Hierarchical clustering with discrete latent variable models and the integrated classification likelihood

Etienne Come, Nicolas Jouvin, Pierre Latouche, Charles Bouveyron

HIERARCHICAL CLUSTERING WITH DISCRETE LATENT VARIABLE MODELS AND THE INTEGRATED CLASSIFICATION LIKELIHOOD

Etienne Côme
COSYS/GRETTIA, Université Gustave-Eiffel,
Noisy-Le-Grand, France

Nicolas Jouvin
Université Paris 1 Panthéon-Sorbonne, SAMM, France
FP2M, CNRS FR 2036, Paris, France

Pierre Latouche
Université de Paris, MAP5, CNRS,
FP2M, CNRS FR 2036, Paris, France

Charles Bouveyron
Université Côte d'Azur, Inria, CNRS, Laboratoire J.A. Dieudonné
Maasai research team, Nice, France

April 22, 2021

ABSTRACT

Finding a set of nested partitions of a dataset is useful to uncover relevant structure at different scales, and is often dealt with a data-dependent methodology. In this paper, we introduce a general two-step methodology for model-based hierarchical clustering. Considering the integrated classification likelihood criterion as an objective function, this work applies to every discrete latent variable models (DLVMs) where this quantity is tractable. The first step of the methodology involves maximizing the criterion with respect to the partition. Addressing the known problem of sub-optimal local maxima found by greedy hill climbing heuristics, we introduce a new hybrid algorithm based on a genetic algorithm which allows to efficiently explore the space of solutions. The resulting algorithm carefully combines and merges different solutions, and allows the joint inference of the number K of clusters as well as the clusters themselves. Starting from this natural partition, the second step of the methodology is based on a bottom-up greedy procedure to extract a hierarchy of clusters. In a Bayesian context, this is achieved by considering the Dirichlet cluster proportion prior parameter α as a regularization term controlling the granularity of the clustering. A new approximation of the criterion is derived as a log-linear function of α, enabling a simple functional form of the merge decision criterion. This second step allows the exploration of the clustering at coarser scales. The proposed approach is compared with existing strategies on simulated as well as real settings, and its results are shown to be particularly relevant. A reference implementation of this work is available in the **R** package **greed** accompanying the paper[1].

***K*eywords** Mixture models · block modeling · co-clustering · genetic algorithm · model-based

1 Introduction

Partitional approaches to clustering seek a partition $\mathcal{P}$ of the observations into K class, or clusters. Hierarchical clustering extends this idea by seeking a path of nested partitions, from finer to coarser. Usually dealt with *ad hoc* similarity functions depending on the data at hand, this approach is popular in unsupervised data analysis such as biological taxonomy, phylogenetics or social network analysis for uncovering hierarchical structures (Everitt et al. 2011, Section 4). In this paper, we address the problem of building hierarchies of partitions in a unified approach within the statistical framework of model-based clustering.

[1] available at http://github.com/comeetie/greed

1.1 Model-based clustering with discrete latent variable models

Model-based clustering is a principled approach for clustering, with a variety of flexible models depending on the data at hand (Bouveyron et al. 2019). Aiming at understanding the different sources of randomness in observations, it can therefore help in interpreting the clusters uncovered in practice. In this paper, we consider a general class of models used in model-based clustering that we call discrete latent variable models (DLVMs). This class encompasses finite mixture models (McLachlan and Peel 2000), which is the most popular instance of model-based approaches. Moreover, it also includes other popular models, which do not exactly fit the definition of finite mixtures. Popular examples are the stochastic block model (SBM) for network analysis (Wang and Wong 1987; Nowicki and Snijders 2001) and its extensions (see Karrer and Newman 2011, for instance), as well as the latent block model (LBM, Govaert and Nadif 2010) for co-clustering. In model-based clustering, the partition is a latent variable $\boldsymbol{Z}$, where each element of $\boldsymbol{Z}$ is a binary vector of size K indicating clustering membership, and K denotes the number of clusters. The general definition of a DLVM assumes that the observations, denoted as $\boldsymbol{X}$, are drawn from a two-step process: first, the latent partition $\boldsymbol{Z}$ is drawn independently from a product of multinomial distributions parameterized by $\boldsymbol{\pi}$. Then, the observations are supposed to be independent given the whole partition. The complete likelihood, known as the *classification* likelihood in this context, is written as:

$$p(\boldsymbol{X}, \mathbf{Z} \mid \boldsymbol{\pi}, \boldsymbol{\theta}) = \prod_{\boldsymbol{z} \in \boldsymbol{Z}} p(\boldsymbol{z} \mid \boldsymbol{\pi}) \underbrace{\prod_{\boldsymbol{x} \in \boldsymbol{X}} p(\boldsymbol{x} \mid \boldsymbol{Z}, \boldsymbol{\theta})}_{\text{factorized}}, \tag{1}$$

where $(\boldsymbol{\pi}, \boldsymbol{\theta})$ are a set of parameters respectively controlling the cluster membership and the conditional distributions.

In the case of finite mixture models, the observations are n independent random vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ in dimension p, which can be summarized in a data matrix $\boldsymbol{X}$ of size $n \times p$. In this context, each observation is assigned to a latent multinomial variable $\boldsymbol{z}_i \in \{0, 1\}^K$, defining its cluster assignment. The latter is independently drawn from a multinomial distribution, with proportions $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$. Then, an observation $\boldsymbol{x}_i$ follows some conditional distribution depending on the value of $\boldsymbol{z}_i$, and the sampling process for all i is as follows:

$$\begin{aligned} \boldsymbol{z}_i \mid \boldsymbol{\pi} &\sim \mathcal{M}(1, \boldsymbol{\pi}), \\ \boldsymbol{x}_i \mid z_{ik} = 1, \boldsymbol{\theta}_k &\sim p(\boldsymbol{x}_i \mid \boldsymbol{\theta}_k). \end{aligned} \tag{2}$$

The parameters $\boldsymbol{\pi}$ controls the prior probability of belonging to each group, while the mixture parameters control the distribution in the k-th cluster, and depend on the observational model at hand. For instance, in a Gaussian mixture we have $p(\boldsymbol{x}_i \mid \boldsymbol{\theta}_k) = \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{m}_k, \boldsymbol{S}_k)$, respectively the mean and covariance matrix in cluster k.

In the case of network analysis with the stochastic block model, the observations are the edges $\boldsymbol{X} = \{x_{ij}\}$, where x_{ij} represents the presence of absence of an edge. It can be binary, $x_{ij} \in \{0, 1\}$, or weighted $x_{ij} \in \mathbb{R}$ (Mariadassou et al. 2010). Observing the edges, *e.g.* the topology of the graph, we wish to cluster the nodes $\{1, \ldots, n\}$. Thus, each node i is assigned to a cluster latent variable $\boldsymbol{z}_i$ and the edges are supposed to be conditionally independent given the partition, with a conditional distribution depending only on the clusters of their out and end-nodes. The sampling process is then written as:

$$x_{ij} \mid z_{ik} z_{jl} = 1, \boldsymbol{\theta}_{kl} \sim p(x_{ij} \mid \boldsymbol{\theta}_{kl}). \tag{3}$$

The *block* terminology stems from the assumption that edges with extremities in the same pair of clusters (k, l) are independent and identically distributed, hence forming homogeneous block in the adjacency matrix $\boldsymbol{X}$. As in mixture models, the parameter $\boldsymbol{\pi}$ controls the group proportions and the latent partition is drawn independently from $\mathcal{M}_K(1, \boldsymbol{\pi})$. However, the SBM does not exactly fit the definition of finite mixture models, since the observations are no longer marginally independent. Matias and Robin (2014) discuss this fact using the moralized graphical model of SBM, revealing the posterior dependencies that do not arise in finite mixture models. The set of parameters $\boldsymbol{\theta}_{kl}$ is now specific to the pair (k, l) of clusters, and depends on the specific model. For instance, in the case of a binary SBM, the block distributions are Bernoulli parameterized by $\theta_{kl} \in [0; 1]$. Degree-corrected versions have also been proposed to account for a strong degree heterogeneity inside blocks, displayed by real-world networks (Karrer and Newman 2011; Zhu et al. 2014). In these models the Bernoulli distribution of edges is replaced by a Poisson, the latter making a good approximation for the Bernoulli when the mean parameter is small (Zhao et al. 2012).

In co-clustering, the observations $\boldsymbol{X} = \{x_{ij}\}$ are supposed to be given in a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$, and one seeks a bipartition $\boldsymbol{Z} = (\boldsymbol{Z}^r, \boldsymbol{Z}^c)$ with K_r clusters over the n rows and K_c clusters over the p columns. The latent block model (LBM, Govaert and Nadif 2010) supposes conditional independence of entries $\boldsymbol{x}_{ij}$ given $\boldsymbol{z}_i^r$ and $\boldsymbol{z}_j^c$. The sampling scheme is given as:

$$x_{ij} \mid z_{ik}^r z_{jl}^c = 1, \boldsymbol{\theta}_{kl} \sim p(x_{ij} \mid \boldsymbol{\theta}_{kl}), \tag{4}$$

and is very close to SBM. Indeed, the latter may be viewed as a particular instance of LBM when $n = p$ and $\boldsymbol{Z}^c = \boldsymbol{Z}^r$. Moreover, the row partition $\boldsymbol{Z}^r$ and column partition $\boldsymbol{Z}^c$ are supposed to be respectively drawn *i.i.d.* from $\mathcal{M}_{K_r}(1, \boldsymbol{\pi}^r)$

and $\mathcal{M}_{K_c}(1, \boldsymbol{\pi}^c)$. Thus, the distribution of $\boldsymbol{Z}$ is a product of multinomials parameterized by $\boldsymbol{\pi} = (\boldsymbol{\pi}^r, \boldsymbol{\pi}^c)$, hence fitting the definition of Equation (1).

Statistical inference constitutes the most popular approach to model-based clustering. In the one hand, the frequentist approach seeks to estimate the model parameters $(\boldsymbol{\pi}, \boldsymbol{\theta})$ via maximum-likelihood estimation. As the partition $\boldsymbol{Z}$ is considered to be a latent variable, the expectation-maximization (EM) algorithm (Dempster et al. 1977) has become quite a universal tool, especially for finite mixture models (McLachlan and Krishnan 2007). For other DLVMs, variational extensions (Blei et al. 2017) have been introduced to deal with the problem of the intractability of the posterior distributions. Notably, this is the case for the binary SBM (Daudin et al. 2008) and the LBM (Govaert and Nadif 2010). On the other hand, the Bayesian paradigm considers model parameters as random variables with so-called prior distributions. In this context, statistical estimation seeks to sample from the posterior distributions of parameters, which is usually done via some Markov-Chain Monte-Carlo scheme (Fruhwirth-Schnatter et al. 2019, chap. 4). Clustering comes as a byproduct of these procedures, as one can then estimate $\boldsymbol{Z}$ from its exact or approximated posterior distribution.

1.2 Exact ICL criterion and greedy hill climbing heuristics

In addition to its flexibility, the statistical framework provides a sound way to choose between different values of K as a model selection problem. Specific statistical tools are required in this context, and several penalized likelihood criteria were proposed such as the Akaike information criterion (AIC, Akaike 1974), or the Bayesian information criterion (BIC, Schwarz 1978). For a recent and detailed review on model selection for model-based clustering, we refer to Fruhwirth-Schnatter et al. (2019, chap. 8). In this paper, we focus on the ICL criterion, which was specifically introduced in the context of model-based clustering. Biernacki et al. (2000) first used a combination of Laplace and Stirling approximations on $\log p(\boldsymbol{X}, \boldsymbol{Z} \mid K)$ to find an asymptotic criterion called ICL_{BIC} due to its close link to the BIC. Since this criterion involves an estimation of $\boldsymbol{Z}$, that is a hard assignment of observations to clusters, it is specifically designed for the clustering task. Thus, as shown in Biernacki et al. 2010, it is more robust to model miss-specification than BIC which focuses on the estimation of the model density. While ICL_{BIC} depends explicitly on an estimation of $\boldsymbol{Z}$, it is still referred to as a model selection criterion in the literature and used in the model-based clustering context. Indeed, its derivation is dependent on the statistical models considered.

Recent works have also considered exact expressions of the ICL, putting a factorized conjugate prior distribution $p(\boldsymbol{\pi}, \boldsymbol{\theta} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}) = p(\boldsymbol{\pi} \mid \boldsymbol{\alpha}) p(\boldsymbol{\theta} \mid \boldsymbol{\beta})$ over the model parameters, and defined as:

$$\begin{aligned} \mathrm{ICL}_{ex}(\boldsymbol{Z}; \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \log \left(\int_{\boldsymbol{\theta}} \int_{\boldsymbol{\pi}} p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{\beta}) p(\mathbf{Z}|\boldsymbol{\pi}) p(\boldsymbol{\pi}|\boldsymbol{\alpha}) \, \mathrm{d}\boldsymbol{\theta} \, \mathrm{d}\boldsymbol{\pi} \right), \\ &= \log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\beta}) + \log p(\boldsymbol{Z} \mid \boldsymbol{\alpha}). \end{aligned} \tag{5}$$

The $\boldsymbol{\alpha}$ parameters control the conjugate distribution, which, in the case of a DLVM, is a Dirichlet[2] over group proportions $\mathrm{Dir}_K(\boldsymbol{\alpha})$. This part is thus common to all DLVMs in the sense that it does not depend on the observational model on $\boldsymbol{X}$. If a symmetric Dirichlet is chosen, with $\alpha_k = \alpha$, the second term can be made explicit using the independence of the elements of $\boldsymbol{Z}$:

$$\mathrm{ICL}_{ex}(\boldsymbol{Z}; \alpha, \boldsymbol{\beta}) = \log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\beta}) + \log \left(\frac{\Gamma(K\alpha) \prod_{k=1}^{K} \Gamma(\alpha + n_k)}{\Gamma(\alpha)^K \Gamma(n + \alpha K)} \right), \tag{6}$$

where $\Gamma(\cdot)$ is the Gamma function and $n_k = \sum_i z_{ik}$. Usually, the hyper-parameter α is set to 1 or $\frac{1}{2}$ to specify uninformative a uniform or a Jeffreys prior.

The hyper-parameters $\boldsymbol{\beta}$ control the conjugate prior over the mixture parameters $\boldsymbol{\theta}$, and depends on the generative model at hand. Naturally, such criterion is restricted to particular DLVMs, where such conjugate distributions are easy to derive, so that the first term in Equation (5) is analytic. However, this class is quite large and expressions are available for the mixture of multinomials (Biernacki et al. 2010) and Gaussian mixture (Bertoletti et al. 2015), while being virtually feasible for any mixture of exponential families as they admit natural conjugate priors (Gelman et al. 2004, p. 42). Exact ICL criteria were also derived for the SBM (Côme and Latouche 2015), the LBM (Wyse et al. 2017) and degree-corrected variants (Newman and Reinert 2016; Riolo et al. 2017). For the sake of self-consistency and to ensure unified notations, Appendix A contains expressions and technical derivations of ICL_{ex} for all the considered models, in particular for the directed formulation of SBM and its degree-corrected variants.

[2]Or a product of Dirichlet distributions $\mathrm{Dir}_{K_r}(\boldsymbol{\alpha}_r) \times \mathrm{Dir}_{K_c}(\boldsymbol{\alpha}_c)$ in the case of co-clustering with the LBM. Except for an additional notation burden, the rest of the discussion easily extends to this case, which is discussed in detail in appendix A

In between the frequentist and the Bayesian approaches, a new line of work started to consider direct maximization of ICL_{ex} with respect to the partition $\mathbf{Z}$, avoiding the inference step over the parameters $(\boldsymbol{\pi}, \boldsymbol{\theta})$ and tackling more directly the clustering task. In order to solve this discrete and combinatorial optimization problem, greedy heuristics were successfully tested to directly optimize this criterion over the space of possible partitions. These approaches consist in hill climbing algorithms, starting from an initial partition and greedily swapping clusters until some local maximum is met. Eventually, at the end, some clusters are merged up to the point where no more merge moves can maximize the ICL_{ex}. Such a type of algorithms performs model selection and clustering at the same time and are computationally attractive compared to approximate or exact inference alternatives. This approach dates back to Tessier et al. (2006), for the latent class model. It was then extended in Côme and Latouche (2015) for SBM, and applied to other DLVMs such as Gaussian mixture models (Bertoletti et al. 2015), LBM (Wyse et al. 2017) and dynamic variants of SBM (Corneli et al. 2016; Zreik et al. 2016). The aforementioned greedy maximization procedure comes with a cost. In practice, the objective is highly multimodal, and the combinatorial nature of the search space multiplies the presence of local maxima in which the methods may be stuck. Different techniques were proposed to tackle this problem, including several restarts (Côme and Latouche 2015), batch versions (Bertoletti et al. 2015), and genetic evolutionary algorithms (GAs, Tessier et al. 2006). This latter methodology borrows from biological evolution principles, combining solutions via crossover operators, allowing random modifications and discarding poor solutions, in an analogy to genetic inheritance, mutations and natural selection. In the context of ICL_{ex} maximization, they represent a promising avenue to efficiently explore the partition space, avoiding the pitfalls of sub-optimal local maxima through the recombination and mutation operators (Fruhwirth-Schnatter et al. 2019, p. 137).

1.3 Contributions and organization of the paper

This paper builds on two main contributions to propose a two-step methodology for hierarchical clustering.

First, we address the issue of spurious local maxima in greedy maximization of ICL_{ex}. We propose a hybrid genetic algorithm mixing an evolutionary strategy with local search to optimize the ICL_{ex} criterion, efficiently exploring the space of partitions. The novelty and efficiency of this approach resides in the representation of solutions as set partitions, and in the crossover operator used to recombine solutions, carefully preserving their structure. This algorithm, presented in Section 2, is adaptable to a wide variety of DLVMs, as soon as swap and merge moves can be efficiently computed. Such an algorithm allows the inference of the number of clusters K and of the clusters themselves. However, it usually leaves a partition with $K > 1$ and does not allow to build a hierarchy of partitions up to $K = 1$.

Second, we introduce an agglomerative hierarchical algorithm considering ICL_{ex} as a function of the hyper-parameter α and relying on a new approximation, using the asymptotic of the log-Gamma function when α goes to 0. We show that decreasing α can unlock fusions in the sense that coarser partitions achieve a greater ICL value. Starting from an ICL-dominant solution at a given level α, typically 1, the proposed heuristic extracts a set of nested clustering that are each dominants with respect to this new criterion over some range of α values. In addition, this strategy, presented in Section 3, enables the construction of a cluster dendrogram, giving a natural ordering for the clusters which is interesting for interpretation and visualization purposes, particularly on real datasets.

These two contributions are generically applicable in the framework of DLVMs for which conjugate priors can be easily derived for the observational model. Moreover these algorithms are naturally linked, working with similar objectives, and the first one can be used as an initialization for the second to extract a hierarchical clustering. One of the particularity of this approach is that it only extracts the relevant part of the dendrogram, since the latter typically starts with an optimal partition obtained at $\alpha = 1$ or $1/2$. Therefore, it avoids the analysis of uninformative fusions commonly encountered in the firsts stages of classical hierarchical agglomerative clustering algorithms. This approach is also computationally efficient and may handle large datasets which could be hard to grasp with classical fully hierarchical algorithms.

Section 4 gives a detailed investigation of the two algorithms behaviors on simulated and real datasets, along with a thorough comparison with related model-based clustering algorithms. Three types of DLVMs are considered: the mixture of multinomials, the SBM and LBM, as well as their degree-corrected versions.

Finally, the open-source **R** package (R Core Team 2019) **greed** provides a reference implementation of the algorithms introduced in this paper. The implementation is extendable and new models can be integrated. The main computationally demanding methods were developed in **Cpp** thanks to the **Rcpp** package (Eddelbuettel and Balamuta 2017) taking advantages of sparse matrix computational efficiency thanks to the **RcppArmadillo** and **Matrix** packages (Eddelbuettel and Sanderson 2014; Bates and Maechler 2019). Eventually, the **future** package (Bengtsson 2019) was used to enable easy parallelization of the computations of the proposed hybrid genetic algorithm.

1.4 Motivating example

As a motivating example for the proposed two-step methodology, we simulate a random SBM graph with $n = 1500$ nodes and a hierarchical cluster structure with 3 clusters each composed of 5 small clusters. The small clusters have an intra-connectivity probability of 0.1 and a probability of connecting a node from the same cluster of 0.025. Otherwise, two random nodes may be connected with a probability of 0.001. Figure 1 compares the clustering results in the form of adjacency matrices obtained by different methods : a greedy hill climbing optimization with a random starting partition with twenty clusters, the proposed hybrid optimization algorithm and the same results after a reordering of the clusters with the hierarchical heuristic. As clearly shown by this example, the greedy heuristic with a random starting point suffers from under-fitting with only four clusters extracted among the 15 simulated. The hybrid algorithm does not suffer from the same problem in this example, and recovers correctly the 15 simulated clusters. Finally, the hierarchical ordering enable a clear visualization of the hierarchical structure of this dataset, that is also clearly depicted in the extracted dendrogram presented in Figure 2.

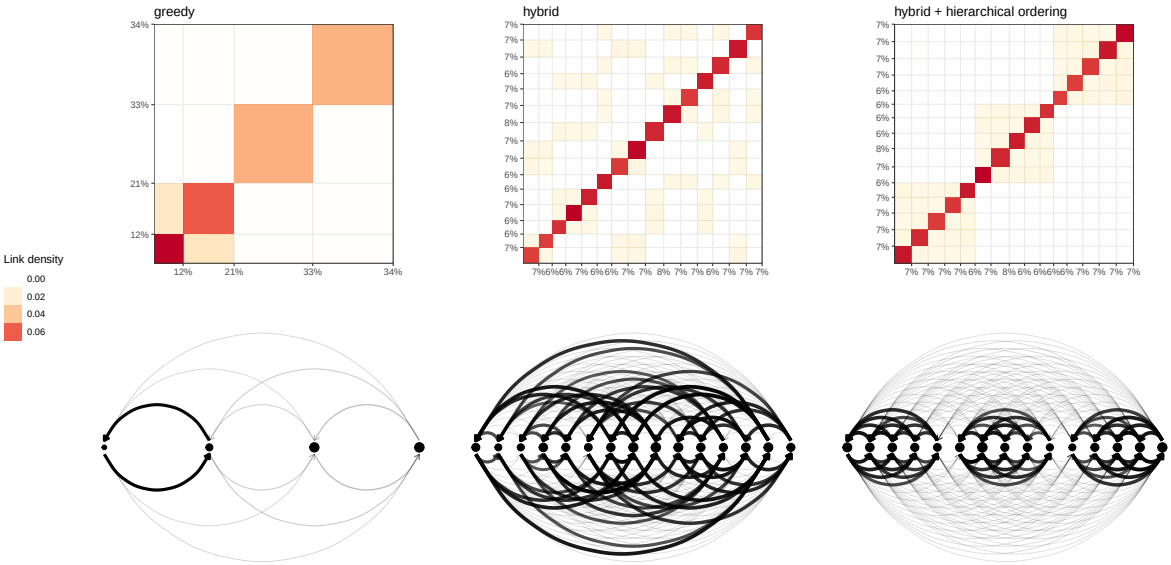

Figure 1: Motivating example for the proposed algorithms. Block matrix representation of the solutions (upper row) and cluster node link diagram (bottom row) obtained with (from left to right) a greedy algorithm with a random starting point, the proposed hybrid algorithm and the same clustering but with clusters re-arranged thanks to the hierarchical ordering.

2 A hybrid genetic algorithm for DLVMs

As explained above, several works rely on the ICL_{ex} criterion as an objective function to maximize with respect to the partition $\boldsymbol{Z}$. These are mainly based on greedy hill climbing algorithms: starting from a carefully chosen over-segmented initial partition, or *seed*, swaps and eventually merges are applied to increase the criterion. In addition to the competitive computational complexity and the ease of implementation, these algorithms may be seen as an automatic way to perform model selection, as clusters may be emptied during the process. In the SBM case, Côme and Latouche (2015) proposed a thorough comparison with state-of-the-art methods that illustrates the interest of such algorithms.

However, a major drawback of this approach is its dependency to the initialization. Indeed, defining a relevant initial partition is not trivial, and the method may lead to under-fitting as demonstrated in the introductory example in Figure 1. Here, the issue seems to lie in the lack of exploration of the partition space, and genetic algorithms (GAs) have been proposed to improve the exploration. Starting from a given solution, the latter evolves a population of candidate solutions by selecting some of the most promising ones, combining them, and mutating them until a specified number of generations or some stopping criterion is met. As described in Eiben and Smith (2004, Chapter 2), the fundamental components of such algorithms are the solution representation, the selection strategy and the variation operators used for recombination and mutation. However, while GAs are very good at identifying near-optimal regions of the search space, they can take a relatively long time to reach a local optimum in the region of interest. In order to improve their

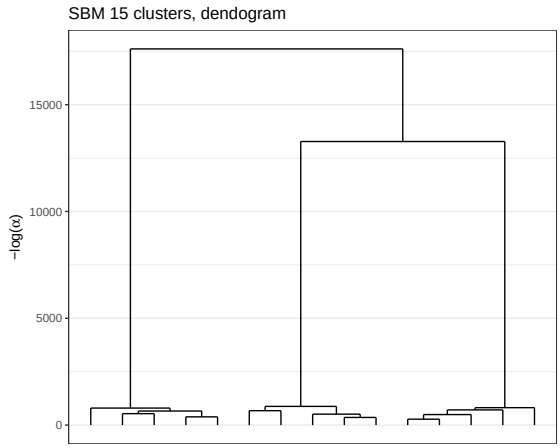

Figure 2: Motivating example for the proposed algorithms. Clusters dendrogram extracted with the hierarchical regularization path heuristic.

exploitation capacity, a number of works suggested hybridizing GAs with efficient local search algorithms capable of improving solutions between each generation (see Eiben and Smith 2004, Chapter 10). These evolutionary methods have been named in various ways, such as hybrid GAs, memetic GAs, and genetic local search algorithms.

In the case of ICL_{ex} maximization, existing greedy heuristics may be seen as such local search algorithms, locally improving a partition, and we build on this idea to propose a hybrid GA. In the following, we discuss the practical choices made when designing the genetic algorithm. Moreover, we emphasize that, in this section, the prior parameters are considered to be fixed to uninformative or default values, and we only optimize $\mathrm{ICL}_{ex}(\boldsymbol{Z})$ with respect to the partition defined by $\boldsymbol{Z}$.

2.1 Solution recombination with the cross-partition operator

The first step towards building a GA is to define a way to represent candidate solutions inside the algorithm. The latter is also called the *genotype space*, with genotypes as points in this space. This choice is fundamental as it guides the variation operators such as the recombination operator, also known as crossover, which combines two parent genotypes into a new one, and the mutation operator, which randomly modifies genotypes. In the case of clustering, the elements of the original space of solutions are partitions $\mathcal{P} = \{\boldsymbol{C}_1, ..., \boldsymbol{C}_K\}$ of $\{1, \dots, n\}$ into K clusters, with a variable K. Tessier et al. (2006) used integer encoding, which consists in a vector of length n where each individual is assigned to an integer $k \in \{1 \dots, K\}$ representing its cluster assignment, and is similar to $\boldsymbol{Z}$. However, this approach presents a major drawback. Indeed, akin to the label switching problem in statistical inference, the ICL_{ex} objective function is invariant under a permutation of the cluster indices, and this representation is therefore heavily redundant. Thus, as emphasized in Hruschka et al. (2009), popular crossover operators based on crossover points will not consider this specificity and will completely break the structure of the solution. This is notably the case in Tessier et al. (2006), leading to slow evolution of the population of solutions. We propose to circumvent this issue by directly choosing the space of partitions as the genotype space, defining crossover and mutation operators on it. Such operators will not suffer from label switching, and will preserve the clustering structure present in the genotypes.

Crossover operator The crossover operator defines how two parent genotypes $\mathcal{P}^1 = \{\boldsymbol{C}_1^1, ..., \boldsymbol{C}_{K_1}^1\}$ and $\mathcal{P}^2 = \{\boldsymbol{C}_1^2, ..., \boldsymbol{C}_{K_2}^2\}$ are combined together to form an offspring. We propose to use the cross-partition, defined as the set of all possible intersections between the elements of the two partitions:

$$\mathcal{P}^1 \times \mathcal{P}^2 := \left\{ \boldsymbol{C}_k^1 \cap \boldsymbol{C}_l^2 \,,\, \forall k \in \{1, ..., K_1\}, \forall l \in \{1, ..., K_2\} \right\} \setminus \{\emptyset\} \,.$$

This operator produces a new partition with at most $K_1 \times K_2$ clusters, which is a refinement of $\mathcal{P}^1$ and $\mathcal{P}^2$ in the sense that both parents may be reconstructed using merge operations. It is also the first common ancestor of both $\mathcal{P}^1$ and $\mathcal{P}^2$ in the partition lattice. Hence, its interest is twofold. First, as in the motivating example, if both parent partitions are under-fitted, crossing them allows the algorithm to go backward in the partition lattice, considering finer clustering. Second, it is particularly appropriate for the hybridization with greedy heuristics. Indeed, unnecessary clusters may be created when the crossed solutions are around the best one. Then, a greedy local search based on merge moves may be used to remove these clusters efficiently.

2.2 Selection, mutation and the hybrid algorithm

The remaining aspects of the genetic algorithm concern the selection procedure and the mutation operator. As the population size V is kept fixed throughout the algorithm, selection defines which parent genotypes are combined together to form offspring. On the basis on numerical experiments, we propose to keep a rank-based selection policy (see Eiben and Smith 2004, pp.81-82). In this scheme, the selected genotypes for building the next generation are chosen according to a probability proportional to their rank in terms of ICL_{ex}.

As for the mutation operator, it randomly acts on the elements of a genotype, here the clusters of a partition. Together with the recombination operator, it allows introducing variability in the algorithm allowing for a better exploration. Again, a desirable property is the refinement of a given partition, and a natural mutation to consider is to split a cluster in two new ones at random. Then, local searches consisting in swaps and merges can either undo a poor split or explore new directions. The resulting hybrid greedy algorithm is represented as pseudo-code in Algorithm 1.

Algorithm 1: Hybrid genetic algorithm

Data: population size: V, probability of mutation: pm, maximum number of generations: $maxgen$, dataset $\boldsymbol{X}$
Result: a partition $\mathcal{P}^*$

Build a population $G = \{\mathcal{P}^1, ..., \mathcal{P}^V\}$ of initial solutions using V random initializations and greedy swaps.
$nbgen \leftarrow 1$
while $nbgen < maxgen$ **do**
  add the best solution $\mathcal{P}^*$ in the population to the new generation $G_n = \{\mathcal{P}^*\}$
  sample according to their rank in terms of ICL, $(V-1)$ pairs of solution in G
  for *each sampled pairs* $(\mathcal{P}^1, \mathcal{P}^2)$ *of partitions* **do**
    build the cross partition $\mathcal{P}$ of $\mathcal{P}^1$ and $\mathcal{P}^2$
    $\mathcal{P} = \mathcal{P}^1 \times \mathcal{P}^2$
    update $\mathcal{P}$ using greedy merge
    if $random < pm$ **then**
      sample a cluster of $\mathcal{P}$ and split it randomly in two
    update $\mathcal{P}$ using greedy swap
    add $\mathcal{P}$ to the new generation $G_n = \{G_n, \mathcal{P}\}$
  replace the population by the new generation $G = G_n$
  $nbgen \leftarrow nbgen + 1$
return the best solution $\mathcal{P}^*$ of G_n

Computational efficiency From a computational perspective, the crossover and mutation operator can easily be parallelized since they are independent for each pair of solutions to combine. In addition, while already efficient, this first version was optimized by taking advantage of a special feature of the problem. Indeed, after having formed the crossed partition, one may determine the pairs of clusters (k, l) that have a common parent either in $\mathcal{P}^1$ or $\mathcal{P}^2$:

$$\left\{(\boldsymbol{C}_k, \boldsymbol{C}_l) \in (\mathcal{P}_1 \times \mathcal{P}_2)^2 : \exists \boldsymbol{C} \in \mathcal{P}^1 \cup \mathcal{P}^2, (\boldsymbol{C}_k \cap \boldsymbol{C} \neq \emptyset) \text{ and } (\boldsymbol{C}_l \cap \boldsymbol{C} \neq \emptyset)\right\},$$

only allowing merge and swap movements between them. This allows gaining a factor K, which can be interesting for a large number of clusters, especially in the first iterations of the algorithm. The rationale behind this restriction is that both initial partitions may be recovered if needed, while the inspection of a non-negligible quantity of merge and swap moves having a low chance of being relevant can be avoided. Moreover, the computational cost of a swap or a merge is model dependent, although the ICL_{ex} expressions for the considered models generally allows for efficient formulas. For example, in the case of the binary SBM, Côme and Latouche (2015, Appendix B and C) derived swaps and merge updates with $\mathcal{O}(l + K^2)$ and $\mathcal{O}(K)$ costs respectively, where l is the average number of edges per node.

Setting genetic hyperparameters The size of the population of solutions V, the probability of mutation and the maximum number of generations are hyperparameters to be set beforehand. They are used to tune the trade-off between computational efficiency and exploration capacity. The hybridization with local maximization methods enhances the exploration capability, therefore reducing the need for large values of these parameters. In Section 4 and in the package, we typically set the population size around 50, the probability of mutation to 0.25 and the maximum number of generations to 10.

This hybrid genetic algorithm allows the extraction of a natural clustering when the number of clusters is unknown, by carefully exploring the space of partitions and exploiting relevant solutions. The trade-off between the two is controlled by a few tuning parameters, namely the population size and the probability of mutation, and the computational complexity, which is model-dependent, is competitive with other approaches. The experiments carried in Section 4 will demonstrate its performances in real and simulated settings. In practice, relying on a greedy optimization algorithm and allowing merges between clusters leaves a solution with $K > 1$ clusters. In the following, we propose a regularization strategy based on the parameter α to allow extra merges and to build a complete hierarchy of clusters up to $K = 1$. We emphasize that α is key to unfold complete hierarchies.

3 Hierarchical extension from regularization path

So far, we have seen how to maximize the ICL_{ex} with respect to $\boldsymbol{Z}$, the prior hyper-parameters being kept fixed, leaving a clustering result at a given level K. We now consider the problem of building a complete hierarchy of clusters using the same criterion. In this section, we introduce the second contribution of this paper: a greedy agglomerative algorithm for hierarchical clustering, based on an approximation of ICL_{ex}. Hereafter, ICL_{ex} is viewed not only as a function of the partition $\boldsymbol{Z}$ but also of the hyper-parameter α. The asymptotic behavior of the log-Gamma function near 0 is used to derive a simple functional form for the criterion as a function of α. The resulting approximation is called ICL_{lin} due to its log-linear dependency in α. Then, α is used as a regularization parameter which unlocks access to simpler, coarser, solutions. The algorithm produces a hierarchy of nested partitions along with the sequence of the regularization parameters which enabled the fusions : $(\boldsymbol{Z}^{(k)},\, \alpha^{(k)})_{k=K,\dots,1}$. Eventually, the extracted partitions may be investigated, and the hierarchical structure employed to get a pseudo-ordering of the initial clusters to enhance the graphical representation of the clustering results. As we shall see, the key advantage of ICL_{lin} compared to ICL_{ex} is that it allows to obtain explicit values for α allowing merges without having to rely on prohibitive non-linear equations resolution or grid search strategies.

3.1 A new approximation for the exact ICL

As shown in Equation (6), ICL_{ex} decomposes as the sum of two terms. The first one is $\log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\beta})$, the conditional integrated log-likelihood of the data, given the partition $\boldsymbol{Z}$. In the following, it will be denoted by $D(\boldsymbol{Z})$. This quantity only depends on the observed data $\boldsymbol{X}$, the partition $\boldsymbol{Z}$, and the model specification. The second term is the integrated log-likelihood of $\boldsymbol{Z}$ and depends on the Dirichlet hyper-parameter α:

$$\log p(\boldsymbol{Z} \mid \alpha, K) = \log\Gamma(\alpha K) + \sum_{k=1}^{K} \log\Gamma(\alpha + n_k) - K\log\Gamma(\alpha) - \log\Gamma(n + \alpha\, K). \tag{7}$$

Here, the dependency between K and $\boldsymbol{Z}$ is made explicit, the former representing the number of clusters in the latter. Then, we consider the asymptotic behavior of the expression above when α becomes small. First, recall that the log-Gamma function behaves as minus the natural logarithm near 0:

$$\log\Gamma(\alpha) = \log(\alpha^{-1}\,\Gamma(\alpha + 1)) \;\approx_0\; -\log(\alpha)\,. \tag{8}$$

Then, considering K bounded, we can use this approximation on $\log\Gamma(\alpha)$ and $\log\Gamma(\alpha K)$ respectively. Finally, we use the two additional approximations $\log\Gamma(n_k + \alpha) \approx \log\Gamma(n_k)$ and $\log\Gamma(n + \alpha K) \approx \log\Gamma(n)$ when alpha is close to 0. Combining these approximations, a simpler expression of Equation (7) as a log-linear function of α, can be derived:

$$\log p(\boldsymbol{Z} \mid \alpha, K) \approx_0 (K-1)\log(\alpha) - \log(K) + \sum_{k=1}^{K} \log\Gamma(n_k) - \log\Gamma(n)\,.$$

The algorithm introduced in this paper relies on this approximation, and the corresponding criterion is named ICL_{lin}, where *lin* stands for linear:

$$\mathrm{ICL}_{lin}(\boldsymbol{Z}, \alpha) = D(\boldsymbol{Z}) + (K-1)\log(\alpha) - \log(K) + \sum_{k=1}^{K} \log\Gamma(n_k) - \log\Gamma(n)\,.$$

All quantities that do not depend on α may be grouped in an intercept:

$$I(\boldsymbol{Z}) \coloneqq D(\boldsymbol{Z}) - \log(K) + \sum_{k=1}^{K} \log\Gamma(n_k) - \log\Gamma(n)\,. \tag{9}$$

Note that the $\Gamma(n_k)$ term is always well-defined here, since we do not consider empty clusters. Then, the log-linearity of the new criterion appears explicitly:

$$\mathrm{ICL}_{lin}(\boldsymbol{Z}, \alpha) = (K-1)\log(\alpha) + I(\boldsymbol{Z})\,. \tag{10}$$

Naturally, the quality of this approximation depends on how small both α and αK are. For the first one, the approximation of Equation (8) is quite mild, even for standard α values such as 1 or $\frac{1}{2}$. As for αK, while its value may be relatively far from 0 for $\alpha = 1$, we verify in practice that α rapidly decreases several orders of magnitude below 1 as of the first fusion. This ensures that the approximation is correct throughout the procedure.

3.2 Hierarchy construction

Looking at the functional form of the previous approximation, a natural goal is to search for the Pareto front in the $(\log\alpha,\ \mathrm{ICL}_{lin}(\boldsymbol{Z}, \alpha))$ plane. The latter corresponds to a set of dominating partitions with respect to ICL_{lin}, for a certain range of α values in $]0, 1]$, or equivalently for a range of $\log(\alpha)$ values in $]-\infty, 0]$. Formally, we define the Pareto front as:

$$P = \{(\boldsymbol{Z}^\star, I_\alpha^\star)\ :\ \forall \alpha \in I_\alpha^\star,\ \forall \boldsymbol{Z} \neq \boldsymbol{Z}^\star,\ \mathrm{ICL}_{lin}(\boldsymbol{Z}^\star, \alpha) \geq \mathrm{ICL}_{lin}(\boldsymbol{Z}, \alpha)\}, \tag{11}$$

where $I_\alpha^\star$ are intervals of $]0, 1]$. Finding this set of dominating partitions and ranges is not a trivial task. However, the difficulty is reduced if we consider a dominant partition $\boldsymbol{Z}$ for certain level α, and restrict ourselves to look for partitions that results from merges of $\boldsymbol{Z}$. Indeed, we will show that it is direct, for a given partition $\boldsymbol{Z}$, to find the hyper-parameter $\alpha^\star$ and the pair (g^*, h^*) of clusters to merge, such that the obtained coarser partition $\boldsymbol{Z}_{g^*\cup h^*}$ will dominate $\boldsymbol{Z}$, along with any other partition $\boldsymbol{Z}_{g\cup h}$, over $]0, \alpha^*]$. Starting from an initial clustering $\boldsymbol{Z}^{(K)}$, these locally optimal merges can be used to build a heuristic, in the spirit of hierarchical agglomerative clustering, that will extract a sequence of nested partitions to approximate the Pareto front defined by Equation (11). While this heuristic is not guaranteed to extract the Pareto front, it may still provide good results, especially starting from a dominant partition, *e.g.* obtained by maximizing $\mathrm{ICL}_{ex}(\boldsymbol{Z}, 1)$ with the hybrid optimization algorithm introduced in the previous section. Intuitively, if a partition $\boldsymbol{Z}$ is locally dominant for some α value, there is a good chance that the next dominant partition for some $\alpha' < \alpha$ will be a coarse version of $\boldsymbol{Z}$. Indeed, to surpass a dominant solution in α', the new dominant solution must be coarser in order to benefit from a reduced decreasing slope, while it must also have a high intercept $I(\boldsymbol{Z}')$. Solutions built from merging two clusters of $\boldsymbol{Z}$ are coarser, therefore fulfilling the first requirement. Moreover, since $\boldsymbol{Z}$ is already dominant, we may also hope that a coarser version of it also has a high intercept, and therefore dominates other partitions for this new α' value. Let us therefore detail this heuristic, and the conditions under which a fusion opportunity exists.

3.2.1 Fusion opportunity

For any given partition $\boldsymbol{Z}^{(k)}$, with $k \geq 2$ clusters, let us define $\mathbb{Z}^{(k-1)}$ as the space of all the partitions with $(k-1)$ clusters that are coarser than $\boldsymbol{Z}^{(k)}$:

$$\mathbb{Z}^{(k-1)} = \left\{ \boldsymbol{Z}_{g\cup h}\ : \text{the partition } \boldsymbol{Z}^{(k)} \text{ with clusters g and h merged, } g \neq h \right\}\,.$$

Note that we will use the terminology *mother* partition for $\boldsymbol{Z}^{(k)}$ and *child* partition for any element of $\mathbb{Z}^{(k-1)}$.

As pointed out previously, with $\boldsymbol{Z}^{(k)}$ fixed, the function $\mathrm{ICL}_{lin}(\boldsymbol{Z}^{(k)}, \cdot)$ is log-linear with slope $(k-1)$ and intercept $I(\boldsymbol{Z}^{(k)})$. This implies that the slope of the ICL_{lin} functions decreases incrementally to 0 as k decreases to 1. Figure 3 illustrates this behavior of ICL_{lin}, with respect to the number of clusters k. It can easily be seen that the slopes decrease until k reaches 1 which corresponds to an horizontal line.

From Equation (10), we are able to derive the expression of the variation of the ICL_{lin} between a mother partition $\boldsymbol{Z}^{(k)}$ and any of its child $\boldsymbol{Z}_{g\cup h}$ as a function of α. Graphically, this variation, denoted as $\Delta_{g\cup h}(\alpha)$, is the difference between two straight lines of slope $k-2$ and $k-1$ respectively. Moreover, the dominance shifting point corresponds to its zero, which can be easily derived and will be denoted by $\alpha_{g,h}$:

$$\begin{aligned} \Delta_{g\cup h}(\alpha_{g,h}) = 0 &\iff \mathrm{ICL}_{lin}\left(\boldsymbol{Z}_{g\cup h}, \alpha_{g,h}\right) - \mathrm{ICL}_{lin}\left(\boldsymbol{Z}^{(k)}, \alpha_{g,h}\right) = 0\,, \\ &\iff \log(\alpha_{g,h}) \coloneqq I(\boldsymbol{Z}_{g\cup h}) - I(\boldsymbol{Z}^{(k)})\,. \end{aligned} \tag{12}$$

In geometric terms, we know that below this level, the child partition $\boldsymbol{Z}_{g\cup h}$ dominates its mother $\boldsymbol{Z}^{(k)}$ in terms of ICL_{lin}. Thus, for any mother partition $\boldsymbol{Z}^{(k)}$, we are able to compute the tipping points $(\alpha_{g,h})_{g<h}$ for the $k(k-1)/2$ possible child partitions. To find the best fusion, we recall the form of the ICL_{lin} for any partition $\boldsymbol{Z}_{g\cup h} \in \mathbb{Z}^{(k-1)}$ from Equation (10):

$$\mathrm{ICL}_{lin}(\boldsymbol{Z}_{g\cup h}, \alpha) = (k-2)\log(\alpha)\ +\ I(\boldsymbol{Z}_{g\cup h})\,, \quad \forall g, h\,.$$

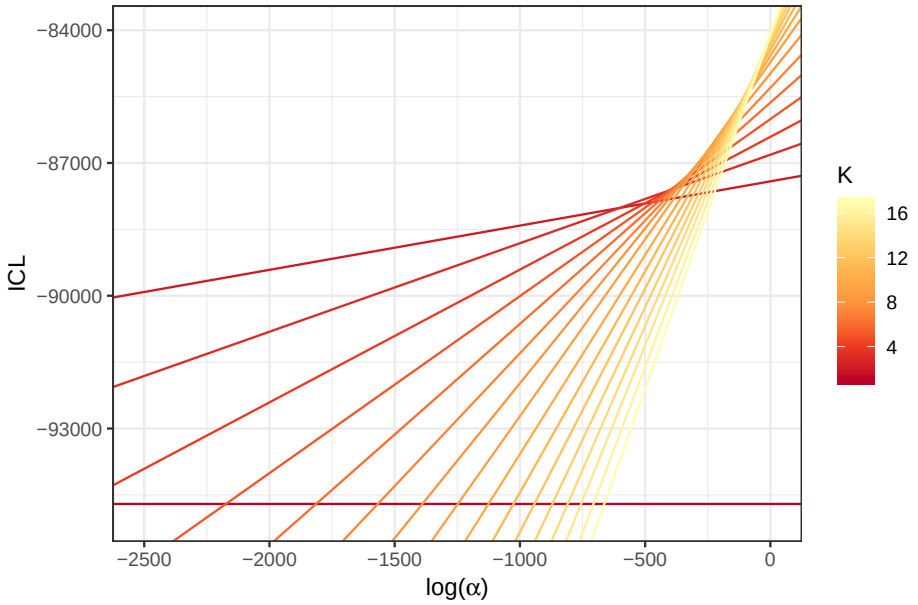

Figure 3: Lines of slope $k-1$ representing the functions $\log \alpha \mapsto ICL(\boldsymbol{Z}^{(k)}, \log\, \alpha)$ for a collection of partitions $\boldsymbol{Z}^{(k)}$ with a decreasing number of clusters $k = 21, \ldots, 1$. We see that the ICL_{lin} order changes as α decreases, favoring coarser partitions. The x-axis slice at $\log \alpha = 0$ corresponds to the intercepts $I(\boldsymbol{Z}^{(k)})$.

So it is clear that, viewed as functions of $\log \alpha$, the ICL_{lin} of all child partitions in $\mathbb{Z}^{(k-1)}$ are parallel straight lines of slopes $(k-2)$, only differing by their intercepts. This guarantees us that there exists a unique partition, uniformly dominating in α, in $\mathbb{Z}^{(k-1)}$. Formally:

$$\exists! \boldsymbol{Z}_{g^\star \cup h^\star} \in \mathbb{Z}^{(k-1)} \text{ s.t. } : \forall \alpha > 0,\ \forall \boldsymbol{Z}_{g \cup h} \in \mathbb{Z}^{(k-1)} \\ \text{ICL}_{lin}\left(\boldsymbol{Z}_{g^\star \cup h^\star}, \alpha\right) \geq \text{ICL}_{lin}\left(\boldsymbol{Z}_{g \cup h}, \alpha\right) . \tag{13}$$

This partition corresponds to the one with the greatest intercept which, by Equation (12), also happens to be the one intersecting with $\boldsymbol{Z}^{(k)}$ at the greatest $\alpha_{g,h}$:

$$(g^\star, h^\star) = \arg\max_{g,h} I(\boldsymbol{Z}_{g \cup h}) = \arg\max_{g,h} I(\boldsymbol{Z}_{g \cup h}) - I(\boldsymbol{Z}^{(k)}) = \arg\max_{g,h} \alpha_{g,h}.$$

This discussion describes how to find the best fusion, going from a partition $\boldsymbol{Z}^{(k)}$ to $\boldsymbol{Z}^{(k-1)} = \boldsymbol{Z}_{g^\star \cup h^\star}$ by setting $\alpha^{(k-1)} = \alpha_{g^\star, h^\star}$. Taking this greedy approach, one may perform such locally optimal merges sequentially in a fast and efficient bottom-up procedure until all clusters have been merged into a unique cluster. Hence, we can see how α acts as a regularization parameter, enabling for fusions. Taking an initial partition $\boldsymbol{Z}^{(K)}$ and a given initial $\alpha^{(K)}$, typically 1, this will provide a set of nested clustering solutions $(\boldsymbol{Z}^{(k)}, \alpha^{(k)})_{k=K,\ldots,1}$.

Finally, the log-scale used for the x-axis of Figure 3 highlights that α quickly decreases toward 0 from the first iteration, thus insuring the validity of the approximation at the first stages. This fact is also empirically verified in all the experiments of Section 4, and is easily visualized via the dendrogram representation introduced below.

3.2.2 Post-processing

The previous strategy outputs a hierarchy, meaning a set of nested clustering with a number of clusters ranging from K to 1. Each merge performed by the algorithm is stored into a binary tree, keeping track of the hierarchical relations between clusters. However, one important point to observe is that some of the partitions extracted by this agglomerative greedy algorithm may not be dominant anywhere in $\alpha \in]0, 1]$, with respect to the others. This corresponds to situations where combining several merges in one step is better than performing them sequentially. Indeed, in geometrical terms, there is no guarantee that the intersection between the ICL_{lin} of $\boldsymbol{Z}^{(k)}$ and $\boldsymbol{Z}^{(k-1)}$ is at a greater α than between $\boldsymbol{Z}^{(k)}$ and $\boldsymbol{Z}^{(k-2)}$. Or, equivalently, there is no guarantee that the sequence $(\alpha^{(k)})_k$ is non-increasing. This is quite natural since ICL_{lin} is a penalized criterion, thus it does not necessarily increase with the model complexity. Since

such partitions cannot belong to the approximated Pareto front, we propose to remove them. Indeed, they are easy to track since they correspond to merge k where $\alpha^{(k-1)} > \alpha^{(k)}$. Then, having extracted the $F \leq K$ dominating partitions, it is possible to recompute the α_f where they cross each other to get a sequence $(\boldsymbol{Z}_f, \alpha_f)_{f=F,\dots,1}$ with a non-increasing sequence $(\alpha_f)_f$. Although the index of $\boldsymbol{Z}_f$ does not indicate its number of clusters anymore, the sequence still consists in a hierarchy of nested partitions, which are now ordered in terms of ICL_{lin} in their ranges of dominance: $\text{ICL}_{lin}(\boldsymbol{Z}_f, \alpha) > \text{ICL}_{lin}(\boldsymbol{Z}_l, \alpha)$, $\forall\, f \neq l$, $\forall \alpha \in [\alpha_{f-1}, \alpha_f]$. Figure 4 illustrates this post-processing, where the ICL_{lin} lines associated with each $\boldsymbol{Z}^{(k)}$ extracted by the greedy agglomerative algorithm are depicted with their corresponding dominance ranges, and the nowhere dominant partitions are highlighted.

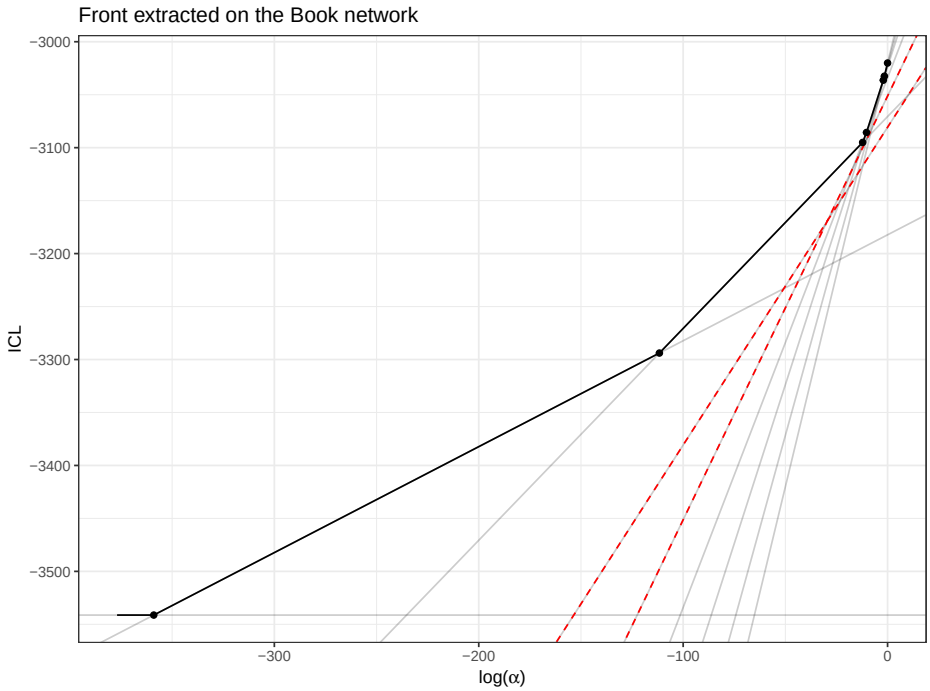

Figure 4: $\text{ICL}_{lin}(\boldsymbol{Z}, \alpha)$ as a function of $\log(\alpha)$ for every partition extracted by the greedy hierarchical algorithm on the **Books** co-purchasing network (see Section 4.3 for dataset details), with a degree corrected SBM model. The partitions that do not have any range of dominance are highlighted with dashed red lines, and the dominant ranges with solid black lines. The intersection between dominant partitions correspond to the recomputed tipping, dominance shifting points $(\alpha_f)_f$. The initial partition $\boldsymbol{Z}^{(K)}$ was built using Algorithm 1.

3.2.3 Visualization

Along with its property discussed above, the proposed algorithm possesses interesting graphical features for the visualization of both the hierarchy, with a dendrogram, as well as the initial clustering $\boldsymbol{Z}^{(K)}$ using the partial ordering of the leaves.

Dendrogram The sequence $(\alpha_f)_{f=F,\dots,1}$ may be used for the construction of a dendrogram representing the cluster merge tree from $\boldsymbol{Z}^{(K)}$ to $\boldsymbol{Z}^{(1)}$, with the non-increasing sequence $(-\log(\alpha_f))_f$ in the y-axis. Thus, the hierarchical structures of the clusters can be visualized as well as the amount of regularization needed for each fusion(s). Indeed, as discussed above, the y-axis can then be seen as the drop in ICL_{lin} induced by each merge, acting as an analog of the traditional *dissimilarity* in agglomerative strategies. Figure 5 presents the obtained dendrogram for the Book network of Section 4.3.

Leaves ordering Another interesting feature of the proposed procedure is the partial ordering of the initial clustering $\boldsymbol{Z}^{(K)}$ that can be obtained from the merge tree structure. Indeed, for a binary tree with K leaves, there are 2^{K-1} permutations of its leaves that are compatible with its structure. In other words, there are 2^{K-1} possible dendrograms representing the same hierarchy. However, some are more relevant than others and we seek to find the optimal tree

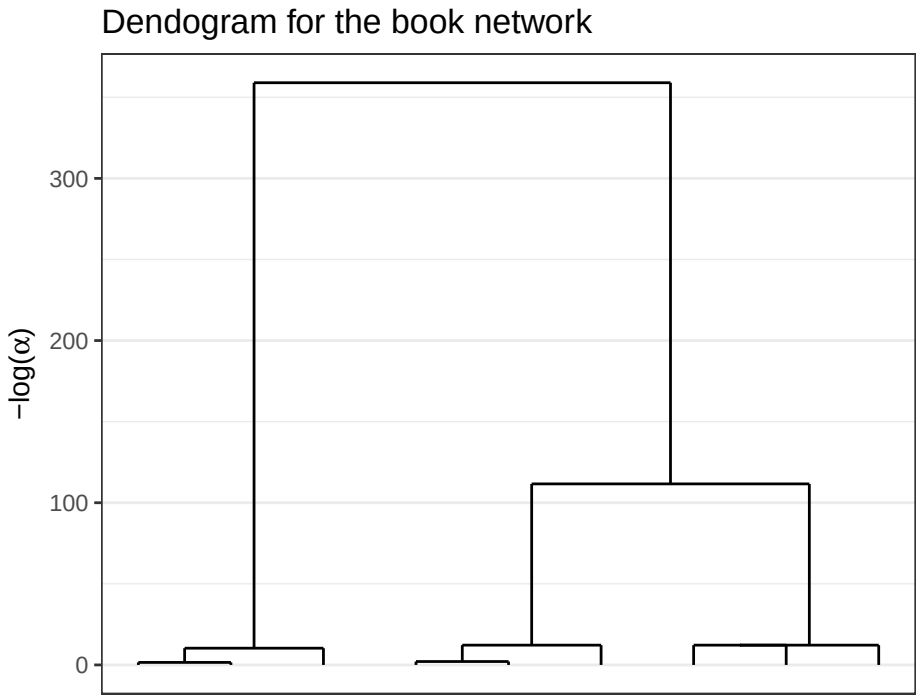

Figure 5: Dendrogram representation of the extracted hierarchy for the **Books** co-purchasing network (see Section 4.3 for dataset details).

consistent ordering (or permutation) σ that minimizes the sum of merge costs between successive clusters at $\alpha = 1$:

$$\sigma = \arg\min_{\sigma} \sum_{k=1}^{K-1} \Delta_{\sigma(k)\cup\sigma(k+1)}. \tag{14}$$

An efficient algorithm based on dynamic programming (Bar-Joseph et al. 2001) is already available to solve this optimization problem using the binary tree structure. As shown in Figure 1, such ordering of the initial clusters may be used advantageously to draw node-link diagrams or block adjacency matrix, enhancing visualization and simplifying the interpretation of the clustering results. This approach is used in the **greed** package to provide the final ordering of the clusters.

4 Numerical experiments

Thus far, the discussion has been purposely general in order to express the generic aspect of the proposed two-fold methodology. We now illustrate the behavior of the two algorithms in simulated and real settings for several particular instances of DLVMs. First, the hybrid optimization algorithm is compared with other clustering algorithms in simulated scenario for count-data and network clustering. The results highlight both its improvement of hill climbing heuristics, as well as its advantage compared to standard statistical approaches. The results of the hierarchical algorithm is then analyzed on several real datasets for graph clustering and co-clustering, demonstrating its interest in finding relevant hierarchical structures. The formula of ICL_{ex} are derived in appendix A for each specific model.

4.1 Medium-scale SBM simulations

To investigate the performances of the hybrid algorithm, we pursue with our motivating example defined in Section 1. The simulation consists of a SBM graph with 1500 nodes and 15 clusters with a hierarchical structure of 3 clusters each divided into 5 small clusters. Figure 6 (left) presents the evolution of the ICL criterion among the different generations of solutions built by the algorithm. As clearly shown by this figure, the criterion improves at each generation until it reaches a plateau around the fourth generation. A comparison of the hybrid algorithm with other methods is also

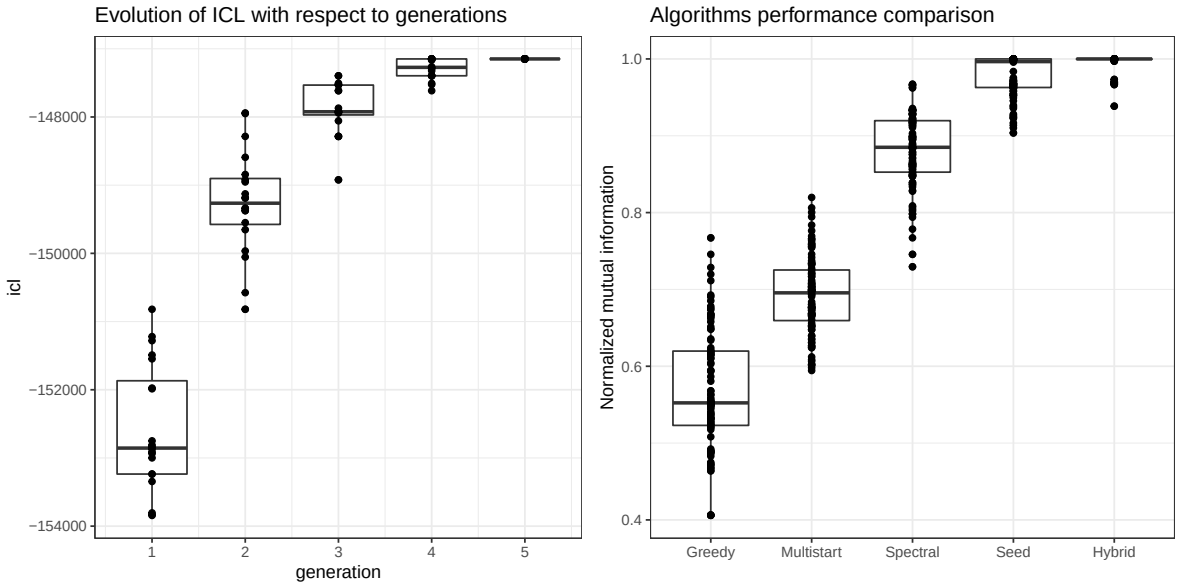

Figure 6: Evolution of ICL_{ex} with respect to the generation for one run of the hybrid algorithm (left), NMI between simulated and reconstructed clusters for one hundred simulations for the different algorithms (right).

performed on the same problem by running the different algorithms with one hundred simulated graphs. The hybrid algorithm is compared with a greedy algorithm with random starting point, a greedy algorithm with multiple random starting partitions, a regularized spectral algorithm (Qin and Rohe 2013), and a greedy algorithm initialized with the spectral algorithm. All the methods include model selection in their core, except for the spectral clustering which is thus advantageously run with the true number of clusters. For all the variants of the greedy algorithm and our hybrid proposal default values were used for their parameters: initial number of clusters equal to twenty, size of the population equal to fifty, probability of mutation equal to 0.25 and maximum number of generations fixed to ten. The comparison is made in terms of normalized mutual information (NMI, Vinh et al. 2010) between the extracted and simulated clusters. The NMI allows comparing partitions with a different number of clusters, as is needed in this setting, and an NMI of 1 means a perfect match between two partitions. As expected, the greedy algorithm with random starting point suffers from quite severe under-fitting and gives an NMI around 0.55, using multistart helps a little and the solutions then are around an NMI of 0.7. The spectral algorithm does also improve with an NMI around 0.85. Eventually, the two best algorithms are the simple greedy algorithm carefully initialized (here using the results of the spectral algorithm with twenty clusters) and our proposed hybrid algorithm which recovers almost perfectly the simulated partitions in all of the simulations (93% of perfect recovery) whereas some simulations are still not perfectly recovered by the greedy algorithm with careful initialization (51% of perfect recovery).

4.2 Medium-scale mixture of multinomials simulations

As a second scenario, we focused on a mixture of multinomials. The simulation setup was as follows: 15 clusters with equal proportions were generated. The sample size was fixed to 500 and the number of possible outcomes for the multinomials to 100. The multinomial parameters were set such that each cluster has a uniform distribution on $\{1, \ldots, 100\}$ except for 10 randomly chosen outcomes that have their probabilities multiplied by 4. Eventually the number of draws for each multinomial sample was set to 50. The simulation was performed one hundred times and for each generated dataset the solutions found by the different variants of the greedy heuristic, an EM algorithm (from the **mixtools** R package) with model selection performed with AIC and BIC were recorded. We may first look at the number of clusters extracted by each algorithm. Figure 7 presents the bar graphs of the number of extracted clusters for each of the algorithms over the 100 generated datasets.

The solutions found using an EM algorithm and BIC or AIC for model selection suffer from a lot of variance. AIC gives more satisfactory results on this problem but the number of extracted clusters is still quite variable, between 10 and 22. BIC leads to too simple models with fewer than 5 clusters in all the simulations. Some of these results can be explained by the random initialization of the EM algorithm. Greedy maximization of ICL gives better results in this problem and found the correct number of clusters in around 60% of the simulations with the multistart version of the algorithm (which is a little bit better than the version seeded with a simple k-means). Eventually, the hybrid algorithm found the

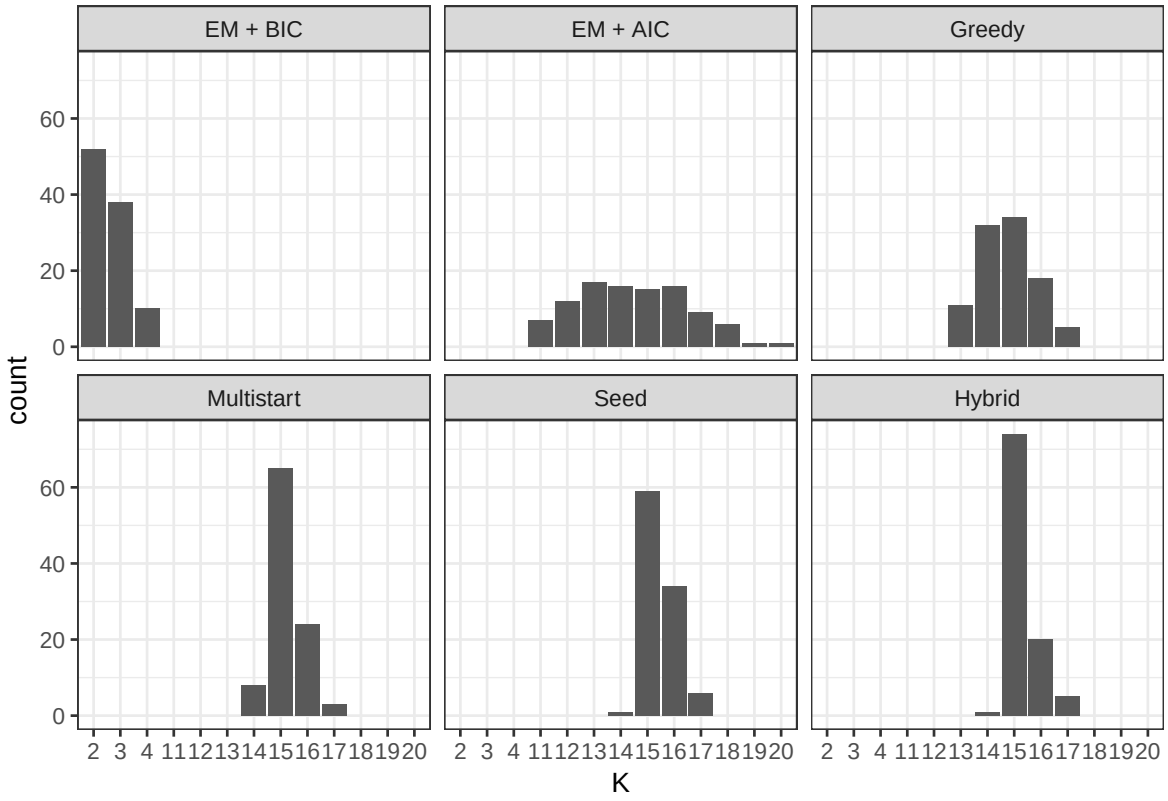

Figure 7: Bar graphs of the number of extracted clusters over one hundred simulated datasets for the different algorithms. The datasets were generated with $K = 15$.

correct number of clusters in more than 75% of the simulations and is therefore also better here. If we inspect the results with respect to the NMI with the simulated labels, or with the obtained ICL values as shown in Figure 8, the ranking of the different solutions does not differ. The hybrid algorithm leads to the best results even though the differences with the seeded version of the greedy algorithm are less important with respect to these metrics in this experiment.

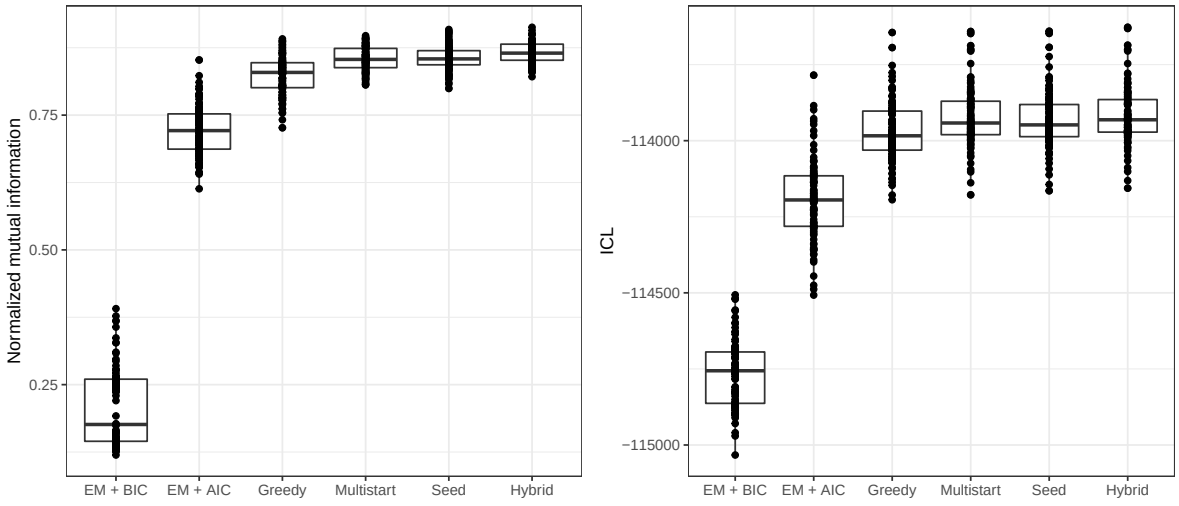

Figure 8: NMI between simulated and extracted clusters and ICL for the different algorithms on the mixture of multinomial simulation over one hundred simulations.

4.3 Clustering real network data

The performances of the proposed solution were also investigated with real datasets. Classical graph clustering datasets were first analyzed:

- **Blog**: a directed network from Adamic and Glance (2005) of hyperlinks between 1222 blogs on US politics, recorded during the 2004 presidential election,
- **Books**: a network of 105 books about US politics also published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com (edges between books represent frequent co-purchasing of books by the same buyers),
- **Jazz**: an undirected network of 198 jazz bands (Gleiser and Danon 2003),
- **Football**: an undirected network of American football games between 115 colleges during the regular Fall 2000 season (Newman and Girvan 2004).

All of these classical datasets were downloaded from Mark Newman datasets page[3]. Two co-clustering datasets were also benchmarked:

- **French parliament**: this dataset concerns the votes of 593 French deputies during a part of the current legislature and covers 1839 ballots, the data were extracted from the French national assembly open data api[4] and gathered into a binary matrix where the presence of a one indicates a positive vote of a deputy for a specific ballot.
- **Jazz bands / musicians**: is a recreation of the raw data in Gleiser and Danon (2003). These raw data were extracted by scrapping the same source namely *The Red Hot Jazz Archive*[5]. For each available band, the list of its members was extracted leading to a binary matrix of 4475 musicians and 965 bands. For all the performed analyses, we removed all the musicians that played in fewer than 3 bands and all the bands with fewer than 3 musicians, leaving a final matrix of 690 musicians and 539 bands.

These original datasets were produced for this paper and are available together with the classical network datasets in the **R** package accompanying the paper. For each of these datasets, and in order to get some information on the variability of results, we ran the algorithms 25 times, with a degree corrected SBM (dc-SBM) model for networks and degree corrected LBM (dc-LBM) model for co-clustering datasets, and the resulting ICL_{ex} values were recorded. The algorithms are the same as previously: greedy with multiple random starts, seeded greedy (spectral algorithm for dc-SBM and independent k-means on rows and columns for dc-LBM) and our proposed hybrid approach. To study the impact of the population size on the results of the hybrid algorithm, this parameter was also set to vary in $\{20, 40, 80\}$. These numbers are quite small with respect to the ones commonly encountered in pure GA, which is allowed by the use of hybridization with local search reducing the need for a large population.

The results are presented with boxplots in Figure 9 for all the methods that maximize the ICL_{ex}. For all the datasets, the best results are achieved by the hybrid algorithm with a population of 80 partitions. For each experiment, while a bigger population size leads to better results with less variation, a small population size of 20 already achieves a significant improvement over the multiple and seeded strategies. Indeed, an important performance gap in terms of ICL_{ex} is visible between the three hybrid solutions and the two others. Moreover, some datasets like Jazz, Blogs and Political books highlights the interest of the multiple restart over the seeded strategy. This is expected for the experiments with directed networks (**Blogs**, **Books**), where the seed partitions are found using an undirected network model. Thus, it advocates for the use of directed model whenever possible for these datasets. This last experiment on the proposed hybrid algorithm clearly shows a benefit of using such an approach on real data. In the next section, we illustrate the interest of the hierarchical algorithm, giving a more detailed discussion about the clustering results on real datasets.

4.4 Hierarchical analysis of real datasets

In continuity with the motivating example of Figure 2, the interest of the hierarchical procedure is illustrated on the real datasets introduced previously. Starting from the best solution of Algorithm 1, with a population size of 40, we build the hierarchy and the dendrogram for each of the examples. We start by describing the results on the four graph clustering datasets, then detailing the French parliament votes co-clustering one.

[3] available at http://www-personal.umich.edu/ mejn/netdata/
[4] available at http://data.assemblee-nationale.fr/
[5] available at http://www.redhotjazz.com/

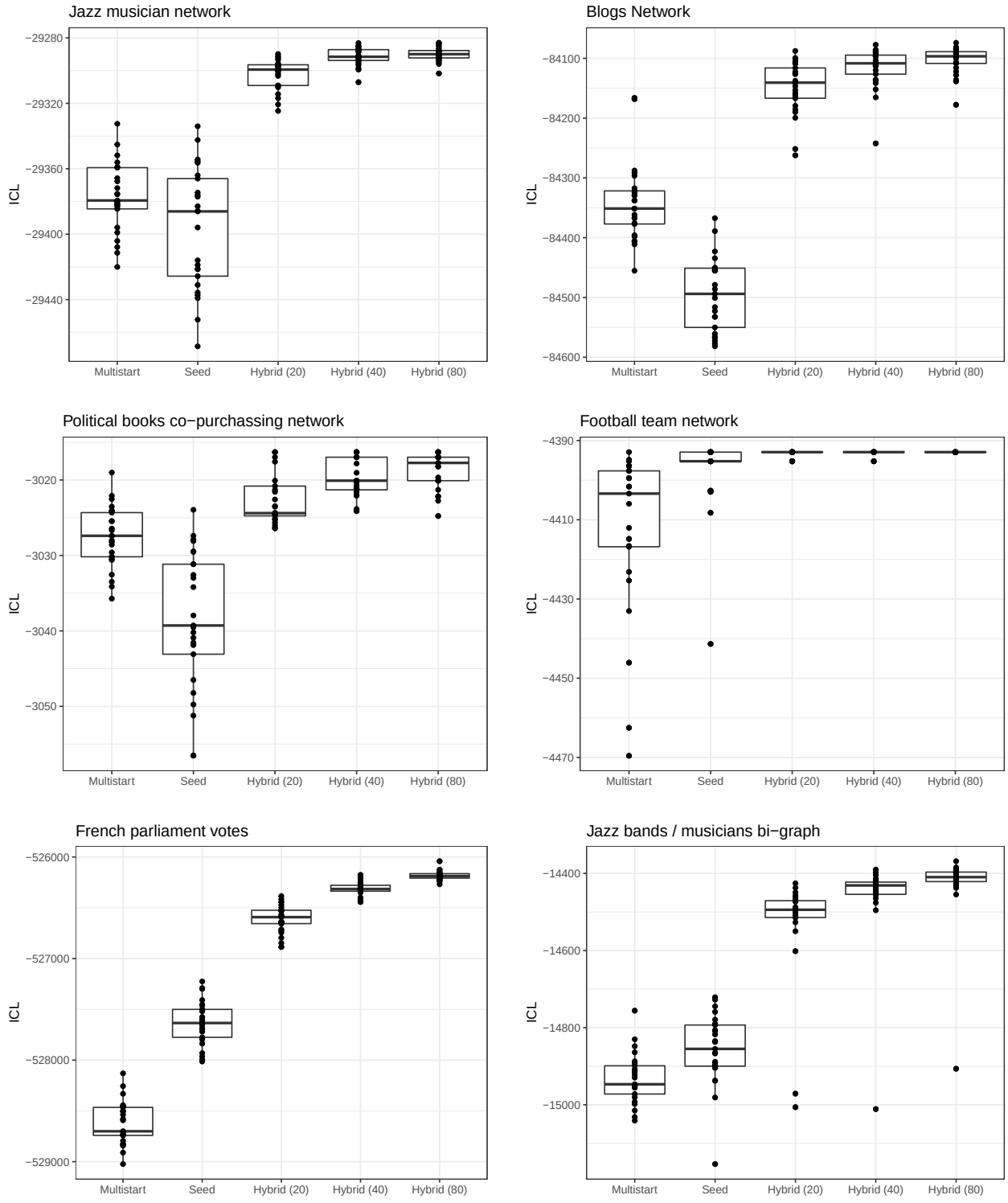

Figure 9: Boxplots of the ICL_{ex} values obtained from 25 runs of the different algorithms on the six different datasets.

Newtork clustering Figure 10 shows the results of the proposed two-step methodology with the dc-SBM as the underlying model, highlighting its analytical and visual interest. Columns represent datasets and the first row corresponds to the adjacency matrices of each network, with the rows/columns arranged per cluster numbers and the color indicating the link density between clusters. Notice that clusters are reordered according to the leaf ordering of the dendrogram, bringing *linked* clusters next to each other, enhancing the visualization of the block clustering. Next, the second row represents the cluster node link diagram, another representation of a graph clustering where the size of nodes is proportional to cluster size and the width of arrows to link density between clusters. Once again, we use the leaf ordering provided by the binary tree. The latter is then plotted as a dendrogram in the third row, emphasizing the amount of regularization (drop in α) needed for each fusion.

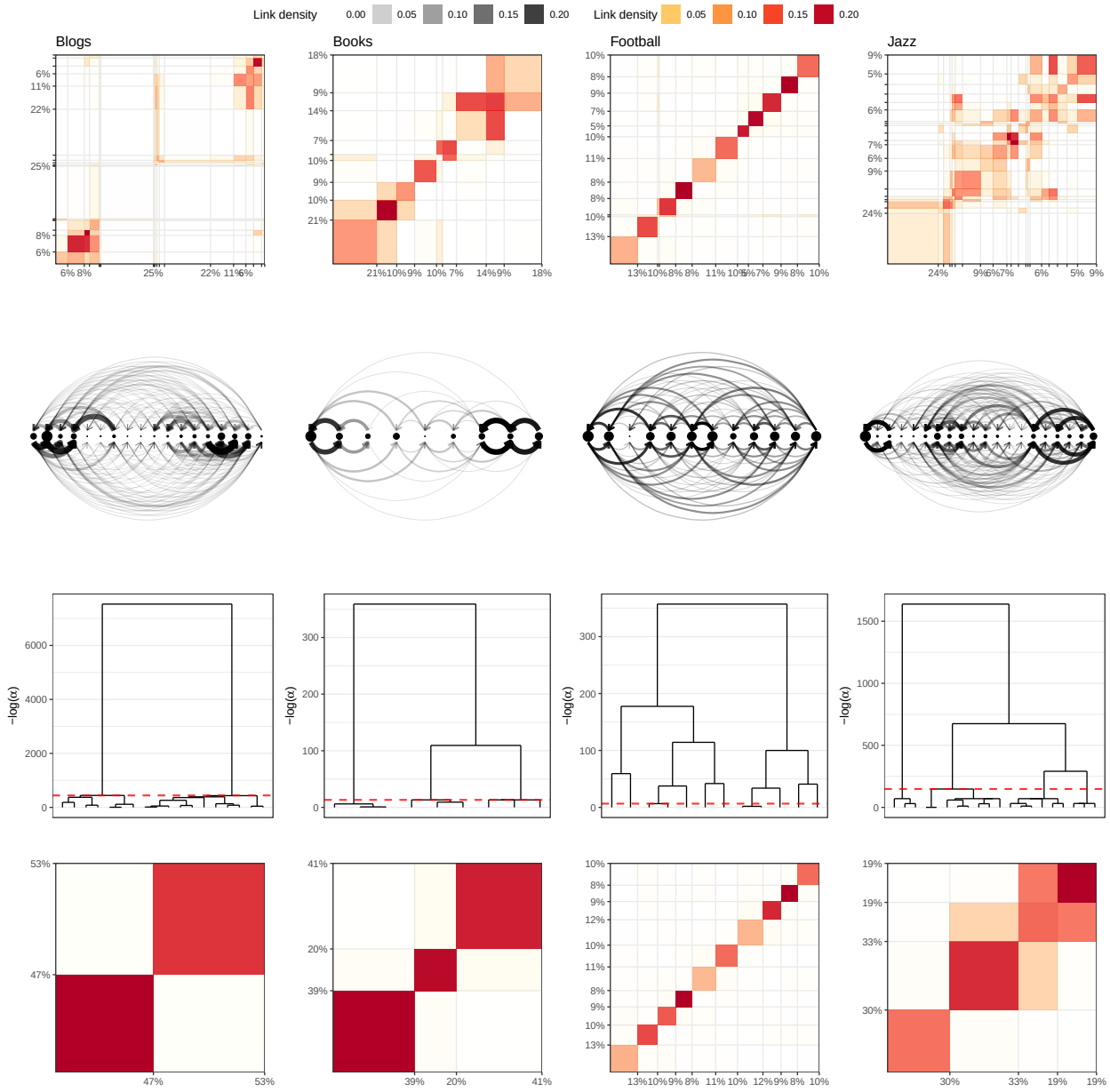

Figure 10: Illustration of the hierarchical agglomerative strategy on four real networks: blogs, books, football and jazz. First row: aggregated adjacency matrix according to the initial partition $\boldsymbol{Z}^{(K)}$, with cluster reordering given by the leaf ordering of the dendrogram. Second row: node link diagram of $\boldsymbol{Z}^{(K)}$. Third row: dendrogram of the hierarchy extracted from the initial partition. Fourth row: exploration of some clustering $\boldsymbol{Z}^{(f_h)}$ alongside the hierarchy.

In order to spot interesting levels in the dendrogram, we use a heuristic consisting in pruning the tree at a certain level $\alpha^{(f_h)}$ where the amounts of regularization needed for the next fusion is considered too important, relatively to the amount needed for past fusions. The fourth row of Figure 10 represents the same adjacency matrices as in the first row, except the new clustering $\boldsymbol{Z}_{f_h}$ is now used. For the Blogs network, starting from a solution with 18 clusters, the heuristic finds a lot of fusions for reasonable α levels, leaving 2 clusters at the selected level. The partition obtained at this level strongly aligns with the expected structure of this dataset: a divide between liberal/conservative blogs with an assortative structure of the two communities. Here, the NMI between the extracted partition and the labels provided with the dataset is 0.73. The method therefore allows to extract and analyse fine details with the initial partition and the bigger structures at another level of the dendrogram.

Likewise, for the Books dataset, the heuristic selects 3 clusters with assortative behavior which is the expected structure for this network with liberal, conservative and a few neutral books. The corresponding partition has an NMI of 0.57 with the manual labelling of the books, and displays small differences in the "neutral" cluster.

The Football network has a more pronounced and balanced community structure, with the initial partition $\boldsymbol{Z}^{(11)}$ already strongly aligned with the additional information available on the football teams, i.e. their conference structure, which corresponds to an NMI of 0.86. There are a few independent teams that do not belong to any conference, which explains the observed differences. Remarkably, the dendrogram structure of this network is more balanced without a clear jump in $\log(\alpha)$. Still, the heuristic cuts the dendrogram after the first fusion at 10 clusters and the obtained partition is also close to the expected conference structure between the teams.

As for the Jazz network, it starts with 21 clusters and we propose to cut at 4 clusters according to the heuristic, with the corresponding $\boldsymbol{Z}^{(4)}$ presenting an interesting block structure of three assortative communities and a fourth who bridges two of them. While this dataset does not possess side information to analyse the extracted structure quantitatively, the reordering of the 21 initial clusters along with the dendrogram allows for a multi-level analysis from $\boldsymbol{Z}^{(21)}$ to $\boldsymbol{Z}^{(4)}$. This gives a complementary and clear view of the structure found in this graph with finer details available at the first levels of the hierarchy.

Overall, this highlights the relevance of the proposed hierarchical agglomeration in term of clustering quality and interpretability as well.

Co-clustering on French assembly votes We illustrate the hierarchical heuristic on the French assembly votes co-clustering dataset. The initial partition $\boldsymbol{Z}^{(K)}$ found by Algorithm 1 has 116 clusters divided in 70 row clusters and 46 columns clusters. These are quite large numbers for a dataset of this size, and one might want to explore solutions with fewer row clusters. As explained above, the hierarchical algorithm can build two separate dendrograms for rows and columns, which are linked by their merging sequence $(\alpha_f)_f$. Then, using the same heuristic on this regularization sequence, we chose to cut both dendrogram at the same level, thus determining a number of row and column clusters. In this example, the chosen level leaves the same number of 13 rows and 13 columns clusters. Inspecting the row clustering, we found it consistent with the true labels, which are the political party memberships. Some members of Parliament (MPs) in different opposition groups from the left (communists, socialists) are gathered in a single cluster, whereas MPs from the majority group (LREM) are split into 5 different clusters, with some having centrists or right-wing opposition members. This agrees with the current separations and relationships in the French Parliament and the French political field.

5 Related works

This paper introduced a two-fold contribution relying on two distinct optimization strategies, for which we now highlight connections with the existing literature.

Local maxima and genetic clustering algorithms Apart from the work of Tessier et al. (2006) for the latent class model, evolutionary algorithms were proposed for Gaussian model-based clustering, maximizing the (non integrated) classification likelihood (Andrews and McNicholas 2013), and in the context of feature selection (Scrucca 2016). More generally, the specific use of GAs for clustering problems is not new (Cole 1998), and we refer to Hruschka et al. (2009) for a recent and detailed review on the subject.

Hierarchical clustering using the ICL Model-based hierarchical clustering extends the idea of non-parametric and similarity-based hierarchical clustering strategy, such as Ward's methods (Ward Jr 1963) or complete-link (Sokal and Michener 1958) and single-link (Sneath 1957) clustering. The first work of Murtagh and Raftery (1984) extends Ward's criterion as the likelihood in an isotropic Gaussian mixture models, and was later extended to the general case of spectral constraints $\boldsymbol{S}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{\Delta}_k \boldsymbol{D}_k^\top$ (Banfield and Raftery 1993; Fraley 1998). In this spirit, Zhong and Ghosh

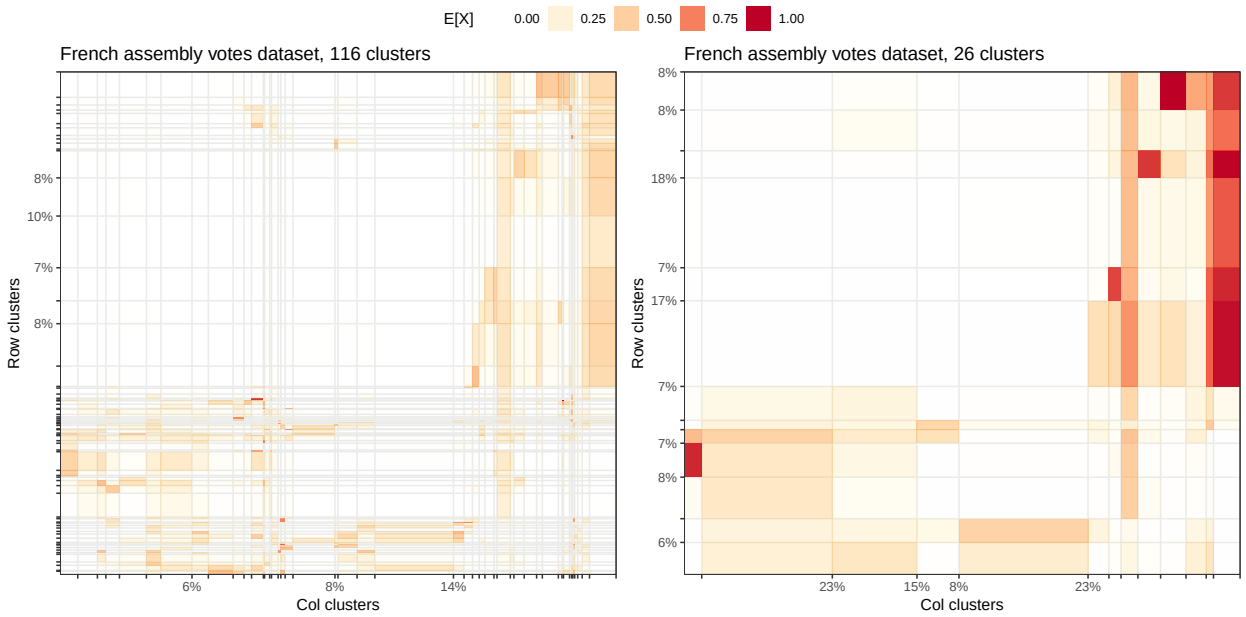

Figure 11: Block matrix representation of the **French Parliament** dataset after cluster reordering (left) and coarser clustering extraction (right).

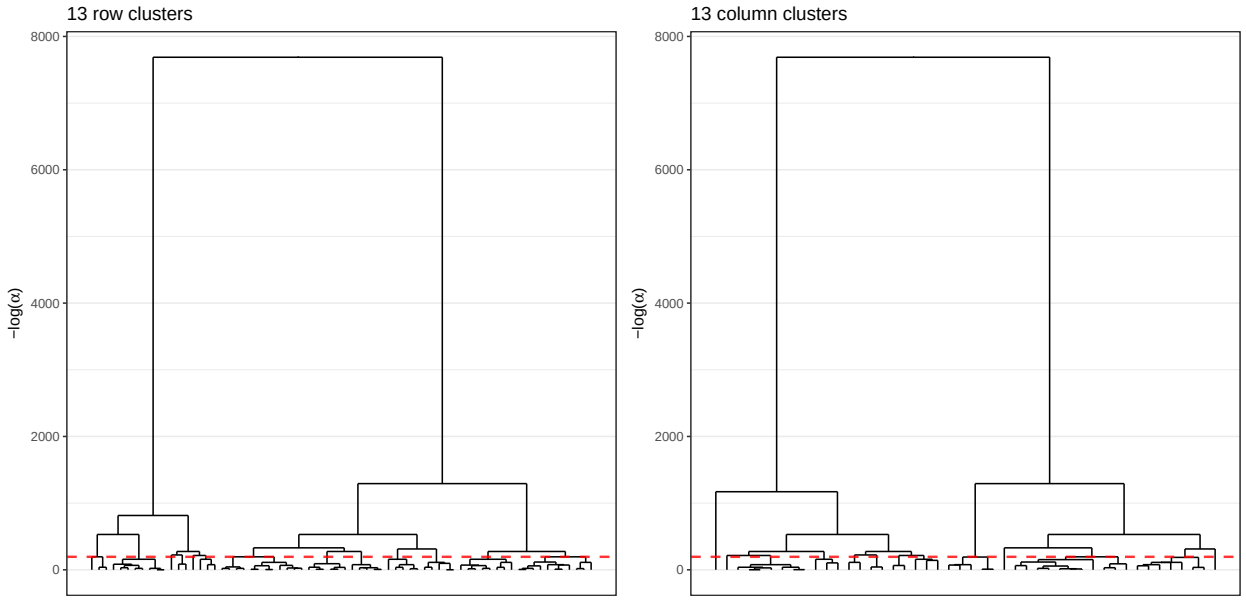

Figure 12: Row clusters dendrogram (left), and columns clusters dendrogram (right) for the **French Parliament** dataset. The dashed red line represents the height used to cut the dendrogram and to extract a coarser clustering.

(2003) proposed an extension of Ward's distance as the difference of log-likelihoods before and after a merge, along with ways to approximate it when the inference step is too costly to be done for each fusion.

More recently, model selection criteria were proposed as objective functions in hierarchical clustering algorithms. Heller and Ghahramani (2005) proposed a hierarchical Bayesian clustering algorithm, based on hypothesis testing. Marginal likelihoods of clusters are computed at each stage, using conjugate priors involving similar expressions as in the ICL_{ex}. Explicitly working with a ICL_{BIC} criterion, Baudry et al. (2010) proposed a soft hierarchical clustering algorithm for finite mixture models. Relying on an asymptotic approximation rather than exact derivation, it chooses

the merge inducing lowest posterior entropy for the cluster memberships probabilities. Thus, the latter is used to assess clustering quality, and the output is a hierarchy of soft partitions. In the context of network analysis, Peixoto (2014) proposed a greedy hierarchical clustering algorithm for a hierarchical formulation of the SBM, using another model selection criterion: the *description length*. Although the criterion differs, the author shows that it matches the ICL_{ex} when the prior on the connection probabilities of the SBM is replaced by a nested sequence of priors and hyper-priors.

6 Conclusion

In this paper, we proposed a new methodology for model-based hierarchical clustering with discrete latent variables models, based on two related contributions. The first one uses a hybrid genetic algorithm to jointly cluster the data and select the number of cluster K. The second one uses the former as an initialization and completes the hierarchy by including a Dirichlet hyper-parameter α in the objective criterion, allowing to access coarser partitions. Both methods share the ICL as an objective criterion to maximize, and their interest lies on their computational efficiency as well as the wide variety of models they can handle. Numerical experiments assessed the interest and superiority of the genetic clustering algorithm over existing methods, for some of the most common models for discrete data or graphs clustering. In addition, experiments on real datasets were conducted to illustrate the interest of the hierarchical algorithm in real-world applications for both clustering and co-clustering. The resulting hierarchy may be visualized as a dendrogram, and explored as well as the amount of regularization needed for each fusion. Moreover, we illustrated how the leaf ordering of the dendrogram may be used to reorder clusters in the initial partition, enhancing the visualization of any clustering.

Regarding further works, we plan to focus on the special case of Gaussian mixtures. Indeed, the latter perfectly fit into the DLVM framework and an exact ICL is available (Bertoletti et al. 2015). However, the difficulty of setting uninformative priors must be addressed carefully, as the clustering results is greatly influenced by these.

Acknowledgement

The authors would like to thank the editor and the two anonymous referees for their fruitful comments which helped to improve this paper.

A Deriving exact ICL: application to some DLVMs

This appendix discusses the detail of ICL_{ex} derivation for the discrete latent variable models used in the experiments of Section 4. First, we detail how the marginal distribution of $\boldsymbol{Z}$ is obtained in Equation (6): this part is common to all DLVMs. Then, the only quantity needed to explicit a particular model is $\log p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta})$, namely the supposed generative model at hand in Equation (1). The latter additionally depends on the prior distribution on $\boldsymbol{\theta}$, which is governed by the $\boldsymbol{\beta}$ hyper-parameters, and we discuss model dependent specifications of the latter.

A.1 Marginal distribution of Z: Dirichlet-Multinomial conjugacy

We recall the expression of ICL_{ex} in Equation (5):

$$\mathrm{ICL}_{ex}(\boldsymbol{Z}) = \log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\beta}) + \log p(\boldsymbol{Z} \mid \boldsymbol{\alpha}).$$

As explained in the introduction, the second term is analytically tractable when $\boldsymbol{\pi} \sim \mathrm{Dir}_K(\boldsymbol{\alpha} = (\alpha, \ldots, \alpha))$, leading to the expression in Equation (6). This is obtained by an application of standard Dirichlet-Multinomial conjugacy, which is detailed in the following for the sake of completeness.

Let us denote by $C(\boldsymbol{t})$ the normalization constant of the Dirichlet distribution:

$$C(\boldsymbol{t}) = \frac{\prod_{k=1}^K \Gamma(t_k)}{\Gamma(\sum_{k=1}^K t_k)}.$$

Then, we want to compute the following integral:

$$\begin{aligned}
p(\boldsymbol{Z} \mid \alpha) &= \int_{\boldsymbol{\pi}} p(\boldsymbol{Z} \mid \boldsymbol{\pi}) p(\boldsymbol{\pi} \mid \alpha) \, \mathrm{d}\boldsymbol{\pi}, \\
&= \int_{\boldsymbol{\pi}} \left(\prod_{i=1}^n \mathcal{M}_K(\boldsymbol{z}_i \mid 1, \boldsymbol{\pi}) \right) \mathrm{Dir}_K(\boldsymbol{\pi} \mid \boldsymbol{\alpha}) \, \mathrm{d}\boldsymbol{\pi}, \\
&= \int_{\boldsymbol{\pi}} \left(\prod_{k=1}^K \pi_k^{\sum_i z_{ik}} \right) \frac{1}{C(\boldsymbol{\alpha})} \prod_{k=1}^K \pi_k^{\alpha - 1} \, \mathrm{d}\boldsymbol{\pi}, \\
&= \frac{C(\boldsymbol{n} + \boldsymbol{\alpha})}{C(\boldsymbol{\alpha})} \int_{\boldsymbol{\pi}} \mathrm{Dir}_K(\boldsymbol{\pi} \mid \boldsymbol{\alpha} + \boldsymbol{n}) \, \mathrm{d}\boldsymbol{\pi}, \\
&= \frac{C(\boldsymbol{n} + \boldsymbol{\alpha})}{C(\boldsymbol{\alpha})},
\end{aligned}$$

with $n_k = \sum_i z_{ik}$. Thus, we obtain the desired result as:

$$\log p(\boldsymbol{Z} \mid \alpha) = \log \left\{ \frac{\Gamma(\alpha K) \prod_k \Gamma(\alpha + n_k)}{\Gamma(\alpha)^K \Gamma(n + \alpha K)} \right\} \tag{15}$$

A.2 Mixture of multinomials

Multivariate count data arise in many scientific fields in the form of frequency counts, such as word occurrence in text analysis, read counts in RNA-seq data, or species abundance data in ecology. Formally, an observation $\boldsymbol{x}_i$ is supposed to be a count vector in $\mathbb{N}^p$, where x_{ij} represents the count of modality j, with total count $c_i = \sum_{j=1}^p x_{ij}$. Here, we consider the mixture of multinomials (MoM) model, which is a mixture model for the clustering of discrete data. In a Bayesian context, we define a symmetric conjugate Dirichlet prior on each parameter $\boldsymbol{\theta}_k$ and the generative model of Equation (2) is given by:

$$\begin{aligned}
\boldsymbol{\theta}_k &\sim \mathcal{D}_p(\boldsymbol{\beta} = (\beta, \ldots, \beta)), \\
\boldsymbol{x}_i | z_{ik} = 1, \boldsymbol{\theta} &\sim \mathcal{M}_p(c_i, \boldsymbol{\theta}_k).
\end{aligned} \tag{16}$$

Then, each parameter $\boldsymbol{\theta}_k$ can be marginalized out exactly, giving a Dirichlet-multinomial distribution (Minka 2000) per cluster.

Proposition 1. *Under the mixture of multinomials model of Equation (16), we have:*

$$\log p(\boldsymbol{X}|\boldsymbol{Z}) = \sum_k \log \left(\frac{\Gamma(\beta p) \prod_{j=1}^{p} \Gamma(o_{kj} + \beta)}{\Gamma(\beta)^p \, \Gamma(c_k + \beta p)} \right) + \log B(\boldsymbol{X}), \tag{17}$$

with $o_{kj} = \sum_{i=1}^{n} z_{ik} x_{ij}$, $c_k = \sum_{j=1}^{p} o_{kj}$ *and* $B(\boldsymbol{X})$ *is a constant that does not depend on* $\boldsymbol{Z}$ *or* $\boldsymbol{\beta}$.

Proof of Proposition 1. Here, $\boldsymbol{\theta} = (\boldsymbol{\theta}_k)_k \in \Delta_p^K$ and the conditional likelihood, given $\boldsymbol{Z}$, of the MoM generative model is:

$$p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta}) = \prod_{k=1}^{K} \prod_{i=1}^{n} \mathcal{M}_p(\boldsymbol{x}_i \mid c_i, \boldsymbol{\theta}_k)^{z_{ik}},$$

We wish to integrate out the parameters $\boldsymbol{\theta} \sim \otimes_k \mathrm{Dir}_p(\boldsymbol{\beta} = (\beta, \ldots, \beta))$. A use of Fubini's formula allows leveraging Dirichlet-Multinomial conjugacy for K different integrals:

$$\begin{aligned}
p(\boldsymbol{X} \mid \boldsymbol{Z}, \beta) &= \int_{\boldsymbol{\theta}} p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{\theta}, \\
&= \prod_{k=1}^{K} \int_{\boldsymbol{\theta}_k} \prod_{i=1}^{n} \mathcal{M}_p(\boldsymbol{x}_i \mid c_i, \boldsymbol{\theta}_k)^{z_{ik}} \, \mathrm{Dir}_p(\boldsymbol{\theta}_k \mid \boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{\theta}_k, \\
&= \frac{1}{\prod_{i,j} x_{ij}!} \prod_{k=1}^{K} \int_{\boldsymbol{\theta}_k} \left(\prod_{j=1}^{p} \theta_{kj}^{\sum_i z_{ik} x_{ij}} \right) \frac{1}{C(\boldsymbol{\beta})} \prod_{j=1}^{p} \theta_{kj}^{\beta - 1} \, \mathrm{d}\boldsymbol{\theta}_k, \\
&= \frac{1}{\prod_{i,j} x_{ij}!} \prod_{k=1}^{K} \frac{C(\boldsymbol{o}_k)}{C(\boldsymbol{\beta})} \int_{\boldsymbol{\theta}_k} \mathrm{Dir}_p(\boldsymbol{\theta}_k \mid \boldsymbol{\beta} + \boldsymbol{o}_k) \, \mathrm{d}\boldsymbol{\theta}_k, \\
&= \frac{1}{\prod_{i,j} x_{ij}!} \times \prod_{k=1}^{K} \frac{\Gamma(\beta p) \prod_{j=1}^{p} \Gamma(o_{kj} + \beta)}{\Gamma(\beta)^p \, \Gamma(c_k + \beta p)},
\end{aligned}$$

with $o_{kj} = \sum_{i=1}^{n} z_{ik} x_{ij}$ and $c_k = \sum_{j=1}^{p} o_{kj}$. Finally, denote

$$B(\boldsymbol{X}) = \frac{1}{\prod_{i,j} x_{ij}!},$$

which solely depends on $\boldsymbol{X}$. Thus, taking the log concludes the proof. □

Tessier et al. (2006) and Biernacki et al. (2010) analogously derived an exact ICL criterion for the latent class model (LCM) which is closely related to the MoM model, and we emphasize that the LCM model also fits in the proposed framework. The derivation of greedy updates for merge or swap moves does not present difficulties for these models. As for setting the β hyper-parameter, uninformative prior or Jeffreys prior can be used by setting β to 1 or $\frac{1}{2}$.

A.3 Stochastic block models and degree correction

We now describe the derivations for the standard binary SBM described in Equation (3) as well as its degree-corrected variant.

Binary SBM In the binary SBM framework, x_{ij} are Bernoulli random variables indicating the presence or absence of an edge. As mentioned above, the probability of a connection between the nodes i and j only depends on their cluster assignments $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$. Hence, there is a connection probability parameter θ_{kl} for each pair of clusters. Ultimately, a Bayesian formulation of SBM is given by:

$$\begin{aligned}
\theta_{kl} &\sim \mathrm{Beta}(\eta^0, \zeta^0), \\
x_{ij} | z_{ik} z_{jl} = 1, \boldsymbol{\theta} &\sim \mathcal{B}(\theta_{kl}),
\end{aligned} \tag{18}$$

where the Beta prior on the connection probabilities is used as a conjugate of the Bernoulli distribution with hyper-parameter $\boldsymbol{\beta} = (\eta^0, \zeta^0)$. Côme and Latouche (2015) derived an exact ICL criterion for this model, relying on Beta-Bernoulli conjugacy.

Proposition 2 (Proof in Côme and Latouche (2015), Appendix A)**.** *Under the SBM model, we have:*

$$\log p(\boldsymbol{X}|\boldsymbol{Z}) = \sum_{k,l} \log\left(\frac{\Gamma(\eta^0+\zeta^0)\Gamma(\eta_{kl})\Gamma(\zeta_{kl})}{\Gamma(\eta^0)\Gamma(\zeta^0)\Gamma(\eta_{kl}+\zeta_{kl})}\right), \tag{19}$$

with $\eta_{kl} = \eta^0 + \sum_{i\neq j} z_{ik}z_{jl}x_{ij}$ *and* $\zeta_{kl} = \zeta^0 + \sum_{i\neq j} z_{ik}z_{jl}(1-x_{ij})$.

Again, a commonly accepted value for setting the hyper-parameter $\boldsymbol{\beta}$ is $\eta^0 = \zeta^0 = 1$ or $1/2$, for a uniform or Jeffreys prior respectively.

Degree correction Real world networks tend to exhibit a specific degree distribution, with some nodes having a number of links greatly superior to the average. In the SBM, all nodes inside a cluster are statistically equivalent, hence a simple SBM model may have some difficulty in reproducing such heterogeneous degree distributions. Karrer and Newman (2011) proposed a slight modification of the SBM to respect the degree sequences of the observed graph. It can be expressed as an SBM generative model, where the connection probability between two nodes now also depends on node parameters $\boldsymbol{\Phi}$ in order to introduce disparity between the nodes. This new model is called degree-corrected stochastic block model (dc-SBM). We introduce a slightly more general version of this model for directed graphs similar to the model introduced in Zhu et al. (2014), where the parameters $\boldsymbol{\Phi}^-$ and $\boldsymbol{\Phi}^+$ govern the out-degree and in-degree distributions of nodes respectively. Then, defining the degree prior distributions as in Newman and Reinert (2016) and Riolo et al. (2017), the model writes as follows:

$$\begin{aligned} \boldsymbol{\Omega}_{kl} &\sim \mathcal{E}(\beta^{-1}), \\ \boldsymbol{\Phi}^+_k,\, \boldsymbol{\Phi}^-_k \mid \boldsymbol{Z} &\sim \mathcal{U}(\mathbb{S}_k), \\ x_{ij}|z_{ik}z_{jl}=1, \boldsymbol{\Omega}, \boldsymbol{\Phi} &\sim \mathcal{P}(\Phi^-_i \Omega_{kl} \Phi^+_j). \end{aligned} \tag{20}$$

Here, $\boldsymbol{\Phi}^{\cdot}_k = (\Phi^{\cdot}_i)_{i:z_{ik}=1}$, and $\mathbb{S}_k = n_k\Delta_{n_k}$ the rescaled simplex of dimension (n_k-1) induced by the constraints $\sum_i \Phi^{\cdot}_i z_{ik} = n_k$. The latter must be set for the model to be identifiable. In this model the Bernoulli distribution of edges is replaced by a Poisson, in part to ease the computations, and the exponential distribution is used to leverage standard Gamma-Poisson conjugacy as $\mathcal{E}(\beta^{-1}) = \Gamma(1,\beta)$. Thus, the ICL_{ex} of this model can be derived as detailed in the following proposition.

Proposition 3. *Under the dc-SBM model we have:*

$$\begin{aligned} \log p(\boldsymbol{X}|\boldsymbol{Z}) = &\sum_k \log\left(\frac{(n_k-1)!\, n_k^{dg^+_k}}{(n_k+dg^+_k-1)!}\frac{(n_k-1)!\, n_k^{dg^-_k}}{(n_k+dg^-_k-1)!}\right) \\ &+ \sum_{k,l}\log\left(\frac{(\nu_{kl})!\,\beta^{\nu_{kl}}}{(\beta n_k n_l+1)^{\nu_{kl}+1}}\right) + \log B(\boldsymbol{X}), \end{aligned} \tag{21}$$

where $\nu_{kl} = \sum_{i,j} z_{ik}z_{jl}x_{ij}$ *is the total counts in block* (k,l), $d^-_i = \sum_j x_{ij}$ *and* $d^+_j = \sum_i x_{ij}$ *correspond to node* i *out-degree and in-degree respectively, and* dg^-_k, dg^+_k *to their sums in cluster* k. $B(\boldsymbol{X})$ *is a constant detailed in the Appendix, that does not depend on* $\boldsymbol{Z}$ *or* $\boldsymbol{\beta}$.

Proof of proposition 3. Putting $\boldsymbol{\theta} = (\boldsymbol{\Phi}^+, \boldsymbol{\Phi}^-, \boldsymbol{\Omega})$, the conditional likelihood, given $(\boldsymbol{Z}, \boldsymbol{\theta})$, of the generative model described in Equation (20) writes as:

$$\begin{aligned} p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta}) &= \prod_{i,j}^{N}\prod_{k,l}^{K} \mathcal{P}(x_{ij} \mid \Phi^+_i \Phi^-_j \Omega_{kl})^{z_{ik}z_{jl}}, \\ &= \frac{1}{\prod_{i,j} x_{ij}!} \times \prod_{i,j}^{N} (\Phi^+_i \Phi^-_j)^{x_{ij}} \prod_{k,l}^{K} \Omega_{kl}^{z_{ik}z_{jl}x_{ij}} \exp(\Phi^+_i \Phi^-_j \Omega_{kl} z_{ik} z_{jl}), \\ &= \frac{1}{\prod_{i,j} x_{ij}!} \times \prod_i (\Phi^+_i)^{d^+_i} \times \prod_i (\Phi^-_i)^{d^-_i} \times \prod_{k,l} \Omega_{kl}^{\nu_{kl}} \exp(-n_k n_l \Omega_{kl}), \end{aligned} \tag{22}$$

$d^-_i = \sum_j x_{ij}$, $d^+_j = \sum_i x_{ij}$ and $\nu_{kl} = \sum_{i,j} z_{ik}z_{jl}x_{ij}$. Calculating the ICL_{ex} implies to integrate over $\boldsymbol{\theta}$. Notice that Equation (22) is separable as the product of two parts, one depending on $\boldsymbol{\Phi}$ and the other on $\boldsymbol{\Omega}$. In the following, we detail calculations separately for both parts.

Integrating over $\mathbf{\Phi}$ Recall that $\mathbf{\Phi}_k^{\cdot} = (\phi_i^{\cdot})_{i:z_{ik}=1}$ and that:

$$p(\mathbf{\Phi}_k^{\cdot}) = \frac{1}{\text{vol}(\mathcal{S}_k)} \mathbb{1}_{\mathcal{S}_k}(\mathbf{\Phi}_k^{\cdot}), \quad \text{with } \mathbb{S}_k = \left\{ \mathbf{\Phi}_k^{\cdot} \in \mathbb{R}^{n_k} : \sum_{i:z_{ik}=1} \frac{\Phi_i^{\cdot}}{n_k} = 1 \right\}, \tag{23}$$

which is simply the simplex of dimension $(n_k - 1)$, rescaled by a factor n_k. Hence the volume of $\mathbb{S}_k$ is given by:

$$\int_{\mathbb{S}_k} \mathrm{d}\mathbf{\Phi}_k^{\cdot} = \frac{n_k^{n_k}}{(n_k-1)!}. \tag{24}$$

The situation is symmetric for $\mathbf{\Phi}_k^+$ or $\mathbf{\Phi}_k^-$. Thus, calculations are detailed only for the former. One needs to compute:

$$\begin{aligned}
\frac{(n_k-1)!}{n_k^{n_k}} \int_{\mathbb{S}_k} \prod_{i:z_{ik}=1} \left(\Phi_i^+\right)^{d_i^+} \mathrm{d}\mathbf{\Phi}_k^+ &= \frac{(n_k-1)!}{n_k^{n_k}} n_k^{d_k^+} \int_{\mathbb{S}_k} \prod_{i:z_{ik}=1} \left(\frac{\Phi_i^+}{n_k}\right)^{d_i^+} \mathrm{d}\mathbf{\Phi}_k^+, \\
&= \frac{(n_k-1)!}{n_k^{n_k}} n_k^{d_k^+} \int_{\Delta_{n_k}} \prod_{i:z_{ik}=1} (x_i)^{d_i^+} \left|n_k \mathbf{I}_{n_k}\right| \mathrm{d}\mathbf{x}, \\
&= \frac{n_k^{n_k}}{n_k^{n_k}} (n_k-1)!\, n_k^{d_k^+} C(\boldsymbol{a}_k) \int_{\Delta_{n_k}} \mathrm{Dir}_{n_k}\left(\boldsymbol{x} \mid \boldsymbol{a}_k = (d_i+1)_{i:z_{ik}=1}\right) \mathrm{d}\mathbf{x}, \\
&= (n_k-1)!\, n_k^{d_k^+} \frac{\prod_{i:z_{ik}=1} d_i^+!}{(n_k + d_k^+ - 1)!}, \\
&= \frac{(n_k-1)!}{(n_k + d_k^+ - 1)!} n_k^{d_k^+} \prod_{i:z_{ik}=1} d_i^+!,
\end{aligned} \tag{25}$$

with $d_k^+ = \sum_i z_{ik} d_i^+$. Then,

$$\begin{aligned}
\int_{\Phi^+} p(\Phi^+) \prod_i \left(\Phi_i^+\right)^{d_i^+} \mathrm{d}\mathbf{\Phi}^+ &= \int_{\prod_k \mathbb{S}_k} \prod_k \frac{(n_k-1)!}{n_k^{n_k}} \prod_{i:z_{ik}=1} \left(\Phi_i^+\right)^{d_i^+} \mathrm{d}(\mathbf{\Phi}_1^+, \ldots, \mathbf{\Phi}_K^+), \\
&= \prod_k \frac{(n_k-1)!}{n_k^{n_k}} \int_{\mathbb{S}_k} \prod_{i:z_{ik}=1} \left(\Phi_i^+\right)^{d_i^+} \mathrm{d}\mathbf{\Phi}_k^+, \\
&= \prod_k \frac{(n_k-1)!}{(n_k + d_k^+ - 1)!} n_k^{d_k^+} \prod_i d_i^+!.
\end{aligned} \tag{26}$$

Integrating over $\mathbf{\Omega}$ This is done using a standard Gamma-Poisson conjugacy in each pair of clusters. Indeed:

$$\begin{aligned}
\int_{\Omega_{kl}} p(\Omega_{kl}) \Omega_{kl}^{\nu_{kl}} \exp(-n_k n_l \Omega_{kl}) \,\mathrm{d}\Omega_{kl} &= \int_{\Omega_{kl}} \frac{1}{\beta} \Omega_{kl}^{\nu_{kl}} \exp\left(-(n_k n_l + \frac{1}{\beta})\Omega_{kl}\right) \mathrm{d}\Omega_{kl}, \\
&= \frac{\Gamma(\nu_{kl}+1)}{\beta \left(n_k n_l + \beta^{-1}\right)^{\nu_{kl}+1}}, \\
&= \frac{\nu_{kl}!\, \beta^{\nu_{kl}}}{\left(\beta n_k n_l + 1\right)^{\nu_{kl}+1}}.
\end{aligned} \tag{27}$$

Utimately, we have:

$$\begin{aligned}
p(\boldsymbol{X} \mid \boldsymbol{Z}) &= \int_{\boldsymbol{\theta}} p(\boldsymbol{X}, \boldsymbol{\theta} \mid \boldsymbol{Z}) \,\mathrm{d}\boldsymbol{\theta}, \\
&= \frac{\prod_i d_i^+! d_i^-!}{\prod_{ij} x_{ij}!} \prod_k \frac{(n_k-1)!}{(n_k + d_k^+ - 1)!} n_k^{d_k^+} \frac{(n_k-1)!}{(n_k + d_k^- - 1)!} n_k^{d_k^-} \\
&\qquad \times \prod_{k,l} \frac{\nu_{kl}!\, \beta^{\nu_{kl}}}{\left(\beta n_k n_l + 1\right)^{\nu_{kl}+1}}.
\end{aligned} \tag{28}$$

Putting

$$B(\boldsymbol{X}) = \frac{\prod_i d_i^+! d_i^-!}{\prod_{ij} x_{ij}!},$$

and noticing that the latter does not depend on the partition $\boldsymbol{Z}$ concludes the proof. □

Contrary to the previous models where proper Jeffreys or uniform prior could be used, the exponential distribution does not admit a conventional uninformative prior. An acceptable solution to fix β is, however, proposed in Newman and Reinert (2016), where the authors use the mean connection probability of the network. From a practical point of view, deriving swap and merge updates is also quite easy for these models, even though some care is needed to avoid unnecessary computations (Côme and Latouche 2015, Appendices B and C) and can be done efficiently using sparse matrices.

A.4 Co-clustering and latent block model

Co-clustering aims at clustering simultaneously the rows and columns of a data matrix $\boldsymbol{X}$ of size $n \times p$ into homogeneous groups. For example, in text analysis one may be interested into grouping documents and words together. The latent block model (LBM, Govaert and Nadif 2010), introduced in Equation (4), is a popular generative model to perform such task, forming a flexible class of models depending on the supposed observational model (Wyse et al. 2017). The main feature of the LBM is its block generation hypothesis:

$$\begin{aligned} \boldsymbol{z}_i^r &\sim \mathcal{M}_{K_r}(1, \boldsymbol{\pi}^r), \ \boldsymbol{z}_j^c \sim \mathcal{M}_{K_c}(1, \boldsymbol{\pi}^c), \\ x_{ij} \mid z_{ik}^r z_{jl}^c = 1, \boldsymbol{\theta} &\sim p(\cdot \mid \boldsymbol{\theta}_{kl}). \end{aligned}$$

Here, $\boldsymbol{Z}^r$ and $\boldsymbol{Z}^c$ are binary matrices defining a partition of the n rows in K_r clusters and of the p columns into K_c clusters respectively. The LBM may be handled similarly as other DLVMs, with a slight variation of the prior to handle the bipartition aspect:

$$p(\boldsymbol{\pi} \mid \alpha) = \mathrm{Dir}_{K_r}(\boldsymbol{\pi}^r \mid \alpha) \times \mathrm{Dir}_{K_c}(\boldsymbol{\pi}^c \mid \alpha). \tag{29}$$

With such a prior, the likelihood of the bipartition integrated with respect to $\boldsymbol{\pi}$ is factorized $p(\boldsymbol{Z} \mid \alpha) = p(\boldsymbol{Z}^r \mid \alpha) p(\boldsymbol{Z}^c \mid \alpha)$ and writes as:

$$p(\boldsymbol{Z} \mid \alpha) = \frac{\Gamma(\alpha K_r)}{\Gamma(\alpha)^{K_r}\, \Gamma(n + \alpha\, K_r)} \prod_{k=1}^{K_r} \Gamma(\alpha + n_k) \times \frac{\Gamma(\alpha K_c)}{\Gamma(\alpha)^{K_c}\, \Gamma(p + \alpha\, K_c)} \prod_{l=1}^{K_c} \Gamma(\alpha + m_l). \tag{30}$$

Again, this part is common to any LBM, and independent on the observational model at hand. Thus, the only quantity needed to derive ICL_{ex} for the LBM is $\log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\beta})$. The latter is often explicit when working with standard distributions for x_{ij}, leveraging on known conjugacy results. This is notably the case for standard discrete data distributions using Beta-Bernoulli or Gamma-Poisson conjugacy. Other types of distributions may be considered, *e.g.* for continuous data x_{ij}, and exponential family distributions are good candidates to derive natural conjugate priors on $\boldsymbol{\theta}_{kl}$.

Moreover, as already emphasized, the SBM and LBM are very similar and a degree-corrected LBM can also be derived for discrete Poisson observations as follows:

$$\begin{aligned} \boldsymbol{\Omega}_{kl} &\sim \mathcal{E}(\beta^{-1}), \\ \boldsymbol{\Phi}_k^r \mid \boldsymbol{Z}^r &\sim \mathcal{U}(\mathbb{S}_k), \\ \boldsymbol{\Phi}_l^c \mid \boldsymbol{Z}^c &\sim \mathcal{U}(\mathbb{S}_l), \\ x_{ij} | z_{ik}^r z_{jl}^c = 1, \boldsymbol{\Omega}, \boldsymbol{\Phi}^r, \boldsymbol{\Phi}^c &\sim \mathcal{P}(\Phi_i^r \Omega_{kl} \Phi_j^c). \end{aligned} \tag{31}$$

Then, an ICL_{ex} can be derived which closely resembles the one of Proposition 3, using similar arguments and calculations.

Proposition 4. *Under the dc-LBM model we have:*

$$\begin{aligned} \log p(\boldsymbol{X} | \boldsymbol{Z}) = &\sum_k \log\left(\frac{(n_k - 1)!\, n_k^{r_k}}{(n_k + r_k - 1)!}\right) + \sum_l \log\left(\frac{(m_l - 1)!\, m_l^{c_l}}{(m_l + c_l - 1)!}\right) \\ &+ \sum_{k,l} \log\left(\frac{\nu_{kl}!}{(\beta n_k m_l + 1)^{\nu_{kl}+1}}\right) + \log B(\boldsymbol{X}), \end{aligned} \tag{32}$$

where $\nu_{kl} = \sum_{i,j} z_{ik}^r z_{jl}^c x_{ij}$. Here, $r_k = \sum_{ij} z_{ik}^r x_{ij}$ and $c_l = \sum_{ij} z_{jl}^c x_{ij}$ correspond to row and column cluster degrees, and $B(\boldsymbol{X})$ is a constant detailed in the Appendix, that does not depend on $\boldsymbol{Z}$ or $\boldsymbol{\beta}$.

Proof of Proposition 4. Putting $\boldsymbol{\theta} = (\boldsymbol{\Omega}, \boldsymbol{\Phi}^r, \boldsymbol{\Phi}^c)$, the conditional likelihood, given $\boldsymbol{Z}$, of the generative model described in Equation (31) writes as:

$$\log p(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{j=1}^{p} \prod_{k}^{K_r} \prod_{l}^{K_c} \mathcal{P}(x_{ij} \mid \Phi_i^r \Phi_j^c \Omega_{kl})^{z_{ik}^c z_{jl}^r}, \tag{33}$$

Calculations for each of the term in Equation (32) are similar to Appendix A.4, with a slight difference in the $B(\boldsymbol{X})$ term. Indeed, the out (resp. in) degrees are now replaced by rows (resp. columns) degrees:

$$B(\boldsymbol{X}) = \frac{\prod_i r_i! \prod_j c_j!}{\prod_{i,j} x_{ij}!}, \tag{34}$$

with $r_i = \sum_j x_{ij}$ and $c_j = \sum_i x_{ij}$ the row (resp. columns) degrees. □

Merge and swap updates for dc-LBM closely resemble those of dc-SBM and can be derived in the same fashion. Moreover, the prior parameter β can be set using the same approach as for dc-SBM. However, dealing with bi-partitions induces some particular constraints for both the genetic and hierarchical algorithms. The next section details how they can be extended to co-clustering.

A.5 Dealing with bipartitions

Genetic algorithm The hybrid algorithm presented in Section 2 can be easily extended to the co-clustering problem. The latter simultaneously seeks for a partition of the n rows and p columns of a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$. In this case, we work with a partition $\mathcal{P}$ of $\{1, \ldots, n+p\}$ with the additional constraints that it decomposes into two disjoint sets of clusters that corresponds to a partition of $\{1, \ldots, n\}$ and $\{n+1, \ldots, n+p\}$ respectively (one for the rows and one for the columns):

$$\mathcal{P} = \left\{ \boldsymbol{C}_1^r, \ldots, \boldsymbol{C}_{K_r}^r, \boldsymbol{C}_1^c, \ldots, \boldsymbol{C}_{K_c}^c \right\} : \begin{cases} \bigcup_k \boldsymbol{C}_k^r &= \{1, \ldots, n\}, \\ \bigcup_l \boldsymbol{C}_l^c &= \{n+1, \ldots, n+p\} \end{cases}. \tag{35}$$

This constraint can be easily incorporated by defining $\mathrm{ICL}_{ex}(\mathcal{P}) = -\infty$ for partitions that do not fulfill this constraint and by initializing the algorithm with admissible solutions. This is sufficient to ensure that the obtained solutions will also be compatible with the constraints, since the admissible set of partitions is closed under the crossover and mutation operations used by the algorithm.

Hierarchical algorithm Furthermore, the hierarchical methodology of Section 3 can also be easily extended to bi-partitions. Indeed, the LBM prior

$$p(\boldsymbol{\pi} \mid \alpha) = \mathrm{Dir}_{K_r}(\boldsymbol{\pi}^r \mid \alpha) \times \mathrm{Dir}_{K_c}(\boldsymbol{\pi}^c \mid \alpha),$$

leaves a factorized integrated likelihood for $p(\boldsymbol{Z} \mid \alpha)$, with a common parameter α. Thus, the ICL_{lin} approximation of Equation (10) is still log-linear in α and writes:

$$\mathrm{ICL}_{lin}(\boldsymbol{Z}, \alpha) = (K_r - 1)\log(\alpha) + (K_c - 1)\log(\alpha) + I(\boldsymbol{Z}), \tag{36}$$

with $I(\boldsymbol{Z}) = I(\boldsymbol{Z}^r) + I(\boldsymbol{Z}^c)$ the intercepts defined in Equation (9). Hence, with the constraint that a merge cannot be done between rows and columns clusters, one can look for the best row or column fusion to do at each step, therefore building two dendrograms in parallel, with a shared $(\alpha_f)_f$ sequence.

References

Adamic, Lada A. and Natalie Glance (2005). "The Political Blogosphere and the 2004 U.S. Election: Divided They Blog". In: *Proceedings of the 3rd International Workshop on Link Discovery*. LinkKDD '05. Chicago, Illinois: ACM, pp. 36–43 (cit. on p. 15).

Akaike, Hirotugu (1974). "A new look at the statistical model identification". In: *IEEE transactions on automatic control* 19.6, pp. 716–723 (cit. on p. 3).

Andrews, Jeffrey L and Paul D McNicholas (2013). "Using evolutionary algorithms for model-based clustering". In: *Pattern Recognition Letters* 34.9, pp. 987–992 (cit. on p. 18).

Banfield, Jeffrey D and Adrian E Raftery (1993). "Model-based Gaussian and non-Gaussian clustering". In: *Biometrics*, pp. 803–821 (cit. on p. 18).

Bar-Joseph, Ziv, David K. Gifford, and Tommi S. Jaakkola (June 2001). "Fast optimal leaf ordering for hierarchical clustering ". In: *Bioinformatics* 17.1, S22–S29 (cit. on p. 12).
Bates, Douglas and Martin Maechler (2019). *Matrix: Sparse and Dense Matrix Classes and Methods*. R package version 1.2-17 (cit. on p. 4).
Baudry, Jean-Patrick et al. (2010). "Combining Mixture Components for Clustering." In: *Journal of computational and graphical statistics : a joint publication of American Statistical Association, Institute of Mathematical Statistics, Interface Foundation of North America* 9 2, pp. 332–353 (cit. on p. 19).
Bengtsson, Henrik (2019). *future: Unified Parallel and Distributed Processing in R for Everyone*. R package version 1.13.0 (cit. on p. 4).
Bertoletti, Marco, Nial Friel, and Riccardo Rastelli (2015). "Choosing the number of clusters in a finite mixture model using an exact integrated completed likelihood criterion". In: *METRON* 73.2, pp. 177–199 (cit. on pp. 3, 4, 20).
Biernacki, Christophe, Gilles Celeux, and Gerard Govaert (2000). "Assessing a mixture model for clustering with the integrated completed likelihood". In: *IEEE Transaction on Pattern Analysis and Machine Intelligence* 7, pp. 719–725 (cit. on p. 3).
Biernacki, Christophe, Gilles Celeux, and Gerard Govaert (2010). "Exact and monte carlo calculations of integrated likelihoods for the latent class model". In: *Journal of Statistical Planning and Inference* 140, pp. 2991–3002 (cit. on pp. 3, 22).
Blei, David M, Alp Kucukelbir, and Jon D McAuliffe (2017). "Variational inference: A review for statisticians". In: *Journal of the American Statistical Association* 112.518, pp. 859–877 (cit. on p. 3).
Bouveyron, Charles et al. (2019). *Model-Based Clustering and Classification for Data Science: With Applications in R*. Vol. 50. Cambridge University Press (cit. on p. 2).
Cole, R. M. (1998). *Clustering with Genetic Algorithms* (cit. on p. 18).
Côme, Etienne and Pierre Latouche (2015). "Model selection and clustering in stochastic block models based on the exact integrated complete data likelihood". In: *Statistical Modelling* 15.6, pp. 564–589 (cit. on pp. 3–5, 7, 22, 23, 25).
Corneli, Marco, Pierre Latouche, and Fabrice Rossi (2016). "Exact ICL maximization in a non-stationary temporal extension of the stochastic block model for dynamic networks". In: *Neurocomputing* 192. Advances in artificial neural networks, machine learning and computational intelligence, pp. 81 –91 (cit. on p. 4).
Daudin, J., F. Picard, and S. Robin (2008). "A mixture model for random graph". In: *Statistics and computing* 18, pp. 1–36 (cit. on p. 3).
Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1, pp. 1–38 (cit. on p. 3).
Eddelbuettel, Dirk and James Joseph Balamuta (2017). "Extending extitR with extitC++: A Brief Introduction to extitRcpp". In: *PeerJ Preprints* 5, e3188v1 (cit. on p. 4).
Eddelbuettel, Dirk and Conrad Sanderson (2014). "RcppArmadillo: Accelerating R with high-performance C++ linear algebra". In: *Computational Statistics and Data Analysis* 71, pp. 1054–1063 (cit. on p. 4).
Eiben, A. E. and J. E. Smith (2004). *Introduction to Evolutionary Computing, 2^{nd} Edition*. Springer-Verlag (cit. on pp. 5–7).
Everitt, Brian S., Sabine Landau, and Morven Leese (2011). *Cluster Analysis, Fifth Edition (Wiley Series in Probability and Statistics)*. 5th. Wiley Series in Probability and Statistics. Wiley (cit. on p. 1).
Fraley, Chris (1998). "Algorithms for model-based Gaussian hierarchical clustering". In: *SIAM Journal on Scientific Computing* 20.1, pp. 270–281 (cit. on p. 18).
Fruhwirth-Schnatter, Sylvia, Gilles Celeux, and Christian P Robert (2019). *Handbook of mixture analysis*. Chapman and Hall/CRC (cit. on pp. 3, 4).
Gelman, Andrew et al. (2004). *Bayesian data analysis*. 2nd ed. Chapman & Hall/CRC (cit. on p. 3).
Gleiser, P. M. and L. Danon (2003). "COMMUNITY STRUCTURE IN JAZZ". In: *Advances in Complex Systems* 06.04, pp. 565–573 (cit. on p. 15).
Govaert, Gérard and Mohamed Nadif (2010). "Latent Block Model for Contingency Table". In: *Communications in Statistics - Theory and Methods* 39.3, pp. 416–425 (cit. on pp. 2, 3, 25).
Heller, Katherine A and Zoubin Ghahramani (2005). "Bayesian hierarchical clustering". In: *Proceedings of the 22nd international conference on Machine learning*. ACM, pp. 297–304 (cit. on p. 19).
Hruschka, E. R. et al. (2009). "A Survey of Evolutionary Algorithms for Clustering". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 39.2, pp. 133–155 (cit. on pp. 6, 18).
Karrer, Brian and Mark EJ Newman (2011). "Stochastic blockmodels and community structure in networks". In: *Physical review E* 83.1, p. 016107 (cit. on pp. 2, 23).
Mariadassou, Mahendra, Stéphane Robin, and Corinne Vacher (2010). "Uncovering latent structure in valued graphs: a variational approach". In: *The Annals of Applied Statistics* 4.2, pp. 715–742 (cit. on p. 2).
Matias, Catherine and Stéphane Robin (2014). "Modeling heterogeneity in random graphs through latent space models: a selective review". In: *ESAIM: Proceedings and Surveys* 47, pp. 55–74 (cit. on p. 2).
McLachlan, Geoffrey and David Peel (2000). *Finite Mixture Models*. John Wiley & Sons, Inc. (cit. on p. 2).

McLachlan, Geoffrey J and Thriyambakam Krishnan (2007). *The EM algorithm and extensions*. Vol. 382. John Wiley & Sons (cit. on p. 3).
Minka, Thomas (2000). *Estimating a Dirichlet distribution* (cit. on p. 21).
Murtagh, Fionn and Adrian E Raftery (1984). "Fitting straight lines to point patterns". In: *Pattern recognition* 17.5, pp. 479–483 (cit. on p. 18).
Newman, M. E. J. and M. Girvan (2004). "Finding and evaluating community structure in networks". In: *Phys. Rev. E* 69 (2), p. 026113 (cit. on p. 15).
Newman, M. E. J. and Gesine Reinert (2016). "Estimating the Number of Communities in a Network". In: *Phys. Rev. Lett.* 117 (7), p. 078301 (cit. on pp. 3, 23, 25).
Nowicki, Krzysztof and Tom A B Snijders (2001). "Estimation and prediction for stochastic blockstructures". In: *Journal of the American statistical association* 96.455, pp. 1077–1087 (cit. on p. 2).
Peixoto, Tiago P (2014). "Hierarchical block structures and high-resolution model selection in large networks". In: *Physical Review X* 4.1, p. 011047 (cit. on p. 20).
Qin, Tai and Karl Rohe (2013). "Regularized Spectral Clustering under the Degree-Corrected Stochastic Blockmodel". In: *Proceedings of Nips* (cit. on p. 13).
R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria (cit. on p. 4).
Riolo, Maria A. et al. (2017). "Efficient method for estimating the number of communities in a network". In: *Phys. Rev. E* 96 (3), p. 032310 (cit. on pp. 3, 23).
Schwarz, Gideon (1978). "Estimating the dimension of a model". In: *The annals of statistics* 6.2, pp. 461–464 (cit. on p. 3).
Scrucca, Luca (2016). "Genetic algorithms for subset selection in model-based clustering". In: *Unsupervised Learning Algorithms*. Springer, pp. 55–70 (cit. on p. 18).
Sneath, Peter HA (1957). "The application of computers to taxonomy". In: *Microbiology* 17.1, pp. 201–226 (cit. on p. 18).
Sokal, R. R. and C. D. Michener (1958). "A statistical method for evaluating systematic relationships". In: *University of Kansas Science Bulletin* 38, pp. 1409–1438 (cit. on p. 18).
Tessier, Damien et al. (2006). *Evolutionary latent class clustering of qualitative data*. Tech. rep. (cit. on pp. 4, 6, 18, 22).
Vinh, Nguyen Xuan, Julien Epps, and James Bailey (2010). "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance". In: *Journal of Machine Learning Research* 11.Oct, pp. 2837–2854 (cit. on p. 13).
Wang, Yuchung J and George Y Wong (1987). "Stochastic blockmodels for directed graphs". In: *Journal of the American Statistical Association* 82.397, pp. 8–19 (cit. on p. 2).
Ward Jr, Joe H (1963). "Hierarchical grouping to optimize an objective function". In: *Journal of the American statistical association* 58.301, pp. 236–244 (cit. on p. 18).
Wyse, Jason, Nial Friel, and Pierre Latouche (2017). "Inferring structure in bipartite networks using the latent blockmodel and exact ICL". In: *Network Science* 5.1, 45–69 (cit. on pp. 3, 4, 25).
Zhao, Yunpeng, Elizaveta Levina, and Ji Zhu (2012). "Consistency of community detection in networks under degree-corrected stochastic block models". In: *The Annals of Statistics* 40.4, pp. 2266–2292 (cit. on p. 2).
Zhong, Shi and Joydeep Ghosh (2003). "A unified framework for model-based clustering". In: *Journal of machine learning research* 4.Nov, pp. 1001–1037 (cit. on p. 18).
Zhu, Yaojia, Xiaoran Yan, and Cristopher Moore (2014). "Oriented and degree-generated block models: generating and inferring communities with inhomogeneous degree distributions". In: *Journal of Complex Networks* 2.1, 1–18 (cit. on pp. 2, 23).
Zreik, Rawya, Pierre Latouche, and Charles Bouveyron (2016). "The dynamic random subgraph model for the clustering of evolving networks". In: *Computational Statistics* (cit. on p. 4).