Neurocomputing 193 (2016) 133–147

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Hard and fuzzy diagonal co-clustering for document-term partitioning

Charlotte Laclau *, Mohamed Nadif

LIPADE, University of Paris Descartes. 45, rue des Saints-Pères. Paris, France

ARTICLE INFO

Article history:
Received 15 June 2015
Received in revised form
17 December 2015
Accepted 1 February 2016
Communicated by Zhaohong Deng
Available online 18 February 2016

Keywords:
Co-clustering
Fuzzy co-clustering
Document clustering

ABSTRACT

We propose a hard and a fuzzy diagonal co-clustering algorithms built upon the double K-means to address the problem of document-term co-clustering. At each iteration, the proposed algorithms seek a diagonal block structure of the data by minimizing a criterion based on both the variance within the class and the centroid effect. In addition to be easy-to-interpret and effective on sparse binary and continuous data, the proposed algorithms, Hard Diagonal Double K-means (DDKM) and Fuzzy Diagonal Double K-means (F-DDKM), are also faster than other state-of-the-art clustering algorithms. We evaluate our contribution using synthetic data sets, and real data sets commonly used in document clustering.

1. Introduction

Co-clustering, also known as biclustering or block-clustering, involves simultaneous clustering of a set of observations and a set of features in a data matrix. By creating permutations of rows and columns, co-clustering algorithms aim to reorganize the initial data matrix into homogeneous blocks. These blocks, also called co-clusters, can therefore be seen as subsets of the data matrix characterized by a set of observations and a set of features whose elements are similar. Since the work of Hartigan [1,2] and Govaert [3], co-clustering techniques have proven their importance in many areas such as bioinformatics [4–10], recommender systems [11,12], web mining [13,14] and text mining [15]. For a survey of the different structures of co-clusters and the different algorithms and approaches employed, the reader is referred to [16–22].

Co-clustering algorithms offer several advantages over simple clustering algorithms: for instance, they reduce the initial matrix into a simpler form with the same basic structure and require far less computation when compared with separate processing of the initial data set and its transpose. As a result, these methods are of increasing interest to the data mining community. One of the constraints that can be incorporated into a co-clustering method is to seek a block diagonal structure, *i.e.*, the number of clusters of observations is equal to the number of clusters of features. An illustration of this idea is given in Fig. 1 where (a) represents an original binary matrix, (b) represents the same matrix after a proper permutation of rows whilst (c) adds a permutation of columns resulting in a clear block diagonal structure. When the data set is typically represented by a sparse high-dimensional *document* × *term* matrix, these methods have proven again to be efficient when dealing with the problem of clustering or co-clustering. Their objective is to group documents based on words within them and to group words based on documents in which they appear. As we will see in this paper, seeking a diagonal structure on *document* × *term* matrices improve clustering results.

Conventional clustering methods often encounter issues when dealing with the sparsity and the high dimensionality that characterize this type of data. In [23], the author proposed a block diagonal algorithm applied to binary data. This algorithm alternates the clustering of observations and features minimizing the error between the original data matrix and the reconstructed matrix based on the cluster structure. In [24], the authors proposed ITCC, a divisive algorithm that directly minimizes an objective function based on the mutual information. In [15] a spectral algorithm has been proposed; it consists in building a bipartite graph from the *document* × *term* matrix which is partitioned to minimize the cut objective function. Another attempt to use graph related criteria for block diagonal clustering was presented in [25]. The proposed spectral co-clustering algorithm maximizes a generalization of the modularity that is further casted as a trace maximization problem.

Among different approaches devoted to co-clustering, the fuzzy approach is probably the least investigated one. In fuzzy co-clustering, observations no longer belong to one particular co-cluster but to all of them with a certain degree of belonging. Similar to hard co-clustering, we can distinguish between partitioning and probabilistic (co)-clustering approaches. The probabilistic methods can use the framework of mixture model as in [26] where the authors proposed block fuzzy c-models (block FCM) to deal with

* Corresponding author.
E-mail addresses: charlotte.laclau@parisdescartes.fr (C. Laclau), mohamed.nadif@parisdescartes.fr (M. Nadif).

http://dx.doi.org/10.1016/j.neucom.2016.02.003
0925-2312/

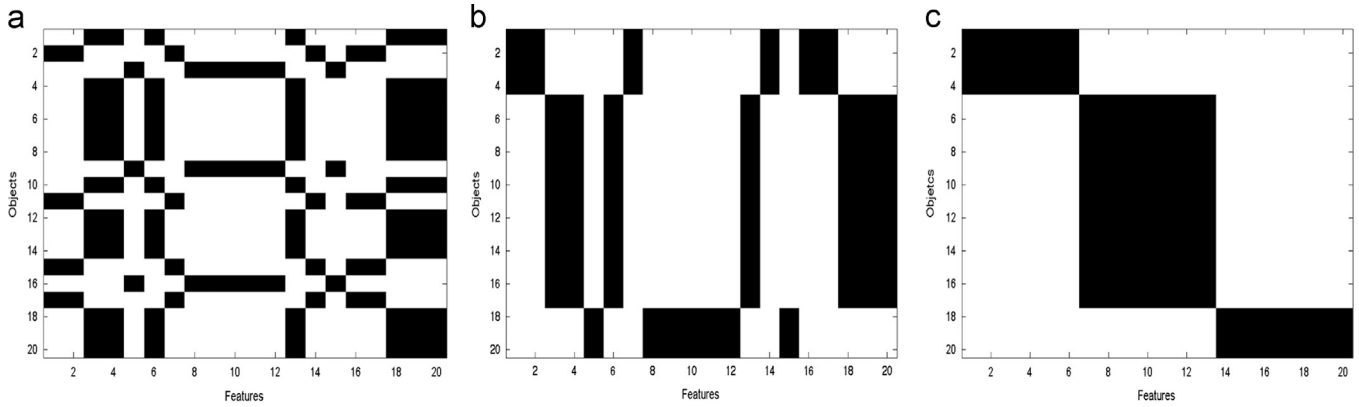

Fig. 1. (a) Original binary data, (b) data reorganized according to rows and (c) data reorganized according to rows and columns.

binary data. More recently, in [27] the authors proposed a block fuzzy k-modes (block FKM) to handle categorical data. Most of the fuzzy co-clustering algorithms belong to the family of partitioning methods and were developed to deal with co-occurrence tables and thus to address the problem of document clustering. Among them, we can cite the FCCM algorithm introduced in [28] where the fuzziness was introduced based on the entropy. They aim at clustering data in which features can be categorical and the distance between objects is not explicitly available by maximizing the degree of aggregation among the clusters. In [29], a version of FCCM more suitable for high dimensional data named Fuzzy-CoDok has been proposed; it uses Gini Index instead of the entropy to introduce the fuzziness. One major issue of Fuzzy-CoDok is that the fuzziness is introduced using Gini Index without taking into account the Khun and Tucker conditions [30] needed to ensure the optimal solution. Therefore, the convergence to a local optimum cannot always be guaranteed. For both FCCM and Fuzzy-CoDok the co-clustering is achieved to be a combination of object partitioning and feature ranking while in [31] and then [32], the authors proposed dual partitioning based on fuzzy co-clustering algorithms using the Ruspinis condition as the partitioning constraint. Some recent advances in fuzzy (co)-clustering and related subjects include [33–36].

In this paper we propose two new diagonal co-clustering algorithms based on the minimization of heterogeneity measures of blocks. Each measure relies on the variance intra blocks and the centroid effect studied in hierarchical clustering [37] and defined as the squared deviation from the mean entry in each block and the maximum entry in the input matrix. The proposed algorithms have strong convergence guarantees, are very efficient in terms of both co-clustering quality and computational speed on sparse data and therefore can deal with high dimensional data sets.

The remainder of this paper is organized as follows. Section 2 provides the needed background on Double K-means (DKM) algorithm and presents the challenge of diagonal co-clustering. Section 3 presents the Diagonal Double K-means (DDKM) algorithm that we propose. In order to deal with the situation of overlapping co-clusters we introduce Fuzzy Diagonal Double K-means (F-DDKM) in Section 4. Section 5 is devoted to numerical experiments on synthetic data sets to assess both algorithms on binary and continuous data. In Section 6, we compare DDKM and F-DDKM with state-of-the-art (co)-clustering algorithms on real *document* × *term* data sets showing the appropriateness of our contribution. Section 7 is devoted to a discussion of the limitations of our approach. The final section sums up the study and gives recommendations for further research.

Notation. Let $\mathbf{X}:=\{x_{ij}; i \in I; j \in J\}$ be a data matrix of size $n \times p$ where $I=\{1,\ldots,n\}$ and $J=\{1,\ldots,p\}$. The set I corresponds to the set of n objects and the set J to the set of p attributes. In the sequel, our aim consists in obtaining co-clustering of $\mathbf{X}$. Let $\mathbf{Z}=\{z_1,\ldots,z_n\}$ be a label vector, where $z_i \in \{1,\ldots,K\}$, that denotes the partition of I into K clusters and $\mathbf{W}=\{w_1,\ldots,w_p\}$ where $w_j \in \{1,\ldots,H\}$ denotes the partition of J into H clusters. The partition of I (respectively J) can be represented by a matrix of elements in $\{0,1\}^K$ (respectively $\{0,1\}^H$) satisfying $\sum_{k=1}^{K} z_{ik}=1$ (respectively $\sum_{h=1}^{H} w_{jh}=1$). Finally, to simplify the notation, the sums relating to rows, columns, row and column clusters are subscripted respectively by the letters $i=1,\ldots,n$, $j=1,\ldots,p$ and, $k=1,\ldots,K$, respectively, without indicating the implicit limits of variation. For example, the sum $\sum_{i,k}$ stands for $\sum_{i=1}^{n}\sum_{k=1}^{K}$.

2. Co-clustering and diagonal block structure

The co-clustering can be formulated as the search for a good matrix approximation of the original data matrix $\mathbf{X}$. The quality of the obtained result is determined by the approximation error that can be measured by a large class of loss functions, *e.g.*, the squared Euclidean distance. This approximation is generally achieved through an alternate least square minimization process (see, for instance, [38,16,39]). The Double K-means algorithm (DKM) [22] is also based on this principle.

2.1. Double K-means algorithm

The aim of DKM is to minimize an objective function $J(\mathbf{Z},\mathbf{W},\mathbf{G})$ where $\mathbf{Z}$ and $\mathbf{W}$ are the partitions and $\mathbf{G}:=\{g_{kh}; k \in \{1,\ldots,K\}, h \in \{1,\ldots,H\}\}$ is a $K \times H$ matrix, which can be viewed as a summary of the data matrix X (see Fig. 2).

Each element g_{kh} of $\mathbf{G}$ is called a prototype of the co-cluster $\mathbf{X}_{kh}:=\{x_{ij}; z_{ik}w_{jh}=1\}$. DKM adopts the squared Euclidean distance to measure the dissimilarity between the matrix $\mathbf{X}$ and the structure described in $\mathbf{Z}$, $\mathbf{W}$ and $\mathbf{G}$. Therefore, $J(\mathbf{Z},\mathbf{W},\mathbf{G})$ is given by

$$\mathcal{J}(\mathbf{Z},\mathbf{W},\mathbf{G})=\sum_{i,k,j,h} z_{ik}\times w_{jh}(x_{ij}-g_{kh})^2=\|\mathbf{X}-\mathbf{Z}\mathbf{G}\mathbf{W}^T\|^2, \quad (1)$$

where $\|\cdot\|$ denotes the Frobenius norm. It is easy to see that for a fixed $(\mathbf{Z};\mathbf{W})$ the optimal values of $\mathbf{G}$ are the means of $\mathbf{X}_{kh}$'s. The optimal partitions $\mathbf{Z}$ and $\mathbf{W}$ are obtained using an iterative algorithm. A version of DKM is presented in Algorithm 1 where $z_{.k}$ (resp. $w_{.h}$) represents the cardinality of the k-th cluster (resp. h-th cluster).

Algorithm 1. Double K-means (DKM).

input: $\mathbf{X}$, K, H
initialization: $\mathbf{Z}$ and $\mathbf{W}$
repeat
(1) Compute $g_{kh}=\sum_{i,j}\frac{z_{ik}w_{jh}x_{ij}}{z_{.k}w_{.h}}$, $\forall k,h$

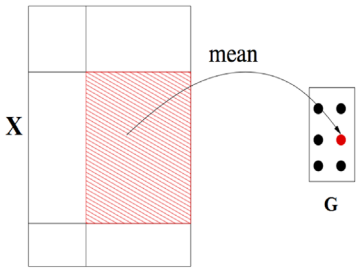

Fig. 2. Original data matrix **X** and its summary after co-clustering into 6 co-clusters.

(2) Update $z_i = \arg\min_k \sum_{j,h} w_{jh}(x_{ij} - g_{kh})^2,\ \forall i$

(3) Update $w_j = \arg\min_h \sum_{i,k} z_{ik}(x_{ij} - g_{kh})^2,\ \forall j$

until the J value change is small or there is no change.

output: **G**, **Z** and **W**

2.2. Block diagonal structure

The DKM algorithm appears to be not inefficient when looking for a one-to-one correspondence between two partitions **Z** and **W**. In order to deal with this specific case, we have to consider two assumptions: (1) we assume that $H=K$; (2) the diagonal structure involves imposing some constraints on **G**, for instance by taking $g_{kk} = \delta\ \forall k$. This leads us to the following criterion:

$$J(\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \sum_{i,j,k} z_{ik} w_{jk} (x_{ij} - \delta)^2, \tag{2}$$

where δ is assumed to be known. The choice of this parameter will be discussed in the next section. The partitions couple $(\mathbf{Z}, \mathbf{W})$ optimizing the criterion given in Eq. (2) is found using the following iterative algorithm:

- Update **Z**, the partition of objects, with **W** fixed. This leads to the following formula $z_i = \arg\min_k \sum_j w_{jk}(x_{ij} - \delta)^2$,
- Update **W**, the partition of features, with **Z** fixed. This leads to the following formula $w_j = \arg\min_k \sum_i z_{ik}(x_{ij} - \delta)^2$.

From these formulae, one observes that seeking the diagonal structure indirectly introduces a strong dependency between objects assignments (respectively, features assignments) to a block and the number of features that belong to this block (respectively, the number of objects). If we consider the assignment of objects, we have $(x_{ij} - \delta)^2 \geq 0,\ \forall i, j$; therefore, a higher number of features in a given block will decrease the chance for an object to be assigned to this particular block. The same phenomenon occurs in the assignment of features. Consequently, we have to take into account the size of each co-cluster in order to avoid empty blocks.

3. Diagonal Double K-means

3.1. Criterion and proposed algorithm

In order to correct the bias introduced by the diagonal structure and to avoid vanishing blocks, we propose a modified criterion that takes into account the number of elements in a block. This criterion takes the following form:

$$J(\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \sum_k \frac{1}{z_{.k} w_{.k}} \sum_{i,j} z_{ik} w_{jk} (x_{ij} - \delta)^2, \tag{3}$$

where $z_{.k} = \sum_i z_{ik}$ and $w_{.k} = \sum_j w_{jk}$ denote the number of objects and the number of features in the k-th block, respectively. Furthermore, it is interesting to note that the criterion given in Eq. (3) may be expressed depending on the variance of a given block ($\mathbf{Z}_k$, $\mathbf{W}_k$) and the squared deviation of its mean from the maximum input of the data:

$$J(\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \sum_{i,j,k} \frac{z_{ik} w_{jk}}{z_{.k} w_{.k}} (x_{ij} - \overline{x}_k)^2 + \sum_{i,j,k} \frac{z_{ik} w_{jk}}{z_{.k} w_{.k}} (\overline{x}_k - \delta)^2,$$

where $\overline{x}_k = \frac{1}{z_{.k} w_{.k}} \sum_{i,j} z_{ik} w_{jk} x_{ij}$ denotes the mean of the k-th block. Thus, the criterion can be rewritten as

$$J(\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \sum_k \frac{1}{z_{.k} w_{.k}} \sum_{i,j} z_{ik} w_{jk} (x_{ij} - \overline{x}_k)^2 + \sum_k (\overline{x}_k - \delta)^2 \sum_{i,j} \frac{z_{ik} w_{jk}}{z_{.k} w_{.k}}.$$

Since $\sum_{i,j} \frac{z_{ik} w_{jk}}{z_{.k} w_{.k}} = \frac{1}{z_{.k}} \sum_i z_{ik} \times \frac{1}{w_{.k}} \sum_j w_{jk} = 1$, we obtain

$$J(\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \sum_k \overline{s}_k^2 + \sum_k (\overline{x}_k - \delta)^2, \tag{4}$$

where $\overline{s}_k^2 = \frac{1}{z_{.k} w_{.k}} \sum_{i,j} z_{ik} w_{jk} (x_{ij} - \overline{x}_k)^2$ denotes the variance within the k-th block. The first term in Eq. (4) ensures the homogeneity of each block while the second one provides the homogeneity between centers of the blocks and δ. This objective function (Eq. (3)) can be optimized by an alternating optimization of two conditional criteria given **W** and **Z**, respectively:

$$\tilde{J}_1(\mathbf{X}, \mathbf{Z} | \mathbf{W}) = \sum_k \frac{1}{z_{.k}} \sum_i z_{ik} \frac{1}{w_{.k}} \sum_j w_{jk} (x_{ij} - \delta)^2$$

and

$$\tilde{J}_2(\mathbf{X}, \mathbf{W} | \mathbf{Z}) = \sum_k \frac{1}{w_{.k}} \sum_j w_{jk} \frac{1}{z_{.k}} \sum_i z_{ik} (x_{ij} - \delta)^2.$$

The optimization of $\tilde{J}_1$ and $\tilde{J}_2$ leads to the following update rules:

$$z_i = \arg\min_k \frac{1}{w_{.k}} \sum_j w_{jk} (x_{ij} - \delta)^2, \tag{5}$$

$$w_j = \arg\min_k \frac{1}{z_{.k}} \sum_i z_{ik} (x_{ij} - \delta)^2. \tag{6}$$

The proposed algorithm, called Diagonal Double K-means (DDKM) is computationally efficient as its complexity can be shown to be $O(\tau \times npK)$, where τ denotes the number of iterations required for convergence, n, p and K are the number of objects (*i.e.*, rows), features (*i.e.*, columns) and co-clusters, respectively. The DDKM algorithm defines a sequence $(\mathbf{Z}^{(t)}, \mathbf{W}^{(t)})$ which monotonically decreases the criterion $J(\mathbf{X}, \mathbf{Z}, \mathbf{W})$ and is summarized in Algorithm 2.

Algorithm 2. Diagonal Double K-means (DDKM).

input: **X** and K

initialization: **Z**, **W** and δ

repeat

(1) Update **Z** according to Eq. (5)

(2) Update **W** according to Eq. (6)

until the J value change is small or there is no change

output: **Z** and **W**

3.2. Choice of δ

Herein, we discuss the choice of δ. Specifically, we compare two different settings that can be possibly used to define δ: (1) the value of δ obtained using the optimization procedure and (2) the value of δ set up manually to the maximum entry of the matrix.

1. If we consider δ as an unknown parameter, its optimal value for the criterion to be minimized is equal to the average of blocks means. Indeed, with **Z** and **W** fixed and by setting the derivative of J (Eq. (3)) to zero we obtain $\delta = \frac{1}{K} \sum_k \overline{x}_k$. Although this value of

δ is optimal, we can observe that in the context of sparse data, *i.e*, when the data matrix contains a high percentage of 0, its value will tend to 0 leading to a diagonal structure of blocks of 0's. An illustration of the resulting co-clustering solution with this value on the CSTR data set (described in the numerical experiments section) is given in Fig. 3(c).

2. Another way to proceed is to set the value of δ at the initialisation step. DDKM aims at grouping objects and features with the strongest association possible. For instance, in the case of a binary data matrix $\mathbf{X}$, the strongest association between an object i and a feature j is given by $x_{ij}=1$ which is the maximum possible value of $\mathbf{X}$. As a matter of fact, choosing the maximum allows to guarantee the homogeneity of diagonal blocks while ensuring blocks of 0 outside. In [37,40], the authors proposed hierarchical algorithms based on this idea. Fig. 3(b) presents the reorganized matrix obtained using this value of δ. From this Figure, we can see that the resulting matrix has a more clear diagonal structure than the one presented in Fig. 3(c) and thus it justifies our motivation to study this parameter setting.
   It is important to stress that this approach requires values of a data matrix to be comparable. This is the case for binary or normalized data, as we will see in the section devoted to the document-term partitioning.

4. Fuzzy block diagonal structure

The fuzziness principle allows a description of uncertainties that often appear in real world applications. For both previous algorithms, Double K-means and Diagonal Double K-means, we assume that objects and variables belong to one single block. However in some cases, different co-clusters may overlap and hence objects and variables may have multiple memberships. Fig. 4 illustrates a situation of a binary matrix where two co-clusters overlap at features 9–10 and objects 8–12. In this case an equal membership degree assignment to these objects and features instead of a hard assignment would be more relevant and accurate. In order to manage this type of situation we propose a fuzzy version of DDKM called Fuzzy Diagonal Double K-means.

4.1. Criterion and proposed algorithm

The Fuzzy Diagonal Double K-means (F-DDKM) algorithm optimizes an adequacy criterion J_F defined as

$$J_F=\sum_k \frac{1}{u_{.k}^{\alpha} v_{.k}^{\beta}} \sum_{i,j} (u_{ik})^{\alpha}(v_{jk})^{\beta}(x_{ij}-\delta)^2 \tag{7}$$

in which δ denotes the maximum value of the data matrix $\mathbf{X}$. $\mathbf{U}:=\{u_{ik}; i=1,\ldots,n; k=1,\ldots K\}$ where u_{ik} denotes the membership degree of object i in cluster k and $\mathbf{V}:=\{v_{jk}; j=1,\ldots,p; k=1,\ldots,K\}$ where v_{jk} denotes the membership degree of variable j in cluster k. The parameters $\alpha\in(1,\infty)$ and $\beta\in(1,\infty)$ control the fuzziness of membership for each object and each variable, respectively. The larger they are, the fuzzier the resulting co-clusters will be. The parameters $u_{.k}^{\alpha}$ and $v_{.k}^{\beta}$ denote $\sum_i (u_{ik})^{\alpha}$ and $\sum_j (v_{jk})^{\beta}$, respectively. Finally, the objective function is minimized subject to the following constraints:

$$\sum_k u_{ik}=1, \quad u_{ik}\in[0,1] \tag{8}$$

and

$$\sum_k v_{jk}=1, \quad v_{jk}\in[0,1]. \tag{9}$$

Following the relation established for the hard co-clustering context, we obtain

$$J_F(\mathbf{X},\mathbf{U},\mathbf{V})=\sum_k \hat{s}_k^2+\sum_k (\hat{x}_k-\delta)^2 \tag{10}$$

where

$$\hat{x}_k=\frac{1}{u_{.k}^{\alpha} v_{.k}^{\beta}}\sum_{i,j}(u_{ik})^{\alpha}(v_{jk})^{\beta}x_{ij} \quad \text{and} \quad \hat{s}_k^2=\frac{1}{u_{.k}^{\alpha} v_{.k}^{\beta}}\sum_{i,j}(u_{ik})^{\alpha}(v_{jk})^{\beta}(x_{ij}-\hat{x}_k)^2$$

are the mean and the variance of the k-th block. In the same manner, the first term of Eq. (10) ensures the homogeneity of each block while the second one provides the homogeneity between centers of the blocks and δ.

To minimize the criterion J_F defined in Eq. (7) with respect to the necessary optimality conditions given by Eqs. (8) and (9), we use the method of Lagrange multipliers and obtain the following Lagrangian function:

$$J_F=\sum_{i,j,k}\frac{1}{u_{.k}^{\alpha} v_{.k}^{\beta}}(u_{ik})^{\alpha}(v_{jk})^{\beta}(x_{ij}-\delta)^2-\sum_i \theta_i\left(\sum_k u_{ik}-1\right)-\sum_j \lambda_j\left(\sum_k v_{jk}-1\right)$$

where θ_i and λ_j are the Lagrange multipliers. Calculating the partial derivative of J_F and setting them to zero, gives the following formulae:

$$u_{ik}=\left[\sum_{r=1}^{K}(D_{ik}/D_{ir})^{\frac{1}{\alpha-1}}\right]^{-1} \quad \forall i,k, \tag{11}$$

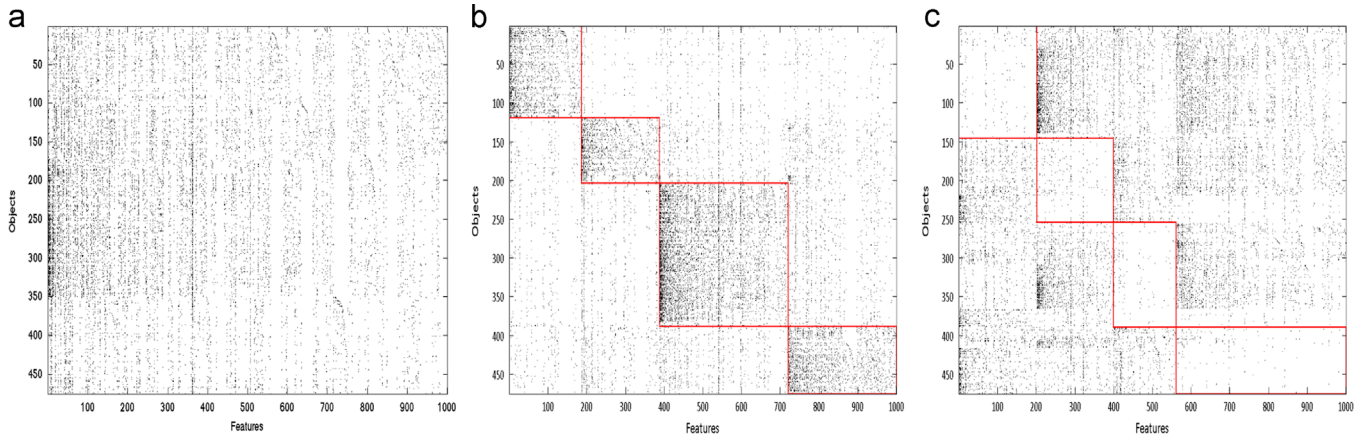

Fig. 3. (a) CSTR the original data set, (b) CSTR reorganised according to the partitions when $\delta=\max_{i,j}x_{ij}$, (c) CSTR reorganised according to the partitions when δ is estimated by $\frac{1}{K}\sum_k \overline{x}_k$.

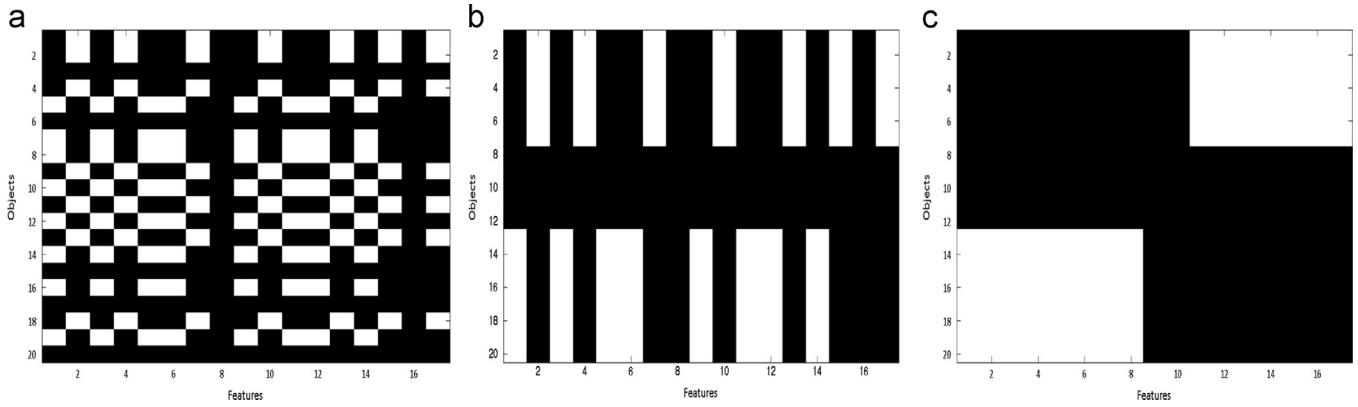

Fig. 4. Illustration of two overlapping co-clusters: (a) original binary data, (b) data reorganized according to the partition of rows, and (c) data reorganized according to row and column clusters with overlapping.

where $D_{ik} = \frac{1}{v_k^{\beta}} \sum_j (v_{jk})^{\beta} (x_{ij} - \delta)^2$ and

$$v_{jk} = \left[\sum_{r=1}^{K} (D'_{ik}/D'_{ir})^{\frac{1}{\beta-1}} \right]^{-1} \quad \forall j, k, \tag{12}$$

where $D'_{jk} = \frac{1}{u_k^{\alpha}} \sum_i (u_{ik})^{\alpha} (x_{ij} - \delta)^2$. The optimization of $J_F(\mathbf{X}, \mathbf{U}, \mathbf{V})$ can then be performed by optimizing two conditional criteria $J_F(\mathbf{X}, \mathbf{U}|\mathbf{V})$ and $J_F(\mathbf{X}, \mathbf{V}|\mathbf{U})$ given $\mathbf{V}$ and $\mathbf{U}$ respectively. Thus, F-DDKM defines a sequence $(\mathbf{U}^{(t)}, \mathbf{V}^{(t)})$ which monotonically decreases alternatively $J_F(\mathbf{X}, \mathbf{U}|\mathbf{V})$ and $J_F(\mathbf{X}, \mathbf{V}|\mathbf{U})$ and therefore the update of $\mathbf{U}$ and $\mathbf{V}$ defined in Eqs. (11) and (12) causes the objective function J_F to be non-increasing. We have

$$J_F(\mathbf{X}, \mathbf{U}^{(t)}, \mathbf{V}^{(t)}) \geq J_F(\mathbf{X}, \mathbf{U}^{(t+1)}, \mathbf{V}^{(t)}) \geq J_F(X, \mathbf{U}^{(t+1)}, \mathbf{V}^{(t+1)}).$$

The sequence $(\mathbf{U}^{(t)}, \mathbf{V}^{(t)})$ reaches its stationary value at iteration T. Due to the uniqueness of $\mathbf{U}$ and $\mathbf{V}$, at T we have

$$J_F(\mathbf{X}, \mathbf{U}^{(T)}, \mathbf{V}^{(T)}) = J_F(\mathbf{X}, \mathbf{U}^{(T+1)}, \mathbf{V}^{(T)}) = J_F(X, \mathbf{U}^{(T+1)}, \mathbf{V}^{(T+1)}).$$

The criterion $J_F(\mathbf{X}, \mathbf{U}, \mathbf{V})$ converges to a stationary point and is bounded because $J_F(\mathbf{X}, \mathbf{U}, \mathbf{V}) \geq 0$. The exactly same reasoning is valid for the convergence of DDKM.

F-DDKM is also computationally efficient and its complexity can be easily shown to be $O\left(\tau \times \text{npK}^2\right)$. The principal steps of F-DDKM are presented in Algorithm 3 and illustrated on a toy example in the next subsection.

Algorithm 3. Fuzzy Diagonal Double K-means (F-DDKM).

input: $\mathbf{X}$ and K
initialization: $\mathbf{U}$, $\mathbf{V}$ and δ
repeat
(1) Update $\mathbf{U}$ according to Eq. (11).
(2) Update $\mathbf{V}$ according to Eq. (12).
until the J_F value change is small or there is no change
output: $\mathbf{U}$ and $\mathbf{V}$

4.2. Illustration of F-DDKM

In this subsection, we aim to illustrate the behaviour of F-DDKM on a simple example. Let us consider a binary data matrix $\mathbf{X}$ of size 16×14 presented in Table 1. The number of clusters in this matrix is equal to 2.

In the sequel, we describe the main steps of F-DDKM allowing to obtain the partitioning of $\mathbf{X}$. Tables 2 and 3 show the object and the feature partitions obtained from the initialization to the convergence of F-DDKM.

Input: Let K be the real number of row and column clusters, *i.e.* 2.

Table 1
Binary data matrix $\mathbf{X}$.

Objects	Features													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	1	1	1	0	1	1	1	1	0	0	0	0	0	0
B	0	1	1	1	1	1	1	1	1	0	0	0	0	0
C	1	0	1	1	1	1	1	1	0	0	0	0	0	0
D	0	0	1	1	1	0	1	0	0	0	0	0	0	0
E	0	0	1	0	1	1	0	1	0	0	0	0	0	0
F	0	0	0	0	1	1	1	1	0	0	0	0	0	0
G	0	0	0	0	0	1	0	1	1	0	0	0	0	0
H	0	0	0	0	1	0	1	1	1	0	0	0	0	0
I	0	0	0	0	0	0	1	1	1	0	0	1	0	0
J	0	0	0	0	0	0	0	1	1	1	0	1	0	0
K	0	0	0	0	0	0	0	1	1	1	0	1	1	1
L	0	0	0	0	0	0	1	1	1	1	0	1	1	1
M	0	0	0	0	0	1	1	0	1	1	1	1	1	1
N	0	0	0	0	0	0	1	1	0	1	1	1	0	0
O	0	0	0	0	0	0	0	1	1	0	0	0	0	0
P	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Table 2
Step by step partitions of objects into two clusters obtained with F-DDKM.

Objects	$U^{(0)}$		$U^{(1)}$		...	$U^{(5)}$	
	1	2	1	2		1	2
A	0.41	0.59	0.90	0.10	...	1.00	0.00
B	0.37	0.63	0.10	0.90	...	1.00	0.00
C	0.32	0.68	0.37	0.63	...	1.00	0.00
D	0.73	0.27	0.96	0.04	...	1.00	0.00
E	0.65	0.35	0.85	0.15	...	1.00	0.00
F	0.36	0.64	0.99	0.01	...	1.00	0.00
G	0.59	0.41	0.06	0.94	...	0.89	0.11
H	0.43	0.57	0.96	0.04	...	1.00	0.00
I	0.27	0.73	0.94	0.06	...	0.08	0.92
J	0.19	0.81	0.13	0.87	...	0.00	1.00
K	0.51	0.49	0.04	0.96	...	0.00	1.00
L	0.79	0.21	0.88	0.12	...	0.00	1.00
M	0.51	0.49	1.00	0.00	...	0.00	1.00
N	0.74	0.26	1.00	0.00	...	0.00	1.00
O	0.50	0.50	0.09	0.91	...	0.27	0.73
P	0.27	0.73	0.78	0.22	...	0.00	1.00

Initialization: We randomly initialize the membership matrices $\mathbf{U}$ and $\mathbf{V}$ with respect to the constraints given by Eqs. (8) and (9). We also set δ to 1 (the maximum).

Step 1: $\mathbf{V}^{(0)}$ and δ are fixed. We update the membership matrix of objects $\mathbf{U}$ following Eq. (11).

Table 3
Step by step partitions of features into two clusters obtained with F-DDKM.

Features	$V^{(0)}$		$V^{(1)}$		...	$V^{(5)}$	
	1	2	1	2		1	2
1	0.1489	0.85	0.53	0.47	...	1.00	0.00
2	0.30	0.70	0.16	0.84	...	1.00	0.00
3	0.54	0.46	0.59	0.41	...	1.00	0.00
4	0.07	0.93	0.04	0.96	...	1.00	0.00
5	0.72	0.28	1.00	0.00	...	1.00	0.00
6	0.43	0.58	0.22	0.78	...	1.00	0.00
7	0.93	0.07	1.00	0.00	...	1.00	0.00
8	0.40	0.60	0.00	1.00	...	1.00	0.00
9	0.21	0.79	0.00	1.00	...	0.00	1.00
10	0.37	0.63	0.32	0.68	...	0.00	1.00
11	0.91	0.09	1.00	0.00	...	0.00	1.00
12	0.68	0.32	0.87	0.13	...	0.00	1.00
13	0.61	0.39	0.55	0.45	...	0.00	1.00
14	0.25	0.75	0.55	0.45	...	0.00	1.00

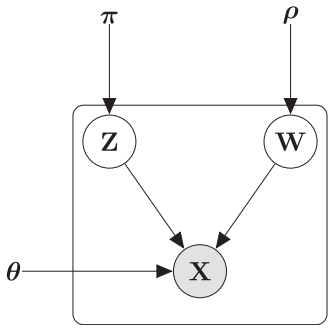

Fig. 5. Latent block model as a graphical model.

Step 2: $\mathbf{U}^{(0)}$ and δ are fixed. We update the membership matrix of features $\mathbf{V}$ (Eq. (12)).

Finally, we compute the value of the criterion J_F (see Eq. (7)). After the first iteration, its value is 1.18. We repeat steps 1 and 2 until convergence that is achieved after 5 iterations. The values of the criterion at each iteration are 1.18,1.03,0.96,0.84,0.84. From matrices $\mathbf{U}^{(5)}$ and $\mathbf{V}^{(5)}$, we can deduce $\mathbf{Z}$ and $\mathbf{W}$, the hard partition of objects and variables, respectively. This is achieved by using the maximum a posteriori (MAP) principle. Therefore we obtain the following partitions: $\mathbf{Z}=(1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2)^T$ and $\mathbf{W}=(1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2)^T$.

5. Evaluation of DKKM and F-DDKM on synthetic data sets

Before evaluating our two algorithms in the document-term partitioning context, we propose to study their behaviour on binary and continuous simulated data. Our motivation is two-fold: first, both types of data can be fully considered by our hard and fuzzy approaches, secondly in document clustering these two types are commonly handled.

5.1. Simulation process

We use the latent block model proposed in [41,21] to simulate binary and continuous data (for details see, for instance, [39]). Denoting by $\mathcal{Z}$ and $\mathcal{W}$ the sets of possible labels $\mathbf{Z}$ and $\mathbf{W}$, in this model, we assume that for each co-cluster $\mathbf{X}_{kh}$ the values x_{ij} are distributed according to the following probability density function:

$$f(\mathbf{X};\boldsymbol{\Theta})=\sum_{(\mathbf{Z},\mathbf{W})\in\mathcal{Z}\times\mathcal{W}}\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,h}\rho_h^{w_{jh}}\prod_{i,j,k,h}\varphi(x_{ij};\theta_{kh}). \tag{13}$$

This model can be represented by a graphical model depicted in Fig. 5.

In this article, we set the number of rows cluster and the number of columns cluster to the same value, *i.e.*, $K=H$. For continuous data, the values x_{ij} are distributed according to a Gaussian distribution $\mathcal{N}(\mu_{kh},\sigma^2_{k\ell})$ with $\mu_{kh}\in\mathbb{R}$, $\sigma^2_{kh}\in\mathbb{R}^+$, and $\varphi(x_{ij};\theta_{kh})=\frac{1}{\sqrt{2\pi\sigma^2_{kh}}}\exp-\left\{\frac{(x_{ij}-\mu_{kh})^2}{2\sigma^2_{kh}}\right\}^{z_{ik}w_{jh}}$. This model is then parametrised by $\boldsymbol{\Theta}=(\boldsymbol{\pi},\boldsymbol{\rho},\boldsymbol{\theta})$, where $\boldsymbol{\pi}=(\pi_1,\ldots,\pi_K)$, $\boldsymbol{\rho}=(\rho_1,\ldots,\rho_K)$ and $\boldsymbol{\theta}=(\theta_{11}=(\mu_{11},\sigma^2_{11}),\ldots,\theta_{kh}=(\mu_{kh},\sigma^2_{kh}),\ldots,\theta_{KK}=(\mu_{KK},\sigma^2_{KK}))$.

For binary data sets, the values x_{ij} are distributed according to a Bernoulli distribution $\mathcal{B}(\gamma)$ with $\gamma\in[0;1]$ and $\varphi(x_{ij};\theta_{kh})=\left[\gamma_{kh}^{x_{ij}}(1-\gamma_{kh})^{(1-x_{ij})}\right]^{z_{ik}w_{jh}}$. This model is then parametrised by $\boldsymbol{\Theta}=(\boldsymbol{\pi},\boldsymbol{\rho},\boldsymbol{\theta})$ where $\boldsymbol{\pi}=(\pi_1,\ldots,\pi_K)$, $\boldsymbol{\rho}=(\rho_1,\ldots,\rho_K)$ and $\boldsymbol{\theta}=(\theta_{11}=\gamma_{11},\ldots,\theta_{\mathrm{KK}}=\gamma_{\mathrm{KK}})$.

Algorithm 4 presents in details, the simulation process for both types of data.

Algorithm 4. Simulation of data.

input: n, p, K.
(1) Simulate $\mathbf{Z}$ according to a Multinomial distribution with parameters $(1,\pi_1,\ldots,\pi_K)$.
(2) Simulate $\mathbf{W}$ according to a Multinomial distribution with parameters $(1,\rho_1,\ldots,\rho_K)$.
(3) Simulate each co-cluster $\mathbf{X}_{\mathrm{kh}}$ according to Gaussian density with (μ_{kh},σ^2_{kh}) for continuous data and Bernoulli density with γ_{kh} for binary data.
output: data matrix $\mathbf{X}$ of size $(n\times p)$

5.2. Performance evaluation

In order to assess and to compare the performance of the proposed algorithms, we use three commonly adopted metrics including accuracy, the Normalized Mutual Information [42] and the Adjusted Rand Index [43]. We focus only on the quality of row clustering. Clustering accuracy noted Acc is one of the most widely used evaluation criterion and is defined as

$$\mathrm{Acc}=\frac{1}{n}\max\left[\sum_{\mathcal{C}_k,\mathcal{L}_\ell}T(\mathcal{C}_k,\mathcal{L}_\ell)\right],$$

where $\mathcal{C}_k$ is the k-th cluster in the final result, and $\mathcal{L}_\ell$ is the true ℓ-th class. $T(\mathcal{C}_k,\mathcal{L}_\ell)$ is the proportion of objects that were correctly recovered by the clustering algorithm, *i.e.*, $T(\mathcal{C}_k,\mathcal{L}_\ell)=\mathcal{C}_k\cap\mathcal{L}_\ell$. Accuracy computes the maximum sum of $T(\mathcal{C}_k,\mathcal{L}_\ell)$ for all pairs of clusters and classes that do not overlap.

The second measure used is the Normalized Mutual Information (NMI) calculated as follows:

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

where n_k denotes the number of data contained in the cluster $\mathcal{C}_k(1\le k\le K)$, $\hat{n}_\ell$ is the number of data belonging to the class $\mathcal{C}'_k(1\le\ell\le K)$ and $n_{k\ell}$ denotes the number of data that are in the intersection between the cluster $\mathcal{C}_k$ and the class $\mathcal{C}'_k$.

The last measure, Adjusted Rand, noted ARI, measures the similarity between two clustering partitions. From a mathematical standpoint, the Rand index is related to the accuracy. The adjusted

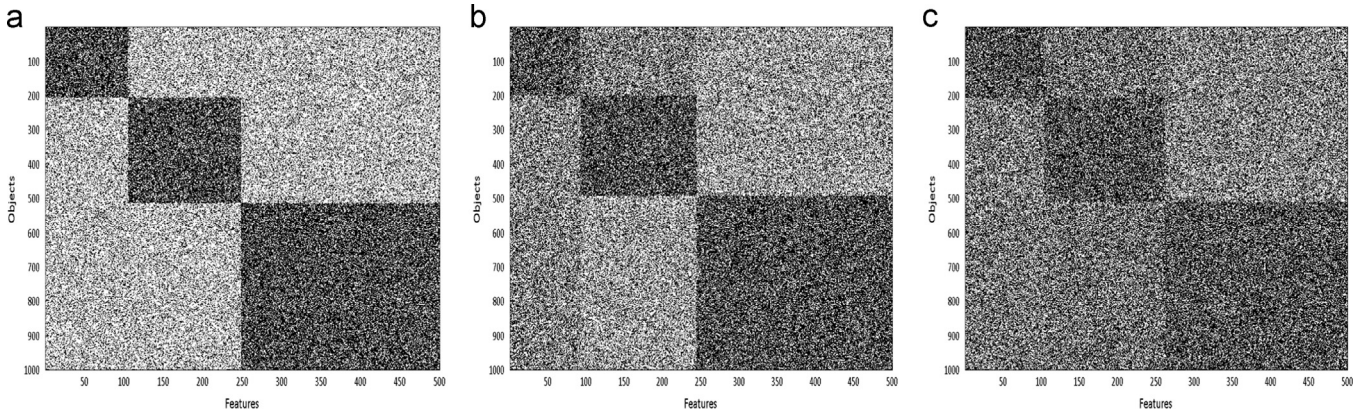

Fig. 6. Visualisation of binary reorganized data matrix with three degree of overlap: (a) $+$, (b) $++$, (c) $+++$. Dark points represent 1's and white point 0's.

Table 4
Means of Acc, NMI and ARI ($\pm$ standard errors), in %, computed on 100 simulated samples.

Size	Degree of overlap	Metric	Algorithms			
			Binary		Continuous	
			DDKM	F-DDKM	DDKM	F-DDKM
(1000, 500)		Acc	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.01
	$+$	NMI	0.96 ± 0.02	0.96 ± 0.02	0.94 ± 0.01	0.95 ± 0.02
		ARI	0.98 ± 0.01	0.98 ± 0.01	0.96 ± 0.01	0.97 ± 0.02
(1000, 500)		Acc	0.92 ± 0.00	0.92 ± 0.00	0.88 ± 0.01	$\mathbf{0.90 \pm 0.02}$
	$++$	NMI	0.82 ± 0.02	0.82 ± 0.02	0.85 ± 0.03	0.86 ± 0.03
		ARI	0.89 ± 0.02	0.89 ± 0.02	0.86 ± 0.02	$\mathbf{0.89 \pm 0.02}$
(1000, 500)		Acc	0.71 ± 0.02	$\mathbf{0.77 \pm 0.01}$	0.72 ± 0.06	$\mathbf{0.81 \pm 0.04}$
	$+++$	NMI	0.62 ± 0.03	$\mathbf{0.69 \pm 0.04}$	0.57 ± 0.06	$\mathbf{0.78 \pm 0.06}$
		ARI	0.64 ± 0.02	$\mathbf{0.73 \pm 0.04}$	0.61 ± 0.05	$\mathbf{0.79 \pm 0.06}$
(1000, 1000)		Acc	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00
	$+$	NMI	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.01	0.99 ± 0.00
		ARI	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00	0.99 ± 0.00
(1000, 1000)		Acc	0.97 ± 0.01	0.97 ± 0.02	0.90 ± 0.01	$\mathbf{0.93 \pm 0.02}$
	$++$	NMI	0.94 ± 0.01	0.94 ± 0.02	0.87 ± 0.03	$\mathbf{0.91 \pm 0.04}$
		ARI	0.96 ± 0.01	0.97 ± 0.02	0.89 ± 0.03	$\mathbf{0.92 \pm 0.04}$
(1000, 1000)		Acc	0.74 ± 0.03	$\mathbf{0.82 \pm 0.04}$	0.74 ± 0.03	$\mathbf{0.83 \pm 0.05}$
	$+++$	NMI	0.69 ± 0.02	$\mathbf{0.78 \pm 0.04}$	0.70 ± 0.03	$\mathbf{0.79 \pm 0.05}$
		ARI	0.73 ± 0.02	$\mathbf{0.79 \pm 0.03}$	0.72 ± 0.04	$\mathbf{0.81 \pm 0.06}$

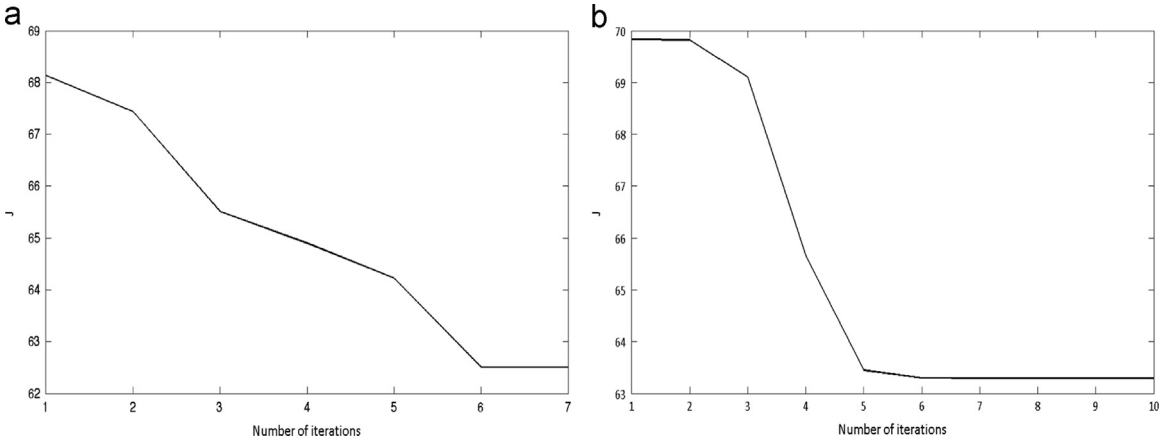

Fig. 7. Convergence of (a) DDKM and (b) F-DDKM on a continuous data set of size 1000×500 with a medium degree of overlap ($++$).

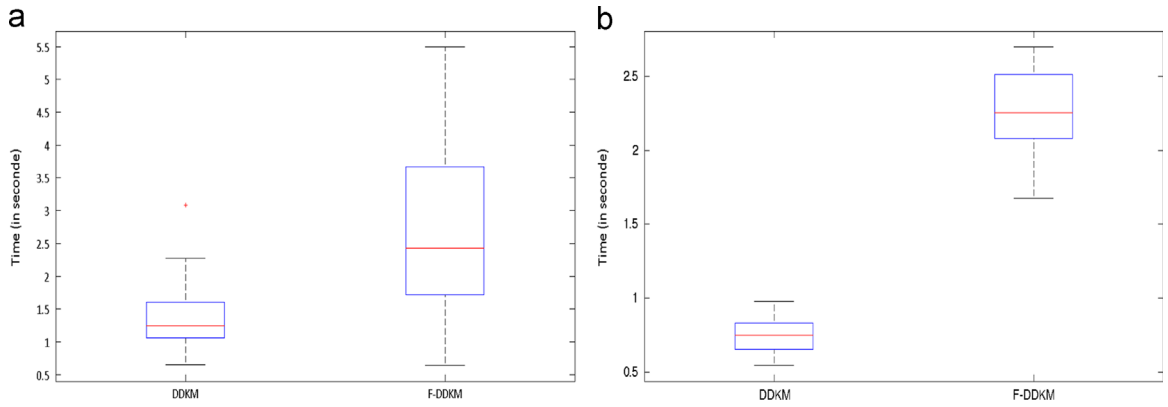

Fig. 8. Running time (in seconds) of algorithms on (a) binary and (b) continuous data sets of size 1000×1000 with a high degree of overlap ($+++$).

Table 5
Means of the number of iterations ($\pm$ standard errors) computed on 100 simulated samples.

Size	Degree of overlap	Algorithms			
		Binary		Continuous	
		DDKM	F-DDKM	DDKM	F-DDKM
(1000, 500)	+	7.78 ± 2.45	8.28 ± 4.34	4.59 ± 0.79	6.77 ± 0.74
(1000, 500)	++	16.44 ± 4.24	22.68 ± 9.84	8.46 ± 2.39	13.26 ± 3.94
(1000, 500)	+++	27.12 ± 8.59	32.80 ± 10.81	14.14 ± 5.36	20.7 ± 15.41
(1000, 1000)	+	6.04 ± 1.03	6.30 ± 1.54	4.62 ± 1.78	6.82 ± 0.70
(1000, 1000)	++	15.88 ± 4.68	20.46 ± 6.77	6.32 ± 1.17	13.48 ± 1.87
(1000, 1000)	+++	19.56 ± 8.00	35.76 ± 15.57	14.58 ± 6.05	14.88 ± 13.16

Table 6
Description of the data sets in terms of size ($n \times p$), number of clusters (K), sparsity (%0) and degree of balance of the clusters.

Data sets	$n \times p$	K	%0	Balance
CSTR	475×1000	4	96.60	0.399
Classic3	3891×4303	3	98.95	0.71
Classic4	7095×5896	4	99.41	0.323
WebKB4	4199×1000	4	91.83	0.307
Reviews	$4069 \times 18\,483$	5	98.99	0.099
Sports	$8580 \times 14\,870$	7	99.04	0.036

Table 7
Description of the NG20 data set in term of size ($n \times p$), number of clusters (K), sparsity (%0), degree of balance of the clusters and subjects included.

Data sets	$n \times p$	K	%0	Balance	Newsgroups included
NG2	500×2000	2	96.90	1	alt.atheism, comp.graphics
NG5	500×2000	5	97.19	1	comp.os.ms-windows, comp. windows.x, rec.motorcycles, sci.crypt,sci. space
NG10	500×2000	10	96.44	1	comp.graphics, comp.sys.ibm.pc. hardware,rec.autos rec.sport.baseball, sci.crypt, sci. med, talk.religion.misc comp.windows.x, soc.religion. christian, talk.politics.mideast

form of the Rand Index is defined as

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

The values for these three metrics are between 0 and 1, a value close to 1 means a good result in terms of clustering. These metrics are expressed in percentage in the following sections.

5.3. Synthetic data sets and assessment of DDKM and F-DDKM

We use synthetic data in order to evaluate our methods DDKM and F-DDKM. We simulate continuous and binary data sets by varying size, degree of block overlap and assuming unequal proportions of $K=3$ co-clusters with $(\pi_1, \pi_2, \pi_3) = (0.2, 0.3, 0.5)$ and $(\rho_1, \rho_2, \rho_3) = (0.2, 0.3, 0.5)$. Regarding the size, we consider $n \times p = 1000 \times 500$ and 1000×1000. Fig. 6 shows data matrices with 500 features, 1000 objects, unequal proportions and three different degrees of overlap that we obtain by varying the parameter $\boldsymbol{\theta}$.

For each data structure, we generate 100 samples and for each sample we run the algorithms 100 times. In Table 4, the averages and standard deviations of Acc's, NMI's and ARI's for each situation are reported. To highlight significant differences in terms of performance, results for which Student's t-test returns a p-value lower than 0.05 are in bold. Fig. 7 shows the convergence of the criterion for both algorithms.

On continuous data sets with a low degree of overlap (+) both algorithms perform equally well in terms of Acc's, NMI's and ARI's. On data sets with a high degree of overlap (+++) F-DDKM outperforms

Table 8
Accuracy, Normalized Mutual Information and Adjusted Rand Index (in %) obtained on binary data sets. (-) denotes that the algorithm cannot propose a partition with a required number of co-clusters.

Data set	Metric	Algorithms								
		NMF	K-means	SpKM	DKM	SpCo	ITCC	Block	DDKM	F-DDKM
CSTR	Acc	85.68	85.05	88.63	62.95	79.79	69.94	81.32	91.37	**92.00**
	NMI	67.08	64.74	74.07	27.93	66.67	70.00	63.25	79.06	**79.22**
	ARI	70.65	68.14	76.57	46.72	70.20	65.79	67.33	83.00	**84.15**
Classic3	Acc	97.66	90.47	97.33	94.17	70.60	**98.46**	84.21	98.20	98.36
	NMI	88.78	73.81	91.41	80.90	59.64	**92.32**	54.36	91.31	91.72
	ARI	93.06	75.34	94.38	82.82	40.20	**95.42**	58.77	94.61	95.12
Classic4	Acc	82.49	74.88	77.42	74.40	–	64.87	52.51	84.69	**85.54**
	NMI	60.21	51.46	58.35	48.38	–	50.96	16.35	**65.22**	65.17
	ARI	57.25	43.28	50.02	40.73	–	42.56	17.01	61.09	**62.84**
WebKB4	Acc	73.26	49.46	62.85	38.75	64.32	64.85	62.18	72.80	**76.16**
	NMI	41.05	25.15	37.00	04.65	41.04	39.73	36.65	**44.14**	43.53
	ARI	43.60	18.03	31.55	04.54	38.09	37.07	34.95	44.29	**49.80**
Reviews	Acc	69.99	53.58	68.99	33.84	57.13	57.97	53.29	**71.02**	69.41
	NMI	50.09	42.76	52.20	5.75	41.57	45.71	43.57	51.03	**53.71**
	ARI	50.40	29.81	53.20	2.36	32.18	34.51	41.43	51.46	**53.73**
Sports	Acc	56.97	41.81	60.26	30.56	55.73	44.73	47.08	**76.69**	72.11
	NMI	51.47	33.21	58.85	5.49	47.18	49.16	38.13	**61.39**	60.09
	ARI	36.94	13.67	43.79	2.18	34.45	18.67	19.62	**64.18**	58.56
NG2	Acc	60.60	57.60	60.80	54.60	90.20	90.80	72.01	94.60	**95.40**
	NMI	12.85	12.93	13.09	2.38	55.35	56.57	20.35	70.55	**73.57**
	ARI	4.41	2.26	4.58	0.77	64.57	66.52	20.55	79.53	**82.41**
NG5	Acc	50.10	33.07	52.10	29.66	60.32	54.31	53.21	76.75	**84.87**
	NMI	32.25	16.61	28.31	8.15	50.75	36.74	34.15	51.49	**63.80**
	ARI	20.68	4.08	24.84	2.82	37.31	31.45	29.96	51.37	**65.84**
NG10	Acc	47.90	39.08	43.29	40.08	48.90	46.49	42.68	44.08	**50.30**
	NMI	47.36	30.41	39.72	37.26	**51.87**	41.45	41.21	31.00	39.60
	ARI	24.46	19.47	23.52	21.15	**32.07**	30.25	32.14	20.59	24.80

DDKM but suffers from slow convergence (up to three times slower), as illustrated in Fig. 8 and Table 5). It is worth noting that both algorithms perform better as the number of features increases.

On binary data sets with low and medium degrees of overlap both algorithms perform equally well. Regarding the performance on data sets of increasing size, we observe the same behaviour than on continuous data sets. The difference in terms of performance between the two algorithms on data sets with a high degree of overlap is less important than on continuous data sets.

On both types of data sets DDKM is faster than F-DDKM. It is also more stable which is reflected by lower values of standard deviation for the performance, the number of iterations and the time required for convergence. Although F-DDKM requires more time and more iterations to converge, the performances recorded on on both continuous and binary data sets with a high degree of overlap, make it more attractive.

6. Document-term partitioning

We study the effectiveness of our algorithms for some well-known text data sets with different sizes and balances (the balance coefficient is defined as the ratio of the number of documents in the smallest class to the number of documents in the largest class).

6.1. Data sets

We compare the clustering performance of our algorithms DDKM and F-DDKM with state-of-the-art (co)-clustering algorithms commonly used in the context of document-term clustering. Hereafter, we give a detailed description of the data sets we use (see also Tables 6 and 7).

Hereafter, we give a detailed description of chosen data sets.

- **Classic4** consists of 4 different document collections: MED, CISI, CRAN and CACM. We also use a subset of this data set (CISI, CRAN and MED only) referred to as Classic3, in the sequel. Terms which appear in less than 3 documents, or in more than 95% of the documents were removed. Moreover, Porters stemming was applied as a pre-processing step.
- **CSTR** contains abstracts of technical reports published in the department of Computer Science at a research university. These abstracts were divided into four research areas: Natural Language Processing (NLP), Robotics/Vision, Systems and Theory.
- **WebKB** consists of seven classes of web pages collected from computer science departments: student, faculty, course, project, department, staff and other. Frequently, only four classes are used (student, faculty, course, project); this subset is called WebKB4.
- **Reviews** and **Sports** are two *document* × *term* matrices from the software *CLUTO*.[2] They were derived from the San Jose Mercury newspaper articles. Reviews contain documents on such topics as food, movies, music, radio and restaurants; Sports contains articles about baseball, basketball, bicycling, boxing, football, golfing and hockey.
- **20 Newsgroups** is a set of Usenet articles organized into 20 topics. We use two subsets of NG20 including both topics closely related or not related (Table 7).

The characteristics of all data sets used are presented in Tables 6 and 7. Originally each cell of these data sets denotes the

[2] http://www.cs.umn.edu/~cluto.

Table 9
Accuracy, Normalized Mutual Information and Adjusted Rand Index (in %) obtained on TF-IDF data sets. (–) denotes that the algorithm cannot propose a partition with a required number of co-clusters.

Data set	Metric	Algorithms							
		NMF	K-means	SpKM	DKM	SpCo	ITCC	DDKM	F-DDKM
CSTR	Acc	81.47	75.58	89.47	57.26	83.16	81.05	90.95	**92.00**
	NMI	69.91	51.04	74.95	41.38	71.71	65.71	76.39	**79.13**
	ARI	70.26	47.53	79.38	30.07	72.45	66.30	82.99	**84.20**
Classic3	Acc	96.32	96.83	97.83	96.02	97.87	**98.79**	**98.79**	98.12
	NMI	84.53	92.22	91.83	85.67	91.49	93.95	**94.24**	90.56
	ARI	89.04	93.59	94.18	88.67	93.89	96.42	**96.83**	94.47
Classic4	Acc	53.21	76.84	64.34	87.57	–	60.37	**87.85**	76.43
	NMI	39.40	55.02	62.64	68.53	–	55.12	**69.25**	48.85
	ARI	24.16	45.93	49.75	**68.21**	–	49.17	67.58	44.01
WebKB4	Acc	**80.38**	57.11	80.09	38.08	61.99	61.01	78.90	78.07
	NMI	**52.08**	29.97	51.83	04.85	48.64	46.21	51.15	50.15
	ARI	**56.48**	22.20	55.81	04.36	41.94	41.27	54.93	53.02
Reviews	Acc	67.71	67.42	**74.69**	40.82	52.67	62.97	64.19	69.97
	NMI	43.74	44.74	**63.15**	29.07	31.15	51.31	50.33	42.07
	ARI	38.20	39.91	**67.89**	16.18	19.67	47.06	47.58	46.40
Sports	Acc	55.28	48.72	**73.37**	47.88	55.18	54.11	71.45	62.79
	NMI	35.71	43.22	**66.61**	31.59	36.13	58.12	56.81	40.93
	ARI	33.95	22.51	**66.49**	17.83	26.18	39.61	56.85	49.31
NG2	Acc	88.40	-	76.18	69.60	88.98	90.35	91.80	**94.20**
	NMI	23.04	-	30.48	19.15	53.16	54.61	59.33	**68.17**
	ARI	29.89	-	34.23	15.24	60.69	62.36	69.83	**78.10**
NG5	Acc	44.49	44.98	79.96	44.29	53.91	53.69	55.71	**87.17**
	NMI	24.52	24.53	54.76	17.87	45.59	55.86	26.46	**68.17**
	ARI	20.49	17.52	56.06	12.16	30.03	30.09	23.98	**70.43**
NG10	Acc	42.89	31.86	43.69	42.89	47.09	48.08	42.08	**64.13**
	NMI	41.97	24.55	35.55	33.56	49.33	49.60	33.19	**53.30**
	ARI	19.08	14.20	26.64	22.33	27.63	31.90	24.84	**43.10**

number of occurrences of a term in a document. We conduct two types of comparisons on *document* × *term* matrices. The first, on the original data sets after applying a TF-IDF transformation. As mentioned in Section 3, the fact that we use to set δ to the maximum value of the data matrix, required a normalization of the data. We use the TF-IDF weighting scheme implemented in scikit-learn [44] which is defined as follows:

$$w_{ij} = tf_{ij}\left(1+\log\left(\frac{1+n}{1+d_j}\right)\right),$$

where w_{ij} is the weight of term i in document j, tf_{ij} is the frequency of term i in document j, n is the total number of documents and d_j is the number of documents containing term j. In order to eliminate the bias induced by the length of a document, the ℓ_2 normalization is applied to the input matrices before launching the co-clustering algorithm.

We also used a second version where the original data were converted into binary, *i.e.*, each cell having a value higher than 0 is considered equal to 1 and 0 otherwise.

6.2. Compared algorithms

We compare our method to some state-of-the-art (co)-clustering methods including Spherical K-means (SpKM) [45], Double K-means (DKM), Spectral Co-Clustering (SpCo) [15], ITCC [24] and Block [23]. We also report the clustering results obtained using K-means and Nonnegative Matrix Factorization (NMF) [46] as baselines. The Spherical K-means algorithm is basically a K-means algorithm that uses the cosine dissimilarity instead of the Euclidean distance. It is known to be very efficient on sparse data and to converge quickly. We use the matlab implementation for K-means and NMF. For SpCo algorithm we use the implementation proposed by Assaf Gottlieb.[3] We use the SpKM implementation given in [47]. For ITCC, we use the Matlab Toolbox for Biclustering Analysis (MTBA) [48]. Finally, we implement Block and CROEUC [3,39], a fast version of Double K-means (DKM) that consists in working on intermediate reduced matrices.

6.3. Results

We set the number of clusters to the true number of classes for all data sets. Because F-DDKM has two parameters to be tuned , we run this algorithm under different parameter settings and select the best result according to the criterion. We set $\alpha=\beta$ and assess F-DDKM according to different values of these parameters. We discuss their possible choices in the next subsection. We run all algorithms 100 times and report the best result, *i.e.*, the one with the minimum criterion value of all trials in Tables 8 and 9. We also perform t-tests comparing the best state-of-the-art (co)-clustering algorithm with the best of our algorithms for each data set and report the results in Table 10. Several observations can be made based on these results:

- On binary versions of data sets, F-DDKM significantly outperforms all other methods but is only slightly better than DDKM for data sets with no overlapping clusters (CSTR, Classic3, Classic4, WebKB4 and NG2). On NG5 and NG10, whose classes are not well separated, the difference is all the more important.
- On TF-IDF versions of data sets, the same comment can be made for NG5 and NG10. DDKM outperforms other methods on Classic4 but give comparable results with ITCC on Classic3. On

[3] http://adios.tau.ac.il/.

Table 10
T-Test results comparing the best state-of-the-art (co-) clustering algorithms with the best of our algorithms. For instance, on the binary version of CSTR, we compare SpKM with F-DDKM while on the TF-IDF version of Classic4 we compare DKM with DDKM. Boldface type indicates a significant difference (p-value below the 0.05 threshold).

Data set	Metric	Binary		TF-IDF	
		Best	(F)-DDKM	Best	(F)-DDKM
CSTR	Acc	80.64 ± 5.19	**90.23 ± 0.72***	86.74 ± 2.88	**87.74 ± 1.64***
	NMI	72.04 ± 3.11	**76.40 ± 1.24***	70.16 ± 2.76	**72.25 ± 2.77***
	ARI	69.65 ± 4.15	**81.36 ± 1.02***	73.33 ± 3.82	**76.08 ± 3.26***
Classic3	Acc	98.27 ± 0.00	98.11 ± 0.00	98.79 ± 0.00	98.79 ± 0.00
	NMI	91.64 ± 0.21	90.81 ± 0.22	93.25 ± 0.11	93.69 ± 0.19
	ARI	95.02 ± 0.18	94.36 ± 0.15	96.12 ± 0.25	96.47 ± 0.12
Classic4	Acc	80.91 ± 4.81	**82.07 ± 2.29**	85.46 ± 2.35	**87.77 ± 0.37**
	NMI	58.24 ± 5.09	**61.83 ± 2.02***	65.09 ± 4.01	**68.80 ± 0.52**
	ARI	56.06 ± 5.84	**59.38 ± 4.24***	63.66 ± 5.20	**68.63 ± 0.84**
WebKB4	Acc	69.02 ± 1.86	**70.01 ± 2.35***	**79.76 ± 0.21***	76.65 ± 0.92
	NMI	36.83 ± 1.57	**38.06 ± 2.78***	**51.18 ± 0.44***	48.69 ± 0.88
	ARI	38.10 ± 2.55	**39.49 ± 3.14***	**55.28 ± 0.44***	50.57 ± 1.46
Reviews	Acc	66.22 ± 1.94	**67.03 ± 2.64***	**72.57 ± 2.14***	62.83 ± 0.62
	NMI	49.04 ± 0.55	**49.65 ± 1.24**	**59.55 ± 4.84***	49.33 ± 1.07
	ARI	47.81 ± 1.54	**48.87 ± 2.64***	**61.74 ± 6.64***	44.30 ± 1.59
Sports	Acc	55.86 ± 4.07	**72.80 ± 5.21***	55.83 ± 7.03	**62.87 ± 5.18**
	NMI	53.42 ± 2.65	**58.39 ± 1.96***	**55.78 ± 6.06**	51.16 ± 3.08
	ARI	41.24 ± 1.54	**62.86 ± 3.20***	40.42 ± 9.59	**49.49 ± 5.69**
NG2	Acc	89.54 ± 0.08	**92.37 ± 0.98***	89.58 ± 0.11	**94.01 ± 0.00***
	NMI	55.81 ± 0.12	**66.18 ± 3.06***	55.39 ± 0.14	**67.46 ± 0.19***
	ARI	65.19 ± 0.13	**74.80 ± 3.31***	65.67 ± 0.18	**77.44 ± 0.18***
NG5	Acc	58.36 ± 2.34	**76.15 ± 6.72***	68.86 ± 5.93	**84.80 ± 0.89***
	NMI	46.33 ± 2.91	**52.77 ± 7.77***	43.40 ± 4.99	**63.78 ± 1.73***
	ARI	33.03 ± 3.59	**52.06 ± 10.16***	41.31 ± 6.47	**65.58 ± 1.78***
NG10	Acc	47.01 ± 2.44	47.90 ± 3.61	47.03 ± 0.24	**61.35 ± 2.14***
	NMI	**49.35 ± 3.54***	37.73 ± 1.81	48.33 ± 0.45	**51.24 ± 1.74***
	ARI	**30.18 ± 2.37***	24.50 ± 0.98	30.67 ± 0.15	**39.75 ± 2.84***

* Numbers marked indicate that the *p*-value is below 0.001.

WebKB4, NMF and SpKM give better clustering results than both proposed algorithms.

- On binary version of Reviews and Sports, which are data sets with a low balance coefficient, both proposed algorithms are the most efficient, whereas on the TF-IDF version of these same data sets, SpKM is significantly better.
- On the TF-IDF version of NG2, K-means is unable to find a partition into two clusters as required whereas on the TF-IDF version of Classic4, SpCo is unable to propose a partition into 4 clusters. Fig. 9
- summarizes Tables 8 and 9, we observe that whatever the type of data is (binary or continuous after TF-IDF transformation), our two algorithms DDKM and F-DDKM (the last two bars) are almost always better than the others. However, one can note that, except for SpKM, the TF-IDF transformation does not always improve the performance of the algorithms we have studied. For instance, if the TF-IDF transformation on NG10 significantly improved the accuracy of F-DDKM, it has a less significant impact on Sports.

Although both algorithms are more efficient on document data sets than other state-of-the-art algorithms, it is important to note that F-DDKM relies on two unknown parameters. The next subsection is devoted to the analysis of the possible parameters configuration.

6.4. Choice of α and β

One shortcoming of F-DDKM is that the user has to specify the fuzziness parameters α and β. The analysis of empirical results give us a hint on how we can set these values. Fig. 10 presents different situations we ran into while doing the experiments on real data sets. Each figure shows the variations of the accuracy, the NMI and the ARI according to different values of α and β. As α and β control the fuzziness of the partition, we can expect to find optimum for higher value of these parameters on data sets with overlapping rather than with well separated co-clusters.

For all four data sets we can see that the optimal choice of α and β is between 1.001 and 1.003. CSTR is a data set with no overlap, therefore values of α and β lower than 1.001 can also be considered as good choices. On NG5 and NG10, accuracies, NMI's and ARI's dropped significantly for values of the parameters greater than 1.003 and 1.005, respectively. The same phenomenon occurs on CSTR for values of parameters greater than 1.005. For Webkb4 (Fig. 10) we can observe several peaks including 1.002. In the latter case, it is worth noting that no real drop happens so that, other values like 1.008 and 1.00001 can also be considered as an option. Based on these empirical observations we recommend to set the value of α and β between 1.001 and 1.003.

6.5. Computational complexity

We study the computational complexity of the compared clustering and co-clustering algorithms. DDKM and F-DDKM have the same complexity but fuzzy approaches are known to require more computation time than hard ones. We repeat clustering 100 times for each algorithm on each data set. For illustration, we report the average convergence time on CSTR, Classic3, NG5 and NG10 in Fig. 11. We have deliberately chosen not to report running time performance obtained with ITCC - as in our opinion the matlab implementation that we used is not optimized.

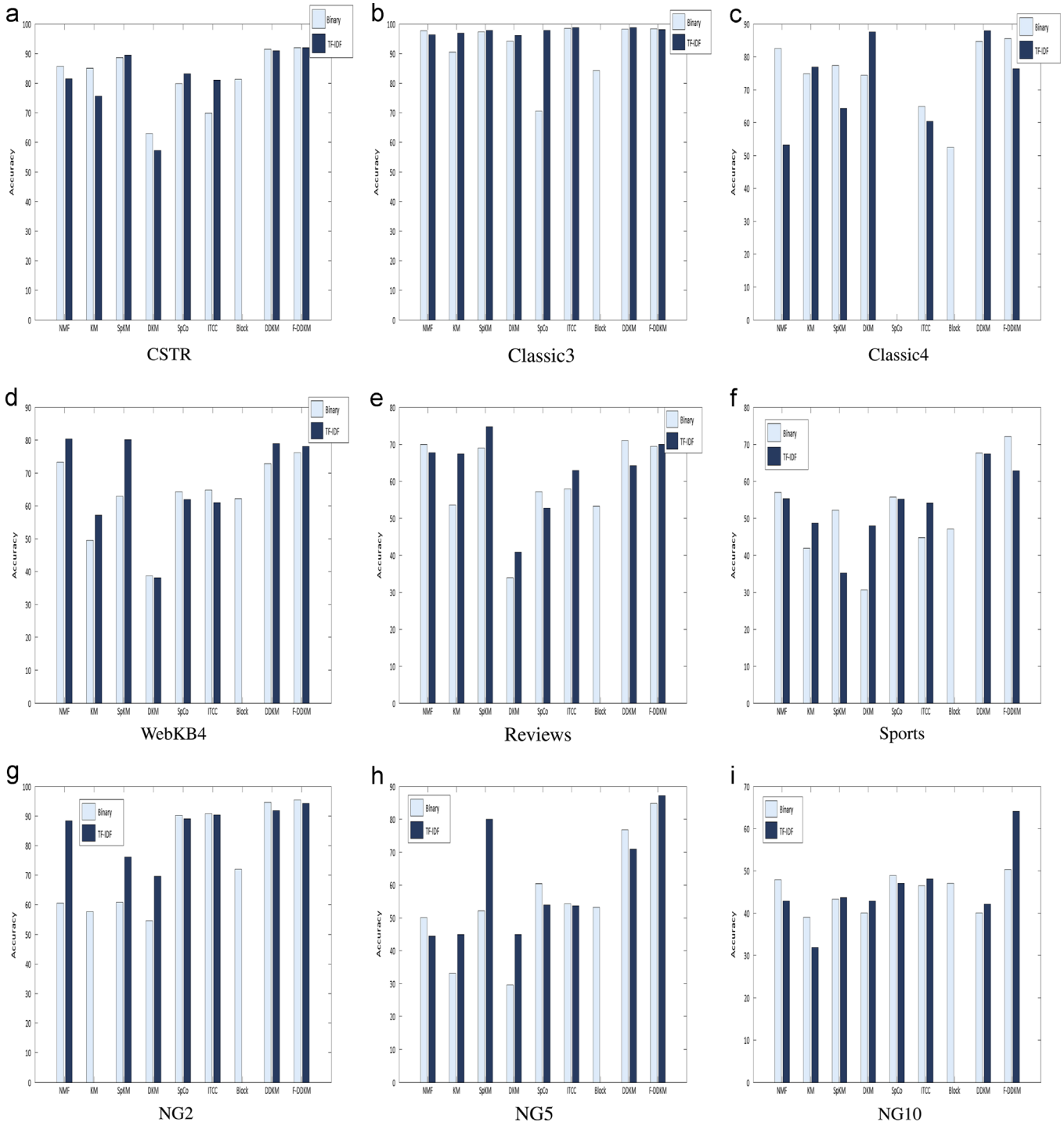

Fig. 9. Comparison of all algorithms in terms of Acc on binary and continuous data sets.

It is also important to stress that NMF, K-means and SpKM are only working on the set of objects while other algorithms are working on both sets of objects and features. The results show that the proposed algorithm DDKM is only slightly slower than NMF method while it requires far less time to converge than all other state-of-the-art algorithms. The fact that our algorithms do not require to look for an appropriate δ nor update δ during the co-clustering process, explain the speed of the proposed approach. On TF-IDF versions data sets, F-DDKM needs more time to converge than on binary ones and is always faster than SpCo. The same observations can be made for other data sets presented in this article.

7. Limitations

In our approach, we assume that there exists a block diagonal structure and that the values of $\mathbf{X}$ are comparable. Although this assumption is often true for many applications like document clustering or social networks, it can become a weakness when the diagonal structure is less evident, as it is the case of Reviews (see Fig. 12) or when there are outliers.

Indeed, since the proposed criteria are based on the maximum value of the data matrix, the presence of objects with abnormally high value(s) for one or more features will have an impact on the performance of our algorithms. In order to illustrate this point, we

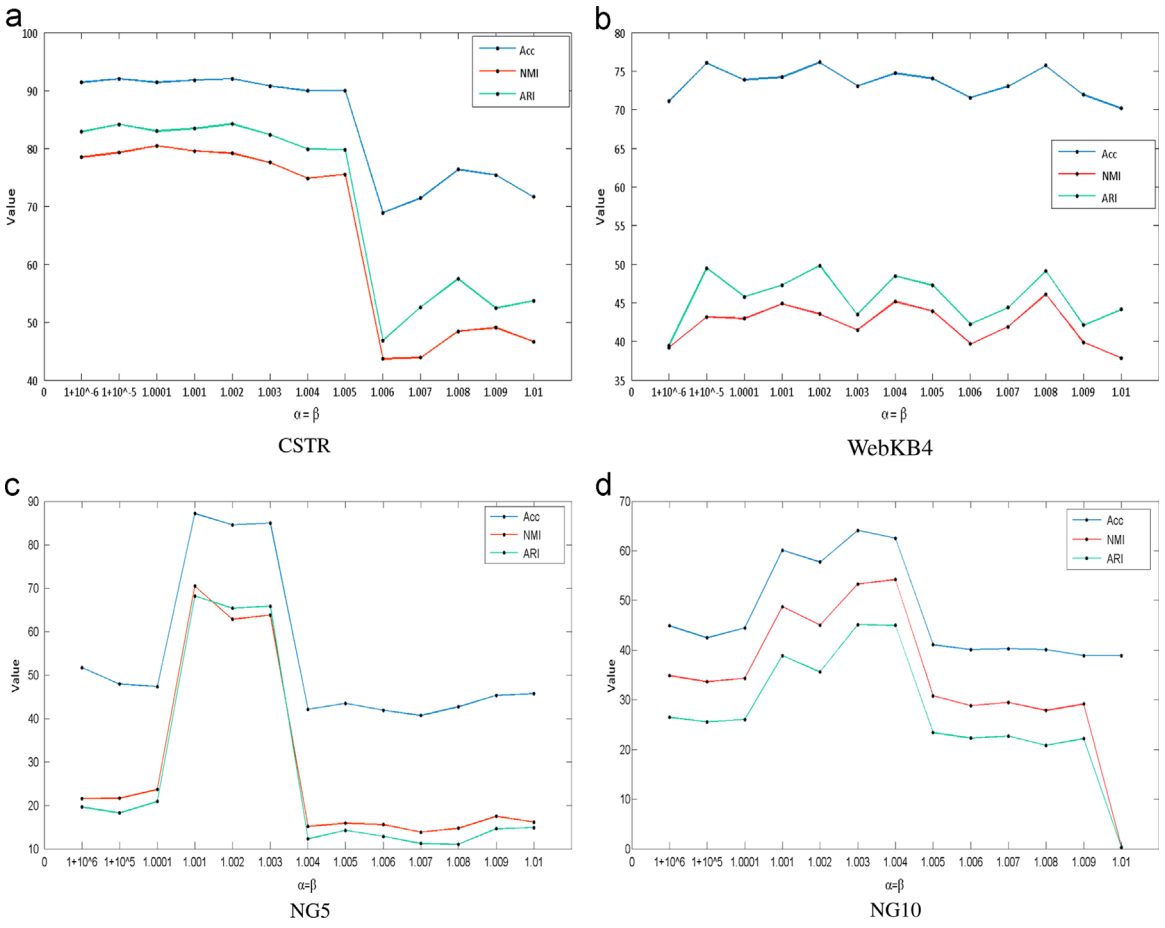

Fig. 10. Impact of α and β on the performance of F-DDKM on the binary (top row) and TF-IDF versions (bottom row) of data sets.

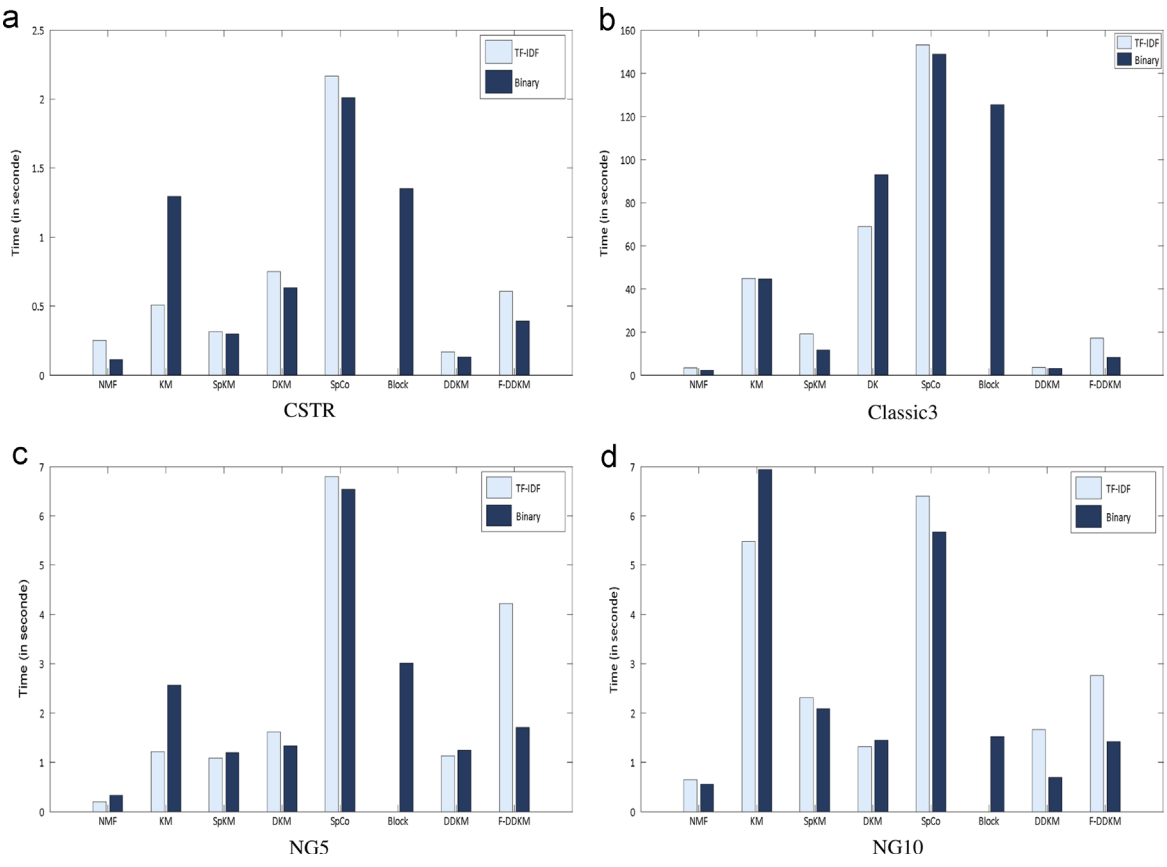

Fig. 11. Running time (in seconds) of algorithms on binary and TF-IDF versions of data sets.

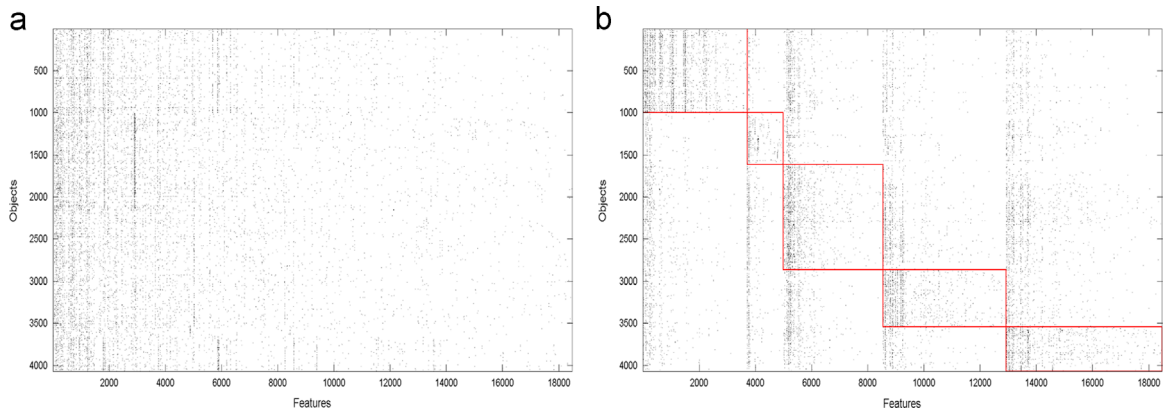

Fig. 12. Visualisation of the binary version of Reviews. (a) is the original data matrix reorganized according to the true partition while (b) is the same matrix reorganized according to the partitions obtained by DDKM.

Table 11
Parameters used to simulate the data set.

$$\mu = \begin{pmatrix} 10 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 2.5 \end{pmatrix} \quad \sigma = \begin{pmatrix} 2 & 1.5 & 1.5 \\ 1.5 & 2 & 1.5 \\ 1.5 & 1.5 & 2 \end{pmatrix}$$

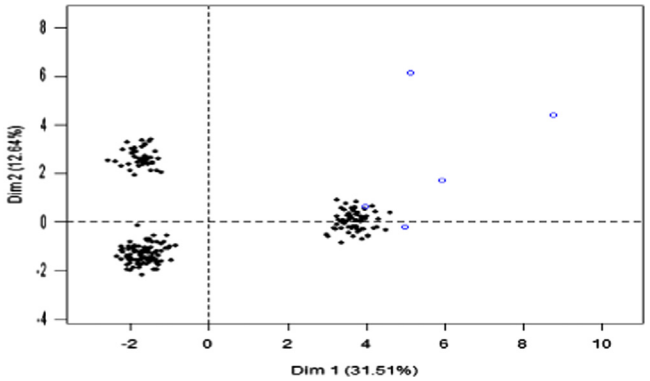

Fig. 13. Visualisation of the data set using the 2 first components from the PCA. This projection reveals the presence of five outliers on the right of the plan.

simulate a continuous data set of size 200×100 with a very clear diagonal structure composed of 3 blocks. We use the process described in the Section 5.1 with the parameters reported in Table 11.

We apply a standard normalisation to the data; we subtract the mean from each feature and divided this difference by the standard deviation of the feature. We run both DDKM and F-DDKM on this data set and obtain accuracy close to 100% for both of them. Then, we introduce 5 outliers; we generate these outliers uniformly at random from the interval $[0, 20]^d$. After the same standardization process than for the original data set, we run both algorithms and they both fail gathering all the objects into one single block. A simple and classic way to overcome this limitation is to visualise the data carefully before running the algorithms. For instance, the boxplots of features or the Principal Component Analysis (PCA) (see Fig. 13) can be used to identify the outliers and possibly to exclude them from the analysis.

Finally, both algorithms require an appropriate normalization such as binarization or TF-IDF to work efficiently. The choice of δ as the maximum value appears a good choice in document clustering context. However, a proper statistical analysis of the data upstream can allow a more suitable choice of δ and to properly handle the presence of outliers.

8. Conclusion

In this paper we presented DDKM, a fast co-clustering algorithm that looks for homogeneous diagonal blocks and F-DDKM, its fuzzy version. Compared with other methods, we demonstrated that our proposed algorithms are more effective for document-term partitioning (both on binary data and with TF-IDF transformation) and especially in the presence of classes having a high degree of overlap. In addition, DDKM requires less time to converge; up to 20 times less time than DKM and 40 times less time than SpCo commonly used in the domain of document clustering.

For further research, it will be worthwhile to investigate an efficient theoretical way to choose the values of the parameters α and β and to study in more detail their impact on the clustering performance.

Finally, in real world application, the knowledge of the number of co-clusters is mostly required. Another initiative will be to investigate an efficient way to assess this parameter.

References

[1] J.A. Hartigan, Direct clustering of a data matrix, J. Am. Stat. Assoc. (1972) 123–129.
[2] J.A. Hartigan, Clustering Algorithms, 99th Edition., John Wiley & Sons, Inc., New York, NY, USA, 1975.
[3] G. Govaert, Classification croisée (Ph.D. thesis), Université Paris 6, France (1983).
[4] Y. Cheng, G. Church, Biclustering of expression data, in: ISMB2000, 8th International Conference on Intelligent Systems for Molecular Biology, San Diego, California, 2000, pp. 93–103.
[5] H. Cho, I. Dhillon, Coclustering of human cancer microarrays using minimum sum-squared residue coclustering, IEEE/ACM Trans. Comput. Biol. Bioinform. 5 (3) (2008) 385–400.
[6] N. Gupta, S. Aggarwal, Mib: Using mutual information for biclustering gene expression data, Pattern Recognit. 43 (8) (2010) 2692–2697.
[7] S.C. Madeira, A.L. Oliveira, Biclustering algorithms for biological data analysis: a survey, IEEE/ACM Trans. Comput. Biol. Bioinform. 1 (2004) 24–45.
[8] A. Tanay, R. Sharan, R. Shamir, Biclustering algorithms: a survey, Handbook of Computational Molecular Biology, Vol. 9 (1–20), 2005, pp. 122–124.
[9] B. Hanczar, M. Nadif, Using the bagging approach for biclustering of gene expression data, Neurocomputing 74 (10) (2011) 1595–1605.
[10] B. Hanczar, M. Nadif, Ensemble methods for biclustering tasks, Pattern Recognit. 45 (11) (2012) 3938–3949.
[11] T. George, S. Merugu, A scalable collaborative filtering framework based on co-clustering, in: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM '05, 2005, pp. 625–628.

[12] M. Deodhar, J. Ghosh, Scoal: a framework for simultaneous co-clustering and learning from complex data, ACM Trans. Knowl. Discov. Data (TKDD) 4 (3) (2010) 11.
[13] G. Xu, Y. Zong, P. Dolog, Y. Zhang, Co-clustering analysis of weblogs using bipartite spectral projection approach, in: Knowledge-Based and Intelligent Information and Engineering Systems, Springer, Cardiff, 2010, pp. 398–407.
[14] M. Charrad, Y. Lechevallier, M.B. Ahmed, G. Saporta, Block clustering for web pages categorization, in: Intelligent Data Engineering and Automated Learning-IDEAL 2009, Springer, Burgos, 2009, pp. 260–267.
[15] I.S. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, in: KDD '01, 2001, pp. 269–274.
[16] M. Vichi, Double k-means clustering for simultaneous classification of objects and variables, in: Advances in Classification and Data Analysis, Springer, Berlin, 2001, pp. 43–52.
[17] I.V. Mechelen, H.H. Bock, P.D. Boeck, Two-mode clustering methods: a structured overview, Stat. Methods Med. Res. 13 (5) (2004) 363–394.
[18] H. Bock, Convexity based clustering criteria: theory, algorithm and applications in statistics, Stat. Methods Appl. 12 (2003) 293–318.
[19] C. Ding, X. He, H. Simon, On the equivalence of nonnegative matrix factorization and spectral clustering, in: SIAM Data Mining Conference, 2005.
[20] C. Ding, T. Li, W. Peng, H. Park, Orthogonal nonnegative matrix trifactorizations for clustering, in: ACM SIGKDD, 2006, pp. 126–135.
[21] G. Govaert, M. Nadif, Block clustering with bernoulli mixture models: comparison of different approaches, Comput. Stat. Data Anal. 52 (6) (2008) 3233–3245.
[22] R. Rocci, M. Vichi, Two-mode multi-partitioning, Comput. Stat. Data Anal. 52 (4) (2008) 1984–2003.
[23] T. Li, A general model for clustering binary data, in: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD '05, ACM, New York, NY, USA, 2005, pp. 188–197.
[24] I.S. Dhillon, S. Mallela, R. Kumar, A divisive information theoretic feature clustering algorithm for text classification, J. Mach. Learn. Res. 3 (2003) 1265–1287.
[25] L. Labiod, M. Nadif, Co-clustering for binary and categorical data with maximum modularity, in: 2011 IEEE 11th International Conference on Data Mining (ICDM), 2011, pp. 1140–1145.
[26] G. Govaert, M. Nadif, Fuzzy clustering to estimate the parameters of block mixture models, Soft Comput. 10 (5) (2006) 415–422.
[27] M.-S. Yang, C.-Y. Lin, Block fuzzy k-modes clustering algorithm, in: Proceedings of the 18th International Conference on Fuzzy Systems, FUZZ-IEEE'09, IEEE Press, Piscataway, NJ, USA, 2009, pp. 384–389.
[28] C.-H. Oh, K. Honda, H. Ichihashi, Fuzzy clustering for categorical multivariate data, in: Joint 9th IFSA World Congress and 20th NAFIPS International Conference, 2001.
[29] K. Kummamuru, A. Dhawale, R. Krishnapuram, Fuzzy co-clustering of documents and keywords, in: The 12th IEEE International Conference on Fuzzy Systems FUZZ '03, 2003.
[30] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, 9th Edition., Wiley-Interscience, New York, 1990.
[31] A partitioning based algorithm to fuzzy co-cluster documents and words, Pattern Recognit. Lett. 27 (3) (2006) 151–159.
[32] W.-C. Tjhi, L. Chen, A heuristic-based fuzzy co-clustering algorithm for categorization of high-dimensional data, Fuzzy Sets Syst. 159 (4).
[33] Y. Jiang, F.-L. Chung, S. Wang, Z. Deng, J. Wang, P. Qian, Collaborative fuzzy clustering from multiple weighted views, IEEE Trans. Cybern. 45 (4) (2015) 688–701.
[34] M. Hanmandlu, O.P. Verma, S. Susan, V. Madasu, Color segmentation by fuzzy co-clustering of chrominance color features, Neurocomputing 120 (2013) 235–249, Image Feature Detection and Description.
[35] T. Havens, J. Bezdek, C. Leckie, L. Hall, M. Palaniswami, Fuzzy c-means algorithms for very large data, IEEE Trans. Fuzzy Syst. 20 (6) (2012) 1130–1146.
[36] Y. Jiang, F.L. Chung, S. Wang, Enhanced fuzzy partitions vs data randomness in fcm, J. Intell. Fuzzy Syst. 27 (4) (2014) 1639–1648.
[37] T. Eckes, P. Orlik, An error variance approach to two-mode hierarchical clustering, J. Classif. 10 (1) (1993) 51–74.
[38] D. Baier, W. Gaul, M. Schader, Two-mode overlapping clustering with applications to simultaneous benefit segmentation and market structuring, in: R. Klar, O. Opitz (Eds.), Classification and Knowledge Organization, Springer, Heidelberg, 1997.
[39] G. Govaert, M. Nadif, Co-Clustering, John Wiley & Sons, 2013.
[40] B. Mirkin, P. Arabie, L. Hubert, Additive two-mode clustering: the error-variance approach revisited, J. Classif. 12 (2) (1995) 243–263.
[41] G. Govaert, M. Nadif, Clustering with block mixture models, Pattern Recognit. 36 (2003) 463–473.
[42] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. (2003) 583–617.
[43] L. Hubert, P. Arabie, Comparing partitions, J. Classif. (1985) 193–218.
[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[45] I.S. Dhillon, D.S. Modha, Concept decompositions for large sparse text data using clustering, Mach. Learn. 42 (1–2) (2001) 143–175.
[46] M.W. Berry, M. Browne, A.N. Langville, V.P. Pauca, R.J. Plemmons, Algorithms and applications for approximate nonnegative matrix factorization, in: Computational Statistics and Data Analysis, 2006, pp. 155–173.
[47] X.V. Nguyen, Gene clustering on the unit hypersphere with the spherical k-means algorithm: Coping with extremely large number of local optima, in: International Conference on Bioinformatics & Computational Biology, BIOCOMP, 2008, pp. 226–233.
[48] N.K.V.J.K. Gupta, S. Singh, Mtba: Matlab toolbox for biclustering analysis, IEEE, 2013, pp. 94–97.

Charlotte Laclau is currently a third year Ph.D. student in Computer Science working under the supervision of Mohamed Nadif at the LIPADE (Laboratory of Informatics Paris Descartes) at the University of Paris Descartes. Her main research interests involve clustering and co-clustering algorithms with a focus on the feature selection problem.

Mohamed Nadif is Professor of the University Paris Descartes and researcher at the Laboratory LIPADE (Laboratory of Informatics Paris Descartes). He is the head of “Machine Learning for Data Science” team. He teaches data mining, machine learning techniques and multivariate analysis. He is responsible for the master's degree in machine learning at Paris Descartes University. His current researches interests include cluster analysis, co-clustering, data analysis, data mining, visualization, text mining and mixture models.