Journal of Machine Learning Research 20 (2019) 1-44 Submitted 3/17; Revised 10/18; Published 3/19

Matched Bipartite Block Model with Covariates

Zahra S. Razaee RAZAEE@UCLA.EDU
Arash A. Amini AAAMINI@STAT.UCLA.EDU
Jingyi Jessica Li JLI@STAT.UCLA.EDU
University of California, Los Angeles
Department of Statistics
8125 Math Sciences Bldg., Box 951554
Los Angeles, CA 90095-1554, USA

Editor: Edo Airoldi

Abstract

Community detection or clustering is a fundamental task in the analysis of network data. Many real networks have a bipartite structure which makes community detection challenging. In this paper, we consider a model which allows for matched communities in the bipartite setting, in addition to node covariates with information about the matching. We derive a simple fast algorithm for fitting the model based on variational inference ideas and show its effectiveness on both simulated and real data. A variation of the model to allow for degree-correction is also considered, in addition to a novel approach to fitting such degree-corrected models.

Keywords: bipartite networks, community detection, stochastic block model, bipartite matching, node attributes

1. Introduction

Network analysis has been a very active area of research with applications to social sciences, biology and marketing, to name a few. A fundamental problem in network data analysis is community detection, or clustering: Given a collection of nodes and a similarity matrix among them, interpreted as the adjacency matrix of a (weighted) network, one wants to partition the nodes into clusters, or communities, of high similarity. For undirected networks, a popular model for community-structured networks is the stochastic block model (SBM) (Holland et al., 1983) and its variants (Karrer and Newman, 2011; Gopalan and Blei, 2013), which have been extensively investigated in recent years both in terms of theoretical properties and efficient fitting algorithms. See for instance Bickel and Chen (2009); Decelle et al. (2011); Rohe et al. (2011); Mossel et al. (2015); Zhao et al. (2012); Amini et al. (2013); Qin and Rohe (2013); Mossel et al. (2013); Massoulié (2014); Amini and Levina (2018); Gao et al. (2017); Abbe (2017); Jing and Rinaldo (2015); Hajek et al. (2016); Abbe et al. (2016); Gao et al. (2016) for a sample of the work. On the other hand, a natural structure is often present in many real networks, that of being bipartite, where nodes are divided into two sets, or *sides*, and only connections between nodes of different sides are allowed. Examples include networks of actors and movies, scientific papers and their authors, shoppers and products, and proteins and the genes they regulate. Block-modeling with the explicit aim of taking into account the bipartite nature of a network has received comparatively less attention. Interesting new modeling possibilities emerge in the bipartite case, chief among them being the issue of matching between the communities of the two sides.

Finding matched node communities in a bipartite network is a necessary task for network analysis in many applications, especially in biomedical sciences. For example, in molecular biology, a key question is to understand how thousands of proteins regulate their downstream genes in a collaborative manner (Davidson and Levin, 2005; Hecker et al., 2009). Provided with a protein-gene interaction network, constructed from high-throughput biological data, a much needed task is to find clusters of proteins that co-regulate a cluster of genes (Barabasi and Oltvai, 2004). Understanding this complicated protein-gene relationship will shed new light on understanding molecular mechanisms underlying diseases, such as cancers (Madhamshettiwar et al., 2012) and neurodegenerative diseases (Parikshak et al., 2015). For another example, in evolutionary biology, certain genes in two different species share common ancestors in the evolutionary history (Harvey et al., 1991). The pairwise gene conservation relationship is described by an ortholog bipartite network, where an edge connects two conserved genes, one from each species, and such two genes are referred to as *orthologs* (Liao and Zhang, 2005). How to identify two clusters of genes, one cluster in each species, that are jointly evolutionarily conserved, is a matched community detection question. For a third example, in cancer biology, researchers are often interested in linking DNA profiles with gene expression (RNA) profiles of cancer patients (Hedenfalk et al., 2001; Iwakawa et al., 2015; Robinson et al., 2015). In other words, researchers would like to learn what DNA mutations would cause what gene expression changes, so that they can design treatment strategies at the RNA or protein levels given a patient's DNA profile. It is well known that multiple mutations often have a joint effect on the expression levels of multiple genes (Vogelstein and Kinzler, 2004). Thus how to detect such a joint effect given a mutation-gene bipartite network is a matched community detection problem. Another key question in cancer biology is to discover new and rare subtypes for a given type of cancer (Sørlie et al., 2001, 2003; Banerji et al., 2012; Wang et al., 2011). To address this question, one can consider the bipartite gene-patient network where the edges are mutation status (or the expression value) of a gene in a patient. This is clearly a matched community detection problem where the co-cluster of the genes and patients determine the cancer subtypes.

In addition to biomedical sciences, finding matched clusters in a bipartite network also has broad applications in social sciences. For example, consider the author–paper bipartite networks where an edge signifies the authorship. There is a matching between the authors and the papers in these networks since most authors follow a theme in their publications. The matched communities in this case correspond to high-level fields of study. As an another example, consider the Wikipedia page–user networks where an edge captures the act of editing a page. In these networks, the users also congregate in groups based on their interests and the subject matter of the pages or, say, their language. The common subject or the language constitutes a matched community pair in this case. We refer to Section 5 for a detailed analysis of some of these networks.

The problem of community detection in bipartite networks is closely related to that of co-clustering, also known as bi-clustering, which goes back at least to (Hartingan, 1972). Co-clustering refers to simultaneous clustering of the rows and columns of a matrix, the *bi-adjacency matrix* of a bipartite graph. It has been extensively used in biological applications (Cheng and Church, 2000; Madeira et al., 2010) and text mining (Dhillon, 2001, 2003; Bisson and Hussain, 2008). Recently, (Choi and Wolfe, 2014; Flynn and Perry, 2012) studied likelihood-based co-clustering. Rohe et al. (2016) proposed a spectral co-clustering algorithm for directed networks and discussed how it can be applied to bipartite setting. Often, the co-clustering formulation ignores the issue of matching of the clusters, in the sense that in general any row cluster can be in relation to any column cluster.

Another common approach is to reduce community detection in the bipartite setting to two separate instances of usual clustering of (undirected) unipartite networks, by forming *one-mode projections* of the network onto the two sides (Zhou et al., 2007). Despite a moderate reduction in the dimension (having to deal with two smaller networks), the projection approach suffers from information loss and identifiability issues (Zhou et al., 2007). Projection can also turn a structured bipartite network into unstructured unipartite ones or vice versa (Larremore et al., 2014). Another major difficulty is establishing a link between the communities on the two sides. One can come up with ad-hoc association measures between communities of the two sides, e.g., by counting links between each pair. This, however, leads to another bipartite graph on the communities, leading to the difficulty of interpretation. In effect, the problem transfers from community detection on the individual nodes, to that on the newly-discovered communities, or supernodes.

Block-modeling in the bipartite setting has recently gained more attention. Wyse et al. (2014) proposed a method to infer both community memberships as well as the number of communities in a bipartite network using a block model and an algorithm similar to the iterated conditional modes (Besag, 1986). Larremore et al. (2014) has proposed a bipartite stochastic block model (BiSBM) that built on the work of Karrer and Newman (2011) to infer bipartite community structure in both degree-corrected and uncorrected regimes by maximizing a profile likelihood over all partitions. In both cases, the issue of matching of the communities on the two sides is not the main concern.

Motivated by the matching problem, in this paper, we consider the problem of *matched community detection* in a bipartite network. In many practical examples, one either expects a one-to-one correspondence between the communities of the two sides, or it is reasonable to postulate such structure, due to ease of interpretation (Section 2). The problem of “finding communities in a bipartite network that are in one-to-one correspondence between the two sides” is what we refer to as matched community detection. In its simplest form, the model assumes that nodes belonging to matched communities have a higher probability of connection (cf. Equation (3)) and may also have correlated values for their nodal covariates. In other words, we will propose a generative model for such networks where there is a hidden matched community structure that affects the distribution of the observed network and the nodal covariates. This avoids the need for post-hoc matching of the communities: the matching is built into the model and inferred simultaneously along with the communities in the process of fitting the model. Our model is a natural extension of the well-known stochastic block model (SBM) and is discussed in detail in Section 2. We also discuss an extension of our model to allow for degree-correction (Section A.3), providing a matched version of the degree-corrected block model (DC-SBM).

In another direction, many networks come with metadata, often in the form of node attributes, or covariates. In our four motivating examples from biomedical sciences, node covariates are often available and provide useful information for detecting matched communities. In the protein-gene example, expression levels of the proteins and genes are natural node covariates (Segal et al., 2003; Bansal et al., 2006). Incorporating the covariate information will allow us to more accurately identify a group of proteins that co-regulate a cluster of genes, because it is expected that the proteins in a group should share similar expression patterns, and likewise for the genes. In the ortholog example, the expression levels of genes in each species are useful node covariates, which will help researchers better identify clusters of genes that are not only conserved between the two species but also share similar functional characteristics within each species (Li et al., 2014; Gerstein et al., 2014). In the mutation-gene network, gene expression levels are the covariates of the gene nodes. Using this covariate information will be useful for identifying the active genes as a result of mutations (Hedenfalk et al., 2001; Iwakawa et al., 2015; Robinson et al., 2015). For cancer sub-type detection example, using the patient's data such as age, gender, race, progression-free and overall survival time, the primary site of the tumor as well as its stage and grade, etc. as node covariates will help in identifying rare and often more aggressive subtypes (Cheang et al., 2009; Arvold et al., 2011; Von Minckwitz et al., 2012). For the author–paper example, the frequencies of the words in each paper can be considered as node covariates for the paper side. In the Wikipedia user–page networks, the location of the users can be informative covariates.

The potential for improving quality of the clusters by incorporating node covariates has been explored in recent work, in the context of unipartite networks (Binkiewicz et al., 2017; Zhang et al., 2016; Yan and Sarkar, 2016; Newman and Clauset, 2016). The bipartite setting adds another challenge to modeling node covariates, in particular, how to jointly model the covariates on the two sides, considering that one often has covariates of different dimensions on each side. (The extreme case is when only one side has node covariates.) We extend our proposed model to allow for the presence of node covariates that are aware of the matching between communities of the two sides. In other words, covariates corresponding to nodes in matched communities are statistically linked. The linkage can be tuned using a general cross-covariance matrix, allowing for varying degrees of covariate influence on the community detection problem (Section 2). It is worth noting that we specifically model the problems where the network and node covariates are driven by a single latent community structure. This is often a plausible assumption in many applications. In these cases, one expects to obtain more accurate community estimates by properly combining the two sources of information; our model allows for a natural incorporation of these two sources by maximizing the joint likelihood. Modeling cases where the node covariates and the network provide conflicting information about a potential clustering remains a challenging task and is outside the scope of the present paper.

To fit our models, we derive an algorithm based on the variational inference, also known as the variational Bayes (Jordan et al., 1999; Blei et al., 2017) ideas (Section 3). We derive both the degree-corrected and uncorrected versions of algorithm within the same unified framework, namely, sequential block-coordinate ascent on the variational likelihood. This in particular leads to a novel approach to fitting degree-corrected likelihoods using methods of continuous optimization (as opposed to profiling out the degree-correction parameters and optimizing over the space of discrete labels.). As part of the initialization of the algorithm, we revisit a bipartite spectral clustering algorithm, `biSC` first proposed in Dhillon (2001), and identify it as an effective algorithm for matched bipartite clustering. We show the effectiveness of our approach on simulated (Section 4) and real data, namely, page-user networks collected from Wikipedia and two sets of author-paper networks, one extracted from Arnetminer collection by Tang et al. (2012) and the other scraped from DBLP(Section 5).

To summarize, our contributions in this paper are the following:

(i) Identify the matching problem in bipartite community detection more clearly and give it the prominent role, by showing that it is possible to consider matched communities from the start in the modeling process. Bringing attention to matched bipartite clustering (or community detection) as a well-defined problem also allows us to identify an earlier spectral algorithm, namely that of Dhillon (2001), originally proposed in the context of topic modeling, as effectively solving the matched version of bipartite clustering. At present, we are unaware of any other algorithm that attempts to solve this problem directly.

(ii) Propose a natural bipartite extension of the SBM and DC-SBM: *matched bipartite stochastic block model* (`mbiSBM`), which has a latent structure of matched communities and allows for node covariates that are potentially informative about the matching (see Section 2). Some of the challenges involved in joint modeling of the node covariates of the two sides are resolved by appealing to hierarchical Bayesian modeling ideas (Gelman et al., 2003).

(iii) Show the effectiveness of the variational Bayes approach in fitting the overall `mbiSBM` model, when combined with good initialization, especially a variant of the `biSC` algorithm of Dhillon (2001). The algorithm is a block-coordinate ascent with a closed-form, fairly cheap iterations, and can be scaled to large networks.

Notation. We write $[K] := \{1, \dots, K\}$ and $\mathcal{P}_K := \{p \in \mathbb{R}_+^K : \mathbf{1}^T p = 1\}$, for the set of probability vectors on $[K]$. Here, $\mathbf{1}$ is the all-ones vector of dimension K. We identify $[K]$ with $\{0,1\}^K \cap \mathcal{P}_K$, the set of binary vectors of length K having exactly a single entry equal to one. The identification is via the so-called *one-hot* encoding: $z = k$ as an element of $[K]$ iff $z_k = 1$, treating z as element of $\{0,1\}^K \cap \mathcal{P}_K$. We will use I_d to denoted the $d \times d$ identity matrix.

The probability density function (PDF) of a multivariate Gaussian distribution with mean μ and covariance Σ is denoted as $x \mapsto N(x; \mu, \Sigma)$. The constant terms in an expression are denoted as "const.". We write $\doteq$ for equality up to additive constants. We use $[Z_1; Z_2]$ to denote the vertical concatenation of two matrices Z_1 and Z_2, having the same number of columns.

2. Matched Bipartite SBM

The stochastic block model (SBM) is a generative model for networks with communities (or blocks). In the most basic SBM, sometimes called the *planted partition model*, each node is assigned to one of the K communities and the edges are placed independently between two nodes i and j, with probability p if i and j belong to the same community, and with probability q otherwise. When $p = q$, one recovers the famous Erdős–Rényi model, where there is no genuine community structure. The interesting cases are the *assortative* model where $p > q$ and the *dissortative* model where $p < q$. Our focus in this paper is mainly on the assortative case, though the results can be easily adapted to the other case.

We start with the ingredients needed to define the matched bipartite SBM (`mbiSBM`). Assume that we have two groups of nodes $[N_1] = \{1, \dots, N_1\}$ and $[N_2] = \{1, \dots, N_2\}$, representing nodes on the two sides of a bipartite network. We assume that there is a partition $\{C_{rk}\}_{k=1}^K$ of $[N_r]$, for each $r = 1, 2$. This is our latent community structure. In referring to C_{rk}, we will use the terms *community* and *cluster* interchangeably. We assume the following implicit (true) *one-to-one matching* between these communities:

$$C_{1k} \leftrightarrow C_{2k}, \quad k = 1, \dots, K. \tag{1}$$

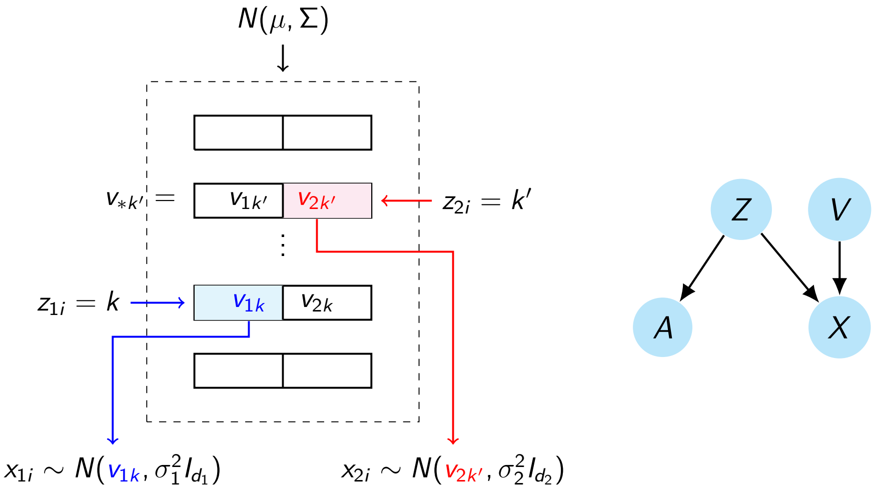


Figure 1: Schematic diagram for the hierarchical generation of node covariates (left) and the overall graphical representation of the model (right). The label z_{ri} selects which of $v_{k*} = (v_{1k}, v_{2k})$ for $k = 1, \dots, K$ we select and the index "$r \in \{1,2\}$" determines which component of v_{k*} is used to generate x_{ri}. For example, if $z_{1i} = k$ then the v_{1k} component of v_{k*} is used as the mean of x_{1i}, and similarly if you replace 1 with 2. Here, k and k' distinguish two different clusters. For example, if $z_{2i} = k'$ then $v_{2k'}$ will be the mean of x_{2i}, and so on.

To each node i in group r, we assign a community membership variable z_{ri} showing which community it belongs to:

$$z_{ri} = k \iff i \in C_{rk}, \quad \forall i \in [N_r],\ r = 1, 2.$$

Recalling the identification $[K] \cong \{0,1\}^K \cap \mathcal{P}^K$, we treat z_{ri} as both an element of $[K]$ and a binary vector of length K, hence, with some abuse of notation, $z_{ri} = k$ and $z_{rik} = 1$ are equivalent. We collect these labels in *membership matrices* $Z_r := (z_{ri} : i \in [N_r]) \in \{0,1\}^{N_r \times K}$, $r = 1, 2$, where each z_{ri}, treated as a binary vector, appears as a row in Z_r. We also let $Z = [Z_1; Z_2] \in \{0,1\}^{(N_1+N_2)\times K}$ be the *matched membership matrix* obtained by vertical concatenation of Z_1 and Z_2.

For each node i in group r, we observe a covariate vector $x_{ri} \in \mathbb{R}^{d_r}$. If we want to specify the components of this vector we write $x_{rij}, j = 1, \dots, d_r$. Let $X := (x_{ri},\ i \in [N_r],\ r = 1, 2)$. We often think of X as a matrix in $\mathbb{R}^{(N_1+N_2)\times(d_1+d_2)}$, by padding covariate vectors with zeros on the left or right: x_{1i} form rows $(x_{1i}, 0_{d_2})$ for $i \in [N_1]$ and x_{2j} form rows $(0_{d_1}, x_{2j})$ for $j \in [N_2]$.

In addition to the covariate matrix X, we also observe a bipartite network on $[N_1] \times [N_2]$ represented as a *bi-adjacency matrix* $A \in \{0,1\}^{N_1 \times N_2}$. Thus, the observed data is (X, A). We assume that given the latent community labels Z, X is independent of A. Below, we outline how each of these components are generated given Z.

2.1. Generating Covariates

To generate x_{ri}, we use a hierarchical mixture model: First we generate the mean vector $v_{rk} \in \mathbb{R}^{d_r}$ associated with each cluster C_{rk}, and then we draw x_{ri} from a normal distribution with mean v_{rk} when $z_{ri} = k$; see Figure 1. In order to model the correlation (i.e., a statistical link) between covariates of matched clusters, we draw the entire vector $v_{*k} := (v_{rk}, r = 1, 2) =$

(v_{1k}, v_{2k}) from a multivariate normal distribution with possibly nonzero covariance matrix between the two components v_{1k}, v_{2k}. We have the following model:

$$\begin{aligned} z_{ri} &\sim \text{Mult}(1, \pi_r), \\ (v_{rk}, r = 1, 2) &\overset{\text{iid}}{\sim} N(\mu, \Sigma), \quad k = 1, \dots, K. \\ (x_{ri} \mid z_{ri} = k,\, v_{rk}) &\sim N(v_{rk}, \sigma_r^2 I_{d_r}), \quad i \in [N_r],\ r = 1, 2 \end{aligned} \tag{2}$$

where the draws are independent over r and $i \in [N_r]$, on each line. Here, $\pi_r = (\pi_{r1}, \dots, \pi_{rK})$ is the prior on cluster proportions for group r ($\pi_r \in [0, 1]^K$ with $\sum_{k=1}^K \pi_{rk} = 1$). $(\sigma_r^2, r = 1, 2)$ models the variance of measurement noise in the two groups.

The idea behind the covariate generation is as follows: Each covariate x_{ri} is going to to be determined by v_{rk}, that is x_{1i} will have mean v_{1k} and x_{2j} will have mean v_{2k}, assuming nodes i and j belong to the same cluster k (a single cluster encompasses both sides of the network, based on the matching). That is, we break each $v_{*k} = (v_{1k}, v_{2k})$ into two pieces v_{1k} and v_{2k} and these two pieces determine the covariates of the two sides, x_{1i} and x_{2j}, assuming nodes i and j belong to the same cluster k.

Note that the covariates for two nodes that are in the same cluster on the same side have the same mean: If $z_{1i} = z_{1j} = k$ then both x_{1i} and x_{1j} have mean v_{1k}. On the other hand the covariates for two nodes that are in the same cluster but on different sides will have different means: If $z_{1i} = z_{2j} = k$, then x_{1i} has mean v_{1k} while x_{2j} as mean v_{2k}. These two mean vectors however are still related since they are the components of the single vector $v_{k*} = (v_{1k}, v_{2k})$ which is derived from a multivariate Gaussian distribution. That is, v_{1k} and v_{2k} are jointly Gaussian with a potential nonzero cross-covariance matrix.

To make the correlation structure in $(v_{rk}, r = 1, 2)$ more explicit, we can partition $\mu = (\mu_r, r = 1, 2)$ and Σ, so that

$$\begin{pmatrix} v_{1k} \\ v_{2k} \end{pmatrix} \overset{\text{ind}}{\sim} N\left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12}^T & \Sigma_{22} \end{pmatrix} \right], \quad k = 1, \dots, K.$$

Note that $\mu_r \in \mathbb{R}^{d_r}$. In subsection 2.5, we discuss how this model provides a statistical link between covariates of the two groups. For future reference, $v_{*k} := (v_{rk}, r = 1, 2)$ collects the matched hidden covariate means of the clusters C_{1k} and C_{2k}. On the other hand, we write $v_{r*} = (v_{rk}, k \in [K])$ which collects all the hidden covariate means for side $r = 1, 2$ of the network.

2.2. Generating the Network

Given Z, the bipartite graph is generated as follows: For each node i in $[N_1]$ and each node j in $[N_2]$, we put an edge between them with probability p if they belong to matched clusters, and with probability $q \neq p$ otherwise. With $A = (A_{ij}) \in \{0, 1\}^{N_1 \times N_2}$ denoting the resulting bi-adjacency matrix, we have

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \begin{cases} \text{Ber}(p) & z_{1i} = z_{2j} \\ \text{Ber}(q) & z_{1i} \neq z_{2j} \end{cases}, \quad i \in [N_1],\ j \in [N_2]. \tag{3}$$

Combined, (2) and (3) describe our full matched bipartite SBM model. The objective is to find the posterior probability of Z given A and $X = \{x_{ri} : i \in [N_r], r = 1, 2\}$.

Although we will focus on the simple model (3) in deriving the algorithms, it is possible to allow for a more general edge probability structure as in the usual SBM, by assuming

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Ber}(\Psi_{z_{1i},z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{4}$$

where $\Psi \in [0,1]^{K\times K}$ is a connectivity (or edge probability) matrix. Model (3) corresponds to the case where $\Psi_{kk} = p$ and $\Psi_{k\ell} = q$ for $k \neq \ell$. We will refer to this model as `mbiSBM` for matched bipartite SBM.

Remark 1. Parameter Σ in (2) is key in tuning the effect of the node covariates on community detection. Assume for simplicity that $\sigma_r^2 = 0$, $r = 1,2$. Then, when $\Sigma = 0$, $v_{*k} = \mu$ a.s. for all k, hence $x_{*i} = \mu$ for all i, and the covariates carry no information about communities. When, $\Sigma \neq 0$ there is variability in v_{*k} across k, hence community detection benefits from the covariate information. On the other hand, it is well-known that the information in the adjacency matrix A about community structure is roughly controlled by the expected degree of the network, i.e., the scaling of $Q = (p,q)$, in addition to the separation of p and q. By scaling of (p,q) we mean the following: One can take $p = a/n$ and $q = b/n$; then, how fast a and b increase as a function of n determines the difficulty of the network community detection problem. For the case $a, b = O(1)$, the so-called sparse regime, only partial recovery of the labels is possible (given only the network information), whereas when $a, b \to \infty$ one can recover with asymptotically vanishing misclassification error; at higher densities, namely $a, b \gtrsim \log n$, it is possible to exactly recover the labels for sufficiently large n. See Abbe (2017) for more details. Thus, by rescaling Σ and $Q = (p,q)$, we can control the balance of the two sources of information (i.e., the network versus node covariate information). This is explored in Section 4 through simulation studies.

2.3. Connection with the Usual SBM

Ignoring the covariate part of the model, one might wonder whether `mbiSBM`, introduced in (4), can be thought of as a sub-model of a usual SBM with perhaps increased number of communities. First, it should be clear that the model is not a usual SBM with K communities. However, it can be thought of as a SBM with $2K$ communities with *restrictions* imposed on both its membership and connectivity matrix. To see this, let us recall the matrix representation of the usual SBM with K blocks, where one has the connectivity matrix $\Psi \in [0,1]^{K\times K}$ and binary membership matrix $Z \in \{0,1\}^{N\times K}$. Such model can be compactly represented as $\mathbb{E}[A|Z] = Z\Psi Z^T$.

Now, consider model (4), and let $Z_r = (z_{ri}) \in \{0,1\}^{N_r\times K}$ for $r = 1,2$. We express the model compactly as

$$\mathbb{E}\underbrace{\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}}_{\widetilde{A}} = \underbrace{\begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix}}_{\widetilde{Z}} \underbrace{\begin{pmatrix} 0 & \Psi \\ \Psi^T & 0 \end{pmatrix}}_{\widetilde{\Psi}} \begin{pmatrix} Z_1^T & 0 \\ 0 & Z_2^T \end{pmatrix}. \tag{5}$$

given Z_1, Z_2. Letting $N := N_1 + N_2$ and defining the matrices $\widetilde{A} \in \{0,1\}^{N\times N}$, $\widetilde{Z} \in \{0,1\}^{N\times 2K}$ and $\widetilde{\Psi} \in [0,1]^{2K\times 2K}$ as in (5), it is clear that model (4) is equivalent to $\mathbb{E}[\widetilde{A}|\widetilde{Z}] = \widetilde{Z}\widetilde{\Psi}\widetilde{Z}^T$. This is a SBM with restrictions on both $\widetilde{Z}$ and $\widetilde{\Psi}$: Nodes $1,\dots,N_1$ can only belong to communities $1,\dots,K$ and nodes $N_1+1,\dots,N_1+N_2$ can only belong to communities $K+1,\dots,2K$. As for $\widetilde{\Psi}$, the restriction imposes zero connectivity among communities $1,\dots,K$ and among those of

$K+1,\dots,2K$. With these restrictions in place, we have a natural bipartite matching between communities: $\ell \leftrightarrow K+\ell$ for $\ell \in [K]$.

2.4. Degree-corrected Version

A limitation of the SBM is that nodes in the same community have the same expected degree. To allow for degree heterogeneity within communities, bringing the model closer to real networks, a common approach is to use the DC-SBM (Dasgupta et al., 2004; Karrer and Newman, 2011). It is fairly straightforward to introduce degree-correction in our setup. Consider the form of the matched SBM introduced in (4). To each node i in group r, we associate a propensity parameter $\theta_{ri} > 0$. Thus, we have additional parameters $\theta_r := (\theta_{ri},\ i \in [N_r])$ for $r = 1, 2$. The degree-corrected (DC) version of the model replaces (4) with

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Poi}(\theta_{1i}\theta_{2j}\Psi_{z_{1i},\,z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{6}$$

Replacing the Bernoulli with Poisson is for convenience in later derivations, and is common in dealing with DC-SBM (Karrer and Newman, 2011). In order for the parameters $(\theta_1, \theta_2, \Psi)$ to be identifiable, we need to agree on a normalization of θ_{ri} per each community. Here, we adopt the following:

$$\frac{1}{|C_{rk}|}\sum_{i \in C_{rk}} \theta_{ri} = 1 \iff \sum_{i=1}^{N_r}(\theta_{ri}-1)z_{rik} = 0, \quad k \in [K],\ r = 1, 2. \tag{7}$$

With this normalization, we recover the original model when $\theta_{ri} = 1$ for all i and r. Our normalization is similar to the one considered in Gao et al. (2016).

Remark 2. A normalization of the form (7) is often assumed when one considers both $\theta = (\theta_1, \theta_2)$ and Z to be deterministic unknown parameters, or alternatively when working conditioned on θ and Z. Throughout, we assume θ to be an unknown parameter. However, to be pedantic, (7) is inconsistent with i.i.d. random generation of z_{ri} from a $\text{Mult}(1, \pi_r)$ as in (2). One way to get around this is to assume that the labels are generated a priori from the product multinomial distribution described in (2) *conditioned* on the set of labels satisfying (7). We will ignore the change in the label prior this conditioning makes in deriving the algorithms. In the end, we enforce (7) in an “averaged” sense, replacing z_{rik} with the corresponding (approximate) posterior τ_{rik}, as detailed in subsection A.3. Viewed as a set of constraints on the collection of soft-labels (τ_{rik}), (7) is not that restrictive.

2.5. Covariate Correlation on Matched Clusters

One desirable feature in modeling covariates, in the context of a matched bipartite network, is the ability to gain some information about whether a pair $(i,j) \in [N_1] \times [N_2]$ belongs to a matched cluster, by just looking at their respective covariates x_{1i} and x_{2j}. Assume for the moment that there is no measurement noise, i.e., $\sigma_r^2 = 0,\ r = 1, 2$. Then, the question boils down to whether we can tell $(v_{1k}, v_{2k'})$ for $k \neq k'$ apart from (v_{1k}, v_{2k}). According to the model, (v_{1k}, v_{2k}) and $(v_{1k'}, v_{2k'})$ are independent Gaussian vectors, hence $(v_{1k}, v_{2k}, v_{1k'}, v_{2k'})$ is Gaussian with mean $(\mu, \mu) = (\mu_1, \mu_2, \mu_1, \mu_2)$ and covariance $\left(\begin{smallmatrix} \Sigma & 0 \\ 0 & \Sigma \end{smallmatrix}\right)$. Recalling the decomposition of Σ, it follows that $(v_{1k}, v_{2k'}) \sim N(\mu, \left(\begin{smallmatrix} \Sigma_{11} & 0 \\ 0 & \Sigma_{22} \end{smallmatrix}\right))$ for $k \neq k'$ whereas $(v_{1k}, v_{2k}) \sim N(\mu, \left(\begin{smallmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{smallmatrix}\right))$. As long as $\Sigma_{12} \neq 0$, these two distributions are different, hence the model is able to distinguish

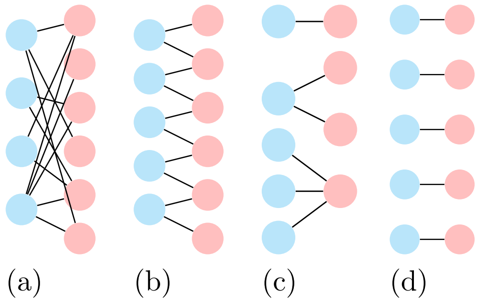


Figure 2: Possible relations between communities of the two sides, in a bipartite network. (a) and (b) are hard to interpret. Structures like (c), i.e., collections of disjoint stars, are interpretable and (d) is the simplest within this class.

between the two cases. In other words, there is information in the covariates about the matching of the clusters in the two groups. However, this information (in itself) is quite weak since it amounts to distinguishing between two multivariate Gaussian distributions, based only on a single draw from each. Fortunately, the model also carries information about the matching in the adjacency matrix A.

2.6. Interpretability and Identifiability

We alluded earlier to the merits of having a 1-1 matching between the communities of the two sides built into the model. Our main argument for the advantage of a 1-1 matching is interpretability. Figure 2 shows some possible relations that could exist between communities of the two side (each circle represents a community). The closer this relation is to a complete bipartite graph, the harder it is to interpret; Figure 2(a) is perhaps the least informative relation among the four. In Figure (b), the relation is much sparser. However, it is still hard to interpret: all communities seem to be related, albeit indirectly. We would like to argue that structures like (c) where the graph is a collection of disjoint *stars* is interpretable. One would like to fit such models, though in full generality, this seems to be a difficult task. One has to somehow control the branching numbers of the stars, which indirectly control the number of communities on either side. Thus, the problem is at least as hard as deciding the number of communities in the usual SBM. Our 1-1 matching relation, Figure 2(d), is the simplest structure of the type depicted in (c). It is in a sense a first-order approximation of the models in this class. It is easiest to fit and is the most interpretable.

3. Model Fitting

In order to fit the model, we derive algorithms based on variational inference ideas. The algorithm starts from some initial guess of the labels and parameters and proceeds to improve the likelihood via simple iterative updates to the parameters and an approximate posterior on the labels. We first discuss the case with no degree correction. The extension to the degree-corrected model is discussed in subsection A.3.

3.1. The Likelihood

Let us introduce some notation. We write $v_{*k} = (v_{rk}, r = 1, 2) \in \mathbb{R}^{d_1+d_2}$ and $v_{r*} = (v_{rk}, k \in [K]) \in \mathbb{R}^{Kd_r}$, and $V = (v_{rk},\, r = 1, 2,\, k \in [K]) \in \mathbb{R}^{K(d_1+d_2)}$. Similarly, $Z = (z_{ri},\, i \in [N_r],\, r = 1, 2)$ and $X = (x_{ri},\, i \in [N_r],\, r = 1, 2)$. (In this section, the particular matrix form of Z and X are not of interest. Z and X are simply placeholders for the collections of labels and covariates.) Let

$$y_{ij} := \mathbb{1}\{z_{1i} = z_{2j}\}, \quad z_{rik} := \mathbb{1}\{z_{ri} = k\}.$$

The joint distribution of all the variables in the model factorizes as follows:

$$\begin{aligned} p(A, X, Z, V) &= p(A|Z)\, p(X|Z, V)\, p(Z)\, p(V) \\ &= \prod_{i=1}^{N_1} \prod_{j=1}^{N_2} p(A_{ij}|z_{1i}, z_{2j}) \prod_{r=1}^{2} \prod_{i=1}^{N_r} \Big\{ p(x_{ri}|z_{ri}, v_{r*}) p(z_{ri}) \Big\} \prod_{k=1}^{K} p(v_{*k}). \end{aligned}$$

We have $p(x_{ri}|z_{ri}, v_{r*}) = \prod_{k=1}^{K} [f_r(x_{ri}; v_{rk})]^{z_{rik}}$ where $f_r(x_{ri}; v_{rk}) := N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r})$. In addition, $p(z_{ri}) = \prod_{k=1}^{K} {\pi_{rk}}^{z_{rik}}$. For the network part, we in general have

$$\ell_1(\Psi) =: \log p(A|Z) = \sum_{ij} \sum_{k\ell} z_{1ik} z_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) \tag{8}$$

where g is either the log-likelihood of the Bernoulli, $g_{\text{ber}}(p, \alpha) = \alpha \log \frac{p}{1-p} + \log(1 - p)$, or Poisson, $g_{\text{poi}}(p, \alpha) = \alpha \log p - p$.

In the special planted partition case, $\log p(A|Z)$ greatly simplifies: By breaking up over $k = \ell$ and $k \neq \ell$, we obtain

$$\begin{aligned} \ell_1(\Psi) = \log p(A|Z) &= \sum_{ij} \Big[\sum_{k} z_{1ik} z_{2jk}\, g(p, A_{ij}) + \sum_{k \neq \ell} z_{1ik} z_{2j\ell}\, g(q, A_{ij}) \Big] \\ &= \sum_{ij} y_{ij} g(p, A_{ij}) + (1 - y_{ij}) g(q, A_{ij}). \end{aligned} \tag{9}$$

where we have used $\sum_{k\ell} z_{1ik} z_{2j\ell} = 1$. The complete log-likehood of the model, i.e., assuming we observe the latent variables (Z, V), is then

$$\ell(\mu, \Sigma, \sigma, \pi, \Psi) = \ell_1(\Psi) + \ell_2(\mu, \Sigma, \sigma, \pi)$$

where $\ell_1(\Psi)$ is as defined in (8) and

$$\ell_2(\mu, \Sigma, \sigma, \pi) = \sum_{r=1}^{2} \sum_{i=1}^{N_r} \sum_{k} z_{rik} \log \big[\pi_{rk}\, f_r(x_{ri}; v_{rk}) \big] + \sum_{k} \log p(v_{*k}|\mu, \Sigma).$$

3.2. Mean-Field Approximation

Variational inference is often regarded as the approximation of a posterior distribution by solving an optimization problem (Wainwright and Jordan, 2008; Blei et al., 2017). The idea is to pick an approximation q from some tractable family of distributions over the latent variables (Z, V) and try to make this approximation as close as possible in KL divergence to the true posterior. We prefer to think of the approach as a generalization of the EM algorithm, i.e., a general approach to maximize the incomplete likelihood by maximizing a lower bound on it. This lower bound, which we call variational likelihood, also known as the evidence lower bound (ELBO) (Jordan et al., 1999), involves both the likelihood parameters and a distribution q, namely,

$$J := \mathbb{E}_q[\ell(\mu, \Sigma, \sigma, \pi, \Psi) - \log q(Z, V)]. \tag{10}$$

Here the expectation is taken, assuming $(Z, V) \sim q$. One maximizes J by alternating between maximizing over likelihood parameters $(\mu, \Sigma, \sigma, \pi, \Psi)$ and the variational posterior q. Without additional constraints, the optimization over q leads to the posterior distribution of (Z, V) given (X, A), resulting in the EM algorithm. A genuine variational inference procedure, however, imposes some simplifying constraints on q. In particular, we impose the following factorized form, often referred to as the *mean-field approximation*:

$$q(Z, V) = q_V(V) q_Z(Z), \quad q_Z(Z) = \textstyle\prod_{r,i} q_{ri}(z_{ri}), \quad q_V(V) = \prod_{k=1}^K N(v_{*k}; \tilde{\mu}_k, \tilde{\Sigma}_k) \tag{11}$$

where $q_{ri}(z_{ri}) = \prod_{k=1}^K \tau_{rik}^{z_{rik}}$ is a multinomial distribution. In keeping up with our notation we write $\tau_{ri} = (\tau_{rik},\ k \in [K])$. Note that $\tau = (\tau_{ri})$ collects the approximate posteriors on node labels. They are the key parameters in our inference.

The particular form assumed for q_V in (11) is motivated by looking at the (true) posterior of V given Z. We could have assumed a factorized form $q(Z, V) = q_Z(Z)p(V|Z)$ where $p(V|Z)$ is the the true posterior of V given Z. However, the parameters of $p(V|Z)$ have a complicated dependence on Z. We have kept the form of $p(V|Z)$ while freeing the parameters, letting them be optimized by the algorithm.

To simplify notation, let us define $\widetilde{\Gamma} := ((\widetilde{\Sigma}_k, \widetilde{\mu}_k), k = 1, \dots, K)$, collecting the parameters for the variational posterior q_V. Plugging in the variational distribution (11) into the variational likelihood (10) using expression (9) for $\ell_1(\Psi)$, after some algebra detailed in Appendix B.2, we get

$$\begin{aligned} J = \sum_{i,j} \Big[\gamma_{ij}(\tau) g(p; A_{ij}) + (1 - \gamma_{ij}(\tau)) g(q; A_{ij})\Big] + \sum_{r,i,k} \tau_{rik} \big[\beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \frac{\pi_{rk}}{\tau_{rik}}\big] \\ - \frac{1}{2} \sum_r d_r N_r \log \sigma_r^2 - \frac{K}{2} \big\{ \log |\Sigma| + \operatorname{tr}[\Sigma^{-1} S(\widetilde{\Gamma}, \mu)] \big\} + \frac{1}{2} \sum_k \log |\widetilde{\Sigma}_k| + \text{const.} \end{aligned} \tag{12}$$

where

$$\gamma_{ij}(\tau) := \mathbb{E}_{q_Z}(y_{ij}) = \sum_{k=1}^K \tau_{1ik} \tau_{2jk}, \quad \beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2} \Big[\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2 \Big], \tag{13}$$

$(\widetilde{\Sigma}_k)_{rr}, r = 1, 2$ refers to the two diagonal blocks of $\widetilde{\Sigma}_k$ of sizes $d_r \times d_r$, and

$$S(\widetilde{\Gamma}, \mu) := \frac{1}{K} \sum_{k=1}^K \big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]. \tag{14}$$

3.3. Optimizing the Variational Likelihood

We proceed to maximize J by alternating between the likelihood parameters $(\mu, \Sigma, \sigma, \pi, \Psi)$ and variational parameters $(\tau, \widetilde{\Gamma})$. Each of these two sets of parameters is also optimized by alternating maximization. In other words, the overall optimization algorithm is a block coordinate ascent. The key update is that of label distributions τ, which we describe in details below. The other updates are more or less standard and detailed in Appendix B.4.

Updating node labels (τ). To optimize τ, we use block coordinate ascent, by fixing $\tau_2 := (\tau_{2j}, j \in [N_2])$ and optimizing over $\tau_1 := (\tau_{1j}, j \in [N_1])$ and vice versa. Here we only consider optimization over τ_1 given τ_2. To simplify notation, let $h(p, q; \alpha) := g(p; \alpha) - g(q; \alpha)$. Considering only the terms in J that depend on τ, we have

$$J = \sum_{ij} \gamma_{ij}(\tau)\, h(p, q; A_{ij}) + \sum_{r,i,k} \tau_{rik} \big[\beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \frac{\pi_{rk}}{\tau_{rik}}\big] + \text{const.}$$

where const. collects terms that do not depend on τ. Let $\xi_{rik} := \beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \pi_{rk}$. Using the definition of $\gamma_{ij}(\tau) = \sum_{k=1}^K \tau_{1ik}\tau_{2jk}$,

$$J = \sum_{ij} \Big(\sum_k \tau_{1ik}\tau_{2jk}\Big)\, h(p, q; A_{ij}) + \sum_{r,i,k} \tau_{rik} \big[\xi_{rik} - \log \tau_{rik}\big] + \text{const.}$$

Now, assume further that τ_2 is constant. Then,

$$\begin{aligned} J &= \sum_i \sum_k \tau_{1ik} \Big(\sum_j \tau_{2jk}\, h(p, q; A_{ij})\Big) + \sum_{i,k} \tau_{1ik} \big[\xi_{1ik} - \log \tau_{1ik}\big] + \text{const.} \\ &= \sum_i \sum_k \tau_{1ik} \Big(\sum_j \tau_{2jk}\, h(p, q; A_{ij}) + \xi_{1ik} - \log \tau_{1ik}\Big) + \text{const.} \qquad (15) \end{aligned}$$

where const. includes terms also dependent on τ_2, but not on τ_1. The cost function above is separable over i, and for each i we have an instance of the problem given in the following lemma. Recall that $\mathcal{P}_K$ is the set of probability vectors in $\mathbb{R}^K$.

Lemma 1. *For any nonnegative vector $(a_1, \dots, a_K)$, let $f_a : \mathcal{P}_K \to \mathbb{R}$ be defined by $f_a(p) := \sum_{k=1}^K p_k(a_k - \log p_k)$. Then the maximizer of f_a over $\mathcal{P}_K$ is given by the softmax operation:*

$$\operatorname*{argmax}_{p \in \mathcal{P}_k} f_a(p) = \text{softmax}(a) := \frac{e^{a_k}}{\sum_\ell e^{a_\ell}}. \qquad (16)$$

We write the solution of Lemma 1 simply as $p_k \propto_k e^{a_k}$ where $\propto_k$ means proportional as a function of k. Then, τ_1 update is $\tau_{1ik} \propto_k \exp\big[\sum_j \tau_{2jk}\, h(p, q; A_{ij}) + \xi_{1ik}\big]$, or after unpacking ξ_{1ik},

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big[\sum_j \tau_{2jk}\, h(p, q; A_{ij}) + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big] \quad i = 1, \dots, N_1. \qquad (17)$$

The update for τ_2 is similar.

Updating $\widetilde{\Sigma}$ and $\widetilde{\mu}$. Let us define $\bar{\tau}_{rk} := \sum_{i=1}^{N_r} \tau_{rik}$ and $D_k^{-1} := \operatorname{diag}\big(\frac{\bar{\tau}_{1k}}{\sigma_1^2} I_{d_1}, \frac{\bar{\tau}_{2k}}{\sigma_2^2} I_{d_2}\big)$ Then, as a function of $\widetilde{\Sigma}$, J can be written as (see Appendix B.3)

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2} \sum_k \operatorname{tr}\big[(D_k^{-1} + \Sigma^{-1})\widetilde{\Sigma}_k\big] - \log|\widetilde{\Sigma}_k|. \tag{18}$$

This is separable over k, with each term being the likelihood of a multivariate Gaussian with covariance parameter. The maximizers are then simply $\widetilde{\Sigma}_k = (D_k^{-1} + \Sigma^{-1})^{-1}$ for $k \in [K]$.

To derive the updates for $\widetilde{\mu}$, let $\bar{x}_{rk} := \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$ and $\bar{\mu}_{rk} := \bar{x}_{rk}/\bar{\tau}_{rk}$. Then, as a function of $\widetilde{\mu}$, J can be written as (see Appendix B.3):

$$J(\widetilde{\mu}) \doteq -\frac{1}{2} \sum_k (\widetilde{\mu}_k - m_k)^T (D_k^{-1} + \Sigma^{-1})(\widetilde{\mu}_k - m_k) \tag{19}$$

where $m_k = (D_k^{-1} + \Sigma^{-1})^{-1}(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$. It is clear that the optimal value of $\widetilde{\mu}_k$ is equal to m_k, which using the optimal value of $\widetilde{\Sigma}_k$, can be written as $\widetilde{\mu}_k = \widetilde{\Sigma}_k(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$.

Updating Σ, μ and σ^2. As a function of Σ, we have $J(\Sigma) \doteq -\frac{K}{2}\big[\log|\Sigma| + \operatorname{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big]$ which is the standard Gaussian likelihood, giving the optimal value $\Sigma = S(\widetilde{\Gamma}, \mu)$. Similarly, as a function of μ, $J(\mu) \doteq -\frac{K}{2}\big[\operatorname{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big] \doteq -\frac{1}{2}\sum_k \big[(\widetilde{\mu}_k - \mu)^T \Sigma^{-1}(\widetilde{\mu}_k - \mu)\big]$ giving the optimal solution $\mu = \frac{1}{K}\sum_k \widetilde{\mu}_k$. The update for $\sigma^2 = (\sigma_1^2, \sigma_2^2)$ can be easily obtained too (see Appendix B.4)

$$\sigma_r^2 = \frac{1}{N_r d_r}\Big[\sum_k \bar{\tau}_{rk} \operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \sum_{i,k} \tau_{rik}\|x_{ri} - \widetilde{\mu}_{rk}\|^2\Big], \quad r = 1, 2. \tag{20}$$

Updating π and $\Psi \equiv (p, q)$. Updating these parameters is standard (See Appendix B.4):

$$\pi_r = \frac{(\bar{\tau}_{r1}, \dots, \bar{\tau}_{rK})}{\sum_k \bar{\tau}_{rk}}, \; r = 1, 2, \quad p = \frac{\sum_{ij} \gamma_{ij}(\tau) A_{ij}}{\sum_{ij} \gamma_{ij}(\tau)}, \quad q = \frac{\sum_{ij} \big(1 - \gamma_{ij}(\tau)\big) A_{ij}}{\sum_{ij} \big(1 - \gamma_{ij}(\tau)\big)}. \tag{21}$$

3.4. Extensions

Various extensions and improvements of the basic algorithm discussed in Section 3.3 are presented in Appendix A. A detailed derivation of an algorithm for fitting the degree-corrected model is given in Appendix A.3. The algorithm employs a novel application of the Douglas–Rachford splitting algorithm for optimization over the degree inhomogeneity parameter θ. We also provide in Appendix A.2 an extension of the algorithm to the general matched SBM model (4). Improvement to the speed for the basic algorithm of Section 3.3 is discussed in Appendix A.1.1 as well as the possibility of adding a diagonal restriction on covariate covariance matrices (Appendix A.1.2).

Overall the algorithm has 16 variations based on four options: (1) Poisson versus binomial likelihood, (2) planted partition versus general edge probability matrix, (3) full versus diagonal covariances, (4) with or without degree correction. In the following we will focus on the following defaults for the first three options: A Poisson likelihood with planted partition connectivity and full covariances.

Algorithm 1 Variational block coordinate ascent for fitting `mbiSBM`

```
1: Initialize $\tau_r$ using biSC, and $\theta_r = \mathbf{1}_{N_r}$ for $r = 1, 2$. Pick tolerance $\varepsilon \in (0, 1]$.
2: Initialize $\Sigma, \widetilde{\Sigma}_k$ with $I_{d_1+d_2}$ and $\mu, \widetilde{\mu}_k$ with 0, for $k \in [K]$, and $\sigma_r^2 = 1$ for $r = 1, 2$.
3: while not CONVERGED, nor maximum iterations reached do
4:     Update $(p, q)$ using (26) and $\pi_r, r = 1, 2$ using (21).
5:     Update $(\phi_0, \phi_1) \leftarrow (q - p,\, \log(p/q))$.
6:     if DC-version then
7:         Update $\theta_r$ by repeating (36) till convergence.
8:     end if
9:     Update $\beta_r, r = 1, 2$ using (13).
10:    $\tau_r^{\text{old}} \leftarrow \tau_r,\ r = 1, 2$.
11:    Update $\tau_1$ by repeating (34) till convergence.
12:    Update $\tau_2$ by repeating (34), with subscripts 1 and 2 switched and $A$ replaced with $A^T$, till convergence.
13:    Update the following for for $r = 1, 2$ and $k \in [K]$:        ▷ Update parameters
14:    $\bar{\tau}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik}$, and $D_k^{-1} \leftarrow \operatorname{diag}\big(\bar{\tau}_{1k} I_{d_1}/\sigma_1^2, \bar{\tau}_{2k} I_{d_2}/\sigma_2^2\big)$,
15:    $\bar{x}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$, and $\bar{\mu}_{rk} \leftarrow \bar{x}_{rk}/\bar{\tau}_{rk}$.
16:    Update $\widetilde{\Sigma}_k \leftarrow (D_k^{-1} + \Sigma^{-1})^{-1}$ and $\widetilde{\mu}_k \leftarrow \widetilde{\Sigma}_k(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$.
17:    Update $\mu \leftarrow \frac{1}{K}\sum_k \widetilde{\mu}_k$ and $\Sigma \leftarrow \frac{1}{K}\sum_{k=1}^{K}\big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]$.
18:    Update $\sigma_r^2, r = 1, 2$ using (27).
19:    CONVERGED $\leftarrow \big[\max\{\delta_1, \delta_2\} < \varepsilon/K\big]$, where $\delta_r := \vert\!\vert\!\vert \tau_r - \tau_r^{\text{old}} \vert\!\vert\!\vert_\infty, r = 1, 2$
20: end while
```

3.5. Summary of the Algorithms

Algorithm 1 summarizes the updates for fitting the proposed matched bipartite SBM model, to which we refer as `mbiSBM`. We have stated the general form of the algorithm with degree correction (DC) and covariates. Note for example that if no degree-correction is desired, θ_r remains equal to $\mathbf{1}_{N_r}$ and the iterations (34) in steps 11 and 12, for updating the label distributions (τ_r), automatically reduce to the simple update (25) (that is, the iterations converge in one step.). There are other variations available. For example, if desired, step 5 can be replaced with $(\phi_0, \phi_1) \leftarrow (\log\frac{1-p}{1-q}, \log\frac{p(1-q)}{q(1-p)})$ to use values based on a Bernoulli likelihood instead of a Poisson. Empirically, we have not found much difference between the two. With minor modifications, the algorithm can be used when only one side has covariates or without covariates for either side. The code is available on `Github` (Razaee et al.).

3.5.1. Initialization of the Algorithms

It is known that variational inference is sensitive to initialization (Blei et al., 2017). The main component of the algorithm that needs careful initialization is the matrix of (approximate) posterior node labels $\tau = [\tau_1; \tau_2]$. We propose to initialize τ using a bipartite spectral clustering algorithm, `biSC` for short, which is a variant of the approach of Dhillon (2001). The difference

Algorithm 2 Bipartite Spectral Clustering (`biSC`)

```
1: Input: bi-adjacency matrix $A \in \{0,1\}^{N_1 \times N_2}$.
2: Let $D_1 = \text{diag}(\sum_j A_{ij}, i = 1, \dots, N_1)$ and $D_2 = \text{diag}(\sum_i A_{ij}, j = 1, \dots, N_2)$.
3: Form $L = D_1^{-1/2} A D_2^{-1/2}$.
4: Let $L = USV^T$ be the SVD of $L$ truncated to $K$ largest singular values ($U \in \mathbb{R}^{N_1 \times K}$ and $V \in \mathbb{R}^{N_2 \times K}$).
5: Normalize each row of $U$ and $V$ to unit $\ell_2$ norm to get $\widetilde{U}$ and $\widetilde{V}$, resp., then form
       $$Z = \begin{bmatrix} D_1^{-1/2}\widetilde{U} \\ D_2^{-1/2}\widetilde{V} \end{bmatrix}.$$
6: Run $k$-means with $K$ clusters on the rows of $Z$.
```

between our version and that of Dhillon (2001) is that Dhillon (2001) does not normalize the rows of the singular vectors and keeps top $\lceil \log_2 K \rceil$ singular vectors, as opposed to K. We have found that row normalization greatly improves the performance, and it is fairly standard in usual (non-bipartite) Laplacian-based spectral clustering. Algorithm 2 summarizes our version.

In simulation studies, we also consider a couple of competing initializations. One interesting choice is to use the usual Laplacian-based spectral clustering, which is oblivious to the bipartite nature of the problem. For this choice, we use the regularized version described in Amini et al. (2013) as `SCP`. Note that `SCP` will be applied to the (symmetric) adjacency matrix $\widetilde{A}$; see (5). It is also possible to regularize `biSC` using similar ideas, though surprisingly, we found the simple unregularized version of `biSC` is quite robust, and we have used this simple version when reporting results.

When working with simulated data, since we have access to the true labels, we will also consider a perturbed version of truth as an initialization. Specifically, we generate from a mixture of the true label distribution and Dirichlet noise, i.e. $\tau_{ri} = \omega z_{ri} + (1-\omega)\varepsilon_{ri}$ where $\varepsilon_{ri} \sim \text{Dir}(.5\mathbf{1}_K)$. Here, we treat z_{ri}, the true label of node i in group r, as a distribution on the K labels. Parameter $\omega \in [0,1]$ measures the degree of initial perturbation towards noise. For example, with $\omega = 0.1$, about 10% of the initial labels are correct. We will refer to this initialization as ∼`rnd`, for approximately random. This initialization will act as a proxy for a "good enough" initialization and allows us to study the behavior of our variational inference procedure decoupled from specific initializations produced by spectral clustering (or other methods).

Let us say a few words about the initialization of other parameters. The algorithm is moderately sensitive to the initialization of p, q and $\pi_r, r = 1, 2$. When the quality of the initial labels (τ_1 and τ_2) is good, one can initialize these parameters, based on (τ_r), by running the corresponding updates first, as is done in Algorithm 1, lines 4–5. This is the form we suggest in practice when using the `biSC` initialization, and is used in the real data application (Section 5). However, when the quality of the initial labels is not good, for example, when using `SCP` in the simulations, p and q obtained based on initial (τ_r) can become quite close leading to numerical instability. We have found in those cases that initializing these parameters with fixed values, say $(p,q) = (0.1, 0.01)$ and π_r set to uniform distribution of $[K]$, greatly improves the stability of the algorithm. (This is since even one iteration of the algorithm could significantly improve upon initial labels.) This fixed initialization of (p, q, π_r), independent of τ_r is what we have used in Monte Carlo simulations on synthetic data, when comparing different label initializations (Section 4).

4. Simulations

In this section, we show that effectiveness of our proposed algorithm in recovering the true labels in synthetic bipartite networks. For the most part, we generate data from our proposed model (2)–(3). In the plots investigating the degree-corrected version of the algorithm, we generate from the degree-corrected version of the network described in subsection 2.4.

4.1. Data generation

Key parameters regarding covariate generation in (2) are (μ, Σ) for generating v_{*k}. We take $\mu = 0$ and $\Sigma = \nu I_{d_1+d_2}$ throughout. Varying ν (or dimensions d_r) changes the information provided by the covariates (Appendix E). Larger ν causes v_{*k} to be further apart, hence covariates are more informative. $\nu = 0$ corresponds to zero covariate information. We also fix covariate noise levels at $\sigma_r = 0.5$ for $r = 1, 2$, and the network size at $N = (N_1, N_2) = (200, 800)$.

Key parameters regarding network generation in (3) are p and q. We reparametrize our planted partition model in terms of expected average degree

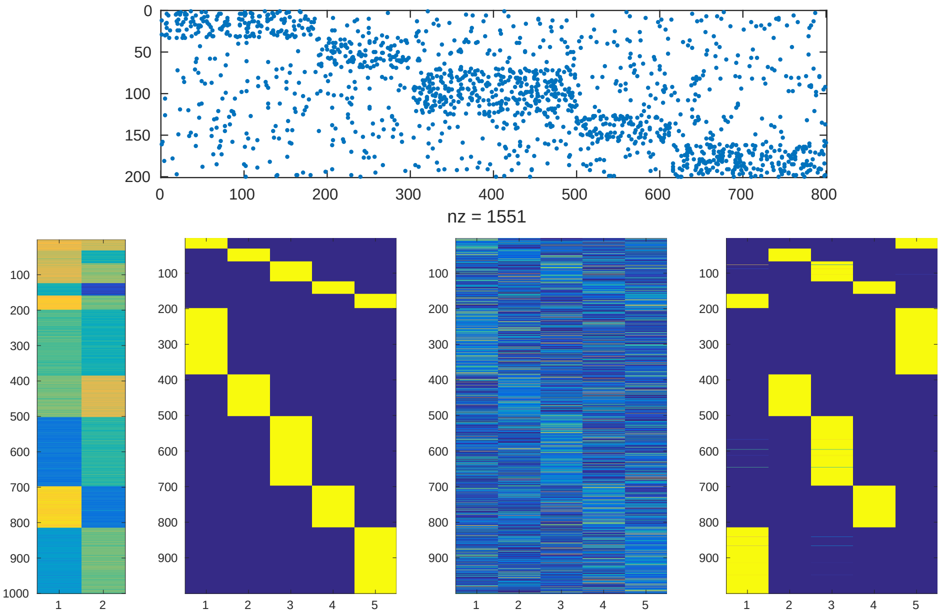
$$\lambda = \frac{2N_1N_2}{N_1 + N_2}\big[q + (p - q)\sum_k \pi_{1k}\pi_{2k}\big] \tag{22}$$

(see Appendix D) and the out-in-ratio $\alpha = q/p \in [0, 1)$. Estimation becomes harder when λ decreases (few edges) or when α increases (communities are not well separated). We fix $\alpha = 1/7$ and vary λ in the subsequent simulations.

When generating from the degree-corrected version, we draw $(\theta_i, i \in C_{rk})$ from a Pareto (i.e., power-law) distribution, for each $k \in [K]$ and $r = 1, 2$. Real networks are frequently reported to have power-law degree distributions Barabási and Albert (1999). The Pareto(a, R) in general has density $\theta \mapsto (aR^a)\theta^{-a-1}1\{\theta > a\}$, with mean $aR/(a-1)$ for $a > 1$ and variance $R^2a/[(a-1)^2(a-2)]$ for $a > 2$. Since $|C_{rk}|^{-1}\sum_{i\in C_{rk}}\theta_i$ will be approximately equal to the mean of the Pareto, and we want this average to be 1, we have to choose $R = (a-1)/a$, that is, we generate $\theta_i \overset{\text{iid}}{\sim} \text{Pareto}(a, (a-1)/a)$ for $i \in C_{rk}$. (To comply with our model specification, we further normalize θ_i for their within-community averages to be exactly one; this will have little effect since the average is already close to 1.) The variance in this case is $[a(a-2)]^{-1}$ which is decreasing in a over $(2, \infty)$. In order to get maximum degree heterogeneity (i.e., the worse case in terms of the difficulty of fitting), we take $a = 2$, corresponding to infinite variance. We note that expression (22) remains valid for the degree-corrected case without modification, assuming normalization (7); see Appendix D.

4.2. Matched NMI for evaluation

In general, we measure the accuracy of the algorithms by the normalized mutual information (NMI) between the inferred and correct communities which is defined as the mutual information of the (empirical) joint distribution of the two label assignments divided by the joint entropy (Malvestuto, 1986). NMI has a maximum value of 1 for perfect agreement and a minimum of 0 for no agreement. One could measure NMI individually between $Z_r \in \{0, 1\}^{N_r\times K}$



(a) (b) (c) (d)

Figure 3: Typical output of the algorithm. Top row: bi-adjacency matrix. Bottom row: (a) Concatenated covariate matrix $[X_1; X_2]$. (b) Concatenated true labels $[Z_1; Z_2]$. (c) Initial labels for the algorithm, Dirichlet-perturbed truth $0.1[Z_1; Z_2] + 0.9\,\mathrm{Dir}(0.5\mathbf{1}_K)$. (d) Concatenated output of the algorithm $[\tau_1; \tau_2]$.

(the true label matrix) and $\tau_r \in [0,1]^{N_r \times K}$ (the estimated soft-label matrix) for each $r = 1, 2$. However, one can also measure a *matched NMI* by concatenating the labels of two sides vertically, i.e., forming $[Z_1; Z_2]$ and $[\tau_1; \tau_2]$ and measuring a single NMI between the resulting $(N_1 + N_2) \times K$ matrices. Some thought should convince the reader that this the natural way to also measure the effectiveness of the matching between the communities of the two sides: We have a matched NMI of 1, if the true and estimated clusters on each side are in perfect agreement, and the matching between them is perfectly recovered.

4.3. Typical output

Figure 3 shows the typical output of the algorithm on the data generated from the model without degree correction (DC), i.e., $a = \infty$. Here the empirical average degree is $\hat{\lambda} = 3.1$, $K = 5$, $\nu = 10$ and $d = (2, 2)$, the dimensions of the covariates. Concatenated matrices of the true labels and the initial and final labels are shown. Vertical concatenation is used as discussed earlier, giving matrices of dimension $(N_1 + N_2) \times K$. Initial labels are the Dirichlet-perturbed truth with $\rho = 0.1$, i.e. 90% noise, as discussed in Section 3.5.1. It is interesting to note that the output of the algorithm has recovered the communities with a nontrivial permutation of the community labels.

In other words, the perturbation of the initial labels is high enough that the convergence of the algorithm cannot simply be explained by a local perturbation analysis: the algorithm has not converged to the original labels, but to a perfectly valid permuted version of the original labels. That is, $\tau_r \approx Z_r Q_r$ for $r = 1, 2$ where Q_1 and Q_2 are $K \times K$ permutation matrices. The matched NMI and misclassification rate for the algorithm are 0.98 and 0.30% in this case. If

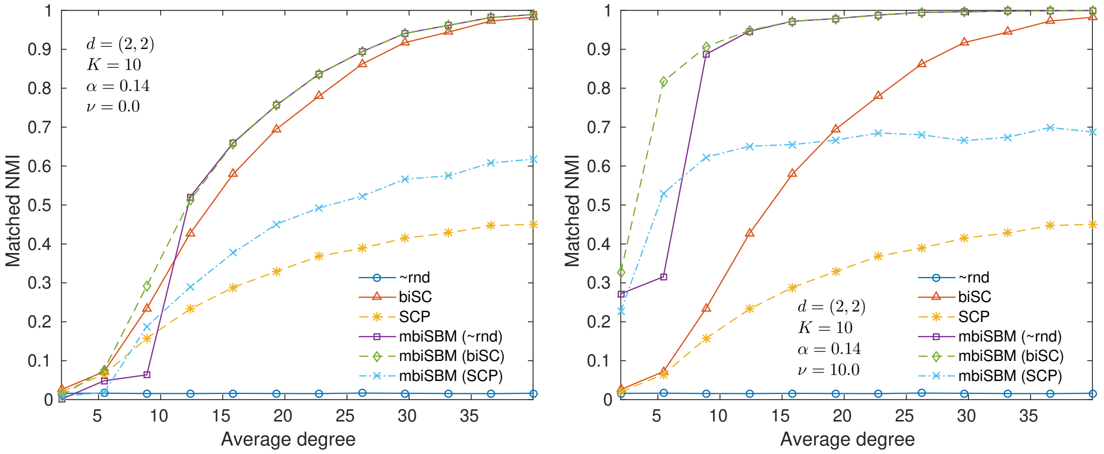


Figure 4: Effect of different initialization methods on mbiSBM with (left) $\nu = 0$ and (right) $\nu = 10$. The case $\nu = 0$ corresponds to no covariate information whereas $\nu > 0$ gives some covariates information.

one runs k-means on the concatenated matrix of covariates $[X_1; X_2]$, disregarding the network information, one gets matched NMI and misclassification rate, 0.44 and 38.50%. That is, the covariates themselves are not as informative alone as in combination with the network.

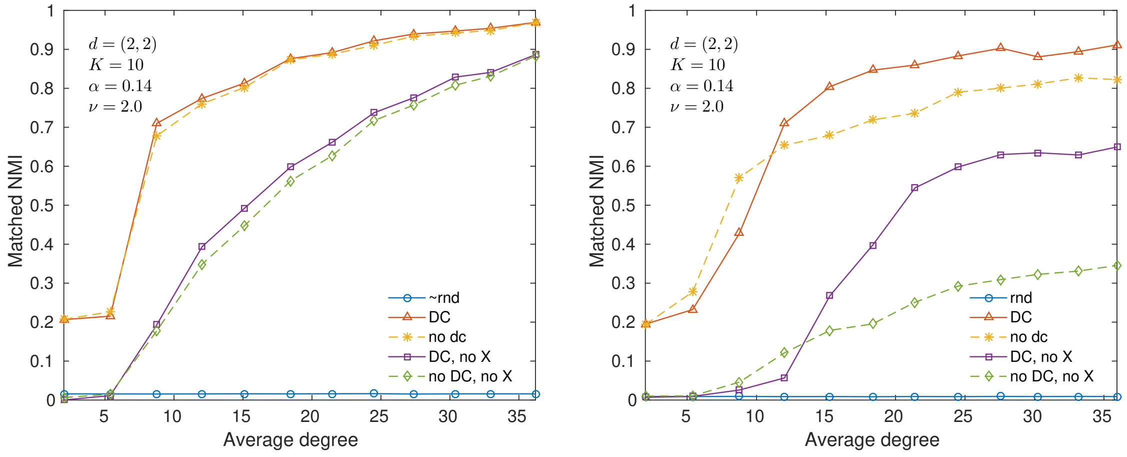
4.4. Average behavior

Figure 4 shows the mathched NMI versus average expected degree λ for various methods. The results are averaged over 50 Monte Carlo replications. Naive (regularized) spectral clustering, denoted as `SCP`, is shown in addition to `biSC` as discussed in Section 3.5.1. Moreover, the plots show our algorithm `mbiSBM`, initialized with both spectral methods and with Dirichlet-perturbed truth ($\rho = 0.1$) denoted as $\sim$`rnd`. The two plots correspond to the case with no covariate information, $\nu = 0$, and the case with covariate information $\nu = 10$. In both cases, covariate dimensions are $d = (2, 2)$, number of communities $K = 10$ and out-in-ratio is $\alpha = 1/7$. There is no degree-correction in the model or `mbiSBM` algorithm.

As can be seen, `biSC` outperforms SCP significantly. Without covariates, `mbiSBM` started with `biSC` slightly improves upon `biSC`; initializing with $\approx$ 10% truth (`mbiSBM (`$\sim$`rnd)`) has similar performance for sufficiently large λ, showing that `mbiSBM` behaves well with any sufficiently good initialization. Note also that `mbiSBM` initialized with SCP, improves upon SCP for large λ. With covariate information ($\nu = 10$), `mbiSBM` significantly outperforms `biSC` which does not incorporate the covariates.

4.5. Effect of degree correction

Figure 5 investigates the effect of employing degree-correction in the algorithm. In both plots of the figure, we are generating from the same DC-version of the model using within-community Pareto degree distribution with parameter $a = 2$ as described earlier, $d = (2, 2)$, $K = 10$, $\alpha = 1/7$, and $\nu = 2$. The difference between the two plots is how we initialize `mbiSBM` algorithm. The left panel corresponds to "Dirichlet-perturbed truth" initialization ($\rho = 0.1$) denoted as $\sim$`rnd`, whereas the right panel corresponds to completely random initialization, denoted as `rnd`. Four versions of the algorithm are considered, with or without covariate (X) incorporation, and with or without degree-correction (DC).



(a) (b)

Figure 5: Effect of the degree correction steps 6–8 in the algorithm, for a power-law network. Data is generated from DC version of the model (subsection 2.4) with Pareto distribution $p(\theta) \propto \theta^{-3},\ \theta > 2$ for degree parameters within each community. (Left) shows the results for a good initialization, the Dirichlet perturbed truth, denoted as $\sim$`rnd` (with $\approx$ 10% true labels) and the (right) shows the results for a completely random initialization, denoted as `rnd`.

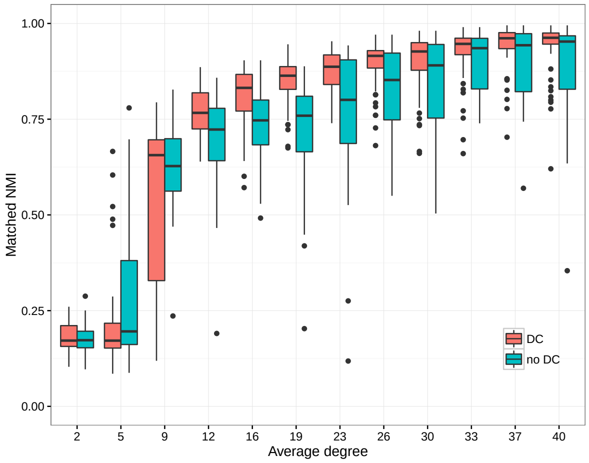


Figure 6: Effect of degree correction steps on variability, for a power-law network.

Surprisingly, as the left panel shows, with sufficiently good initialization ($\sim$`rnd`), degree-correction step of the algorithm provides only a slight improvement. However, the improvement of degree-correction is quite significant when starting from a poor initialization (`rnd`). In general, it is advisable to use the DC version since its solution has less variance. Figure 6, illustrates the algorithm with DC correction and without, in the same setup of the left panel of Figure 5, that is, both cases initialized with $\sim$`rnd` (and both incorporating covariates). Though Figure 5(a) shows that mean behaviors are close, Figure 6 shows that the distributions of the outputs are quite different, with the solution of DC version having less variability. This is expected as the DC version is solving an optimization problem with much restricted feasible region.

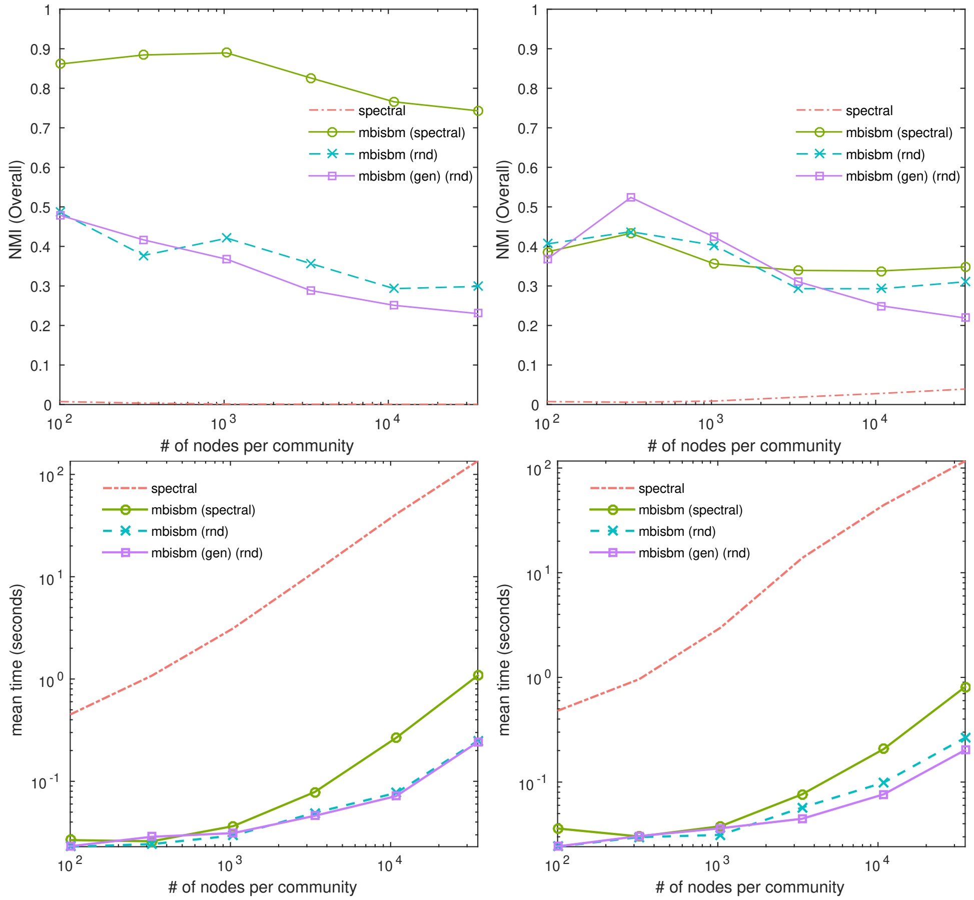


Figure 7: Scaling behavior. Both the average NMI (top) and the mean execution time (time) are shown as a function of the number of clusters per community for three-community ($K = 3$) networks. The two columns show two block models generated according to model (23): (Left) $B_0^{(1)}$ and (Right) $B_0^{(2)}$ as defined in (24).

4.6. Scaling behavior

Figure 7 illustrates how the algorithm scales to large networks compared to the spectral approach. The simulation setup for this figure is as follows: We consider a base connectivity matrix $B_0 \in \mathbb{R}_+^{K \times K}$ and let the edge probability matrix be

$$B = \frac{\log(N_1 N_2)}{\sqrt{N_1 N_2}} B_0, \quad N_1 = n_0 K, \; N_2 = [0.75 n_0 K] \tag{23}$$

where n_0 is the number of nodes per community; that is, we assume that communities of both sides are balanced and each contains n_0 nodes. We then let n_0 vary from 100 to 35000, while $K = 3$ is fixed, to study the large-scale behavior of the network. This simulation setup has recently been considered in Zhou and Amini (2018) and is suitable for studying the semi-sparse asymptotic regime. Note that the average degree of the network will be $\sim \log(n_0)$ as $n_0 \to \infty$ resulting in fairly sparse large networks. We consider two versions of B_0, corresponding to the

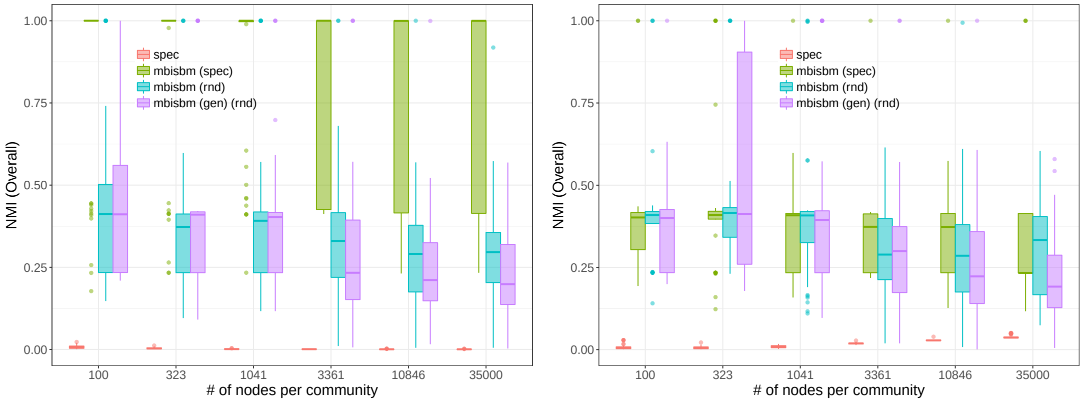


Figure 8: Boxplot of the NMI corresponding to Figure 7.

two columns in Figure 7:

$$B_0^{(1)} = \frac{1}{2}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \quad B_0^{(2)} = \frac{1}{3}\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{bmatrix}. \tag{24}$$

Note that $B_0^{(1)}$ gives a planted partition model whereas $B_0^{(2)}$ corresponds to the more general class of SBM given by (4), i.e., it contains both assortative and dissortative relations. We also note that for $B_0^{(2)}$, the within-community connectivities are variable (i.e., $B_0^{(2)}$ has unequal diagonal elements). Both models are quite hard as the so-called out-in-ratio is quite high; in the planted partition case $B_0^{(1)}$, for which the ratio directly characterizes hardness, we have $q/p = 0.5$. The covariates are generated similar to Section 4.4 with $\nu = 10$ and $d = (5, 5)$. The results are averaged over 50 replications.

The hardness of the two problems are reflected in the NMI plots of Figure 7 where the spectral method (`spectral`) is barely above the random assignment. In addition to the spectral method, three versions of `mbiSBM` have been shown: (1) Algorithm 1 with spectral initialization, `mbiSBM (spectral)`, (2) Algorithm 1 with random initialization, `mbiSBM (rnd)`, (3) modification of Algorithm 1 to allow for a general connectivity matrix (Section A.2), with random initialization, `mbiSBM (gen)(rnd)`.

All the three versions perform reasonably and comparably except for the `mbiSBM` with `spectral` initialization which significantly outperforms other versions on the planted partition model $B_0^{(1)}$. It is also worth nothing that the performance of `mbiSBM` with general connectivity matrix is not much better than the planted partition version (i.e., the default Algorithm 1), even on $B_0^{(2)}$ which is far from satisfying the simplifying planted partition assumption. We note that the spectral approach alone performs very poorly in both cases. (We also experimented with moderate perturbations for the spectral approach which did not result in noticeable improvement.) Figure 8 shows the box plots corresponding to the NMI plots in Figure 7, illustrating the variability of the results. It is worth noting that for planted partition model, `mbiSBM (spectral)` achieves a NMI of 1 for most replications.

The bottom row in Figure 7 illustrates the average run time of the algorithms. For `mbiSBM (spectral)` only the overhead relative the spectral initialization is considered. It is clear the family of `mbiSBM` algorithms are about an order of magnitude faster than the `spectral` approach. We conclude that when the signal is high, `mbiSBM` with random initialization is outperforms in both speed and accuracy. When the signal is low, we can initialize with the spectral approach and then boost the performance by computationally cheap `mbiSBM` iterations.

5. Application to Real Data

5.1. Wikipedia networks

We have applied the algorithm to two wikipedia page–user networks, which we will call TopArticles and Cities. Each is a bipartite network between a collection of Wikipedia pages and the users who edited them: An edge is placed between a user and a page if the user has edited that page (at least once). In the TopArticles, the pages are selected from the top articles (based on monthly contributions) from Chinese (CN), Korean (KR) and Japanese (JP) language Wikipedia, corresponding to the period from January to October 2016. In the Cities network, the pages correspond to city names in English language Wikipedia; the cities were chosen from five countries: Unites States (US), United Kingdom (GB), Australia (AU), India (IN), Japan (JP). In both cases, on the user side, only those with IP addresses were retained. Although, not perfect, IP addresses were the only means by which we could obtain additional information about each user, esp. geo-location data. Wikipedia usage statistics were scraped from Wikimedia Statistics using code inspired by Keegan (2014). For geo-data we used both the `ggmap` R package by Kahle and Wickham (2013) and the API provided by ipapi.

In TopArticles, the true labels are the language assigned to each page and each user, that is, matched communities are specified by common language. The user language was assigned based on the dominant language of the country from which the IP address originates. In Cities network, the true labels are the country names assigned to each user based on user's IP address and assigned to each city page based on its geo-location tag. The IPs were also used to obtain latitude and longitude coordinates on each user, providing us with user covariate matrix $X_2 \in \mathbb{R}^{N_2 \times 2}$,

For TopArticles, we do not have any page covariate. For Cities, we use the geo-location data of the city (latitude and longitude) to give us the page covariate matrix $X_1 \in \mathbb{R}^{N_1 \times 2}$.

Figure 9 shows the two networks along with the true communities. Note that Cities is specially hard to cluster based only on network data due to the presence of nodes of different communities among each community (as positioned by the layout algorithm). Tables 1 and 2 show the break-down of pages/users based on community for the two networks. Also shown are the average degrees of each side of the network, as well as the overall average degree. For each of the two networks, we first obtained a 2-core, restricted to the giant component, then removed users from countries not under consideration. (If the last step created disjoint components we restricted again to the giant component. This only happened for Cities and only removed 5 nodes.)

Results on Wikipedia networks. Table 3 illustrates the result of the application of `biSC`, and the `mbiSBM` (`biSC`) algorithm with various combination of covariates. In all cases the degree-corrected (DC) version of `mbiSBM` is used. For TopArticles, without using covariates, there is no improvement on `biSC` while using X_2 gives significant boost to `mbiSBM`. For Cities, `biSC` outperforms `mbiSBM` with no covariates. One the other hand, adding X_2 or both X_1 and X_2 significantly improves the result of `mbiSBM`.

	CN	JP	KR	Total	Avg. deg.	Covariates
Pages	139	143	171	453	14.2	N/A
Users	579	695	828	2102	3.1	X_2 = user (lat.,lon.)
Total	718	838	999	2555	5	

Table 1: TopArticles page–user network

	US	IN	AU	JP	GB	Total	Avg. deg.	Covariates
Pages	267	235	182	113	59	856	10.2	X_1 = city (lat.,lon.)
Users	1054	1029	705	101	201	3090	2.8	X_2 = user (lat.,lon.)
Total	1321	1264	887	214	260	3946	4.4	

Table 2: Cities page–user network

To get a more refined understanding of the relative standing of `biSC` and `mbiSBM` (`biSC`), we have run two Monte Carlo analyses based on these real networks, one using *subsampling* and the other by adding Erdös–Rényi noise. Figure 10 shows the results when we subsample the network to retain a fraction of the nodes on each side (from 95% down to 10%). The x-axis shows the resulting overall average degree of the network at each subsampling level. The results are averaged over 50 replications and the interquartile range (IQR) is also shown as a measure of variability. The figures in Table 3 correspond to the rightmost point of these plots. (Average degrees of Cities vary in these ranges: overall $\in [0.4, 4.2]$, page $\in [1, 9.7]$ and user $\in [0.3, 2.7]$, whereas for TopArticles the ranges are: overall $\in [0.6, 4.9]$, page $\in [1.6, 13.8]$ and user $\in [0.3, 3]$.)

The plots show that `mbiSBM` (`biSC`) with covariates outperforms `biSC`, and the improvement is quite significant the sparser the network becomes. Note for example that in Cities adding X_1 does not have much effect in the original network, however, there is a considerable improvement when average degree starts to drop under subsampling. In the Cities case, the two covariates X_1 and X_2 together are quite strong leading to an NMI ≈ 1 and masking the effect of the network to some extent.

However, by looking at cases where only one of X_1 and X_2 is present, we observe that `mbiSBM` (`biSC`) manges to pass the covariate information via the network to the side without covariates, thus improving matched NMI significantly. To see this, consider for example the TopArticles, where only X_2 is present. In this case, even if a method could cluster X_2 perfectly and, in the absence of network information randomly guessed the labels of the other side, the NMI would be 0.42. That in Figure 10(b), the NMI starts at 0.98 and remains much

Network	`biSC`	`mbiSBM` (`biSC`), DC			
		X_1 & X_2	X_1	X_2	no X
TopArticles	0.86	-	-	0.98	0.86
Cities	0.6	1.0	0.59	0.85	0.47

Table 3: Matched NMI for `biSC` and `mbiSBM` on the two Wikipedia networks.

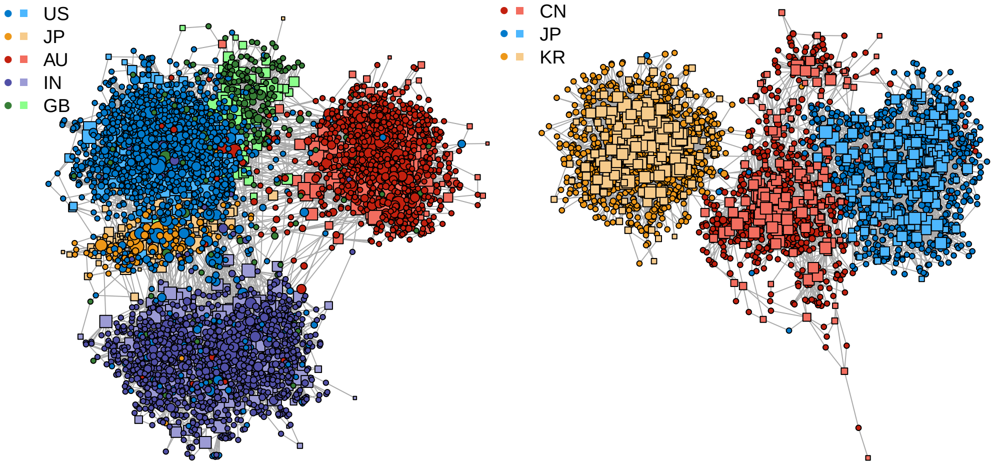


Figure 9: The two Wikipedia networks: (left) Cities (right) TopArticles. Nodes are colored according to true communities. Pages are denoted with squares and users with circles. Node sizes are proportional to log-degrees.

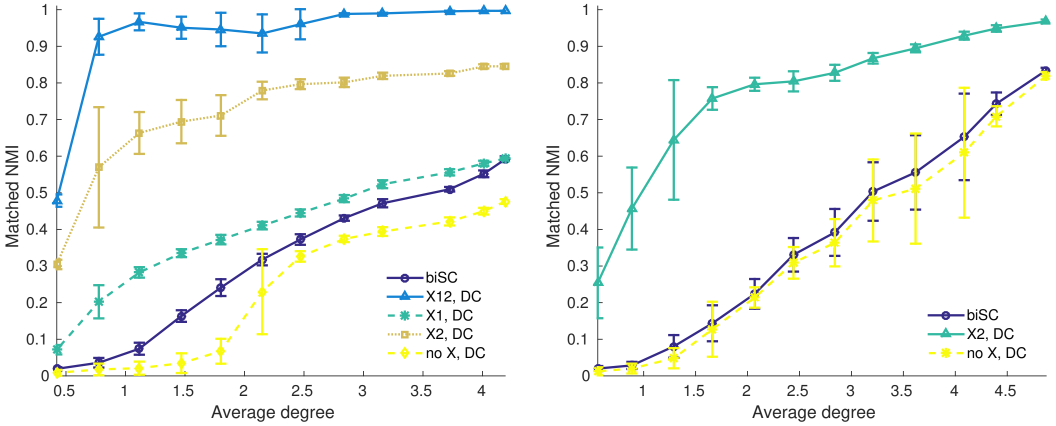


Figure 10: Effect of subsampling on Wikipedia networks: (left) Cities (right) TopArticles.

above 0.42 for most of the range of subsampling illustrates the ability of mbiSBM to effectively utilize both covariate and network information to correctly infer the labels of the other side. The same can be observed in the case of Cities. Finally, we note that without covariates, biSC usually performs better. We expect this since it is hard for local methods starting from biSC to improve upon it. The strength comes when we use the covariate information.

Figure 11 shows another experiment where we added Erdös–Rényi noise of average degree from 0 to 10. Again, the advantage gained by mbiSBM from using covariates can be quite clearly observed when one increases the noise. The covariates mitigate the effect of noise and lead to a much graceful degradation of performance for mbiSBM relative to biSC.

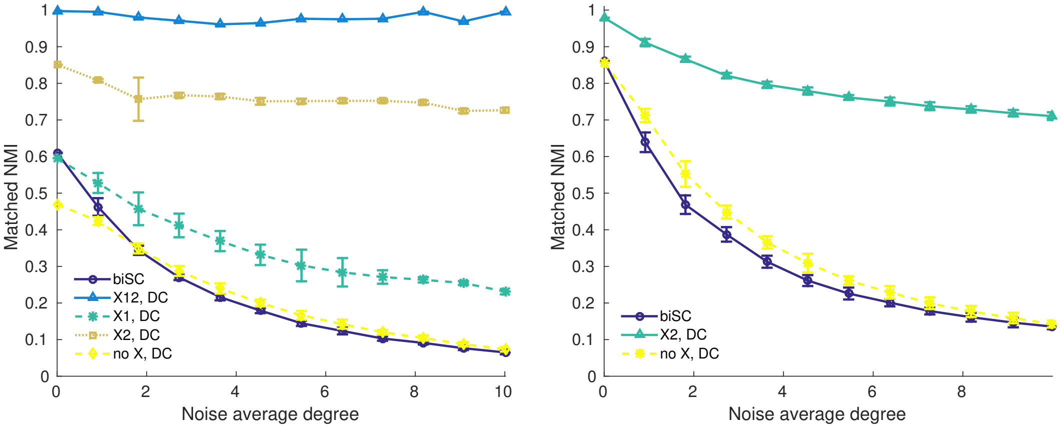


Figure 11: Effect of adding Erdös–Rényi noise on Wikipedia networks: (left) Cities (right) TopArticles.

5.2. Author-Paper networks

In this subsection, we use real data examples to showcase the performance of our algorithm when the the network is large, has high dimensional covariates and under model misspecification, that is, when the covariates are discrete.

We have applied the algorithm to two sets of author-paper networks. The first set represents papers and their authors in different sub-domains of computer science (CS): (1) data mining, (2) medical informatics, (3) visualization , (4) database, and (5) theory. The data for the first set was extracted from the Arnetminer collection, based on papers published from 1990 to 2005 in certain CS venues by Tang et al. (2012). The second set of data was scraped from DBLP, using the papers published in bioinformatics (BI), biomedical/medical informatics (BM) and computational neuroscience (CN) venues from 1995 to 2018. Each set contains various subnetworks of the two basic bipartite network described above.

In both sets, we first consider the paper-author bipartite networks with covariates being high-dimensional vectors of word frequencies appearing in the title of the papers. To deal with the binary nature of the covariates, we take their z-scores. Although the Gaussian assumption about these transformed covariates does not hold, we have found that such transformation allows us to apply our model to discrete covariates with minimal modification and with good empirical performance as discussed below. A more principled approach to dealing with discrete covariates would be to model them using a multinomial likelihood (cf. Section 6). The process of building the network and extracting the covariates is done by restricting to the giant component of the author-paper network such that each paper is associated with at least one word after removing sparse words. For the second set, we also removed some non-informative words to slightly boost the covariate signals.

These datasets have an interesting feature: They can be viewed as tri-partite author-paper-word networks. Thus, although not intuitive, we can also view the paper-word frequency matrix as a bipartite network and use the (binary) author vectors as covariates for the papers. Taking z-scores again bring these binary covariates within our framework.

For these datasets, the ground truth labels are only available for the papers: we treat their venues (i.e., their main topic) as the true community of the paper. As a result, we can only find the NMI for the paper labels. Note that assuming a matching between authors and

	Dimensions			Avg. deg.	
	authors	papers	words	authors	papers
CS12345	5126	8500	1089	4.75	2.87
BM-BI-CN	10561	8482	1649	3.4	4.23

Table 4: Statistics on the author–paper networks with words as covariates

Network	X_2	`biSC`			`mbiSBM` (`biSC`)			`mbiSBM` (`rnd`)	
		$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	mean	max
CS12345	0.23	0.29	0.28	0.10	0.23	**0.36**	0.34	0.16	0.23
CS2345	0.30	0.39	0.34	0.15	0.28	**0.45**	0.38	0.13	0.26
CS345	0.38	0.35	0.45	0.09	0.21	**0.62**	0.35	0.14	0.29
CS45	0.35	0.00	**0.76**	0.04	0.00	0.73	0.71	0.02	0.08
BM-BI-CN	0.32	0.00	0.01	0.02	0.00	0.01	0.04	0.23	**0.35**
BI-CN	0.36	0.32	0.00	0.00	**0.55**	0.00	0.00	0.15	0.37
BM-BI	0.00	0.00	0.00	0.04	0.08	0.00	**0.27**	0.17	**0.27**

Table 5: NMI for `biSC` and `mbiSBM` on the Author-Paper networks.

papers is reasonable as most authors have a theme; similarly a matching between words and papers is plausible, as previously has been considered in the text mining applications by Dhillon (2001). Table 4 shows the break-down of authors/papers/words for CS12345 and BM-BI-CN networks. Also shown are the average degrees of each side of the networks.

CS12345 refers to the the dataset including all the five topics. CS2345 refers to the dataset excluding the first topic (data mining), and so on. Note that CS2345, for instance, is not necessarily an induced subnetwork of CS12345 as we take the giant component each time to make the networks for each set of topics.

Results on paper-author-word networks. Tables 5 and 6 illustrate the results for various algorithms. The first column from the left is the NMI obtained by applying k-means on the truncated SVD of the covariates (X_2 column); this approach captures the covariate-only information. The next three columns show the result for the application of `biSC` with various levels of perturbation α. Here, we are adding the constant perturbation $\alpha(N_1 N_2)^{-1/2}$ to every entry of the adjacency matrix before forming the Laplacian. This is similar to the approach of Amini et al. (2013) for unipartite setups and is known to have a regularization effect on the spectral clustering, especially for sparse networks. Note that the case $\alpha = 0$ corresponds to the unregularized version as shown in Algorithm 2.

The remaining columns show the performance of `mbiSBM` initialized with each of `biSC` results, collected under `mbiSBM` (`biSC`) columns, as well as `mbiSBM` initialized at random, where both the mean and the maximum achieved on 25 random initializations is recorded, under `mbiSBM` (`rnd`) columns.

The results show that `mbiSBM` outperforms covariate-only clustering in all cases and outperforms `biSC` in 12 out of 14 datasets, indicating that utilizing covariates as well as network structure can boost the signals compared to using each source of the information alone. It is

Network	X_2	`biSC`			`mbiSBM` (`biSC`)			`mbiSBM` (`rnd`)	
		$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	$\alpha = 0$	$\alpha = 1$	$\alpha = 10$	mean	max
CS12345	0.00	0.26	0.26	0.24	0.43	**0.45**	0.45	0.03	0.11
CS2345	0.00	0.27	0.28	0.28	0.52	0.52	**0.54**	0.02	0.20
CS345	0.00	0.24	0.39	0.37	0.74	0.74	**0.76**	0.02	0.29
CS45	0.00	0.37	0.38	0.36	0.70	0.70	**0.71**	0.01	0.02
BM-BI-CN	0.00	0.00	0.31	0.35	0.00	**0.45**	0.43	0.01	0.03
BI-CN	0.00	0.01	0.02	0.36	0.00	0.00	**0.68**	0.00	0.01
BM-BI	0.00	0.00	0.01	**0.26**	0.00	0.00	0.17	0.00	0.00

Table 6: NMI for `biSC` and `mbiSBM` on the Word-Paper networks.

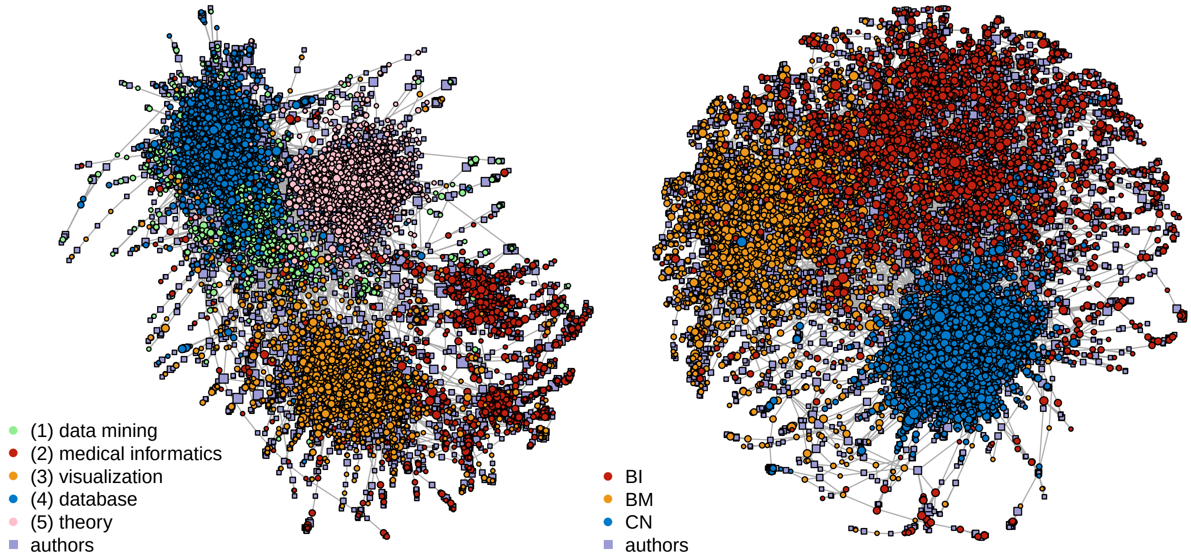


Figure 12: The two paper-author networks: (left) CS12345 (right) BM-BI-CN. Nodes corresponding to the papers are colored according to their true communities. The authors are denoted with squares and the papers with circles. Node sizes are proportional to log-degrees.

also worth noting that perturbation often boosts the performance of `biSC`, but not always. In cases where a version of `biSC` works well, `mbiSBM` often boosts the results; this is especially pronounced for the word-paper networks. Overall random initialization of `mbiSBM` performs respectably and surprisingly in some outperforms `biSC` and `biSC`-initialized `mbiSBM`.

6. Discussion

In this paper, we considered the problem of matched community detection in the bipartite setting, where one assumes a latent one-to-one correspondence between communities of the two sides. This matching is built into the model and inferred simultaneously with the communities in the process of fitting the model. Our model is an extension of the stochastic block model (SBM) and its degree-corrected version (DC-SBM). We extended our proposed model to allow for the presence of node covariates that are aware of the matching between communities: Covariates corresponding to nodes in matched communities are statistically linked using hierarchical Bayesian modeling ideas.

Although we only considered Gaussian distributions in generating covariates, our hierarchical mixture approach has the potential for extension to more general settings. For example, one can easily model discrete covariates, such as word counts in documents, as mixtures of multinomial distributions. Some care however is needed when deciding the distribution of the top layer if one wants to allow for information sharing among the lower level variables (i.e., v_{rk}) from the two sides. In addition, as mentioned (cf. Section 2.5), our current statistical linkage carries weak information about the matching and it would be interesting to design models in which the degree of covariate information about the matching can be tuned more effectively.

Our model has natural extensions to r-partite ($r > 2$) networks where some of the modes may or may not have node covariates. We note that the general r-partite case is related to the so-called multilayer or multiplex community detection problem (Kivela et al., 2014). Finally, one would like to allow for edge covariates to accommodate many cases in real data, where edges are annotated in some way, say by the ratings as in recommender systems, by timestamps as in our Wikipedia user-page examples, and so on. Poisson model of edge generation can, to some extent, take simple edge weights into account. Whether one can go beyond that in modeling more complex edge information is an interesting avenue for future work.

Acknowledgment

We would like to thank the referees and Associate Editor for their helpful comments that improved this paper and Nan Deng for helpful discussions. This work used computational and storage services associated with the Hoffman2 Shared Cluster provided by UCLA Institute for Digital Research and Education's Research Technology Group.

Appendix A. Extensions and improvements

A.1. Speed improvement and diagonal restriction

A.1.1. Improving the Speed

For $r = 1, 2$, we treat each $(\tau_{rik})_{ik}$ as a matrix $\tau_r \in [0, 1]^{N_r \times K}$. The τ-update in (17) can be simplified to improve computational complexity for sparse networks A. We can write $h(p, q; \alpha) = \alpha\phi_1 + \phi_0$, where for the binary likelihood, $\phi_1 = \log \frac{p(1-q)}{q(1-p)}$ and $\phi_0 = \log \frac{1-p}{1-q}$, and for the Poisson likelihood considered in Section A.3 below, $\phi_0 = q - p$ and $\phi_1 = \log(p/q)$. Then, in matrix notation

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big(\phi_1 [A\tau_2]_{ik} + \phi_0 \bar{\tau}_{2k} + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big) \quad i = 1, \dots, N_1$$

where $[A\tau_2]_{ik} = \sum_j \tau_{2jk} A_{ij}$ and we recall $\bar{\tau}_{2k} := \sum_{i=1}^{N_2} \tau_{2ik}$. When A is sparse, the matrix-vector product $A\tau_2$ can be computed quite fast. Letting $\beta_r = (\beta_{rik})_{ik} \in \mathbb{R}^{N_r \times K}$, we have the τ-update in vector form:

$$\tau_1 = \texttt{row-softmax}\big[\phi_1 A\tau_2 + \phi_0 \mathbf{1}_{N_1} (\bar{\tau}_2 + \log \pi_1)^T + \beta_1\big] \tag{25}$$

and similarly for τ_2. Here, `row-softmax` is the row-wise softmax operator, applying (16) to each row of a matrix.

Further improvements are possible in estimating p and q. Note that we can write $\bar{\tau}_{rk} := [\tau_r^T \mathbf{1}_{N_r}]_k$. Let us treat $\bar{\tau}_r$ as a K-vector, with elements $\bar{\tau}_{rk}$. Then, we have $\sum_{ij} \gamma_{ij}(\tau) A_{ij} = \mathrm{tr}(\tau_1^T A \tau_2)$ and $\sum_{ij} \gamma_{ij}(\tau) = \langle \bar{\tau}_1, \bar{\tau}_2 \rangle$, and

$$p = \frac{\mathrm{tr}(\tau_1^T A \tau_2)}{\langle \bar{\tau}_1, \bar{\tau}_2 \rangle}, \quad q = \frac{\delta\rho - p}{\delta - 1}, \quad \text{where } \frac{1}{\delta} = \langle \frac{\bar{\tau}_1}{N_1}, \frac{\bar{\tau}_2}{N_2} \rangle \text{ and } \rho = \frac{1}{N_1 N_2} \sum_{ij} A_{ij}. \tag{26}$$

Note that ρ is the density of the graph (or A) and that $\langle \bar{\tau}_1, \bar{\tau}_2 \rangle = \mathrm{tr}(\tau_1^T E \tau_2)$ where E is the all-ones matrix of appropriate dimension. Finally, let us define $\beta'_{rik} = \mathrm{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2$ noting that $\beta_{rik} = \frac{1}{2\sigma_r^2}\beta'_{rik}$ from definition (13). Letting $\beta'_r = (\beta'_{rik})_{ik}$ be its matrix form, we can write the update for σ_r^2 compactly as

$$\sigma_r^2 = \frac{1}{N_r d_r} \sum_{ik} \tau_{rik} \beta'_{rik} = \frac{1}{N_r d_r} \mathrm{tr}(\tau_r^T \beta'_r). \tag{27}$$

A.1.2. Diagonal covariance restriction

For ultra high-dimensional covariates, one can restrict the covariance matrix Σ to be diagonal, which greatly improves the speed of the algorithm. This choice is also reasonable from a statistical perspective since in high-dimensions, without additional restrictions (such as sparsity), estimates of a full-dimensional covariance matrix are unreliable. Under the diagonal restriction, one only needs to make a minor modification to the algorithm: Recalling the variational likelihood $J(\Sigma) \doteq -\frac{K}{2}\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1} S(\widetilde{\Gamma}))\big]$, the maximizer over Σ under the diagonal constraint is $\Sigma_{ii} = [S(\widetilde{\Gamma}))]_{ii}$ and $\Sigma_{ij} = 0$ for $i \neq j$, or compactly

$$\Sigma = \mathrm{ddiag}(S(\widetilde{\Gamma})),$$

where $A \mapsto \mathrm{ddiag}(A)$ is an operator that take a square matrix A and outputs a diagonal matrix with the same diagonal as A. We note that the update for $\widetilde{\Sigma}_k$ goes as before: $\widetilde{\Sigma}_k = (D_k^{-1} + \Sigma^{-1})^{-1}$ for $k \in [K]$. Since both D and Σ are diagonal, $\widetilde{\Sigma}_k$ will be diagonal as well. Thus, all the covariance matrices throughout the algorithm will remain diagonal which improves the speed and scalability.

A.2. Extension to general SBM

Let us now briefly discuss the changes needed for fitting the general SBM model (4). We only consider the case with no degree correction. Let us derive the label updates first. Using the general form of the network likelihood (8), the variational likelihood in (15) is replaced with

$$J = \sum_i \sum_k \tau_{1ik} \Big(\sum_j \sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) + \xi_{1ik} - \log \tau_{1ik} \Big) + \text{const.} \tag{28}$$

which by the same argument used for (17), gives the following τ_{1ik} update

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big[\sum_j \sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2) \Big] \quad i = 1, \dots, N_1. \tag{29}$$

The update for τ_2 follows similarly. The only other modification to the algorithm is the update for the edge probabilities Ψ, replacing the (p, q) updates. For Ψ updates, we need to maximize

$$\mathbb{E}_q[\ell_1(\Psi)] = \sum_{ij} \sum_{k\ell} \tau_{1ik}\tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij})$$

over Ψ. Assume that $g(p, \alpha) = \alpha\phi_1(p) + \phi_0(p)$. For any fixed (k, ℓ), we need to maximize

$$\Big(\sum_{ij} \tau_{1ik}\tau_{2j\ell}A_{ij}\Big)\phi_1(\Psi_{k\ell}) + \bar{\tau}_{1k}\bar{\tau}_{2\ell}\, \phi_0(\Psi_{k\ell})$$

where we have used $\bar{\tau}_{rk} := \sum_i \tau_{rik}$. The problem reduces to maximizing $p \mapsto a\phi_1(p) + b\phi_0(p)$ for some positive $a, b \in \mathbb{R}$. It is not hard to see that the maximizer is the same for both the Bernoulli likelihood ($\phi_1(p) = \log[p/(1-p)]$, $\phi_0 p) = \log(1-p)$) and the Poisson ($\phi_1(p) = \log p$, $\phi_0(p) = -p$), and is equal to $p^* = a/b$. This give the following Ψ-update

$$\Psi_{k\ell} = \frac{1}{\bar{\tau}_{1k}\bar{\tau}_{2\ell}} \sum_{ij} \tau_{1ik}\tau_{2j\ell}A_{ij}. \tag{30}$$

Speeding up the updates. Using the notation introduced for g, we have

$$\sum_j \sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) = \sum_\ell \Big(\sum_j \tau_{2j\ell}A_{ij}\Big)\phi_1(\Psi_{k\ell}) + \sum_\ell \bar{\tau}_{2\ell}\, \phi_0(\Psi_{k\ell}).$$

Let $\Phi_1, \Phi_0 \in \mathbb{R}^{K\times K}$ be matrices with entries $[\Phi_s]_{k\ell} = \phi_s(\Psi_{k\ell})$ for $s = 0, 1$. Then, τ_1 update is

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big([A\tau_2\Phi_1^T]_{ik} + [\Phi_0\bar{\tau}_2]_k + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big) \quad i = 1, \dots, N_1$$

or in matrix form (similar to (25))

$$\tau_1 = \texttt{row-softmax}\big(A\tau_2\Phi_1^T + \mathbf{1}_{N_1}[\Phi_0\bar{\tau}_2 + \log\pi_1]^T + \beta_1\big). \tag{31}$$

Here $\bar{\tau}_2 = (\bar{\tau}_{2\ell}) \in \mathbb{R}^K$ is viewed as a column vector.

A.3. Extension to the Degree-Corrected Case

In this case the network-dependent part of the likelihood is replaced with

$$\ell_1(\Psi, \theta) = \sum_{ij} \sum_{k\ell} z_{1ik}z_{2j\ell}\, g(\theta_{1i}\theta_{2j}\Psi_{k\ell}, A_{ij}).$$

Again, we focus on the case where $\Psi_{kk} = p$ and $\Psi_{k\ell} = q$ for $k \neq \ell$. Recalling the notation $h(p, q, \alpha) = g(p, \alpha) - g(q, \alpha)$, we have

$$\ell_1(\Psi, \theta) = \sum_{ij} \big[y_{ij}\, h\big(p\theta_{1i}\theta_{2j},\, q\theta_{1i}\theta_{2j},\, A_{ij}\big) + g\big(\theta_{1i}\theta_{2j}q, A_{ij}\big)\big]. \tag{32}$$

We assume a Poisson log-likelihood with $g(p, \alpha) = \alpha\log p - p$ for which $h(p, q, \alpha) = \alpha\log(p/q) + q - p$. We also recall the normalization assumption (7), $\sum_i \theta_{ri}z_{rik} = \sum_i z_{rik}$, which implies

$\sum_{ij} y_{ij}\theta_{1i}\theta_{2j} = \sum_{ij} y_{ij}$ and $\sum_{ij} \theta_{1i}\theta_{2j} = N_1 N_2$. Using these two implications, the first term in (32) simplifies to

$$\sum_{ij} y_{ij}\big[(q-p)\theta_{1i}\theta_{2j} + A_{ij}\log(p/q)\big] + \sum_{ij}\big[-\theta_{1i}\theta_{2j}q + A_{ij}\log(\theta_{1i}\theta_{2j}q)\big]$$
$$= \sum_{ij} y_{ij}\big[(q-p) + A_{ij}\log(p/q)\big] - qN_1N_2 + \sum_{ij} A_{ij}\log(\theta_{1i}\theta_{2j}q)$$

Let $\phi_0 = q - p$ and $\phi_1 = \log(p/q)$.

τ-update. Let us fix θ and obtain updates for the label posteriors τ. Taking expectations of the objective and the constraints, the τ-portion of the update is equivalent to maximizing

$$\sum_{ij} \gamma_{ij}(\tau)\big[\phi_0 + A_{ij}\phi_1\big] + \sum_{rik} \tau_{rik}\big[\beta_{rik}(\widetilde{\Gamma},\sigma^2) + \log\frac{\pi_{rk}}{\tau_{rik}}\big]$$

subject to constraints $\sum_i \tau_{rik}(\theta_{ri} - 1) = 0$ for all k. Note that these constraints follow by taking expectations of the normalization constraints (7) under $Z \sim q$. Focusing on updating τ_{1k}, we have the following optimization problem:

$$\begin{array}{ll} \max\limits_{\tau_1} & \sum_{i,k}\tau_{1ik}\big(\sum_j \tau_{2jk}\big[\phi_0 + A_{ij}\phi_1\big] + \xi_{1ik} - \log\tau_{1ik}\big) \\ \text{subject to} & \sum_i \tau_{1ik}(\theta_{1i}-1) = 0, \quad \sum_k \tau_{1ik} = 1, \quad \tau_{1ik} \geq 0 \end{array} \tag{33}$$

where $\xi_{1ik} = \beta_{1ik} + \log\pi_{1k}$ as before. In Appendix C.1, we derive a dual ascent algorithm for solving this problem with the following updates:

$$\begin{aligned} \tau_1(\lambda) &= \texttt{row-softmax}\big[\phi_1 A\tau_2 + \phi_0 \mathbf{1}_{N_1}(\bar{\tau}_2 + \log\pi_1)^T + \beta_1 + (\theta_1 - \mathbf{1})\lambda^T\big], \\ \lambda^+ &= \lambda - \mu[\tau_1(\lambda)]^T(\theta_1 - \mathbf{1}). \end{aligned} \tag{34}$$

Here, $\lambda \in \mathbb{R}^K$ is the dual variable, λ^+ is its update, $\theta_1 = (\theta_{1i}) \in \mathbb{R}^n$, and μ is a proper step-size. These two iterations are repeated till convergence, before updating other parameters. Note that when $\theta_1 = \mathbf{1}$, the dual ascent algorithm reduces to the single step of (25) obtained for the case without degree correction.

θ-update. Let us now fix τ and the rest of the parameters and optimize over θ. The relevant portion of the objective function is

$$\sum_{ij}\gamma_{ij}(\tau)\big[q - p + A_{ij}\log(p/q)\big] - qN_1N_2 + \sum_{ij} A_{ij}\log(\theta_{1i}\theta_{2j}q).$$

Consider optimizing over (θ_{1i}), which is equivalent to maximizing $\sum_i d_{1i}\log\theta_{1i}$, subject to $\sum_i \tau_{1ik}(\theta_{1i} - 1) = 0$ for all k, and $\theta_{1i} \geq 0$ for all i. This problem is suitable for an application of the Douglas–Rachford (DR) splitting algorithm (Douglas and Rachford, 1956; O'Connor and Vandenberghe, 2014). Let $f_t(\cdot; d) : \mathbb{R}^n_+ \to \mathbb{R}^n_+$ with $d \in \mathbb{R}^n_+$ and $t > 0$, be defined by

$$[f_t(x;d)]_i := \frac{1}{2}\big[x_i + \sqrt{x_i^2 + 4td_i}\big]. \tag{35}$$

Also, let $H_1 := \tau_1(\tau_1^T\tau_1)^{-1}\tau_1^T$ be the projection operator onto the span of $\tau_1 \in \mathbb{R}^{N_1\times K}$. The algorithm performs the following iterations for updating (ξ_1, θ_1) to (ξ_1^+, θ_1^+):

$$\begin{aligned} \theta_1^+ &= f_t(\xi_1; d_1) \\ \xi_1^+ &= \theta_1^+ - H_1(2\theta_1^+ - \xi_1 - \mathbf{1}) \end{aligned} \tag{36}$$

where $\xi_1 \in \mathbb{R}^{N_1}$ is an auxiliary variable, $d_1 = (d_{1i}) \in \mathbb{R}^{N_1}$ collects the degrees of side 1, and $t > 0$ is the fixed parameter of DR algorithm (often set to 1). The details for the derivation of this algorithm can be found in Appendix C.2. The same updates apply to θ_2, replacing subscript 1 with 2.

Remark 3. Note that if $\tau_1 = (\tau_{1ik})$ was a hard label assignment, then the optimization for (θ_{1i}) would have a simple solution. To see this, let $C_k(\tau_1)$ be the kth cluster of hard label τ_1. Then, the optimal value of θ_1 is given by

$$\theta_{1i} = \frac{d_{1i}}{\sum_{i' \in C_k(\tau_1)} d_{1i'}}, \quad \text{for } i \in C_{1k}(\tau_1).$$

This is in fact, the choice in profile-likelihood approaches to fitting DC-SBM, where one replaced θ_1 with this optimal value, in addition to optimal values of edge probabilities and class priors, all in terms of $\{C_{1k}(\tau_1)\}$, and then optimize the resulting profile likelihood over $\{C_{1k}(\tau_1)\}$. See for example (Karrer and Newman, 2011). Our approach here, allows us to keep a soft-label assignment τ_1 throughout the algorithm, viewing optimization over θ_1 as another phase of block-coordinate ascent for the overall constrained optimization problem.

(p, q)-update. To optimize over p and q we note that because of the Poisson model, p and q are not tied together and the only constraint we have is $p, q \geq 0$. Optimizing over p is equivalent to maximizing $-p\sum_{ij}\gamma_{ij} + \log p \sum_{ij}\gamma_{ij}A_{ij}$ and optimizing over q, is equivalent to maximizing over $-q\sum_{ij}(1-\gamma_{ij}) + \log q \sum_{ij}(1-\gamma_{ij})A_{ij}$, both giving the same updates as those in (21).

Appendix B. Details of Section 3

B.1. Proof of Lemma 1

We have $f_a(p) = -\sum_k p_k \log(p_k/e^{a_k})$. If $\sum_k e^{a_k} = 1$, then $-f_a(p)$ is the KL-divergence between (p_k) and (e^{a_k}) and the result follows. Otherwise, normalizing only adds a constant to f_a, that is, with $C = 1/\sum_k e^{a_k}$ and $q_k = Ce^{a_k}$, we have $f_a(p) = -\sum_k p_k \log(p_k/q_k) - \log C$ and the result follows.

B.2. Derivation of (12)

We write $\mathbb{E}_q$ for expectation w.r.t. the above joint distribution on (Z, V). Similarly, we write $\mathbb{E}_{q_Z}$ and $\mathbb{E}_{q_V}$ for the expectation (or integration) w.r.t. to each of q_Z and q_V. Note that $\mathbb{E}_q[\,\cdot\,] = \mathbb{E}_{q_V}\mathbb{E}_{q_Z}[\,\cdot\,]$. Plugging in the variational distribution (11) into the variational likelihood (10). we have

$$J = \mathbb{E}_q\Big[\ell(\mu, \Sigma, \sigma, Q)] - \log q(Z, V)\Big] = T_1 + T_2 + T_3 - T_4 - T_5$$

where

$$T_1 = \mathbb{E}_{q_Z}\Big[\sum_{i,j}\psi(A_{ij}, y_{ij})\Big], \quad T_2 = \mathbb{E}_{q_V}\mathbb{E}_{q_Z}\Big[\sum_{r,i,k} z_{rik}\big[\log f_r(x_{ri}; v_{rk}) + \log \pi_{rk}\big]\Big]$$

$$T_3 = \mathbb{E}_{q_V}\Big[\sum_k \log p(v_{*k}|\mu, \Sigma)\Big], \quad T_4 = \mathbb{E}_{q_Z}\log q(Z), \quad T_5 = \mathbb{E}_{q_V}\log q(V)$$

Let $\gamma_{ij}(\tau) := \mathbb{E}_{q_Z}(y_{ij}) = \sum_{k=1}^K \tau_{1ik}\tau_{2jk}$, so that

$$T_1 = \sum_{i,j}\Big[\gamma_{ij}(\tau)\log(p^{A_{ij}}(1-p)^{1-A_{ij}}) + (1-\gamma_{ij}(\tau))\log(q^{A_{ij}}(1-q)^{1-A_{ij}})\Big] \tag{37}$$

We frequently use the following elementary result in the sequel. Let $x \mapsto N(x; \mu, \Sigma)$ be the PDF of the multivariate normal distribution with mean μ and covariance Σ.

Lemma 2. *Let ε be a random vector with mean $\widetilde{\mu}$ and covariance $\widetilde{\Sigma}$, and x a nonrandom vector. Then,*

$$\mathbb{E}[\varepsilon^T\Lambda\varepsilon] = \operatorname{tr}[\Lambda\widetilde{\Sigma}] + \widetilde{\mu}^T\Lambda\widetilde{\mu}, \tag{38}$$

$$\begin{aligned}\mathbb{E}[\log N(x;\varepsilon,\Sigma)] = \mathbb{E}[\log N(\varepsilon; x, \Sigma)] &= -\frac{1}{2}\Big\{\log|\Sigma| + (x-\widetilde{\mu})^T\Sigma^{-1}(x-\widetilde{\mu}) + \operatorname{tr}(\Sigma^{-1}\widetilde{\Sigma})\Big\}\\ &= -\frac{1}{2}\Big\{\log|\Sigma| + \operatorname{tr}(\Sigma^{-1}\Psi)\Big\}, \end{aligned}\tag{39}$$

where $\Psi = \widetilde{\Sigma} + (x-\widetilde{\mu})(x-\widetilde{\mu})^T$.

Proof Let us prove (39). We have $\log N(x;\varepsilon,\Sigma) = -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(x-\varepsilon)^T\Sigma^{-1}(x-\varepsilon)$. Noting that $x - \varepsilon$ has mean $x - \widetilde{\mu}$ and covariance $\widetilde{\Sigma}$ and applying (38) gives the desired result. ■

Recall that $f_r(x_{ri}; v_{rk}) = N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r})$, and note that under q_V, v_{rk} has mean $\widetilde{\mu}_{rk}$ and covariance $(\widetilde{\Sigma}_k)_{rr}$. Note that we are partitioning $\widetilde{\Sigma}_k$ into four blocks of sizes $\{d_1, d_2\} \times \{d_1, d_2\}$ and $(\widetilde{\Sigma}_k)_{rr}, r = 1, 2$ correspond to the two diagonal blocks in this partition. Using Lemma 2, we have

$$\begin{aligned} T_2 &= \mathbb{E}_{q_V}\Big\{\sum_{r,i,k}\tau_{rik}\big[\log N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r}) + \log\pi_{rk}\big]\Big\}\\ &= \sum_{r,i,k}\tau_{rik}\big[-\frac{d_r}{2}\log\sigma_r^2 - \frac{\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2}{2\sigma_r^2} + \log\pi_{rk}\big]. \end{aligned}$$

Recall that $\widetilde{\Gamma} := ((\widetilde{\Sigma}_k, \widetilde{\mu}_k), k = 1, \dots, K)$ and $S(\widetilde{\Gamma}) := \frac{1}{K}\sum_{k=1}^K\big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]$. Another application of Lemma 2 gives

$$\begin{aligned} T_3 = \mathbb{E}_{q_V}\Big[\sum_k \log N(v_{*k}; \mu, \Sigma)\Big] &= -\frac{1}{2}\sum_k\big[\log|\Sigma| + \operatorname{tr}(\Sigma^{-1}\tilde{\Sigma}_k) + (\tilde{\mu}_k - \mu)^T\Sigma^{-1}(\mu_k - \mu)\big]\\ &= -\frac{1}{2}\sum_k\big[\log|\Sigma| + \operatorname{tr}(\Sigma^{-1}\Psi_k(\widetilde{\Gamma}))\big]\\ &= -\frac{K}{2}\big[\log|\Sigma| + \operatorname{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big]. \end{aligned}$$

Using Lemma 2 once more, we have

$$\begin{aligned} T_5 = \mathbb{E}_{q_V} \log q(V) &= \sum_k \mathbb{E}_{q_V} \log N(v_{*k}; \widetilde{\mu}_k, \widetilde{\Sigma}_k) \\ &= \sum_k -\frac{1}{2}\big[\log|\widetilde{\Sigma}_k| + \operatorname{tr}(I_{d_1+d_2})\big] = -\frac{1}{2}K(d_1+d_2) - \frac{1}{2}\sum_k \log|\widetilde{\Sigma}_k| \end{aligned}$$

Finally, we have

$$T_4 = \mathbb{E}_{q_Z} \log q(Z) = \mathbb{E}_{q_Z} \sum_{r,i,k} z_{rik} \log \tau_{rik} = \sum_{r,i,k} \tau_{rik} \log \tau_{rik}. \tag{40}$$

Putting the pieces together we get expression (12).

B.3. Updates of $\widetilde{\Sigma}$ and $\widetilde{\mu}$

From (12), the relevant portion of J which is a function of $\widetilde{\Gamma} = (\widetilde{\mu}, \widetilde{\Sigma})$ is given by

$$J(\widetilde{\mu}, \widetilde{\Sigma}) \doteq \sum_{r,i,k} \tau_{rik}\beta_{rik}(\widetilde{\Gamma}, \sigma^2) - \frac{K}{2}\operatorname{tr}[\Sigma^{-1}S(\widetilde{\Gamma}, \mu)] + \frac{1}{2}\sum_k \log|\widetilde{\Sigma}_k|$$

Substituting $\beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2}[\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2]$, and $S(\widetilde{\Gamma}, \mu) := \frac{1}{K}\sum_{k=1}^K \big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]$ from their definitions, and looking at the result only as a function of $\widetilde{\Sigma}$, we obtain

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2}\sum_k \Big[\sum_{r,i} \tau_{rik}\frac{\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big)}{\sigma_r^2} + \operatorname{tr}[\Sigma^{-1}\widetilde{\Sigma}_k] - \log|\widetilde{\Sigma}_k|\Big] \tag{41}$$

Recalling $\bar{\tau}_{rk} := \sum_{i=1}^{N_r} \tau_{rik}$ and $D_k^{-1} := \operatorname{diag}\big(\frac{\bar{\tau}_{1k}}{\sigma_1^2}I_{d_1}, \frac{\bar{\tau}_{2k}}{\sigma_2^2}I_{d_2}\big)$, we have

$$\sum_{r,i} \tau_{rik}\frac{\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big)}{\sigma_r^2} = \sum_r \bar{\tau}_{rk}\frac{\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big)}{\sigma_r^2} = \operatorname{tr}\Big[\sum_r \frac{\bar{\tau}_{rk}}{\sigma_r^2}(\widetilde{\Sigma}_k)_{rr}\Big] = \operatorname{tr}(D_k^{-1}\widetilde{\Sigma}_k).$$

Hence, we obtain $J(\widetilde{\Sigma}) \doteq -\frac{1}{2}\sum_k \operatorname{tr}[(\Sigma^{-1} + D_k^{-1})\widetilde{\Sigma}_k] - \log|\widetilde{\Sigma}_k|$ which is the desired result.

Similarly, by substituting $\beta_{rik}(\widetilde{\Gamma}, \sigma^2)$ and $S(\widetilde{\Gamma}, \mu)$ and looking at the result as a function only of $\widetilde{\mu}$, we obtain

$$J(\tilde{\mu}) = -\frac{1}{2}\sum_k \Big[\sum_{r,i} \tau_{rik}\frac{\|x_{ri} - \widetilde{\mu}_{rk}\|^2}{\sigma_r^2} + (\widetilde{\mu}_k - \mu)^T\Sigma^{-1}(\widetilde{\mu}_k - \mu)\Big].$$

Let us simplify the sum over r and i. Up to constants as function of $\widetilde{\mu}$, we have

$$\begin{aligned} \sum_i \tau_{rik}\|x_{ri} - \widetilde{\mu}_{rk}\|^2 &\doteq \sum_i \tau_{rik}\big(\|\widetilde{\mu}_{rk}\|^2 - 2\langle x_{ri}, \widetilde{\mu}_{rk}\rangle\big) \\ &= \bar{\tau}_{rk}\|\widetilde{\mu}_{rk}\|^2 - 2\langle \bar{x}_{rk}, \widetilde{\mu}_{rk}\rangle \\ &= \bar{\tau}_{rk}\big(\|\widetilde{\mu}_{rk}\|^2 - 2\langle \bar{\mu}_{rk}, \widetilde{\mu}_{rk}\rangle\big) \\ &\doteq \bar{\tau}_{rk}\|\widetilde{\mu}_{rk} - \bar{\mu}_{rk}\|^2 \end{aligned}$$

where the second to last equality is by definition of $\bar{\mu}_{rk} := \bar{x}_{rk}/\bar{\tau}_{rk}$. Recalling the definitions of $\bar{\tau}_{rk}$ and $\bar{x}_{rk} := \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$. Hence,

$$\sum_{r,i} \tau_{rik} \frac{\|x_{ri} - \widetilde{\mu}_{rk}\|^2}{\sigma_r^2} \doteq \sum_r \frac{\bar{\tau}_{rk}}{\sigma_r^2} \|\widetilde{\mu}_{rk} - \bar{\mu}_{rk}\|^2 = (\widetilde{\mu}_k - \bar{\mu}_k)^T D_k^{-1} (\widetilde{\mu}_k - \bar{\mu}_k)$$

Thus, we obtain

$$J(\widetilde{\mu}) \doteq -\frac{1}{2} \sum_k \big[(\widetilde{\mu}_k - \bar{\mu}_k)^T D_k^{-1} (\widetilde{\mu}_k - \bar{\mu}_k) + (\widetilde{\mu}_k - \mu)^T \Sigma^{-1} (\widetilde{\mu}_k - \mu) \big].$$

Desired expression (19) follows by applying the following lemma.

Lemma 3 (Sum of quadratic forms)**.** *For symmetric matrices $Q_1, Q_2, \dots,$*

$$\sum_r (x - m_r)^T Q_r^{-1} (x - m_r) = (x - m)^T Q^{-1} (x - m) + \mathit{const.}, \quad \forall x$$

where $Q = (\sum_r Q_r^{-1})^{-1}$ and $m = \sum_r Q Q_r^{-1} m_r$.

Proof *Since the two sides are quadratic functions, they are equal up to constants if their derivatives up to second-order match. Equating the Hessians gives $\sum_r Q_r^{-1} = Q^{-1}$. Then, equating the gradients gives $\sum_r Q_r^{-1}(x - m_r) = Q^{-1}(x - m)$, which simplifies to $\sum_r Q_r^{-1} m_r = Q^{-1} m$ in light of the Hessian equality.* ■

B.4. Updates of σ^2, π, p and q

The relevant portion of J as a function (σ_r^2) is

$$J((\sigma_r^2)) = -\frac{1}{2} \sum_r \Big[\frac{1}{\sigma_r^2} \sum_{i,k} \tau_{rik} \big[\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr} \big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2 \big] + d_r N_r \log \sigma_r^2 \Big]. \tag{42}$$

The maximizer of the function $x \mapsto A x^{-1} + B \log x$ is A/B (assuming $A, B > 0$), from which (20) follows.

As a function π, J has the form $J(\pi) \doteq \sum_{r,i,k} \tau_{rik} \log \pi_{rk} = \sum_{r,k} \bar{\tau}_{rk} \log \pi_{rk}$ using the definition of $\bar{\tau}_{rk}$. The following lemma is standard. (Recall that $\mathcal{P}_K$ is the set of probability K-vectors.)

Lemma 4. *For any nonnegative vector $(a_1, \dots, a_K)$,*

$$\operatorname*{argmax}_{p \,\in\, \mathcal{P}_K} \sum_k a_k \log p_k = \frac{1}{\sum_k a_k} (a_1, \dots, a_K).$$

Based on the lemma, π_1-update is $\pi_1 = (\bar{\tau}_{11}, \dots, \bar{\tau}_{1K}) / (\sum_k \bar{\tau}_{1k})$. The update for π_2 is similar.

To update p we note that $J(p) \doteq \sum_{i,j} \gamma_{ij}(\tau)(A_{ij} \log p + (1 - A_{ij}) \log(1 - p))$. The update is obtained by setting the derivative to zero. The q-update is similar.

Appendix C. Details for degree-corrected algorithm

C.1. τ-update with degree restriction

In this section, we derive a dual ascent algorithm for the optimization problem (33) which has to be solved for updating τ under the degree corrected model. Letting $a_{ik} := \phi_1[A\tau_2]_{ik} + \phi_0\bar{\tau}_{2k} + \xi_{1ik}$, and with some notational simplifications, problem (33) can be stated as

$$\min_{X=(x_{ik})} -\sum_{ik} x_{ik}(a_{ik} - \log x_{ik}), \quad \text{s.t. } X \in \mathcal{P}_{n,K}, \ \sum_i x_{ik}(\theta_i - 1) = 0, \forall k \tag{43}$$

where $\mathcal{P}_{n,K} := \{(x_{ik}) \in \mathbb{R}_+^{n\times K} : \sum_k x_{ik} = 1, \forall i\}$. Let $f(X)$ be the objective function in (43), and let us write the constraint in vector form $\sum_i x_{ik}(\theta_i - 1) = X^T(\theta - \mathbf{1}) = 0$. With the Lagrangian $L(X, \lambda) = f(X) - \lambda^T[X^T(\theta - \mathbf{1})]$, the dual function is

$$\Phi(\lambda) := \min_{X\in\mathcal{P}_{n,K}} L(X, \lambda),$$

and the dual problem is $\max_\lambda \Phi(\lambda)$. A dual-descent algorithm maximizes Φ by performing a gradient ascent on Φ: $\lambda^+ = \lambda + \mu\nabla\Phi(\lambda)$. We know that the gradient of Φ is given by $\partial_\lambda L(X, \lambda)$ evaluated at $X^*(\lambda)$, the optimizer of the Lagrangian. More precisely,

$$\nabla\Phi(\lambda) = [X^*(\lambda)]^T(\theta - \mathbf{1}), \quad \text{where } X^*(\lambda) = \underset{X\in\mathcal{P}_{n,K}}{\operatorname{argmax}} -L(X, \lambda)$$

Solving for $X^*(\lambda)$ is an instance of the problem in Lemma 1. The problem is separable over i and for fixed i, we are maximizing $\sum_k x_{ik}(a_{ik} - \log x_{ik}) + \sum_k \lambda_k x_{ik}(\theta_i - 1) = \sum_k x_{ik}\{[a_{ik} + \lambda_k(\theta_i - 1)] - \log x_{ik}\}$ over $x_{i*} \in \mathcal{P}_{1,K}$, the solution of which is given by the softmax operation

$$x^*_{ik}(\lambda) \propto_k \exp(a_{ik} + \lambda_k(\theta_i - 1)). \tag{44}$$

Thus, the update for the dual descent can be written

$$\lambda_k^+ = \lambda_k - \mu\sum_i x^*_{ik}(\lambda)(\theta_i - 1), \quad \forall k. \tag{45}$$

C.2. θ-update

Simplyfying the notation, let $h(\theta) = -\sum_i d_i \log\theta_i$ and $V := \{\theta : \sum_i \tau_{ik}(\theta_i - 1) = 0\}$. The problem is equivalent to minimizing $h(\theta) + \delta_V(\theta)$ over θ where δ_V is the indicator of V in the sense of convex analysis. Douglas-Rachford algorithm, also known as Spingarn's method of partial inverses in this special case, is given by

$$\begin{aligned} \theta^+ &= \operatorname{prox}_{th}(\xi) \\ \xi^+ &= \xi + P_V(2\theta^+ - \xi) - \theta^+ \end{aligned} \tag{46}$$

where prox_{th} is the proximal operator of $th(\cdot)$ and P_V is the projection onto V. Due to separability, it is not hard to see that $[\operatorname{prox}_{th}(\theta)]_i = \operatorname{prox}_{td_i\log(\cdot)}(\theta_i)$. This univariate proximal operator can be easily shown to coincide with $[f_t(\theta, d)]_i$ as given in (35).

As for the projection, in general with $C = \{x : \ Ax = b\}$, we have $P_C(x) = x + A^T(AA^T)^{-1}(b - Ax)$. Note that $V = \{\theta : \ \tau^T(\theta - \mathbf{1}) = 0\}$. Applying the general result with $A = \tau^T$ and $b = \tau^T\mathbf{1}$, we get $P_V(\theta) = \theta + \tau(\tau^T\tau)^{-1}\tau^T(\mathbf{1} - \theta) = \theta - H(\theta - \mathbf{1})$, with the obvious choice for H. Thus (46) simplifies to

$$\xi^+ = \xi + (2\theta^+ - \xi) - H(2\theta^+ - \xi - \mathbf{1}) - \theta^+$$

which gives the claimed update.

Appendix D. Expected average degree

Let $\hat{\lambda}_i = \sum_j A_{ij}$ and $\hat{\lambda}_j = \sum_i A_{ij}$ be the degree of node i from group 1, and node j from group 2, respectively. Then, the average degree of the network is

$$\hat{\lambda} = \frac{\sum_i \hat{\lambda}_i + \sum_j \hat{\lambda}_j}{N_1 + N_2} = \frac{2\sum_{i,j} A_{ij}}{N_1 + N_2}.$$

Recall the definition of clusters C_{rk} from (1). The expected average degree can be derived as follows: Assume that $i \in C_{1k}$, then $\mathbb{E}[A_{ij}] = \theta_{1i}\theta_{2j}(p1\{j \in C_{2k}\} + q1\{j \notin C_{2k}\})$. Then

$$\mathbb{E}(\hat{\lambda}_i) = \theta_{1i}\Big(p \sum_{j \in C_{2k}} \theta_{2j} + q \sum_{j \notin C_{2k}} \theta_{2j}\Big) = \theta_i r_k, \quad \text{where} \quad r_k := |C_{2k}|p + (N_2 - |C_{2k}|)q$$

Here, we have used the normalization (7): $\sum_{j \in C_{2\ell}} \theta_{2j} = |C_{2\ell}|$ for all $\ell \in [K]$. Now,

$$\sum_{i=1}^{N_1} \mathbb{E}(\hat{\lambda}_i) = \sum_{i=1}^{N_1} \sum_k 1\{i \in C_{1k}\} \mathbb{E}(\hat{\lambda}_i) = \sum_k r_k \sum_{i=1}^{N_1} \theta_{1i} 1\{i \in C_{1k}\} = \sum_k r_k |C_{1k}|$$

using the normalization of θ_{1i}. Dividing by $N_1 N_2$ and using $|C_{rk}|/N_r = \pi_{rk}$ for $r = 1, 2$,

$$\sum_{i=1}^{N_1} \mathbb{E}(\hat{\lambda}_i) = N_1 N_2 \sum_k \big[\pi_{1k} q + \pi_{1k}\pi_{2k}(p - q)\big] = N_1 N_2 (q + \Pi(p - q)),$$

where $\Pi := \sum_k \pi_{1k}\pi_{2k}$. By symmetry, $\sum_{j=1}^{N_2} \mathbb{E}(\hat{\lambda}_j) = N_1 N_2 (q + \Pi(p - q))$. Hence,

$$\lambda = \mathbb{E}[\hat{\lambda}] = \frac{2N_1 N_2}{N_1 + N_2}\big(q + \Pi(p - q)\big), \quad \text{where } \Pi := \sum_k \pi_{1k}\pi_{2k}.$$

Note that $\frac{2N_1 N_2}{N_1 + N_2} = 2/(N_1^{-1} + N_2^{-1})$ is the harmonic mean of N_1 and N_2.

Appendix E. More simulations

Figure 13 shows the effect of varying the dimension of the covariates $d = (d_1, d_2)$ and the scale of the covariance matrix ν. The setup is as in Section 4, and in particular $\Sigma = \nu I$ controls how far apart the centers of the covariate clusters $v_{*k} \in \mathbb{R}^{d_1 + d_2}$ are. Only the `mbiSBM` algorithm is shown (with no degree correction and) initialized with Dirichlet-perturbed truth (∼`rnd`). As one expects, increasing the dimensions of the covariates increases the performance (seemingly without bound). Increasing ν improves the performance up to a point, but there is a saturation effect beyond that point, where the performance remains more or less the same.

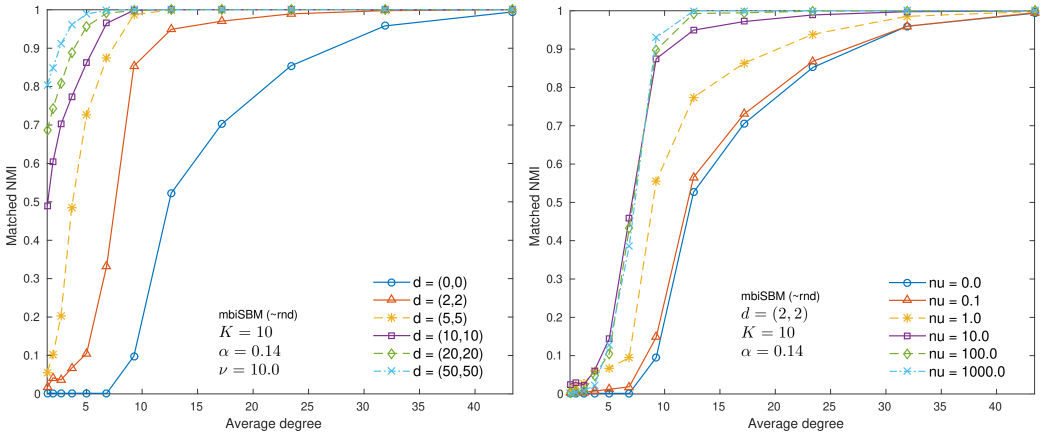


Figure 13: (Left) Effect of changing the dimension of covariates $d = (d_1, d_2)$ and (right) the parameter ν where $\Sigma = \nu I_{d_1+d_2}$.

References

Emmanuel Abbe. Community detection and stochastic block models: recent developments. *arXiv preprint arXiv:1703.10146*, 2017.

Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1):471–487, 2016.

Arash A. Amini and Elizaveta Levina. On semidefinite relaxations for the block model. *The Annals of Statistics*, 46(1):149–179, 2018.

Arash A. Amini, Aiyou Chen, Peter J. Bickel, and Elizaveta Levina. Pseudo-likelihood methods for community detection in large sparse networks. *Annals of Statistics*, 41(4):2097–2122, 2013.

Nils D Arvold, Alphonse G Taghian, Andrzej Niemierko, Rita F Abi Raad, Meera Sreedhara, Paul L Nguyen, Jennifer R Bellon, Julia S Wong, Barbara L Smith, and Jay R Harris. Age, breast cancer subtype approximation, and local recurrence after breast-conserving therapy. *Journal of Clinical Oncology*, 29(29):3885, 2011.

Shantanu Banerji, Kristian Cibulskis, Claudia Rangel-Escareno, Kristin K Brown, Scott L Carter, Abbie M Frederick, Michael S Lawrence, Andrey Y Sivachenko, Carrie Sougnez, Lihua Zou, et al. Sequence analysis of mutations and translocations across breast cancer subtypes. *Nature*, 486(7403):405, 2012.

Mukesh Bansal, Giusy Della Gatta, and Diego Di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, 2006.

Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(October):509–512, 1999.

Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101, 2004.

Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B*, 48(3):259–302, 1986.

Peter J Bickel and Aiyou Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proceedings of the National Academy of Sciences of the United States of America*, 106(50):21068–21073, 2009.

Norbert Binkiewicz, Joshua T Vogelstein, and Karl Rohe. Covariate-assisted spectral clustering. *Biometrika*, 104(2):361–377, 2017.

Gilles Bisson and Fawad Hussain. Chi-Sim: A new similarity measure for the co-clustering task. *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008*, pages 211–217, 2008.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Maggie CU Cheang, Stephen K Chia, David Voduc, Dongxia Gao, Samuel Leung, Jacqueline Snider, Mark Watson, Sherri Davies, Philip S Bernard, Joel S Parker, et al. Ki67 index, her2 status, and prognosis of patients with luminal b breast cancer. *JNCI: Journal of the National Cancer Institute*, 101(10):736–750, 2009.

Yizong Cheng and George M Church. Biclustering of expression data. *Proceedings of the International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, 8:93–103, 2000.

David Choi and Patrick J. Wolfe. Co-clustering separately exchangeable network data. *Annals of Statistics*, 42(1):29–63, 2014.

Anirban Dasgupta, J.E. Hopcroft, and F. McSherry. Spectral Analysis of Random Graphs with Skewed Degree Distributions. *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 602–610, 2004.

Eric Davidson and Michael Levin. Gene regulatory networks. *Proceedings of the National Academy of Sciences*, 102(14):4935–4935, 2005.

DBLP. *Computer Science Bibliography.* URL `https://dblp.uni-trier.de`.

Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters*, 107(6):1–5, 2011.

Inderjit S Dhillon. Co-clustering documents and words using Bipartite Co-clustering documents and words using Bipartite Spectral Graph Partitioning. *Proc of 7th ACM SIGKDD Conf*, pages 269–274, 2001.

Inderjit S Dhillon. Information-Theoretic Co-clustering. *Proc. of 9th ACM SIGKDD Int'l Conf.*, pages 89–98, 2003.

Jim Douglas and H.H Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical society*, 82:421–439, 1956.

Cheryl J. Flynn and Patrick O. Perry. Consistent Biclustering. *arXiv preprint arXiv:1206.6927*, 2012.

Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. Community Detection in Degree-Corrected Block Models. *arXiv preprint arXiv:1607.06993*, 2016.

Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. Achieving Optimal Misclassification Proportion in Stochastic Block Model. *The Journal of Machine Learning Research*, 18(1):1980–2024, 2017.

Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC Texts in Statistical Science, 2003.

Mark B Gerstein, Joel Rozowsky, Koon-Kiu Yan, Daifeng Wang, Chao Cheng, James B Brown, Carrie A Davis, LaDeana Hillier, Cristina Sisu, Jingyi Jessica Li, et al. Comparative analysis of the transcriptome across distant species. *Nature*, 512(7515):445, 2014.

Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences of the United States of America*, 110(36):14534–9, 2013.

Bruce Hajek, Yihong Wu, and Jiaming Xu. Achieving Exact Cluster Recovery Threshold via Semidefinite Programming: Extensions. *IEEE Transactions on Information Theory*, 62(10): 5918–5937, 2016.

J. A. Hartingan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Society*, 67(337):123–129, 1972.

Paul H Harvey, Mark D Pagel, et al. *The comparative method in evolutionary biology*, volume 239. Oxford university press Oxford, 1991.

Michael Hecker, Sandro Lambeck, Susanne Toepfer, Eugene Van Someren, and Reinhard Guthke. Gene regulatory network inference: data integration in dynamic models-a review. *Biosystems*, 96(1):86–103, 2009.

Ingrid Hedenfalk, David Duggan, Yidong Chen, Michael Radmacher, Michael Bittner, Richard Simon, Paul Meltzer, Barry Gusterson, Manel Esteller, Mark Raffeld, et al. Gene-expression profiles in hereditary breast cancer. *New England Journal of Medicine*, 344(8):539–548, 2001.

Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

ipapi. *IP Lookup API*. URL `http://ipapi.co`.

Reika Iwakawa, Takashi Kohno, Yasushi Totoki, Tatsuhiro Shibata, Katsuya Tsuchihara, Sachiyo Mimaki, Koji Tsuta, Yoshitaka Narita, Ryo Nishikawa, Masayuki Noguchi, et al. Expression and clinical significance of genes frequently mutated in small cell lung cancers defined by whole exome/rna sequencing. *Carcinogenesis*, 36(6):616–621, 2015.

Lei Jing and Alessandro Rinaldo. Consistency of spectral clustering in stochastic block models. *Annals of Statistics*, 43(1):215–237, 2015.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

David Kahle and Hadley Wickham. ggmap : Spatial Visualization with. *The R Journal*, 5(1): 144–161, 2013.

Brian Karrer and M. E J Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):1–11, 2011.

Brian Keegan. *GitHub repository: 2014-On-Wikipedia*, 2014. URL `https://github.com/brianckeegan/2014-On-Wikipedia`.

Mikko Kivela, Alex Arenas, Marc Barthelemy, James P. Gleeson, Yamir Moreno, and Mason A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, 2014.

Daniel B. Larremore, Aaron Clauset, and Abigail Z. Jacobs. Efficiently inferring community structure in bipartite networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 90(1):17–22, 2014.

Jingyi Jessica Li, Haiyan Huang, Peter J Bickel, and Steven E Brenner. Comparison of d. melanogaster and c. elegans developmental stages, tissues, and cells by modencode rna-seq data. *Genome research*, 24(7):1086–1101, 2014.

Ben-Yang Liao and Jianzhi Zhang. Evolutionary conservation of expression profiles between human and mouse orthologous genes. *Molecular biology and evolution*, 23(3):530–540, 2005.

Sara C Madeira, Miguel C Teixeira, Isabel Sa-Correia, and Arlindo L Oliveira. Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 7(1): 153–165, 2010.

Piyush B Madhamshettiwar, Stefan R Maetschke, Melissa J Davis, Antonio Reverter, and Mark A Ragan. Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome medicine*, 4(5):41, 2012.

Francesco M. Malvestuto. Statistical treatment of the information content of a database. *Information Systems*, 11(3):211–223, 1986.

Laurent Massoulié. Community detection thresholds and the weak Ramanujan property. *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 694–706, 2014.

Elchanan Mossel, Joe Neeman, and Allan Sly. A Proof Of The Block Model Threshold Conjecture. *Combinatorica*, pages 1–44, 2013.

Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probab. Theory Relat. Fields*, 162(3):431–461, 2015.

M. E. J. Newman and Aaron Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(May):1–16, 2016.

Daniel O'Connor and Lieven Vandenberghe. Primal-Dual Decomposition by Operator Splitting and Applications to Image Deblurring. *SIAM Journal on Imaging Sciences*, 7(3):1724–1754, 2014.

Neelroop N Parikshak, Michael J Gandal, and Daniel H Geschwind. Systems biology and gene networks in neurodevelopmental and neurodegenerative disorders. *Nature Reviews Genetics*, 16(8):441, 2015.

Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. *Advances in Neural Information Processing Systems*, pages 1–9, 2013.

Zahra S. Razaee, Arash A. Amini, and Jingyi Li. *GitHub repository: mbiSBM*. URL `https://github.com/aaamini/mbisbm`.

Dan Robinson, Eliezer M Van Allen, Yi-Mi Wu, Nikolaus Schultz, Robert J Lonigro, Juan-Miguel Mosquera, Bruce Montgomery, Mary-Ellen Taplin, Colin C Pritchard, Gerhardt Attard, et al. Integrative clinical genomics of advanced prostate cancer. *Cell*, 161(5):1215–1228, 2015.

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *Annals of Statistics*, 39(4):1878–1915, 2011.

Karl Rohe, Tai Qin, and Bin Yu. Co-clustering directed graphs to discover asymmetries and directional communities. *Proceedings of the National Academy of Sciences of the United States of America*, 113(45):12679–12684, 2016.

Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166, 2003.

Therese Sørlie, Charles M Perou, Robert Tibshirani, Turid Aas, Stephanie Geisler, Hilde Johnsen, Trevor Hastie, Michael B Eisen, Matt Van De Rijn, Stefanie S Jeffrey, et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, 98(19):10869–10874, 2001.

Therese Sørlie, Robert Tibshirani, Joel Parker, Trevor Hastie, James Stephen Marron, Andrew Nobel, Shibing Deng, Hilde Johnsen, Robert Pesich, Stephanie Geisler, et al. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the national academy of sciences*, 100(14):8418–8423, 2003.

Jie Tang, Sen Wu, Jimeng Sun, , and Hang. Su. Cross-domain collaboration recommendation. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1285–1293, 2012.

Bert Vogelstein and Kenneth W Kinzler. Cancer genes and the pathways they control. *Nature medicine*, 10(8):789, 2004.

Gunter Von Minckwitz, Michael Untch, Jens-Uwe Blohmer, Serban D Costa, Holger Eidtmann, Peter A Fasching, Bernd Gerber, Wolfgang Eiermann, Jörn Hilfrich, Jens Huober, et al. Definition and impact of pathologic complete response on prognosis after neoadjuvant chemotherapy in various intrinsic breast cancer subtypes. *J Clin oncol*, 30(15):1796–1804, 2012.

Martin J. Wainwright and Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Kai Wang, Junsuo Kan, Siu Tsan Yuen, Stephanie T Shi, Kent Man Chu, Simon Law, Tsun Leung Chan, Zhengyan Kan, Annie SY Chan, Wai Yin Tsui, et al. Exome sequencing identifies frequent mutation of arid1a in molecular subtypes of gastric cancer. *Nature genetics*, 43(12): 1219, 2011.

Wikimedia Statistics. URL `https://stats.wikimedia.org`.

Jason Wyse, Nial Friel, and Pierre Latouche. Inferring structure in bipartite networks using the latent block model and exact ICL. (2013):23, 2014.

Bowei Yan and Purnamrita Sarkar. Convex Relaxation for Community Detection with Covariates. *arXiv preprint arXiv:1607.02675*, 2016.

Yuan Zhang, Elizaveta Levina, and Ji Zhu. Community Detection in Networks with Node Features. *Electronic Journal of Statistics*, 10(2):3153–3178, 2016.

Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *Annals of Statistics*, 40(4):2266–2292, 2012.

Tao Zhou, Jie Ren, Matúš Medo, and Yi Cheng Zhang. Bipartite network projection and personal recommendation. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 76(4):1–7, 2007.

Zhixin Zhou and Arash A. Amini. Optimal bipartite network clustering. *arXiv preprint arXiv:1803.06031*, 2018.