Adv Data Anal Classif
https://doi.org/10.1007/s11634-018-0323-4

REGULAR ARTICLE

Directional co-clustering

Aghiles Salah[1] · **Mohamed Nadif**[2]

Received: 13 September 2017 / Revised: 16 April 2018 / Accepted: 20 April 2018

Abstract Co-clustering addresses the problem of simultaneous clustering of both dimensions of a data matrix. When dealing with high dimensional sparse data, co-clustering turns out to be more beneficial than one-sided clustering even if one is interested in clustering along one dimension only. Aside from being high dimensional and sparse, some datasets, such as document-term matrices, exhibit directional characteristics, and the L_2 normalization of such data, so that it lies on the surface of a unit hypersphere, is useful. Popular co-clustering assumptions such as Gaussian or Multinomial are inadequate for this type of data. In this paper, we extend the scope of co-clustering to directional data. We present Diagonal Block Mixture of Von Mises–Fisher distributions (dbmovMFs), a co-clustering model which is well suited for directional data lying on a unit hypersphere. By setting the estimate of the model parameters under the maximum likelihood (ML) and classification ML approaches, we develop a class of EM algorithms for estimating dbmovMFs from data. Extensive experiments, on several real-world datasets, confirm the advantage of our approach and demonstrate the effectiveness of our algorithms.

Keywords Co-clustering · Directional data · von Mises-Fisher distribution · EM algorithm · Document clustering

Availablity: All presented algorithms in the paper are available at https://github.com/dbmovMFs/DirecCoclus.

✉ Aghiles Salah
asalah@smu.edu.sg

Mohamed Nadif
mohamed.nadif@parisdescartes.fr

1 SIS, Singapore Management University, Singapore 178902, Singapore

2 LIPADE, Paris Descartes University, 75006 Paris, France

Published online: 30 April 2018

Mathematics Subject Classification Main 62H30; Secondary 62H11

1 Introduction

Clustering is a powerful unsupervised learning technique that has been widely used to group together "similar" objects. Due to its practical importance, clustering has attracted a lot of attention in various scientific areas such as machine learning, data mining and information retrieval. This led to the development of a large variety of clustering approaches. However, when dealing with high dimensional sparse data (typically more than 1000 dimensions and 90% of zeros), most of the popular clustering methods suffer from poor performance, in terms of both the scalability and quality of clustering.

In this context, different approaches exist and the mixture model is undoubtedly a very useful contribution to clustering; it offers considerable flexibility. Dealing with directional data distributed on a unit hypersphere, the mixture of von Mises–Fisher (vMF) distributions may turn out to be a wise choice (Banerjee et al. 2005; Salah and Nadif 2017). In fact, this model noted *movMFs* is one of the most appropriate model for clustering high dimensional sparse data, such as document-term matrices arising in text mining. In this domain, it has been empirically demonstrated that vMF-based clustering methods perform better than several existing approaches; see, e.g., (Zhong and Ghosh 2005; Gopal and Yang 2014). In particular, vMF-based approaches, focus on the directions of objects (documents) which turn out to be relevant when clustering text documents. From a statistical point of view this means that document-term matrices possess directional characteristics (Mardia and Jupp 2000). Existing vMF-based clustering models, however, focus only on clustering along one dimension, i.e, either row or column clustering. Hence, they do not exploit the inherent duality between rows and columns of data matrices. In the clustering context, it turns out that, the exploitation of this duality presents a real advantage to improve the quality of clustering and alleviate the aforementioned difficulties related to high dimensionality and sparsity. This, can be achieved by using a co-clustering.

Co-clustering (Govaert and Nadif 2013) is an important extension of traditional one-sided clustering that addresses the problem of simultaneous clustering of both dimensions of data matrices. Since the works of Hartigan (1975), Bock (1979, 1994), Govaert (1995) or Vichi (2001), co-clustering, under various names, has been successfully used in a wide range of application domains where the co-clusters can take different forms (Van Mechelen et al. 2004). For instance, in bioinformatics co-clustering, referred to as biclustering (Madeira and Oliveira 2004; Hanczar and Nadif 2010), is used to cluster genes and experimental conditions simultaneously, in collaborative filtering (Deodhar and Ghosh 2010) to group users and items simultaneously, and in text mining (Ailem et al. 2016; Govaert and Nadif 2016) to group terms and documents simultaneously.

Co-clustering exhibits several practical advantages making it possible to meet the growing needs in several current areas of interest:

– By intertwining row clustering and column clustering at each stage, co-clustering performs an implicitly adaptive dimensionality reduction, which is imperative to deal with high dimensional sparse data. This makes it possible (i) to develop

efficient algorithms with a dramatically smaller number of parameters (ii) to reduce the original data matrix into a much simpler and condensed data matrix with the same structure.
- Co-clustering exploits the inherent duality between rows and columns of data matrices making it possible to enhance the clustering along both dimensions, by using the information contained in column clusters during row assignments and vice versa.
- Far from adding complexity, co-clustering is more informative than one-sided clustering, and produces meaningful clusters. In the case of document-term matrices, for example, co-clustering annotates sets of documents automatically by clusters of terms.

Several approaches have been proposed in order to address the problem of co-clustering and to date, there is no co-clustering approach that works better than the others in all situations. In this paper, we retain the generative mixture model-based approach (Govaert and Nadif 2013) for its flexibility, its ability to model different types of data and uncover various specific cluster structures. In particular, we concentrate on the practical problem of co-clustering document-term matrices arising in text mining. In this domain, it is well known that normalizing documents in order to lie on the surface of a unit hypersphere removes the bias induced by their lengths. Thus, and as pointed above, modelling such data as directional is the most appropriate choice. Despite the importance of such modelling, existing co-clustering approaches are based on popular assumptions such as Gaussian or Multinomial, which are inadequate to model L_2 normalized data. Thus, it seems natural to question whether it is possible to get the best of both directional modelling and co-clustering within the same model. In this paper, we provide an answer for the above question. We present a general co-clustering framework tailored for directional data. Our key contributions are:

- We present a novel model for co-clustering high dimensional sparse matrices. This model is based on the vMF distribution and successfully integrates a directional measure—cosine similarity—into a co-clustering framework.
- We provide theoretical connections between our model and existing ones, namely the vMF mixture model (Banerjee et al. 2005), the Gaussian (Banfield and Raftery 1993) and the block Gaussian mixture models (Nadif and Govaert 2010).
- Setting the estimate of the model parameters under the maximum likelihood (ML) approach, we formulate various co-clustering algorithms, including soft, hard, stochastic and two simulated annealing variants.
- To demonstrate the benefits of our approach for the analysis of high dimensional sparse data, we conducted extensive experiments, on several challenging real-world datasets.
- The dimensionality reduction property of our model alleviates the problem of high concentration parameters κ, which induce over and under flows; a well known difficulty in vMF based models. We validate this finding (i) theoretically by a theorem guaranteeing that our model leads to concentration parameters that are bounded from above by that of *movMFs*, (ii) empirically by showing that our algorithms yield substantially lower concentration parameters than *movMFs*-based algorithms, on real-world datasets.

2 Related work

Most of the earlier works using vMF distributions focused on low dimensional data, i.e, using 2- or 3-dimensional vMF distributions (McLachlan and Peel 2004), due to difficulties related to the estimation of the concentration parameter κ, that involves the inversion of ratios of Bessel functions. In the context of clustering and high dimensionality, Banerjee et al. (2005) proposed algorithms derived from a mixture of vMF distributions *movMFs*. They used an EM-based solution to estimate the parameters of their model and proposed an accurate approximation to estimate the concentration parameter κ for a high dimensional vMF distribution. Since this contribution, different vMF based models for clustering high dimensional sparse data have been proposed. For instance, Reisinger et al. (2010) proposed a spherical topic model based on a mixture of vMF distributions. More recently, for text data clustering, Gopal and Yang (2014) proposed a full Bayesian formulation of *movMFs* and developed two novel variants of *movMFs*, namely hierarchical and temporal. All these works, however, focused on one-sided clustering only.

Unlike previous vMF models, the model we propose acts simultaneously on both dimensions of data matrices, and thereby exhibits the advantages of co-clustering. Specifically, our model seeks a diagonal structure by co-clustering, meaning that rows and columns have the same number of clusters, and after a proper reorganization of rows and columns we obtain a block diagonal structure, as illustrated in Fig. 1.

In text mining, the domain which we focus on, the diagonal structure arises naturally in the document-term matrices, as it has been demonstrated by several earlier works. For example, Dhillon and Modha (2001) proposed the spherical k-means algorithm, which arises form a vMF mixture model, and they empirically demonstrated that documents are grouped together because they use similar terms yielding a block diagonal structure. In Dhillon et al. (2003) the information theoretic co-clustering algorithm was proposed, that seeks a more general block structure, i.e, not necessarily diagonal. However, during the experiments, the authors observed that some term clusters are highly indicative of individual document clusters inducing a block diagonal substructure. Among popular diagonal co-clustering, we can mention spectral approaches (Dhillon 2001; Labiod and Nadif 2011; Ailem et al. 2017b) and the block diagonal model for co-clustering binary data (Laclau and Nadif 2017) or count data; see, e.g., (Ailem et al. 2017a).

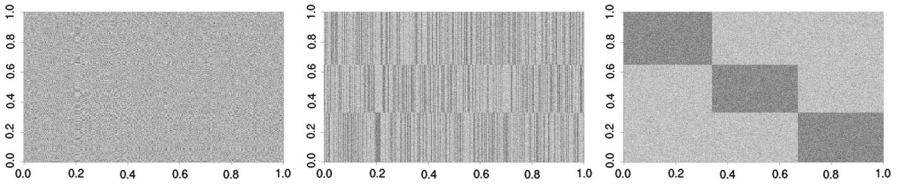

Fig. 1 dbmovMFs-based co-clustering: (left) original data, (middle) data reorganized according to $\mathbf{z}$, (right) data reorganized data according to ($\mathbf{z}$, $\mathbf{w}$)

The diagonal assumption may seem restrictive, however, when dealing with high dimensional sparse data, such as document-term matrices, this assumption exhibits several advantages as opposed to a non-binding model:

- It makes it possible to develop more parsimonious models, which lends itself more efficient algorithms since we focus on the most important co-clusters (diagonal ones), only.
- Unlike common co-clustering algorithms that treats similarly useful (relevant) and noisy (irrelevant) co-clusters, a diagonal co-clustering algorithm concentrates on the most relevant co-clusters which best allow us to identify document/word clusters.
- Due to sparsity, a general co-clustering algorithm may result in a poor locally optimal solution. More precisely, a general co-clustering algorithm seeks "homogeneous" co-clusters and as all co-clusters are treated equally, the quality of co-clustering may be biased by co-clusters containing a majority of zero entries, that are homogeneous but not useful (or irrelevant). This difficulty increases with the number of co-clusters.
- In the context of document-term matrices, diagonal co-clustering has the advantage of producing directly interpretable document clusters. Let us assume a co-clustering of a document-term matrix into 20 document- and 20 term- clusters. A naive partitioning by co-clustering will produce 400 co-clusters, and it is left to the user to identify the most useful co-clusters, so as to determine which document clusters should go with which term clusters, while a diagonal co-clustering will produce only 20 co-clusters that capture the most useful information.

Notation

- Matrices are denoted with boldface uppercase letters, vectors with boldface lowercase letters and sets by script style uppercase letters. The L_2 norm is denoted by $\|.\|$. The $(d-1)$ dimensional unit sphere embedded in $\mathbb{R}^d$ is denoted by $\mathbb{S}^{d-1}$.
- Data is represented by a matrix $\mathbf{X} = (x_{ij})$ of size $n \times d$, $x_{ij} \in \mathbb{R}$, the ith row of this matrix is represented by a vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})^\top$, where $\top$ denotes the transpose.
- The partition of the set of rows $\mathcal{I}$ into g clusters can be represented by a classification matrix $\mathbf{Z}$ of elements z_{ih} in $\{0, 1\}^g$ satisfying $\sum_{h=1}^{g} z_{ih} = 1$. The notation $\mathbf{z} = (z_1, \ldots, z_n)^\top$, where $z_i \in \{1, \ldots, g\}$ represents the cluster label of i, will be also used.
- Similarly the notations $\mathbf{W} = (w_{jh})$ of size $d \times g$, $w_{jh} \in \{0, 1\}^g$ satisfying $\sum_{h=1}^{g} w_{ih} = 1$, and $\mathbf{w} = (w_1, \ldots, w_d)$, where $w_j \in \{1, \ldots, g\}$ represents the cluster label of j, will be used to represent the partition of the set of columns $\mathcal{J}$.
- In the same way, the fuzzy classification matrix of $\mathcal{I}$ will be denoted by $\tilde{\mathbf{Z}} = (\tilde{z}_{ih})$ where $\tilde{z}_{ih} \in [0, 1]$, satisfying $\sum_{h=1}^{g} \tilde{z}_{ih} = 1$, for all i in $\mathcal{I}$.

3 Preliminaries

In this section, we review the von Mises–Fisher (vMF) distribution, which is well known in directional statistics (Mardia and Jupp 2000).

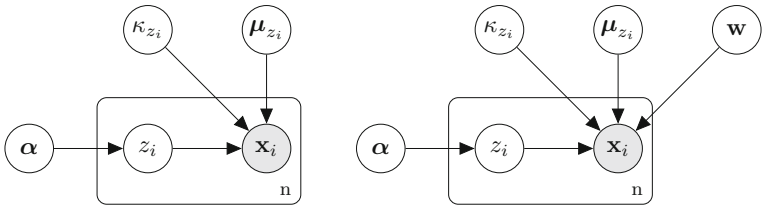

Fig. 2 Left *movMFs* (Banerjee et al. 2005), right *dbmovMFs* (this paper)

A d dimensional vMF (d-vMF) distribution, i.e, $d \geq 2$ is a continuous probability distribution on a unit hypersphere $\mathbb{S}^{d-1}$. Thus, if a d dimensional data point $\mathbf{x}_i$ on $\mathbb{S}^{d-1}$, i.e, $\mathbf{x}_i \in \mathbb{R}^d$ and $\|\mathbf{x}_i\| = 1$ follows a d-vMF distribution, its probability density function is given by:

$$f(\mathbf{x}_i|\boldsymbol{\mu}, \kappa) = c_d(\kappa) \exp^{\kappa \boldsymbol{\mu}^\top \mathbf{x}_i}, \tag{1}$$

where $\boldsymbol{\mu}$ is the mean direction (centroid) parameter and κ denotes the concentration parameter, such that $\|\boldsymbol{\mu}\| = 1$ and $\kappa \geq 0$. The normalization term $c_d(\kappa)$ is equal to $c_d(\kappa) = \frac{\kappa^{\frac{d}{2}-1}}{(2\pi)^{\frac{d}{2}} I_{\frac{d}{2}-1}(\kappa)}$ where $I_r(\kappa)$ represents the modified Bessel function of the first kind and order r. In the vMF distribution the parameter κ controls the concentration of data points $\mathbf{x}_i \in \mathbb{S}^{d-1}$ following (1), around the mean direction $\boldsymbol{\mu}$. Thus, $f(\mathbf{x}_i|\boldsymbol{\mu}, \kappa)$ reduces to the uniform density on $\mathbb{S}^{d-1}$ for $\kappa = 0$, and it is uni-modal if $\kappa > 0$. In particular, when $\kappa \to \infty$, $f(\mathbf{x}_i|\boldsymbol{\mu}, \kappa)$ tends to a point density.

4 A mixture of von Mises–Fisher distributions for co-clustering

Recall that the original mixture of vMF distribution (*movMFs*) (Banerjee et al. 2005) focuses solely on clustering along one dimension of a data matrix. Here, we develop a vMF mixture model for co-clustering. Following the results in Dhillon and Modha (2001), which state that the unit centroids produced by the spherical `kmeans` algorithm (a vMF-based clustering algorithm) are localized in the feature space and tend towards orthonormality, we propose to capture and exploit this structure during the clustering process. More precisely, we assume that the centroids are orthonormal and homogeneous from the beginning. To this end, we introduce a new parameter $\mathbf{w}$ (see Fig. 2) that simultaneously guarantees the above assumption and plays the role of a column partition. Formally, for each centroid $\boldsymbol{\mu}_h^{\mathbf{w}}$: $\mu_{hj} = 0$ if $w_{jh} = 0$ (column j does not belongs to cluster h), and $\mu_{hj} = \mu_{hh}$ for all j such as $w_{jh} = 1$. For instance, assuming a mixture of 3 vMF distributions, i.e, $g = 3$ and $h, k = 1, 2, 3$, each centroid $\boldsymbol{\mu}_h^{\mathbf{w}} \in \mathbb{S}^{d-1}$ takes this form:

$$\boldsymbol{\mu}_h^{\mathbf{w}} = (\mu_{h1}, \ldots, \mu_{h1}, \mu_{h2}, \ldots, \mu_{h2}, \mu_{h3}, \ldots, \mu_{h3})^\top \tag{2}$$

where μ_{hk} is repeated $w_{.h}$ times; $w_{.h}$ denotes the cardinality of the hth column cluster. In addition, we have $\mu_{hk} = 0 \ \forall k \neq h$, leading implicitly to the orthonormality of centroids, i.e., $\boldsymbol{\mu}_h^\top \boldsymbol{\mu}_{h'} = 0$, for all $h \neq h'$, and $\boldsymbol{\mu}_h^\top \boldsymbol{\mu}_h = 1$.

From a co-clustering point of view, this is equivalent to assume that rows and columns have the same number of clusters and that each column cluster is associated (or describes) a single row cluster involving a block diagonal structure (see Fig. 1). Our model called *dbmovMFs* (diagonal block mixture of vMF distributions) seeks to partition simultaneously the set of rows $\mathcal{I}$ and columns $\mathcal{J}$. Its density function is given by:

$$f(\mathbf{x}_i|\Theta) = \sum_h \alpha_h f_h(\mathbf{x}_i|\boldsymbol{\mu}_h^{\mathbf{w}}, \kappa_h^{\mathbf{w}}, \mathbf{w}), \tag{3}$$

where Θ is now formed by $\boldsymbol{\mu}_1^{\mathbf{w}},\ldots, \boldsymbol{\mu}_g^{\mathbf{w}}, \alpha_1, \ldots, \alpha_g, \kappa_1^{\mathbf{w}}, \ldots, \kappa_g^{\mathbf{w}}$ and the column partition $\mathbf{w}$, i.e, $w_j = h$ if the jth column belongs to the hth column cluster, that is associated with the hth row cluster. Notice that, the centroid and the concentration parameters $\boldsymbol{\mu}_h^{\mathbf{w}}, \kappa_h^{\mathbf{w}}$, respectively, depend on the column partition $\mathbf{w}$.

Let $\mathcal{X}$ denote a set of n randomly sampled data points $\mathbf{x}_i$ on $\mathbb{S}^{d-1}$ according to (3). Using the row and column classification matrices $\mathbf{Z}$ and $\mathbf{W}$, respectively, the classification likelihood of $\mathcal{X}$ takes the following form:

$$\mathcal{L}(\mathbf{W}, \boldsymbol{\mu}, \boldsymbol{\alpha}, \kappa|\mathcal{X}, \mathbf{Z}) = \prod_i \prod_h \left(\alpha_h c_d(\kappa_h) \times \prod_j (\exp^{\kappa_h \mu_{hh} x_{ij}})^{w_{jh}} \right)^{z_{ih}}.$$

The corresponding classification log-likelihood of $\mathcal{X}$ is given by:

$$\begin{aligned} L_c(\Theta|\mathcal{X}, \mathbf{Z}) &= \sum_{i,h} z_{ih} \log \alpha_h + \sum_{i,h} z_{ih} \log(c_d(\kappa_h)) + \sum_{i,h,j} z_{ih} w_{jh} \kappa_h \mu_{hh} x_{ij} \\ &= \sum_h z_{.h} \log \alpha_h + \sum_h z_{.h} \log(c_d(\kappa_h)) + \sum_{i,h} z_{ih} \kappa_h \mu_{hh} u_{ih} \end{aligned} \tag{4}$$

where $u_{ih} = \sum_j w_{jh} x_{ij}$. This leads to

$$L_c(\Theta|\mathcal{X}, \mathbf{Z}) = \sum_h z_{.h} \log \alpha_h + \sum_h z_{.h} \log(c_d(\kappa_h)) + \sum_{i,h} z_{ih} y_{ih} \tag{5}$$

where $y_{ih} = \kappa_h \mu_{hh} u_{ih}$, and in the same manner, we can give another expression of $L_c(\Theta|\mathcal{X}, \mathbf{Z})$ in terms of column assignments as follows

$$\sum_h z_{.h} \log \alpha_h + \sum_h z_{.h} \log(c_d(\kappa_h)) + \sum_{j,h} w_{jh} t_{jh} \tag{6}$$

where $t_{jh} = \kappa_h \mu_{hh} v_{hj}$, with $v_{hj} = \sum_i z_{ih} x_{ij}$.

4.1 Connection to existing models

Assuming that the column partition $\mathbf{w}$ is fixed, the *dbmovMFs* model can be viewed as a classical *movMFs* (Banerjee et al. 2005) where the mean directions vectors $\boldsymbol{\mu}_h$, $h = 1, \ldots, g$, take the form (2) described above.

Moreover, we can establish connections with the Gaussian mixture model (Banfield and Raftery 1993) and the block Gaussian mixture model (Govaert and Nadif 2013). More precisely, using the equivalence between the vMF and the Gaussian distribution (Mardia and Jupp 2000) and assuming that $\mathbf{w}$ is fixed, it can be shown that *dbmovMFs* is equivalent to a mixture of Gaussian distributions of spherical form, i.e, the variance of the hth cluster is given by $\sigma_h^2 = \|\mathbf{m}_h^{\mathbf{w}}\|/\kappa_h$, $\mathbf{m}_h^{\mathbf{w}}$ is the centroid of the corresponding Gaussian component and $\boldsymbol{\mu}_h^{\mathbf{w}} = \mathbf{m}_h^{\mathbf{w}}/\|\mathbf{m}_h^{\mathbf{w}}\|$. The latter model, is also equivalent to a diagonal version of the block Gaussian mixture model (Nadif and Govaert 2010), i.e, each Gaussian component is parameterized by the variance σ_h^2 and mean vector $\mathbf{m}_h^{\mathbf{w}}$, described above.

4.2 Maximum likelihood estimates and the $\texttt{EM}_b$ algorithm

To obtain the maximum likelihood estimates for the parameters Θ, we use the *Generalized* EM algorithm (Dempster et al. 1977; McLachlan and Krishnan 2007). The E-step computes the posterior probabilities defined by $\tilde{z}_{ih} \propto \alpha_h^{(t)} f_h(\mathbf{x}_i|\boldsymbol{\mu}_h^{(t)}, \kappa_h^{(t)})$. The M-step consists in estimating all parameters maximizing or increasing the expectation of the classification log-likelihood (4), subject to the constraints $\sum_h \alpha_h = 1$, $\|\boldsymbol{\mu}_h^{\mathbf{w}}\|^2 = \sum_j w_{jh}\mu_{hh}^2 = 1$ and $\kappa_h > 0$. We obtain the following update formulas (see "Appendix A.1").

$$\hat{w}_{jh} \leftarrow \begin{cases} 1, & \text{if } h = \arg\max_{h'} \tilde{t}_{jh'} \\ 0, & \text{otherwise} \end{cases}$$

$$\text{where } \tilde{t}_{jh} = \kappa_h \mu_{hh} \tilde{v}_{hj} \text{ with } \tilde{v}_{hj} = \sum_i \tilde{z}_{ih} x_{ij}, \tag{7a}$$

$$\hat{\alpha}_h = \frac{\sum_i \tilde{z}_{ih}}{n}, \tag{7b}$$

$$\hat{\mu}_{hh} = \frac{r_h^{\mathbf{w}}}{\|\mathbf{r}_h^{\mathbf{w}}\|} = \pm\frac{1}{\sqrt{\hat{w}_{.h}}} \quad \text{with} \quad r_h^{\mathbf{w}} = \sum_{i,j} \tilde{z}_{ih}\hat{w}_{jh}x_{ij} \text{ and } \hat{w}_{.h} = \sum_j \hat{w}_{jh}, \tag{7c}$$

$$\hat{\kappa}_h \approx \frac{\bar{r}_h^{\mathbf{w}} d - \left(\bar{r}_h^{\mathbf{w}}\right)^3}{1 - \left(\bar{r}_h^{\mathbf{w}}\right)^2} \quad \text{where} \quad \bar{r}_h^{\mathbf{w}} = \frac{I_{d/2}(\hat{\kappa}_h)}{I_{d/2-1}(\hat{\kappa}_h)} = \frac{\|\mathbf{r}_h^{\mathbf{w}}\|}{\tilde{z}_{.h}\hat{w}_{.h}}, \tag{7d}$$

with $\mathbf{r}_h^{\mathbf{w}}$ a d dimensional vector such that $r_{hj}^{\mathbf{w}} = r_h^{\mathbf{w}}$ if $w_{jh} = 1$ and $r_{hj}^{\mathbf{w}} = 0$, otherwise. Alternating the above E and M steps leads to our soft-dbmovMF algorithm described in Algorithm 1. Observe that, unlike the classical *movMFs* where it is easy to verify that $\bar{r}_h \leq 1$ given the definition of $\mathbf{r}$ (Banerjee et al. 2005), it is not straightforward to verify that $\bar{r}_h^{\mathbf{w}} \leq 1$, without careful analysis. Such a result is imperative, to guarantee that the concentration parameters are positive, i.e, $\kappa_h > 0, \ \forall h$, especially when using

Algorithm 1 soft-dbmovMF ($\texttt{EM}_{\mathbf{b}}$).

Input: $\mathcal{X}$ ($\mathbf{x}_i \in \mathbb{S}^{d-1}$), g the number of co-clusters.
Output: $\tilde{\mathbf{Z}}$ and $\mathbf{W}$,
Steps:
Initialization: $\Theta \leftarrow \Theta^{(0)}$;
repeat
1. Expectation step: $\tilde{z}_{ih} \propto \alpha_h f_h(\mathbf{x}_i|\boldsymbol{\mu}_h, \kappa_h)$, for all i, h
2. Maximization step:
for $j = 1$ **to** d **do**
$\tilde{v}_{hj} \leftarrow \sum_i \tilde{z}_{ih} x_{ij}$; $\tilde{t}_{jh} \leftarrow \kappa_h \mu_{hh} \tilde{v}_{hj}$, for all h
$\hat{w}_{jh} \leftarrow 1$, if $h = \arg\max_{h'} \tilde{t}_{jh'}$, and $\hat{w}_{jh} \leftarrow 0$, otherwise.
end for
for $h = 1$ **to** g **do**
$\hat{\alpha}_h \leftarrow \frac{\sum_i \tilde{z}_{ih}}{n}$; $\hat{\mu}_{hh} \leftarrow \pm \frac{1}{\sqrt{\hat{w}_{.h}}}$; $r_h^{\mathbf{w}} \leftarrow \sum_{i,j} \tilde{z}_{ih} \hat{w}_{jh} x_{ij}$
$\bar{r}_h^{\mathbf{w}} \leftarrow \frac{\|\mathbf{r}_h^{\mathbf{w}}\|}{\tilde{z}_{.h} \hat{w}_{.h}}$; $\hat{\kappa}_h \leftarrow \frac{\bar{r}_h^{\mathbf{w}} d - (\bar{r}_h^{\mathbf{w}})^3}{1-(\bar{r}_h^{\mathbf{w}})^2}$
end for
until convergence

the approximation of Eq. (7d). Proposition 1 provides theoretical guarantee about the fact that $0 \leq \bar{r}_h^{\mathbf{w}} \leq 1$. The proof is available in "Appendix (A.2)". By replacing $\mathbf{r}^d$, d and p_i in Proposition 1 with $\mathbf{r}_h^{\mathbf{w}}$, $\hat{w}_{.h}$ and $\tilde{z}_{ih}$ respectively, it is easy to verify $0 \leq \bar{r}_h^{\mathbf{w}} \leq 1$.

Proposition 1 *Let* $\mathbf{r}$ *be a non-zero vector in* $\mathbb{R}^d$ *(i.e.,* $\mathbf{r} = (r_1, \ldots, r_d)^\top$*, such that* $d \geq 1$*) which results from a weighted sum of* n d*-dimensional unit vector, i.e,* $\mathbf{r} = \sum_i p_i \mathbf{x}_i$, $\mathbf{x}_i \in \mathbb{R}^d$ *and* $\|\mathbf{x}_i\| = 1$, $\forall i \in \{1, \ldots, n\}$, $n \geq 2$*, and weights* $p_i \geq 0$, $\forall i$*. Let* $\mathbf{r}^d$ *be a vector in* $\mathbb{R}^d$*, such as all its components are equal to the sum of elements of* $\mathbf{r}$ *(i.e,* $\mathbf{r}_1^d = \cdots = \mathbf{r}_d^d = \sum_{j=1}^d r_j$*). Then* $0 \leq \|\mathbf{r}^d\| \leq d \times \sum_i p_i$ *with equality only if all unit vectors* $\mathbf{x}_i$ *are equal/collinear.*

More interestingly, the concentration parameters depend on the column partition $\mathbf{w}$, hence, the dimensionality reduction of *dbmovMFs* alleviates the problem of high concentration parameters. The following theorem guarantees that *dbmovMFs* leads to a concentration parameter that is bounded from above by that of *movMFs*, for each cluster. The proof is provided in "Appendix (A.2)". In practice and when dealing with high dimensional datasets, we empirically observe that *dbmovMFs*-based co-clustering algorithms yield substantially lower concentration parameters than *movMFs*-based algorithms, see Sect. 7.2.6.

Theorem 1 *Let* $\mathbf{X}$ *be a* $n \times d$ *matrix, its ith row (object)* $\mathbf{x}_i$ *is a* d*-dimensional unit vector in* $\mathbb{S}^{d-1}$ *(i.e,* $\mathbf{x}_i \in \mathbb{R}^d$ *and* $\|\mathbf{x}_i\| = 1$, $\forall i \in \{1, \ldots, n\}$, $n \geq 2$, $d \geq 3$*). Let* $\mathbf{z} = (z_1, \ldots, z_n)$ *denote a partition of the set of objects of* $\mathbf{X}$ *into* g *disjoint clusters. Then, whatever the partition* $\mathbf{w}$ *of attributes of* $\mathbf{X}$ *into* g *disjoints clusters, the concentration parameter of each dbvMF component* $\hat{\kappa}_h^{\mathbf{w}}$ *estimated via approximation* (7d) *is always bounded from above by its vMF counterpart* $\hat{\kappa}_h$*. That is,* $\hat{\kappa}_h^{\mathbf{w}} \approx \frac{\bar{r}_h^{\mathbf{w}} d - (\bar{r}_h^{\mathbf{w}})^3}{1-(\bar{r}_h^{\mathbf{w}})^2} \leq \hat{\kappa}_h \approx \frac{\bar{r}_h d - (\bar{r}_h)^3}{1-(\bar{r}_h)^2}$ *with equality only if* $\bar{r}_h^{\mathbf{w}} = \bar{r}_h$.

4.3 Classification maximum likelihood estimates and the $\texttt{CEM}_{\boldsymbol{b}}$ algorithm

Setting *dbmovMFs* under the CML approach, which consists in maximizing the classification likelihood instead of its expectation (Celeux and Govaert 1992), we derive a hard version of *dbmovMFs* called $\texttt{CEM}_{\mathbf{b}}$. This is done by incorporating a classification step (C-step) between the E and M steps as follows: $z_{ih} = 1$ if $h = \arg\max_{h'} \tilde{z}_{ih'}$ and 0 otherwise. The C-step of $\texttt{CEM}_{\mathbf{b}}$, generates a completed sample $(\mathbf{x}_i, z_i)$ by allocating each object $\mathbf{x}_i$ to cluster z_i with the highest posterior probability $\tilde{z}_{ih}$, $\forall h$. Then, unlike in the $\text{EM}_{\mathbf{b}}$ algorithm, the M-step of $\texttt{CEM}_{\mathbf{b}}$ consists in maximizing the classification likelihood instead of its expectation, thereby the update of Θ is based on the completed sample $(\mathbf{x}_i, z_i)$. The corresponding M-step can be deduced from the M-Step of $\texttt{EM}_{\mathbf{b}}$ by replacing $\tilde{z}_{ih}$ by z_{ih} and thereby $\tilde{t}_{jh}$ by t_{jh}.

In this way, $\texttt{CEM}_{\mathbf{b}}$ simultaneously estimates the parameters and the partition $\mathbf{z}$. Note that, $\texttt{CEM}_{\mathbf{b}}$ is not meant to converge to the ML estimate of Θ, thereby, it yields inconsistent estimates of the parameters, especially when the mixture components overlap and exhibit dramatically different proportions. However, $\texttt{CEM}_{\mathbf{b}}$ has some useful properties listed below.

- $\texttt{CEM}_{\mathbf{b}}$ is considerably more faster and scalable than $\texttt{EM}_{\mathbf{b}}$, for instance, consider the update of the parameter $\hat{w}_{jh}$, under the ML approach (see, Eq. 7a) we iterate over all objects $\mathbf{x}_i$ to compute $\tilde{t}_{jh}$, while with $\texttt{CEM}_{\mathbf{b}}$ we only iterate over objects within the hth cluster to compute t_{jh}.
- $\texttt{CEM}_{\mathbf{b}}$ allows us to avoid numerical difficulties due to the computation of $\tilde{z}_{ih}$, especially in the case of vMF distribution where the terms $c_d(\kappa_h)$ involve Bessel functions, and the concentration parameters act as multipliers in the exponent. In fact, with $\texttt{CEM}_{\mathbf{b}}$ we do not need to compute the probabilities $\tilde{z}_{ih}$, the quantity $\log \alpha_h + \log f_h(\mathbf{x}_i|\Theta^{(t)})$ is sufficient to perform the C-step.
- It eases the derivation of a large number of standard clustering algorithms as special cases from a mixture framework, which allows to give them a probabilistic interpretation.

5 Stochastic variants

It is well known that EM is strongly dependent on its starting position. The stochastic variant of EM (SEM) (Celeux and Diebolt 1985) however makes it possible to alleviate this limitation. It consists in incorporating a stochastic step (S-step) between the E and M steps, which generates a completed sample $(\mathbf{x}_i, z_i)$ by drawing a cluster $z_i \in \{1, \ldots, g\}$ for each data point $\mathbf{x}_i$ according to the multinomial distribution $\mathcal{M}(\tilde{z}_{i1}, \ldots, \tilde{z}_{ig})$.

As in CEM the update of Θ is based on the completed sample $(\mathbf{x}_i, z_i)$ i.e, classification likelihood instead of its expectation. The SEM algorithm does not share the convergence properties of EM and CEM. In fact, SEM can allow an update estimate $\Theta^{(t+1)}$ even if $L(\Theta^{(t+1)}) < L(\Theta^{(t)})$. Thereby, SEM does not necessarily converge to the first encountered stationary point of the log-likelihood, which allows it to ignore saddle points and insignificant local optima.

5.1 Stochastic *dbmovMFs* and the $\texttt{SEM}_{\mathbf{b}}$ algorithm

Based on SEM we formulate a stochastic version of soft-dbmovMF called $\texttt{SEM}_{\mathbf{b}}$. In our case, we further propose a stochastic column assignment, by converting $\tilde{t}_{jh}$ to probabilities $\tilde{w}_{jh}$, i.e, $\tilde{w}_{jh} \propto \tilde{t}_{jh}$ and drawing a cluster w_j for each column j according to a Multinomial distribution $\mathcal{M}(\tilde{w}_{j1}, \ldots, \tilde{w}_{jg})$. This makes it possible to avoid a quick convergence to a bad local optimum of the likelihood function, due to hard column assignments. More intuitively, in the case of document-term matrices, during the first iterations document clusters are mixed, hence each term is involved in describing several document clusters. Therefore, imposing hard term assignments at the beginning, by using Eq. (7a), can lead to a poor local optimum of the likelihood function, especially when the clusters are poorly separated.

5.2 Simulated annealing *dbmovMFs*-based algorithms

One of the main drawback of the SEM algorithm is that it does not share the convergence properties of EM; SEM converges in distribution and does not converge point-wise. Furthermore, when the sample size of the observed data is small, SEM may converge to a stationary distribution with a high variance, which leads to poor estimation of the parameters. In order to overcome the aforementioned limitations, the *Simulated Annealing* version of EM (SAEM) has been proposed (Celeux and Diebolt 1992), that reaps the benefit of both EM and SEM simultaneously. To do so, let γ_t be a sequence of positive real numbers starting from 1 and decreasing to zero, at each iteration the parameters are updated as follows: $\Theta^{(t+1)}_{\text{SAEM}} = (1-\gamma_{t+1})\Theta^{(t+1)}_{\text{EM}} + \gamma_{(t+1)}\Theta^{(t+1)}_{\text{SEM}}$, where $\Theta^{(t+1)}_{\text{EM}}$ and $\Theta^{(t+1)}_{\text{SEM}}$ denote the parameters estimated using EM and SEM, respectively. Thus, SAEM goes from pure SEM at the beginning towards pure EM at the end. If we further impose the following constraint $\lim_{t\to\infty} \frac{\gamma_{(t)}}{\gamma_{(t+1)}} = 1$, we can ensure the asymptotic convergence of SAEM to a local maximum of the log-likelihood.

Based on SAEM, we derive the $\texttt{SAEM}_{\mathbf{b}}$ algorithm, which is a simulated annealing version of $\texttt{EM}_{\mathbf{b}}$. In our case, however, it is difficult to use the above update formula of SAEM due to parameter $\mathbf{w}$. In order to overcome this difficulty, we propose to use the following simplified form to update the parameters, which turns out to be effective and less costly.

$$\Theta^{(t+1)}_{\text{SAEM}_{\mathbf{b}}} \leftarrow \Theta^{(t+1)}_{\text{SEM}_{\mathbf{b}}}, \text{ if } \gamma_{(t+1)} \geq (1-\gamma_{(t+1)}), \text{ and } \Theta^{(t+1)}_{\text{SAEM}_{\mathbf{b}}} \leftarrow \Theta^{(t+1)}_{\text{EM}_{\mathbf{b}}}, \text{ otherwise.}$$

We propose to use the following exponentially decreasing form for $\gamma_{(t)} = 1 - \exp^{\frac{it^{(t)} - it^{max}}{\beta}}$ where $it^{(t)}, it^{max}$ denote respectively, the current iteration and the maximum number of iterations, β is a positive real parameter that controls the number of $\texttt{SEM}_{\mathbf{b}}$ and $\texttt{EM}_{\mathbf{b}}$ iterations to be performed. More precisely when $\beta \to \infty$, $\texttt{SAEM}_{\mathbf{b}}$ tends to pure $\texttt{EM}_{\mathbf{b}}$, similarly when $\beta \to 0$, $\texttt{SAEM}_{\mathbf{b}}$ tend to pure $\texttt{SEM}_{\mathbf{b}}$. In our experiments, we set $\beta = 20$ and $it^{max} = 100$, in such a way that most iterations at the beginning are done using $\texttt{SEM}_{\mathbf{b}}$, in order to reach a steady state and avoid poor local

optima, and the last few iterations are performed using $\texttt{EM}_{\mathbf{b}}$, to ensure the convergence of $\texttt{SAEM}_{\mathbf{b}}$ to a local optimum of the log-likelihood function.

This simplified version consists in initializing $\texttt{EM}_{\mathbf{b}}$ with $\texttt{SEM}_{\mathbf{b}}$. seeing the initialization of $\texttt{EM}_{\mathbf{b}}$ via $\texttt{SEM}_{\mathbf{b}}$ as a *Simulated Annealing* variant, justifies why $\texttt{EM}_{\mathbf{b}}$ is expected to provide better performances with such an initialization. In the same manner, we derived a hard simulated annealing variant, denoted in this paper as $\texttt{CAEM}_{\mathbf{b}}$. The difference with the $\texttt{SAEM}_{\mathbf{b}}$ algorithm is that the iterations at the end are performed using $\texttt{CEM}_{\mathbf{b}}$ instead of $\texttt{EM}_{\mathbf{b}}$.

6 Computational complexity in the worst case

Next we analyse the computational complexity of hard-dbmovMF and soft-dbmovMF.

Proposition 2 *Let* $\mathbf{X}$ *be a* $n \times d$ *matrix, let* nz *denote the number of non-zero entries in* $\mathbf{X}$*, "it" is the number of iterations and* g *is the number of co-clusters to be found. Then (i) the computational complexity of hard-dbmovMF is given in* $O(it \cdot nz)$ *(ii) the computational complexity of soft-dbmovMF described in Algorithm* 1 *is given in* $O(it \cdot g \cdot nz)$.

The proofs of (i) and (ii) are given in "Appendix (A.3)". Proposition 2, states that theoretically, the computational time of our algorithms is linear with respect to the number of non-zero entries in $\mathbf{X}$. Hence, the proposed algorithms are very efficient (in particular the hard variant) and therefore suitable for large sparse datasets.

7 Experimental results

In the sequel, we conduct extensive experiments to validate and illustrate the interest of the proposed model *dbmovMFs* and the derived co-clustering algorithms.

7.1 Competing methods

We benchmark our algorithms, i.e, $\texttt{EM}_{\mathbf{b}}$, $\texttt{CEM}_{\mathbf{b}}$, $\texttt{SEM}_{\mathbf{b}}$, $\texttt{SAEM}_{\mathbf{b}}$ and $\texttt{CAEM}_{\mathbf{b}}$ against several strong baselines denoted in this paper as follows

- EM denotes the soft-movMF algorithm proposed in Banerjee et al. (2005).
- CEM denotes the hard-movMF algorithm proposed in Banerjee et al. (2005).
- DAEM is the *Deterministic Annealing* version of soft-movMF proposed in Zhong and Ghosh (2005).
- Skmeans denotes the spherical kmeans algorithm (Dhillon and Modha 2001), which is also a simplified version of soft-movMF, where $\kappa_h = \kappa \rightarrow \infty$, $\alpha_h = \alpha$, for all h. It is also a restricted version of hard-movMF where $\kappa_h = \kappa$, $\alpha_h = \alpha$, for all h.

Table 1 Description of datasets

Datasets	Characteristics				
	#Documents	#Terms	#Clusters	Sparsity (%)	Balance[a]
CSTR	475	1000	4	96.60	0.399
WEBACE	2340	1000	20	91.83	0.169
CLASSIC4	7094	5896	4	99.41	0.323
NG20	19949	43586	20	99.82	0.991
SPORTS	8580	14870	7	99.14	0.036
TDT2	9394	36771	30	99.64	0.028

[a] The balance coefficient is the ratio of the minimum cluster size to the maximum cluster size

– `CoclusMod`[1] is a recent co-clustering algorithm proposed by Ailem et al. (2016). Through extensive experiments on text datasets, the authors showed that `CoclusMod` outperforms several other notable co-clustering methods.

Notice that, in the context of text document clustering, it has been empirically shown on numerous real-world datasets, that the aforementioned baselines perform better than several existing clustering and co-clustering algorithms, namely: `kmeans` using the euclidean distance, generative mixture models using Bernoulli, Gaussian, and Multinomial distributions, spectral co-clustering, and Latent Dirichlet Allocation (LDA). Therefore, we do not include these approaches in our comparisons. For further details see Zhong and Ghosh (2005), Gopal and Yang (2014), Ailem et al. (2017a).

7.2 Real-world datasets

We concentrate on the challenging task of document clustering using high dimensional and sparse document-term matrices. We retain six benchmark text datasets: CSTR used in Li (2005), CLASSIC4,[2] WEBACE, the 20-newsgroups data (NG20), SPORTS used in Zhong and Ghosh (2005), and TDT2.[3] All these datasets are carefully selected to represent various particular challenging situations in clustering: balanced clusters, unbalanced clusters, different number of clusters, different sizes, different degrees of cluster overlap. The statistics of these different datasets are summarized in Table 1.

7.2.1 Evaluation measures

Evaluating clustering results is not a trivial task, however, when the true category labels are known, a commonly used approach to validate clustering results consists in comparing the estimated partition with the true one. To this end, we retain two widely used measures to assess the quality of clustering, namely the Normalized

[1] https://pypi.python.org/pypi/coclust.

[2] http://www.dataminingresearch.com/.

[3] http://www.cad.zju.edu.cn/home/dengcai/.

Mutual Information (NMI) (Strehl and Ghosh 2003) and the Adjusted Rand Index (ARI) (Hubert and Arabie 1985). Intuitively, NMI quantifies how much the estimated clustering is informative about the true clustering, while the ARI measures the degree of agreement between an estimated clustering and a reference clustering. Higher NMI/ARI is better.

7.2.2 Experimental setting

We use the TF-IDF normalized data representation. For each dataset we set g to the real number of clusters, and in order to make fair comparisons, all non-stochastic algorithms are initialized using the same row partition resulting from `Skmeans`[4] started using a random initial point, unless stated otherwise. For our algorithms, we further initialize the concentration parameters to 10 and the centroids to random initial vectors, in order to be able to estimate the initial column partition. Concerning our stochastic variants, i.e, $\texttt{SEM}_{\mathbf{b}}$, $\texttt{CAEM}_{\mathbf{b}}$ and $\texttt{SAEM}_{\mathbf{b}}$ they are initialized using the same random row and column partitions.

7.2.3 Evaluation of document clusters

Tables 2 and 3, summarize the results of the different methods over all datasets. All results are averaged over thirty different starting points, obtained using the initialization strategy described above. Between brackets, we report the results corresponding to the trial with the highest criterion. Overall, we observe that *dbmovMFs*-based algorithms offer the best performances in almost all situations, except in terms of ARI on TDT2, where `CoclusMod` achieves the best performances. Importantly, $\texttt{EM}_{\mathbf{b}}$ and $\texttt{CEM}_{\mathbf{b}}$ always outperform their *movMFs* counterparts (`EM` and `CEM`), which provides strong support for the advantage of our co-clustering formulation. Surprisingly, we note that `Skmeans` performs substantially better than its generalized variants `EM` and `CEM` in most cases. We discuss the latter point in more details in Sect. 7.2.6.

7.2.4 Advantages of $\texttt{SAEM}_{\mathbf{b}}$ *and* $\texttt{CAEM}_{\mathbf{b}}$ *over* $\texttt{SEM}_{\mathbf{b}}$, $\texttt{EM}_{\mathbf{b}}$ *and* $\texttt{CEM}_{\mathbf{b}}$

We observe that our simulated annealing variants, $\texttt{SAEM}_{\mathbf{b}}$ and $\texttt{CAEM}_{\mathbf{b}}$, provide the best performances in almost all situations (although they are initialized randomly), except on SPORTS, where $\texttt{SEM}_{\mathbf{b}}$ achieves substantially better results than the other methods. As opposed to $\texttt{EM}_{\mathbf{b}}$ and $\texttt{CEM}_{\mathbf{b}}$ which depend strongly on their starting positions, the stochastic variants, i.e, $\texttt{SAEM}_{\mathbf{b}}$ and $\texttt{CAEM}_{\mathbf{b}}$ are not sensitive to their initial positions. For instance, in the case of high overlapping clusters as in CLASSIC4, NG20 and SPORTS, we observe that $\texttt{EM}_{\mathbf{b}}$ and $\texttt{CEM}_{\mathbf{b}}$ suffer from a convergence to a poor local maximum of the likelihood function, when they are initialized randomly. The $\texttt{SAEM}_{\mathbf{b}}$ and $\texttt{CAEM}_{\mathbf{b}}$ algorithms, however, avoid this difficulty and converge to a better local optimum of the likelihood function, see Table 4; this is due to the advantage of $\texttt{SEM}_{\mathbf{b}}$ in the begining.

[4] All algorithms, except stochastic variants, provide substantially better results when they are initialized using `Skmeans` (in almost all situations).

Table 2 Comparison of average NMI and ARI on CSTR, CLASSIC4, WEBACE

Methods	CSTR		CLASSIC4		WEBACE	
	NMI	ARI	NMI	ARI	NMI	ARI
`Skmeans`	0.732 ± 0.026	0.772 ± 0.025	0.591 ± 0.020	0.468 ± 0.011	0.613 ± 0.008	0.423 ± 0.026
	(0.759)	(0.807)	(0.595)	(0.476)	(0.620)	(0.394)
`CEM`	0.734 ± 0.025	0.774 ± 0.024	0.413 ± 0.011	0.199 ± 0.018	0.619 ± 0.011	0.398 ± 0.021
	(0.759)	(0.807)	(0.410)	(0.194)	(0.623)	(0.412)
`EM`	0.741 ± 0.026	0.777 ± 0.026	0.406 ± 0.013	0.190 ± 0.015	0.614 ± 0.014	0.385 ± 0.034
	(0.768)	(0.808)	(0.403)	(0.184)	(0.623)	(0.397)
`DAEM`	0.779 ± 0.013	0.813 ± 0.014	0.591 ± 0.002	0.471 ± 0.002	0.620 ± 0.008	0.427 ± 0.022
	(0.783)	(0.817)	(0.592)	(0.472)	(0.628)	(0.468)
`CoclusMod`	0.701 ± 0.02	0.693 ± 0.04	0.709 ± 0.020	0.675 ± 0.050	0.602 ± 0.020	0.566 ± 0.03
	(0.722)	(0.730)	(0.727)	(0.680)	(0.615)	(0.570)
$\texttt{CEM}_\mathbf{b}$	0.754 ± 0.024	0.804 ± 0.022	0.660 ± 0.003	0.467 ± 0.003	0.623 ± 0.011	0.479 ± 0.038
	(0.789)	(0.830)	(0.665)	(0.473)	(0.637)	(0.519)
$\texttt{EM}_\mathbf{b}$	0.754 ± 0.022	0.803 ± 0.022	0.660 ± 0.002	0.466 ± 0.002	0.624 ± 0.008	0.481 ± 0.025
	(0.792)	(0.837)	(0.668)	(0.473)	(0.639)	(0.523)
$\texttt{SEM}_\mathbf{b}$	0.776 ± 0.022	0.820 ± 0.024	0.691 ± 0.031	0.705 ± 0.053	0.567 ± 0.044	0.582 ± 0.035
	(0.807)	(0.846)	(0.721)	(0.735)	(0.597)	(0.588)
$\texttt{CAEM}_\mathbf{b}$	0.794 ± 0.014	0.833 ± 0.013	0.735 ± 0.033	0.751 ± 0.048	0.640 ± 0.007	0.666 ± 0.019
	(0.817)	(**0.851**)	(0.746)	(0.772)	(0.658)	(0.688)
$\texttt{SAEM}_\mathbf{b}$	0.795 ± 0.011	0.830 ± 0.010	0.746 ± 0.023	0.756 ± 0.039	0.644 ± 0.015	0.656 ± 0.021
	(**0.821**)	(**0.851**)	(**0.773**)	(**0.798**)	(**0.661**)	(**0.689**)

Bold face numbers indicate best performing method

Table 3 Comparison of average NMI and ARI on NG20, SPORTS, TDT2

Methods	NG20		SPORTS		TDT2	
	NMI	ARI	NMI	ARI	NMI	ARI
Skmeans	0.542 ± 0.013	0.375 ± 0.016	0.614 ± 0.044	0.405 ± 0.053	0.790 ± 0.012	0.492 ± 0.031
	(0.555)	(0.379)	(0.627)	(0.442)	(0.801)	(0.514)
CEM	0.467 ± 0.013	0.149 ± 0.021	0.446 ± 0.048	0.151 ± 0.067	0.750 ± 0.021	0.436 ± 0.048
	(0.484)	(0.150)	(0.455)	(0.177)	(0.769)	(0.466)
EM	0.465 ± 0.013	0.143 ± 0.020	0.444 ± 0.049	0.149 ± 0.067	0.751 ± 0.019	0.438 ± 0.041
	(0.481)	(0.141)	(0.453)	(0.174)	(0.771)	(0.489)
DAEM	0.556 ± 0.018	0.369 ± 0.022	0.618 ± 0.005	0.398 ± 0.016	0.795 ± 0.007	0.462 ± 0.011
	(0.576)	(0.393)	(0.620)	(0.416)	(0.806)	(0.456)
`CoclusMod`	0.487 ± 0.02	0.285 ± 0.019	0.590 ± 0.03	0.536 ± 0.04	0.793 ± 0.01	0.795 ± 0.02
	(0.516)	(0.311)	(0.623)	(0.549)	(0.818)	(**0.809**)
$\text{CEM}_\mathbf{b}$	0.582 ± 0.011	0.388 ± 0.024	0.558 ± 0.039	0.508 ± 0.080	0.799 ± 0.014	0.657 ± 0.032
	(0.594)	(0.403)	(0.608)	(0.602)	(0.817)	(0.719)
$\text{EM}_\mathbf{b}$	0.585 ± 0.010	0.390 ± 0.025	0.564 ± 0.037	0.517 ± 0.072	0.799 ± 0.015	0.658 ± 0.034
	(0.601)	(**0.425**)	(0.617)	(0.605)	(**0.821**)	(0.699)
$\text{SEM}_\mathbf{b}$	0.534 ± 0.013	0.347 ± 0.022	0.670 ± 0.018	0.713 ± 0.033	0.718 ± 0.020	0.619 ± 0.022
	(0.553)	(0.367)	(**0.700**)	(**0.767**)	(0.748)	(0.656)
$\text{CAEM}_\mathbf{b}$	0.605 ± 0.010	0.381 ± 0.019	0.601 ± 0.013	0.602 ± 0.021	0.794 ± 0.011	0.723 ± 0.020
	(0.608)	(0.385)	(0.612)	(0.613)	(0.806)	(0.749)
$\text{SAEM}_\mathbf{b}$	0.610 ± 0.011	0.393 ± 0.012	0.613 ± 0.015	0.620 ± 0.016	0.791 ± 0.010	0.709 ± 0.021
	(**0.615**)	(0.390)	(0.621)	(0.625)	(**0.821**)	(0.752)

Bold face numbers indicate best performing method

Table 4 Comparison of classification log-likelihood

Methods	CSTR	CLASSIC4	WEBACE	NG20	SPORTS	TDT2
$\mathrm{EM_b}$	976,533.3	122,329,895	4,866,978	3,411,592,328	432,277,941	1,299,234,559
$\mathrm{CEM_b}$	976,520.4	122,329,033	4,866,769	3,411,588,346	432,276,368	1,299,232,734
$\mathrm{CAEM_b}$	976,540	122,341,547	4,867,537	3,411,836,825	432,282,583	1,299,303,290
$\mathrm{SAEM_b}$	**976,547**	**122,341,562**	**4,868,739**	**3,411,838,557**	**432,286,341**	**1,299,304,500**

Bold face numbers indicate best performing method

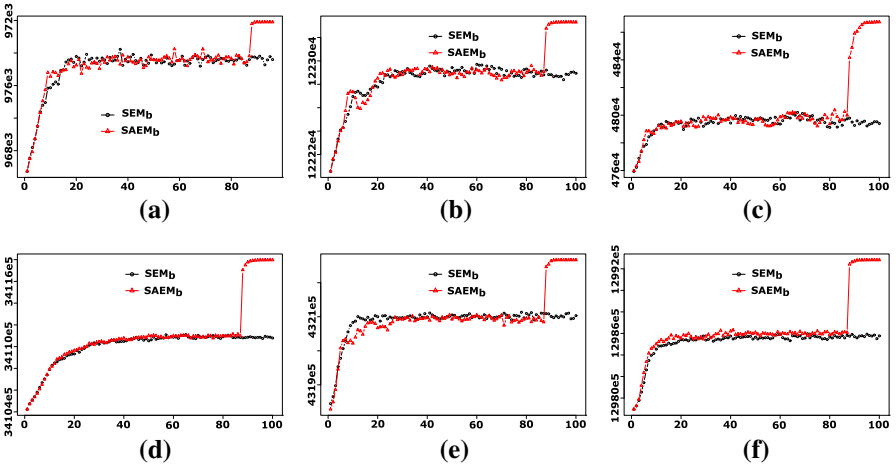

Fig. 3 Classification log-likelihood over iterations, on the different datasets

Figure 3 illustrates the behaviour of $\mathrm{SAEM_b}$ and its advantage compared to $\mathrm{SEM_b}$, the same behaviour is observed for $\mathrm{CAEM_b}$.

7.2.5 NMI versus ARI

The proposed algorithms provide high performances in terms of both NMI and ARI, while the other *movMFs*-based methods sometimes provide good NMI but low ARI as this is the case with almost all datasets, except CSTR. Figure 4 confirms this remark; we observe that the behavior of NMI and that of ARI are more often in keeping with our algorithms rather than with *movMFs*-based algorithms,[5] i.e, DAEM, EM, CEM and Skmeans. The explanation is that *movMFs*-based clustering methods tend to merge small clusters and try to split larger ones into comparably sized clusters, as it has been already emphasized in Banerjee et al. (2005). In fact, unlike ARI, the NMI measure is less sensitive to clusters merging and/or splitting. Our *dbmovMFs*-based algorithms, however, thanks to the centroids orthonormality assumption, avoid the above difficulty, and are able to discover large as well as small clusters (see Fig. 5 and confusion matrices

[5] For presentation purposes we omit CEM, $\mathrm{EM_b}$ and $\mathrm{CEM_b}$ from Fig. 4, but from Tables 2 and 3 it is straightforward to verify that our comments still holds for these algorithms.

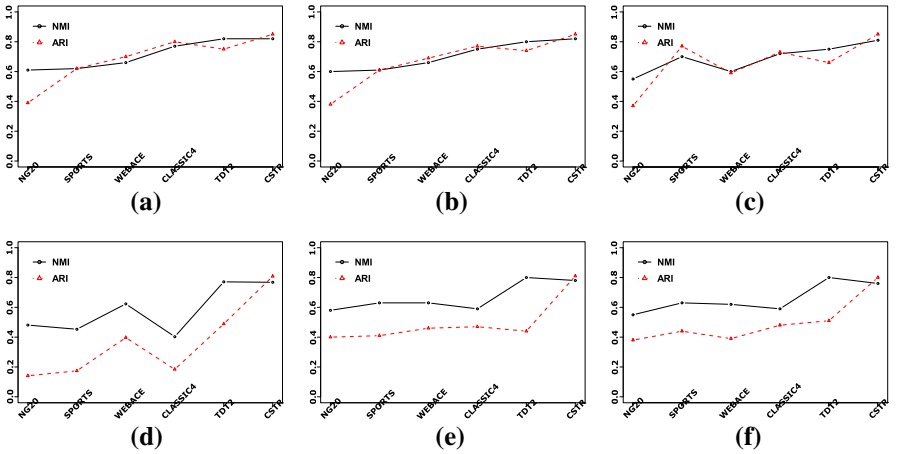

Fig. 4 Best NMI and ARI over all datasets for each method. Our algorithms provide good performances in terms both NMI and ARI while *movMFs*-based approach sometimes provide good NMI but low ARI. For instance, on SPORTS $\text{EM}_{\mathbf{b}}$ provides NMI = 0.62 and ARI = 0.61 while `Skmeans` provides NMI = 0.63 and ARI = 0.44 (see confusion matrices of Table 5)

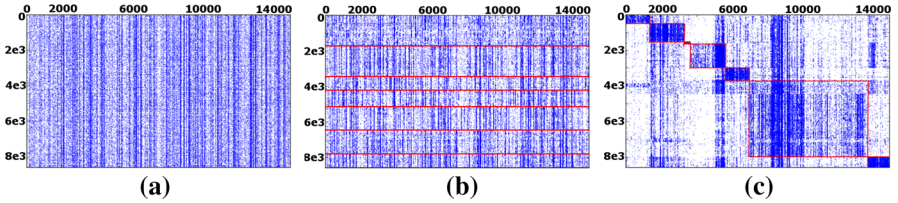

Fig. 5 SPORTS dataset: **a** original, **b** reorganized according to `Skmeans`'s row partition, **c** reorganized according to $\text{EM}_{\mathbf{b}}$'s row and column partitions

of Table 5). Furthermore, as we can see from Fig. 5, $\text{EM}_{\mathbf{b}}$ reveals a more interesting structure (i.e, block diagonal) than `Skmeans`.

7.2.6 Impact of concentration parameters

The `Skmeans` algorithm almost always outperforms its generalized variants `EM` and `CEM`. This is due to high concentration parameters κ_h in the normalization terms $c_d(\kappa_h)$ that involve Bessel functions. As it has been highlighted by Banerjee et al. (2005), in the case of large positive matrices, all the data lie on the first orthant of a d-dimensional hypersphere, thereby the concentration of such data is implicitly high. As a result, the concentration parameters κ_h of the vMF distributions are high and increase exponentially with the dimensionality of the data. Our model *dbmovMFs* alleviates issue thanks to its implicit dimensionality reduction property. In Fig. 6 we report the distribution of the concentration parameters estimated by our *dbmovMFs*-based algorithms and *movMFs*-based algorithms. We clearly observe that, $\text{CEM}_{\mathbf{b}}$, $\text{EM}_{\mathbf{b}}$, $\text{CAEM}_{\mathbf{b}}$ and $\text{SAEM}_{\mathbf{b}}$ yield substantially lower concentration parameters than `EM` and `CEM`.

Table 5 SPORTS dataset: confusion matrices crossing the row clusters obtained by both algorithms (rows) and the true row clusters (columns). The column $z_{.h}$ indicates the sizes of clusters

	EM_b (NMI = 0.617, ARI = 0.605)								Skmeans (NMI = 0.627 , ARI = 0.442)							
	1	2	3	4	5	6	7	$z_{.h}$	1	2	3	4	5	6	7	$z_{.h}$
1	3338	219	41	27	487	48	82	4242	1705	1	0	1	1	0	0	1708
2	8	1008	0	0	16	0	0	1032	0	904	0	0	1	0	1	906
3	24	22	5	0	39	5	7	102	1	1	0	0	798	1	0	801
4	36	138	0	0	449	2	2	627	1326	0	0	0	6	0	0	1332
5	3	14	0	1	1345	1	10	1374	81	30	24	6	1171	5	4	1321
6	3	8	99	94	10	280	0	494	298	473	121	115	369	330	23	1729
7	0	1	0	0	0	0	708	709	1	1	0	0	0	0	781	783

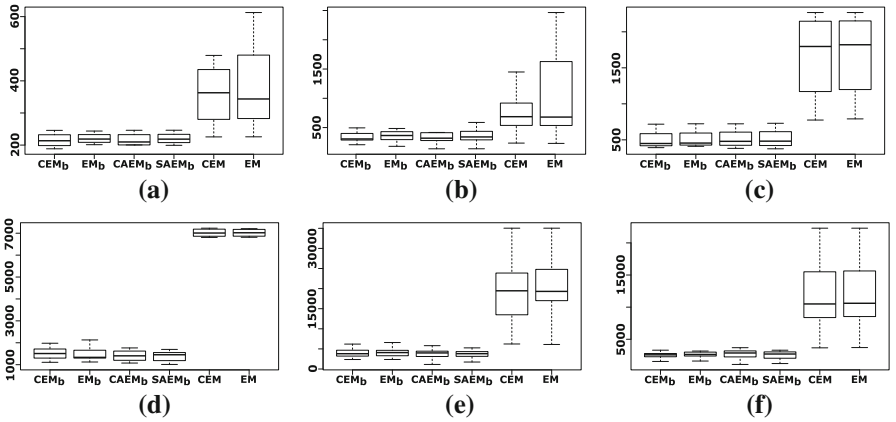

Fig. 6 Distribution of concentration parameters

7.2.7 Assessing the number of co-clusters

Often in practice the number of clusters is not known and should be determined by the user. Assessing the number of clusters is, however, not straightforward and still remains one of the biggest challenges in machine learning. Fortunately, in our case we can rely to well-established statistical theory of model selection since our algorithms are based on the maximization of the likelihood. More precisely, we can use Information Criteria (IC), such the Akaike information criterion (AIC) (Akaike 1998), AIC3 (Bozdogan 2000), Bayesian information criterion (BIC) (Schwarz 1978) or integrated classification likelihood (ICL) (Biernacki et al. 2000). These criteria (IC) measure the quality of a model given some observed data and take the following form

$$\mathrm{IC}(k) = -2 \ln \hat{L} + 2\gamma \times k \tag{8}$$

where k denotes the number of parameters to be estimated, γ a penalty coefficient and $\hat{L}$ is the maximized value of the log-likelihood function. With $\gamma = 1$ we have the AIC criterion, $\gamma = 3/2$ we obtain AIC3 and $\gamma = (\log n)/2$ leads to the BIC criterion where n is the sample size—the number of rows/documents in our case. And last, ICL can be obtained from (8) by replacing $\hat{L}$ with the classification log-likelihood $\hat{L}_c$ and setting γ to $(\log n)/2$.

Information criteria can be used to select the best model, which yields the lowest value in terms of the selected information criterion, among a set of models. In our case, we are interested in selecting the number of clusters, we therefore study the behaviour of AIC, AIC3 and BIC when varying the number of clusters, on each dataset.

For our experiments, we use the value of $\hat{L}$ resulting from the simulated annealing *dbmovMFs* ($\mathtt{SAEM}_{\mathbf{b}}$) algorithm as it seems to provide the best performances in almost all cases. Assessing the number of free parameters in *dbmovMFs* is not straightforward due to the mixing of continuous and discrete parameters. The parameters $\boldsymbol{\alpha}$ and κ contain $g - 1$ and g parameters, respectively. The column partition $\mathbf{w}$—that is treated as a parameter in our case—contains the same number of parameters whatever the

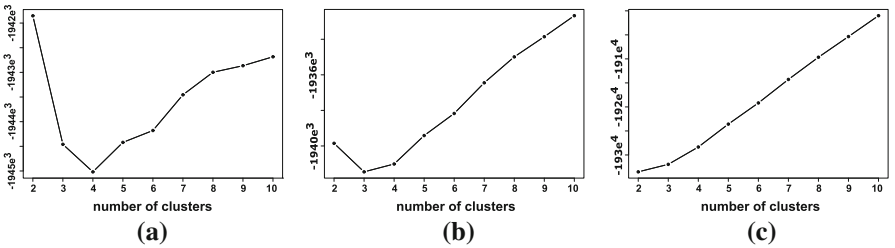

Fig. 7 CSTR dataset (True number of classes: 4)

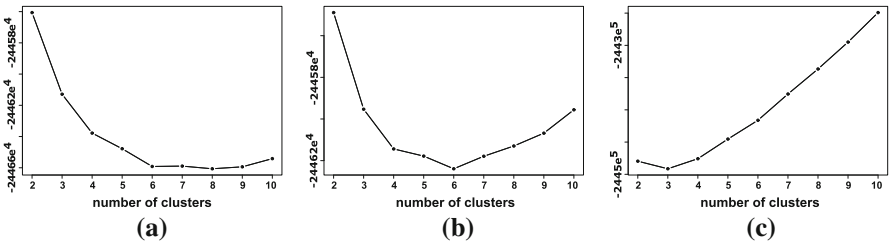

Fig. 8 CLASSIC4 dataset (True number of classes: 4)

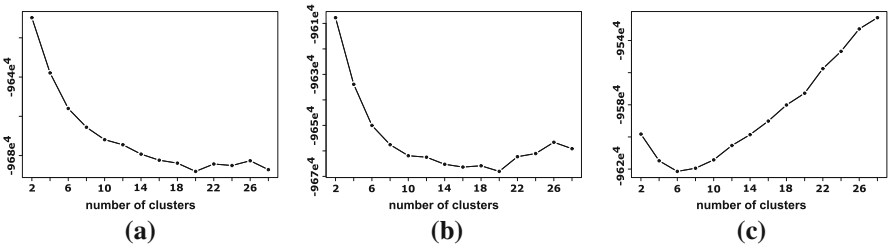

Fig. 9 WEBACE dataset (True number of classes: 20)

number of column clusters. Nevertheless, as pointed by van Dijk et al. (2009), the number of possible values for each parameter increases with the number of clusters, it is therefore more convenient to consider $\mathbf{w}$ under its matrix form $\mathbf{W}$ of size $(d \times g)$, where each row indicates to which cluster the corresponding column belongs. The effective number of free parameters in *dbmovMFs* is therefore $k = g \times (d + 2) - 1$ due to g concentration parameters κ_h's, $g - 1$ mixing proportions α_h's and $d \times g$ column cluster parameters w_{jh}'s.

In Figs. 7, 8, 9, 10, 11 and 12 are depicted the values of AIC, AIC3 and BIC in function of the number of clusters, over different data sets (notice that the ICL criterion, not reported here for presentation purpose, behaves like BIC). First of all, the penalty term of the BIC/ICL criterion, due to $\log(n)$ in comparison with AIC and AIC3 which do not depend on the number of documents, seems to be too strong when n is very high; it is the case of WEBACE, NG20, SPORTS and TDT2. From Figs. Figs. 7, 8, 9, 10, 11 and 12 we observe that AIC and AIC3 are able to identify a number of clusters that is close or even equal to the right number of clusters on almost all data sets, except on TDT2

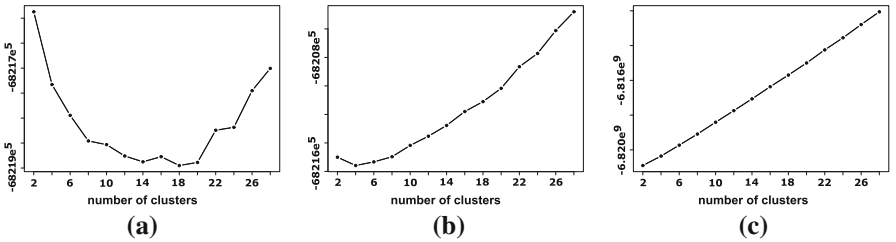

Fig. 10 NG20 dataset (True number of classes: 20)

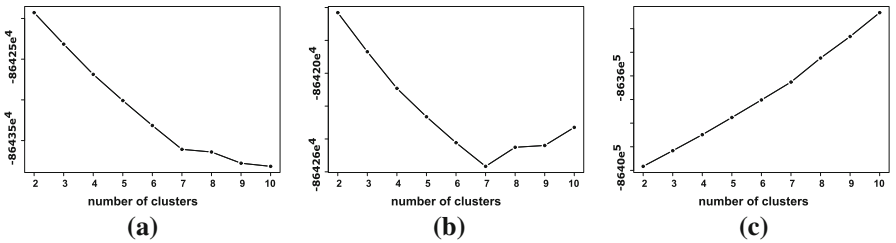

Fig. 11 SPORTS dataset (True number of classes: 7)

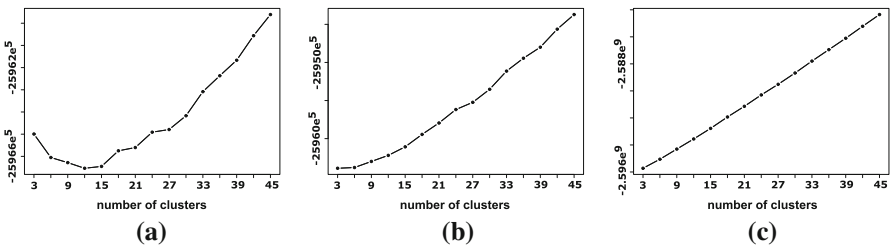

Fig. 12 TDT2 dataset (True number of classes: 30)

where both criteria underestimate the number of clusters. This suggests that, in our case, a convenient penalty term γ lies between 1 and $3/2$.

Furthermore, AIC seems to be slightly better than AIC3 since the latter tends to underestimate the right number of clusters on some data sets such as NG20 and CSTR. Another notable result is that, on CLASSIC4, that contains 4 classes, AIC3 suggests 6 clusters while AIC suggests a number of clusters between 6 and 8, this is, however, not an issue. In fact, as illustrated in Table 7 some classes in CLASSIC4 are composed of several semantically coherent sub-clusters. Thereby, an interesting approach would be to consider the number of clusters determined by AIC/AIC3 as a starting point, which can then be tuned by exploring the top terms of the obtained term clusters so as to decide whether the number of clusters should be increased, decreased or kept unchanged. Next, we investigate the interpretation of term clusters.

Table 6 Word clusters discovered by $\texttt{SAEM}_{\mathbf{b}}$ on CLASSIC4

C1	C2	C3	C4
Algorithm, program	Librari, inform	Flow, boundari	Cell, patient
System, comput	Scienc, research	Layer, pressur	Rat, growth
Languag, method	Book, servic	Heat, wing	Blood, acid
Function, gener	Scientif, index	Number, bodi	Hormon, tissu
Problem, data	Journal, retriev	Solut, shock	Diseas, cancer

Each cluster is represented by its top 10 words. Clusters C1, C2, C3 and C4 correspond respectively to CACM, CISI, CRANFIELD and MEDLINE

Table 7 Clustering of CLASSIC4 into 7 co-clusters by $\texttt{SAEM}_{\mathbf{b}}$: the obtained word clusters represented by their top 10 terms, sorted according to their popularity

CACM		CISI	CRANFIELD	MEDLINE		
C1	C2	C3	C4	C5	C6	C7
Algorithm	Program	Librari	Flow	Cell	Children	Patient
Function	System	Inform	Boundari	Rat	Child	Cancer
Integr	Comput	Scienc	Layer	Growth	Speech	Ventricular
Matrix	Languag	Research	Pressur	Acid	Autist	Diseas
Polynomi	Data	Book	Heat	Hormon	Anxieti	Arteri
Permut	Problem	Servic	Wing	Tissu	Disord	Therapi
Squar	Structur	Index	Number	Activ	Joint	Treatment
Invers	Process	Scientif	Bodi	Dna	Visual	Pulmonari
Fit	Time	Retriev	Solut	Protein	Syndrome	Defect
Random	Gener	Journal	Shock	Kidnei	Autism	Breast

7.2.8 Interpretation of term clusters

Although we focused on document clustering, notice that our algorithms offer term clusters. Each of them describes a single document cluster and thereby, allows to understand the semantic meaning of document clusters. Tables 6, 7 and 8 provide typical examples, where the four, seven and top six term clusters resulting from $\texttt{SAEM}_{\mathbf{b}}$ on CLASSIC4 and TDT2 respectively, are represented by their top 15 terms. The top terms of each co-cluster were obtained by keeping only the terms that appear in most documents in the considered cluster. The obtained term clusters are meaningful and help to make sense of the corresponding document clusters; they are semantically coherent. More interestingly, and as illustrated in Table 7 on CLASSIC4, when the requested number of clusters (7) is greater than the natural number of clusters (4), our algorithm tries to split the natural clusters into semantically coherent sub-clusters. This suggests that, when the true number of clusters is not known, one can request a relatively high number of clusters and then merge them adequately, based on the obtained term clusters.

Table 8 The top 6 word clusters resulting from $\texttt{SAEM}_{\mathbf{b}}$ on dataset TDT2

C10	C24	C28	C12	C25	C3
Iraq	Percent	Lewinsky	Suharto	Nuclear	Tobacco
Un	Asian	Starr	Indonesia	Pakistan	Smoking
Weapons	Economic	Clinton	Jakarta	India	Industry
Iraqi	Financial	President	Indonesian	Test	Bill
Inspectors	Economy	White	Habibie	Treaty	Senate
Baghdad	Market	House	Imf	Kashmir	Legislation
United	Stock	Monica	Suhartos	Sharif	Companies
Saddam	Crisis	Grand	Political	Islamabad	Settlement
Annan	Yen	Jury	Student	Conducted	Congress
Council	Dollar	Counsel	Reform	Five	Cigarette

Each cluster is represented by its top 10 terms, sorted according to their popularity

8 Conclusion and future work

We propose *dbmovMFs* a novel generative co-clustering model tailored for directional data lying on the surface of a unit hypersphere. The introduction of column clustering into a mixture of vMF distribution seems to be beneficial from a number of perspectives: (i) in terms of inference this involves a substantial reduction of the complexity of the model regarding the number of free parameters to be estimated and thereby yielding a parsimonious model, (ii) it makes it possible to exploit the inherent duality between rows and columns which improves the performance, in terms of document clustering, of vMF-based mixture models by a noticeable amount, as demonstrated in our experiments, (iii) it alleviates the high concentration parameter κ issue, by performing an implicitly adaptive dimensionality reduction at each stage, and finally, (iiii) it has the advantage of producing directly interpretable clusters and co-clusters, which may also help to assess the number of clusters when the latter is not known, as emphasized in Sect. 7.2.8.

The good performances of our algorithms motivate further investigations. For instance, it would be interesting to propose a way to simultaneously use estimates obtained by $\texttt{SEM}_{\mathbf{b}}$ and $\texttt{EM}_{\mathbf{b}}$/$\texttt{CEM}_{\mathbf{b}}$ in an iterative process to overcome therefore the difficulty of managing parameter $\mathbf{w}$. We could also improve *dbmovMFs* by considering a Bayesian formulation that would enable sharing of information between co-clusters. Future improvements could also involve the development of temporal, incremental and on-line variants of our algorithms. Such variants would be of great interest for applications such as collaborative filtering where available information evolves frequently. Finally, it would be opportune to investigate again the problem of the number of co-clusters which remains one of the most important challenge in co-clustering. For instance, an interesting strategy could be to treat the problem of parameter estimation and assessing the number of co-clusters simultaneously (Wyse and Friel 2012).

A. Appendix

A.1 Maximum likelihood estimate

The expectation of the classification log-likelihood of *dbmovMFs* is given by

$$\mathbb{E}[L_c(\Theta|\mathcal{X}, \mathbf{Z})] = \sum_h \tilde{z}_{.h} \log \alpha_h + \sum_h \tilde{z}_{.h} \log(c_d(\kappa_h)) + \sum_{h,i,j} \tilde{z}_{ih} w_{jh} \kappa_h \mu_{hh} x_{ij} \tag{9}$$

where $\tilde{z}_{.h} = \sum_i \tilde{z}_{ih}$. We first maximize the expectation of the classification log-likelihood (9) with respect to α_h, subject to the constraint $\sum_h \alpha_h = 1$. The corresponding Lagrangian, up to terms which are not function of α_h, is given by $L(\boldsymbol{\alpha}, \lambda) = \sum_h \tilde{z}_{.h} \log \alpha_h + \lambda_h (1 - \sum_h \alpha_h)$. Taking derivatives with respect to α_h, we obtain $\frac{\partial L(\boldsymbol{\alpha}, \lambda)}{\partial \alpha_h} = \frac{\tilde{z}_{.h}}{\alpha_h} - \lambda$. Setting this derivative to zero leads to $\tilde{z}_{.h} = \lambda \alpha_h$. Summing both sides over all h yields $\lambda = n$, thereby the maximizing value of the parameter α_h is given by $\hat{\alpha}_h = \frac{\tilde{z}_{.h}}{n}$. In the same manner, to maximize expectation (9) with respect to $\boldsymbol{\mu}_h^{\mathbf{w}}$, subject to the constraint $||\boldsymbol{\mu}_h^{\mathbf{w}}|| = 1$, we form the corresponding Lagrangian by isolating the terms depending on $\boldsymbol{\mu}_h^{\mathbf{w}}$, this leads to

$$L(\boldsymbol{\mu}, \lambda) = \sum_{h,i,j} \tilde{z}_{ih} w_{jh} \kappa_h \mu_{hh} x_{ij} + \lambda_h \left(1 - \sum_j w_{jh} \mu_{hh}^2\right).$$

Taking the derivative with respect to μ_{hh}, we obtain:

$$\frac{\partial L(\boldsymbol{\mu}, \lambda)}{\partial \mu_h} = \sum_{i,j} \tilde{z}_{ih} w_{jh} \kappa_h x_{ij} - 2\lambda w_{.h} \mu_{hh}$$

where $w_{.h} = \sum_j w_{jh}$. Setting this derivative to zero, we obtain $\lambda \mu_{hh} = \frac{\sum_{i,j} \tilde{z}_{ih} w_{jh} \kappa_h x_{ij}}{2 w_{.h}}$. Thus, $\lambda^2 \mu_{hh}^2 = \frac{(\sum_{i,j} \tilde{z}_{ih} w_{jh} \kappa_h x_{ij})^2}{4 w_{.h}^2}$. Multiplying both sides by $w_{.h}$, yields: $\lambda^2 w_{.h} \mu_{hh}^2 = \frac{w_{.h} (\sum_{i,j} \tilde{z}_{ih} w_{jh} \kappa_h x_{ij})^2}{4 w_{.h}^2}$. Since $w_{.h} \mu_{hh}^2 = 1$, we have $\lambda = \kappa_h \frac{\sqrt{w_{.h} (\sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij})^2}}{2 w_{.h}}$ or $\lambda = \kappa_h \frac{\|\mathbf{r}_h^{\mathbf{w}}\|}{2 w_{.h}}$ where $\mathbf{r}_h^{\mathbf{w}}$ is a d dimensional vector, i.e, let $j' = 1, \ldots, d$, $r_{hj'}^{\mathbf{w}} = r_h^{\mathbf{w}} = \sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}$ if $w_{jh} = 1$ and $r_{hj'}^{\mathbf{w}} = 0$, otherwise. Hence, the maximizing value of the parameter μ_{hh} is given

$$\text{by} \quad \hat{\mu}_{hh} = \frac{\sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}}{\|\mathbf{r}_h^{\mathbf{w}}\|} = \frac{\sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}}{\sqrt{w_{.h} (\sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij})^2}} = \pm \frac{1}{\sqrt{w_{.h}}} \tag{10}$$

according to whether $r_h^{\mathbf{w}} = \sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}$ is positive or negative.

Next we concentrate on maximizing Eq. (9), with respect to the concentration parameters κ_h, subject to the constraint $\kappa_h > 0, \ \forall h$. The Lagrangian up to terms which

do not contain κ_h is given by $L(\kappa) = \sum_h \tilde{z}_{.h} \log(c_d(\kappa_h)) + \sum_{h,i,j} \tilde{z}_{ih} w_{jh} \kappa_h \hat{\mu}_{hh} x_{ij}$. Note that, by KKT conditions, the Lagrangian multiplier for the constraint $\kappa_h > 0$ has to be equal to zero. Taking the partial derivative of Eq. (8) with respect to κ_h, we obtain

$$\frac{\partial L(\kappa)}{\partial \kappa_h} = \tilde{z}_{.h} \frac{c_d'(\kappa_h)}{c_d(\kappa_h)} + \sum_{i,j} \tilde{z}_{ih} w_{jh} \hat{\mu}_{hh} x_{ij}.$$

Setting this derivative equal to zero, leads to: $\frac{c_d'(\kappa_h)}{c_d(\kappa_h)} = -\frac{\hat{\mu}_{hh} \times \sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}}{\tilde{z}_{.h}}$. Replacing $\hat{\mu}_{hh}$ by $\frac{\sum_{i,j} \tilde{z}_{ih} w_{jh} x_{ij}}{\|\mathbf{r}_h^{\mathbf{w}}\|}$ (see, Eq. 10), we obtain $\frac{c_d'(\kappa_h)}{c_d(\kappa_h)} = -\frac{\|\mathbf{r}_h^{\mathbf{w}}\|}{\tilde{z}_{.h} \hat{w}_{.h}}$. Let $s = d/2 - 1$, then:

$$\begin{aligned} c_d'(\kappa_h) &= \frac{s\kappa_h^{s-1}(2\pi)^{s+1} I_s(\kappa_h) - \kappa_h^s (2\pi)^{s+1} I_s'(\kappa_h)}{(2\pi)^{2s+2} I_s^2(\kappa_h)} \\ &= \frac{s\kappa_h^{s-1}}{(2\pi)^{s+1} I_s(\kappa_h)} - \frac{\kappa_h^s I_s'(\kappa_h)}{(2\pi)^{s+1} I_s^2(\kappa_h)} = c_d(\kappa_h) \left(\frac{s}{\kappa_h} - \frac{I_s'(\kappa_h)}{I_s(\kappa_h)} \right). \end{aligned} \tag{11}$$

Hence, we obtain

$$\frac{-c_d'(\kappa_h)}{c_d(\kappa_h)} = \frac{I_{s+1}(\kappa_h)}{I_s(\kappa_h)} = \frac{I_{d/2}(\kappa_h)}{I_{d/2-1}(\kappa_h)}. \tag{12}$$

The latter Eq. (12), arises from the use of the following recurrence formula (Abramowitz and Stegun (1964), page 376): $\kappa_h I_{s+1}(\kappa_h) = \kappa_h I_s'(\kappa_h) - s I_s(\kappa_h)$. Note that computing the maximizing value $\hat{\kappa}_h$ from Eq. (11) implies to inverse a ratio of Bessel function, a problem for which no a closed-form solution can be obtained. Thus, following (Banerjee et al. 2005), we propose to derive an accurate approximation of the concentration parameter, by using the following continued fraction formula:

$$\frac{I_{d/2}(\kappa_h)}{I_{d/2-1}(\kappa_h)} = \frac{1}{\frac{d}{\kappa_h} + \frac{1}{\frac{d+2}{\kappa_h} + \ldots}}. \tag{13}$$

Letting $\bar{r}_h^{\mathbf{w}} = \frac{\|\mathbf{r}_h^{\mathbf{w}}\|}{\tilde{z}_{.h} \hat{w}_{.h}} = \frac{I_{d/2}(\kappa_h)}{I_{d/2-1}(\kappa_h)}$ and using Eq. (13), we obtain: $\frac{1}{\bar{r}_h^{\mathbf{w}}} \approx \frac{d}{\kappa_h} + \bar{r}_h^{\mathbf{w}}$ which yields the following approximation: $\hat{\kappa}_h = \frac{d\bar{r}_h^{\mathbf{w}}}{1-(\bar{r}_h^{\mathbf{w}})^2}$. Finally, Banerjee et al. (2005) have empirically shown that adding the following correction term $\frac{-(\bar{r}_h^{\mathbf{w}})^3}{1-(\bar{r}_h^{\mathbf{w}})^2}$ results in a better approximation of $\hat{\kappa}_h$, which leads to: $\hat{\kappa}_h = \frac{d\bar{r}_h^{\mathbf{w}} - (\bar{r}_h^{\mathbf{w}})^3}{1-(\bar{r}_h^{\mathbf{w}})^2}$.

A.2 Concentration parameters: *dbmovMFs* versus *movMFs*

Hereafter, we provide the proofs for Proposition 1 and Theorem 1. The following proposition is useful and necessary to prove both Proposition 1 and Theorem 1.

Proposition 3 *Let* $\mathbf{r}$ *be a non-zero vector in* $\mathbb{R}^d$ *(i.e.,* $\mathbf{r} = (r_1, \ldots, r_d)^\top$*, such as* $d \geq 1$*). Let* $\mathbf{r}^d$ *be a vector in* $\mathbb{R}^d$*, such as all its component are equal to the sum of elements of* $\mathbf{r}$ *(i.e,* $\mathbf{r}^d = \sum_{j=1}^{d} r_j \mathbb{1}$ *where* $\mathbb{1}$ *denotes the constant one vector). Then* $\frac{\|\mathbf{r}^d\|}{d} \leq \|\mathbf{r}\|$ *with equality only if all components of* $\mathbf{r}$ *are equal, i.e,* $(r_1 = \cdots = r_d)$*.*

Proof Let $\mathbf{d}$ and $\mathbf{r}^+$ be two vectors in $\mathbb{R}^d$ defined as follows: $\mathbf{d} = \frac{1}{\sqrt{d}}\mathbb{1}$ and $\mathbf{r}^+$ with $r_j^+ = |r_j|, \quad \forall j \in \{1, \ldots, d\}$. We have

$$\frac{\|\mathbf{r}^d\|}{d} = \frac{\sqrt{d} \times \left|\sum_j r_j\right|}{d} \leq \frac{1}{\sqrt{d}} \times \sum_j \left|r_j\right| = \mathbf{d}^\top.\mathbf{r}^+ = \|\mathbf{d}\|\|\mathbf{r}^+\| \cos(\mathbf{d}, \mathbf{r}^+).$$

By definition of $\mathbf{r}^+$ and $\mathbf{d}$, we have $\|\mathbf{r}^+\| = \|\mathbf{r}\|$ and $\|\mathbf{d}\| = 1$, hence $\frac{\|\mathbf{r}^d\|}{d} \leq \|\mathbf{r}\| \cos(\mathbf{d}, \mathbf{r}^+)$. Since both $\mathbf{d}$ and $\mathbf{r}^+$ are non-zero vectors and lie on the first orthant of a d-dimensional unit hypersphere, by dividing both sides of the above inequality by $\|\mathbf{r}\|$, we get $0 \leq \frac{\|\mathbf{r}^d\|}{d\|\mathbf{r}\|} \leq \cos(\mathbf{d}, \mathbf{r}^+) \leq 1$. The right hand side equality holds only if $\mathbf{d}$ and $\mathbf{r}^+$ are collinear, thereby all components of $\mathbf{r}$ are equal (i.e,$r_1 = \cdots = r_d$). We now prove Proposition 1. □

Proof of Proposition 1. Based on Proposition 3 and the following inequality: $\|\mathbf{r}\| = \|p_1\mathbf{x}_1 + \cdots + p_n\mathbf{x}_n\| \leq \|p_1\mathbf{x}_1\| + \cdots + \|p_n\mathbf{x}_n\| = \sum_i p_i$, it is straightforward to verify that

$$0 \leq \|\mathbf{r}^d\| \leq d \times \sum_i p_i$$

In the following, we prove Theorem 1, which states that for a given row clustering $\mathbf{z}$, *dbmovMFs*-based algorithms lead to a concentration parameter that is less or equal to that of movMFs-based algorithms, for each cluster, and whatever the column partition $\mathbf{w}$. The following Lemma will be useful in the proof of Theorem 1. □

Lemma 1 *Let* a *and* b *be two real numbers in the interval* $[0, 1]$ *(i.e,* $0 \leq a \leq 1$ *and* $0 \leq b \leq 1$*). Then for all natural number* $n \geq 2$ $|a^n - b^n| \leq n|a - b|$ *with equality only if* $a = b$*.*

Proof For all natural number $n \geq 2$, we can use the following well known remarkable identity: $a^n - b^n = (a - b)\sum_{k=0}^{n-1} a^{n-1-k}b^k$. Since both a and b are positive, we have $|a^n - b^n| = |a - b|\sum_{k=0}^{n-1} a^{n-1-k}b^k$ as both a and b are in $[0, 1]$, we have $\sum_{k=0}^{n-1} a^{n-1-k}b^k \leq n$ thereby, $|a^n - b^n| = |a - b|\sum_{k=0}^{n-1} a^{n-1-k}b^k \leq n|a - b|$. □

Proof of Theorem 1. We first prove that $\frac{\bar{r}_h^{\mathbf{w}} d}{1-(\bar{r}_h^{\mathbf{w}})^2} \leq \frac{\bar{r}_h d}{1-(\bar{r}_h)^2}$, which corresponds to the approximation of the concentration parameters under *dbmovMFs* and *movMFs* without the correction term. Since both $\bar{r}_h^{\mathbf{w}}$ and $\bar{r}_h$ are in $]0, 1[$, it is straightforward to verify that if $\bar{r}_h^{\mathbf{w}} \leq \bar{r}_h$ then the above inequality is always verified. Thus, in what follows we aim to prove that $\bar{r}_h^{\mathbf{w}} \leq \bar{r}_h$. □

By definition, we have $\bar{r}_h = \frac{\|\mathbf{r}_h\|}{\sum_i \tilde{z}_{ih}}$ where $\mathbf{r}_h = \sum_i \tilde{z}_{ih}\mathbf{x}_i$. Now, let's define $\bar{r}'_h$ as:

$$\bar{r}'_h = \frac{\|\mathbf{r}'_h\|}{\sum_i \tilde{z}_{ih}} \quad \text{where} \quad r'_{hj} = \begin{cases} r_{hj}, & \text{if } w_{jh} = 1 \\ 0, & \text{otherwise.} \end{cases}$$

$\mathbf{r}'_h$ is a $w_{\cdot h}$ dimensional sub vector of $\mathbf{r}_h$, it follows from the above definition that $\bar{r}'_h \leq \bar{r}_h$. On the other hand, $\bar{r}^{\mathbf{w}}_h = \frac{\|\mathbf{r}^{\mathbf{w}}_h\|}{\sum_i \tilde{z}_{ih} \sum_j \hat{w}_{jh}}$, where $\mathbf{r}^{\mathbf{w}}_h$ denotes a $w_{\cdot h}$ dimensional vector, each its elements are equal to sum of elements of the hth co-cluster (i.e, $r^{\mathbf{w}}_{h1} = \cdots = r^{\mathbf{w}}_{hw_{\cdot h}} = r^{\mathbf{w}}_h = \sum_{i,j} \tilde{z}_{ih}\hat{w}_{jh}x_{ij}$), similarly we can show that each elements of $\mathbf{r}^{\mathbf{w}}_h$ are equal to sum of elements of r'_h (i.e, $r^{\mathbf{w}}_h = \sum_j r'_{hj}$). Hence using Proposition 3, where the dimensionality $d = w_{\cdot h}$, we have $\frac{\|\mathbf{r}^{\mathbf{w}}_h\|}{\sum_j \hat{w}_{jh}} \leq \|\mathbf{r}'_h\|$. Thus,

$$\bar{r}^{\mathbf{w}}_h = \frac{\|\mathbf{r}^{\mathbf{w}}_h\|}{\sum_i \tilde{z}_{ih} \sum_j \hat{w}_{jh}} \leq \bar{r}'_h = \frac{\|\mathbf{r}'_h\|}{\sum_i \tilde{z}_{ih}} \leq \bar{r}_h, \text{ thereby, } \frac{\bar{r}^{\mathbf{w}}_h d}{1 - \left(\bar{r}^{\mathbf{w}}_h\right)^2} \leq \frac{\bar{r}_h d}{1 - (\bar{r}_h)^2}.$$

We now prove that, $\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1-\left(\bar{r}^{\mathbf{w}}_h\right)^2} \leq \frac{\bar{r}_h d - \bar{r}^3_h}{1-(\bar{r}_h)^2}$.

As $\bar{r}^{\mathbf{w}}_h \leq \bar{r}_h$ we have $\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1-\left(\bar{r}^{\mathbf{w}}_h\right)^2} \leq \frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1-(\bar{r}_h)^2}$, then it is sufficient to prove that, $\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1-(\bar{r}_h)^2} \leq \frac{\bar{r}_h d - \bar{r}^3_h}{1-(\bar{r}_h)^2}$. We have

$$\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1 - (\bar{r}_h)^2} - \frac{\bar{r}_h d - \bar{r}^3_h}{1 - (\bar{r}_h)^2} = \frac{d(\bar{r}^{\mathbf{w}}_h - \bar{r}_h) + (\bar{r}^3_h - (\bar{r}^{\mathbf{w}}_h)^3)}{1 - (\bar{r}_h)^2}.$$

Based on Lemma 1, for all $d \geq 3$ we have $\left|\bar{r}^3_h - (\bar{r}^{\mathbf{w}}_h)^3\right| \leq d\left|\bar{r}^{\mathbf{w}}_h - \bar{r}_h\right|$. As $\bar{r}^{\mathbf{w}}_h \leq \bar{r}_h$ it is easy to verify that $d(\bar{r}^{\mathbf{w}}_h - \bar{r}_h) + (\bar{r}^3_h - (\bar{r}^{\mathbf{w}}_h)^3) \leq 0$. Based on the fact that $1 - (\bar{r}_h)^2 > 0$ we have $\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1-(\bar{r}_h)^2} \leq \frac{\bar{r}_h d - \bar{r}^3_h}{1-(\bar{r}_h)^2}$. Hence, using the above inequalities we get (for all $d \geq 3$)

$$\frac{\bar{r}^{\mathbf{w}}_h d - (\bar{r}^{\mathbf{w}}_h)^3}{1 - \left(\bar{r}^{\mathbf{w}}_h\right)^2} \leq \frac{\bar{r}_h d - \bar{r}^3_h}{1 - (\bar{r}_h)^2}.$$

A.3 Computational complexity in the worst case

Hereafter, we provide the proof of Proposition 2 (Sect. 6).

Proof (i) The computational bottleneck for *hard-dbmovMF* is with row, column assignments and concentration parameters updates.

First, we prove that the complexity of both row and column assignments is $O(nz)$. Let $\mathbf{x}_i$ denote the ith row of $\mathbf{X}$ (i.e, $\mathbf{x}_i$ is a d dimensional vector in $\mathbb{S}^{d-1}$). Assume that we look for a co-clustering of $\mathbf{X}$ into g co-clusters, and let $\boldsymbol{\mu}^{\mathbf{w}}_h$ be the hth centroid,

characterizing the hth co-cluster. The computational cost of the scalar product $(\boldsymbol{\mu}_h^{\mathbf{w}})^T \mathbf{x}_i$ is $O(x_i^h)$, where x_i^h is number of non-zeros entries of $\mathbf{x}_i$ within the hth column cluster, this complexity holds thanks to the form of $\boldsymbol{\mu}_h^{\mathbf{w}}$, see Eq. (2). Thereby, the complexity of the assignment of $\mathbf{x}_i$ is given in $O(x_i^*)$ (based on $O(x_i^1 + \cdots + x_i^g)$), where x_i^* denotes the number of non-zeros entries of $\mathbf{x}_i$. Therefore, the total cost of one row assignments step is $O(nz)$. Similarly, we can show that the cost of one column assignments step is $O(nz)$.

We now prove that the computational cost for updating concentration parameters is also $O(nz)$. The main computation for updating the hth concentration parameter is with the computation of $r_h^{\mathbf{w}}$. The computational cost of the latter term is given in $O(x_h^*)$, where x_h^* the number of non-zeros entries in the hth co-cluster. Thus, the complexity for updating all concentrations parameters is $O(nz)$, based on $O(x_1^* + \cdots + x_g^*)$ and the fact that at most all non-zeros entries in the matrix $\mathbf{X}$ are contained in the g diagonal co-clusters. Thereby, the total computational complexity of *hard-dbmovMF* is $O(it \cdot nz)$.

□

Proof (ii) As for *hard-dbmovMF* it is easy to verify that the total cost of row assignments and concentration parameters updates is given in $O(it \cdot nz)$ for *soft-dbmovMF*. Now we prove that in contrast to *hard-dbmovMF* the computational cost of column assignments for *soft-dbmovMF* is $O(it \cdot g \cdot nz)$. The computational bottleneck for column assignment step of *soft-dbmovMF* is with the terms $v_{hj} \leftarrow \sum_i \tilde{z}_{ih} x_{ij}$, $h \in \{1, \ldots, g\}$, $j \in \{1, \ldots, J\}$. The cost of v_{hj} is given in $O(x_j^*)$, where x_j^* is the number of non-zeros entries in the jth column. During the column assignment step for each cluster h and column j we compute v_{hj}, hence the computational cost of this step is given in $O(g \cdot nz)$. Therefore, the total computational cost of *soft-dbmovMF* is $O(it \cdot g \cdot nz)$, based on $O(it \cdot nz \cdot (g + 1))$. □

References

Abramowitz M, Stegun IA (1964) Handbook of mathematical functions: with formulas, graphs, and mathematical tables, vol 55. Courier Corporation, North Chelmsford

Ailem M, Role F, Nadif M (2016) Graph modularity maximization as an effective method for co-clustering text data. Knowl Based Syst 109:160–173

Ailem M, Role F, Nadif M (2017a) Model-based co-clustering for the effective handling of sparse data. Pattern Recognit 72:108–122

Ailem M, Role F, Nadif M (2017b) Sparse poisson latent block model for document clustering. IEEE Trans Knowl Data Eng 29(7):1563–1576

Akaike H (1998) Information theory and an extension of the maximum likelihood principle. In: Parzen E, Tanabe K, Kitagawa G (eds) Selected papers of Hirotugu Akaike. Springer, New York, pp 199–213

Banerjee A, Dhillon IS, Ghosh J, Sra S (2005) Clustering on the unit hypersphere using von Mises–Fisher distributions. J Mach Learn Res 6:1345–1382

Banfield JD, Raftery AE (1993) Model-based Gaussian and non-Gaussian clustering. Biometrics 49:803–821

Biernacki C, Celeux G, Govaert G (2000) Assessing a mixture model for clustering with the integrated completed likelihood. IEEE TPAMI 22(7):719–725

Bock HH (1979) Simultaneous clustering of objects and variables. In: Tomassone R (ed) Analyse des Données et Informatique. INRIA, Le Chesnay, pp 187–203

Bock HH (1994) Information and entropy in cluster analysis. In: Bozdogan H et al (eds) Proceedings of the First US/Japan Conference on the Frontiers of Statistical Modeling: an informational approach. Springer, Dordrecht, pp 115–147
Bozdogan H (2000) Akaike's information criterion and recent developments in information complexity. J Math Psychol 44(1):62–91
Celeux G, Diebolt J (1985) The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. Comput Stat Q 2(1):73–82
Celeux G, Diebolt J (1992) A stochastic approximation type EM algorithm for the mixture problem. Stoch Int J Probab Stoch Process 41(1–2):119–134
Celeux G, Govaert G (1992) A classification EM algorithm for clustering and two stochastic versions. Comput Stat Data Anal 14(3):315–332
Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B 39:1–38
Deodhar M, Ghosh J (2010) Scoal: a framework for simultaneous co-clustering and learning from complex data. ACM Trans Knowl Discov Data 4(3):11
Dhillon IS (2001) Co-clustering documents and words using bipartite spectral graph partitioning. In: ACM SIGKDD, pp 269–274
Dhillon IS, Modha DS (2001) Concept decompositions for large sparse text data using clustering. Mach Learn 42(1–2):143–175
Dhillon IS, Mallela S, Modha DS (2003) Information-theoretic co-clustering. In: ACM SIGKDD, pp 89–98. ACM
Gopal S, Yang Y (2014) Von Mises–Fisher clustering models. In: ICML, pp 154–162
Govaert G (1995) Simultaneous clustering of rows and columns. Control Cybern 24:437–458
Govaert G, Nadif M (2013) Co-Clustering. Wiley, New York
Govaert G, Nadif M (2016) Mutual information, phi-squared and model-based co-clustering for contingency tables. Advances in Data Analysis and Classification pp 1–34
Hanczar B, Nadif M (2010) Bagging for biclustering: application to microarray data. In: ECML/PKDD, pp 490–505
Hartigan JA (1975) Clustering algorithms, 99th edn. Wiley, New York
Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218
Labiod L, Nadif M (2011) Co-clustering for binary and categorical data with maximum modularity. In: ICDM, pp 1140–1145
Laclau C, Nadif M (2017) Diagonal latent block model for binary data. Stat Comput 27(5):1145–1163
Li T (2005) A general model for clustering binary data. In: SIGKDD, pp 188–197
Madeira SC, Oliveira AL (2004) Biclustering algorithms for biological data analysis: a survey. IEEE/ACM TCBB 1(1):24–45
Mardia KV, Jupp PE (2000) Directional statistics. Wiley series in probability and statistics. Wiley, New York
McLachlan G, Krishnan T (2007) The EM algorithm and extensions. Wiley, New York
McLachlan G, Peel D (2004) Finite mixture models. Wiley, New York
Nadif M, Govaert G (2010) Model-based co-clustering for continuous data. In: ICMLA, pp 175–180
Reisinger J, Waters A, Silverthorn B, Mooney RJ (2010) Spherical topic models. In: ICML, pp 903–910
Salah A, Nadif M (2017) Social regularized von Mises–Fisher mixture model for item recommendation. Data Min Knowl Discov 31:1–24
Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6(2):461–464
Strehl A, Ghosh J (2003) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. JMLR 3:583–617
van Dijk B, van Rosmalen J, Paap R (2009) A Bayesian approach to two-mode clustering. Econometric Institute, Erasmus University Rotterdam, Report no EI 2009-06, pp 1–26
Van Mechelen I, Bock HH, De Boeck P (2004) Two-mode clustering methods: a structured overview. Stat Methods Med Res 13(5):363–394
Vichi M (2001) Double k-means clustering for simultaneous classification of objects and variables. In: Borra S, Rocci R, Vichi M, Schader M (eds) Advances in classification and data analysis. Springer, Berlin, Heidelberg, pp 43–52
Wyse J, Friel N (2012) Block clustering with collapsed latent block models. Stat Comput 22(2):415–428
Zhong S, Ghosh J (2005) Generative model-based document clustering: a comparative study. Knowl Inf Syst 8(3):374–384