenges to the statistical analysis. On one hand, tensor data are often naturally high-dimensional. This leads to an excessive number of parameters in statistical modeling. On the other hand, the tensor structure contains information that cannot be easily exploited by classical multivariate, i.e. vector-based, methods. Motivated by the prevalence of tensor data and the challenges to statistical analysis, a large number of novel tensor-based methods have been developed in recent years. There is a rapidly growing literature on the analysis of tensor data, for example, on tensor decomposition (Chi & Kolda 2012, Sun et al. 2016, Zhang & Han 2019), regression (Zhou et al. 2013, Hoff 2015, Raskutti et al. 2019, Wang & Zhu 2017, Li & Zhang 2017, Zhang & Li 2017, Lock 2018) and classification (Lyu et al. 2017, Pan et al. 2019). These methods, among many others, take advantage of the tensor structure to drastically reduce the number of parameters, and use tensor algebra to streamline estimation and advance theory.

We study the problem of model-based tensor clustering. When datasets are heterogeneous, cluster analysis sheds light on the heterogeneity by grouping observations into clusters such that observations within each cluster are similar to each other, but there is noticeable difference among clusters. For more background, see Fraley & Raftery (2002) and McLachlan et al. (2019) for overviews of model-based clustering. Various approaches have been proposed in recent years for clustering on high-dimensional vector data (Ng et al. 2001, Law et al. 2004, Arthur & Vassilvitskii 2007, Pan & Shen 2007, Wang & Zhu 2008, Guo et al. 2010, Witten & Tibshirani 2010, Cai et al. 2019, Verzelen & Arias-Castro 2017, Hao et al. 2018). Although many of these vector methods could be applied to tensor data by vectorizing the tensors first, this brute-force approach is generally not recommended, because the vectorization completely ignores the tensor structure. As a result, vectorization could often lead to loss of information, and thus efficiency and accuracy. It is much more desirable to have clustering methods specially designed for tensor data.

Model-based clustering often assumes a finite mixture of distributions for the data. In particular, the Gaussian mixture model (GMM) plays an important role in high-dimensional statistics

due to its flexibility, interpretability and computational convenience. Motivated by GMM, we consider a tensor normal mixture model (TNMM). In comparison to the existing GMM methods for vector data, TNMM exploits the tensor covariance structure to drastically reduce the total number of parameters in covariance modeling. Thanks to the simplicity of matrix/tensor normal distributions, clustering and parameter estimation is straightforward based on the expectation-maximization (EM) algorithm (Dempster et al. 1977) . Among others, Viroli (2011), Anderlucci & Viroli (2015), Gao et al. (2021), Gallaugher & McNicholas (2018) are all extensions of GMM from vector to matrix, but are not directly applicable to higher-order tensors. Moreover, the focus of these works is computation and applications in the presence of additional information, such as covariates, longitudinal correlation, heavy tails and skewness in the data, but no theoretical results are provided for high dimensional data analysis. The GMMs can be straightforwardly extended to higher-order tensors adopting the standard EM algorithm. However, as we demonstrate in numerical studies, the standard EM can be dramatically improved by our Doubly-Enhanced Expectation-Maximization (DEEM) algorithm.

The DEEM algorithm is developed under TNMM to efficiently incorporate tensor correlation structure and variable selection for clustering and parameter estimation. Similar to classical EM algorithms, DEEM iteratively carries out an enhanced E-step and an enhanced M-step. In the enhanced E-step, we impose sparsity directly on the optimal clustering rule as a flexible alternative to popular low-rank assumptions on tensor coefficients. The variable selection empowers DEEM to high-dimensional tensor data analysis. In the enhanced M-step, we employ a new estimator for the tensor correlation structure, which facilitates both the computation and the theoretical studies. These modifications to the standard EM algorithm are very intuitive and practically motivated. More importantly, we show that the clustering error of DEEM converges to the optimal clustering error at the minimax optimal rate. DEEM is also highly competitive in empirical studies.

To achieve variable selection and clustering simultaneously, we impose the sparsity assumption on our model and then incorporate a penalized estimator in DEEM. Although penalized estimation is a common strategy in high-dimensional clustering, there are many different approaches. For example , Wang & Zhu (2008) penalize cluster means; Guo et al. (2010), Verzelen & Arias-Castro (2017) penalize cluster mean differences; Pan & Shen (2007), Witten & Tibshirani (2010), Law et al. (2004) achieve variable selection by assuming independence among variables; Hao

et al. (2018) impose sparsity on both cluster means and precision matrices; Cai et al. (2019) impose sparsity on the discriminant vector. Our approach is similar to Cai et al. (2019) in that our sparsity assumption is directly imposed on the discriminant tensor coefficients – essentially a re-parameterization of the means and covariance matrices to form sufficient statistics in clustering. As a result of this parameterization, the correlations among variables are utilized in variable selection, while the parameter of interest has the same dimensionality as the cluster mean difference.

Due to the non-convex nature of cluster analysis, conditions on the initial value are commonly imposed in theoretical studies. Finding theoretically guaranteed initial values for cluster analysis is an important research area on its own, with many interesting works under GMM (Kalai et al. 2010, Moitra & Valiant 2010, Hsu & Kakade 2013, Hardt & Price 2015). To provide a firmer theoretical ground for the consistency of DEEM, we further develop an initialization algorithm for TNMM in general, which may be of independent interest. A brief discussion on the initialization is provided in Section 4.2. The detailed algorithm (Algorithm S.4) and related theoretical studies are provided in Section G of Supplementary Materials.

Two related but considerably different problems are worth-mentioning, but beyond the scope of this article. The first is the low-rank approximation in K-means clustering (MacQueen 1967, Cohen et al. 2015). For example, Sun & Li (2018) use tensor decomposition in the minimization of the total squared Euclidean distance of each observation to its cluster centroid. While the low-rank approximation is widely adopted in tensor data analysis, our method is more directly targeted at the optimal rule of clustering under the TNMM, and does not require low-rank structure of the tensor coefficients. The second is the clustering of features (variables) instead of, or, along with observations. Clustering variables into similar groups has applications in a wide range of areas such as genetics, text mining and imaging analysis, and also has attracted substantial interest in theoretical studies. For example, Bing et al. (2020), Bunea et al. (2020) studied feature clustering in high dimensions; Lee et al. (2010), Tan & Witten (2014), Chi et al. (2017) developed bi-clustering methods that simultaneously group features and observations into clusters. Extensions of the feature-sample bi-clustering for vector observations are known as the *co-clustering* or *multiway clustering* problems (Kolda & Sun 2008, Jegelka et al. 2009, Chi et al. 2020, Wang & Zeng 2019), where each mode of the tensor is clustered into groups, resulting in a checkerbox structure. Our problem is different from these works in that our sole goal is to cluster the observations.

The rest of the paper is organized as follows. In Section 2, we formally introduce the model and discuss the importance of modeling the correlation structure. In Section 3, we propose the DEEM algorithm. Theoretical results are presented in Section 4. Section 5 contains numerical studies on simulated and real data. Additional numerical studies, proofs and other technical details are relegated to Supplementary Materials.

2 The Model

2.1 Notation and Preliminaries

A multi-dimensional array $\mathbf{A} \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ is called an M-way tensor. We denote $\mathcal{J} = (j_1, \ldots, j_M)$ as the index of one element in the tensor. The vectorization of $\mathbf{A}$ is a vector, $\mathrm{vec}(\mathbf{A})$, of length $(\prod_{m=1}^{M} p_m)$. The mode-k matricization of a tensor is a matrix of dimension $(p_k \times \prod_{m \neq k} p_m)$, denoted by $\mathbf{A}_{(k)}$, where the $(j_1, \ldots, j_M)$-th element of $\mathbf{A}$ is the (j_k, l)-th element of $\mathbf{A}_{(k)}$ with $l = 1 + \sum_{m=1, m\neq k}^{M} \{(j_m - 1) \prod_{t=1, t\neq k}^{m-1} p_t\}$. A tensor $\mathbf{C} \in \mathbb{R}^{d_1 \times \cdots \times d_M}$ can be multiplied with a $d_m \times p_m$ matrix $\mathbf{G}_m$ on the m-th mode, denoted as $\mathbf{C} \times_m \mathbf{G}_m \in \mathbb{R}^{d_1 \times \cdots \times d_{m-1} \times p_m \times d_{m+1} \times \cdots \times d_M}$. If $\mathbf{A} = \mathbf{C} \times_1 \mathbf{G}_1 \times \cdots \times_M \mathbf{G}_M$, we equivalently write the Tucker decomposition of $\mathbf{A}$ as $\mathbf{A} = [\![\mathbf{C}; \mathbf{G}_1, \ldots, \mathbf{G}_M]\!]$. A useful fact is that $\mathrm{vec}([\![\mathbf{C}; \mathbf{G}_1, \ldots, \mathbf{G}_M]\!]) = (\mathbf{G}_M \otimes \cdots \otimes \mathbf{G}_1)\mathrm{vec}(\mathbf{C}) \equiv (\bigotimes_{m=M}^{m=1} \mathbf{G}_m)\mathrm{vec}(\mathbf{C})$, where $\otimes$ represents the Kronecker product. The inner product of two tensors $\mathbf{A}, \mathbf{B}$ of matching dimensions is defined as $\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{\mathcal{J}} a_{\mathcal{J}} b_{\mathcal{J}}$. For more background on tensor algebra, see Kolda & Bader (2009).

The tensor normal distribution is an extension of matrix multivariate normal distribution (Gupta & Nagar 1999, Hoff 2011). For a random tensor $\mathbf{X} \in \mathbb{R}^{p_1 \times \cdots \times p_M}$, if $\mathbf{X} = \boldsymbol{\mu} + [\![\mathbf{Z}; \boldsymbol{\Sigma}_1^{1/2}, \ldots, \boldsymbol{\Sigma}_M^{1/2}]\!]$ for $\boldsymbol{\mu} \in \mathbb{R}^{p_1 \times \cdots \times p_M}$, $\boldsymbol{\Sigma}_m \in \mathbb{R}^{p_m \times p_m}$, and $Z_{\mathcal{J}} \sim N(0, 1)$ independently, we say that $\mathbf{X}$ follows the tensor normal distribution. We often use the shorthand notation $\mathbf{X} \sim TN(\boldsymbol{\mu}; \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M)$. Because $\mathrm{vec}(\mathbf{X}) = \mathrm{vec}([\![\mathbf{Z}; \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M]\!]) = (\bigotimes_{m=M}^{m=1} \boldsymbol{\Sigma}_m)\mathrm{vec}(\mathbf{Z})$, we have that $\mathbf{X} \sim TN(\boldsymbol{\mu}; \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M)$ if $\mathrm{vec}(\mathbf{X}) \sim N(\mathrm{vec}(\boldsymbol{\mu}), \bigotimes_{m=M}^{m=1} \boldsymbol{\Sigma}_m)$. The parameters $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M$ are only identifiable up to M rescaling constants. For example, for any set of positive constants $g_1, \ldots, g_M$ such that $\prod_{m=1}^{M} g_m = 1$, we have $\bigotimes_{m=M}^{m=1} (g_m \boldsymbol{\Sigma}_m) = \bigotimes_{m=M}^{m=1} \boldsymbol{\Sigma}_m$. It is then easy to verify that $TN(\boldsymbol{\mu}; g_1\boldsymbol{\Sigma}_1, \ldots, g_M\boldsymbol{\Sigma}_M)$ is the same distribution as $TN(\boldsymbol{\mu}; \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M)$.

We next briefly review the Gaussian mixture model (GMM, Banfield & Raftery 1993). The

GMM with shared covariance assumes that observations $\mathbf{U}_i \in \mathbb{R}^p$, $i = 1, \ldots, n$, are independent and identically distributed (i.i.d.) with the mixture normal distribution $\sum_{k=1}^{K} \pi_k^* N(\boldsymbol{\phi}_k^*, \boldsymbol{\Psi}^*)$, where K is a positive integer, $\pi_k^* \in (0, 1)$ is the prior probability for the k-th cluster, $\boldsymbol{\phi}_k^* \in \mathbb{R}^p$ is the cluster mean within the k-th cluster, and the symmetric positive definite matrix $\boldsymbol{\Psi}^* \in \mathbb{R}^{p \times p}$ is the within-cluster covariance. We note that the within-cluster covariance could be different across clusters. But we choose to present GMM with constant within-cluster covariance, because it is more closely related to our study. The latent cluster representation of the GMM is often used to connect it with discriminant analysis, optimal clustering rules, and the EM algorithm. Specifically, the GMM can be written equivalently as

$$\Pr(Y_i = k) = \pi_k^*, \quad \mathbf{U}_i \mid (Y_i = k) \sim N(\boldsymbol{\phi}_k^*, \boldsymbol{\Psi}^*), \tag{2.1}$$

where the latent variables $Y_i \in \{1, \ldots, K\}$. We use the superscript * to denote the true value of a parameter in population.

2.2 The Tensor Normal Mixture Model

Consider independent tensor-variate observations $\mathbf{X}_i \in \mathbb{R}^{p_1 \times \cdots \times p_M}$, $i = 1, \ldots, n$. The observations are heterogeneous in that they are drawn from K clusters, but the cluster labels are unavailable to us. To recover these labels, we assume that $\mathbf{X}_i$ follows a mixture of tensor normal (TN) distributions (cf. Section 2.1) such that,

$$\mathbf{X}_i \sim \sum_{k=1}^{K} \pi_k^* TN(\boldsymbol{\mu}_k^*; \boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*), \quad i = 1, \ldots, n, \tag{2.2}$$

where $\boldsymbol{\mu}_k^* \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ is the mean of the k-th cluster, $\boldsymbol{\Sigma}_m^* \in \mathbb{R}^{p_m \times p_m}$ is the common within-class covariance along mode m, and $0 < \pi_k^* < 1$ is the prior probability for $\mathbf{X}_i$ to be in the k-th cluster such that $\sum_{k=1}^{K} \pi_k^* = 1$. Throughout the rest of this paper, we use $\sigma_{m,ij}^*$ to denote the (i, j)-th entry in $\boldsymbol{\Sigma}_m^*$. To ensure the identifiability of the covariance matrices, we assume that $\sigma_{m,11}^* = 1$ for $m > 1$, and $\sigma_{1,11}^*$ is the variance of $X_{i,1\cdots1}$ within clusters, where $X_{i,1\cdots1}$ is the $(1, \cdots, 1)$-th element in the tensor $\mathbf{X}_i$. We will explicitly specify the scale of $\boldsymbol{\Sigma}_1^*$ shortly. We refer to the model (2.2) as the tensor normal mixture model (TNMM).

Parallel to the latent variable representation in GMM, (2.1), we introduce the latent cluster

membership $Y_i \in \{1, \ldots, K\}$ and re-write (2.2) as

$$\Pr(Y_i = k) = \pi_k^*, \quad \mathbf{X}_i \mid (Y_i = k) \sim TN(\boldsymbol{\mu}_k^*; \boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*). \tag{2.3}$$

Intuitively, the TNMM assumes that $\mathbf{X}_i$ follows a tensor normal distribution with mean $\boldsymbol{\mu}_k^*$ within the k-th cluster. The parameter $\boldsymbol{\mu}_k^*$ represents the centroid of the k-th cluster, while the covariance matrices $\boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*$ determine the dependence structure among the features. Also, with the latent variable Y_i, it is easy to specify the scale of $\boldsymbol{\Sigma}_1^*$. Since $\sigma_{m,11}^* = 1$ for all $m > 1$, we must have that $\sigma_{1,11}^* = \mathrm{var}(X_{i,1\cdots1} \mid Y_i = k)$ for all i, k.

To better understand the TNMM, we consider its implication on $\mathrm{vec}(\mathbf{X}_i)$. By vectorizing the data, the model is equivalent to

$$\Pr(Y_i = k) = \pi_k^*, \quad \mathrm{vec}(\mathbf{X}_i) \mid (Y_i = k) \sim N(\mathrm{vec}(\boldsymbol{\mu}_k^*), \bigotimes_{m=M}^{1} \boldsymbol{\Sigma}_m^*), \tag{2.4}$$

which resembles the GMM in (2.1). A major distinction arises from our parsimonious parametrization of the covariance. It is easy to see that, if we ignore the tensor structure and impose GMM on $\mathrm{vec}(\mathbf{X}_i)$, the covariance has $O(\prod_{i=1}^M p_m^2)$ parameters. However, the covariance in (2.4) is determined by $O(\sum_{m=1}^M p_m^2)$ parameters, because of the separable Kronecker product structure in the tensor covariance. The reduction in the number of parameters is drastic even for moderately high dimensions, and improves estimation efficiency, especially when the sample size is small.

Note that the vectorization is only for demonstration purpose. In our estimation algorithm to be introduced, we never vectorize the observations; instead, we preserve the tensor form and use tensor operators for efficient implementation. When it comes to methodology developments and computation, the tensor form also greatly reduces the storage and computation costs in the DEEM algorithm. See Section 3.5 for details.

Many existing methods for tensor data analysis employ the tensor normal assumption in seek of parsimony and simplicity in likelihood-based procedure (Hoff 2011, Fosdick & Hoff 2014, Li & Zhang 2017, Pan et al. 2019). Such an assumption has demonstrated success in regression and classification problems, which motivates the application of TNMM to unsupervised tensor learning. On the other hand, although it is known in low dimensions that modeling the dependence benefits clustering, many high-dimensional clustering methods ignore the correlation structure in data when performing variable selection and dimension reduction. It makes intuitive sense that

modeling the correlation continues to improve clustering accuracy in high dimensions, but careful investigation further reveals that correlations heavily impact the variable selection as well. We explain this point in the next section.

2.3 Optimal clustering rule and variable selection

Many methods in the literature ignore the correlations among features when performing clustering in high dimensions. Some of them are developed based on the K-means clustering, and hence make no attempt to model the correlations (Witten & Tibshirani 2010, Sun & Li 2018, Cao et al. 2013, e.g); others assume that the features are independent within each cluster and thus eliminate the need to model the correlations (Pan & Shen 2007, Guo et al. 2010, e.g). For simplicity, we refer to methods that ignore the correlation structure among features as independence methods. We demonstrate the impacts of correlations on variable selection by comparing the target clustering rule of independence methods to the optimal clustering rule under the TNMM (2.2).

Consider $\mathbf{X}$ with conditional probability density function f_k within the k-th cluster. The optimal classification rule defined on the population level is

$$\widehat{Y}^{opt} = \phi^{Bayes}(\mathbf{X}) = \arg\max_k \pi_k^* f_k(\mathbf{X}), \tag{2.5}$$

where π_k^* is the marginal probability for $\mathbf{X}$ to belong to the k-th cluster. Although the above rule in (2.5) is commonly known as the Bayes rule for classification, it continues to be optimal for clustering.

First of all, we define the clustering error of a population (non-stochastic) classifier ϕ as $\min_\Pi \Pr(\phi(\mathbf{X}) \neq \Pi(Y))$, where we optimize over all possible permutations of the K labels $\Pi : \{1, \ldots, K\} \mapsto \{1, \ldots, K\}$. A major difference between classification and clustering problems is that the K labels are well-defined in classification but are artificially created in clustering. As a result of this completely latent and non-identifiable cluster labels, any clustering rule $\phi(\mathbf{X}) : \mathbb{R}^{p_1 \times \cdots \times p_M} \mapsto \{1, \ldots, K\}$ is equivalent to the permuted $\Pi\{\phi(\mathbf{X})\}$.

It is well-known that the Bayes rule $\phi^{Bayes}(\mathbf{X})$ minimizes classification error. Because $\phi^{Bayes}(\mathbf{X})$ produces a prediction that is solely based on $\mathbf{X}$ regardless of whether we observe Y or not, it also minimizes the clustering error. The rule defined in (2.5) is thus optimal and is the target of our analysis. In estimation, the additional permutation operator needs to be carefully accounted for,

making the clustering analysis much more challenging than classification.

Recall that $X_{\mathcal{J}}$ is the $\mathcal{J}$-th element of $\mathbf{X}$, where $\mathcal{J} = (j_1, \ldots, j_M)$. For ease of presentation, we consider the special case of $K = 2$ and $\mathrm{diag}(\mathbf{\Sigma}_m^*) = 1$ for all m throughout the rest of this section. Under the TNMM (2.2), the optimal rule (2.5) is equivalent to assigning $\mathbf{X}$ to Cluster 2 if and only if

$$\log(\pi_2^*/\pi_1^*) + \langle \mathbf{X} - \frac{\boldsymbol{\mu}_1^* + \boldsymbol{\mu}_2^*}{2}, \mathbf{B}^* \rangle > 0, \tag{2.6}$$

where $\mathbf{B}^* = [\![\boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^*; (\mathbf{\Sigma}_1^*)^{-1}, \ldots, (\mathbf{\Sigma}_M^*)^{-1}]\!]$. Consequently, an element $X_{\mathcal{J}}$ is not important for clustering if and only if $b_{\mathcal{J}}^* = 0$. To achieve optimal clustering, we only need the variables in $\mathcal{D} = \{\mathcal{J} : b_{\mathcal{J}}^* \neq 0\}$.

However, if we treat the variables as independent within each cluster, e.g. as in many existing high-dimensional clustering methods, it is equivalent to assuming that $\mathbf{\Sigma}_m^*$ are all identity matrices under the TNMM. Then (2.5) leads to the following "independence rule":

$$\widehat{Y}^{indep} = \arg\min_k \{-2\log\pi_k^* + \sum_{\mathcal{J}} (X_{\mathcal{J}} - \mu_{k,\mathcal{J}}^*)^2\}. \tag{2.7}$$

That is, $\widehat{Y}^{indep} = 2$ if and only if $\log(\pi_2^*/\pi_1^*) + \langle \mathbf{X} - \frac{\boldsymbol{\mu}_1^* + \boldsymbol{\mu}_2^*}{2}, \boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^* \rangle > 0$. Hence, the variable selection of the independence methods essentially targets at the set $\mathcal{A} = \{\mathcal{J} : \mu_{1,\mathcal{J}}^* \neq \mu_{2,\mathcal{J}}^*\}$.

It can be seen that the optimal rule in (2.6) is usually different from the independence rule, because $\mathbf{B}^* \neq \boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^*$ in general. Consequently, the independence methods can not achieve the optimal error rate when the covariance matrices $\mathbf{\Sigma}_m^*$ are not diagonal. Moreover, the difference between $\mathcal{A}$ and $\mathcal{D}$ implies that the correlation structure also impacts the variable selection results. Since $\mathbf{B}^*$ is a product between $\boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^*$ and $(\mathbf{\Sigma}_m^*)^{-1}, m = 1, \ldots, M$, elements with constant means across clusters (i.e, elements in $\mathcal{A}^c$) could still improve clustering accuracy if they are correlated with $\mathbf{X}_{\mathcal{A}}$. In contrast, elements in $\mathcal{A}$ are not necessarily relevant for clustering, because their corresponding $b_{\mathcal{J}}^*$ could be zero. In Section A of the Supplementary Materials, we construct examples to illustrate this phenomenon. A similar discussion is available in Mai et al. (2012) for discriminant analysis on vector data. But, to the best of our knowledge, we are the first to discuss this point for clustering on tensor data.

Similar to the independence rule, K-means methods may also suffer from ignoring the correlations. Although (sparse) K-means can be viewed as model-free clustering methods, their target set for variable selection is similar to $\mathcal{A}$. K-means clustering (see Equation (14.33) in Friedman et al.

2001) searches for $\arg\min_{\{Y_i\}_{i=1}^n, \{\mu_k\}_{k=1}^K} \sum_{k=1}^K n_k \sum_{Y_i=k} \sum_{\mathcal{J}} (X_{i,\mathcal{J}} - \mu_{k,\mathcal{J}})^2$, where n_k is the size of the k-th cluster. Hence, if a feature has constant mean across clusters, it is not important in the final clustering. We only need the set of variables with different means, which resembles $\mathcal{A}$.

3 The doubly-enhanced EM algorithm

We develop a general estimation procedure for TNMM (2.2) with $K \geq 2$, where we assume that K is known. The clustering rule is directly obtained by plugging in the estimates of model parameters to the (population) optimal rule (2.6). We first describe the standard EM algorithm in Section 3.1. We further discuss the limitations of the standard EM that render it unsuitable for high-dimensional tensor clustering. Then we proceed to develop our DEEM algorithm and discuss its characteristics.

3.1 The standard EM algorithm

The EM algorithm (Dempster et al. 1977) is widely used in model-based clustering. Although we argue that the standard EM is not suitable for high-dimensional tensor clustering, it is nevertheless an inspiration of our DEEM algorithm and applicable in low-dimensional settings. We discuss the standard EM algorithm in what follows.

Define $\boldsymbol{\theta} = \{\pi_k, \boldsymbol{\mu}_k, k = 1, \ldots, K; \boldsymbol{\Sigma}_m, m = 1, \ldots, M\}$ as the model parameters in TNMM. Let $f(y, \mathbf{x}; \boldsymbol{\theta})$ denote the joint probability function of Y and $\mathbf{X}$. If we could observe the latent variables $\{Y_i\}_{i=1}^n$, then the log-likelihood function for the complete data is

$$l_n(\boldsymbol{\theta}) = \sum_{i=1}^n \log f(Y_i, \mathbf{X}_i; \boldsymbol{\theta}) = \sum_{i=1}^n \{\log \pi_{Y_i} + \log f_{Y_i}(\mathbf{X}_i; \boldsymbol{\theta})\}, \tag{3.1}$$

where $f_{Y_i}(\mathbf{X}_i; \boldsymbol{\theta})$ is the conditional density function of $\mathbf{X}_i \mid Y_i$. From the tensor normal distribution, we have

$$f_k(\mathbf{X}_i; \boldsymbol{\theta}) = \frac{\exp(-\frac{1}{2}\langle [\![\mathbf{X}_i - \boldsymbol{\mu}_k; \boldsymbol{\Sigma}_1^{-1}, \ldots, \boldsymbol{\Sigma}_M^{-1}]\!], \mathbf{X}_i - \boldsymbol{\mu}_k \rangle)}{(2\pi)^{p/2} |\boldsymbol{\Sigma}_1|^{q_1/2} \cdots |\boldsymbol{\Sigma}_M|^{q_M/2}}, \tag{3.2}$$

where $p = \prod_{m=1}^M p_m$ and $q_m = p/p_m$.

Clearly, the latent variables $\{Y_i\}_{i=1}^n$ are unobservable and thus we cannot directly maximize the log-likelihood function (3.1) to obtain the estimator of $\boldsymbol{\theta}$. The EM algorithm tries to maximize $l_n(\boldsymbol{\theta})$ by iteratively performing the Expectation-step (E-step) and the Maximization-step (M-step).

Consider the $(t+1)$-th iteration with the current value $\widetilde{\boldsymbol{\theta}}^{(t)}$. In the E-step, we evaluate

$$Q_n(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}^{(t)}) = \mathrm{E}\left[l_n(\boldsymbol{\theta}) \mid \{\mathbf{X}_i\}_{i=1}^n, \widetilde{\boldsymbol{\theta}}^{(t)}\right] = \sum_{i=1}^{n}\sum_{k=1}^{K} \widetilde{\xi}_{ik}^{(t)}\{\log \pi_k + \log f_k(\mathbf{X}_i; \boldsymbol{\theta})\}, \tag{3.3}$$

where

$$\widetilde{\xi}_{ik}^{(t)} = \Pr(Y_i = k \mid \mathbf{X}_i, \widetilde{\boldsymbol{\theta}}^{(t)}) = \frac{\widetilde{\pi}_k^{(t)} f_k(\mathbf{X}_i; \widetilde{\boldsymbol{\theta}}^{(t)})}{\sum_{j=1}^{K} \widetilde{\pi}_j^{(t)} f_j(\mathbf{X}_i; \widetilde{\boldsymbol{\theta}}^{(t)})}. \tag{3.4}$$

In the M-step, we maximize $Q_n(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}^{(t)})$ over $\boldsymbol{\theta}$. The updates for π_k and $\boldsymbol{\mu}_k$ can be easily computed with an explicit form. However, the updates for $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M$ are much more difficult to obtain. With some calculation, we have the following lemma.

Lemma 1. *The maximizers of* (3.3) *must satisfy*

$$\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)} = (nq_m)^{-1} \sum_{i=1}^{n}\sum_{k=1}^{K} \widetilde{\xi}_{ik}^{(t+1)} \{\widetilde{\mathbf{W}}_{ik}^{(t+1)}\}\{\widetilde{\mathbf{W}}_{ik}^{(t+1)}\}^{\intercal}, \tag{3.5}$$

where $\widetilde{\mathbf{W}}_{ik}^{(t+1)}$ *is the mode-*m *matricization of the product*

$$[\![\mathbf{X}_i - \widetilde{\boldsymbol{\mu}}_k^{(t+1)}; \{\widetilde{\boldsymbol{\Sigma}}_1^{(t+1)}\}^{-\frac{1}{2}}, \ldots, \{\widetilde{\boldsymbol{\Sigma}}_{m-1}^{(t+1)}\}^{-\frac{1}{2}}, \mathbf{I}_{p_m}, \{\widetilde{\boldsymbol{\Sigma}}_{m+1}^{(t+1)}\}^{-\frac{1}{2}}, \ldots, \{\widetilde{\boldsymbol{\Sigma}}_M^{(t+1)}\}^{-\frac{1}{2}}]\!]. \tag{3.6}$$

Lemma 1 implies that an iterative algorithm is needed to find $\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)}$. Since $\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)}$ depends on all the other covariance estimates $\widetilde{\boldsymbol{\Sigma}}_{m'}^{(t+1)}, m' \neq m$, we need to update one covariance estimate while keeping all the others fixed until convergence to find $\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)}$. By letting $K = 1$, the results in Lemma 1 also reproduce the maximum likelihood estimation in the tensor normal distribution (e.g. Dutilleul 1999, Manceur & Dutilleul 2013).

The standard EM algorithm has several noticeable issues in our problem of interest. In the E-step, we use all the elements in $\mathbf{X}_i$ to calculate $\widetilde{\xi}_{ik}^{(t)}$. Even for a tensor of dimension $p_m = 10$, $m = 1, 2, 3$, we have one thousand variables and are thus dealing with a high-dimensional estimation problem. Even when Y_i's are all observed, we can do no better than random guessing if we estimate an excessive number of parameters without variable selection (Bickel & Levina 2004, Fan & Fan 2008), because the accumulated estimation errors would dominate the signal in the data. Now that Y_i's are unobservable, variable selection should be more critical in order to reduce the number of parameters. Since the standard EM algorithm unfortunately does not enforce variable selection, it is prone to inaccurate clustering on tensor data, which are often high-dimensional (i.e. $p = \prod_{m=1}^{M} p_m > n$).

In the M-step, an iterative sub-algorithm is needed to maximize $Q_n(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}^{(t)})$ over $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M$. This sub-algorithm drastically adds to the computation cost. Moreover, the consistency for $\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)}$ cannot be easily established in high dimensions. To the best of our knowledge, the most related result is Lyu et al. (2019). They considered the estimation of $(\boldsymbol{\Sigma}_m^*)^{-1}$ under the tensor graphical model where all observations come from the same tensor normal distribution. They had to assume that $(\boldsymbol{\Sigma}_m^*)^{-1}$ are all sparse and constructed penalized estimates to achieve consistency in high dimensions. The sparsity assumption on the precision matrix is central to their proof. However, in the context of clustering, our goal is to recover Y_i's. The covariances are nuisance parameters for this purpose, as the optimal clustering rule in (2.6) does not depend on $\boldsymbol{\Sigma}_m^*$ when we know $\mathbf{B}^*$. Therefore, it is generally more desirable to not impose additional assumptions on $\boldsymbol{\Sigma}_m^*$ so that we can handle arbitrary correlation structure while achieving the optimal clustering error. However, the consistency is very difficult to show for the unpenalized estimate $\widetilde{\boldsymbol{\Sigma}}_m^{(t+1)}$. We need innovative modifications to the M-step to lower the computation cost and achieve theoretical guarantee.

Motivated by the above issues, we propose the doubly-enhanced EM algorithm (DEEM) that greatly improves both the E-step and the M-step in the standard EM algorithm. DEEM consists of iterations between an enhanced E-step and an enhanced M-step. In the enhanced E-step, we impose variable selection to evaluate the Q-function more accurately, while in the enhanced M-step we find better estimates for the covariances. We discuss these two steps in Sections 3.2 & 3.3, respectively. The complete DEEM algorithm is summarized in Section 3.4. Later in our simulation studies in Section 5.1, we confirm that the standard EM algorithm has inferior performance to DEEM.

3.2 The enhanced E-step

To distinguish from the standard EM estimates $\widetilde{\boldsymbol{\theta}}$, we denote $\widehat{\boldsymbol{\theta}}^{(t)}$ as the DEEM estimate of $\boldsymbol{\theta}$ at the t-th iteration. Given $\widehat{\boldsymbol{\theta}}^{(t)}$, we consider the $(t+1)$-th iteration.

In the enhanced E-step, we obtain a more accurate evaluation of the Q-function in (3.3). Obviously, it suffices to estimate $\xi_{ik}^{(t+1)} = \Pr(Y_i = k \mid \mathbf{X}_i, \widehat{\boldsymbol{\theta}}^{(t)})$. As discussed in Section 3.1, estimates of $\xi_{ik}^{(t+1)}$ could contain large estimation error without variable selection. To resolve this issue, we assume that our target $\xi_{ik} = \Pr(Y_i = k \mid \mathbf{X}_i, \boldsymbol{\theta}^*)$ is determined by a subset of elements in $\mathbf{X}$ and hence can be evaluated with a reduced number of parameters. Let

$$\mathbf{B}_k^* = [\![\boldsymbol{\mu}_k^* - \boldsymbol{\mu}_1^*; (\boldsymbol{\Sigma}_1^*)^{-1}, \ldots, (\boldsymbol{\Sigma}_M^*)^{-1}]\!] \in \mathbb{R}^{p_1 \times \cdots \times p_M}, k = 2, \ldots, K. \tag{3.7}$$

The following lemma helps clarify the implication of the sparsity assumption.

Lemma 2. *Suppose that* $\mathbf{X}_i$ *follows the TNMM* (2.2)*. We have that*

$$\xi_{i1} = \frac{\pi_1^*}{\pi_1^* + \sum_{k=2}^K \pi_k^* \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\boldsymbol{\mu}_k^* + \boldsymbol{\mu}_1^*), \mathbf{B}_k^* \rangle]}, \tag{3.8}$$

$$\xi_{ik} = \frac{\pi_k^* \exp[\langle \mathbf{X}_i - \frac{1}{2}(\boldsymbol{\mu}_k^* + \boldsymbol{\mu}_1^*), \mathbf{B}_k^* \rangle]}{\pi_1^* + \sum_{k=2}^K \pi_k^* \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\boldsymbol{\mu}_k^* + \boldsymbol{\mu}_1^*), \mathbf{B}_k^* \rangle]}, \quad k > 1. \tag{3.9}$$

Lemma 2 shows that each ξ_{ik} is determined by the inner products $\langle \mathbf{X}_i - \frac{1}{2}(\boldsymbol{\mu}_j^* + \boldsymbol{\mu}_1^*), \mathbf{B}_j^* \rangle, j = 2, \ldots, K$. Hence, $X_{\mathcal{J}}$ is not important for the E-step if and only if

$$b_{2,\mathcal{J}}^* = \cdots = b_{K,\mathcal{J}}^* = 0. \tag{3.10}$$

Then the sparsity assumption implies that (3.10) holds for most $\mathcal{J}$. In other words, let $\mathcal{D}$ denote the index set of the important variables, i.e. $\mathcal{D}^c = \{\mathcal{J} : b_{2,\mathcal{J}}^* = \cdots = b_{K,\mathcal{J}}^* = 0\}$. The sparsity assumption states that $|\mathcal{D}| \ll \prod_{m=1}^M p_m$. It is worth noting that this assumption is equivalent to assuming that the optimal clustering rule is sparse. By (2.5), the optimal rule under TNMM is

$$\widehat{Y}^{opt} = \arg\max_k \{\log \pi_k^* + \langle \mathbf{X} - (\boldsymbol{\mu}_1^* + \boldsymbol{\mu}_k^*)/2, \mathbf{B}_k^* \rangle\}, \tag{3.11}$$

where $\mathbf{B}_1^* = 0$. Hence, the sparsity in the optimal rule concurs with our assumption on $\mathcal{D}$, where variable selection assists in achieving the lowest clustering error possible.

Note that $\mathbf{B}_k^* = [\![\boldsymbol{\mu}_k^* - \boldsymbol{\mu}_1^*; (\boldsymbol{\Sigma}_1^*)^{-1}, \ldots, (\boldsymbol{\Sigma}_M^*)^{-1}]\!]$ by definition. It follows that

$$(\mathbf{B}_2^*, \ldots, \mathbf{B}_K^*) = \underset{\mathbf{B}_2, \ldots, \mathbf{B}_K \in \mathbb{R}^{p_1 \times \cdots p_M}}{\operatorname{argmin}} \left[\sum_{k=2}^K (\langle \mathbf{B}_k, [\![\mathbf{B}_k, \boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*]\!] \rangle - 2\langle \mathbf{B}_k, \boldsymbol{\mu}_k^* - \boldsymbol{\mu}_1^* \rangle) \right]. \tag{3.12}$$

To obtain sparse estimates for $\mathbf{B}_k^*$, we plug in our current estimates for $\boldsymbol{\Sigma}_m^*$ and $\boldsymbol{\mu}_k^*$, and add the group lasso penalty (Yuan & Lin 2006) to encourage the sparsity pattern in (3.10). More specifically, we let $(\widehat{\mathbf{B}}_2^{(t+1)}, \ldots, \widehat{\mathbf{B}}_K^{(t+1)})$ be the solution to the following minimization problem,

$$\min_{\mathbf{B}_2, \ldots, \mathbf{B}_K} \left[\sum_{k=2}^K (\langle \mathbf{B}_k, [\![\mathbf{B}_k, \widehat{\boldsymbol{\Sigma}}_1^{(t)}, \ldots, \widehat{\boldsymbol{\Sigma}}_M^{(t)}]\!] \rangle - 2\langle \mathbf{B}_k, \widehat{\boldsymbol{\mu}}_k^{(t)} - \widehat{\boldsymbol{\mu}}_1^{(t)} \rangle) + \lambda^{(t+1)} \sum_{\mathcal{J}} \sqrt{\sum_{k=2}^K b_{k,\mathcal{J}}^2} \right], \tag{3.13}$$

where $\lambda^{(t+1)} > 0$ is a tuning parameter. The optimization problem in (3.13) is convex and can be easily solved by a blockwise coordinate descent algorithm similar to that in Pan et al. (2019). See Algorithm S.2 in Section B in Supplementary Materials for details.

After obtaining $\{\widehat{\mathbf{B}}_2^{(t+1)}, \ldots, \widehat{\mathbf{B}}_K^{(t+1)}\}$, we calculate

$$\widehat{\xi}_{i1}^{(t+1)} = \frac{\widehat{\pi}_1^{(t)}}{\widehat{\pi}_1^{(t)} + \sum_{k=2}^K \widehat{\pi}_k^{(t)} \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_k^{(t)} + \widehat{\boldsymbol{\mu}}_1^{(t)}), \widehat{\mathbf{B}}_k^{(t+1)} \rangle]}, \tag{3.14}$$

$$\widehat{\xi}_{ik}^{(t+1)} = \frac{\widehat{\pi}_k^{(t)} \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_k^{(t)} + \widehat{\boldsymbol{\mu}}_1^{(t)}), \widehat{\mathbf{B}}_k^{(t+1)} \rangle]}{\widehat{\pi}_1^{(t)} + \sum_{k=2}^K \widehat{\pi}_k^{(t)} \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_k^{(t)} + \widehat{\boldsymbol{\mu}}_1^{(t)}), \widehat{\mathbf{B}}_k^{(t+1)} \rangle]}, \quad k > 1. \tag{3.15}$$

Combining $\widehat{\xi}_{ik}^{(t+1)}$ with (3.2) and (3.3), we have the Q-function in the $(t+1)$-th iteration as

$$Q^{\text{DEEM}}(\boldsymbol{\theta} \mid \widehat{\boldsymbol{\theta}}^{(t)}) = \sum_{i=1}^n \sum_{k=1}^K \widehat{\xi}_{ik}^{(t+1)} \{\log \pi_k - (\sum_{m=1}^M q_m \log|\boldsymbol{\Sigma}_m|) - \frac{1}{2}\langle [\![\mathbf{X}_i - \boldsymbol{\mu}_k; \boldsymbol{\Sigma}_1^{-1}, \ldots, \boldsymbol{\Sigma}_M^{-1}]\!], \mathbf{X}_i - \boldsymbol{\mu}_k \rangle\}. \tag{3.16}$$

The Q-function in (3.16) will guide us to find $\widehat{\boldsymbol{\theta}}^{(t+1)}$ in the enhanced M-step, which will be discussed in Section 3.3. Since the probabilities $\widehat{\xi}_{ik}^{(t+1)}$ in (3.16) are calculated based on a small subset of variables, they are expected to be close to the truth under the sparsity model assumption, and lay the foundation for accurate parameter estimation in the enhanced M-step.

3.3 The enhanced M-step

In the enhanced M-step, we update estimates for π_k^*, $\boldsymbol{\mu}_k^*$ and $\boldsymbol{\Sigma}_m^*$. By maximizing the Q-function in (3.16), it is straightforward to obtain the estimates for π_k^*, $\boldsymbol{\mu}_k^*$ at the $(t+1)$-th iteration as $\widehat{\pi}_k^{(t+1)} = \sum_{i=1}^n \widehat{\xi}_{ik}^{(t+1)}/n$ and $\widehat{\boldsymbol{\mu}}_k^{(t+1)} = \sum_{i=1}^n \widehat{\xi}_{ik}^{(t+1)} \mathbf{X}_i / \sum_{i=1}^n \widehat{\xi}_{ik}^{(t+1)}$, $k = 1, \ldots, K$. It is also easy to verify that $\sum_{k=1}^K \widehat{\pi}_k^{(t+1)} = \sum_{k=1}^K \widehat{\xi}_{ik}^{(t+1)} = 1$.

As discussed in Section 3.1, directly maximizing the Q-function over $\boldsymbol{\Sigma}_m$ is not ideal. We consider an alternative update for $\boldsymbol{\Sigma}_m$ based on the following result. Recall that $q_m = p_m^{-1} \prod_{h=1}^M p_h$ and $\xi_{ik} = \Pr(Y_i = k \mid \mathbf{X}_i, \boldsymbol{\theta}^*)$.

Lemma 3. *Under the TNMM in* (2.2)*, we have*

$$\boldsymbol{\Sigma}_m^* \propto \frac{1}{q_m} \mathrm{E}\left\{ \sum_{k=1}^K \xi_{ik} (\mathbf{X}_i - \boldsymbol{\mu}_k^*)_{(m)} (\mathbf{X}_i - \boldsymbol{\mu}_k^*)_{(m)}^\intercal \right\}. \tag{3.17}$$

Lemma 3 implies that we can construct a method of moment estimate for $\boldsymbol{\Sigma}_m^*$. Recall that we require $\sigma_{m,11}^* = 1$ for $m > 1$ to ensure identifiable covariance matrices and hence have $\sigma_{1,11}^* = \mathrm{var}(X_{i,1\cdots1} \mid Y_i = k)$, where $X_{i,1\cdots1}$ is the $(1, \ldots, 1)$-th element of $\mathbf{X}_i$. Since we have shown that $\boldsymbol{\Sigma}_m^*$ is proportional to the right hand side of (3.17), we can incorporate the identification

constraints into scaling. Note that Lemma 3 is widely applicable and can be combined with any other identification constraints on $\mathbf{\Sigma}_m^*$, e.g. requiring $\|\mathbf{\Sigma}_2^*\|_F = \cdots = \|\mathbf{\Sigma}_M^*\|_F = 1$.

As a consequence of Lemma 3, we propose the following non-iterative estimator for the covariance parameters in the enhanced M-step. Given $\widehat{\xi}_{ik}^{(t+1)}$, we first compute intermediate estimates,

$$\breve{\mathbf{\Sigma}}_m^{(t+1)} = \frac{1}{nq_m} \sum_{i=1}^{n} \sum_{k=1}^{K} \widehat{\xi}_{ik}^{(t+1)} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(t+1)})_{(m)} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(t+1)})_{(m)}^\intercal, \quad m = 1, \ldots, M. \tag{3.18}$$

Then, for $m > 1$, our DEEM estimator is $\widehat{\mathbf{\Sigma}}_m^{(t+1)} = \breve{\mathbf{\Sigma}}_m^{(t+1)} / \breve{\sigma}_{m,11}^{(t+1)}$; and for $m = 1$, our DEEM estimator is $\widehat{\mathbf{\Sigma}}_m^{(t+1)} = \{\widehat{\sigma}_{1,11}^{(t+1)} / \breve{\sigma}_{1,11}^{(t+1)}\} \breve{\mathbf{\Sigma}}_1^{(t+1)}$, where the conditional variance of the element $X_{i,1\cdots1}$ is estimated as

$$\widehat{\sigma}_{1,11}^{(t+1)} = \widehat{\mathrm{E}}\{\widehat{\mathrm{var}}(X_{i,1\cdots1} \mid Y_i)\} = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} \widehat{\xi}_{ik}^{(t+1)} (X_{i,1\cdots1} - \widehat{\mu}_{k,1\cdots1}^{(t+1)})^2. \tag{3.19}$$

The covariance estimates $\widehat{\mathbf{\Sigma}}_m^{(t+1)}$ will be used in the subsequent enhanced E-step. In comparison to the estimator $\widetilde{\mathbf{\Sigma}}_m^{(t+1)}$ in the standard EM algorithm, $\widehat{\mathbf{\Sigma}}_m^{(t+1)}$ has apparent computational advantages. No iterative sub-algorithm is needed for computing $\widehat{\mathbf{\Sigma}}_m^{(t+1)}$. Instead, all the computation in the enhanced M-step can be carried out explicitly. Moreover, we will later show that $\widehat{\mathbf{\Sigma}}_m^{(t+1)}$ leads to consistent clustering even when the dimension of each mode of the tensor grows at an exponential rate of n without any sparsity assumption on $\mathbf{\Sigma}_m^*$. It is unclear whether such consistency can be achieved by the standard estimator $\widetilde{\mathbf{\Sigma}}_m^{(t+1)}$. Hence, $\widehat{\mathbf{\Sigma}}_m^{(t+1)}$ should be preferred over $\widetilde{\mathbf{\Sigma}}_m^{(t+1)}$ for theoretical considerations as well.

The enhanced M-step in DEEM is delicately designed, but our covariance estimator has a potentially much wider range of applications beyond DEEM. For example, Cao et al. (2013), Sun & Li (2018) considered combining low-rank decomposition of the tensors and the K-means clustering; Gao et al. (2021) proposed to regularize the mean differences of matrix observations. To fill the gap between these works and the optimal clustering rule, which requires covariance modeling, one can potentially adopt our fast and theoretically guaranteed covariance estimators.

3.4 The DEEM algorithm and implementation details

With the enhanced E-step and the enhanced M-step, we iterate between them until convergence similar to the standard EM algorithm. The DEEM algorithm is summarized in Algorithm 1. Given

Algorithm 1 DEEM algorithm for tensor clustering

1. Initialize $\widehat{\pi}_k^{(0)}$, $\widehat{\boldsymbol{\mu}}_k^{(0)}$, $\widehat{\boldsymbol{\Sigma}}_m^{(0)}$.

2. For $t = 0, 1, \ldots$, repeat the following steps until convergence:

   (a) The enhanced E-step:

      i. Minimize the following convex objective function over $\mathbf{B}_2, \ldots, \mathbf{B}_K \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ with Algorithm S.2:

$$\sum_{k=2}^{K} (\langle \mathbf{B}_k, [\![\mathbf{B}_k, \widehat{\boldsymbol{\Sigma}}_1^{(t)}, \ldots, \widehat{\boldsymbol{\Sigma}}_M^{(t)}]\!] \rangle - 2\langle \mathbf{B}_k, \widehat{\boldsymbol{\mu}}_k^{(t)} - \widehat{\boldsymbol{\mu}}_1^{(t)} \rangle) + \lambda^{(t+1)} \sum_{\mathcal{J}} \sqrt{\sum_{k=2}^{K} b_{k,\mathcal{J}}^2}.$$

      Let $(\widehat{\mathbf{B}}_2^{(t+1)}, \cdots, \widehat{\mathbf{B}}_K^{(t+1)})$ denote the solution.

      ii. For $i = 1, \ldots, n$, and $k = 1, \ldots, K$, calculate the probabilities

$$\widehat{\xi}_{ik}^{(t+1)} = \begin{cases} \dfrac{\widehat{\pi}_1^{(t+1)}}{\widehat{\pi}_1^{(t+1)} + \sum_{j=2}^{K} \widehat{\pi}_j^{(t+1)} \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_j^{(t+1)} + \widehat{\boldsymbol{\mu}}_1^{(t+1)}), \widehat{\mathbf{B}}_j^{(t+1)} \rangle]}, & k = 1; \\ \dfrac{\widehat{\pi}_k^{(t+1)} \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_k^{(t+1)} + \widehat{\boldsymbol{\mu}}_1^{(t+1)}), \widehat{\mathbf{B}}_k^{(t+1)} \rangle]}{\widehat{\pi}_1^{(t+1)} + \sum_{k=2}^{K} \widehat{\pi}_j^{(t+1)} \cdot \exp[\langle \mathbf{X}_i - \frac{1}{2}(\widehat{\boldsymbol{\mu}}_j^{(t+1)} + \widehat{\boldsymbol{\mu}}_1^{(t+1)}), \widehat{\mathbf{B}}_j^{(t+1)} \rangle]}, & k > 1. \end{cases}$$

   (b) The enhanced M-step:

      i. Update $\widehat{\pi}_k^{(t+1)} = \sum_{i=1}^{n} \widehat{\xi}_{ik}^{(t+1)} / n$ and $\widehat{\boldsymbol{\mu}}_k^{(t+1)} = \sum_{i=1}^{n} \widehat{\xi}_{ik}^{(t+1)} \mathbf{X}_i / \sum_{i=1}^{n} \widehat{\xi}_{ik}^{(t+1)}$.

      ii. Compute intermediate covariance estimators

$$\breve{\boldsymbol{\Sigma}}_m^{(t+1)} = \frac{1}{n q_m} \sum_{i=1}^{n} \sum_{k=1}^{K} \widehat{\xi}_{ik}^{(t+1)} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(t+1)})_{(m)} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(t+1)})_{(m)}^{\intercal}.$$

      iii. Scale $\breve{\boldsymbol{\Sigma}}_m^{(t+1)}$ to be

$$\widehat{\boldsymbol{\Sigma}}_m^{(t+1)} = \begin{cases} \{ n^{-1} \sum_{i=1}^{n} \sum_{k=1}^{K} \widehat{\xi}_{ik}^{(t+1)} (X_{i,1\cdots1} - \widehat{\mu}_{k,1\cdots1}^{(t+1)})^2 \} \breve{\boldsymbol{\Sigma}}_m^{(t+1)} / \breve{\sigma}_{1,11}^{(t+1)}, & m = 1; \\ \breve{\boldsymbol{\Sigma}}_m^{(t+1)} / \breve{\sigma}_{m,11}^{(t+1)}, & m > 1. \end{cases}$$

3. Output $\widehat{\xi}_{ik}, \widehat{\pi}_k, \widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_m$ at convergence.

the output of $\widehat{\xi}_{ik}$, we assign $\mathbf{X}_i$ to cluster $\widehat{Y}_i^{\text{DEEM}}$, where $\widehat{Y}_i^{\text{DEEM}} = \arg\max_k \widehat{\xi}_{ik}$. We further discuss some implementation details in what follows.

Initialization. In order to implement DEEM, we need to determine the initial value. In our numerical studies, we first perform the K-means clustering on $\text{vec}(\mathbf{X}_i)$ to find $\widehat{Y}_i^{(0)}$. Then we set

the initial values as,

$$\widehat{\pi}_k^{(0)} = \frac{\sum_{i=1}^n 1(\widehat{Y}_i^{(0)} = k)}{n}, \quad \widehat{\boldsymbol{\mu}}_k^{(0)} = \frac{1}{n\widehat{\pi}_k^{(0)}} \sum_{\widehat{Y}_i^{(0)}=k} \mathbf{X}_i, \tag{3.20}$$

$$\widehat{\boldsymbol{\Sigma}}_m^{(0)} \propto \frac{1}{nq_m} \sum_{i=1}^n \sum_{\widehat{Y}_i^{(0)}=k} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(0)})_{(m)} (\mathbf{X}_i - \widehat{\boldsymbol{\mu}}_k^{(0)})_{(m)}^\intercal. \tag{3.21}$$

The scales of $\widehat{\boldsymbol{\Sigma}}_m^{(0)}$ are chosen such that $\widehat{\sigma}_{m,11}^{(0)} = 1$ for $m > 1$ and $\widehat{\sigma}_{1,11}^{(0)} = \frac{1}{n}\sum_{k=1}^K \sum_{\widehat{Y}_i^{(0)}=k} (X_{i,1\cdots 1} - \widehat{\mu}_k^{(0)})^2$. We choose to use the K-means clustering in initialization because it is very fast. Although the K-means clustering is performed on vectorized data and thus ignores the tensor structure, DEEM can recover from this loss of efficiency by incorporating the tensor structure in the later iterations. In our numerical studies, we observe that this initialization leads to good solutions of DEEM at convergence even when the K-means clustering has poor performance.

Convergence. In our implementation, the convergence criterion is based on the sum of squares of mean differences between two consecutive iterations. We stop the DEEM iterations if $\sum_k \|\widehat{\boldsymbol{\mu}}_k^{(t+1)} - \widehat{\boldsymbol{\mu}}_k^{(t)}\|_F^2 \leq 0.1$ or the maximum number of iterations $t_{\max} = 50$ is reached. In our experience, the algorithm usually converges within 50 iterations. See Section 4.4, Figure 4.1, for the number of iterations required to converge as we change the signal strength in simulations.

Tuning. The tuning parameter $\lambda^{(t)}$ in the enhanced E-step could either be fixed or varying across iterations. For computation considerations, it is apparently easier to fix $\lambda^{(t)} = \lambda$ for all t, while for theoretical considerations, one may favor varying $\lambda^{(t)}$. For example, similar to Cai et al. (2019), we could consider

$$\lambda^{(t+1)} = \kappa\lambda^{(t)} + (\frac{1-\kappa^{t+1}}{1-\kappa})C_\lambda\sqrt{\frac{\log p}{n}}, \tag{3.22}$$

where $0 < \kappa < 1/2$ and $C_\lambda > 0$ are constants. In our numerical studies, we note that both choices of $\lambda^{(t)}$ give reasonable results as long as they are properly tuned. Moreover, when t is large, the varying $\lambda^{(t)}$ in (3.22) is roughly constant at the value $C_\lambda\sqrt{\frac{\log p}{n}}$. Therefore, we fix $\lambda^{(t)} = \lambda$ in all the numerical studies, but only consider the varying $\lambda^{(t)}$ in theoretical studies.

When we fix $\lambda^{(t)} = \lambda$, we need to determine λ. Permutation (e.g., Witten & Tibshirani 2010) and Bayesian information criterion (BIC; e.g., Sun & Li 2018, Guo et al. 2010) are two popular ways for tuning in clustering. We adopt a BIC-type criterion. For any λ, we let $\widehat{\boldsymbol{\theta}}^\lambda$ be the output of

DEEM with the tuning parameter fixed at λ. We look for the value of λ that minimizes

$$\text{BIC}(\lambda) = -2\sum_{i=1}^{n}\log(\sum_{k=1}^{K}\widehat{\pi}_k^{\lambda} f_k(\mathbf{X}_i;\widehat{\boldsymbol{\theta}}_k^{\lambda})) + \log(n)\cdot|\widehat{\mathcal{D}}^{\lambda}|, \tag{3.23}$$

where $\widehat{\mathcal{D}}^{\lambda} = \{(k,\mathcal{J}) : \widehat{b}_{k,\mathcal{J}}^{\lambda} \neq 0\}$ is the set of nonzero elements in $\widehat{\mathbf{B}}_2^{\lambda},\ldots,\widehat{\mathbf{B}}_K^{\lambda}$.

Number of clusters. As most clustering methods, DEEM requires users to specify the number of clusters K, the knowledge of which is often unavailable due to the unsupervised nature of clustering problems. In this paper, we focus on the scenario that K is known. In practice, we may use a BIC-type criterion similar to (3.23) to choose λ and K simultaneously. Implementation details and simulation examples of this approach are provided in Section C of Supplementary Materials. Under simulation models M1–M5 in Section 5, the number of clusters can be identified correctly for roughly 60% to 80% of the time. There exist many proposals for the estimation of K in various clustering contexts, such as Tibshirani et al. (2001), Fraley & Raftery (2002), Sugar & James (2003), Chiang & Mirkin (2010), Wang (2010), Fang & Wang (2012), Fujita et al. (2014), Fu & Perry (2020), but consistent selection of K for high-dimensional tensor clustering is still an open question and is left as future research.

As pointed out by a referee, it has been shown in more classical settings that if K is over-specified, the convergence rate could be lower (Chen 1995, Heinrich & Kahn 2018, Dwivedi et al. 2020, e.g). These papers focus on vector models and consider dimensions much lower than what will be presented for our method. It will be an interesting but challenging future topic to know whether similar results hold for tensor clustering in high dimensions.

3.5 Benefits of keeping the tensor form

In this section, we discuss the advantages of keeping the tensor form in developing our method. We consider the enhanced E-step and the enhanced M-step separately.

Our enhanced E-step is conceptually similar to the E-step in Cai et al. (2019), where they proposed a method called CHIME for model-based clustering of high-dimensional vector data. Under the high-dimensional GMM, the authors showed that it suffices to find some linear projections of the features to conduct the E-step. To tackle the high dimensionality, they assume that the linear projections are sparse. Their sparsity assumption is similar to ours on $\mathbf{B}_k^*$.

However, CHIME is only designed for vector data, and is not tailored for tensor data. Our enhanced E-step takes advantage of the tensor structure to reduce the storage cost and improve clustering efficiency for higher-order tensor data. In particular, if we vectorize our tensor observations $\mathbf{X}_i$ and apply CHIME, in each iteration we need to compute the covariance matrix $\widehat{\mathrm{var}}(\mathrm{vec}(\mathbf{X}_i) \mid Y_i)$ with $\prod_{m=1}^{M} p_m^2$ elements. But in our enhanced E-step, the covariance matrices only have $\sum_{m=1}^{M} p_m^2$ elements, and are much lighter on the storage.

On the other hand, even though the vectorized form of TNMM in (2.4) has a reduced number of parameters, it is still advantageous to consider the original tensor form for the sake of computation. To see this subtle point, note that, if we vectorize $\mathbf{X}_i$ and the associated parameters $\boldsymbol{\beta}_k^* \equiv \mathrm{vec}(\mathbf{B}^*) = \left\{\bigotimes_{m=M}^{m=1}(\boldsymbol{\Sigma}_m^*)^{-1}\right\}\mathrm{vec}(\boldsymbol{\mu}_k^* - \boldsymbol{\mu}_1^*)$. Then the optimization problem (3.13) becomes

$$\underset{\boldsymbol{\beta}_2,\ldots,\boldsymbol{\beta}_K\in\mathbb{R}^p}{\mathrm{argmin}} \left[\sum_{k=2}^{K}\left\{\boldsymbol{\beta}_k^{\mathsf{T}}\left(\bigotimes_{m=M}^{m=1}\widehat{\boldsymbol{\Sigma}}_m^{(t)}\right)\boldsymbol{\beta}_k - 2\boldsymbol{\beta}_k^{\mathsf{T}}\mathrm{vec}\left(\widehat{\boldsymbol{\mu}}_k^{(t)} - \widehat{\boldsymbol{\mu}}_1^{(t)}\right)\right\} + \lambda^{(t+1)}\sum_j\sqrt{\sum_{k=2}^{K}\beta_{k,j}^2}\right], \tag{3.24}$$

which can be solved by a blockwise coordinate descent algorithm, such as the one in Mai et al. (2019). However, the storage and the computation costs of $\bigotimes_{m=M}^{m=1}\widehat{\boldsymbol{\Sigma}}_m^{(t)}$ are both at the intimidating order of $O(\prod_{m=1}^{M} p_m^2)$. In comparison, to solve (3.13), our efficient implementation does not require calculating the Kronecker product. Consequently, it may be practically infeasible to solve (3.24) when we can still easily solve (3.13). For example, on a simulated data set from M7 in Section 5.1, we tried to use (3.24) in the enhanced E-step when the tensor dimension is $30 \times 30 \times 30$. On a computer within 16GB of memory, the algorithm would fail due to an out-of-memory error. However, DEEM can be carried out on the same computer.

More importantly, the enhanced M-step is most naturally derived when we keep the tensor form of $\mathbf{X}_i$ rather than considering the vectorized TNMM in (2.4). With the tensor form, the covariance matrices $\boldsymbol{\Sigma}_m^*$ are more "separated" from each other. This fact enables us to find $\widehat{\boldsymbol{\Sigma}}_m^{(t)}$ individually. If we consider the vectorized version, we need to estimate $\bigotimes_{m=M}^{m=1}\boldsymbol{\Sigma}_m^*$, which is not easy without reshaping $\mathrm{vec}(\mathbf{X}_i)$ into tensors.

4 Theoretical studies

4.1 Parameter space and technical definitions

Before presenting the consistency of DEEM, we define our parameter space of interest and formally introduce some technical terms. We assume that the number of cluster is known and focus on the two-cluster case, i.e. $K = 2$.

We define our parameter space for the TNMM parameter as $\boldsymbol{\theta} = \{\pi_1, \pi_2, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M\}$, where $0 < \pi_1 = 1 - \pi_2 < 1$, $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ and $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_M \in \mathbb{R}^{p_m \times p_m}$ are all symmetric positive definite. Two important estimable functions of $\boldsymbol{\theta}$ are $\mathbf{B} = \mathbf{B}(\boldsymbol{\theta}) = [\![\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1; \boldsymbol{\Sigma}_1^{-1}, \ldots, \boldsymbol{\Sigma}_M^{-1}]\!] \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ and $\Delta = \Delta(\boldsymbol{\theta}) = \langle \boldsymbol{\mu}_2 - \boldsymbol{\mu}_1, [\![\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1; \boldsymbol{\Sigma}_1^{-1}, \ldots, \boldsymbol{\Sigma}_M^{-1}]\!] \rangle \in \mathbb{R}$. The tensor parameter $\mathbf{B}$ is used for calculating $\widehat{\xi}_{ik}$ in the enhanced E-step of DEEM. The parameter Δ is the separation between the two clusters. The true population parameters are $\boldsymbol{\theta}^*$, $\mathbf{B}^* = \mathbf{B}(\boldsymbol{\theta}^*)$ and $\Delta^* = \Delta(\boldsymbol{\theta}^*)$. The set of important variables $\mathcal{D} = \{\mathcal{J} : b_{\mathcal{J}} \neq 0\}$ is clearly also an estimable function of $\boldsymbol{\theta}$, as $\mathcal{D} = \mathcal{D}(\mathbf{B}) = \mathcal{D}(\mathbf{B}(\boldsymbol{\theta}))$.

For any two numbers a and b, we write $a \vee b = \max\{a, b\}$ and $a \wedge b = \min\{a, b\}$. We use $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ to denote the largest and the smallest eigenvalues of a matrix, respectively. We define the parameter space $\boldsymbol{\Theta} = \boldsymbol{\Theta}(c_\pi, C_b, s, \{C_m\}_{m=1}^{M=1}, \Delta_0)$ as

$$\{\boldsymbol{\theta} : \pi_k \in (c_\pi, 1 - c_\pi), \|\mathrm{vec}(\mathbf{B})\|_1 \leq C_b, \lambda_{\min}^{-1}(\boldsymbol{\Sigma}_m) \vee \lambda_{\max}(\boldsymbol{\Sigma}_m) \leq C_m, |\mathcal{D}| \leq s, \Delta \geq \Delta_0\}, \quad (4.1)$$

where $C_1, \ldots, C_M, C_b, \Delta_0 > 0$ and $0 < c_\pi < 1$ are constants that do not change as p_m increases, but $s > 0$ can vary with p_m. This parameter space is sufficiently flexible to include a wide range of models. The assumptions in $\boldsymbol{\Theta}$ are intuitive and very mild. First, we require the eigenvalues of $\boldsymbol{\Sigma}_m$ to be bounded from below and above: $C_m^{-1} \leq \lambda_{\min}(\boldsymbol{\Sigma}_m) \leq \lambda_{\max}(\boldsymbol{\Sigma}_m) \leq C_m$. This eigenvalue assumption on the covariances is also common in high dimensions (Cai & Liu 2011, Pan et al. 2019). We also require that π_k is bounded away from 0 and 1, so that each cluster has a decent sample size. The coefficient $\mathbf{B}$ is assumed to be sparse so that we can perform variable selection. Finally, the assumption that $\Delta > \Delta_0$ implies that the two clusters are well separated from each other. If two clusters are indistinguishable even on the population level, of course it will be impossible to separate them with any clustering rule. In our theory, we need Δ_0 to be sufficiently large so that we only consider models with a reasonably large separation. See more detailed discussion of the cluster separation Δ following Theorem 1.

We need some more technical definitions before we present the conditions needed for theoretical results. We set

$$\Gamma(s) = \{\mathbf{u} \in \mathbb{R}^p : 2\|\mathbf{u}_{S^C}\|_1 \leq 4\|\mathbf{u}_S\|_1 + 3\sqrt{s}\|\mathbf{u}\|_2, \text{ for some } S \subset \{1, \ldots, p\}, |S| = s\}, \quad (4.2)$$

where $\mathbf{u}_S \in \mathbb{R}^s$ and $\mathbf{u}_{S^C} \in \mathbb{R}^{p-s}$ are sub-vectors extracted from $\mathbf{u}$ based on the index set S and its complement set. The set $\Gamma(s)$ contains approximately sparse vectors with at most s elements well separated from 0. For a vector $\mathbf{a} \in \mathbb{R}^p$ and a matrix $\mathbf{A} \in \mathbb{R}^{p\times p}$, we denote

$$\|\mathbf{a}\|_{2,s} = \sup_{\|\mathbf{x}\|_2=1, \mathbf{x}\in\Gamma(s)} |\mathbf{a}^\intercal \mathbf{x}|, \quad \|\mathbf{A}\|_{2,s} = \sup_{\|\mathbf{x}\|_2=1, \mathbf{x}\in\Gamma(s)} \|\mathbf{A}\mathbf{x}\|_2. \quad (4.3)$$

For two parameters $\boldsymbol{\theta}$ and $\widetilde{\boldsymbol{\theta}}$, we define their distance as:

$$d_{2,s}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\theta}}) = (\vee_k |\pi_k - \widetilde{\pi}_k|) \vee (\vee_k \|\mathrm{vec}(\boldsymbol{\mu}_k - \widetilde{\boldsymbol{\mu}}_k)\|_{2,s}) \vee \|(\otimes_{m=M}^{m=1} \boldsymbol{\Sigma}_m - \otimes_{m=M}^{m=1} \widetilde{\boldsymbol{\Sigma}}_m)\mathrm{vec}(\widetilde{\mathbf{B}})\|_{2,s}, \quad (4.4)$$

where $\vee_k a_k = \max\{a_k : k = 1, 2, \ldots\}$.

We further define the contraction basin for $\boldsymbol{\theta}^*$ as

$$\begin{aligned} & \mathcal{B}_{con}(\boldsymbol{\theta}^*; a_\pi, a_\Delta, a_b, s) \quad (4.5) \\ = \; & \{\boldsymbol{\theta} : \pi_k \in (a_\pi, 1 - a_\pi), (1 - a_\Delta)(\Delta^*)^2 < |\delta_k(\mathbf{B})|, \sigma^2(\mathbf{B}) < (1 + a_\Delta)(\Delta^*)^2, \\ & \mathrm{vec}(\mathbf{B} - \mathbf{B}^*) \in \Gamma(s), \|\mathrm{vec}(\mathbf{B} - \mathbf{B}^*)\|_1 \leq a_b\Delta^*, \|\mathrm{vec}(\boldsymbol{\mu}_k)\|_{2,s} \leq a_b\Delta^*\}, \end{aligned}$$

where $a_\pi, a_\Delta, a_b > 0$, $a_\pi \leq c_\pi < 1$ are constants, $\delta_k(\mathbf{B}) = \langle \mathbf{B}, \boldsymbol{\mu}_k^* - (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)/2\rangle$ and $\sigma^2(\mathbf{B}) = \langle \mathbf{B}, [\![\mathbf{B}, \boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*]\!]\rangle$.

4.2 Initialization condition

We introduce a condition on the initial value that is important for our study. Define $d_0 = d_{2,s}(\widehat{\boldsymbol{\theta}}^{(0)}, \boldsymbol{\theta}^*)$ as the distance between the initial value $\widehat{\boldsymbol{\theta}}^{(0)}$ and the true parameter $\boldsymbol{\theta}^*$, where the function $d_{2,s}(\cdot,\cdot)$ is defined in (4.4). For an M-way tensor $\mathbf{A} \in \mathbb{R}^{p_1\times\cdots\times p_M}$, we let $\|\mathbf{A}\| = \sqrt{\sum_{\mathcal{J}} A_{\mathcal{J}}^2}$. The consistency of DEEM relies on the following condition, where a_π and a_Δ are defined in (4.5), c_π and $C_0 \equiv \prod_{m=1}^M C_m$ are from (4.1).

(C1) The initial estimator $\widehat{\mathbf{B}}^{(0)}$ satisfies that $d_0 \vee \|\widehat{\mathbf{B}}^{(0)} - \mathbf{B}^*\| \leq r\Delta^*, \mathrm{vec}(\widehat{\mathbf{B}}^{(0)} - \mathbf{B}^*) \in \Gamma(s)$, where $r < \min\{\frac{|a_\pi - c_\pi|}{\Delta}, \frac{\sqrt{9C_0 + 16a_\Delta} - \sqrt{9C_0}}{4}, \frac{a_\Delta}{C_0}, \frac{a_b}{5\sqrt{s}}\}$ and $r^{-1} = o\left(\sqrt{n/s \sum_{m=1}^M \log p_m}\right)$.

Condition (C1) indicates that the initial value is reasonable in the sense that d_0 is relatively small, and $\widehat{\mathbf{B}}^{(0)}$ is close to $\mathbf{B}^*$ and approximately sparse. This condition is important for our theoretical study because it guarantees that each iteration keeps improving our estimate. Due to the non-convex nature of clustering analysis, conditions on the initial value are popular in its theoretical studies; see Wang et al. (2015), Yi & Caramanis (2015), Balakrishnan et al. (2017), Cai et al. (2019) for example. Finding good initial values for cluster analysis is an important research area on its own, with many interesting works for the Gaussian mixture model (Kalai et al. 2010, Moitra & Valiant 2010, Hsu & Kakade 2013, Hardt & Price 2015).

For theoretical interests, we show that there exists an algorithm to generate initial values satisfying Condition (C1). One such initialization algorithm is presented as Algorithm S.4 in Section G of Supplementary Materials. Algorithm S.4 is related to the vector-based algorithm in Hardt & Price (2015), but is specially designed for tensor data. Under TNMM, it produces initial values that satisfy Condition (C1) under appropriate conditions, as shown in the following lemma.

Lemma 4. *Under the TNMM in* (2.2)*, suppose* $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}(c_\pi, C_b, s, \{C_m\}_{m=1}^M, C_b, \Delta_0)$*. If* $s^{12}\sum_{m=1}^M \log p_m = o(n)$*, with a probability greater than* $1 - O(\prod_m p_m^{-1})$*, Algorithm S.4 produces initial values that satisfy Condition (C1).*

Lemma 4 indicates that, under TNMM, when the sample size n is larger than $s^{12}\sum_{m=1}^M \log p_m$, Condition (C1) is satisfied by Algorithm S.4 with a probability tending to 1 as $n \to \infty$. Hence, we can meet Condition (C1) even when the dimension of each mode grows at an exponential rate of the sample size. The term s^{12} results from the theoretical properties of the initialization algorithm proposed by Hardt & Price (2015). Their algorithm solves an equation system that involves the first six moments of Gaussian mixtures. We need s to grow at our specified rate such that all these moments are estimated accurately. Also note that this sample size requirement matches the best one in literature when $M = 1$ and tensors reduce to vectors.

In the literature, there are also interests in removing conditions for initial values completely (Daskalakis et al. 2017, Wu & Zhou 2019). All these works require extensive efforts, and there is a considerable gap between these works and the topic in the manuscript. The existing works focus on low-dimensional vectors with known covariance matrices that are often assumed to be identity matrices, while we have high-dimensional tensors with unknown covariance matrices.

4.3 Main theorems

For our theory, we assume that the tuning parameters in DEEM are generated according to (3.22), with $\lambda^{(0)}$ defined as

$$\lambda^{(0)} = C_d \cdot (|\widehat{\pi}_2| \vee \|\mathrm{vec}(\widehat{\boldsymbol{\mu}}_1^{(0)} - \widehat{\boldsymbol{\mu}}_2^{(0)})\|_{2,s} \vee \|\bigotimes_{m=M}^{m=1} \widehat{\boldsymbol{\Sigma}}_m^{(0)}\|_{2,s})/\sqrt{s} + C_\lambda \sqrt{\sum_{m=1}^{M} \log p_m / n}, \quad (4.6)$$

where $C_d, C_\lambda > 0$ are constants.

Our ultimate goal is to show that the DEEM is asymptotically equivalent to the optimal rule in terms of clustering error. However, because $\mathbf{B}^*$ is the key parameter in clustering, we first present the theoretical properties of $\widehat{\mathbf{B}}^{(t)}$ as an intermediate result.

Theorem 1. *Consider $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}(s, c_\pi, \{C_m\}_{m=1}^M, C_b, \Delta_0)$ with $s = o(\sqrt{n/\sum_m \log p_m})$ and a sufficiently large Δ_0. Assume that Condition (C1) holds with $\sqrt{\sum_m \log p_m / n} = o(r)$, $\lambda^{(0)}$ is specified as in* (4.6) *and $\lambda^{(t)}$ is specified as in* (3.22)*. Then there exist constants $C_d, C_\lambda > 0$ and $0 < \kappa < 1/2$ such that, with a probability greater than $1 - O(\prod p_m^{-1})$, we have*

$$\|\widehat{\mathbf{B}}^{(t)} - \mathbf{B}^*\| \lesssim \kappa^t d_0 + \sqrt{\frac{s \sum_{m=1}^M \log p_m}{n}}. \quad (4.7)$$

Moreover, if $t \gtrsim (-\log(\kappa))^{-1} \log(n \cdot d_0)$, then

$$\|\widehat{\mathbf{B}}^{(t)} - \mathbf{B}^*\| \lesssim \sqrt{\frac{s \sum_{m=1}^M \log p_m}{n}}. \quad (4.8)$$

Theorem 1 implies that, under suitable conditions, DEEM produces an accurate estimate for $\mathbf{B}^*$ even in ultra-high dimensions after a sufficiently large number of iterations. The condition that $s = o(\sqrt{n/\sum_m \log p_m})$ implies that the model should be reasonably sparse. Also note that, this rate is derived under Condition (C1). But so far we are only able to guarantee Condition (C1) when $s = o[\{n/(\sum_m \log p_m)\}^{1/12}]$ (c.f Lemma 4), which necessarily implies that $s = o(\sqrt{n/\sum_m \log p_m})$.

We further require Δ_0 to be sufficiently large such that all the models of interest have large Δ^*. To avoid excessively lengthy expressions and calculations, we do not calculate the explicit dependence of our upper bound on Δ^* here. But we give an intuitive explanation on the impact of Δ^*. Note that (4.7) contains two terms, $\kappa^t d_0$ and $\sqrt{\frac{s \sum_{m=1}^M \log p_m}{n}}$, where d_0 is the distance between the initial value and the true parameters. Since $0 < \kappa < 1/2$, $\kappa^t d_0$ vanishes as long as $t \to \infty$, but

Δ^* is related to how fast this convergence is. Loosely speaking, the value of Δ^* inversely affects κ. For a larger Δ^*, we can find a smaller κ such that (4.7) holds with a high probability, and thus $\widehat{\mathbf{B}}^{(t)}$ converges to $\mathbf{B}^*$ in fewer iterations. When Δ^* is small, we can only find a larger κ, and the algorithmic convergence is slower. In our theory, Δ_0 can be viewed as the lower bound for Δ^* such we can find a $\kappa < 1/2$ to guarantee (4.7) with a high probability. See Section 4.4 for a numerical demonstration of the effect of Δ^*.

Now we present our main results concerning the clustering error. Denote the clustering error of DEEM as

$$R(\text{DEEM}) = \min_{\Pi:\{1,2\}\mapsto\{1,2\}} \Pr\left(\Pi(\widehat{Y}_i^{\text{DEEM}}) \neq Y_i\right). \tag{4.9}$$

Note that the clustering error is defined as the minimum over all permutations $\Pi : \{1, 2\} \mapsto \{1, 2\}$, since there could be label switching in clustering. In the meantime, recall that the lowest clustering error possible is achieved by assigning $\mathbf{X}_i$ to Cluster 2 if and only if (2.6) is true. Define the error rate of the optimal clustering rule as

$$R(\text{Opt}) = \Pr(\widehat{Y}_i^{opt} \neq Y_i), \tag{4.10}$$

where $\widehat{Y}_i^{opt}$ is determined by the optimal rule in (2.6). We study $R(\text{DEEM}) - R(\text{Opt})$.

Theorem 2. *Under the conditions in Theorem 1, we have that*

1. *For the κ that satisfies* (4.7)*, if $t \gtrsim (-\log(\kappa))^{-1}\log(n \cdot d_0)$, then with a probability greater than $1 - O(\prod p_m^{-1})$, we have*

$$R(\textit{DEEM}) - R(\textit{Opt}) \lesssim \frac{s\sum_{m=1}^{M}\log p_m}{n}. \tag{4.11}$$

2. *The convergence rate in* (4.11) *is minimax optimal over $\boldsymbol{\theta} \in \boldsymbol{\Theta}(c_\pi, C_b, s, \{C_m\}_{m=1}^M, \Delta_0)$.*

Theorem 2 shows that the error rate of DEEM converges to the optimal error rate even when the dimension of each mode of the tensor, p_m, grows at an exponential rate of n. Moreover, the convergence rate is minimax optimal. These results provide strong theoretical support for DEEM. The proofs of the upper bounds in Theorems 1 & 2 are related to those in Cai et al. (2019), but require a significant amount of additional efforts. We consider the tensor normal distribution, but non-asymptotic bounds for our estimators of $\boldsymbol{\Sigma}_1^*, \ldots, \boldsymbol{\Sigma}_M^*$ are not available in the literature.

Also, for us to claim the minimax optimality in Theorem 2, we have to find the lower bound for the excessive clustering error. This is achieved by constructing a family of models that characterize the intrinsic difficulty of estimating TNMMs. We consider models with sparse means and covariance matrices $\mathbf{\Sigma}_m$ proportional to identity matrices. The excessive clustering error of these models is no smaller than $O(n^{-1}s\sum_{m=1}^{M}\log p_m)$. Because this lower bound matches our upper bound in (4.11), we obtain the minimax optimality.

4.4 Cluster separation

Recall that we define the cluster separation as $\Delta^* = \langle \boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^*, [\![\boldsymbol{\mu}_2^* - \boldsymbol{\mu}_1^*; (\mathbf{\Sigma}^*)_1^{-1}, \ldots, (\mathbf{\Sigma}_M^*)^{-1}]\!] \rangle$. It quantifies the difficulty of clustering, and affects how fast the algorithmic error vanishes throughout the iterations (c.f Theorem 1). Here we demonstrate this impact with a numerical example.

We consider M1 from the simulation (Section 5) as a baseline. Define the cluster separation in M1 as Δ_1^*. We examine the performance of DEEM and its competitors with varying $\Delta^* = a\Delta_1^*$, where $a \in \{0.5, 0.75, 1, 2, 3, 4\}$. To achieve the specified Δ^*, we proportionally rescale $\boldsymbol{\mu}_2^*$ by $\sqrt{a}$ while keeping $\pi_k^*, \mathbf{\Sigma}_m^*$ unchanged. Since the sparse K-means (SKM; Witten & Tibshirani (2010)) and DEEM are the top two methods under model M1, we plot the clustering error of SKM, DEEM and the optimal rule in Figure 4.1. Clearly, both DEEM and SKM have smaller clustering error as Δ^* increases (left panel), and the relative clustering error shrinks at the same time (middle panel). Therefore, Δ^* is indeed a very accurate measure of the difficulty of a clustering problem. Moreover, the right panel shows that DEEM needs fewer iterations to achieve convergence when Δ^* is larger, which confirms our discussion following Theorem 1.

5 Numerical Studies

5.1 Simulations

In this section, our observations in all models are three-way tensors $\mathbf{X} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$. The prior probabilities are set to be $\pi_k^* = 1/K$, where K is the number of clusters. For simplicity, we let n_k be equal for $k = 1, \ldots, K$ in each model. We fix $\boldsymbol{\mu}_1^* = 0$, and specify covariance matrices $\mathbf{\Sigma}_m^*$, $m = 1, 2, 3$ and $\mathbf{B}_k^*, k = 2, \ldots, K$ for each model. For $\mathbf{B}_k^*$, all the elements not mentioned in the following model specification are set to be 0. For a matrix $\mathbf{\Omega} = [\omega_{ij}]$ and a scalar $\rho > 0$, we say

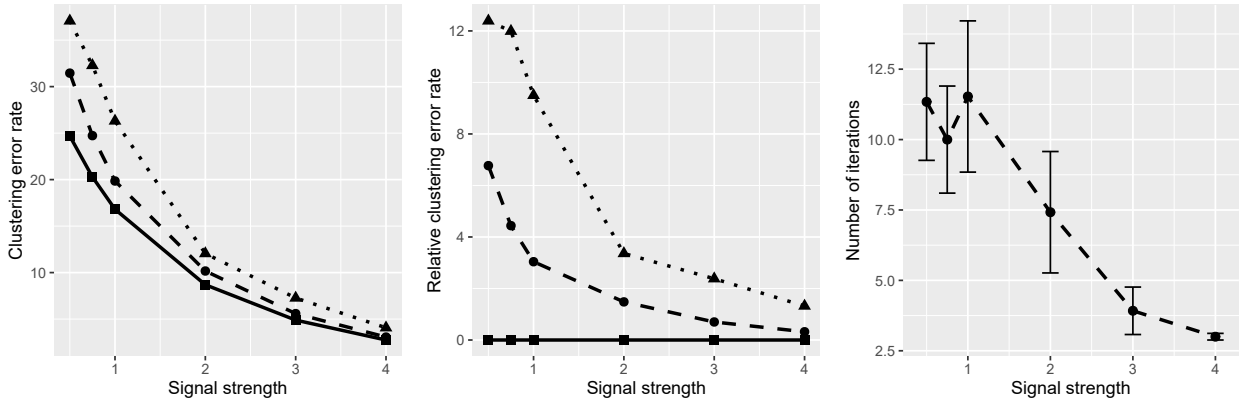

Figure 4.1: Clustering performance under M1 with varying $\Delta^* = a \times \Delta_1^*$ based on 100 replications. In all panels, the results for SKM are drawn in dotted line, those for DEEM are in dashed line, and those for the optimal rule is in solid line. The left panel shows the clustering error rates R of SKM, DEEM and the optimal rule. The middle panel shows the relative clustering error rates $R - R(\text{Opt})$ of SKM, DEEM and the optimal rule, where $R(\text{Opt})$ is the optimal error rate. The right panel shows number of iterations needed for convergence in DEEM with error bars represent 1.96 times standard error.

that $\mathbf{\Omega} = AR(\rho)$ if $\omega_{ij} = \rho^{|i-j|}$; and we say that $\mathbf{\Omega} = CS(\rho)$ if $\omega_{ij} = \rho + (1-\rho)1(i=j)$.

For each of the following seven simulation settings, we generate 100 independent data sets under the TNMM in (2.2). Each cluster has sample size $n_k = 50$ for Models M5 and M6, and $n_k = 75$ for all other models. Specifically, the simulation model parameters are as follows.

M1: $K = 2$, $p = 10 \times 10 \times 4$. $\mathbf{\Sigma}_1^* = CS(0.3)$, $\mathbf{\Sigma}_2^* = AR(0.8)$, $\mathbf{\Sigma}_3^* = CS(0.3)$, $\mathbf{B}^*_{2,[1:6,1,1]} = 0.5$.

M2: Same as **M1** except for $\mathbf{\Sigma}_2^*$, which is specified as follows. Let $\mathbf{\Omega}_0 = (\omega_{ij})$ where $\omega_{ij} = u_{ij}\delta_{ij}$, $\delta_{ij} \sim$ Bernoulli$(1, 0.05)$ and $u_{ij} \sim$ Unif$[0.5, 1] \cup [-1, -0.5]$. The we symmetrize $\mathbf{\Omega}_0$ by setting $\mathbf{\Omega} = (\mathbf{\Omega}_0 + \mathbf{\Omega}_0^\intercal)/2$. Set $\mathbf{\Omega}^* = \mathbf{\Omega} + \{\max(-\lambda_{\min}(\Omega), 0) + 0.05\}\mathbf{I}_{p_2}$. Finally rescale $\mathbf{\Omega}^*$ such that diagonal elements are 1, and $(\mathbf{\Sigma}_2^*)^{-1} = \mathbf{\Omega}^*$.

M3: Same as **M1** except for $K = 3$, $\mathbf{\Sigma}_3^* = CS(0.5)$ and $\mathbf{B}^*_{[2,1:6,1,1]} = -\mathbf{B}^*_{[3,1:6,1,1]} = 0.5$.

M4: $K = 4, p = 10 \times 10 \times 4$, $\mathbf{\Sigma}_1^* = \mathbf{I}_{p_1}$, $\mathbf{\Sigma}_2^* = AR(0.8)$, $\mathbf{\Sigma}_3^* = \mathbf{I}_{p_3}$, $\mathbf{B}^*_{[2,1:6,1,1]} = -\mathbf{B}^*_{[3,1:6,1,1]} = 0.8$.

M5: $K = 6$, $p = 10 \times 10 \times 4$. $\mathbf{\Sigma}_1^* = AR(0.9)$, $\mathbf{\Sigma}_2^* = CS(0.6)$, $\mathbf{\Sigma}_3^* = AR(0.9)$. $\mathbf{B}^*_{[2,1:6,1,1]} = 0.6$, $\mathbf{B}^*_{[3,1:6,1,1]} = 1.2$, $\mathbf{B}^*_{[4,1:6,1,1]} = 1.8$, $\mathbf{B}^*_{[5,1:6,1,1]} = 2.4$, $\mathbf{B}^*_{[6,1:6,1,1]} = 3$.

M6: $K = 6$, $p = 10 \times 10 \times 4$. We specify $\boldsymbol{\mu}_k^*$ instead of $\mathbf{B}_k^*$. The corner $u_1 \times u_2 \times u_3 = 8 \times 1 \times 1$ sub-tensor of $\boldsymbol{\mu}_k$ is filled with independently Unif$[0, 1]$ numbers, while we fill in zeros elsewhere. Then we center it as $\boldsymbol{\mu}_k^* = \boldsymbol{\mu}_k - \boldsymbol{\mu}_1$ for $k = 1, \ldots, K$. The covariance matrices $\mathbf{\Sigma}_m^*$'s are all two-

	Optimal	K-means	SKM	**DEEM**	DTC	TBM	**EM**	AFPF	CHIME
M1	16.81 (0.34)	32.43 (0.40)	26.31 (0.68)	19.85 (0.35)	34.10 (0.42)	32.91 (0.38)	34.38 (0.42)	32.69 (0.39)	32.45 (0.41)
M2	9.59 (0.25)	31.26 (0.42)	32.01 (0.67)	12.99 (0.53)	34.91 (0.87)	31.43 (0.41)	28.20 (0.54)	42.44 (0.66)	46.75 (0.24)
M3	17.27 (0.25)	34.57 (0.39)	22.32 (0.29)	20.16 (0.33)	40.72 (0.42)	34.75 (0.34)	32.84 (0.35)	35.88 (0.35)	NA (–)
M4	22.31 (0.27)	44.62 (0.42)	40.21 (0.56)	26.84 (0.39)	45.28 (0.65)	45.78 (0.39)	42.74 (0.41)	42.89 (0.42)	NA (–)
M5	8.47 (0.16)	24.53 (0.67)	15.93 (0.28)	10.07 (0.26)	64.24 (0.38)	20.78 (0.33)	19.64 (0.33)	21.88 (0.33)	NA (–)
M6	10.40 (0.16)	34.69 (0.79)	23.93 (0.60)	16.00 (0.47)	71.18 (0.33)	34.13 (0.59)	27.36 (0.46)	29.16 (1.50)	NA (–)
M7	8.30 (0.20)	34.08 (0.64)	25.85 (1.17)	12.27 (0.74)	44.05 (0.51)	33.61 (0.63)	33.48 (0.64)	NA (–)	NA (–)

Table 1: Reported are the averages and standard errors (in parentheses) of clustering error rates based on 100 replicates.

block-diagonal, where the block sizes corresponding to the zero versus nonzero in means. Each block is generated as $\mathbf{ODO}^T$, where $\mathbf{O}$ is a randomly generated orthogonal matrix and $\mathbf{D}$ is a diagonal matrix that contains the eigenvalues. The first block's $\mathbf{D}$ is set as $5u$, $u = 1, \ldots, u_m$ and the second block's $\mathbf{D}$ is set as $2 \times \log(v+1)$, $v = 1, \ldots, p_m - u_m$. Finally we standardize $\mathbf{\Sigma}_m^*$ to have unit Frobenius norm.

M7: $K = 2$, $p = 30 \times 30 \times 30$. $\mathbf{\Sigma}_1^* = CS(0.5)$, $\mathbf{\Sigma}_2^* = AR(0.8)$, $\mathbf{\Sigma}_3^* = CS(0.5)$. $\mathbf{B}^*_{[2,1:6,1,1]} = 0.6$.

Models M1–M7 cover a wide range of models. In M1 and M2, we consider two mixtures, where we include various covariance structure such as auto-correlation $AR(\rho)$, compound symmetric $CS(\rho)$, and sparse inverse covariance (in M2). Then in M3 and M4, we increase the number of clusters to $K = 3$ and slightly modify other parameters to keep the optimal clustering error around 0.2. In M5 and M6, we further increase the number of clusters to $K = 6$ and decrease the cluster size n_k from 75 to 50. In M6, we consider a type of mean-covariance joint parameterization that corresponds to the envelope mixture models (Wang et al. 2020). This mimics strong correlation but separable signals. Finally, M7 is constructed so that $p = 30^3 = 27,000$ is significantly higher than the other models.

We consider several popular methods as competitors of DEEM, including K-means, and standard EM (EM; Section 3.1), sparse K-means (SKM; Witten & Tibshirani (2010)), adaptive pairwise fusion penalized clustering (APFP; Guo et al. (2010)), high-dimensional Gaussian mixtures with EM algorithm (CHIME; Cai et al. (2019)), dynamic tensor clustering (DTC; Sun & Li (2018)), tensor block model (TBM; Wang & Zeng (2019)). We want to remark that the most direct competitor is the standard EM for TNMM. The DTC and TBM methods are designed for tensor data but from a different perspective. As discussed in the Section 1, DTC's advantage is from tensor decomposition and TBM is a co-clustering method (clustering variables and observations simultaneously). Other methods are designed for vector data. We vectorize the tensors before applying the vector-based methods. We use the built-in function in `R` for K-means, the `R` package `sparcl` for SKM, the `R` package `PARSE` for APFP, and the `R` package `tensorsparse` for TBM. The code of DTC is downloaded from the authors' websites. In addition, we include the error rates of the optimal rule as a baseline.

The implementation of TBM works on three-way data tensor. Our data is a four way tensor of dimension $n \times p_1 \times p_2 \times p_3$, with the observations being an additional mode. Hence, when we apply TBM, we first apply mode-1 matricization to each observation and then combine the observations as a three-way tensor of dimension $n \times p_1 \times (p_2 p_3)$. Also, TBM requires specifying the number of clusters along each mode. We use true K as number of clusters along the first mode (i.e, the mode of the observations) and apply the BIC in Wang & Zeng (2019) to tune the numbers of clusters on the second and the third mode. In Section D of Supplementary Materials, we also conduct additional simulations under the TBM data generating process.

We compare the clustering error rates of all the methods. We calculate the clustering error rate to be $\min_{\Pi} \frac{1}{n} \sum_{i=1}^{n} 1(\widehat{Y}_i \neq \Pi(Y_i))$ over all possible permutations $\Pi : \{1, \ldots, K\} \mapsto \{1, \ldots, K\}$ of cluster labels. The clustering error rates are summarized in Table 1. Due to its excessively long computation time, the results of AFPF are based on 30 replications in M6, and are not reported for M7. The results for CHIME are only reported for M1 and M2, because M3–M6 have $K > 2$ clusters, while the implementation for CHIME is only available for $K = 2$. On the other hand, CHIME exceeds the memory limit of 16GB we set for all methods for M7.

We make a few remarks on Table 1. First of all, DEEM is significantly better than all the other methods across a wide range of TNMM parameter settings. Such results suggest that DEEM has

very competitive numerical performance in the presence of different correlation structure, number of clusters and dimensions. The advantage of DEEM is likely a consequence of exploiting the tensor structure, modeling the correlation and imposing variable selection, as no competitor combines all these three components together. Secondly, the tensor methods DTC and TBM assume different statistical models and do not account for the correlation among variables. Therefore, they are less efficient than DEEM under the TNMM. Finally, variable selection generally improves clustering accuracy in high dimensions. DEEM and EM fit the same model, but a major distinction between them is that DEEM enforces variable selection while EM does not. Analogously, the sparse K-means (SKM) is uniformly better than K-means. This demonstrates the importance of variable selection in clustering problems.

5.2 Real data illustration

We further compare DEEM with the competitors on the BHL (brain, heart and lung) dataset, available at `https://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS1083`. This dataset contains the expression levels of 1124 genes on 27 brain, heart or lung tissues. On each tissue, the measurement is repeated four times. Hence, our observation $\mathbf{X}_i \in \mathbb{R}^{4\times 1124}$, with each row being the gene expression level of one measurement. We attempt to recover the type of each tissue based on $\mathbf{X}_i, i = 1, \ldots, 27$.

We preprocess the data by performing the Kolmogorov-Smirnov test (KS test) on each column to compare its overall distribution with the normal distribution. Only the columns with small p-values are preserved for clustering. We consider reducing the dimension of $\mathbf{X}_i$ to 4×20 and 4×30. We apply DEEM along with all the competitors in Section 5.1 on this dataset. For DEEM, we generate 30 different initial values and use BIC to tune the initial value along with the tuning parameter. The same is done for DTC. For APFP, it is suggested by the authors to first fit GMM for 100 times without penalty with different random initial values and select the one with the highest likelihood. We follow this suggestion. The implementations of SKM do not allow users to specify initial values, so we let it pick its own initial value. The clustering error rates are reported in Table 2. It can be seen that DEEM has comparable or superior performance to all the competitors in both dimensions. The lowest clustering error rate is achieved by DEEM with dimension 4×20.

	DEEM	K-means	SKM	DTC	TBM	EM	AFPF
4×20	**7.41**	14.81	14.81	22.22	33.33	14.81	14.81
4×30	11.11	11.11	11.11	18.52	11.11	11.11	11.11

Table 2: Clustering error rates of the BHL data.

6 Discussion

In this paper, we propose and study the tensor normal mixture model (TNMM). It is a natural extension of the popular GMM to tensor data. The proposed method simultaneously performs variable selection, covariance estimation and clustering for tensor mixture models. While Kronecker tensor covariance structure is utilized to significantly reduce the number of parameters, it incorporates the dependence between variables and along each tensor modes. This distinguishes our method from independence clustering methods such as K-means. We enforce variable selection in the enhanced E-step via convex optimization, where sparsity is directly derived from the optimal clustering rule. We propose completely explicit updates in the enhanced M-step, where the new moment-based estimator for covariance is computationally fast and does not require sparsity or other structural assumptions on the covariance. Encouraging theoretical results are established for DEEM, and are further supported by numerical examples.

Our DEEM algorithm is developed for multi-cluster problem, e.g. $K \geq 2$, and has been shown to work well in simulations when K is not too large. Since the number of parameters in TNMM grows with K, extensions such as low-rank decomposition on $\mathbf{B}_k^*$ may be needed for problems where the number of clusters are expected to be large. Moreover, theoretical study is challenging for $K > 2$ and for unknown K. Such extensions of our theoretical results from $K = 2$ to general K are yet to be studied. Relatedly, consistent selection of K remains an open question for TNMM.

References

Anderlucci, L. & Viroli, C. (2015), 'Covariance pattern mixture models for the analysis of multivariate heterogeneous longitudinal data', *The Annals of Applied Statistics* **9**(2), 777–800.

Arthur, D. & Vassilvitskii, S. (2007), K-means++: the advantages of careful seeding, *in* 'In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms'.

Balakrishnan, S., Wainwright, M. J. & Yu, B. (2017), 'Statistical guarantees for the em algorithm: From population to sample-based analysis', *The Annals of Statistics* **45**(1), 77–120.

Banfield, J. D. & Raftery, A. E. (1993), 'Model-based gaussian and non-gaussian clustering', *Biometrics* pp. 803–821.

Bickel, P. J. & Levina, E. (2004), 'Some theory for fisher's linear discriminant function,naive bayes', and some alternatives when there are many more variables than observations', *Bernoulli* **10**(6), 989–1010.

Bing, X., Bunea, F., Ning, Y., Wegkamp, M. et al. (2020), 'Adaptive estimation in structured factor models with applications to overlapping clustering', *Annals of Statistics* **48**(4), 2055–2081.

Bunea, F., Giraud, C., Luo, X., Royer, M. & Verzelen, N. (2020), 'Model assisted variable clustering: minimax-optimal recovery and algorithms', *The Annals of Statistics* **48**(1), 111–137.

Cai, T. & Liu, W. (2011), 'A direct estimation approach to sparse linear discriminant analysis', *Journal of the American Statistical Association* **106**(1), 1566–1577.

Cai, T. T., Ma, J. & Zhang, L. (2019), 'Chime: Clustering of high-dimensional gaussian mixtures with em algorithm and its optimality', *The Annals of Statistics* **47**(3), 1234–1267.

Cao, X., Wei, X., Han, Y., Yang, Y. & Lin, D. (2013), Robust tensor clustering with non-greedy maximization, *in* 'Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence', IJCAI '13, AAAI Press, pp. 1254–1259.

Chen, J. (1995), 'Optimal rate of convergence for finite mixture models', *The Annals of Statistics* pp. 221–233.

Chi, E. C., Allen, G. I. & Baraniuk, R. G. (2017), 'Convex biclustering', *Biometrics* **73**(1), 10–19.

Chi, E. C., Gaines, B. R., Sun, W. W., Zhou, H. & Yang, J. (2020), 'Provable convex co-clustering of tensors', *Journal of Machine Learning Research* **21**(214), 1–58.

Chi, E. C. & Kolda, T. G. (2012), 'On tensors, sparsity, and nonnegative factorizations', *SIAM Journal on Matrix Analysis and Applications* **33**(4), 1272–1299.

Chiang, M. M.-T. & Mirkin, B. (2010), 'Intelligent choice of the number of clusters in k-means clustering: an experimental study with different cluster spreads', *Journal of classification* **27**(1), 3–40.

Cohen, M. B., Elder, S., Musco, C., Musco, C. & Persu, M. (2015), Dimensionality reduction for k-means clustering and low rank approximation, *in* 'Proceedings of the forty-seventh annual ACM symposium on Theory of computing', pp. 163–172.

Daskalakis, C., Tzamos, C. & Zampetakis, M. (2017), Ten steps of em suffice for mixtures of two gaussians, *in* 'Conference on Learning Theory', pp. 704–710.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977), 'Maximum likelihood from incomplete data via the em algorithm', *Journal of the Royal Statistical Society: Series B (Methodological)* **39**(1), 1–22.

Dutilleul, P. (1999), 'The mle algorithm for the matrix normal distribution', *Journal of statistical computation and simulation* **64**(2), 105–123.

Dwivedi, R., Ho, N., Khamaru, K., Wainwright, M. J., Jordan, M. I., Yu, B. et al. (2020), 'Singularity, misspecification and the convergence rate of em', *Annals of Statistics* **48**(6), 3161–3182.

Fan, J. & Fan, Y. (2008), 'High dimensional classification using features annealed independence rules', *Annals of statistics* **36**(6), 2605.

Fang, Y. & Wang, J. (2012), 'Selection of the number of clusters via the bootstrap method', *Computational Statistics & Data Analysis* **56**(3), 468–477.

Fosdick, B. K. & Hoff, P. D. (2014), 'Separable factor analysis with applications to mortality data', *The annals of applied statistics* **8**(1), 120.

Fraley, C. & Raftery, A. E. (2002), 'Model-based clustering, discriminant analysis, and density estimation', *Journal of the American statistical Association* **97**(458), 611–631.

Friedman, J., Hastie, T. & Tibshirani, R. (2001), *The elements of statistical learning*, Vol. 1, Springer series in statistics Springer, Berlin.

Fu, W. & Perry, P. O. (2020), 'Estimating the number of clusters using cross-validation', *Journal of Computational and Graphical Statistics* **29**(1), 162–173.

Fujita, A., Takahashi, D. Y. & Patriota, A. G. (2014), 'A non-parametric method to estimate the number of clusters', *Computational Statistics & Data Analysis* **73**, 27–39.

Gallaugher, M. P. & McNicholas, P. D. (2018), 'Finite mixtures of skewed matrix variate distributions', *Pattern Recognition* **80**, 83–93.

Gao, X., Shen, W., Zhang, L., Hu, J., Fortin, N. J., Frostig, R. D. & Ombao, H. (2021), 'Regularized matrix data clustering and its application to image analysis', *Biometrics* .

Guo, J., Levina, E., Michailidis, G. & Zhu, J. (2010), 'Pairwise variable selection for high-dimensional model-based clustering', *Biometrics* **66**(3), 793–804.

Gupta, A. & Nagar, D. (1999), *Matrix Variate Distributions*, Vol. 104, CRC Press.

Hao, B., Sun, W. W., Liu, Y. & Cheng, G. (2018), 'Simultaneous clustering and estimation of heterogeneous graphical models', *Journal of Machine Learning Research* **18**(217), 1–58.

Hardt, M. & Price, E. (2015), Tight bounds for learning a mixture of two gaussians, *in* 'Proceedings of the forty-seventh annual ACM symposium on Theory of computing', ACM, pp. 753–760.

Heinrich, P. & Kahn, J. (2018), 'Strong identifiability and optimal minimax rates for finite mixture estimation', *The Annals of Statistics* **46**(6A), 2844–2870.

Hoff, P. D. (2011), 'Separable covariance arrays via the tucker product, with applications to multivariate relational data', *Bayesian Analysis* **6**(2), 179–196.

Hoff, P. D. (2015), 'Multilinear tensor regression for longitudinal relational data', *The Annals of Applied Statistics* **9**(3), 1169–1193.

Hsu, D. & Kakade, S. M. (2013), Learning mixtures of spherical gaussians: moment methods and spectral decompositions, *in* 'Proceedings of the 4th conference on Innovations in Theoretical Computer Science', pp. 11–20.

Jegelka, S., Sra, S. & Banerjee, A. (2009), Approximation algorithms for tensor clustering, *in* 'International Conference on Algorithmic Learning Theory', Springer, pp. 368–383.

Kalai, A. T., Moitra, A. & Valiant, G. (2010), Efficiently learning mixtures of two gaussians, *in* 'Proceedings of the forty-second ACM symposium on Theory of computing', pp. 553–562.

Kolda, T. G. & Bader, B. W. (2009), 'Tensor decompositions and applications', *SIAM Review* **51**(3), 455–500.

Kolda, T. G. & Sun, J. (2008), Scalable tensor decompositions for multi-aspect data mining, *in* '2008 Eighth IEEE international conference on data mining', IEEE, pp. 363–372.

Law, M. H. C., Figueiredo, M. A. T. & Jain, A. K. (2004), 'Simultaneous feature selection and clustering using mixture models', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**(9), 1154–1166.

Lee, M., Shen, H., Huang, J. Z. & Marron, J. (2010), 'Biclustering via sparse singular value decomposition', *Biometrics* **66**(4), 1087–1095.

Li, L. & Zhang, X. (2017), 'Parsimonious tensor response regression', *Journal of the American Statistical Association* **112**(519), 1131–1146.

Lock, E. F. (2018), 'Tensor-on-tensor regression', *Journal of Computational and Graphical Statistics* **27**(3), 638–647.

Lyu, T., Lock, E. F. & Eberly, L. E. (2017), 'Discriminating sample groups with multi-way data', *Biostatistics* **18**(3), 434–450.

Lyu, X., Sun, W. W., Wang, Z., Liu, H., Yang, J. & Cheng, G. (2019), 'Tensor graphical model: Non-convex optimization and statistical inference', *IEEE transactions on pattern analysis and machine intelligence* .

MacQueen, J. (1967), Some methods for classification and analysis of multivariate observations, *in* 'Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics', pp. 281–297.

Mai, Q., Yang, Y. & Zou, H. (2019), 'Multiclass sparse discriminant analysis', *Statistica Sinica* **29**, 97–111.

Mai, Q., Zou, H. & Yuan, M. (2012), 'A direct approach to sparse discriminant analysis in ultra-high dimensions', *Biometrika* **99**(1), 29–42.

Manceur, A. M. & Dutilleul, P. (2013), 'Maximum likelihood estimation for the tensor normal distribution: Algorithm, minimum sample size, and empirical bias and dispersion', *Journal of Computational and Applied Mathematics* **239**, 37–49.

McLachlan, G. J., Lee, S. X. & Rathnayake, S. I. (2019), 'Finite mixture models', *Annual review of statistics and its application* **6**, 355–378.

Moitra, A. & Valiant, G. (2010), Settling the polynomial learnability of mixtures of gaussians, *in* '2010 IEEE 51st Annual Symposium on Foundations of Computer Science', IEEE, pp. 93–102.

Ng, A. Y., Jordan, M. I. & Weiss, Y. (2001), On spectral clustering: Analysis and an algorithm, *in* 'Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic', NIPS'01, pp. 849–856.

Pan, W. & Shen, X. (2007), 'Penalized model-based clustering with application to variable selection', *J. Mach. Learn. Res.* **8**, 1145–1164.

Pan, Y., Mai, Q. & Zhang, X. (2019), 'Covariate-adjusted tensor classification in high dimensions', *Journal of the American statistical association* **114**(527), 1305–1319.

Raskutti, G., Yuan, M. & Chen, H. (2019), 'Convex regularization for high-dimensional multiresponse tensor regression', *The Annals of Statistics* **47**(3), 1554–1584.

Sugar, C. A. & James, G. M. (2003), 'Finding the number of clusters in a dataset: An information-theoretic approach', *Journal of the American Statistical Association* **98**(463), 750–763.

Sun, W. W. & Li, L. (2018), 'Dynamic tensor clustering', *Journal of the American Statistical Association* **0**(ja), 1–30.

Sun, W. W., Lu, J., Liu, H. & Cheng, G. (2016), 'Provable sparse tensor decomposition', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **79**(3), 899–916.

Tan, K. M. & Witten, D. M. (2014), 'Sparse biclustering of transposable data', *Journal of Computational and Graphical Statistics* **23**(4), 985–1008.

Tibshirani, R., Walther, G. & Hastie, T. (2001), 'Estimating the number of clusters in a data set via the gap statistic', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **63**(2), 411–423.

Verzelen, N. & Arias-Castro, E. (2017), 'Detection and feature selection in sparse mixture models', *Ann. Statist.* **45**(5), 1920–1950.

Viroli, C. (2011), 'Finite mixtures of matrix normal distributions for classifying three-way data', *Statistics and Computing* **21**(4), 511–522.

Wang, J. (2010), 'Consistent selection of the number of clusters via crossvalidation', *Biometrika* **97**(4), 893–904.

Wang, M. & Zeng, Y. (2019), Multiway clustering via tensor block models, *in* 'Advances in Neural Information Processing Systems', pp. 715–725.

Wang, S. & Zhu, J. (2008), 'Variable selection for model-based high-dimensional clustering and its application to microarray data', *Biometrics* **64**(2), 440–448.

Wang, W., Zhang, X. & Mai, Q. (2020), 'Model-based clustering with envelopes', *Electronic Journal of Statistics* **14**(1), 82–109.

Wang, X. & Zhu, H. (2017), 'Generalized scalar-on-image regression models via total variation', *Journal of the American Statistical Association* **112**(519), 1156–1168.

Wang, Z., Gu, Q., Ning, Y. & Liu, H. (2015), High dimensional em algorithm: Statistical optimization and asymptotic normality, *in* 'Advances in neural information processing systems', pp. 2521–2529.

Witten, D. M. & Tibshirani, R. (2010), 'A framework for feature selection in clustering.', *Journal of the American Statistical Association* **105**(490), 713–726.

Wu, Y. & Zhou, H. H. (2019), 'Randomly initialized em algorithm for two-component gaussian mixture achieves near optimality in $o(\sqrt{n})$ iterations', *arXiv preprint arXiv:1908.10935* .

Yi, X. & Caramanis, C. (2015), Regularized em algorithms: A unified framework and statistical guarantees, *in* 'Advances in Neural Information Processing Systems', pp. 1567–1575.

Yuan, M. & Lin, Y. (2006), 'Model selection and estimation in regression with grouped variables', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **68**(1), 49–67.

Zhang, A. & Han, R. (2019), 'Optimal sparse singular value decomposition for high-dimensional high-order data', *Journal of the American Statistical Association* **114**(528), 1708–1725.

Zhang, X. & Li, L. (2017), 'Tensor envelope partial least-squares regression', *Technometrics* **59**(4), 426–436.

Zhou, H., Li, L. & Zhu, H. (2013), 'Tensor regression with applications in neuroimaging data analysis', *Journal of the American Statistical Association* **108**(502), 540–552.