International Journal of Data Science and Analytics (2020) 10:161–175
https://doi.org/10.1007/s41060-020-00205-5

REGULAR PAPER

Tensor latent block model for co-clustering

Rafika Boutalbi[1] · Lazhar Labiod[1] · Mohamed Nadif[1]

Received: 10 July 2019 / Accepted: 12 October 2019 / Published online: 19 February 2020

Abstract
With the exponential growth of collected data in different fields like recommender system (user, items), text mining (document, term), bioinformatics (individual, gene), co-clustering, which is a simultaneous clustering of both dimensions of a data matrix, has become a popular technique. Co-clustering aims to obtain homogeneous blocks leading to a straightforward simultaneous interpretation of row clusters and column clusters. Many approaches exist; in this paper, we rely on the latent block model (LBM), which is flexible, allowing to model different types of data matrices. We extend its use to the case of a tensor (3D matrix) data in proposing a *Tensor LBM* (TLBM), allowing different relations between entities. To show the interest of TLBM, we consider continuous, binary, and contingency tables datasets. To estimate the parameters, a variational EM algorithm is developed. Its performances are evaluated on synthetic and real datasets to highlight different possible applications.

Keywords Co-clustering · Tensor · Data science

1 Introduction

Co-clustering addresses the problem of simultaneous clustering of both dimensions of a data matrix. Many of the datasets encountered in data science are two-dimensional in nature and can be represented by a matrix. Classical clustering procedures seek to construct separately an optimal partition of rows (individuals) or, sometimes (features), of columns. In contrast, co-clustering methods cluster the rows and the columns simultaneously and organize the data into homogeneous blocks (after suitable permutations); see, for instance, [1,4,11,15,16,18,19,25,26,30,31]. Methods of this kind have practical importance in a wide variety of applications where data are typically organized in two-way tables. However, in modern datasets, instead of collecting data on every individual-feature pair, we may collect supplementary individual or item information leading to tensor representation. This kind of data has emerged in many fields such as recommender systems where the data are collected on multiple items rated by multiple users; information about users and items is also available yielding as a tensor rather than a data matrix.

Despite the great interest for co-clustering techniques, on the one hand, and the tensor representation, on the other, few works tackle co-clustering from tensor data. We mention the work based on minimum Bregman information (MBI) to carry out co-clustering [3] and the *general tensor spectral co-clustering* (GTSC) method suitable for nonnegative tensor data [35]. Other approaches can be cited although the goal is not exactly co-clustering but only extracting a bicluster. For instance, in [12] the authors aim to extract a bicluster composed of a subset of tensor rows and columns whose corresponding trajectories form a low-dimensional subspace. However, the majority of authors consider the same entities for the row and columns or do not consider the tensor co-clustering under a probabilistic approach. To the best of our knowledge, this is the first attempt to formulate our objective when both sets—row and column—are different and with model-based co-clustering.

To this end, we rely on the latent block models [19] for their flexibility to consider any type of data matrices.

This submission is an extension version of the PAKDD 2019 paper 'Co-clustering from Tensor Data'.

✉ Rafika Boutalbi
rafika.boutalbi@u-paris.fr

Lazhar Labiod
lazhar.labiod@u-paris.fr

Mohamed Nadif
mohamed.nadif@u-paris.fr

[1] LIPADE, Université de Paris, 75006 Paris, France

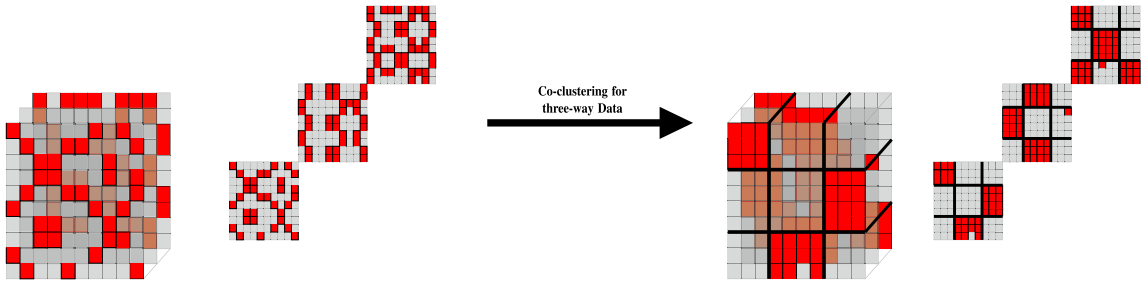


Fig. 1 Goal of co-clustering for binary three-way data

In this paper, we propose a co-clustering model for tensor data, where clustering of row (indexed from $i = 1$ to n) and column (indexed from $j = 1$ to d) entities is done not only on principal relation matrix but on tensor including multiple covariates and/or relations between entities. We focus on three-way data, but this contribution can be extended whatever the dimension. The proposed model can also be viewed as multi-way clustering approach where each slice (indexed from $a = 1$ to v) of the third dimension of represents a relation or covariate (see Fig. 1). Thereby, the purpose is to simultaneously discover the row (indexed from $k = 1$ to g) and column (indexed from $\ell = 1$ to m) clusters and the relationship between these clusters for all slices. To achieve this, we propose to extend *latent block model* (LBM) to tensor data referred to as *Tensor LBM* (TLBM). TLBM is suitable for several applications; our first investigation has appeared recently as a paper published in [6]. In the present manuscript, we delve in depth into this idea and present several new theoretical and empirical results. The main contributions of this paper are summarized as follows: (i) We propose an extension of LBM for tensor data (TLBM); (ii) we show its flexibility to be applied with different types of data; (iii) we derive a variational EM and a hard version for co-clustering.

The remainder of this paper is organized as follows: Section 2 describes LBM and presents its extension TLBM. Section 3 details the proposed algorithm variational EM for co-clustering. In Sect. 4, we present a hard version of the proposed algorithm. Section 5 presents experimental results on the synthetic dataset, and in Sect. 6 are reported comparisons with several algorithms on real-world datasets. Section 7 concludes this paper and provides some directions for future work.

2 Extension of latent block model

2.1 Latent block model

The latent block model [19] in $g \times m$ blocks is defined as follows: Given a matrix $\mathbf{X}$ of size $n \times d$, we assume that there is a couple of partitions $(\mathbf{z}, \mathbf{w})$ where $\mathbf{z}$ is partitioned into g clusters on the set of rows I and $\mathbf{w}$ is partitioned into m clusters on the set of columns J, such that each element x_{ij} belonging to the block $k\ell$ is generated according to a probability distribution, where k represents the class of row i, while ℓ represents the class of column j. The $\mathbf{z}$ partition can be represented by a vector of labels or by $\mathbf{z} = (z_{ik})$ of size $n \times g$ where $z_{ik} = 1$ if i belongs to the class k, and $z_{ik} = 0$ otherwise. In the same way, the $\mathbf{w}$ partition can be represented by a label vector or by a column classification matrix $\mathbf{w} = (w_{j\ell})$ of size $d \times m$ where $w_{j\ell} = 1$ if j belongs to the class ℓ, and $w_{j\ell} = 0$ otherwise. Under the independence assumption $p(\mathbf{z}, \mathbf{w}) = p(\mathbf{z})p(\mathbf{w})$ and noting $\mathcal{Z}$ and $\mathcal{W}$ the sets of all possible partitions $\mathbf{z}$ and $\mathbf{w}$, the likelihood of the observed data $f(\mathbf{X}; \boldsymbol{\Omega})$ is given by:

$$\sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} \prod_{i,k} \pi_k^{z_{ik}} \prod_{j,\ell} \rho_\ell^{w_{j\ell}} \prod_{i,j,k,\ell} \left(\Phi(x_{ij}; \boldsymbol{\lambda}_{k\ell})\right)^{z_{ik}w_{j\ell}} \quad (1)$$

where $\boldsymbol{\Omega} = (\boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\lambda})$ are the unknown parameters of LBM with $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_g)$ and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m)$ where $(\pi_k = p(z_{ik} = 1), k = 1, \ldots, g)$, $(\rho_\ell = p(w_{j\ell} = 1), \ell = 1, \ldots, m)$ are the proportions of clusters and $\boldsymbol{\lambda}_{k\ell}$ represents the parameter of Φ of the $(k, \ell)^{th}$ block. The classification log-Likelihood takes the following form:

$$\begin{aligned} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega}) &= \sum_{i,k} z_{ik} \log \pi_k + \sum_{j,\ell} w_{j\ell} \log \rho_\ell \\ &\quad + \sum_{i,j,k,\ell} z_{ik} w_{j\ell} \log(\Phi(x_{ij}; \boldsymbol{\lambda}_{k\ell})). \end{aligned} \quad (2)$$

2.2 Latent block model for tensor data (TLBM)

Hereafter, we propose an extension of LBM for tensor data (TLBM). Few studies have addressed the issue of co-clustering for tensor data [12,35]. Unlike classical LBM which considers data matrix $\mathbf{X} = [x_{ij}] \in \mathbb{R}^{n\times d}$, TLBM considers 3D data matrix $\mathbf{X} = [\mathbf{x}_{ij}] \in \mathbb{R}^{n\times d\times v}$ where n is the number of rows, d the number of columns, and v the number of covariates. Figure 2a presents the data structure. Note that in our cases, a co-cluster is a parallelepiped. The generative process is described in Algorithm 1; TLBM is flexible and can be used with different types of data.

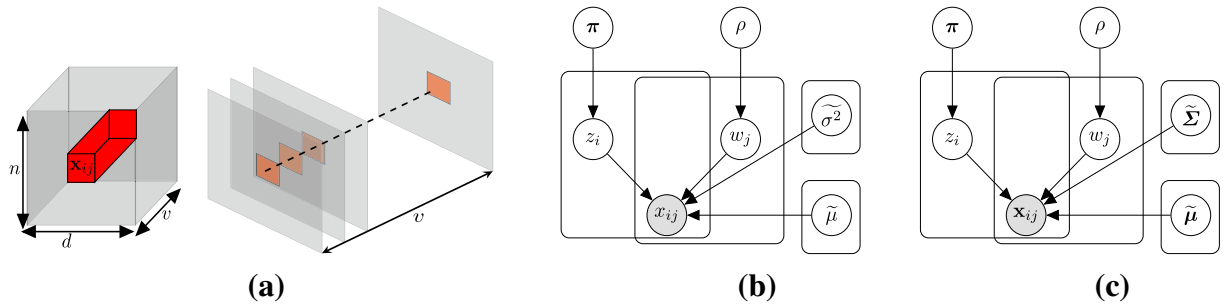


Fig. 2 **a** Data structure, **b** Gaussian LBM with $\widetilde{\mu} = \{\mu_{11}, \ldots, \mu_{gm}\}$, $\widetilde{\sigma^2} = \{\sigma^2_{11}, \ldots, \sigma^2_{gm}\}$ where $\forall k, \ell, \quad \mu_{k\ell}, \sigma^2_{k,\ell} \in \mathbb{R}$, **c** Gaussian TLBM with $\widetilde{\boldsymbol{\mu}} = \{\boldsymbol{\mu}_{11}, \ldots, \boldsymbol{\mu}_{gm}\}$, $\widetilde{\boldsymbol{\Sigma}} = \{\boldsymbol{\Sigma}_{11}, \ldots, \boldsymbol{\Sigma}_{gm}\}$ where $\forall k, \ell, \quad \boldsymbol{\mu}_{k\ell} \in \mathbb{R}^{v\times 1}$ and $\boldsymbol{\Sigma}_{k\ell} \in \mathbb{R}^{v\times v}$

Continuous data We can assume that $\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})$ is a multivariate normal distribution with $\boldsymbol{\lambda}_{k\ell} = (\boldsymbol{\mu}_{k\ell}, \boldsymbol{\Sigma}_{k\ell})$ where $\boldsymbol{\mu}^\top_{k\ell} = (\mu^1_{k\ell}, \ldots, \mu^v_{k\ell})$ is the mean vector and $\boldsymbol{\Sigma}_{k\ell}$ the covariance matrix of size $v \times v$. The parameter $\boldsymbol{\Omega}$ is formed by $\boldsymbol{\pi}$, $\boldsymbol{\rho}$ and $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_{11}, \ldots, \boldsymbol{\lambda}_{gm})$. Hence, $\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})$ takes the following form:

$$\frac{1}{(2\pi)^{v/2}|\Sigma_{k\ell}|^{0.5}} \exp\left\{-\frac{1}{2}(\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell})^\top \boldsymbol{\Sigma}^{-1}_{k\ell}(\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell})\right\}$$

and,

$$\begin{aligned} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega}) = &\sum_{i,k} z_{ik} \log \pi_k + \sum_{j,\ell} w_{j\ell} \log \rho_\ell \\ &- \frac{1}{2} \sum_{k,\ell} z_{.k} w_{.\ell} \log |\Sigma_{k\ell}| \\ &- \frac{1}{2} \sum_{i,j,k,\ell} z_{ik} w_{j\ell} (\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell})^\top \boldsymbol{\Sigma}^{-1}_{k\ell} (\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell}) \\ &+ constant. \end{aligned} \tag{3}$$

The graphical models of Gaussian LBM and Gaussian TLBM are depicted, respectively, in Fig. 2b, c. With Gaussian LBM, for each block (k, ℓ) $x_{ij} \in \mathbb{R} \sim \mathcal{G}(\mu_{k\ell}, \sigma^2_{k\ell})$, while with Gaussian TLBM, $\mathbf{x}_{ij} \in \mathbb{R}^{v\times 1} \sim \mathcal{G}(\boldsymbol{\mu}_{k\ell}, \boldsymbol{\Sigma}_{k\ell})$ allowing to take into account the covariances between all v variables.

Binary data We can consider an extension of the Bernoulli LBM (Bernoulli TLBM) where, this time, $\boldsymbol{\mu}_{k\ell}$ is a vector of probabilities. Specifically, assuming the conditional independence (independence per block), which is the basis for many statistical models, Φ is given by

$$\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell}) = \prod_{a=1}^{v} (\mu^a_{k\ell})^{x^a_{ij}} (1 - \mu^a_{k\ell})^{1-x^a_{ij}},$$

and the classification log-likelihood can be written as

$$\begin{aligned} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega}) = &\sum_{i,k} z_{ik} \log \pi_k + \sum_{j,\ell} w_{j\ell} \log \rho_\ell \\ &+ \sum_{k\ell} z_{.k} w_{.\ell} \sum_{a=1}^{v} \log(1 - \mu^a_{k\ell}) \\ &+ \sum_{i,j,k,\ell} z_{ik} w_{j\ell} \left(\sum_{a=1}^{v} x^a_{ij} \log \frac{\mu^a_{k\ell}}{1 - \mu^a_{k\ell}} \right) \end{aligned} \tag{4}$$

with $z_{.k} = \sum_i z_{ik}$ and $w_{.\ell} = \sum_j w_{j\ell}$.

Algorithm 1: Generative process of `Tensor LBM` model

Input: $n, d, g, m, \boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\lambda}$

for $i \leftarrow 1$ **to** n **do**
  Generate the row label z_i according to $\mathcal{M}(\pi_1, \ldots, \pi_g)$

for $j \leftarrow 1$ **to** d **do**
  Generate the column label w_j according to $\mathcal{M}(\rho_1, \ldots, \rho_m)$

for $i \leftarrow 1$ **to** n **and** $j \leftarrow 1$ **to** d **do**
  Generate a vector $\mathbf{x}_{ij}$ according to the density $\Phi(\mathbf{x}_{ij}; \lambda_{k\ell})$.

return Tensor matrix $\mathbf{X}$, $\mathbf{z}$ and $\mathbf{w}$

Contingency table (also known as a cross-tabulation). In this case, we consider a particular Poisson TLBM [20] which takes into account, for each slice a, the effects of the margins $x^a_{i.} = \sum_j x^a_{ij}$, $x^a_{.j} = \sum_i x^a_{ij}$ and the block effect $\gamma^a_{k\ell}$. Thereby, each value x^a_{ij} is distributed according to $Poisson(\lambda^a_{ij})$ with $\lambda^a_{ij} = x^a_{i.} x^a_{.j} \sum_{k,\ell} z_{ik} w_{j\ell} \gamma^a_{k\ell}$. Like with Bernoulli TLBM, we assume the conditional independence, Φ, is given by

$$\Phi(\mathbf{x}_{ij};\boldsymbol{\lambda}_{k\ell}) = \prod_{a=1}^{v} \frac{e^{-\lambda_{ij}^a}\lambda_{ij}^{a\,x_{ij}^a}}{x_{ij}^a!},$$

where $\lambda_{ij}^a = x_{i.}^a x_{.j}^a \sum_{k,\ell} z_{ik} w_{j\ell} \gamma_{k\ell}^a$ with the margins $x_{i.}^a = \sum_j x_{ij}^a$ and $x_{.j}^a = \sum_i x_{ij}^a$ and the block effect $\gamma_{k\ell}^a$. Therefore, the parameter $\boldsymbol{\Omega}$ to be estimated is formed by $\boldsymbol{\pi}$, $\boldsymbol{\rho}$ and $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_{11}, \ldots, \boldsymbol{\gamma}_{gm})$ where $\boldsymbol{\gamma}_{k\ell} = (\gamma_{k\ell}^1, \ldots, \gamma_{k\ell}^v)$. The classification log-likelihood (up to a constant) can be written as

$$L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega}) = \sum_{i,k} z_{ik} \log(\pi_k) + \sum_{j,\ell} w_{j\ell} \log \rho_\ell + \sum_{i,j,k,\ell} z_{ik} w_{j\ell} \sum_a \left(-x_{i.}^a x_{.j}^a \gamma_{k\ell}^a + x_{ij}^a \log(\gamma_{k\ell}^a)\right). \tag{5}$$

In Table 1, we repport the expressions of $L_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ according to various distributions. Next, we propose a co-clustering algorithm able to propose solutions for different types of tensors data encountered in practice.

3 Variational EM algorithm

To estimate $\boldsymbol{\Omega}$, the EM algorithm [9] is a candidate for this task. It maximizes the log-likelihood $f(\mathbf{X}, \boldsymbol{\Omega})$ w.r. to $\boldsymbol{\Omega}$ iteratively by maximizing the conditional expectation of the complete data log-likelihood $L_C(\mathbf{z}, \mathbf{w}; \boldsymbol{\Omega})$ w.r. to $\boldsymbol{\Omega}$, given a previous current estimate $\boldsymbol{\Omega}^{(c)}$ and the observed data $\mathbf{X}$. Unfortunately, difficulties arise owing to the dependence structure among the variables x_{ij} of the model. Hence, we propose a variational approach to perform an approximate maximum likelihood inference on the parameters. We follow the strategy used in the co-clustering framework; see, e.g., [16,18,19,24]. This leads to assume the following independence:

$$P(z_{ik} = 1, w_{j\ell} = 1|\mathbf{X}) = \underbrace{P(z_{ik} = 1|\mathbf{X})}_{\tilde{z}_{ik}} \underbrace{P(w_{j\ell} = 1|\mathbf{X})}_{\tilde{w}_{j\ell}}.$$

Thereby, the purpose deals with maximizing the following criterion:

$$F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}; \boldsymbol{\Omega}) = L_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega}) + H(\tilde{\mathbf{z}}) + H(\tilde{\mathbf{w}}) \tag{6}$$

where $\tilde{\mathbf{z}}$, $\tilde{\mathbf{w}}$ are fuzzy matrices, $L_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}; \boldsymbol{\Omega})$ is the fuzzy complete data log-likelihood and

$$\begin{cases} H(\tilde{\mathbf{z}}) = -\sum_{i,k} \tilde{z}_{ik} \log \tilde{z}_{ik} \\ H(\tilde{\mathbf{w}}) = -\sum_{j,\ell} \tilde{w}_{j\ell} \log \tilde{w}_{j\ell}. \end{cases}$$

The maximization of $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ can be reached by realizing the three successive optimizations:

$$\begin{cases} \arg\max_{\tilde{\mathbf{z}}} F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega}), \\ \arg\max_{\tilde{\mathbf{w}}} F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega}), \\ \arg\max_{\boldsymbol{\Omega}} F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega}). \end{cases}$$

In what follows, we detail the expectation (E) and maximization (M) step of the variational EM algorithm for tensor data.

3.1 E-step

The E-step consists in computing, for all i, k, j, ℓ, the posterior probabilities $\tilde{z}_{ik}$ and $\tilde{w}_{j\ell}$ maximizing $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ given the estimated parameters $\boldsymbol{\Omega}_{k\ell}$. It is easy to show that the posterior probability $\tilde{z}_{ik}$ maximizing $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ (see "Appendix A") is given by:

$$\tilde{z}_{ik} \propto \pi_k \exp\left(\sum_{j,\ell} \tilde{w}_{j\ell} \log\left(\Phi(\mathbf{x}_{ij};\boldsymbol{\lambda}_{k\ell})\right)\right).$$

In the same manner, the posterior probability $\tilde{w}_{j\ell}$ is given by:

Table 1 Expression of $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ according to various TLBMs

TLBM	$F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$ up to a constant
Gaussian	$\sum_{i,k} \tilde{z}_{ik} \log(\pi_k) + \sum_{j,\ell} \tilde{w}_{j\ell} \log \rho_\ell + H(\tilde{\mathbf{z}}) + H(\tilde{\mathbf{w}})$ $-\frac{1}{2}\sum_{k,\ell} \tilde{z}_{.k}\tilde{w}_{.\ell} \log\lvert\Sigma_{k\ell}\rvert - \frac{1}{2}\sum_{i,j,k,\ell} \tilde{z}_{ik}\tilde{w}_{j\ell}(\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell})^\top \boldsymbol{\Sigma}_{k\ell}^{-1}(\mathbf{x}_{ij} - \boldsymbol{\mu}_{k\ell})$
Bernoulli	$\sum_{i,k} \tilde{z}_{ik} \log(\pi_k) + \sum_{j,\ell} \tilde{w}_{j\ell} \log \rho_\ell + H(\tilde{\mathbf{z}}) + H(\tilde{\mathbf{w}})$ $+ \sum_{k,\ell} \tilde{z}_{.k}\tilde{w}_{.\ell} \sum_a \log(1 - \mu_{k\ell}^a) + \sum_{i,j,k,\ell} \tilde{z}_{ik}\tilde{w}_{j\ell} \left(\sum_a x_{ij}^a \log \frac{\mu_{k\ell}^a}{1-\mu_{k\ell}^a}\right)$
Poisson	$\sum_{i,k} \tilde{z}_{ik} \log(\pi_k) + \sum_{j,\ell} \tilde{w}_{j\ell} \log \rho_\ell + H(\tilde{\mathbf{z}}) + H(\tilde{\mathbf{w}})$ $+ \sum_{i,j,k,\ell} \tilde{z}_{ik}\tilde{w}_{j\ell} \sum_a \left(-x_{i.}^a x_{.j}^a \gamma_{k\ell}^a + x_{ij}^a \log(\gamma_{k\ell}^a)\right)$

$$\tilde{w}_{j\ell} \propto \rho_\ell \exp\left(\sum_{i,k} \tilde{z}_{ik} \log\left(\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})\right)\right).$$

3.2 M-step

Given the previously computed posterior probabilities $\tilde{\mathbf{z}}$ and $\tilde{\mathbf{w}}$, the M-step consists in updating , $\forall k, \ell$, the parameters π_k, ρ_ℓ, $\boldsymbol{\mu}_{k\ell}$ and $\boldsymbol{\lambda}_{k\ell}$ maximizing $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}, \boldsymbol{\Omega})$. The estimated parameters are defined as follows: First, taking into account the constraints $\sum_k z_{ik} = 1$ and $\sum_\ell w_{j\ell} = 1$, it is easy to show that $\hat{\pi}_k = \frac{\sum_i \tilde{z}_{ik}}{n} = \frac{\tilde{z}_{.k}}{n}$ and $\hat{\rho}_\ell = \frac{\sum_j \tilde{w}_{j\ell}}{d} = \frac{\tilde{w}_{.\ell}}{d}$. Secondly, the update of $\boldsymbol{\lambda}_{k\ell}$ depends on the choice of Φ.
Gaussian TLBM. With this model, $\boldsymbol{\lambda}_{k\ell}$ is formed by $(\boldsymbol{\mu}_{k\ell}, \boldsymbol{\Sigma}_{k\ell})$ where $\boldsymbol{\mu}_{k\ell}$ is the mean vector and $\boldsymbol{\Sigma}_{k\ell}$ the covariance matrix. It is easy to show that the estimation of mean vector $\hat{\boldsymbol{\mu}}_{k\ell}$ is given by $\frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}\mathbf{x}_{ij}}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}}$, and thereby deduce:

$$\hat{\boldsymbol{\Sigma}}_{k\ell} = \frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}(\mathbf{x}_{ij} - \hat{\boldsymbol{\mu}}_{k\ell})(\mathbf{x}_{ij} - \hat{\boldsymbol{\mu}}_{k\ell})^\top}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}}.$$

(see "Appendix B"). Bernoulli TLBM. It easy to show that the update of $\boldsymbol{\lambda}_{k\ell}$ can be performed by the update of $\lambda^a_{k\ell}$'s separately. Hence, from (4), for each triplet (k, ℓ, a) the partial derivative of

$$z_{.k}w_{.\ell}\log(1-\mu^a_{k\ell}) + \sum_{i,j} z_{ik}w_{j\ell}\left(x^a_{ij}\log\frac{\mu^a_{k\ell}}{1-\mu^a_{k\ell}}\right)$$

set to 0 leads to $\hat{\mu}^a_{k\ell} = \frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}x^a_{ij}}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}}$. Hence, $\boldsymbol{\lambda}_{k\ell}$ which is a probability vector is given by $\frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}\mathbf{x}_{ij}}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}}$.
Poisson TLBM. Similarly, we update $\gamma^a_{k\ell}$ $\forall k, \ell, a$. We have

$$\begin{aligned}\hat{\gamma}^a_{k\ell} &= \arg\max_{\gamma_{k\ell}} \sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}(x^a_{ij}\log\gamma_{k\ell} - x^a_{i.}x^a_{.j}\gamma_{k\ell}) \\ &= \frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}x^a_{ij}}{\sum_i z_{ik}x^a_{i.}\sum_j w_{j\ell}x^a_{.j}} = \frac{x^a_{k\ell}}{x^a_{k.}x^a_{.\ell}}.\end{aligned}$$

where

$$x^a_{k\ell} = \sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}x^a_{ij},\ x^a_{k.} = \sum_i \tilde{z}_{ik}x^a_{i.},\ x^a_{.\ell} = \sum_j \tilde{w}_{j\ell}x^a_{.j}.$$

The proposed algorithm referred to as VEM-T in Algorithm 2 alternates the two previously described steps expectation–maximization. At the convergence, a hard co-clustering is deduced from the posterior probabilities.

Algorithm 2: VEM-T

Input: $\mathbf{X}$, g, m.
Initialization $(\mathbf{z}, \mathbf{w})$ randomly, compute $\boldsymbol{\Omega}$
repeat
  E-Step
  – **Compute** $\tilde{z}_{ik}$ **using**
    $\tilde{z}_{ik} \propto \pi_k \exp\left(\sum_{j,\ell} \tilde{w}_{j\ell}\log\left(\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})\right)\right)$
  – **Compute** $\tilde{w}_{j\ell}$ **using**
    $\tilde{w}_{j\ell} \propto \rho_\ell \exp\left(\sum_{i,k} \tilde{z}_{ik}\log\left(\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})\right)\right)$
  M-Step
  Update $\boldsymbol{\Omega}$
until *convergence*;
return z, w, Ω

4 Classification maximum likelihood approach

4.1 CEM-T algorithm

Another likelihood-based approach to clustering besides the mixture likelihood is what is sometimes called the *classification maximum likelihood* (CML) approach [7,8,14]. Unlike the *maximum likelihood* (ML) approach which aims to maximize log-likelihood, with the CML approach, $(\mathbf{z}, \mathbf{w}, \Omega)$ are chosen to maximize the complete data log-likelihood $L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega})$ (2). Doing so, the maximization can be obtained by alternating the three following computations: $\arg\max_{\mathbf{z}} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega})$, $\arg\max_{\mathbf{w}} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega})$, and $\arg\max_{\boldsymbol{\Omega}} L_C(\mathbf{z}, \mathbf{w}, \boldsymbol{\Omega})$. These optimizations can be performed by using the classification EM algorithm proposed in [18]. It is a direct clustering algorithm which consists in inserting a classification step (C-step) between E-step and M-step. The principal steps of the algorithm, which we refer as CEM-T, are reported in Algorithm 3. Note that in M-step all the update formulas can be used by replacing $\tilde{z}_{ik}$ by $z_{ik} \in \{0, 1\}$ and $\tilde{w}_{j\ell}$ by $w_{j\ell} \in \{0, 1\}$. In other words, the update is done by co-cluster.

Note with the CML approach, we can establish some connections with popular algorithms. Next, we show the connection in the case of contingency tables.

4.2 CEM-T for contingency tables

When we consider Poisson TLBM, we have seen that the computation of $\hat{\boldsymbol{\gamma}}^a = \{\hat{\gamma}_{k\ell} | k = 1, \ldots, g; \ell = 1, \ldots, m; a = 1, \ldots, v\}$ maximizing L_C leads to $\hat{\boldsymbol{\gamma}}^a_{k\ell} = \frac{x^a_{k\ell}}{x^a_{k.}x^a_{.\ell}}$ for all a, k, ℓ. Then, plugging $\hat{\gamma}^a_{k\ell}$ in (5), the complete data log-likelihood becomes

$$L_C(\mathbf{z}, \mathbf{w}, \hat{\boldsymbol{\gamma}}) = \sum_{a=1}^{v}\sum_{k=1}^{g}\sum_{\ell=1}^{m} x^a_{k\ell}\log\frac{x^a_{k\ell}}{x^a_{k.}x^a_{.\ell}} - \sum_{a=1}^{v} N^a \qquad (7)$$

Algorithm 3: CEM-T

Input: $\mathbf{X}$, g, m.
Initialization ($\mathbf{z}$, $\mathbf{w}$) randomly, compute $\boldsymbol{\Omega}$
repeat
  E-Step:
  – **Compute** $\tilde{z}_{ik}$ **using**
    $\tilde{z}_{ik} \propto \pi_k \exp\left(\sum_{j,\ell} \tilde{w}_{j\ell} \log\left(\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})\right)\right)$
  – **Compute** $\tilde{w}_{j\ell}$ **using**
    $\tilde{w}_{j\ell} \propto \rho_\ell \exp\left(\sum_{i,k} \tilde{z}_{ik} \log\left(\Phi(\mathbf{x}_{ij}; \boldsymbol{\lambda}_{k\ell})\right)\right)$

  C-Step:

  – **Compute** $z_{ik} = \arg\max_k \tilde{z}_{ik}$
  – **Compute** $w_{j\ell} = \arg\max_\ell \tilde{w}_{j\ell}$

  M-Step:
  Update $\boldsymbol{\Omega}$
until *convergence*;
return z, w, Ω

where $N^a = \sum_{i,j} x^a_{ij}$. The distribution that can be associated with $\mathbf{z}$ and $\mathbf{w}$ is the distribution defined by $p^a_{k\ell} = \frac{x^a_{k\ell}}{N^a}$ for all a, k, ℓ. The row and column margins are, respectively, defined by $p^a_{k.} = \frac{x^a_{k.}}{N^a}$ and $p^a_{.\ell} = \frac{x^a_{.\ell}}{N^a}$. Plugging these expressions in (7), the complete data log-likelihood can be expressed as follows:

$$\sum_{a=1}^{v} N^a \sum_{k=1}^{g}\sum_{\ell=1}^{m} p^a_{k\ell} \log \frac{p^a_{k\ell}}{p^a_{k.}p^a_{.\ell}} - \sum_{a=1}^{v} N^a(1+\log N^a),$$

and its maximization is equivalent to the maximization of the *total mutual information* $\sum_{a=1}^{v}\sum_{k,\ell} p^a_{k\ell} \log \frac{p^a_{k\ell}}{p^a_{k.}p^a_{.\ell}}$ or the minimization of the loss in mutual information due to co-clustering, i.e.,

$$\sum_{a=1}^{v}\sum_{i=1}^{n}\sum_{j=1}^{d} p^a_{ij} \log \frac{p^a_{ij}}{p^a_{i.}p^a_{.j}} - \sum_{a=1}^{v}\sum_{k=1}^{g}\sum_{\ell=1}^{m} p^a_{k\ell} \log \frac{p^a_{k\ell}}{p^a_{k.}p^a_{.\ell}}. \quad (8)$$

Note that for $v = 1$, (8) is the objective function optimized by ITCC [11] or the Croinfo algorithm [29]. Hence, the CEM-T algorithm can be viewed as a model-based clustering version of ITCC/Croinfo where the proportions of row clusters (resp. column clusters) are assumed to be equal; see, for instance, [2,20].

5 Experimental results

5.1 Evaluation in terms of clustering

The evaluation of co-clustering is generally carried out based on benchmarks datasets where only one of the two partitions is known. In the same way, we compare VEM-T with competitive (co)-clustering methods. We retain three widely used measures to assess the quality of clustering, namely the accuracy, the normalized mutual information (NMI) [33] and the adjusted Rand index (ARI) [32].

The clustering accuracy noted (Acc) discovers the one-to-one relationship between two partitions and measures the extent to which each cluster contains data points from the corresponding class. It is defined as follows:

$$\text{Acc} = \frac{1}{n}\sum_{i=1}^{n} \delta(\mathcal{C}_i, map(\mathcal{P}_i))$$

where n is the total number of samples, $\mathcal{P}_i$ is the i^{th} obtained cluster, and $\mathcal{C}_i$ is the true i^{th} class provided by the dataset. $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $map(\mathcal{P}_i)$ is the permutation mapping function that maps the obtained label $\mathcal{P}_i$ to the equivalent label from the dataset. The best mapping can be found by using the Kuhn–Munkres algorithm [5,27]. The NMI measure is estimated by

$$\text{NMI} = \frac{\sum_{k,\ell} \frac{n_{k\ell}}{n} \log \frac{n n_{k\ell}}{n_k \hat{n}_\ell}}{\sqrt{(\sum_k n_k \log \frac{n_k}{n})(\sum_\ell \hat{n}_\ell \log \frac{\hat{n}_\ell}{n})}}$$

where n_k denotes the number of data contained in cluster $\mathcal{C}_k (1 \le k \le K)$, $\hat{n}_\ell$ is the number of data belonging to the class $\mathcal{L}_\ell (1 \le \ell \le K)$, and $n_{k\ell}$ denotes the number of data that are in the intersection between cluster $\mathcal{C}_k$ and class $\mathcal{L}_\ell$. Intuitively, NMI quantifies how much the estimated clustering is informative about the true clustering.

The ARI measure quantifies the similarity between two data clustering partitions. From a mathematical standpoint, the Rand index is related to accuracy. The adjusted form of the Rand Index is:

$$\text{ARI} = \frac{\sum_{k,\ell}\binom{n_{k\ell}}{2} - \left[\sum_k \binom{n_k}{2}\sum_\ell \binom{\hat{n}_\ell}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_k \binom{n_k}{2} + \sum_\ell \binom{\hat{n}_\ell}{2}\right] - \left[\sum_k \binom{n_k}{2}\sum_\ell \binom{\hat{n}_\ell}{2}\right]/\binom{n}{2}}.$$

The ARI is related to the clustering accuracy and measures the degree of agreement between an estimated clustering and a reference clustering. Both NMI and ARI are equal to 1 if the resulting clustering is identical to the true one. Secondly, we present results on real datasets for three different areas, namely recommender systems, multi-spectral images clustering, and documents categorization. Through

Table 2 Evaluation of co-clustering in terms of NMI for binary datasets

Algorithms	Metric		Example 1 $400 \times 400 \times 3$ $(g, m) = (4, 4)$ $\boldsymbol{\pi} = [0.23, 0.3, 0.23, 0.24]$ $\boldsymbol{\rho} = [0.27, 0.23, 0.3, 0.2]$			Example 2 $400 \times 400 \times 3$ $(g, m) = (4, 4)$ $\boldsymbol{\pi} = [0.23, 0.3, 0.23, 0.24]$ $\boldsymbol{\rho} = [0.27, 0.23, 0.3, 0.2]$		
	NMI		Slice 1	Slice 2	Slice 3	Slice 1	Slice 2	Slice 3
K-means	NMI	Row	0.80 ± 0.00	0.81 ± 0.00	0.98 ± 0.00	0.83 ± 0.00	0.82 ± 0.00	0.94 ± 0.00
		Column	0.83 ± 0.01	0.76 ± 0.00	0.76 ± 0.00	0.83 ± 0.012	0.80 ± 0.00	0.80 ± 0.00
EMgmm	NMI	Row	0.81 ± 0.00	0.81 ± 0.00	1.00 ± 0.00	0.82 ± 0.00	0.82 ± 0.00	0.90 ± 0.02
		Column	0.79 ± 0.02	0.77 ± 0.01	0.76 ± 0.01	0.83 ± 0.01	0.88 ± 0.00	0.83 ± 0.01
VEM	NMI	Row	0.66 ± 0.00	0.72 ± 0.00	0.78 ± 0.01	0.71 ± 0.00	0.73 ± 0.00	0.86 ± 0.01
		Column	0.70 ± 0.00	0.71 ± 0.00	0.71 ± 0.00	0.72 ± 0.00	0.73 ± 0.00	0.80 ± 0.00
VEM-T	NMI	Row		0.94 ± 0.00			0.90 ± 0.00	
		Column		0.93 ± 0.00			0.97 ± 0.00	

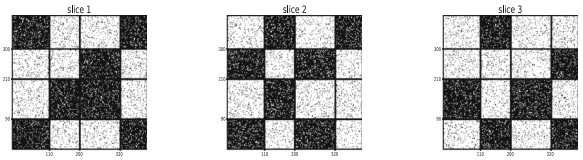
Table 3 Evaluation of co-clustering in terms of NMI for continuous datasets

Algorithms	Metric		Example 3 $200 \times 200 \times 3$ $(g, m) = (3, 2)$ $\boldsymbol{\pi} = [0.3, 0.35, 0.35]$ $\boldsymbol{\rho} = [0.55, 0.45]$			Example 4 $500 \times 500 \times 3$ $(g, m) = (3, 3)$ $\pi = [0.34, 0.34, 0.32]$ $\rho = [0.28, 0.34, 0.38]$		
	NMI		Slice 1	Slice 2	Slice 3	Slice 1	Slice 2	Slice 3
K-means	NMI	Row	1.0 ± 0.0	0.62 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.09 ± 0.0	1.0 ± 0.0
		Column	1.0 ± 0.0	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.29 ± 0.01	1.0 ± 0.0
EMgmm	NMI	Row	1.0 ± 0.0	0.62 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.15 ± 0.0	1.0 ± 0.00
		Column	1.0 ± 0.0	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	0.24 ± 0.01	1.0 ± 0.00
VEM	NMI	Row	0.98 ± 0.0	0.77 ± 0.0	0.98 ± 0.0	0.95 ± 0.01	0.50 ± 0.0	1.0 ± 0.00
		Column	1.0 ± 0.0	0.0 ± 0.0	1.0 ± 0.0	0.95 ± 0.01	0.60 ± 0.0	0.95 ± 0.01
VEM-T	NMI	Row		1.0 ± 0.00			0.95 ± 0.01	
		Column		1.0 ± 0.00			0.95 ± 0.00	

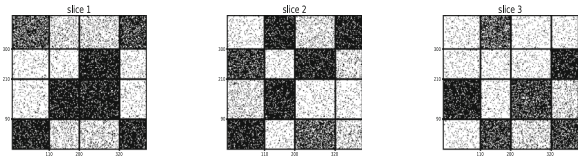
this evaluation, we aim to demonstrate the impact of covariate information on interpretation and improvement of clustering results.

5.2 Synthetic datasets and competitive methods

Before proceeding to evaluate VEM-T on real datasets, we give here two simple illustrative examples. We generated tensor data $\mathbf{X}$ according to the Bernoulli and Gaussian TLBM (Algorithm 1) with $v = 3$. Following each model, we considered two scenarios by varying the centers $\boldsymbol{\mu}_{k\ell}$'s, an example where the co-clusters are well separated and the other where the co-clusters are not. The size of each tensor, number of co-clusters, and their proportions are reported in Tables 2 and 3. Herein other characteristics of each tensor dataset: For continuous data, we take the same covariance matrix for all blocks $\begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}$ for example 3 and $\begin{bmatrix} 1 & 0.8 & 0.8 \\ 0.8 & 1 & 0.8 \\ 0.8 & 0.8 & 1 \end{bmatrix}$ for example 4. All variables (slice) are standardized to have values between zero and one. In Figs. 4 and 3 are depicted the true simulated tensor data into $v = 3$ slices.

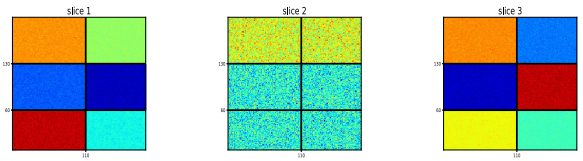


Example 1

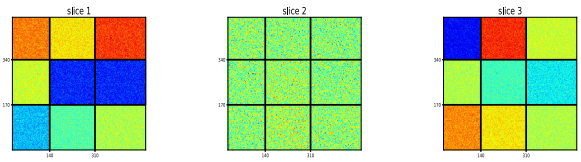


Example 2

Fig. 3 Simulated binary datasets



Example 3



Example 4

Fig. 4 Simulated continuous datasets

In our experiments, we compare VEM-T with K-means, Gaussian mixture model (EMgmm: EM with the full model, see, for instance, [13]) and VEM for co-clustering applied on each slice [17].

The NMI metric computed according to row and column clusters is obtained by averaging ten random initializations. Thereby, in Tables 2 and 3 are reported the performances for the three slices obtained by K-means, EMgmm, VEM for data matrix and by VEM-T. From these comparisons, we observe that whatever the block structure, easy to identify (Examples 1,3) or not (Examples 2,4), the VEM-T outperforms the other algorithms applied on each slice separately.

6 Real datasets

6.1 Recommender system application

To show the benefits of our approach, we use the binary model on Movielens100K, which is one of the more popular datasets on the recommender system field. The objective of this study is to identify patterns according to users and movies characteristics. The Movielens100K[1] database consists of 100,000 ratings of 943 users and 1682 movies, where each user has rated at least 20 movies. We convert the users–movies rating matrix (943 × 1682) to binary matrix by assigning 0 to the movie without rating and 1 to rated movies. This binary matrix can be considered as a viewing matrix; in fact, most users rate movies after watching them. Furthermore, Movielens includes 22 user covariates, including age, gender, and 21 employment status. The age covariate is used to analyze clustering results and does not take into account co-clustering. There are also 19 movie covariates related to movie genres, considering that movie may belong to one or more genres. The data structure can be represented as tensor with size 943 × 1682 × 42. The objective of this work is not being to select the number of clusters; then, we fixed the number of row clusters $g = 2$ and the number of column clusters $m = 3$, based on the works of [34].

Figures 5 and 6a represent the mean vectors $\boldsymbol{\mu}_{k\ell}$ and co-clustering of rating matrix, respectively. We observe two row clusters, a smaller cluster of 202 users which is more active in reviewing than a second large cluster. On the other hand, we obtain three movies clusters of different sizes 232, 355, and 1095, respectively. The first cluster represents the most attractive movies.

The first row cluster includes three blocks (1, 1), (1, 2), and (1, 3). The two first ones represent the more active users with a higher proportion of rating. The MovieLens100K dataset includes 29% of female reviews, an important part of them (64%) belong to a first row cluster. In addition, we notice that the top 3 of occupations for users of the first row cluster are a student, educator, and administrator. Figure 6b shows that 65% of them are quite young and under 31 years of age. However, the two blocks (1, 1) and (1, 2) are distinguished by movie genres, since the top 3 ones for first and second column clusters are Action-Thriller-Sci-Fi and Comedy-Drama-Romance, respectively. Hence, we can identify two profiles of young active users; the first are attracted by both categories of movies, namely Action-Thriller-Sci-Fi, while the second by Comedy-Drama-Romance. The second row cluster regroups the users of different ranges of age with almost equal proportions (see Fig. 6b), and different occupations since the top three occupations include engineer, student, and another employment status. Finally, the third column cluster seems to represent movies with different genres of Action-Drama-Comedy. The block (2,3) represents the less attractive movies watched by the less active users.

[1] http://grouplens.org/datasets/movielens/.

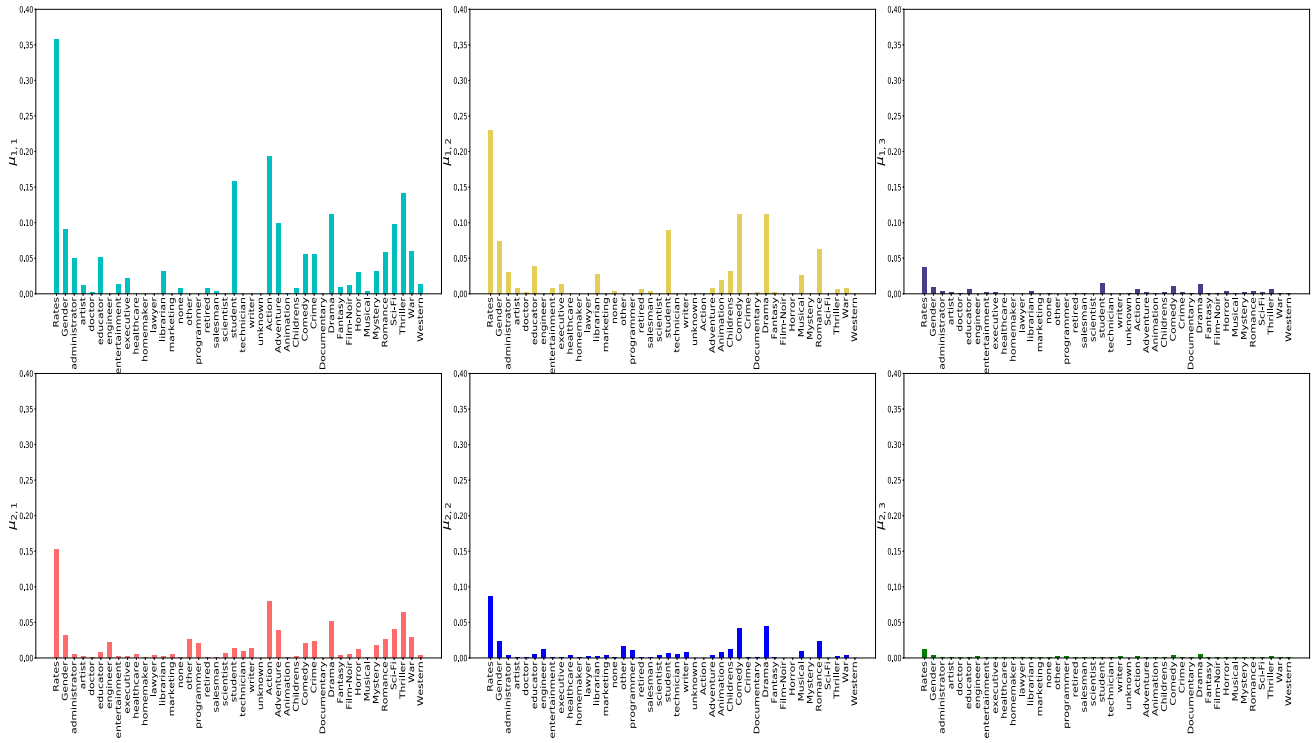
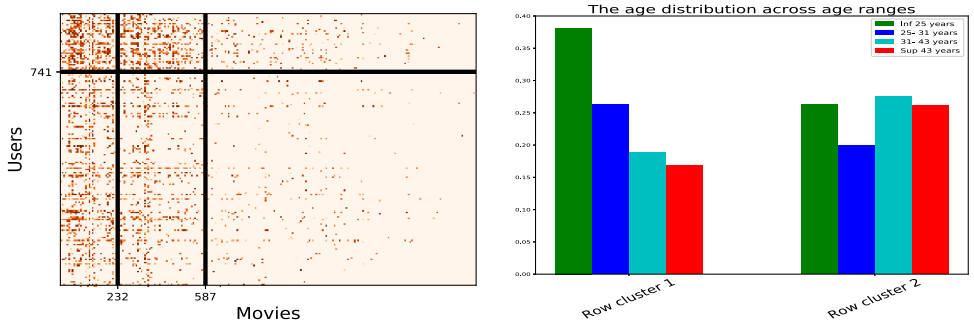


Fig. 5 Distribution of the centers $\boldsymbol{\mu}_{k\ell}$ for all co-clusters



(a) **(b)**

Fig. 6 **a** Co-clustering data matrix, **b** distribution of age per row clusters

6.2 Multi-spectral images analysis

The used dataset is composed of 37 multi-spectral images [36] of prostate cells with 16 bands which have size 512 × 512 pixels. Several studies showed that clustering accuracy increases according to bands number [23]. The four types of multi-spectral images cells are: normal cells (stroma), benign hyperplasia (BHp), intraepithelial neoplasia (PIN) which is a precursory cancer state, and the carcinoma (CA) which corresponds to cancer of the abnormal tissue proliferation. Figure 7a presents cell's types, and the examples of 16 bands of stroma cells type are shown in Fig. 7b. Some elements allow differentiating the cell's types, among those morphological and textural features. In this way, we limited ourselves to textural characteristics for clustering. Haralick [21] defined several metrics computed from the gray-level co-occurrence matrix (GLCM). The Haralick's parameters showed their efficiency in the literature for the textures analysis [21,23]. The 14 Haralick's features are the following: energy, correlation, contrast, entropy, homogeneity, inverse difference moment, sum average, sum variance, sum entropy, difference average, difference variance, difference entropy, and two information measure of correlation.

In the most previous studies, the extraction of 14 Haralick's features from all bands is performed, and the 14 × 16 features are extracted for each image involving features selection or dimensionality reduction with popular methods such as PCA. These operations can provide interesting results, but lead to a loss of information. To overcome this drawback, we propose to construct tensor data $Images \times Bands \times$

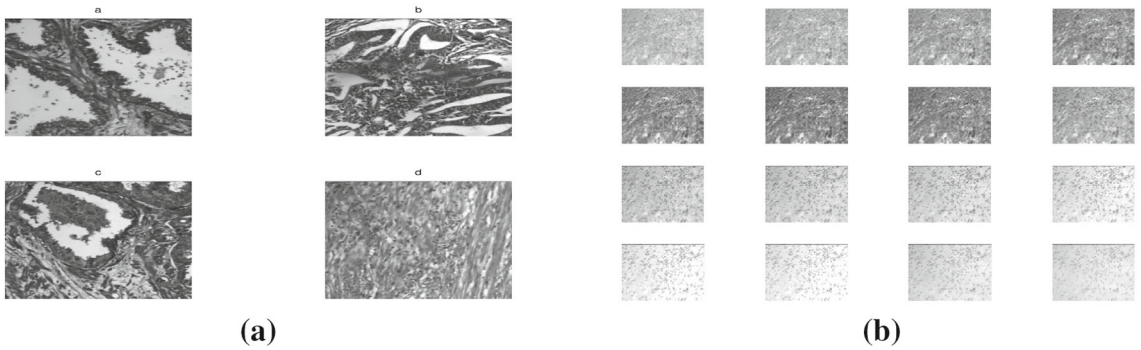


Fig. 7 **a** The four cells type, **b** example of multi-spectral image from dataset

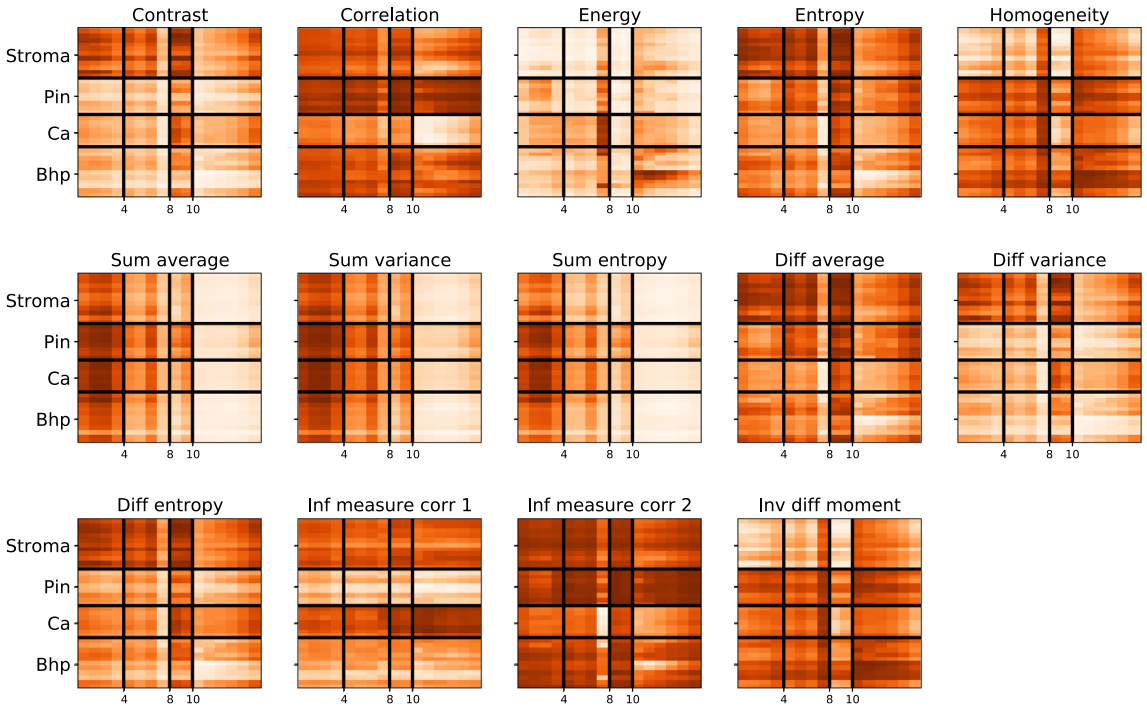


Fig. 8 Co-clustering matrix of different slice of features

Table 4 Evaluation of `K-means`, `EMgmm`, `VEM`, and `VEM-T` in terms of `NMI`, `ARI`, and `ACC`

Algorithms	NMI	ARI	ACC
`K-means`	0.67	0.56	0.78
`EMgmm`	0.7	0.59	0.78
`VEM`	0.61	0.49	0.7
`VEM-T`	0.90	0.87	0.95

Features in order to exploit all available data without requiring dimensionality reduction. The objective of this study is improving the clustering results of multi-spectral images which highly used in biomedical and geology fields.

As we knew the true number of image clusters, we take $g = 4$, and as we have no information about column clusters, we postulate $m = g = 4$. As shown in Fig. 8, the stroma cells are characterized by higher values of entropy, contrast, and difference variance on the first three column clusters, and low values of inverse difference moment feature on two first band clusters. The PIN type is characterized by low values of information measure correlation 1 on bands cluster 2, 3, and 4. The cell type with closer values of features is BHP. The CA type is characterized by higher values of information measure correlation 1 on the third and fourth band clusters and the lower values of information measure correlation 2 on all bands. Finally, BHP cells are characterized by the lowest values of sum average on two last bands clusters.

The `VEM-T` algorithm is compared with `K-means`, `EMgmm`, and `VEM`. For this, a reduced matrix of tensor

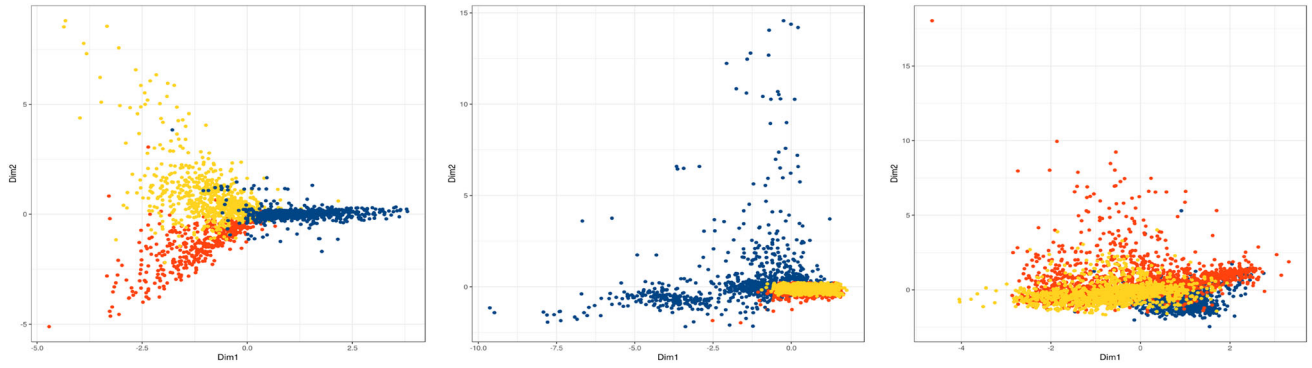


Fig. 9 Obtained results using multiple factor analysis

data by averaging all bands for each feature provides a $Images \times Features$ data matrix used to perform classical clustering. Table 4 summarizes the obtained results. For each algorithm, the best result rather than 100 random initial runs is used. Clearly, the proposed algorithm achieves best results in terms of NMI, ARI, and ACC (accuracy).

6.3 Document categorization

In our experiments, we aim to evaluate VEM-T for contingency tables in terms of clustering leading to measure the impact of mixing different information. Thereby, we compare VEM-T with Spherical K-means, Itcc [10,11], and VEM-T$_a$ applied on each slice (a) of tensor and three other algorithms applied on tensor data, namely PARAFAC [22] and GTSC [35]. Note that PARAFAC is used with ranks number equal to 10 and followed by K-means. We perform 50 random initializations and compute the ACC, ARI, and NMI metrics by averaging the ten top runs.

6.3.1 Datasets

We use three text datasets DBLP1, DBLP2, and PubMed Diabetes[2] to highlight the objective of the proposed algorithm. DBLP1 and DBLP2 are constructed from DBLP,[3] by selecting three journals for each one. The selected journals for DBLP1 are SIGMOD, STOC, and SIGIR. The journals selected for DBLP2 are Discrete Applied Mathematics, IEEE software, and SIGIR. For PubMed Diabetes dataset, the papers are categorized into three types, the first one deals with diabetes mellitus of type 1, the second with diabetes mellitus of type 2, and the third with diabetes mellitus experimental.

From these different datasets, we construct the following adjacency matrices:

- Co-terms matrix on the title: each cell represents the number of times that a term is present simultaneously in the title of a pair of papers. This matrix is computed using $\mathcal{T}\mathcal{T}^T$ where $\mathcal{T}$ is a binarized document-terms matrix.
- Co-terms matrix on the abstract: each cell represents the number of times that a pair of papers share a term extracting from abstract. We use the same process used in *co-terms title matrix*.
- Co-authors matrix: each cell represents the number of common authors for a pair of papers. This matrix is computed using $\mathcal{A}\mathcal{A}^T$ where $\mathcal{A}$ is a binarized document-authors matrix.
- Citations matrix: it is a binary data matrix where 1 indicates the presence of a citation between two papers.

The constructed tensor ($Paper \times Paper \times Relation$) for each dataset DBLP1, DBLP2, and PubMed Diabetes has, respectively, size ($2223 \times 2223 \times 4$), ($1949 \times 1949 \times 4$), and ($4354 \times 4354 \times 4$) and different rates of sparsity 0.93, 0.94, and 0.69, respectively. Plots in Fig. 9 represent the low-dimensional projection of papers from tensor data of DBLP1, DBLP2, and PubMed Diabetes, respectively, using the *multiple factor analysis* (MFA). MFA deals with multiple tables where the slices are contingency tables [28]. We notice that the three datasets have a different degree of complexity (Fig. 10).

6.3.2 Evaluation of VEM-T

In Fig. 11 are reported the performances of the six algorithms (cited above) on the three datasets. In terms of ACC, NMI, and ARI, we observe in most cases that VEM-T is better than other algorithms applied on each slice and those applied on tensor data. With PubMed Diabetes which is the least sparse dataset, we obtain the lowest results for the three measures ACC, NMI, and ARI due to the complex structure of dataset appearing in Fig. 9c. Further note that GTSC, less effective than VEM-T, reaches better results

[2] https://linqs.soe.ucsc.edu/data.

[3] https://aminer.org/citation.

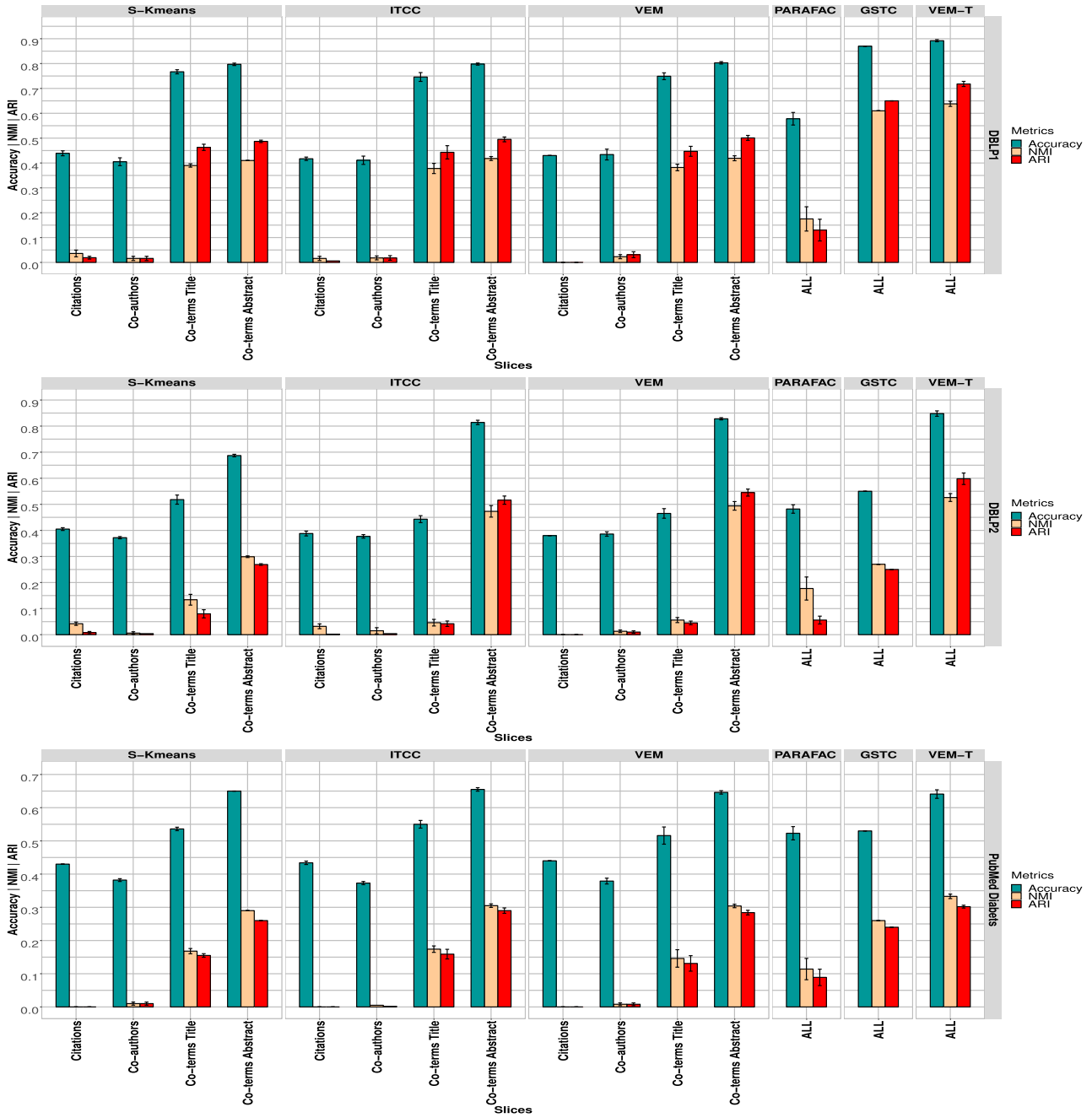


Fig. 10 Comparaison_Results_TPLBM_time

than PARAFAC, followed by K-means. We can notice that VEM-T$_a$ applied on each slice does a good job on the well-separated slices like co-terms Title and co-terms Abstract. Finally, we can say that the VEM-T with considering all slices (the well-separated one and the ill-separated one) can find the best trade-off in terms of clustering results.

Furthermore, Fig. 12 shows the behavior of the $\gamma_{k\ell}^{[.]}$ parameter for each block at each iteration, for the three datasets. $\gamma_{k\ell}^{[.]}$ is computed at each iteration by averaging all $\gamma_{k\ell}^{a}$ as $\gamma_{k\ell}^{[.]} = \frac{1}{v}\sum_{a=1}^{v}\gamma_{k\ell}^{a}$. Interestingly, for DBLP1 and DBLP2, we can see that while the average of diagonal parameters $\gamma_{kk}^{[.]}$ increases, the value of the parameter $\gamma_{k\ell}^{[.]}$, where $k \neq \ell$, decreases at each iteration. For these two datasets, we have three well-separated clusters on diagonal which explain that $\gamma_{kk}^{[.]}$ increases perfectly and $\gamma_{k\ell}^{[.]}$ where $k \neq \ell$ also decreases perfectly. For PubMed Diabets, the data structure seems more complicated, and then, the interpretation of gamma evolution is more complicated. We

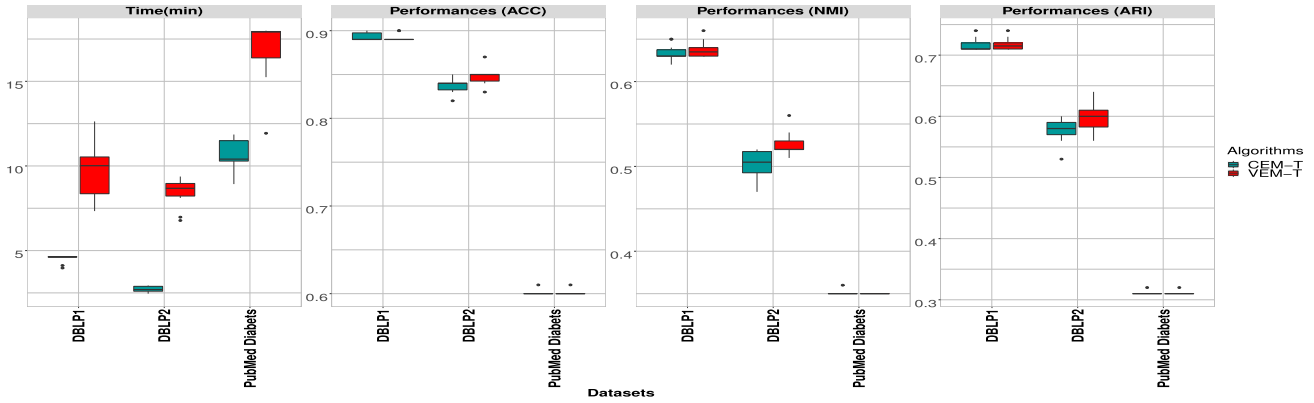


Fig. 11 Comparison of CEM−T and VEM−T in terms of time complexity and performances

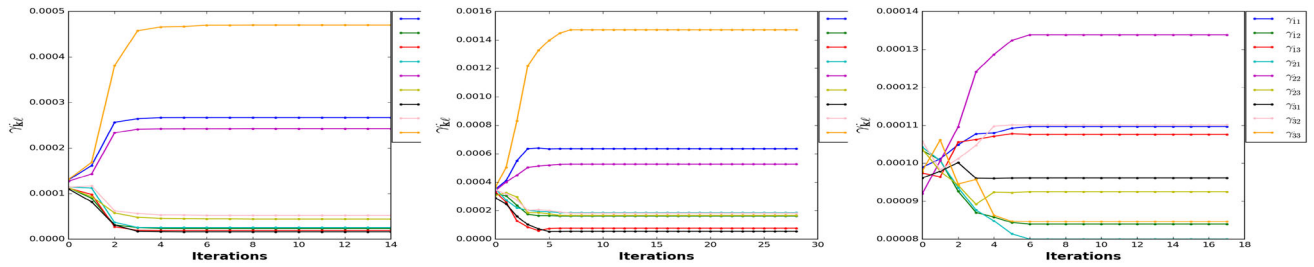


Fig. 12 Behavior of the $\boldsymbol{\gamma}_{kl}$ parameters at each iteration

can see that $\gamma_{k\ell}^{[.]}$ increases for four blocks and decreases otherwise.

6.3.3 Comparison of VEM−T and CEM−T

As we have seen CEM−T is a hard version of VEM−T, what is then its behavior in terms of computational time and clustering performance? Thereby, in Fig. 10 we report the comparison between CEM−T and VEM−T with the three datasets. The CEM−T algorithm is faster than VEM−T, but in terms of clustering performances (accuracy, NMI, and ARI), we can see that VEM−T is at least equivalent.

7 Conclusion

Inspired by the flexibility of the latent block model (LBM), we propose to provide a tensor version (TLBM). This gives rise to a new variational EM algorithm for co-clustering of different types of data. Empirical results on synthetic and real-world datasets—binary, continuous and contingency tables—show that VEM−T and its hard version CEM−T do a better job than other algorithms devoted to the same task or other algorithms applied on each slice of tensor data. More interestingly, our findings open up good opportunities for future research such as dealing with temporal data or challenges such as assessing the number of co-clusters.

A Appendix: Update $\tilde{z}_{ik}$ and $\tilde{w}_{j\ell}$ $\forall i, k, j, \ell$

To obtain the expression of $\tilde{z}_{ik}$, we maximize the above soft criterion $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}; \Omega)$ with respect to $\tilde{z}_{ik}$, subject to the constraint $\sum_k \tilde{z}_{ik} = 1$. The corresponding Lagrangian, up to terms which are not function of $\tilde{z}_{ik}$, is given by :

$$L(\tilde{\mathbf{z}}, \beta) = \sum_{i,k} \tilde{z}_{ik} \log \pi_k + \sum_{i,j,k,\ell} \tilde{z}_{ik} \tilde{w}_{jk} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k\ell})) - \sum_{i,k} \tilde{z}_{ik} \log(\tilde{z}_{ik}) + \beta(1 - \sum_{k} \tilde{z}_{ik}). \quad (9)$$

Taking derivatives with respect to $\tilde{z}_{ik}$, we obtain:

$$\frac{\partial L(\tilde{\mathbf{z}}, \beta)}{\partial \tilde{z}_{ik}} = \log \pi_k + \sum_{j,\ell} w_{j\ell} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k\ell})) - \log \tilde{z}_{ik} - 1 - \beta.$$

Setting this derivative to zero yields:

$$\tilde{z}_{ik} = \frac{\pi_k \exp(\sum_{j,\ell} w_{j\ell} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k\ell}))}{\exp(\beta + 1)}.$$

Summing both sides over all k' yields

$$\exp(\beta+1) = \sum_{k'} \pi_{k'} \exp(\sum_{j,\ell} w_{j\ell} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k'\ell})).$$

Plugging $\exp(\beta)$ in $\tilde{z}_{ik}$ leads to:

$$\tilde{z}_{ik} \propto \pi_k \exp(\sum_{j,\ell} w_{j\ell} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k\ell})).$$

In the same way, we can estimate $\tilde{w}_{jk}$ maximizing $F_C(\tilde{\mathbf{z}}, \tilde{\mathbf{w}}; \Omega)$ with respect to $\tilde{w}_{j\ell}$, subject to the constraint $\sum_\ell \tilde{w}_{j\ell} = 1$; we obtain

$$\tilde{w}_{j\ell} \propto \rho_k \exp(\sum_{i,k} \tilde{z}_{ik} \log(\Phi(\mathbf{x}_{ij}, \boldsymbol{\lambda}_{k\ell})).$$

B Appendix: Estimation of the $\mu_{k\ell}$'s and $\Sigma_{k\ell}$'s of Gaussian TLBM

The $\mu_{k\ell}$'s and $\Sigma_{k\ell}$'s can be obtained from the following derivatives:

$$\frac{\partial \mathcal{F}_C^{k\ell}}{\partial \boldsymbol{\mu}_{k\ell}} = \frac{\partial \mathcal{L}_C^{k\ell}}{\partial \boldsymbol{\mu}_{k\ell}} \quad \text{and} \quad \frac{\partial \mathcal{F}_C^{k\ell}}{\partial \boldsymbol{\Sigma}_{k\ell}} = \frac{\partial \mathcal{L}_C^{k\ell}}{\partial \boldsymbol{\Sigma}_{k\ell}}$$

where

$$\begin{aligned}\mathcal{L}_C^{k\ell} = &-\frac{1}{2}\tilde{z}_{.k}\tilde{w}_{.\ell} \log|\Sigma_{k\ell}| \\ &-\frac{1}{2}\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell})^\top \boldsymbol{\Sigma}_{k\ell}^{-1}(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell}),\end{aligned}$$

with $\tilde{z}_{.k} = \sum_i \tilde{z}_{ik}$ and $\tilde{w}_{.\ell} = \sum_j \tilde{w}_{j\ell}$. The following formulas involving the vector-by-vector ($\mathbf{x}$) and matrix-by-matrix ($\mathbf{M}$) derivatives

$$\begin{cases} \dfrac{\partial \mathbf{x}^\top \mathbf{M}\mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{M}\mathbf{x}, \\ \dfrac{\partial \log|\mathbf{M}|}{\partial \mathbf{M}} = (\mathbf{M}^{-1})^\top, \\ \dfrac{\partial \mathbf{x}^\top \mathbf{M}\mathbf{x}}{\partial \mathbf{M}} = (\mathbf{M}^{-1})\mathbf{x}\mathbf{x}^\top(\mathbf{M}^{-1})^\top \end{cases}$$

lead to

$$\frac{\partial \mathcal{L}_C^{k\ell}}{\partial \boldsymbol{\mu}_{k\ell}} = -\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}\boldsymbol{\Sigma}_{k\ell}^{-1}(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell})$$

and

$$\begin{aligned}\frac{\partial \mathcal{L}_C^{k\ell}}{\partial \boldsymbol{\Sigma}_{k\ell}} = &-\tilde{z}_{.k}\tilde{w}_{.\ell} \log(\Sigma_{k\ell}^{-1})^\top \\ &+\frac{1}{2}\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}(\boldsymbol{\Sigma}_{k\ell}^{-1})^\top(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell}) \\ &\times(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell})^\top(\boldsymbol{\Sigma}_{k\ell}^{-1})^\top.\end{aligned}$$

The two partial derivatives set to 0 lead to

$$\hat{\boldsymbol{\mu}}_{k\ell} = \frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}\mathbf{x}_{ij}}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}},$$

and

$$\hat{\boldsymbol{\Sigma}}_{k\ell} = \frac{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell})(\mathbf{x}_{ij}-\boldsymbol{\mu}_{k\ell})^\top}{\sum_{i,j} \tilde{z}_{ik}\tilde{w}_{j\ell}}.$$

References

1. Ailem, M., Role, F., Nadif, M.: Model-based co-clustering for the effective handling of sparse data. Pattern Recognit. **72**, 108–122 (2017)
2. Ailem, M., Role, F., Nadif, M.: Sparse poisson latent block model for document clustering. IEEE Trans. Knowl. Data Eng. **29**(7), 1563–1576 (2017)
3. Banerjee, A., Krumpelman, C., Ghosh, J., Basu, S., Mooney, R.J.: Model-based overlapping clustering. In: Proceedings of the Eleventh ACM SIGKDD, pp. 532–537 (2005)
4. Bouchareb, A., Boullé, M., Clérot, F., Rossi, F.: Co-clustering based exploratory analysis of mixed-type data tables. In: Advances in Knowledge Discovery and Management, pp. 23–41. Springer (2019)
5. Bourgeois, F., Lassalle, J.C.: An extension of the Munkres algorithm for the assignment problem to rectangular matrices. Commun. ACM **14**(12), 802–804 (1971)
6. Boutalbi, R., Labiod, L., Nadif, M.: Co-clustering from tensor data. In: Yang, Q., Zhou, Z.H., Gong, Z., Zhang, M.L., Huang, S.J. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 370–383 (2019)
7. Briand, A.S., Côme, E., El Mahrsi, M.K., Oukhellou, L.: A mixture model clustering approach for temporal passenger pattern characterization in public transport. Int. J. Data Sci. Anal. **1**(1), 37–50 (2016)
8. Celeux, G., Govaert, G.: A classification EM algorithm for clustering and two stochastic versions. Comput. Stat. Data Anal. **14**(3), 315–332 (1992)
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. B **39**, 1–38 (1977)
10. Deodhar, M., Ghosh, J.: SCOAL: a framework for simultaneous co-clustering and learning from complex data. ACM Trans. Knowl. Discov. Data **4**, 1–31 (2010)
11. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: Proceedings of the Ninth ACM SIGKDD, pp. 89–98 (2003)
12. Feizi, S., Javadi, H., Tse, D.: Tensor biclustering. In: Advances in Neural Information Processing Systems, vol. 30, pp. 1311–1320. Curran Associates, Inc. (2017)

13. Fraley, C., Raftery, A.E.: How many clusters? Which clustering method? Answers via model-based cluster analysis. Comput. J. **41**(8), 578–588 (1998)
14. Govaert, G., Nadif, M.: Comparison of the mixture and the classification maximum likelihood in cluster analysis with binary data. Comput. Stat. Data Anal. **23**(1), 65–81 (1996)
15. Govaert, G., Nadif, M.: Clustering with block mixture models. Pattern Recognit. **36**, 463–473 (2003)
16. Govaert, G., Nadif, M.: An EM algorithm for the block mixture model. IEEE Trans. Pattern Anal. Mach. Intell. **27**(4), 643–647 (2005)
17. Govaert, G., Nadif, M.: Fuzzy clustering to estimate the parameters of block mixture models. Soft. Comput. **10**(5), 415–422 (2006)
18. Govaert, G., Nadif, M.: Block clustering with bernoulli mixture models: comparison of different approaches. Comput. Stat. Data Anal. **52**(6), 3233–3245 (2008)
19. Govaert, G., Nadif, M.: Co-clustering. Wiley-IEEE Press, Hoboken (2013)
20. Govaert, G., Nadif, M.: Mutual information, phi-squared and model-based co-clustering for contingency tables. Adv. Data Anal. Classif. **12**(3), 455–488 (2018)
21. Haralick, R., Shanmugam, K., Dinstein, I.: Textural features for image classification. IEEE Trans. Syst. Man Cybern. **3**(6), 610–621 (1973)
22. Kossaifi, J., Panagakis, Y., Anandkumar, A., Pantic, M.: Tensorly: tensor learning in python (2018). CoRR arXiv:1610.09555
23. Kumar, R.M., Sreekumar, K.: A survey on image feature descriptors. Int. J. Comput. Sci. Inf. Technol. (IJCSIT) **5**(1), 7668–7673 (2014)
24. Kurban, H., Jenne, M., Dalkilic, M.M.: Using data to build a better em: Em* for big data. Int. J. Data Sci. Anal. **4**(2), 83–97 (2017)
25. Labiod, L., Nadif, M.: Co-clustering under nonnegative matrix trifactorization. In: International Conference on Neural Information Processing, pp. 709–717. Springer (2011)
26. Labiod, L., Nadif, M.: A unified framework for data visualization and coclustering. IEEE Trans. Neural Netw. Learn. Syst. **26**(9), 2194–2199 (2014)
27. Munkres, J.: Algorithms for the assignment and transportation problems. J. Soc. Ind. Appl. Math. **5**(1), 32–38 (1957)
28. Pagès, J.: Multiple Factor Analysis by Example Using R. Chapman and Hall, London (2014)
29. Role, F., Morbieu, S., Nadif, M.: Coclust: a python package for co-clustering. J. Stat. Softw. **88**, 1–29 (2019)
30. Salah, A., Nadif, M.: Model-based von Mises-Fisher co-clustering with a conscience. In: Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 246–254. SIAM (2017)
31. Salah, A., Nadif, M.: Directional co-clustering. In: Advances in Data Analysis and Classification, pp. 1–30 (2018)
32. Steinley, D.: Properties of the Hubert-Arabie adjusted rand index. Psychol. Methods **9**(3), 386 (2004)
33. Strehl, A., Ghosh, J.: Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. **3**, 583–617 (2002)
34. Vu, D., Aitkin, M.: Variational algorithms for biclustering models. In: Computational Statistics and Data Analysis, pp. 12–24 (2015)
35. Wu, T., Benson, A.R., Gleich, D.F.: General tensor spectral co-clustering for higher-order data. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 29, pp. 2559–2567. Curran Associates, Inc., Red Hook (2016)
36. Zhang, H., Yang, M., Yang, W., Lv, J.: Spatial-aware hyperspectral image classification via multifeature kernel dictionary learning. Int. J. Data Sci. Anal. **7**(2), 115–129 (2019)