Model-based co-clustering for mixed type data

Margot Selosse, Julien Jacques, Christophe Biernacki

Model-based co-clustering for mixed type data

Margot Selosse[a,b,*], Julien Jacques[a,b], Christophe Biernacki[c,d]

[a] *Laboratoire ERIC, 5 Avenue Pierre Mendès France, 69500 Bron*
[b] *Université Lumière Lyon 2, 86 Rue Pasteur, 69007 Lyon*
[c] *Université de Lille - UFR de Mathématiques - Cité Scientifique - 59655 Villeneuve d'Ascq Cedex*
[d] *INRIA, 40, av. Halley - Bât A - Park Plaza 59650 Villeneuve d'Ascq*

Abstract

The importance of clustering for creating groups of observations is well known. The emergence of high-dimensional data sets with a huge number of features leads to co-clustering techniques, and several methods have been developed for simultaneously producing groups of observations and features. By grouping the data set into blocks (the crossing of a row-cluster and a column-cluster), these techniques can sometimes better summarize the data set and its inherent structure. The Latent Block Model (LBM) is a well-known method for performing co-clustering. However, recently, contexts with features of different types (here called mixed type data sets) are becoming more common. The LBM is not directly applicable to this kind of data set. Here a natural extension of the usual LBM to the "Multiple Latent Block Model" (MLBM) is proposed in order to handle mixed type data sets. Inference is performed using a Stochastic EM-algorithm that embeds a Gibbs sampler, and allows for missing data situations. A model selection criterion is defined to choose the number of row and column clusters. The method is then applied to both simulated and real data sets.

Keywords: co-clustering, mixed-type data, latent block model

*Corresponding author
Email addresses: margot.selosse@gmail.com (Margot Selosse), julien.jacques@univ-lyon2.fr (Julien Jacques), christophe.biernacki@inria.fr (Christophe Biernacki)

1. Introduction

Clustering algorithms have become a widely used method due to their ability to provide new insights into unlabeled data sets. They consist in forming homogeneous groups of observations referred to as "clusters". Clustering algorithms highlight the data's inherent structure. However, the recent "big-data" phenomenon has greatly increased the number of features, leading to the emergence of high-dimensional data sets. Clustering techniques are consequently not always sufficient to discern the structure. The analysis of a cluster relies on a representative of the cluster (mean, mode...). However, the latter is itself described by a large number of features, which makes it more difficult to interpret and makes the summary of the data set less useful. From this consideration comes the need to also "summarize" the features, which can be done by gathering them into clusters, in parallel with the usual clustering of observations. Co-clustering methods seems to be a good option for performing this task because they perform joint clustering of rows and columns. The initially large data matrix can be summarized by a limited number of blocks that result from combining row-clusters and column-clusters.

Among the most famous co-clustering techniques, the Non-negative Matrix Tri-Factorization consists in factorizing the $N \times P$ data matrix $\boldsymbol{x}$ into three matrices $\boldsymbol{a}$ (of size $N \times G$), $\boldsymbol{b}$ ($G \times H$), $\boldsymbol{c}$ ($H \times P$), with the property that all three matrices have non-negative elements, see for instance [1]. More specifically, the approximation of $\boldsymbol{x}$ by $\boldsymbol{x} \approx \boldsymbol{abc}$ is achieved by minimizing the error function $\underset{(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{min} ||\boldsymbol{x} - \boldsymbol{abc}||_F$, with the constraints ($\boldsymbol{a} \geq 0, \boldsymbol{b} \geq 0, \boldsymbol{c} \geq 0$), meaning that all elements of $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ are greater than 0, and $||.||_F$ being a matrix norm to be chosen. The matrix $\boldsymbol{b}$ represents the *block* matrix: an element b_{gh} of $\boldsymbol{b}$ summarizes the observations belonging to row-cluster g and column-cluster h. Despite the non-negative property of the matrices, it is not always easy to interpret the resulting matrices. For example, matrices $\boldsymbol{a}$ and $\boldsymbol{c}$ are not always normalized which makes it difficult to interpret them in terms of rows and columns belonging to corresponding clusters. Furthermore, this technique depends on the

choice of the distance measure. Conversely, probabilistic approaches propose normalized membership matrices, and do not require the user to choose a particular distance measure. In the Latent Block Model [2], referred to as "LBM", the elements of a block are modeled by a parametric distribution. Therefore, the results give more information than a simple scalar, as mentioned in the previous methods. Each block is therefore interpretable via the parameters of the block-distribution. Moreover, model selection criterion such as the ICL criterion [3] can be used for model selection purposes, including the choice of the number of co-clusters. This technique has proved its efficiency in co-clustering several types of data: continuous [4], nominal [5], binary [6], ordinal [7], and functional [8, 9]. For this reason, an extension of this model is used in the present work, although originally it was not able to take heterogeneous data as an input.

Heterogeneous data sets are composed of features of different types. For example, in medicine, a patient's file can be composed of images (X-rays), text (medical reports), continuous data (age, blood test results...), categorical data (social category, pregnancy, drug addiction...), and even functional data (pulse, blood pressure...). Several clustering frameworks have been developed to address this particularity. The latent class model [10] is frequently used. It assumes that the variables are conditionally independent upon the row-cluster membership. Consequently, the joint probability distribution function (PDF) of the features of different types is obtained by the product of the PDFs of each individual feature (see an implementation using Mixtcomp software [11]). However, when the variables are inherently correlated in a row-cluster, this model is not suitable. To overcome this issue, the authors of [12] want to conserve standard marginal distributions but also try to loosen the conditional independence on the variables. For this purpose, they use copula, which allow definition of both the dependence model and the type of marginal distributions. The proposed model relies on the main assumption that each cluster follows a Gaussian copula. However, the authors note that model complexity increases with the number of variables, which is not suitable in a big-data context. Another way to address the issues of heterogeneous data is to see some variables as the man-

ifestation of a latent vector. For example, in [13], the clustMD model considers continuous and categorical data (nominal and ordinal) and assumes that a categorical variable is the representation of an underlying latent continuous variable. Then, it is assumed that the continuous variables (observed and unobserved) follow a multivariate Gaussian mixture model. Until now, these methods have proposed models for basic data such as categorical (nominal or ordinal) and continuous data. In [14], the authors allow the introduction of more complex data such as functional data or networks by projecting the data set into a reproducing kernel Hilbert space. Regarding the analysis of variables, multiblock methods, widely used in Chemistry and Biology, handle data sets that share the same observations but have variables measured differently. They aim at finding underlying relationships between these data sets. In particular, multiblock component models use latent variables to summarize the relevant information between and within the sets (see [15] for a complete survey).

However, none of these techniques were developed in a co-clustering framework. To the best of our knowledge, the only work to co-cluster heterogeneous data is [16], which extends the LBM for data sets with continuous and binary data. The present work goes further by proposing an extension that can take into account four types of data: categorical, continuous, count and ordinal data. Furthermore, the inference algorithm can deal with missing values and proposes a way to impute them. Finally, the Integrated Completed Likelihood (ICL) criterion [3] is adapted to the proposed model in order to select the number of row-clusters and column-clusters.

Co-clustering techniques can be seen as an efficient alternative method to the selection of variables thanks to its parsimony, especially in very high dimensions. In addition, it can produce interpretable sets of variables since it can group redundant variables or noisy variables. In this way, a first, naive answer is to manually select the informative blocks, but [17] alternatively defines a model that automatically distinguishes the informative blocks for textual data sets. For mixed data, variable selection is more challenging. [18] performs clustering while incorporating variable selection and this method can produce homogeneous row-

clusters. However, compared to co-clustering, it does not provide interpretable column-clusters, which may be essential for the summary of the data set, in particular with a high number of variables.

The paper is organized as follows. Section 2 gives an overview of the LBM to help understanding of this paper. Then, it proposes an extension to a new LBM version that allows heterogeneous data sets. Section 3 proposes an algorithm for model inference, based on a Stochastic Expectation Maximization [19] algorithm coupled with a Gibbs sampler. In Section 4, a description of the different types of data that can be taken into account with this method is given, and formulas for model inference are presented. Section 5 assesses the efficiency of the proposed method on simulated data while Section 6 shows how the method performs on real data sets. Section 7 provides a conclusion.

2. Multiple Latent Block Model

Here, the Latent Block Model is presented. Then its extension to the Multiple Latent Block Model is detailed.

2.1. Latent Block Model

The LBM is a widely used model to perform co-clustering [2]. Basically, it assumes that all elements of a block follow the same distribution. In this section, the assumptions used for the LBM are defined, and the mathematical details are given.

The LBM considers that all features can potentially be grouped together (restrictions will be imposed in the next section to define the "Multiple LBM"). Consider the data matrix $\boldsymbol{x} = (x_{ij})_{i,j}$, where $i \in \{1, \ldots, N\}$ is the row (observation) index and $j \in \{1, \ldots, J\}$ is the column (feature) index. It is assumed that there are G row-clusters and H column-clusters that correspond to a partition $\boldsymbol{v} = (v_{ig})_{i,g}$ of the rows and a partition $\boldsymbol{w} = (w_{jh})_{j,h}$ of the columns, with $1 \leq g \leq G$ and $1 \leq h \leq H$, where v_{ig} is equal to 1 if row i belongs to cluster g, and 0 otherwise; and similarly w_{jh} is equal to 1 when column j belongs to

cluster h, and 0 otherwise. In order to simplify the notations, the underlying range of variation will be omitted in the sums and products, and they will be written $\sum_i$, $\sum_j$, $\sum_g$ and $\sum_h$, $\prod_i$, $\prod_j$, $\prod_g$ and $\prod_h$.

The first LBM assumption is that the univariate random variables x_{ij} are conditionally independent given the row and column partitions $\boldsymbol{v}$ and $\boldsymbol{w}$. Therefore, the conditional probability density function of $\boldsymbol{x}$ given $\boldsymbol{v}$ and $\boldsymbol{w}$ is written:

$$p(\boldsymbol{x}|\boldsymbol{v},\boldsymbol{w};\boldsymbol{\alpha}) \;=\; \prod_{i,j,g,h} p(x_{ij};\alpha_{gh})^{v_{ig}w_{jh}},$$

where $\boldsymbol{\alpha} = (\alpha_{gh})_{g,h}$ is the distribution parameters of block (g,h).

The second LBM assumption is that the latent variables $\boldsymbol{v}$ and $\boldsymbol{w}$ are independent so $p(\boldsymbol{v},\boldsymbol{w};\boldsymbol{\gamma},\boldsymbol{\rho}) \;=\; p(\boldsymbol{v};\boldsymbol{\gamma})p(\boldsymbol{w};\boldsymbol{\rho})$ with:

$$p(\boldsymbol{v};\boldsymbol{\gamma}) \;=\; \prod_{i,g}\gamma_g^{v_{ig}} \;\text{ and }\; p(\boldsymbol{w};\boldsymbol{\rho}) \;=\; \prod_{j,h}\rho_h^{w_{jh}},$$

where $\gamma_g = p(v_{ig}=1)$ and $\rho_h = p(w_{jh}=1)$. This implies that, for all i, the distribution of $\boldsymbol{v}_i$ is the multinomial distribution $\mathcal{M}(\gamma_1,\ldots,\gamma_G)$ and does not depend on i. Similarly, for all j, the distribution of $\boldsymbol{w}_j$ is the multinomial distribution $\mathcal{M}(\rho_1,\ldots,\rho_H)$ and does not depend on j.

From these considerations, the LBM parameter is defined as $\boldsymbol{\theta} = (\boldsymbol{\gamma},\boldsymbol{\rho},\boldsymbol{\alpha})$, with $\boldsymbol{\gamma} = (\gamma_1,\ldots,\gamma_G)$ and $\boldsymbol{\rho} = (\rho_1,\ldots,\rho_H)$ the row and column mixing proportions. Therefore, if V and W are the sets of all possible labels $\boldsymbol{v}$ and $\boldsymbol{w}$, the probability density function of $\boldsymbol{x}$ is written:

$$p(\boldsymbol{x};\boldsymbol{\theta}) \;=\; \sum_{(\boldsymbol{v},\boldsymbol{w})\in V\times W} \prod_{i,g}\gamma_g^{v_{ig}} \prod_{j,h}\rho_h^{w_{jh}} \prod_{i,j,g,h} p(x_{ij};\alpha_{gh})^{v_{ig}w_{jh}}. \qquad (1)$$

2.2. Extension to Multiple Latent Block Model

Now, consider a matrix $\boldsymbol{x}$ composed of D different sets of features. It has N rows and $J = \sum_{d=1}^{D} J_d$ columns, J_d being the number of features of the d-th set:

$$\boldsymbol{x} = (\boldsymbol{x}^1,\ldots,\boldsymbol{x}^D), \text{ with } \boldsymbol{x}^d = (x_{ij}^d)_{i=1,\ldots,N;\; j=1,\ldots,J_d}.$$

Here, the idea of "sets" of features is introduced to define the features we potentially want to group together in a column-cluster, and those we do not

want to be together. Thus, features of a same set can be grouped together in an *intra-set* column-cluster; features of different sets cannot. There are two reasons for separating features into different sets: a technical one and a semantic one. Firstly, two features of different types (*e.g.* a categorical feature and a continuous one) are chosen so as not to be modeled with a similar probability distribution, but rather with a standard distribution suitable to their type. Since it will be assumed later that all the features in a column-cluster have the same PDF., such an assumption is not suitable for features of different types. This is the reason for this work. Secondly, the user can consider, for practical reasons, that some features necessarily have to be separated because it does not make sense to gather them in a same column cluster. This case is not explored in the present work, but the reader can refer to [20] for a detailed example. The sets of elements $(\boldsymbol{x}^1, \ldots, \boldsymbol{x}^D)$ are annotated $(\boldsymbol{x}^d)_d$ with $d \in \{1, \ldots, D\}$.

In the co-clustering framework, it is assumed that G row-clusters and $H = H_1 + \ldots + H_D$ column-clusters exist, and that they are inherent to the matrix $\boldsymbol{x}$. Moreover, the sums and the products relating to sets of features will be written in subscript by the letter d. Again, the underlying range of variation will be omitted in the sums and products, thus they are written $\sum_d$ and $\prod_d$.

Finally, a data set may have missing data. To deal with this aspect, the d^{th} matrix $\boldsymbol{x}^d$ is said to be made up of two sets $\check{\boldsymbol{x}}^d$ and $\hat{\boldsymbol{x}}^d$, where $\check{\boldsymbol{x}}^d$ is the observed data, and $\hat{\boldsymbol{x}}^d$ is the missing data. An element of $\boldsymbol{x}^d$ will be annotated $\check{x}^d_{ij}$ if x^d_{ij} is observed, and $\hat{x}^d_{ij}$ otherwise. To model missing values, three main processes exist in data analysis (we refer to [21] for a complete review). The Missing Completely At Random (MCAR) process assumes that the missing data mechanism is unrelated to the values of any variables: for example, in a survey, participants accidentally skipped questions. The Missing At Random (MAR) process supposes that a missing value has nothing to do with the variable whose value is missing, but it does have to do with the values of other variables. For example, males are less likely to fill in a depression survey but this has nothing to do with their level of depression, after accounting for maleness. The last process is called Missing Not At Random (MNAR) and occurs when the missing value

is directly influenced by the variable itself. For instance, a drug addict may not answer a question about drugs precisely because of their addiction. In the present work, it is assumed that the whole missing process is MAR, because it is the most frequent situation encountered in practice [22].

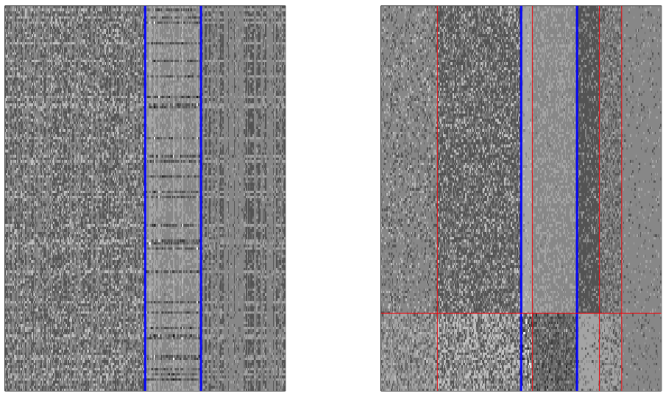
The LBM relies on the assumption that the block's elements are the realizations of a random variable that follows a distribution with parameter $\boldsymbol{\alpha}$. In this work, we chose to adopt a standard distribution for each kind of feature (e.g. Gaussian for continuous data and Poisson for count data). In this context, if the elements of the blocks are not of the same type, it is not possible to consider that they were sampled from the same distribution. The Multiple Latent Block Model (MLBM) was defined in [23] for two matrices of binary data separated into two blocks for semantic reasons. In this paper, the MLBM is extended for $D \geq 1$ matrices such that each matrix may have continuous, categorical, ordinal or count data. In this model, the columns of the matrix $\boldsymbol{x}$ are reordered such that $\boldsymbol{x}$ is composed of D matrices put side by side, each matrix containing features of homogeneous type as described above. The co-clustering is performed in such a way that features of different types cannot be part of a same column-cluster. Consequently, it is possible to define a distribution on each block because it is made of variables of the same type. Figure 1 illustrates the idea behind this model.

Let $\boldsymbol{w}^d$ denote the column partitions of the d-th matrix ($1 \leq d \leq D$), $\boldsymbol{\rho}^d = (\rho_1^d, \ldots, \rho_{H_d}^d)$ the corresponding mixing proportions, and let us introduce the notations $\boldsymbol{w} = (\boldsymbol{w}^d)_d$ and $\boldsymbol{\rho} = (\boldsymbol{\rho}^d)_d$.

The MLBM relies on several assumptions. The first one states that the D matrices data are conditionally independent of the row and column partitions, and specifically that, for all $t \neq d$ the matrix $\boldsymbol{x}^d$ does not depend on the column partitions $\boldsymbol{w}^t$:

$$p(\boldsymbol{x}|\boldsymbol{v}, \boldsymbol{w}) = p(\boldsymbol{x}^1|\boldsymbol{v}, \boldsymbol{w}^1) \times \ldots \times p(\boldsymbol{x}^D|\boldsymbol{v}, \boldsymbol{w}^D).$$

The other assumptions of the MLBM are similar to those of the LBM. Firstly, the univariate random variables x_{ij}^d are assumed to be conditionally independent



(a) (b)

Figure 1: (a) is the matrix $\boldsymbol{x}$. The blue lines represent the separation of the features that are not of same type. (b) is the matrix after having performed a co-clustering. The red lines represent the co-clusters limits.

on partitions $\boldsymbol{v}$ and $\boldsymbol{w}^d$. Thus, the conditional probability function of $\boldsymbol{x}$ given $\boldsymbol{v}$ and $(\boldsymbol{w}^d)_d$ is expressed as:

$$p(\boldsymbol{x}|\boldsymbol{v},\boldsymbol{w};\boldsymbol{\alpha}) = \prod_{i,j,g,h,d} p(x_{ij}^d;\alpha_{gh}^d)^{v_{ig}w_{jh}^d},$$

where $\boldsymbol{\alpha} = (\boldsymbol{\alpha}^d)_d$ with $\boldsymbol{\alpha}^d = (\alpha_{gh}^d)_{g,h}$ is the distribution parameters of block (g,h) of matrix $\boldsymbol{x}^d$.

Second, the latent variables $\boldsymbol{v}, \boldsymbol{w}^1, \ldots, \boldsymbol{w}^D$ are assumed to be independent, so: $p(\boldsymbol{v},\boldsymbol{w};\boldsymbol{\gamma},\boldsymbol{\rho}) = p(\boldsymbol{v};\boldsymbol{\gamma})\prod_d p(\boldsymbol{w}^d;\boldsymbol{\rho}^d)$, where:

$$p(\boldsymbol{v};\boldsymbol{\gamma}) = \prod_{i,g}\gamma_g^{v_{ig}} \text{ and } p(\boldsymbol{w}^d;\boldsymbol{\rho}^d) = \prod_{j,h}{\rho_h^d}^{w_{jh}^d}.$$

The MLBM parameter is thus defined by $\boldsymbol{\theta} = (\boldsymbol{\gamma},\boldsymbol{\rho},\boldsymbol{\alpha})$. Moreover, if V and $(W^d)_d$ are the sets of all possible labels $\boldsymbol{v}$ and $(\boldsymbol{w}^d)_d$, the probability density function $p(\boldsymbol{x};\boldsymbol{\theta})$ is written:

$$p(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{(\boldsymbol{v},(\boldsymbol{w}^d)_d)\in V\times(W^d)_d} \prod_{i,g}\gamma_g^{v_{ig}} \prod_d\prod_{j,h}{\rho_h^d}^{w_{jh}^d} \prod_{i,j,g,h} p(x_{ij}^d;\alpha_{gh}^d)^{v_{ig}w_{jh}^d}. \quad (2)$$

Note that so far the type of $p(x_{ij}^d; \alpha_{gh}^d)$ has not been defined. It will be in Section 4, based on the type of x_{ij}^d (nominal, ordinal, continuous, ...).

3. Model Inference

The MLBM inference aims at estimating $\boldsymbol{\theta}$ that maximizes the observed log-likelihood:

$$l(\boldsymbol{\theta}; \check{\boldsymbol{x}}) = \sum_{\hat{\boldsymbol{x}}} \log p(\boldsymbol{x}; \boldsymbol{\theta}). \tag{3}$$

The EM-algorithm [24] is a well known method for performing this task with latent variables. However, with regard to the co-clustering case, it is not computationally tractable. Indeed, this method needs to compute the expectation of the complete data log-likelihood. However, in the case of $D = 1$, this expression contains the probability $p(v_{ig} = 1, w_{jh} = 1|\boldsymbol{x}; \boldsymbol{\theta})$, which needs to consider all possible values for $\boldsymbol{v}_{i'}$ and $\boldsymbol{w}_{j'}$ with $i' \neq i$ and $j' \neq j$. The E-step would require calculation of $G^N \times H^J$ terms: for example, if $G = 2$, $H = 2$, $N = 20$ and $J = 20$, each E step of the EM algorithm would need to compute $2^{20} \times 2^{20} \approx 10^{12}$ terms. Different alternatives to the EM algorithm exist, such as the variational EM algorithm, the SEM-Gibbs algorithm, and other algorithms linked to Bayesian inference [2]. The SEM-Gibbs version is used in the present work because in addition to being known to be less sensitive to initialization, it is simple to implement. Furthermore, it easily handles missing values $\hat{\boldsymbol{x}}$ in $\boldsymbol{x}$, which is an important advantage for real data sets.

3.1. SEM-Gibbs algorithm

The SEM-Gibbs algorithm begins with an initialization of partitions, parameters and missing values $\boldsymbol{v}^{(0)}, \boldsymbol{w}^{(0)}, \boldsymbol{\theta}^{(0)}, \hat{\boldsymbol{x}}^{(0)}$. This initialization process is described in more details later. The following five steps describe the q-th iteration, with $q \in (1, ..., nbSEM)$. The choice of the number of iterations ($nbSEM$) will also be described later.

(a) Sampling row partitions. Generate the row partitions with:

$$p(v_{ig}^{(q)} = 1 \mid \boldsymbol{x}, \boldsymbol{w}^{(q-1)}; \boldsymbol{\theta}^{(q-1)}) \propto \gamma_g^{(q-1)} \times \prod_d t_g^d(\boldsymbol{x}_{i\cdot}^d | \boldsymbol{w}^{d^{(q-1)}}; \boldsymbol{\alpha}^{d^{(q-1)}}), \quad (4)$$

where $t_g^d(\boldsymbol{x}_{i\cdot}^d | \boldsymbol{w}^{d^{(q-1)}}; \boldsymbol{\alpha}^{d^{(q-1)}}) = \prod_{j,h} f(x_{ij}^d; {\alpha_{gh}}^{d^{(q-1)}})^{{w_{jh}^d}^{(q-1)}}$ with $\boldsymbol{x}_{i\cdot}^d = (x_{ij}^d)_j$. Note that this probability depends on the data type of the d-th matrix through the PDF $f(x_{ij}^d; {\alpha_{gh}}^{d^{(q-1)}})$, whose exact expression will be given in Section 4.

(b) First M-step. This first M-step consists in updating the co-cluster parameters $\boldsymbol{\theta}^{(q)}$ to maximize the completed log-likelihood (3). The row mixing proportions are consequently updated by:

$$\gamma_g^{(q)} = \tfrac{1}{N} \sum_i v_{ig}^{(q)},$$

and the parameter $\boldsymbol{\alpha}^{d^{(q)}}$ is updated as well. However, the computations depend on the type of matrix $\boldsymbol{x}$ features. Section 4 describes how to update $\boldsymbol{\alpha}^{d^{(q)}}$ according to the type of variables.

(c) Sampling column partitions. For all $d \in \{1, \ldots, D\}$ generate the column partitions for the d-th matrix $\boldsymbol{x}^d$ with:

$$p({w_{jh}^d}^{(q)} = 1 \mid \boldsymbol{x}^d, \boldsymbol{v}^{(q)}; \boldsymbol{\theta}^{(q)}) \propto \rho_h^{d(q)} \times s_h^d(\boldsymbol{x}_{\cdot j}^d | \boldsymbol{v}^{(q)}; \boldsymbol{\alpha}^{d^{(q-1)}}), \quad (5)$$

where $s_h^d(\boldsymbol{x}_{\cdot j}^d | \boldsymbol{v}^{(q)}; \boldsymbol{\alpha}^{d^{(q)}}) = \prod_{i,g} f(x_{ij}^d; {\alpha_{gh}}^{d^{(q-1)}})^{v_{ig}^{(q)}}$ with $\boldsymbol{x}_{\cdot j}^d = (x_{ij}^d)_i$. Here, note that s_h^d obviously depends on the type of the d-th matrix (see Section 4).

(d) Second M-step. In this second M-step, the column mixing proportions are updated by:

$$\rho_h^{d(q)} = \tfrac{1}{J_d} \sum_j {w_{jh}^d}^{(q)},$$

and the parameter $\boldsymbol{\alpha}^{d^{(q)}}$ is also updated depending on the data type of the d-th matrix (see Section 4).

(e) Missing values imputation. Generate the missing data $\hat{x}_{ij}^{d^{(q)}}$ according to:

$$p(\hat{x}_{ij}^{d^{(q)}}|\check{\boldsymbol{x}}, \boldsymbol{v}^{(q)}, \boldsymbol{w}^{d^{(q)}}; \boldsymbol{\theta}^{(q)}) = \prod_{g,h} f(\hat{x}_{ij}^{d^{(q)}}; \boldsymbol{\alpha}^{d^{(q)}})^{v_{ig}^{(q)} w_{jh}^{d\ (q)}}.$$

The SEM-Gibbs algorithm is iterated for a given number of iterations. The first part of these iterations is called the burn-in period, meaning that the parameters of $\boldsymbol{\theta}$ are not yet simulated according to its stationary distribution. Consequently, only iterations that occurred after this burn-in period are taken into account and are referred to as the sampling distribution hereafter. While the final estimations of discrete parameters give the mode of the sampling distribution, the final estimations of the continuous parameters give the mean of the sample distribution. This leads to a final estimation of $\boldsymbol{\theta}$ called $\hat{\boldsymbol{\theta}}$. Then, a sample of $(\hat{\boldsymbol{x}}, \boldsymbol{v}, \boldsymbol{w})$ is simulated by iterating steps (a), (c) and (e) of the SEM-Gibbs algorithm with $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. The final partitions $(\hat{\boldsymbol{v}}, \hat{\boldsymbol{w}})$ and the missing observations $\hat{\boldsymbol{x}}$ are estimated using the mode of their marginal sampled distribution.

Initialization. The algorithm starts with an initialization of the partitions. Then the mixing proportions and the block parameters are estimated with regard to these partitions. In the case of $D = 1$, this initialization can be made randomly [25], but with $D > 1$, this often leads to empty clusters. In this work, a specific initialization strategy was worked out to tackle this issue. It relies on an initial random initialization. However, for the first I iterations (such that I is less than or equal to the burn-in number of iterations), whenever a row-cluster becomes empty, a percentage of the row partitions is sampled from the Multinomial distribution $\mathcal{M}(1/G, \ldots, 1/G)$. Concretely, it means that at iteration q, with $q \leq I$, if a row-cluster does not have any element, a percentage of the rows of matrix $\boldsymbol{v}^{(q)}$ are erased, and randomly re-sampled. Similarly when a column-cluster becomes empty on the d^{th} matrix, a percentage of the column partitions is sampled from the multinomial distribution $\mathcal{M}(1/H_d, \ldots, 1/H_d)$. Therefore, if a column-cluster of the d^{th} matrix does not have any element at iteration q ($q \leq I$), a percentage of the rows of matrix $\boldsymbol{w}^{d^{(q)}}$ are erased, and randomly re-sampled.

Choice of the number of iterations. The SEM-algorithm can be slow to reach its stationary state. After having arbitrarily chosen the total number of iterations, the stability of the algorithm has to be checked. To accomplish that, the evolution of the parameters through the iterations can simply be graphically analyzed. If the parameters are "stable" between the burn-in period and the last iteration then the number of iterations was well chosen. Less subjective ways exist to evaluate if the stationary distribution has been achieved. The authors of [26] propose a general approach to monitoring convergence of Markov Chain Monte Carlo (MCMC) output in which parallel chains are run with starting values that are spread relative to the posterior distribution. Convergence is confirmed when the output from all chains is indistinguishable. This method is not used in this paper but could have been. Indeed, in Section 5, we show that we can obtain satisfactory results without this technique.

3.2. Model Selection

To select the number of blocks $(G, H_1, \ldots, H_D)$, a model selection criterion must be used. The most standard ones, like Bayesian Information Criterion (BIC) [27], rely on penalizing the maximum log-likelihood value $l(\hat{\boldsymbol{\theta}}; \check{\boldsymbol{x}})$. However, due to the dependency structure of the observed data $\check{\boldsymbol{x}}$, this value is not available.

Alternatively, an approximation of the ICL information criterion [3], called here ICL-BIC, can be invoked to overcome the previous problem due to the dependency structure in the missing variables $(\check{\boldsymbol{x}}, \boldsymbol{v}, \boldsymbol{w})$. The key point is that this latter vanishes since ICL relies on the completed latent block information $(\boldsymbol{v}, \boldsymbol{w})$, instead of integrating on it as it is the case in BIC. In particular, [28] detailed how to express ICL-BIC for the general case of categorical data. It is possible to straightforwardly transpose the ICL-BIC expression given by these authors by following their work step by step, with no new technical material. As proved in [23], the resulting MLBM-specific ICL-BIC is expressed by:

$$\text{ICL-BIC}(G, H_1, \ldots, H_D) =$$

$$\log p(\check{\boldsymbol{x}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{w}}; \hat{\boldsymbol{\theta}}) - \tfrac{1}{2}(G-1)\log N - \sum_d \tfrac{1}{2}(H_d - 1)\log J_d - \sum_d \tfrac{1}{2}\nu_d \log(N \times J_d),$$

where ν_d is the number of parameters to estimate for the d-th matrix $\boldsymbol{x}^d$. It will depend on G, H_d and the type of the variables of $\boldsymbol{x}^d$. Table 1 in Section 4 gives ν_d for each type of distribution.

In theory, to find the best number of blocks $(G, H_1, \ldots, H_D)$, the co-clustering has to be executed for each possible value and the result with the highest ICL-BIC has to be retained. Let n_G be the number of candidate values for G, while n_{H_d} is the number of candidate values for H_d, $d \in \{1, \ldots, D\}$. Thus, the number of co-clustering to execute is $n_G \times n_{H_1} \times \ldots \times n_{H_D}$. For example, if $D = 3$ and the user wants to try 10 values for G and for each H_d, then it would require execution of 10^4 co-clusterings. Depending on the data set, it might take too much time to find the best solution. In practice, a good set $(G, H_1, \ldots, H_D)$ is found using the following heuristic. Let (G_{min}) be the minimum of the candidate values for G. Then, $(H_{d_{min}})_d$ is the minimum of the candidate values for $(H_d)_d$. The algorithm starts with the set $(G_{min}, H_{1_{min}}, \ldots, H_{D_{min}})$. At iteration p, the current best set $(G, H_1, \ldots, H_D)$ is called $(G, H_1, \ldots, H_D)^{(p)}$ and is made of values: $(G^{(p)}, H_1^{(p)}, \ldots, H_D^{(p)})$. At the p^{th} iteration, $(D+1)$ co-clusterings are realized with sets $(G^{(p)}+1, H_1^{(p)}, \ldots, H_D^{(p)})$, $(G^{(p)}, H^{(p)}+1, \ldots, H_D^{(p)})$,$\ldots$,$(G^{(p)}, H_1^{(p)}, \ldots, H_D^{(p)}+1)$. Then, the ICL-BIC is computed for each result. If none of the ICL-BIC values are better than for the set $(G, H_1, \ldots, H_D)^{(p)}$, the algorithm finishes and $(G, H_1, \ldots, H_D)^{(p)}$ is the set to use. Otherwose, the set with the highest ICL-BIC is retained, and becomes $(G, H_1, \ldots, H_D)^{(p+1)}$. The algorithm then reiterates the same steps.

4. Modeling of the different types of data

Representing the data as a mathematical object is challenging and requires compromise. Often the user has to find a trade-off between information loss, interpretability and feasibility for their representation. The model described here can work with the following types of data: categorical data (nominal, ordinal, binary), count data, continuous data and document-term matrices. While the

probability distributions for nominal (Multinomial), binary (Bernoulli), count (Poisson) and continuous (Gaussian) data are widely accepted, several ways to model textual and ordinal data exist.

The simplest way to represent textual data is as a Document-Term count matrix where a cell counts how many times a term appears in a document. The Poisson distribution is a good distribution for modeling this matrix because it models the occurrences of an event (in this case, the appearances of a word). In a more advanced way, the Document-Term TF-IDF matrix, counts the times a term appears, but penalizes the result if this same term appears in the other documents [29]. The resulting score is continuous numeric which implies the usage of the Gaussian distribution. In the latter, the "stop-words" terms are discarded. However, even with the TF-IDF normalisation, the Gaussian distribution is not the best way to handle Document-Term matrices [30]. Lots of other Document-Term matrix types exist, and they have proven their efficiency in many applications [17, 6]. In this work, a simple Document-Term matrix representation is considered. When handling Document-Term matrix data only (and no other kind of data), diagonal LBM or equivalent approaches are more appropriate since the matrix is sparse [31, 32].

Ordinal data is also a sensitive data type. It may seem very easy to model them as if they were nominal, but doing that would spoil the order between the different levels, which is an intrinsic property of this type of data. In some applications, it can be interpreted as continuous [33] but in other cases it is not an option. For example, for clinical surveys, psychologists sometimes spend years defining ordinal scales on abstract concepts like pain, perception of control or anxiety [34, 35]; it is therefore difficult to project their results onto other scales or into a continuous space. In the present work, a recent distribution for ordinal data (BOS for Binary Ordinal Search model, [36]) is used. It has proven its efficiency for modeling and clustering ordinal data. The main advantages of the BOS model are its parsimony and the interpretability of its parameters.

This section describes the expression of the PDF $f(x_{ij}^d; \boldsymbol{\alpha}^{d^{(q-1)}})$ and the way to update $\boldsymbol{\alpha}^{d^{(q-1)}}$, in the SEM-Gibbs algorithm, depending on the type of the

matrix $\boldsymbol{x}^d$. The superscripts (q) and (d) are omitted to simplify the expressions.

4.1. Modeling nominal data

A nominal variable is a variable that can take on one of a limited, fixed, number of possible values. Each of the possible values of a categorical variable is referred to as a level. For a block (g,h) of nominal data, we consider the multinomial distribution $\mathcal{M}(1, \boldsymbol{\beta}_{gh})$, where $\boldsymbol{\beta}_{gh} = (\beta^r_{gh})_{r=1,\dots,m}$, and $\sum_r^m \beta^r_{gh} = 1$. Therefore, with this type of data, the MLBM block parameter α_{gh} is quoted as $\boldsymbol{\beta}_{gh}$, and the PDF is given by:

$$f(x_{ij}; \boldsymbol{\beta}_{gh}) = \prod_r^m (\beta^r_{gh})^{I(x_{ij}=r)},$$

where $I(x_{ij} = r) = 1$ if $x_{ij} = r$, and 0 otherwise. The update of each β^r_{gh} is:

$$\beta^r_{gh} = \tfrac{1}{n_{gh}} \sum_{i,j} v_{ig} w_{jh} I(x_{ij} = 1),$$

where n_{gh} is the number of elements belonging to block (g, h).

Firstly, note that if two nominal variables do not have the same number of levels m, then their distribution are not defined on the same support. Consequently, such variables should be separated into different matrices $\boldsymbol{x}^d$ of $\boldsymbol{x}$. Secondly, the co-clustering we propose is dependent on the order of the levels. For example two categorical features with $m = 3$ levels having respective parameters $\boldsymbol{\beta} = (0.1, 0.7, 0.2)$ and $\boldsymbol{\beta} = (0.7, 0.2, 0.1)$ won't be detected as two variables following the same distribution. Consequently they won't be grouped together in a similar column cluster, whereas a simple switch in the order of the levels could change this and lead to grouping these variables together. Note that this problem is not specific to co-clustering and is also present in clustering [37]. While the user should be aware that the results are conditional on the encoding of levels, this is not an issue addressed in this work.

4.2. Modeling ordinal data

Ordinal data is a special case of nominal data, where the order between the levels has a meaning. In the present work, the BOS model [36] is chosen to

model ordinal data. It is a probability distribution parametrized by a position parameter $\mu_{gh} \in \{1, \ldots, m\}$ and a precision parameter $\pi_{gh} \in [0, 1]$. This distribution has interesting properties from an interpretation standpoint: it rises from the uniform distribution when $\pi_{gh} = 0$ to a more peaked distribution around the mode μ_{gh} when π_{gh} increases, and it reaches a Dirac distribution at the mode μ_{gh} when $\pi_{gh} = 1$. It is shown in [36] that the BOS distribution is a polynomial function of π_{gh} with degree $m - 1$ whose coefficients depend on the position parameter μ_{gh}.

Therefore, with this type of data, the MLBM block parameter α_{gh} is quoted as (μ_{gh}, π_{gh}), and the PDF is given by:

$$f(x_{ij}; \mu_{gh}, \pi_{gh}) = \sum_{r=0}^{m-1} C_r(\mu_{gh}, x_{ij}) \pi_{gh}^r,$$

where $C_r(\mu_{gh}, x_{ij})$ is a constant depending on μ_{gh} and x_{ij}.

Since BOS inference relies on an EM-algorithm, the update of parameter (μ_{gh}, π_{gh}) is obtained through an EM-algorithm. For further details on this algorithm, see [36]. Similarly to the nominal variables case, if two ordinal variables do not have the same number of levels, they have to be separated into different matrices $\boldsymbol{x}^d$ of $\boldsymbol{x}$.

4.3. Modeling continuous data

In the continuous case, the unidimensional Gaussian distribution $\mathcal{N}(\mu_{gh}, \sigma_{gh}^2)$ is considered. Thus, the MLBM block parameter α_{gh} is here (μ_{gh}, σ_{gh}) and the PDF is given by:

$$f(x_{ij}; \mu_{gh}, \sigma_{gh}) = \exp\{\tfrac{-1}{2\sigma_{gh}^2}(x_{ij} - \mu_{gh})^2\} / \sqrt{2\pi\sigma_{gh}^2}.$$

The update of parameters $(\mu_{gh}, \sigma_{gh}^2)$ is:

$$\mu_{gh} = \frac{1}{n_{gh}} \sum_{i,j} v_{ig} w_{jh} x_{ij} \quad \text{and} \quad \sigma_{gh}^2 = \frac{1}{n_{gh}} \sum_{i,j} v_{ig} w_{jh} (x_{ij} - \mu_{gh})^2.$$

4.4. Modeling count data

Count variables are modeled by the Poisson distribution. For a block (g,h) of count data, a Poisson distribution with a specific parametrization is considered:

$\mathcal{P}(n_{i.}n_{.j}\delta_{gh})$, where $n_{i.} = \sum_j x_{ij}$ and $n_{.j} = \sum_i x_{ij}$ are the number of occurrences in row i and the number of occurrences in column j. The parameters $n_{i.}$ and $n_{.j}$ are independent of the co-clustering and are consequently preliminary estimated from the count data matrix. Consequently, the MLBM parameter α_{gh} are only the parameter δ_{gh}, which is the effect of the block (g, h) [38]. The PDF is given by:

$$f(x_{ij}; \delta_{gh}) = \frac{1}{x_{ij}!} e^{-n_{i.}n_{.j}\delta_{gh}} (n_{i.}n_{.j}\delta_{gh})^{x_{ij}}.$$

The update of each parameter δ_{gh} is obtained by:

$$\delta_{gh} = \frac{1}{n_{g.}n_{.h}} \sum_{i,j} v_{ig} w_{jh} x_{ij},$$

where $n_{g.} = \sum_{i,j} v_{ig} x_{ij}$ and $n_{.h} = \sum_{i,j} w_{jh} x_{ij}$.
Finally, Table 1 summarizes the number of parameters ν for each type of data described above.

Table 1: Number of parameters (ν) of the distribution properties

Data type	Distribution	α_{gh}	ν
Nominal	Multinomial	$\boldsymbol{\beta}_{gh} = (\beta^r_{gh})_{r=1,\ldots,m}$	$(m-1)GH$
Ordinal	BOS	(μ_{gh}, π_{gh})	$2GH$
Continuous	Gaussian	(μ_{gh}, σ_{gh})	$2GH$
Count	Poisson	$(\mu_i, \nu_j, \delta_{gh})$	GH

5. Numerical experiments on artificial data

This section has two goals. The first is to show that the proposed inference algorithm works appropriately. The second is to evaluate the model selection strategy: the efficiency of the ICL-BIC criterion in selecting the true numbers of clusters and the ability of the heuristic search to sparsely explore the space of numbers of clusters.

5.1. Simulation settings

Two simulation settings are considered. While they both have the same parameters, the first is built such that ($N = J_1 = J_2 = J_3 = J_4 = 100$), and the second is built with ($N = J_1 = J_2 = J_3 = J_4 = 500$).

Parameters setup. Both settings were simulated with four types of distribution: nominal (with $m = 5$ levels), continuous, ordinal (with $m = 3$ levels), and count data. The number of blocks was set to $(G, H_1, H_2, H_3, H_4) = (3, 3, 3, 3, 3)$. Furthermore, the mixing row proportions were $\boldsymbol{\gamma} = (0.2, 0.3, 0.5)$ and the mixing column proportions were equal to: $\boldsymbol{\rho}_1 = (0.25, 0.3, 0.45)$, $\boldsymbol{\rho}_2 = (0.2, 0.35, 0.45)$, $\boldsymbol{\rho}_3 = (0.25, 0.35, 0.4)$, $\boldsymbol{\rho}_4 = (0.25, 0.35, 0.4)$. Table 2 details the parameters that were assigned to each block.

Table 2: Value of block parameters. For the count data, parameters are not equal between the first and second simulation because they depend on the margins.

	Nominal $m = 5$ $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$		
	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	0.05,0.05,0.8,0.05,0.05	0.1,0.25,0.3,0.3,0.05	0.1,0.2,0.4,0.2,0.1
row-cluster 2	0.05,0.1,0.7,0.1,0.05	0.8,0.05,0.05,0.05,0.05	0.4,0.05,0.1,0.05,0.4
row-cluster 3	0.2,0.5,0.2,0.05,0.05	0.8,0.05,0.05,0.05,0.05	0.05,0.8,0.05,0.05,0.05

	Continuous μ, σ			Ordinal $m = 5$ μ, π		
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	100,1	0.5,5	-90,5	3,0.4	1,0.2	3,0.7
row-cluster 2	10,4	-15,1	-95,1	2,0.1	3,0.5	2,0.8
row-cluster 3	-20,1	-30,3	500,4	2,0.5	1,0.8	2,0.2

	Count 100 $\delta \times 10^{-5}$			Count 500 $\delta \times 10^{-7}$		
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	1.2	5.5	1.2	4.6	20.5	4.9
row-cluster 2	8.3	5.5	0.5	30.0	20.5	1.6
row-cluster 3	1.3	1.3	3.5	5.5	5.6	14.5

Experimental setup. For both settings, 20 data sets are simulated, and the same process is run through. Firstly, the co-clustering is performed on the 20 data sets with the true numbers of clusters $(G_1, H_1, \ldots, H_D)$ and the correctness of the parameter estimation is evaluated. Then, we assess the efficiency of the ICL-BIC criteria by using an exhaustive search among possible values for the number of clusters. In order to reduce the number of ICL-BIC values to compute, we consider only the number of clusters obtained by adding or removing one to each of the element of the true $(G, H_1, \ldots, H_D)$. Therefore, for each simulation, $3^5 = 243$ co-clusterings are executed, because 3 values are tested for G,H_1,H_2,H_3 and H_4. Then, the set $(G, H_1, \ldots, H_D)$ with the best ICL-BIC value is retained. Afterward, the heuristic search from Section 3.2 is evaluated. In this case, the number of co-clusterings to be performed is not fixed because the algorithm stops once it can't find a better ICL-BIC value.

Choice for the number of iterations. The number of iterations for the SEM-Gibbs algorithm was set to 150 and the burn-in period was considered to take 100 iterations. To check if this number of iterations is enough, the evolution of the parameters is graphically observed, as in Figure 2. Here, only a few parameters are represented as an example, but it is useful to check for several parameters. We notice in this example that some of the parameters reached their stationary state from the beginning of the algorithm, and that other parameters needed 50 to 100 iterations to get stable. Therefore, in order to ensure that all parameters have achieved their stationary distribution, a burn-in period of 100 iterations is considered (over a total number of 150 iterations). Numerical results in Section 5.2 show that these numbers of iterations are large enough since the parameters are well estimated with this particular setting.

Choices for initialization. The number I corresponds to the number of iterations a certain percentage of the partitions are randomly sampled when a cluster becomes empty, as explained in Section 3.1. Here, I is tuned to be equal to the number of iterations for burn-in, while the percentage value was fixed to 20.

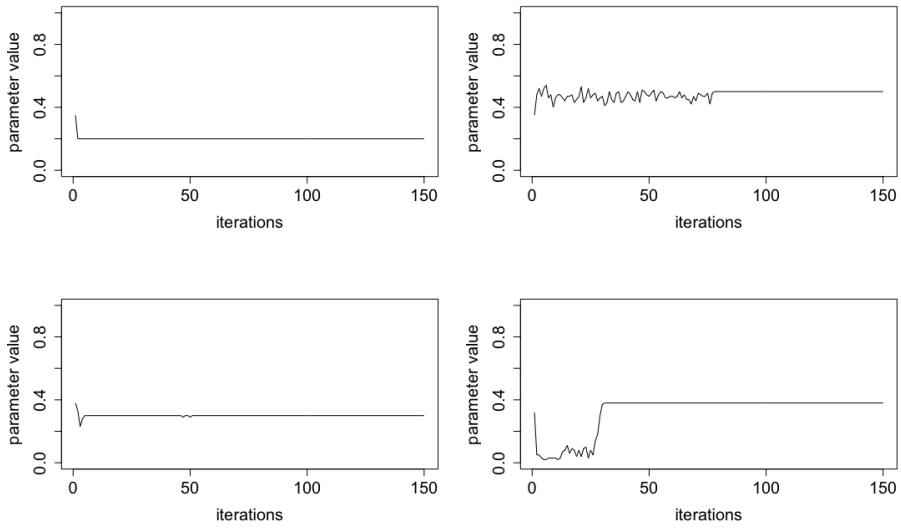


Figure 2: Evolution of parameters $\boldsymbol{\rho}$ through the SEM-Gibbs algorithm iterations. From left to right, and from top to bottom, the graph represents the evolution of the first element of each vector $\boldsymbol{\rho}_1$, $\boldsymbol{\rho}_2$, $\boldsymbol{\rho}_3$ and $\boldsymbol{\rho}_4$.

5.2. Parameter and partition estimation

5.2.1. Parameter estimation

The co-clustering was performed on 20 data sets, with the true numbers of clusters. The mean absolute errors for the mixing proportions are shown in Figure 3 and Figure 4. The mean absolute errors between the parameter values and their estimation are given in Table 3 and Table 4 for the continuous, ordinal and count data. For the nominal data, all the mean absolute errors were less than 0.01. These errors are extremely low, which means that the model parameters are correctly estimated.

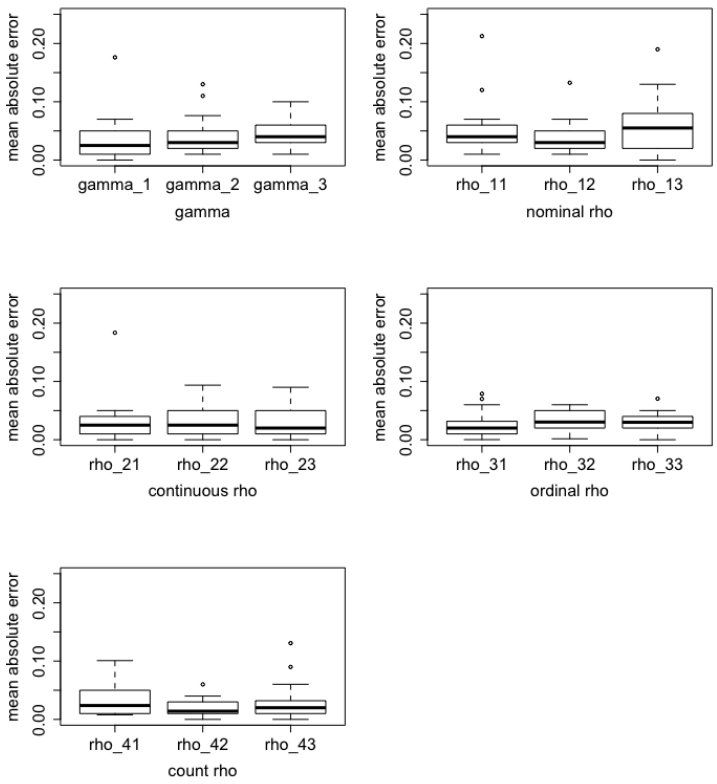


Figure 3: Mean absolute error for the mixing proportions with $N = J_d = 100$.

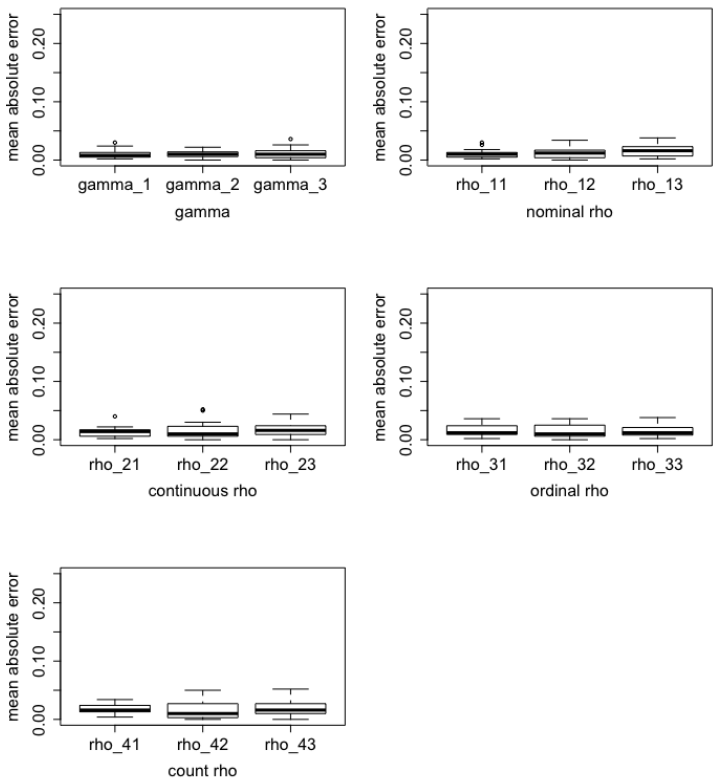


Figure 4: Mean absolute error for the mixing proportions with $N = J_d = 500$.

Table 3: Value of the block parameters mean absolute error on simulation with $N = J_d = 100$ for the continuous, ordinal and count matrices.

	Continuous μ, σ			Ordinal $m = 5$ μ, π			Count $\delta \times 10^{-5}$		
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	0.01,0.01	0.03,0.02	0.02,0.02	0.00,0.05	0.00,0.03	0.00,0.05	0.16	1.89	1.33
row-cluster 2	0.04,0.02	0.00,0.00	0.00,0.00	0.00,0.04	0.00,0.03	0.00,0.05	0.87	1.97	1.4
row-cluster 3	0.00,0.00	0.01,0.01	0.01,0.01	0.00,0.02	0.00,0.03	0.00,0.03	0.34	0.83	1.06

Table 4: Value of the blocks parameters mean absolute error on simulation with $N = J_d = 500$ for the continuous, ordinal and count matrices.

	Continuous μ, σ			Ordinal $m = 5$ μ, π			Count $\delta \times 10^{-7}$		
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	0.2,0.03	0.3,0.09	0.1,0.08	0.00,0.04	0.00,0.03	0.00,0.01	0.1	0.2	0.1
row-cluster 2	0.01,0.02	0.1,0.1	0.1,0.06	0.00,0.02	0.00,0.03	0.00,0.04	0.5	0.1	0.1
row-cluster 3	0.00,0.00	0.01,0.01	0.01,0.01	0.00,0.03	0.00,0.02	0.00,0.03	0.3	0.2	0.3

5.2.2. Partition estimation

The partition estimation is assessed using the Adjusted Rand Index, referred to as "ARI" [39]. The ARIs for the row and column partitions, on the two simulated data sets are given in Table 5. We see that the co-clustering algorithm succeeds in finding the true partitions for the rows and columns.

Table 5: Mean (standard deviation) ARIs for two data sets $N = 100$ and $N = 500$.

N	Rows	Categorical	Continuous	Ordinal	Count
100	0.98 (0.09)	0.95 (0.14)	0.98 (0.07)	1 (0.01)	0.98 (0.09)
500	1 (0.00)	1 (0.00)	1 (0.00)	1 (0.00)	1 (0.00)

5.3. Model selection

In this section, the ICL-BIC criterion's efficiency is assessed for choosing the right number of clusters by row and by column. Furthermore, the heuristic search described in 3.2 is evaluated. The complexity of the problem should be emphasized here. Usually, criteria such as BIC or ICL are used to find the right number of clusters for the row partitions only. In the case of co-clustering, they

Table 6: Exhaustive search results on 20 simulations results.

$N = J_d = 100$									
(G, H_1, H_2, H_3, H_4)	34333	34334	43333	**33333**	44333	34433	34443	44334	44433
number of occurrences	6	3	3	2	2	1	1	1	1

$N = J_d = 500$										
(G, H_1, H_2, H_3, H_4)	**33333**	33332	33323	43333	34342	32323	33343	34332	34322	44332
number of occurrences	6	3	2	2	2	1	1	1	1	1

are extended to find the right number of clusters for the row partitions and the column partitions. In the present work, it is used to find $(D + 1)$ numbers of clusters (one for the rows, and one for each kind of feature). Mathematically, the search space is much larger which makes the problem more complex.

Exhaustive Search. Table 6 presents which sets (G, H_1, H_2, H_3, H_4) had the best ICL-BIC value in the exhaustive search. The number of occurrences indicates how many times the sets were chosen. Note that the right numbers of clusters $(G, H_1, H_2, H_3, H_4) = (3, 3, 3, 3, 3)$ has been chosen more often for the larger data set with $(N = J_d = 500)$. This result is consistent because the proposed ICL-BIC is based on asymptotic approximations. For the data set with $(N = J_d = 100)$, the model with $(G, H_1, H_2, H_3, H_4) = (3, 3, 3, 3, 3)$ is only the fourth one to be chosen. However, the average means of the ARI for the co-clustering when the model chosen was $(G, H_1, H_2, H_3, H_4) \neq (3, 3, 3, 3, 3)$, are equal to (0.94 (0.10), 0.93 (0.08), 0.98 (0.02), 0.98 (0.02), 0.98 (0.02)). This means that when the criterion for model selection does not find the true model, the algorithm still finds good partitions.

Heuristic search. Table 7 presents which sets (G, H_1, H_2, H_3, H_4) were chosen by the heuristic search. Once again, the algorithm works better for the larger data set with $(N = J_d = 500)$, although the results for $(N = J_d = 100)$ are good too.

Search computation time. The simulations were run using a Linux 4.9.0-3-amd64 server, on Debian 9. For the data set with $N = J_d = 100$, the exhaustive

Table 7: Heuristic search results on 20 simulations results.

$N = J_d = 100$												
(G, H_1, H_2, H_3, H_4)	**33333**	22223	22233	22234	22243	22244	22334	23223	32223	33334	34334	37334
number of occurrences	5	3	3	1	1	1	1	1	1	1	1	1

$N = J_d = 500$					
(G, H_1, H_2, H_3, H_4)	**33333**	32233	22333	23234	32333
number of occurrences	10	5	3	1	1

search took 23 minutes, while the heuristic search took at most 18 minutes. For the data set with $N = J_d = 500$, the exhaustive search lasted 33 hours whereas the heuristic search took at most 2 hours. This means that in the case of a small data set, it can be interesting to run an exhaustive search, as it does not take much more time than the heuristic search. However, an exhaustive search as it was realized in this simulation requires knowledge of the neighborhood of the right set (G, H_1, H_2, H_3, H_4). For a larger data set, the heuristic search is recommended as it is very efficient and up to 15 times faster than the exhaustive search. Furthermore, we can expect it to be even more than 15 times faster in case of larger data sets than the ones used in these simulations.

5.4. More challenging data sets

In Section 5.1 the parameter settings for the continuous variables generate well separated clusters since the means are separate and the variances small. Besides, the optimal ARIs (ARIs obtained while knowing the parameters) in line and in column were always equal to 1. In this section, we change these parameters so that the clusters are not well separated with regard to the continuous variables. We used the simulated data set of Section 5.1 with 100 rows and 400 columns and changed the parameters of the continuous variables. In each block of the diagonal, we have $\mu = \epsilon$ and $\sigma = 1$. On the other blocks, $\mu = 0$ and $\sigma = 1$ (see Table 8). We performed the co-clustering algorithm 20 times for ϵ equals to 0.5 and 0.2. Then, we performed the co-clustering 20 times with the data set made of the continuous variables only. The optimal

ARIs of the data set and the ARIs resulting from the co-clustering are given in Table 9. In this simulation, we see that when the co-clustering algorithm

Table 8: Mean and Standard deviation for the blocks of continuous variables for the more challenging data sets cases.

	col-cluster 1	col-cluster 2	col-cluster 3
row-cluster 1	ϵ,1	0,1	0,1
row-cluster 2	0,1	ϵ,1	0,1
row-cluster 3	0,1	0,1	ϵ,1

Table 9: ARIs for the more challenging data set case.

		optimal ARI	Rows ARI	Categorical ARI	Continuous ARI	Ordinal ARI	Count ARI
$\epsilon = 0.5$	all variables	(1,1,0.89,1,1)	0.96 (0.1)	1 (0)	0.85 (0.11)	0.98 (0.09)	0.96 (0.11)
	only continuous	(0.86,0.90)	0.69 (0.2)	-	0.7 (0.16)	-	-
$\epsilon = 0.2$	all variables	(1,1,0.28,1,1)	1 (0)	0.94 (0.17)	0.19 (0.12)	0.94 (0.16)	1 (0)
	only continuous	(0.14,0.28)	0 (0.03)	-	0 (0.03)	-	-

is performed only on the continuous variables, it does not distinguish the different blocks well. Indeed, the row-clusters are too mixed. However, when the other variables (categorical, ordinal and counting) are taken into account, the co-clustering succeeds in finding the true partitions. In addition, the good estimation of the row partitions obtained thanks to the non-continuous variables improves the column partitions estimation for the continuous variables.

5.5. Missing data

In this section, we investigate the behavior of the ARIs when missing values are introduced into the data. Again, we used the data set with 100 rows and 400 columns. We performed the co-clustering algorithm 20 times on the data set with 10%, 20%, 30%, 50% and 75% of missing values. Resulting ARIs for row and column partitions are given in Table 10. We see that up to 30% of missing values, the ARI does not changes significantly. However, with more missing values, the ARI for the partitions reduces.

Table 10: ARIs for a data set with missing values.

ARI type	Rows	Categorical	Continuous	Ordinal	Counting
Original Simulation	0.98 (0.09)	0.95 (0.14)	0.98 (0.07)	1 (0.01)	0.98 (0.09)
10% NA	1 (0)	1 (0)	0.87 (0.2)	1 (0.01)	1 (0.01)
20% NA	1 (0)	1 (0)	0.88 (0.21)	0.99 (0.01)	0.99 (0.02)
30% NA	0.99 (0.04)	0.98 (0.1)	0.98 (0.07)	0.94 (0.14)	0.87 (0.14)
50% NA	0.59 (0.08)	0.99 (0.01)	0.98 (0.07)	0.93 (0.11)	0.76 (0.18)
75% NA	0.23 (0.13)	0.71 (0.07)	0.77 (0.14)	0.46 (0.08)	0.38 (0.2)

5.6. Conclusion

As a conclusion for this simulation study, the SEM-Gibbs algorithm is efficient in estimating the model parameters and the partitions. Regarding model selection, while we know that the ICL-BIC criterion leads to a consistent estimation of the number of blocks when the number of rows and column tends to infinity (see [28]), its behavior for finite sample size remains robust. Moreover, using the proposed heuristic search enables drastic reduction in computing time without significantly decreasing the performance of the estimation.

When the continuous variables have poorly separated parameters, the co-clustering succeeds in finding the true row partitions and the true column partitions of the other variables.

When up to 30% of missing values are introduced, the co-clustering succeeds in finding the true row and column partitions. When there are more than 30% of missing values, it is more difficult for the co-clustering to find the true partitions.

6. Real data applications

In this section, two real data sets are considered. The first one concerns the famous TED talks[1] and contains the transcripts and ratings of TED Talks uploaded to the official TED.com website until September 21st, 2017. It is a mixed data set because the transcripts are textual data whereas the ratings are

[1] `https://www.ted.com/talks`

numbers. The second data set is the result of a survey that Slovakian Statistic students gave to people around them. The responses were categorical with different numbers of levels and some of them were ordinal.

6.1. Co-clustering of count and continuous data

The TED talks data set. TED is a non-profit organization which posts conferences on-line for free distribution. The conferences address a wide range of topics, including science, culture and innovation. The TED talks data set[2] contains information about 2 467 TED Talks. This work is focused on their transcripts and their ratings given by the users. The rating system is particular on this website. A list of fourteen words was defined (beautiful, inspiring, persuasive, fascinating, ok, longwinded, confusing, informative, courageous, ingenious, funny, obnoxious, unconvincing, jaw-dropping). A user wanting to rate a talk is asked to choose the three words that best describe the talk.

Data set pre-processing. First of all, a couple of TED talks were actually a musical performance. Their transcripts were of the form "(Applause)(Music)(Applause)", which is informationless, so these talks were removed from the data set. Then, the other transcripts were projected into a Document-Term matrix, each cell counting the occurrences of a term in a talk. It appears that some terms that occurred only once were onomatopoeia such as "aargh" and "aaaaaaaargh". These terms were removed: we assumed that these words do not bring valuable information, and that at the same time, removing them reduces the dimension of the matrix. The ratings variables were used without any changes: no normalization was performed as a pre-processing. In contrast with the Document-Term matrix, the ratings matrix is not sparse since only 1% of the values are equal to 0. The mean and standard deviation of the ratings matrix are equal to 175.2 and 538.2 respectively. The resulting matrix is therefore of dimension $(2\ 464 \times (40\ 137 + 14))$, in other words, $N = 2\ 464$, $J_1 = 40\ 137$ $J_2 = 14$. The

[2] `https://www.kaggle.com/rounakbanik/ted-talks/data`

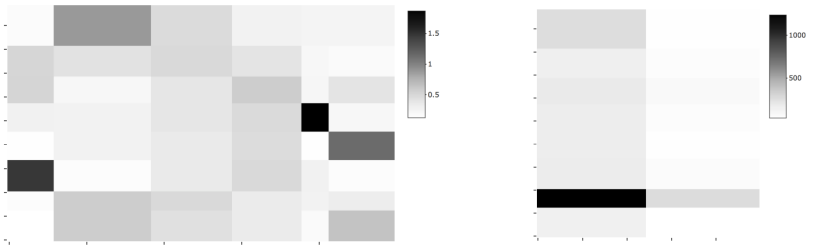
data set is seen as two matrices of different types ($D = 2$). The first one is the Document-Term matrix of the transcripts whose occurrences are modeled by a Poisson distribution. The second matrix represents, for each talk, the number of users that voted for each of the words in the proposed adjectives list. Given the high number of votes, this number is modeled by a Normal distribution. This matrix could be seen as a counting matrix as well. However, the proposed Poisson model takes into account margins on rows and columns (see $n_{i\cdot}$ and $n_{\cdot j}$ in Section 4.4). These margins make sense on document-term matrices. However, on the rating matrix of this application, they are not as relevant. Furthermore, the Gaussian distribution is more suitable because with a Poisson distribution, the mean is equal to the variance: over large numbers like those in the rating matrix, the Poisson parameters are less informative.

Co-clustering as a parsimonious clustering. The main motivation on this data set is to cluster the TED talks to distinguish the different kinds of talks, and to observe the ratings of each row-cluster. Using a classical clustering technique is not conceivable because of the high dimension of the data set. The latent class model, for example, would define a distribution for each of the 40 151 variables and for each class, which is definitely over-parameterized and not interpretable. With a co-clustering technique not only will the talks be clustered, but the variables will be clustered as well, which will result in a small number of interpretable blocks.

Co-clustering results. After having searched for the highest ICL-BIC as explained in Section 3.2 with $(G_{min}, H_{1_{min}}, H_{2_{min}}) = (2, 2, 1)$, the best set (G, H_1, H_2) was found to be equal to $(8, 6, 2)$. Figure 5 gives a representation of the block parameters. For the Document-Term matrix, the δ parameters are represented by shades of gray. The lighter the block, the lower its corresponding δ parameter. When a block's δ parameter is high, this means that the column-cluster terms of this block are quite specific to the corresponding row-cluster. For the ratings matrix, the shades of gray represent the μ parameter of the resulting blocks. The darker the block, the higher the μ parameter.

First of all, we focus on the row-clusters of the document-term-matrix. Note from the titles of talks with the same row-cluster number that the co-clustering grouped talks with similar topics. For example the third group seems to be about high technology and science with titles such as "A robot that runs and swims like a salamander", "A mobile fridge for vaccines" and "The hunt for a supermassive black hole"; whereas the fourth group refers to politics, with talks called as "Why Brexit happened – and what to do next", "How ideas trump crises" , and "Aid for Africa? No thanks.". From the ratings row-cluster parameters, it can be seen that the seventh row-cluster's talks were rated about ten times more than the documents of the other row-clusters. It is interesting to observe that this corresponds to a row-cluster closely related to psychology and introspection. Table 11 gives an overview of the Document-Term matrix row-clusters, giving some titles and the topic that was deduced from them. On the other hand, two row-clusters were more difficult to interpret. For example, the eighth row-cluster gathers talks with titles such as "Dare to educate Afghan girls", "Averting the climate crisis", "Fighting with nonviolence" and "What it's like to be a parent in a war zone". While the talks tend to be about education and parenting, the inherent topic is not obvious nor unique. The same issue was observed with the third row-cluster: with titles such as "The magic of Fibonacci numbers", "A new equation for intelligence" and "New thinking on the climate crisis", it is hard to define a unique subject for this group.

It is not easy to interpret directly the terms clusters because these column-clusters contain on-average about 6 000 variables. However, we have extracted some of the 100 most frequent words for some notable blocks with high δ parameters to check if they are relevant to the row-clusters' topics of Table 11. Firstly, from Figure 5a, block $(6, 1)$,corresponding to the 6^{th} row-cluster and 1^{st} column cluster, was noted. Among the most frequent words are "knowledge", "future", "company", "information", "community", "working", and"imagine", which are relevant to the 6^{th} row-cluster topic about innovation and high-technology. Similarly, block $(4, 5)$ was noted. Some of the most frequent terms are "phenomenon", "coffee", "discovery", "organisms", and "suffering", which



(a) (b)

Figure 5: Block representation of the Document-Term matrix (left) and of the ratings matrix (right). The shades of gray represent the δ parameter of each block for the Document-Term matrix. For the rating matrix, they represent the μ parameter.

correspond to the 4^{th} row-cluster topic about medicine and health. Finally, block $(5, 6)$ was investigated. It appears that its column-cluster terms are specific to the 5^{th} row-cluster about politics, with the words "india", "history", "technology", "program", and "impact" among the most frequent ones.

We now consider the column-clusters of the ratings for the TED talks. The adjectives were split into two groups. The first column cluster is composed of the following adjectives: "Inspiring", "Beautiful", "Courageous", "Persuasive", "Fascinating", "Informative", and "Funny". These adjectives were on average voted for more than those of the second column-cluster, and this for all the row-clusters. The second column-cluster is made up of the adjectives "Ingenious", "Confusing", "Jaw-dropping" "Obnoxious", "Longwinded", "Unconvincing",and "OK".

From these observations, we can conclude that the co-clustering results helped provide understanding and a summary of the data set. Firstly, it clustered the TED-talks documents. The resulting classes were relevant regarding the titles topics and corresponding term column-clusters. Furthermore, the rating matrix gives information about the kinds of talks preferred. Overall, the co-clustering results gave an overview of a big data set that cannot be done easily by a human.

Table 11: Row-cluster interpretation for the TED talks data set.

Row-cluster number	Example titles	Interpreted topics
1	"My year of living biblically", "My journey from Marine to actor", "How I'm preparing to get Alzheimer's", "12 truths I learned from life and writing", "The year I was homeless"	Story-telling
2	"Art that craves your attention", "Building a museum of museums on the web", "How to engineer a viral music video", "A one-man orchestra of the imagination", "Moving sculpture"	Art, Culture
3	"The magic of Fibonacci numbers", "How behavioral science can lower your energy bill", "New thinking on the climate crisis", "A new equation for intelligence", "Winning the oil endgame"	Energy, Climate, Mathematics
4	"A map of the brain", "Your brain hallucinates your conscious reality", "Is anatomy destiny?", "Growing new organs", "A doctor's case for medical marijuana"	Medicine, Health
5	"Why Brexit happened – and what to do next", "How ideas trump crises", "Aid for Africa? No thanks.", "The surprising way groups like ISIS stay in power", "The attitudes that sparked Arab Spring"	Politics
6	"A robot that runs and swims like a salamander", "A mobile fridge for vaccines", "The hunt for a supermassive black hole", "Hands-on science with squishy circuits", "How we'll find life on other planets"	High technology, Science, Innovation
7	"Who are you, really? The puzzle of personality", "How to succeed? Get more sleep", "Your body language may shape who you are", "A kinder, gentler philosophy of success", "What really matters at the end of life"	Psychology, Introspection
8	"What it's like to be a parent in a war zone", "Teachers need real feedback", "Averting the climate crisis", "Dare to educate Afghan girls", "Fighting with nonviolence"	Education, Crisis

Co-clustering of each set separately. In this section we perform the co-clustering algorithm on the Document-Term matrix and on the ratings matrix separately. We also compare these results with the co-clustering performed with both matrices thanks to the MLBM. On the Document-Term matrix, the row and column partitions ARIs were on average equal to 0.36 (0.04) and 0.30 (0.04). On the ratings matrix, the row and column partitions ARIs were on average equal to 0.04 (0.00) and 0.27 (0.16). This means that the co-clusters obtained with separate matrices are very different from the co-clusters obtained using the MLBM. In particular, co-clustering of each set separately is not relevant to providing a unified row partition.

6.2. Co-clustering of ordinal and nominal data

Young People Responses to questionnaires. In 2013, Slovakian students of a statistics class were asked to invite their friends to participate in a survey that concerned several aspects of their life [3]. The responses were defined on different scales; for example, a question such as "I enjoy listening to music." could be answered from 1 ("Don't enjoy at all") to 5 ("Enjoy very much"). The questions regarding music preferences, movie preferences, hobbies and interests, spending habits and phobias are seen as 5 levels ordinal data, not only because the answers are on a scale, but also because two answers can be compared. For example, questions concerning the music preferences could be : "I enjoy classical music." or "I enjoy rock music.", and both could have a reply on a scale from 1 ("Don't enjoy at all") to 5 ("Enjoy very much"). In this case, the order in the responses is clear, and one can easily compare the two answers of a same user. However, in the case of personality traits, views on life and opinion, questions could be: "I have to be well prepared before public speaking."or "I always keep my promises.", still on a 5 level scale from 1 ("Strongly disagree") to 5 ("Strongly agree"). The order of the responses can not be compared, so considering them to be ordinal makes their interpretation too arbitrary. That is why these questions

[3] `https://www.kaggle.com/cardot/se-young-people-survey/data`

were considered to be categorical variables, with a number of levels equal to 5. Furthermore, demographic questions such as "What is my gender?", with responses "Female" and "Male" are modeled as categorical variables with 2 levels. This survey was completed by 1 010 people.

Thus, the resulting matrix is of dimension $(1\ 010 \times (80 + 5 + 54))$, so $N = 1\ 010$, $J_1 = 80$, $J_2 = 5$ and $J_3 = 54$. The data set is seen as three matrices of different types. The first contains the 80 questions with answers considered as ordinal, with 5 levels. The second contains the 5 questions with answers considered as nominal, with 2 levels. Finally, the third contains the 54 questions with answers considered as nominal, with 5 levels.

Finally, the data set had a small amount of missing data (0.4%), which will be estimated using the SEM-Gibbs algorithm as described in Section 3.

Co-clustering results. The SEM-Gibbs algorithm used 150 iterations and the burn-in period was set at 100 iterations. These numbers were defined using the same technique as in Section 6.1, by checking the evolution of several parameters through the SEM-Gibbs iterations. The best set (G, H_1, H_2, H_3) was found to be equal to $(3, 4, 2, 4)$. Figure 6 shows the resulting co-clustering, and Table 12 gives the estimated parameters of each block.

First of all, we notice that the first row-cluster has the lowest position parameter μ on the first column cluster of ordinal data. This means that people from this group have less overall enjoyment – or are less interested in, or are less afraid of – the topics of this column cluster's questions. These topics included classical music, branded clothing, psychology, politics and dangerous dogs. In addition, the parameters show that this row-cluster is quite heterogeneous. This row-cluster has the lowest position parameters π on the two first column-clusters of ordinal data, and they systematically have the highest β_1 and β_5 on categorical data with 5 levels. We will now consider the second row-cluster. We notice that it has a β_3 parameter equal to 0.5 on the personality questions first column-clusters, which is high. It means that people from this row-cluster are quite indecisive about the topics of these column-clusters. The questions in-

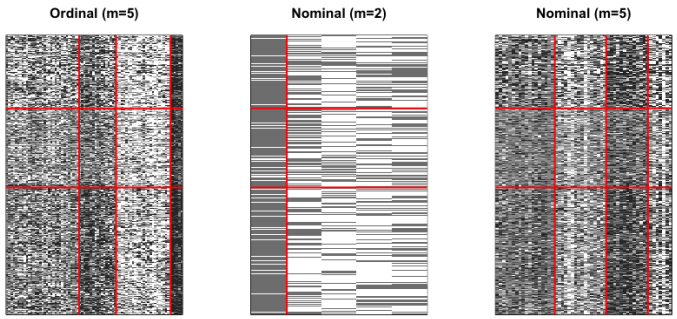


Figure 6: Co-clustering result on Young People Survey.

cluded "I am 100% happy with my life.", "I believe all my personality traits are positive.", "I have lot of friends.", and "My moods change quickly.".

Finally, we analyse the fourth column-cluster of the ordinal variables. We notice that it has the highest position parameter for all the row-clusters with $\mu = 5$. The questions of this column-cluster are: "I enjoy listening to music", "I enjoy watching movies", "I enjoy comedies", "I am interested in internet", and "I am interested in socializing". It means that the interviewed people are in overall agreement about being very interested in these topics.

Table 12: Resulting co-clustering parameters for the student survey data set.

	Ordinal ($m = 5$) μ, π				Nominal ($m = 2$) β_1, β_2	
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 4	col-cluster 1	col-cluster 2
row-cluster1	1,0.08	5,0.16	1,0.41	5,0.69	0.1,0.9	0.63,0.37
row-cluster 2	3,0.25	3,0.21	1,0.31	5,0.52	0.11,0.89	0.62,0.38
row-cluster 3	3,0.14	5,0.28	1,0.24	5,0.74	0.1,0.9	0.7,0.3

	Nominal ($m = 5$) $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$			
	col-cluster 1	col-cluster 2	col-cluster 3	col-cluster 4
row-cluster1	0.10,0.13,0.32,0.2,0.25	0.3,0.15,0.25,0.12,0.18	0.10,0.10,0.19,0.16,0.45	0.39,0.10,0.15,0.09,0.27
row-cluster 2	0.02,0.14,0.5,0.28,0.06	0.13,0.29,0.38,0.16,0.04	0.02,0.13,0.34,0.33,0.18	0.23,0.23,0.26,0.16,0.12
row-cluster 3	0.03,0.11,0.36,0.35,0.15	0.16,0.25,0.31,0.18,0.10	0.03,0.08,0.20,0.31,0.38	0.24,0.16,0.19,0.18,0.23

7. Conclusion

This work presents a model-based co-clustering model for data sets made of mixed type data. It relies on the latent block model and inference is performed using an SEM-Gibbs algorithm. The method has the great advantage of having an efficient criterion to select the number of row and column clusters. Furthermore, the parameters that are estimated on each block allow the user to easily interpret the partitions. Finally, missing data is handled, which is often useful in the case of real data sets. The efficiency of the algorithm was illustrated on a simulated data set and then on real data. An R package implemented using with C++ is available upon request to the authors. Moreover, if a user is interested in clustering the observations, the co-clustering algorithm proposed gives a parsimonious way to do this, by grouping all the features into a small number of clusters.

The proposed model has certain limitations. A major issue is that the variables of different types cannot be part of the same column-cluster as the model is based on the assumption that the elements of a same block share the same distribution. It would be interesting to find an approach to overcome this limitation. Furthermore, as noted in Section 4.1, the way the data is encoded can have a strong impact on the resulting co-clustering partition. Although there are ways to address the matter in some cases, as detailed in [37], the user should be aware of it. Additionally, the influence of each kind of feature on the resulting row partitions is to be investigated more deeply in a future work. Indeed, certain types of data will have more impact on the probability for a row belonging to a particular row-cluster, even if the D matrices have the same number of features J_d. Also, the case where the J_d are not highly unbalanced should be studied. An interesting approach could be to give the same importance to the D matrices, even if they do not have the same number of features. Finally, the way nominal and ordinal variables are modeled can raise the dimensionality of the problem. When the number of nominal and/or ordinal variables with differing levels increases, the number of sets x^d increases. However, the num-

ber of parameters will not significantly increase, because the proposed model is very parsimonious. In addition, even though it may significantly increase the number of competing models, the negative impact on the model selection process time will be limited thanks to the heuristic search procedure introduced in Section 3.2.

References

[1] N. D. Buono, G. Pio, Non-negative matrix tri-factorization for co-clustering: An analysis of the block matrix, Information Sciences 301 (2015) 13 – 26.

[2] G. Govaert, M. Nadif, Co-Clustering, Computing Engineering series, ISTE-Wiley, 2013.

[3] C. Biernacki, G. Celeux, G. Govaert, Assessing a mixture model for clustering with the integrated completed likelihood, IEEE Transactions on Pattern Analysis and Machine Intelligence. 22 (7) (2000) 719–725.

[4] M. Nadif, G. Govaert, Algorithms for model-based block gaussian clustering, in: DMIN'08, the 2008 International Conference on Data Mining, Las Vegas, Nevada, USA, 2008.

[5] P. Singh Bhatia, S. Iovleff, G. Govaert, blockcluster: An R package for model-based co-clustering, Journal of Statistical Software 76 (9) (2017) 1–24.

[6] C. Laclau, M. Nadif, Diagonal latent block model for binary data, Statistics and Computing 27 (5) (2017) 1145–1163.

[7] J. Jacques, C. Biernacki, Model-based co-clustering for ordinal data, Computational Statistics & Data Analysis 123 (C) (2018) 101–115.

[8] Y. B. Slimen, S. Allio, J. Jacques, Model-based co-clustering for functional data, Neurocomputing 291 (2018) 97 – 108.

[9] C. Bouveyron, L. Bozzi, J. Jacques, F. Jollois, The functional latent block model for the co-clustering of electricity consumption curves, Journal of the Royal Statistical Society: Series C (Applied Statistics) 67 (4) (2018) 897–915.

[10] B. S. Everitt, Introduction to Latent Variable Models, Chapman and Hall, 1984.

[11] C. Biernacki, T. Deregnaucourt, V. Kubicki, Model-based clustering with mixed/missing data using the new software MixtComp, in: CMStatistics 2015 (ERCIM 2015), London, United Kingdom, 2015.

[12] M. Marbac, C. Biernacki, V. Vandewalle, Model-based clustering of gaussian copulas for mixed data, Communications in Statistics - Theory and Methods 46 (23).

[13] D. McParland, I. Gormley, Model based clustering for mixed data: Clustmd, Adv. Data Anal. Classif. 10 (2) (2016) 155–169.

[14] C. Bouveyron, M. Fauvel, S. Girard, Kernel discriminant analysis and clustering with parsimonious gaussian process models, Statistics and Computing 25 (6) (2015) 1143–1162.

[15] A. K. Smilde, J. A. Westerhuis, S. d. Jong, A framework for sequential multiblock component methods, Journal of Chemometrics 17 (6) (2003) 323–337.

[16] A. Bouchareb, M. Boullé, F. Rossi, Co-clustering de données mixtes à base des modèles de mélange, in: Actes de la 17ème Conférence Internationale Francophone sur l'Extraction et gestion des connaissances (EGC'2017), Grenoble, France, 2017, pp. 141–152.

[17] M. Ailem, F. Role, M. Nadif, Sparse poisson latent block model for document clustering, IEEE Transactions on Knowledge and Data Engineering 29 (7) (2017) 1563–1576.

[18] D. McParland, C. M. Phillips, L. Brennan, H. M. Roche, I. C. Gormley, Clustering high-dimensional mixed data to uncover sub-phenotypes: joint analysis of phenotypic and genotypic data, Statistics in Medicine 36 (28) (2017) 4548–4569.

[19] G. Celeux, D. Chauveau, J. Diebolt, Some stochastic versions of the em algorithm, Journal of Statistical Computation and Simulation 55 (1996) 287–314.

[20] M. Selosse, J. Jacques, C. Biernacki, F. Cousson-Gélie, Analysing a quality-of-life survey by using a coclustering model for ordinal data and some dynamic implications, Journal of the Royal Statistical Society: Series C (Applied Statistics) in press.

[21] R. J. A. Little, D. B. Rubin, Statistical Analysis with Missing Data, John Wiley & Sons, Inc., New York, NY, USA, 1986.

[22] A. R. T. Donders, G. J. van der Heijden, T. Stijnen, K. G. Moons, Review: A gentle introduction to imputation of missing values, Journal of Clinical Epidemiology 59 (10) (2006) 1087 – 1091.

[23] V. Robert, Classification croisee pour l'analyse de bases de donnees de grandes dimensions de pharmacovigilance, Ph.D. thesis, Universit Paris-Sud (2017).

[24] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, Journal of he Royal Statistical Society, series B 39 (1) (1977) 1–38.

[25] V. Brault, Estimation et selection de modele pour le modele des blocs latents, Ph.D. thesis, these de doctorat dirigee par Celeux, Gilles Mathematiques Paris 11 2014 (2014).

[26] A. Gelman, D. Rubin, Inference from iterative simulation using multiple sequences, Statistical Science 7 (4) (1992) 457–472.

[27] G. Schwarz, Estimating the dimension of a model, The Annals of Statistics 6 (1978) 461–464.

[28] C. Keribin, V. Brault, G. Celeux, G. Govaert, Estimation and Selection for the Latent Block Model on Categorical Data, Research Report RR-8264, INRIA (Nov. 2013).

[29] K. S. Jones, A statistical interpretation of term specificity and its application in retrieval, Journal of Documentation 28 (1) (1972) 11–21.

[30] A. Salah, M. Nadif, Directional co-clustering, Advances in Data Analysis and Classification (2018) 1–30.

[31] M. Ailem, F. Role, M. Nadif, Model-based co-clustering for the effective handling of sparse data, Pattern Recogn. 72 (C) (2017) 108–122.

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

[33] G. Lubke, B. Muthn, Applying multigroup confirmatory factor models for continuous outcomes to likert scale data complicates meaningful group comparisons, Structural Equation Modeling: A Multidisciplinary Journal 11 (4) (2004) 514–534.

[34] E. E. MaloneBeach, S. H. Zarit, Dimensions of social support and social conflict as predictors of caregiver depression, International Psychogeriatrics 7 (1) (1995) 25–38.

[35] A. S. Zigmond, R. P. Snaith, The hospital anxiety and depression scale, Acta Psychiatrica Scandinavica 67 (6) (1983) 361–370.

[36] C. Biernacki, J. Jacques, Model-Based Clustering of Multivariate Ordinal Data Relying on a Stochastic Binary Search Algorithm, Statistics and Computing 26 (5) (2016) 929–943.

[37] C. Biernacki, A. Lourme, Unifying data units and models in (co-)clustering, Adv. Data Anal. Classif. 13 (1) (2019) 7–31.

[38] G. Govaert, M. Nadif, Mutual information, phi-squared and model-based co-clustering for contingency tables, Advances in Data Analysis and Classification 12 (3) (2018) 455–488.

[39] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1) (1985) 193–218.