Title
Community Detection in Networks with Node Covariates

Author
Razaee, Zahra

Publication Date
2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Community Detection in Networks with Node Covariates

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Zahra Razaee

2017

ABSTRACT OF THE DISSERTATION

Community Detection in Networks with Node Covariates

by

Zahra Razaee
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2017
Professor Jingyi Li, Chair

Community detection or clustering is a fundamental task in the analysis of network data. Most networks come with annotations which can be in form of node covariates such as a person's age, gender and location and/or edge covariates such as time stamps and ratings. However, most of the existing community detection approaches infer the community memberships merely based on the network structure. Moreover, many real networks have a bipartite structure which makes community detection challenging. In this dissertation, we first propose a model-based approach 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. 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.

We also propose a unified affinity matrix (USim) to leverage the node covariates information that can be used in unipartite networks (directed and undirected) as well as the bipartite networks that combines the information from the network with that from the node covariates into a single similarity matrix, which can then be input to a spectral clustering algorithm. We show the effectiveness of both approaches on simulated and real data, namely, page-user networks collected from Wikipedia.

The dissertation of Zahra Razaee is approved.

Mark Stephen Handcock

Yingnian Wu

Sharmodeep Bhattacharyya

Jingyi Li, Committee Chair

University of California, Los Angeles

2017

To my family ...

for all their love and support

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank my advisor, Jingyi Jessica Li, for all her help, advice, and encouragement throughout the course of my dissertation research. I would also like to thank the members of my committee, Mark Handcock, Yingnian Wu and Sharmodeep Bhattacharyya for providing insightful comments on my work. I am also very grateful to Mark Rudelson from the University of Michigan for his guidance and invaluable feedback.

Special thanks to my parents for their unconditional love, support and help throughout the years and for generating my love for mathematical science. Special thanks to my husband for fueling my interest in statistics and his support, encouragement, advice and patience. Special thanks to my brother who has always been a role model for me in every aspect of life.

Finally, I would like to thank the staff of statistics department, especially Glenda Jones. I would also like to thank my friends at UCLA, especially Wei Li, Jiaping Zhu, Medha Uppala, Joshua Gordon, Seunghyun Min and numerous others and also my friends in the statistics department of the University of Michigan and the many great friends whose support and well wishing has had an immense impact. I wish I could name you all.

VITA

2007	BSc in Applied Mathematics, University of Tehran, Tehran, Iran
2009	MSc in Industrial Engineering, Amirkabir University of Technology, Tehran, Iran
2011-2014	Graduate Student Instructor, University of Michigan
2014	MA in Statistics, University of Michigan - Ann Arbor
2014-2016	Teaching Assistant, UCLA
2015-2016	Statistical Consultant, UCLA
2016	Main Instructor, Introduction to Statistical Programming with R, UCLA

PUBLICATIONS

Zahra S. Razaee, Arash A. Amini, and Jingyi Jessica Li. "Matched bipartite block model with covariates." arXiv preprint arXiv:1703.04943 (2017).

Li, Wei Vivian, Zahra S. Razaee, and Jingyi Jessica Li. "Epigenome overlap measure (EPOM) for comparing tissue/cell types based on chromatin states." BMC genomics 17.1 (2016): S10.

M.H. Fazel Zarandi, Zahra S. Razaee, A Fuzzy Clustering Model for Fuzzy Data With Outliers, International Journal of Fuzzy System Applications, Volume 1, issue 2, 2011.

M.H. Fazel Zarandi, Zahra S. Razaee, M. Karbasian, A Fuzzy Case Based Reasoning Approach to Value Engineering, International Journal of Expert Systems with Applications, 2009.

CHAPTER 1

Introduction

The analysis of complex systems from network perspective, has gained a lot of attention in a wide variety of disciplines in recent years. Many studies have shown that the network representation of such systems, despite its simplicity, provide insights into the system's structure, dynamics, and function in an intuitive fashion. The interest has been fueled by the increasing availability of network data coming form a variety of sources in science and engineering such as online social networks, various forms of biological networks (protein and gene interaction networks, gene-regulatory networks), fMRI networks, ecological networks, computer and communication networks and so on.

Faced with ever-increasing network data and their sizes, there has been a need for organization and structure. Out of this search for structure, community detection has emerged as one of the fundamental problems of network analysis. 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. Traditionally, the notion of community or cluster has been vaguely defined. However, framing the problem in the context of network analysis, there is now a well-established probabilistic model, namely, the Stochastic Block Model (SBM)[1], which allows for a precise notion of community structure. Stochastic block models and its variants [2, 3], have been extensively investigated in recent years both in terms of theoretical properties and efficient fitting algorithms. See, for instance [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18] for a sample of the work.

Another popular approach in community detection is spectral clustering. Unlike probabilistic models, spectral clustering is a simple model-free approach which is a relaxation of a

cost minimization problem and it is shown to be accurate and computationally feasible for both sparse and dense regime.

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 transcription factors and their binding sites. 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.

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 [19]. 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 [20, 21] and text mining [22, 23, 24]. Recently, [25, 26] studied likelihood-based co-clustering. [27] 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 [28]. Despite a moderate reduction in the dimension (having to deal with two smaller networks), the projection approach suffers from information loss and identifiability issues [28]. Projection can also turn a structured bipartite network into unstructured unipartite ones or vice versa [29]. 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. [30] proposed a method to infer both community memberships as well as the number of communities in a bipartite network using a blockmodel and an algorithm similar to the iterated conditional modes [31]. [29] has proposed a bipartite stochastic block model (BiSBM) that built on the work of [2] 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.

Most networks come with annotations which can be in form of node covariates such as a person's age, gender and location and/or edge covariates such as time stamps and ratings. However, most of the existing community detection approaches infer the community memberships merely based on the network structure. Recently there have been some works to combine the information from the network with that from the node covariates [32, 33, 34, 35, 36]. In [35], the authors proposed a spectral approach where one adds the gram matrix of the node covariates to the regularized graph Laplacian squared weighted by a tuning parameter. They then used spectral clustering on top eigenvectors of the sum.

A joint criterion for community detection (JCDC) with covariates is proposed by [34]. Their approach enables learning different influence on each covariate as well as the flexibility of choosing the amount of influence the covariate information has on communities.

In [32], the authors included node covariate for degree-corrected stochastic block models. They introduced a dependence on node covariate via a set of prior probabilities of nodes belonging to communities. They used belief propagation scheme to perform inference of the optimal community assignments. However, their method only works for unipartite networks with one-dimensional node covariate.

A semidefinite relaxation for clustering a network with covariates was proposed in [33]. They have theoretically shown that their proposed SDP is able to separate clusters even in the extreme case when the network and the covariates carry "orthogonal" pieces of information about the cluster memberships.

The challenge in jointly analyzing the network and covariate information in bipartite setting is that node covariates for the two modes are (usually) of different dimensions. To the best of our knowledge, there has been no study on community detection in bipartite networks with covariates of dimension greater than one. In this dissertation, we aim at filling the gap in the literature by providing scalable methods to do community detection in networks with node covariates.

The dissertation is organized as follows: We start with some general background material in Chapter 2. This chapter reviews some aspects of vector-matrix analysis as well as concentration inequalities and kernel and feature maps.

In Chapter 3, we propose a model (mbiSBM) which allows for matched communities in bipartite network, in addition to node covariates with information about the matching. Our model is an extension of the stochastic block model (SBM) and its degree-corrected version (DC-SBM). We fit our proposed model algorithm based on variational inference ideas.

In Chapter 4, we propose a spectral approach to combine network information with that of node covariates. Our method is a general framework that can be used for most settings including unipartite (directed and undirected) and bipartite. We define a new affinity matrix by combining the adjacency matrix as well as a kernel matrix that measures the similarity between nodes based on their covariates. We then apply the spectral clustering algorithm on this new affinity matrix. In other words, we will reweigh the original network by incorporating its node covariates and will treat it as new weighted network. In case of bipartite network, we will combine two kernel matrices made from the node covariates of each side of the network with the bi-adjacency matrix and will apply a bipartite spectral clustering algorithm, biSC for short, which is a variant of the approach of [22] on our new weighted bi-adjacency matrix. We also propose a similar approach for the directed networks. We show the effectiveness of our algorithms on simulated and real data.

CHAPTER 2

Background

2.1 Primer on Vectors, Matrices and their norms

Vector l_p norms

The standard n-dimensional Euclidean space is denoted as $\mathbb{R}^n$. For a vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, and $p \in (0, \infty)$, we use

$$\|x\|_p := \begin{cases} \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}, & p < \infty, \\ \max_{1 \le i \le n} |x_i|, & p = \infty. \end{cases} \tag{2.1}$$

Note that $\|.\|_p$ is a (proper) norm for $p \in [1, \infty]$. However, for $p \in (0, 1)$, it defines a quasi-norm since it fails the triangle inequality.

We denote the unit ball of l_p as $\mathbb{B}_p := \{x : \|x\|_p \le 1\}$. If we want to emphasize the dimension we write $\mathbb{B}_p^n := \{z \in \mathbb{R}^n : \|x\|_p \le 1\}$. $\mathbb{R}^n$ equipped with l_p norm is usually written as $l_p^n = (\mathbb{R}^n, \|.\|_p)$. For $p \in [1, \infty]$, $\mathbb{B}_p$ is a convex set, a consequence of $\|.\|$ being a norm.

We also use the notation $\mathcal{S}^{n-1}$ to denote the set of unit vectors in $\mathbb{R}^n$.
Precisely, $\mathcal{S}^{n-1} = \{x \in \mathbb{R}^n : \|x\| = 1\}$.

Classes of matrices and spectral theorems

We denote the class of real-valued m-by-n matrices as $\mathbb{R}^{m \times n}$, the class of n-by-n (real-valued) symmetric matrices as $\mathbb{S}^n$ and the class of n-by-n positive semidefinite (PSD) matrices as $\mathbb{S}_+^n$. The class $\mathbb{R}^{n \times n}$ is a vector space and can be identified with $\mathbb{R}^{n^2}$ and hence has dimension n^2. The class $\mathbb{S}^n$ is a subspace of $\mathbb{R}^{n \times n}$ of dimension $n(n+1)/2$ and $\mathbb{S}_+^n$ is a convex cone in

this subspace. The cone structure of $\mathbb{S}^n_+$ induces a natural partial order which we denote by $A \preceq B$ and $B \succeq A$ if $A - B \in \mathbb{S}^n_+$.

We denote a generic eigenvalue of $A \in \mathbb{R}^{n\times n}$ as $\lambda(A)$ and minimum and maximum eigenvalues are denoted as $\lambda_{\min}$ and $\lambda_{\max}(A)$, respectively.

Eigenvalue Decomposition (EVD)

By the spectral theorem, any real symmetric matrix $A \in \mathbb{S}^n$ has a decomposition of the form

$$A = U\Lambda U^T \tag{2.2}$$

where Λ is diagonal with eigenvalue of A on its diagonal and $U \in \mathbb{R}^{n\times n}$ is an *orthogonal* matrix collecting the corresponding eigenvectors. By definition, an orthogonal matrix $U \in \mathbb{R}^{n\times n}$ is such that $U^TU = UU^T = I_n$, where I_n denotes the $n \times n$ identity matrix. The class of n-by-n orthogonal matrices will be denoted as $\mathbb{O}^n$. Note that only diagonalizable matrices can be factorized in this way.

The eigendecomposition allows for much easier computation of power series as well as inversion of matrices. Precisely, if $f(x)$ is given by $f(x) = a_0 + a_1x + a_2x^2 + \ldots$, then we know that

$$f(A) = Uf(\Lambda)U^T \tag{2.3}$$

Since Λ is a diagonal matrix, functions of Λ can be easily calculated:

$$[f(\Lambda)]_{ii} = f(\lambda_i) \tag{2.4}$$

and the inverse of A is given by

$$A^{-1} = U\Lambda^{-1}U^T \tag{2.5}$$

Singular Value Decomposition (SVD)

As a consequence of the spectral decomposition, any matrix $A \in \mathbb{R}^{m\times n}$ can be represented as

$$A = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T, \quad where \;\; r = rank(A). \tag{2.6}$$

Here the non-negative diagonal of Σ are called singular values of A and denoted as $\{\sigma_i(A)\}$, the vectors $u_i \in \mathbb{R}^m$ are called the left singular vectors of A, and the vectors $v_i \in \mathbb{R}^n$ are called the right singular vectors of A.

The left singular vectors u_i are the orthonormal eigenvectors of AA^* and the right singular vectors v_i are the orthonormal eigenvectors of A^*A. The singular values σ_i are the square roots of the eigenvalues λ_i of both AA^* and $A * A$:

$$\sigma_i(A) = \sqrt{\lambda_i(AA^*)} = \sqrt{\lambda_i(A^*A)} \tag{2.7}$$

If A is a symmetric matrix, the singular values of A are the absolute values of the eigenvalues λ_i of A:

$$\sigma_i(A) = |\lambda_i(A)| \tag{2.8}$$

and both left and right singular vectors of A are the eigenvectors of A.

Matrix Operator norm

The class of real-valued m-by-n matrices, denoted as $\mathbb{R}^{m\times n}$ can be equipped with several norms. In this section, we only mention two of them, operator and Frobenius norms.

Consider a matrix $\mathbb{R}^{m\times n}$. We can view the matrix as an operator $A : (\mathbb{R}^n, \|.\|_p) \to (\mathbb{R}^m, \|.\|_q)$. The corresponding operator norm is

$$\|A\|_{p,q} := \|A\|_{p\to q} := \max_{\|x\|_p \le 1} \|Ax\|_q = \max_{\|x\|_p = 1} \|Ax\|_q \tag{2.9}$$

When $p = q$, we also use $\|A\|_p = \|A\|_{p,p}$. Few special cases of $p = 1, 2, \infty$ are as following:

- the *spectral norm*: $\|A\|_2 = \|A\|_{2,2} = \sigma_{\max}(A)$
- the l_∞ *operator norm*: $\|A\|_\infty = \|A\|_{\infty,\infty} = \max_{i=1,\ldots,m} \sum_{j=1}^{n} |A_{ij}|$
- the l_1 norm : $\|A\|_1 = \|A\|_{1,1} = \max_{j=1,\ldots,m} \sum_{i=1}^{m} |A_{ij}|$,

For a symmetric matrix $A \in \mathbb{S}^n$, one has another useful expression for the spectral norm,

$$\|A\|_2 = \sup\{|\lambda| : \lambda \text{ is an eigen value of } A\} \tag{2.10}$$

The right hand side of the above expression is called the *spectral radius* and is denoted as $\rho(A)$. In general, for a non-symmetric matrix A, one only has $\rho(A) \leq \|A\|_2$. For a PSD matrix $A \in \mathbb{S}^n_+$, we get from (2.10) that $\|A\|_2 = \lambda_{\max}$.

Frobenius norm

Recall that the *trace* of a square matrix $A \in \mathbb{R}^{n\times n}$ is given by $tr(A) = \sum_i A_{ii}$. Given two square matrices $X, Y \in \mathbb{R}^{n\times n}$, we define the matrix inner product

$$\langle A, B\rangle := tr(AB^T) = \sum_{ij} A_{ij}B_{ij} \tag{2.11}$$

This inner product induces the *Frobenius norm*, also called *Hilbert-Schmidt* norm $\|A\|_F := \sqrt{\langle A, A\rangle}$. Thus,

$$\|A\|_F = \Big(\sum_{i=1}^{m}\sum_{j=1}^{n} |A_{ij}|^2\Big)^{1/2} = \Big(\sum_{i=1}^{n} \sigma_i^2(A)\Big)^{1/2} \tag{2.12}$$

Unitarily-invariant norms

A matrix norm is unitarily invariant if $\|A\| = \|UAW\|$ for unitary matrices U and W.

Let $A = U\Sigma V^T$ be a SVD of A, where $\Sigma = diag(\sigma_1, \ldots, \sigma_n)$ contains the singular values pf A, then

1. The operator norm $\|.\|_{op} = \|.\|_2$ is unitarily-invariant. Hence, $\|A\|_{op} = \|\Sigma\|_{op} = \max\{\sigma_i x_i^2 : \sum_i x_i^2 = 1\} = \|\sigma\|_\infty$.

2. The Frobenius norm is unitarily-invariant. This can be seen by writing $\|A\|_F^2 = \langle A, A\rangle = tr(A^T A) = tr(\Sigma^2) = \|\sigma\|_2^2$ using invariance of trace under circular permutations.

3. The nuclear norm, also known as the trace norm, defined by $\|A\|_* := \|\sigma\|_1 = \sum_{i=1}^n \sigma_i(A)$ is clearly unitarily invariant.

Using the relation between l_∞, l_2, l_1 norm we have

$$\|A\|_2 \leq \|A\|_F \leq \|A\|_* \tag{2.13}$$

Hermitian Dilation

Consider a matrix $B \in \mathbb{C}^{d_1 \times d_2}$, and set $d = d_1 + d_2$. The Hermitian dilation of B is defined as the matrix

$$\mathcal{D}(B) := \begin{bmatrix} 0 & B \\ B & 0 \end{bmatrix} \in \mathbb{H}^d \tag{2.14}$$

The dilation preserves spectral properties. That is,

$$\lambda_{max}(\mathcal{D}(B)) = \|\mathcal{D}(B)\| = \|B\|. \tag{2.15}$$

Second, the square of the dilation satisfies

$$\mathcal{D}(B)^2 = \begin{bmatrix} BB^* & 0 \\ 0 & B^*B \end{bmatrix} \in \mathbb{H}^d \tag{2.16}$$

One can study a random matrix, not necessarily Hermitian, by applying matrix concentration inequalities to the Hermitian dilation of the random matrix.

2.2 Concentration Inequalities

Consider a random variable X with mean $\mathbb{E}X$. Then, $|X - \mathbb{E}X|$ measures the (two-sided) deviation of X from its mean. A concentration inequality is an upper bound on the probability of the deviation being larger than say t. One of the simplest concentration inequality is the Chebyshev's Inequality:

Lemma 1 (Chebyshevs inequality)**.** *Let X be a random variable with mean $\mathbb{E}X$ and variance σ^2. Then, for any $t > 0$, we have*

$$\Pr(|X - \mathbb{E}X| \geq t) \leq \frac{\sigma^2}{t^2} \tag{2.17}$$

The Chebyshev's inequality is general but not sharp. We are seeking inequalities with the upper bound going to zero exponentially fast as $t \to 0$. More precisely, an inequality of the form

$$\Pr(|X - \mathbb{E}X| > t) \leq c_1 \exp(-c_2 t^2), \quad \forall t \in [0, \epsilon), \tag{2.18}$$

for some constants $c_1, c_2, \epsilon > 0$. In words, such inequalities capture sharp concentration of X around its mean, in a probabilistic sense. A one-sided version of the above, i.e., a bound on $\Pr(X - \mathbb{E}X > t)$ for example, is sometimes called a *deviation bound*; we however will not make that distinction. The exponent of t in (2.18) can be other than 2, although exponent of 2 is what we encounter in most cases of interest to us.

Sub-Gaussian Random Variables

A zero mean random variable X is sub-Gaussian with parameter σ if its moment generating function (m.g.f. for short) is bounded uniformly by that of a Gaussian random variable., that is, $\exists \sigma^2$ such that:

$$\mathbb{E}[\exp(\lambda X)] \leq \exp\left(\frac{\sigma^2 \lambda^2}{2}\right), \forall \lambda \in \mathbb{R} \tag{2.19}$$

We denote a sub-Gaussian random variable X with standard σ as

$$X \sim \text{SubGauss}(\sigma)$$

It follows from (2.19) and Markov inequality that for $\lambda, t \in [0, \infty)$,

$$\Pr(X \geq t) = \Pr(e^{\lambda X} \geq e^{\lambda t}) \leq e^{-\lambda t} \mathbb{E} e^{\lambda X} \leq \exp(\frac{\sigma^2 \lambda^2}{2} - \lambda t)$$

The right-hand side is minimized by taking $\lambda = \frac{t}{\sigma^2}$, leading to

$$\Pr(X \geq t) \leq \exp(-\frac{t^2}{2\sigma^2}) \tag{2.20}$$

This implies that if X is zero-mean and sub-Gaussian(σ), then $-X$ is also a sub-Gaussian random variable. This means:

$$\mathbb{P}(|X| \geq t) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right) \tag{2.21}$$

If X is non-zero mean, call it sub-Gaussian if $(X - \mathbb{E}[X])$ is sub-Gaussian.

Examples

- Gaussian distribution: $X \sim N(0,1) \Rightarrow \mathbb{E}[\exp(\lambda X)] = \exp(\frac{\sigma^2\lambda^2}{2}), \forall\lambda \in \mathbb{R}$
- A zero-mean random variable which is bounded almost surely, that is, $|X| \leq C$, a.s., for some constant $C > 0$.

Properties

The following are equivalent alternative characterizations of sub-Gaussians. Let K_i be constants that differ from each other by a constant factor, for $i = 1, \ldots, 5$.

1. Tail bound: $\forall t \geq 0, \quad \mathbb{P}(|X| \geq t) \leq 2\exp\left(-\frac{t^2}{K_1^2}\right)$
2. Moments: $\forall p \geq 1, \quad \|X\|_p \coloneqq \mathbb{E}[|X|^p]^{1/p} \leq K_2\sqrt{p}$
3. MGF of X^2 (near 0): $\forall\lambda \quad \text{such that} \quad |\lambda| \leq \frac{1}{K_3} \quad \text{or} \quad K_3^2\lambda^2 \leq 1,$

$$\mathbb{E}[\exp(\lambda^2 X^2)] \leq \exp(K_3^2\lambda^2)$$

4. The MGF of X^2 is bounded at some point: $\mathbb{E}\left[\exp\left(\frac{X^2}{K_4^2}\right)\right] \leq 2$
   Alternatively, $\exists c$ such that $\mathbb{E}[\exp(cX^2)] \leq \infty$.

If $\mathbb{E}[X] = 0$, then properties (1-4) are also equivalent to the following 5th property:

5. MGF of X: $\forall\lambda \in \mathbb{R}, \quad \mathbb{E}[\exp(\lambda X)] \leq \exp(K_5^2\lambda^2)$

Sub-Gaussian Norm

The sub-Gaussian norm is defined as follows:

$$\|X\|_{\psi_2} = \inf\left\{t > 0 : \mathbb{E}\left[\exp\left(\frac{X^2}{t^2}\right)\right] \leq 2\right\} \tag{2.22}$$

Which lets us write the following inequalities:

1. $\mathbb{P}(|X| > t) \leq 2\exp(-ct^2/\|X\|_{\psi 2}^2), \forall t \geq 0$
2. $\|X\|_p \leq C\|X\|_{\psi 2}\sqrt{p}, \forall p \geq 1$
3. $\mathbb{E}[\exp(X^2/\|X\|_{\psi 2}^2)] \leq 2$
4. $\mathbb{E}[\exp(\lambda X)] \leq \exp(C\|X\|_{\psi 2}^2\lambda^2)$

Lemma 2. *If $X_1, \ldots, X_n \sim subGauss(\sigma_i)$ and independent, then $\sum_{i=1}^n X_i \sim subGauss(\sum_{i=1}^n \sigma_i^2)$. Equivalently, if $\{X_i\}$ are independent, then $\|\sum_{i=1}^n X_i\|_{\psi 2}^2 \leq C\sum_{i=1}^n \|X_i\|_{\psi 2}^2$. Thus the following inequality holds:*

$$\Pr(|\sum_i (X_i - \mathbb{E}X_i)| > t) \leq 2\exp(-\frac{t^2}{2\sum_i \sigma_i^2}) \tag{2.23}$$

Alternately,

$$\Pr(|\sum_i (X_i - \mathbb{E}X_i)| > t) \leq 2\exp(-\frac{t^2}{\|X_i\|_{\psi 2}^2})$$

Sub-Exponential Random Variables

The class of sub-gaussian distributions is quite large. However, it does not contain some important heavy tail distributions, so it is natural to relax it by imposing milder condition on the moment generating function:

Definition

If a zero-mean random variable X satisfies (2.19) only on a neighborhood of zero, say for $|\lambda| < \frac{1}{\alpha}$, we call it sub-exponential with parameters σ and α, denoted as

$$X \sim \text{SubExp}(\sigma, \alpha)$$

If X is not zero-mean, the above means that $X - \mathbb{E}X$ satisfies the necessary condition.

As with sub-Gaussianity, one can yield concentration inequalities for sub-exponential variables as follows:

$$\begin{aligned} \Pr(X > t) &\leq \begin{cases} \exp(-\frac{t^2}{2\sigma^2}), & \text{if } 0 \leq t \leq \frac{\sigma^2}{\alpha} \\ \exp(-\frac{t}{2\alpha}), & \text{if } t \geq \frac{\sigma^2}{\alpha}. \end{cases} \\ &= \exp(-\min\{\frac{t^2}{2\sigma^2}, -\frac{t}{2\alpha}\}), \quad t \in [0, \infty) \end{aligned} \tag{2.24}$$

This means Sub-exponential random variables have Gaussian tails when the deviation t is relatively small comparing to $\frac{\sigma^2}{\alpha}$,and have exponential-tail when t is relatively large.

Examples

- Any sub-Gaussian variable is also sub-exponential with parameter σ and $\alpha = 0$, where we treat $1/0$ the same is $+\infty$.
- Let $X \sim N(0, 1)$; then $\mathbb{E}e^{\lambda(X^2-1)} = e^{-\lambda}(1 - 2\lambda)^{-\frac{1}{2}}$ for $\lambda \in (-\infty, \frac{1}{2})$ and ∞ otherwise. With some algebra, one can show that

  $$e^{-\lambda}(1 - 2\lambda)^{-\frac{1}{2}} \leq e^{2\lambda^2} = e^{4\lambda^2/2} \quad \forall |\lambda| < 1/4,$$

  Thus $X \sim \text{SubExp}(2, 4)$

Properties

The following are equivalent alternative characterizations of sub-Exponentials. Let K_i be constants that differ from each other by a constant factor, for $i = 1, \ldots, 5$.

1. Tail bound: $\forall t \geq 0, \quad \mathbb{P}(|X| \geq t) \leq 2\exp\left(-\frac{t}{K_1}\right)$

2. Moments: $\forall p \geq 1, \quad \|X\|_p \coloneqq \mathbb{E}[|X|^p]^{1/p} \leq K_2 p$

3. MGF of $|X|$ satisfies: $\forall \lambda$ such that $0 \leq \lambda \leq \frac{1}{K_3}, \quad \mathbb{E}[\exp(\lambda|X|)] \leq \exp(K_3\lambda)$

4. The MGF of $|X|$ is bounded at some point: $\mathbb{E}\left[\exp\left(\frac{|X|}{K_4}\right)\right] \leq 2$
   Alternatively, $\exists c$ such that $\mathbb{E}[\exp(c|X|)] \leq \infty$.

If $\mathbb{E}[X] = 0$, then properties (1) through (4) are also equivalent to the following 5th property:

5. MGF of X: $\forall|\lambda| \leq \frac{1}{K_5}, \quad \mathbb{E}[\exp(\lambda X)] \leq \exp(K_5^2\lambda^2)$

Sub-exponential Norm

The sub-Gaussian norm is defined as follows:

$$\|X\|_{\psi_1} = \inf\left\{t > 0 : \mathbb{E}\left[\exp\left(\frac{|X|}{t}\right)\right] \leq 2\right\} \tag{2.25}$$

$$= \sup_{p \geq 1} p^{-1}(\mathbb{E}|X|^p)^{1/p} \tag{2.26}$$

Lemma 3. *(Sub-exponential is sub-gaussian squared) A random variable X is sub-gaussian if and only if X^2 is sub-exponential. Moreover,*

$$\|X^2\|_{\Psi_1} = \|X\|_{\Psi_2}^2 \tag{2.27}$$

Lemma 4. *(Product of sub-gaussian is sub-exponential) Let X and Y be sub-gaussian random variables. Then XY is sub-exponential. Moreover,*

$$\|XY\|_{\Psi_1} \leq \|X\|_{\Psi_2}\|Y\|_{\Psi_2} \tag{2.28}$$

The following lemma is a Bernstein-type inequality and shows that the sum of independent sub-exponential random variables is sub-exponential.

Lemma 5. *Let $X_i \sim SubExp(\sigma_i, \alpha)$ $i = 1, \ldots, n$. Then,*

$$\Pr(|\sum_i (X_i - \mathbb{E}X_i)| > t) \leq 2\exp(-\min\{\frac{t^2}{2\sum_i \sigma_i^2}, \frac{t}{2\alpha}\}), \quad t \in [0, \infty) \tag{2.29}$$

Gaussian Concentration

Now we provide the result on the concentration properties of Lipschitz functions of Gaussian variables. First, we give a brief primer on Lipschitz functions.

Definition: (Lipschitz functions)

Let (X, d_X) and (Y, d_Y) be metric spaces. A function $f : X \to Y$ is called *Lipschitz* if there exists a real constant $L \geq 0$ such that

$$d_Y(f(u), f(v)) \leq L\, d_X(u, v) \quad \text{for all } \; u, v \in X \tag{2.30}$$

Observe that the constant L does not depend on u, v. The infimum over all such L is called the Lipschitz norm of f denoted $\|f\|_{\text{Lip}}$ and f is said to be L-Lipschitz. We mostly work with functions of the form $f : \mathbb{R}^n \to \mathbb{R}$, whereby

$$|f(u) - f(v)| \leq L\|u - v\|_2 \quad \text{for all } \; u, v \in \mathbb{R}^n \tag{2.31}$$

Lemma 6. ***Gaussian Concentration:*** *If $(X_1, \ldots, X_n)$ be a vector of i.i.d. standard Gaussian variables, and $f : (\mathbb{R}^n, \|.\|_2) \to (\mathbb{R}, |.|)$ is a L-Lipschitz continuous function. Then $f - \mathbb{E}f$ is sub-Gaussian with parameter $\sigma = L$, i.e.,*

$$\mathbb{P}(|f(X) - \mathbb{E}f(X)| \geq t) \leq 2\exp(-\frac{t^2}{2L^2}), \quad \forall t \geq 0 \tag{2.32}$$

Alternatively,

$$\|f(X) - \mathbb{E}f(X)\|_{\Psi_2} \leq \|f\|_{Lip} \tag{2.33}$$

Note that Lemma 6 guarantees that any L-Lipschitz function of a standard Gaussian random vector, regardless of the dimension, concentrates like a scalar Gaussian variable with variance L^2.

Example (Singular values of Gaussian random matrices):

Let $X \in \mathbb{R}^{n\times d}$ be a random matrix with i.i.d $N(0, 1)$ entries and let $\sigma_1(X) \geq \sigma_2(X) \geq \ldots \sigma_d(X) \geq 0$ denote its ordered singular values. By Weyl's theorem, given matrix $Y \in \mathbb{R}^{n\times d}$,

we can write

$$\max_{k=1,\dots,d} |\sigma_k(X) - \sigma_k(Y)| \leq \|X - Y\|_2 \leq \|X - Y\|_{\mathrm{F}}$$

This inequality shows that each singular values $\sigma_k(X)$ is a 1-Lipschitz function of random matrix, so using Lemma 6, we have:

$$\mathbb{P}[|\sigma_k(X) - \mathbb{E}[\sigma_k(X)]| \geq \delta] \leq 2e^{-\frac{\sigma^2}{2}} \quad \forall \delta \geq 0$$

Concentration of random vectors and matrices

It is possible to extend the result of (2.18) to random vectors and random matrices. In such cases, one is usually interested in bounding the deviation $\|X - \mathbb{E}X\|$ for some appropriate norm $\|.\|$ For example, consider a random vector $X \in \mathbb{R}^d$ with sub-Gaussian entries, $X_i \sim$ SubGauss(σ_i). Let $\sigma_\infty := \|(\sigma)_i\|_\infty$. Then, a simple application of union bound combined with the above exponential bounds yields

$$\Pr(\|X - \mathbb{E}X\|_\infty > t) \leq 2d \exp\Big(-\frac{t^2}{2\sigma_\infty}\Big) \tag{2.34}$$

By the equivalence of norms on finite-dimensional spaces, this inequality can be translated to deviations in other norms.

In many applications, it is common that a random matrix can be expressed as a sum of independent random matrices. As an example, the covariance of $X \in \mathbb{R}^{n\times d}$ can be written as $X^T X = \sum_{i=1}^n x_i^T x_i$, where x_i denotes i-th row of X. The matrix extensions of the two famous bounds, Bernstein inequality and Chernoff bound, are two frequently used approaches in these cases. In this section, we only state the Bernstein Inequality.

Concentration of a general matrix

As noted in section 2.1, we can use *dilation* to extend results for hermitian matrices to rectangular matrices. As dilation preserves spectral information, we can apply the matrix concentration inequalities to the Hermitian dilation of the random matrix. That is,

$$\Pr(\|X - \mathbb{E}X\| \geq t) = \Pr(\lambda_{\max}(\mathcal{D}(X) - \mathcal{D}(\mathbb{E}X)) \geq t) \tag{2.35}$$

Lemma 7 (Matrix Bernstein Inequality). *Let* $\{X_1, \ldots, X_n\}$ *be zero-mean, independent* $d_1 \times d_2$ *random matrices with* $\|X_i\|_2 \leq L$.

Define variance parameter as $\sigma^2 = \max\{\|\sum_{i=1}^n \mathbb{E}(X_i X_i^T)\|, \|\sum_{i=1}^n \mathbb{E}(X_i^T X_i)\|\}$, *then*

$$\Pr(|\sum_i X_i)| > t) \leq (d_1 + d_2) \exp\Big(\frac{-t^2}{\sigma^2 + \frac{2}{3}Lt}\Big) \tag{2.36}$$

Concentration of a Chaos

Let $A = (a_{ij})$ be an $n \times n$ matrix of coefficients, and $X = (X_1, \ldots, X_n)$ a random vector with independent coordinates. The following quadratic form is called a chaos.

$$\sum_{i,j=1}^n a_{ij} X_i X_j = X^T A X = \langle AX, X \rangle$$

We will now prove a general concentration inequality for a chaos. It can be viewed as a chaos version of Berstein's inequalty.

Lemma 8 (Hanson-Wright Inequality:). *Let* $X = (X_1, \ldots, X_n) \in \mathbb{R}^n$ *be a random vector with independent, mean zero, sub-gaussian coordinates. Let* A *be a* $n \times n$ *matrix. Then, for every* $t \geq 0$, *we have*

$$\Pr|X^T A X - \mathbb{E}X^T A X| \geq t \leq 2 \exp\Big[-c \min\Big(\frac{t^2}{K^4 \|A\|_F^2}, \frac{t}{K^2 \|A\|}\Big)\Big] \tag{2.37}$$

, *where* $K = \max_i \|X_i\|_{\Psi_2}$

The interested reader is referred to [37] for more materials on the subject and thorough discussions.

2.3 Kernels and feature maps

A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a *kernel* if

- k is symmetric: $k(x, y) = k(y, x)$.
- k is a positive semidefinite function. That is, it gives rise to a positive semi-definite ''Gram matrix". The following is a more precise definition of a PSD kernel function:

Definition: Positive semidefinite kernel function

A symmetric bivariate function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive semidefinite (PSD) if for all integers $n \geq 1$ and elements $\{x_i\}_{i=1}^n \in \mathcal{X}$ matrix with elements $K_{ij} := k(x_i, x_j)$ is positive semidefinite. That is,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j k(x_i, x_j) = a^T K a \geq 0. \tag{2.38}$$

In many machine learning algorithms, one needs to map the data to a higher dimensional space via a user-specified feature map so that the data could become more easily separated or better structured. However, if the transformed data points appear in an inner product form, the inner product can be replaced with kernel functions. That is, given a Hilbert space H(a complete inner product space) and a feature map $\Phi : \mathcal{X} \to H$,

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle \quad \forall x, y \in \mathcal{X} \tag{2.39}$$

This approach is referred to as *kernel trick*. In practice, the kernel k is usually defined directly without explicitly defining Hilbert space H and the map Φ. In other words, to compute the inner product $\langle \Phi(x), \Phi(y) \rangle$ in H, one does not need to know the map Φ, the identity (2.39) allows one to compute $k(x, y)$ instead. This is computationally cheaper than the explicit computation of the coordinates. It is even more desirable when our feature space is infinite-dimensional and thus infeasible to compute explicitly.

By Mercer's theorem, the necessary and sufficient condition so that identity (2.39) holds is that k be a *positive-semidefinite kernel*.

2.3.1 Properties of Kernels

Let k_1 and k_2 be two valid kernels. Than the following are also valid kernels:

1. $k(x, y) = \alpha k_1(x, y) + \beta k_2(x, y) \quad \text{for} \;\; \alpha, \beta \geq 0.$
2. $k(x, y) = k_1(x, y) k_2(x, y)$

3. $k(x, y) = k_1(h(x), h(y))$, where $h : \mathcal{X} \to \mathcal{X}$.

4. $k(x, y) = g(x)g(y)$ for $g : \mathcal{X} \to R$.

5. $k(x, y) = h(k_1(x, y))$ where h is a polynomial with positive coefficents.

6. $k(x, y) = \frac{k_1(x,y)}{\sqrt{k_1(x,x)}\sqrt{k_1(y,y)}}$

7. $k(x, y) = \exp(k_1(x, y))$

8. $k(x, y) = x^T Ay$, where A is a PSD matrix.

Example: Linear kernel

When $\mathcal{X} = \mathbb{R}^d$, we can define the linear kernel function $k(x, y) = \langle x, y \rangle$. This is a symmetric function. To prove the positive semi-definiteness, let $\{x_i\}_{i=1}^n$ be an arbitrary collection of points in $\mathbb{R}^d$, and the matrix $K \in \mathbb{R}^{n \times n}$ with entries $K_{ij} = \langle x_i, x_j \rangle$. For any vector $\alpha \in \mathbb{R}^n$:

$$\alpha^T K \alpha = \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle x_i, x_j \rangle = \| \sum_{i=1}^{n} \alpha_i x_i \|_2^2 \geq 0. \tag{2.40}$$

Thus K is PSD and as a result the linear kernel is a valid kernel.

Example: Gaussian kernel

Given some compact subset $\mathcal{X} \in R^d$, the gaussian kernel is defined as $k(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$. This is a symmetric function. To prove the positive semide-finiteness, we can factorize $k(x, y)$ as follows:

$$\begin{aligned} k(x, y) &= \exp\left(-\tfrac{1}{2\sigma^2}\|x - y\|^2\right) \\ &= \left(\exp\left(-\tfrac{1}{2\sigma^2}\|x\|^2\right) \exp\left(-\tfrac{1}{2\sigma^2}\|y\|^2\right)\right) \exp\left(\tfrac{1}{\sigma^2}\langle x, y \rangle\right). \end{aligned}$$

$g(x)g(y)$ is a kernel according to property(4) and $\exp(k_1(x, y))$ is a kernel according to property(7). According to property(2), the product of two kernels is a kernel. Thus, the gaussian kernel is a valid kernel.

Stationary Kernels

A stationary kernel is one which is translation invariant, that is:

$$k(x, y) = \tilde{k}(x - y) \tag{2.41}$$

Such a kernel is often referred to as *anisotropic stationary kernel*, in order to emphasize the dependence on both the direction and the length of the lag vector. When the stationary kernel is a function of distance, it will only depends on the norm of the lag vector and not the direction. In that case, it is referred to as *isotropic or homogeneous or distance kernel* and is denoted as:

$$k(x, y) = \tilde{k}(\|x - y\|) \tag{2.42}$$

By Bochner's theorem, a continuous stationary function $\tilde{k}$ is positive definite if and only if it is the Fourier transform of a positive finite Borel measure.

$$\tilde{k}(x - y) = \int_{\mathbb{R}^d} e^{-i\omega^T(x-y)} d\mu(\omega) \tag{2.43}$$

$$= \int_{\mathbb{R}^d} \cos\Big(\omega^T(x - y)\Big) d\mu(\omega), \tag{2.44}$$

where the measure $p = \mu / \int d\mu(\omega)$ is a probability measure and the second equality follows from the fact that kernels are symmetric.

CHAPTER 3

Matched Bipartite Block Model with covariates

3.1 Introduction

Motivated by the matching problem, in this chapter, 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 3.2). 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. We will propose a generative model for such networks where there is a hidden matched community structure. This avoids the need for post-hoc matching of the communities: the matching is built into the model and inferred simultaneously 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 details in Section 3.2. We also discuss an extension to allow for degree-correction (Section 3.3.4), providing a matched version of degree-corrected block model (DC-SBM).

In another direction, many networks come with metadata, often in the form of node features, or 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 [35, 34, 33, 32]. Bipartite setting adds another challenge to modeling node covariates, in particular, how to jointly model the covariates from the two sides, considering that one often has covariates of different dimensions on each side. (The extreme case is when only once 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. In other words, covariates

corresponding to nodes in matched communities are statistically linked. The linkage can be tunned using a general cross-covariance matrix, allowing for varying degrees of covariate influence on the community detection problem (Section 3.2).

To fit the model, we derive an algorithm based on variational inference, also known as variational Bayes [38, 39] ideas (Section 3.3). 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, first proposed in [22], and identify it as an effective algorithm for matched bipartite clustering. We show the effectiveness of our approach on simulated (Section 3.4) and real data (Section 3.5), namely, page-user networks collected from Wikipedia (Section 3.5).

To summarize, our contributions in this chapter 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 [22], 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, which we refer to as *matched bipartite stochastic block model* (`mbiSBM`), has a latent structure of matched communities and allows for node covariates that are potentially informative about the matching (see Section 3.2). 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 [40].

(iii) Show the effectiveness of the variational Bayes approach in fitting the overall `mbiSBM` model, when combined with good initialization, especially a variant of `biSC` algorithm of [22]. The algorithm is a block-coordinate ascent with a closed-form, fairly cheap iterations, and can be scaled to large networks.

Notation. We write $[K] := \{1, \ldots, K\}$ 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 $\{0,1\}^K \cap \mathcal{P}_K$. $x \mapsto N(x; \mu, \Sigma)$ denotes the PDF of a multivariate Gaussian distribution with mean μ and covariance Σ. 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.

3.2 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 chapter 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 $[N_1] = \{1, \ldots, N_1\}$ and $[N_2] = \{1, \ldots, N_2\}$, 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:

$$C_{1k} \leftrightarrow C_{2k}, \quad k = 1, \ldots, K. \tag{3.1}$$

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 $x_{rij}, j = 1, \ldots, d_r$. 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.

3.2.1 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 3.1. In order to model the correlation (i.e., a

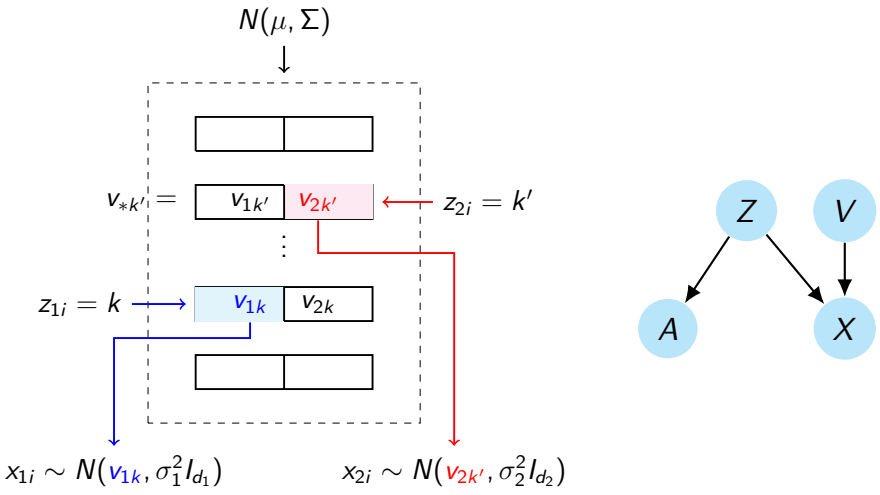


Figure 3.1: (Left) Schematic diagram for the hierarchical generation of node covariates. (Right) Overall graphical representation of the model.

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:

$$\begin{aligned} z_{ri} &\sim \text{Mult}(1, \pi_r), \\ (v_{rk}, r = 1, 2) &\overset{\text{iid}}{\sim} N(\mu, \Sigma), \quad k = 1, \ldots, K. \\ x_{ri} | z_{ri} = k, v_{rk} &\sim N(v_{rk}, \sigma_r^2 I_{d_r}), \quad i \in [N_r], \ r = 1, 2 \end{aligned} \tag{3.2}$$

where the draws are independent over r and $i \in [N_r]$, on each line. Here, $\pi_r = (\pi_{r1}, \ldots, \pi_{rK})$ 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. 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

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

Note that $\mu_r \in \mathbb{R}^{d_r}$. In subsection 3.2.5, 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.

3.2.2 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

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \begin{cases} \text{Ber}(p) & z_{1i} = z_{2j} \\ \text{Ber}(q) & z_{1i} \neq z_{2j} \end{cases}, \quad i \in [N_1],\ j \in [N_2]. \tag{3.3}$$

Combined, (3.2) and (3.3) 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 (3.3) in deriving the algorithms, it is possible to allow for a more general edge probability structure as in the usual SBM, by assuming

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Ber}(\Psi_{z_{1i}, z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{3.4}$$

where $\Psi \in [0,1]^{K \times K}$ is a connectivity (or edge probability) matrix. Model (3.3) 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 (3.2) 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. Thus, by rescaling Σ and $Q = (p, q)$, we can control the balance of these two sources of information. This is explored in Section 3.4 through simulation studies.

3.2.3 Connection with the usual SBM

Ignoring the covariate part of the model, one might wonder whether `mbiSBM`, introduced in (3.4), 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 (3.4), and let $Z_r = (z_{ri}) \in \{0,1\}^{N_r\times K}$ for $r = 1, 2$. We express the model compactly as

$$\mathbb{E}\underbrace{\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}}_{\widetilde{A}} = \underbrace{\begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix}}_{\widetilde{Z}} \underbrace{\begin{pmatrix} 0 & \Psi \\ \Psi^T & 0 \end{pmatrix}}_{\widetilde{\Psi}} \begin{pmatrix} Z_1^T & 0 \\ 0 & Z_2^T \end{pmatrix}. \tag{3.5}$$

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 (3.5), it is clear that model (3.4) 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 $\ell \in [K]$.

3.2.4 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 [41, 2]. It is fairly straightforward to introduce degree-correction in our setup. Consider the form of the matched SBM introduced

in (3.4). 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 $r = 0, 1$. The degree-corrected (DC) version of the model replaces (3.4) with

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Poi}(\theta_{1i}\theta_{2j}\Psi_{z_{1i}, z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{3.6}$$

Replacing the Bernoulli with Poisson is for convenience in later derivations, and is common in dealing with DC-SBM [2]. 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:

$$\frac{1}{|C_{rk}|} \sum_{i \in C_{rk}} \theta_{ri} = 1 \iff \sum_{i=1}^{N_r} (\theta_{ri} - 1) z_{rik} = 0, \quad k \in [K],\ r = 1, 2. \tag{3.7}$$

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 [18].

Remark 2. A normalization of the form (3.7) 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, (3.7) is inconsistent with i.i.d. random generation of z_{ri} from a $\text{Mult}(1, \pi_r)$ as in (3.2). One way to get around this is to assume that the labels are generated a priori from the product multinomial distribution described in (3.2) *conditioned* on the set of labels satisfying (3.7). We will ignore the change in the label prior this conditioning makes in deriving the algorithms. In the end, we enforce (3.7) in an "averaged" sense, replacing z_{rik} with the corresponding (approximate) posterior τ_{rik}, as detailed in subsection 3.3.4. Viewed as a set of constraints on the collection of soft-labels (τ_{rik}), (3.7) is not that restrictive.

3.2.5 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,

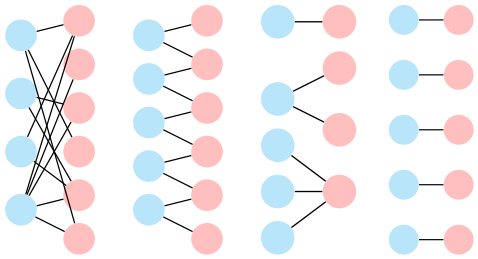


Figure 3.2: Possible relations between communities of the two sides, in a bipartite network. (a) and (b) are hard to interpret. Structures like (c), i.e., collections of disjoint stars, are interpretable and (d) is the simplest within this class.

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 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.

3.2.6 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 3.2 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 3.2(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 3.2(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.

3.3 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 3.3.4.

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

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

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

$$\ell_1(\Psi) =: \log p(A|Z) = \sum_{ij} \sum_{k\ell} z_{1ik} z_{2j\ell}\ g(\Psi_{k\ell}, A_{ij}) \tag{3.8}$$

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

$$\begin{aligned} \ell_1(\Psi) = \log p(A|Z) &= \sum_{ij} \Big[\sum_{k} z_{1ik} z_{2jk}\ g(p, A_{ij}) + \sum_{k \neq \ell} z_{1ik} z_{2j\ell}\ g(q, A_{ij}) \Big] \\ &= \sum_{ij} y_{ij} g(p, A_{ij}) + (1 - y_{ij}) g(q, A_{ij}). \end{aligned} \tag{3.9}$$

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 (3.8) and

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

3.3.2 Mean-field Approximation

Variational inference is often regarded as the approximation of a posterior distribution by solving an optimization problem [42, 39]. 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) [38],

involves both the likelihood parameters and a distribution q, namely,

$$J := \mathbb{E}_q[\ell(\mu, \Sigma, \sigma, \pi, \Psi) - \log q(Z, V)]. \tag{3.10}$$

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*:

$$q(Z, V) = q_V(V)q_Z(Z), \quad q_Z(Z) = \textstyle\prod_{r,i} q_{ri}(z_{ri}), \quad q_V(V) = \prod_{k=1}^K N(v_{*k}; \tilde{\mu}_k, \tilde{\Sigma}_k) \tag{3.11}$$

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 (3.11) 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 $\widetilde{\Gamma} := ((\widetilde{\Sigma}_k, \widetilde{\mu}_k), k = 1, \ldots, K)$, collecting the parameters for the variational posterior q_V. Plugging in the variational distribution (3.11) into the variational likelihood (3.10) using expression (3.9) for $l_1(\psi)$, after some algebra detailed in Appendix 3.A.1, we get

$$\begin{aligned} J = &\sum_{i,j} \Big[\gamma_{ij}(\tau) g(p; A_{ij}) + (1 - \gamma_{ij}(\tau)) g(q; A_{ij})\Big] + \sum_{r,i,k} \tau_{rik} \big[\beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \frac{\pi_{rk}}{\tau_{rik}}\big] \\ &- \frac{1}{2} \sum_r d_r N_r \log \sigma_r^2 - \frac{K}{2} \big\{ \log |\Sigma| + \operatorname{tr}[\Sigma^{-1} S(\widetilde{\Gamma}, \mu)]\big\} + \frac{1}{2} \sum_k \log |\widetilde{\Sigma}_k| + \text{const.} \end{aligned} \tag{3.12}$$

where

$$\gamma_{ij}(\tau) := \mathbb{E}_{q_Z}(y_{ij}) = \sum_{k=1}^K \tau_{1ik}\tau_{2jk}, \quad \beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2} \Big[\operatorname{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2 \Big], \tag{3.13}$$

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

$$S(\widetilde{\Gamma}, \mu) := \frac{1}{K} \sum_{k=1}^{K} \big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]. \tag{3.14}$$

3.3.3 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 3.A.3.

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,

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

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 9. *For any nonnegative vector $(a_1, \ldots, a_K)$, 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:*

$$\operatorname*{argmax}_{p \in \mathcal{P}_k} f_a(p) = \operatorname{softmax}(a) := \frac{e^{a_k}}{\sum_\ell e^{a_\ell}}. \tag{3.15}$$

Proof. 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. □

We write the solution of Lemma 9 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},

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big[\sum_j \tau_{2jk}\, h(p, q; A_{ij}) + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big] \quad i = 1, \ldots, N_1. \tag{3.16}$$

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 3.A.2)

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2} \sum_k \operatorname{tr}\big[(D_k^{-1} + \Sigma^{-1})\widetilde{\Sigma}_k\big] - \log|\widetilde{\Sigma}_k|. \tag{3.17}$$

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 3.A.2):

$$J(\widetilde{\mu}) \doteq -\frac{1}{2} \sum_k (\widetilde{\mu}_k - m_k)^T (D_k^{-1} + \Sigma^{-1})(\widetilde{\mu}_k - m_k) \tag{3.18}$$

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 $J(\Sigma) \doteq -\frac{K}{2}\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big]$ which is the standard Gaussian likelihood, giving the optimal value $\Sigma = S(\widetilde{\Gamma}, \mu)$. Similarly, as a function of μ, $J(\mu) \doteq -\frac{K}{2}\big[\mathrm{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big] \doteq -\frac{1}{2}\sum_k \big[(\widetilde{\mu}_k - \mu)^T\Sigma^{-1}(\widetilde{\mu}_k - \mu)\big]$ 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 3.A.3)

$$\sigma_r^2 = \frac{1}{N_r d_r}\Big[\sum_k \bar{\tau}_{rk}\,\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \sum_{i,k}\tau_{rik}\|x_{ri} - \widetilde{\mu}_{rk}\|^2\Big], \quad r = 1, 2. \tag{3.19}$$

Updating π and $\Psi \equiv (p, q)$. Updating these parameters is standard (See Appendix 3.A.3):

$$\pi_r = \frac{(\bar{\tau}_{r1}, \ldots, \bar{\tau}_{rK})}{\sum_k \bar{\tau}_{rk}},\ r = 1, 2, \quad p = \frac{\sum_{ij}\gamma_{ij}(\tau)A_{ij}}{\sum_{ij}\gamma_{ij}(\tau)}, \quad q = \frac{\sum_{ij}\big(1 - \gamma_{ij}(\tau)\big)A_{ij}}{\sum_{ij}\big(1 - \gamma_{ij}(\tau)\big)}. \tag{3.20}$$

3.3.3.1 Improving the speed

For $r = 0, 1$, we treat each $(\tau_{rik})_{ik}$ as a matrix $\tau_r \in [0, 1]^{N_r \times K}$. The τ-update in (3.16) 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 3.3.4 below, $\phi_0 = q - p$ and $\phi_1 = \log(p/q)$. Then, in matrix notation

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

where $[A\tau_2]_{ik} = \sum_j \tau_{2jk}A_{ij}$. 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:

$$\tau_1 = \texttt{row-softmax}\big[\phi_1 A\tau_2 + \phi_0\mathbf{1}_{N_1}(\bar{\tau}_2 + \log\pi_1)^T + \beta_1\big] \tag{3.21}$$

and similarly for τ_2. Here, `row-softmax` is the row-wise softmax operator, applying (3.15) 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

$$p = \frac{\mathrm{tr}(\tau_1^T A \tau_2)}{\langle \bar{\tau}_1, \bar{\tau}_2 \rangle}, \quad q = \frac{r\rho - p}{r-1}, \quad \text{where } \frac{1}{r} = \langle \frac{\bar{\tau}_1}{N_1}, \frac{\bar{\tau}_2}{N_2} \rangle \text{ and } \rho = \frac{1}{N_1 N_2} \sum_{ij} A_{ij}. \tag{3.22}$$

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 $\beta'_{rik} = \mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2$ noting that $\beta_{rik} = \frac{1}{2\sigma_r^2}\beta'_{rik}$ from definition (3.13). Letting $\beta'_r = (\beta'_{rik})_{ik}$ be its matrix form, we can write the update for σ_r^2 compactly as

$$\sigma_r^2 = \frac{1}{N_r d_r} \sum_{ik} \tau_{rik} \beta'_{rik} = \frac{1}{N_r d_r} \mathrm{tr}(\tau_r^T \beta'_r). \tag{3.23}$$

3.3.4 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

$$\ell_1(\Psi, \theta) = \sum_{ij} \left[y_{ij}\, h\big(p\theta_{1i}\theta_{2j},\, q\theta_{1i}\theta_{2j},\, A_{ij}\big) + g\big(\theta_{1i}\theta_{2j}q, A_{ij}\big) \right]. \tag{3.24}$$

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 (3.7), $\sum_i \theta_{ri} z_{rik} = \sum_i z_{rik}$, 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 (3.24) simplifies to

$$\begin{aligned} &\sum_{ij} y_{ij} \left[(q-p)\theta_{1i}\theta_{2j} + A_{ij} \log(p/q) \right] + \sum_{ij} \left[-\theta_{1i}\theta_{2j}q + A_{ij} \log(\theta_{1i}\theta_{2j}q) \right] \\ &\qquad = \sum_{ij} y_{ij} \left[(q-p) + A_{ij} \log(p/q) \right] - qN_1N_2 + \sum_{ij} A_{ij} \log(\theta_{1i}\theta_{2j}q) \end{aligned}$$

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 (3.7) under $Z \sim q$. Focusing on updating τ_{1k}, we have the following optimization problem:

$$\begin{array}{ll} \max\limits_{\tau_1} & \sum_{i,k} \tau_{1ik}\big(\sum_j \tau_{2jk}\big[\phi_0 + A_{ij}\phi_1\big] + \xi_{1ik} - \log \tau_{1ik}\big) \\ \text{subject to} & \sum_i \tau_{1ik}(\theta_{1i} - 1) = 0 \end{array} \tag{3.25}$$

where $\xi_{1ik} = \beta_{1ik} + \log \pi_{1k}$ as before. In Appendix 3.B.1, we derive a dual ascent algorithm for solving this problem with the following updates:

$$\begin{aligned} \tau_1(\lambda) &= \texttt{row-softmax}\big[\phi_1 A \tau_2 + \phi_0 \mathbf{1}_{N_1}(\bar{\tau}_2 + \log \pi_1)^T + \beta_1 + (\theta_1 - \mathbf{1})\lambda^T\big], \\ \lambda^+ &= \lambda - \mu[\tau_1(\lambda)]^T(\theta_1 - \mathbf{1}). \end{aligned} \tag{3.26}$$

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 (3.21) 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 [43, 44]. Let $f_t(\cdot; d) : \mathbb{R}^n_+ \to \mathbb{R}^n_+$ with $d \in \mathbb{R}^n_+$ and $t > 0$, be defined by

$$[f_t(x; d)]_i := \frac{1}{2}\big[x_i + \sqrt{x_i^2 + 4td_i}\big]. \tag{3.27}$$

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^+):

$$\begin{aligned}\theta_1^+ &= f_t(\xi_1; d_1)\\ \xi_1^+ &= \theta_1^+ - H_1(2\theta_1^+ - \xi_1 - \mathbf{1})\end{aligned} \tag{3.28}$$

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 3.B.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 [2]. 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 (3.20).

3.3.5 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 (3.26) in steps 11 and 12, for updating the label distributions (τ_r), automatically reduce to the simple update (3.21) (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.

3.3.5.1 Initialization of the algorithms

It is known that variational inference is sensitive to initialization [39]. 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 [22]. The difference between our version and that of [22] is that [22] 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 [9] as `SCP`. Note that `SCP` will be applied to the (symmetric) adjacency matrix $\widetilde{A}$; see (3.5). 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 $\sim$`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 3.5). 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 3.4).

3.4 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 (3.2)–(3.3). In the plots investigating the degree-corrected version of the algorithm, we generate from the degree-corrected version of the network described in subsection 3.2.4.

Data generation. Key parameters regarding covariate generation in (3.2) 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 3.D). 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 (3.3) are p and q. We reparametrize our planted partition model in terms of expected average degree

$$\lambda = \frac{2N_1N_2}{N_1 + N_2}\big[q + (p - q)\sum_k \pi_{1k}\pi_{2k}\big] \tag{3.29}$$

(see Appendix 3.C) 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 [45]. 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 (3.29) remains valid for the degree-corrected case without modification, assuming normalization (3.7); see Appendix 3.C.

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

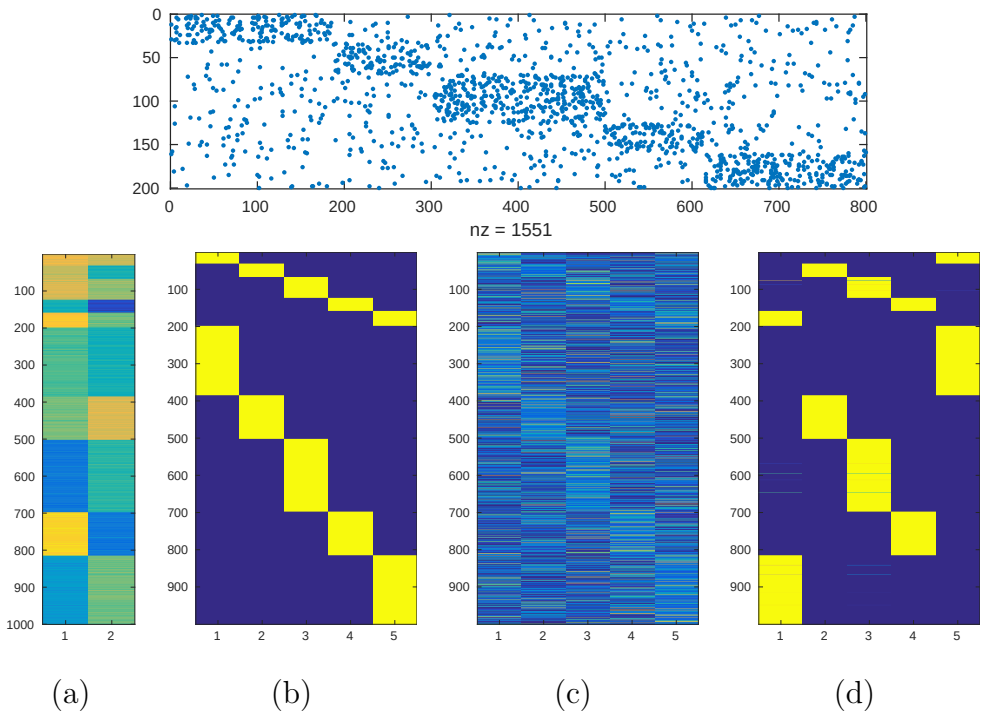


Figure 3.3: Typical output of the algorithm. Top row: bi-adjacency matrix. Bottom row: (a) Concatenated covariate matrix $[X_1; X_2]$. (b) Concatenated true labels $[Z_1; Z_2]$. (c) Initial labels for the algorithm, Dirichlet-perturbed truth $0.1[Z_1; Z_2] + 0.9\,\mathrm{Dir}(0.5\mathbf{1}_K)$. (d) Concatenated output of the algorithm $[\widehat{\tau}_1; \widehat{\tau}_2]$.

which is defined as the mutual information of the (empirical) joint distribution of the two label assignments divided by the joint entropy [46]. NMI has a maximum value of 1 for perfect agreement and a minimum of 0 for no agreement. One could measure NMI individually between $Z_r \in \{0,1\}^{N_r \times K}$ (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.

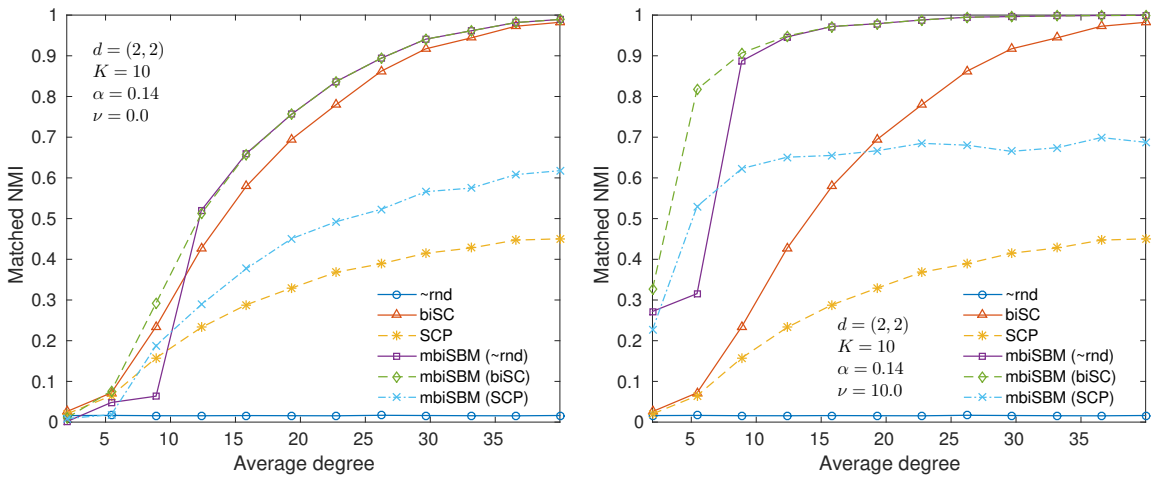


Figure 3.4: Effect of different initialization methods on `mbiSBM` with (left) $\nu = 0$ and (right) $\nu = 10$. The case $\nu = 0$ corresponds to no covariate information whereas $\nu > 0$ gives some covariates information.

Typical output. Figure 3.3 shows the typical output of the algorithm on the data generated from the model without degree correction (DC), i.e., $a = \infty$. Here the 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 3.3.5.1. It is interesting to note that the output of the algorithm has recovered the communities with a nontrivial permutation of the community labels.

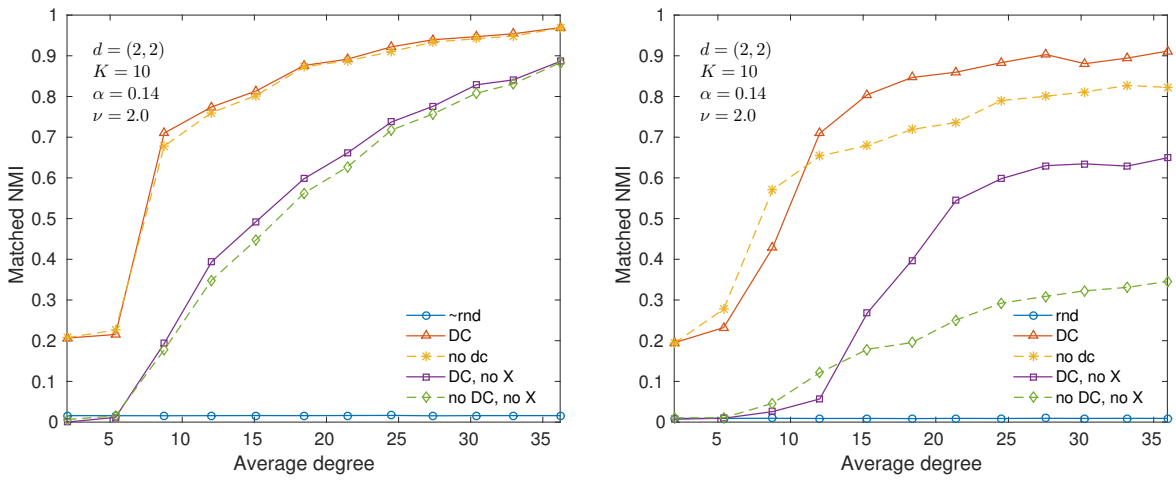
In other words, the perturbation of the initial labels is high enough that the convergence of the algorithm cannot simply be explained by a local perturbation analysis: the algorithm has not converged to the original labels, but to a perfectly valid permuted version of the original labels. That is, $\widehat{\tau}_r \approx Z_r Q_r$ 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 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.

Average behavior. Figure 3.4 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 3.3.5.1. 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.

Effect of degree correction. Figure 3.5 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).

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. Fig-



(a) (b)

Figure 3.5: 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 3.2.4) 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 ∼`rnd` (with $\approx 10\%$ true labels) and the (right) shows the results for a completely random initialization, denoted as `rnd`.

ure 3.6, illustrates the algorithm with DC correction and without, in the same setup of the left panel of Figure 3.5, that is, both cases initialized with ∼`rnd` (and both incorporating covariates). Though Figure 3.5(a) shows that mean behaviors are close, Figure 3.6 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.

3.5 Application to real data

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

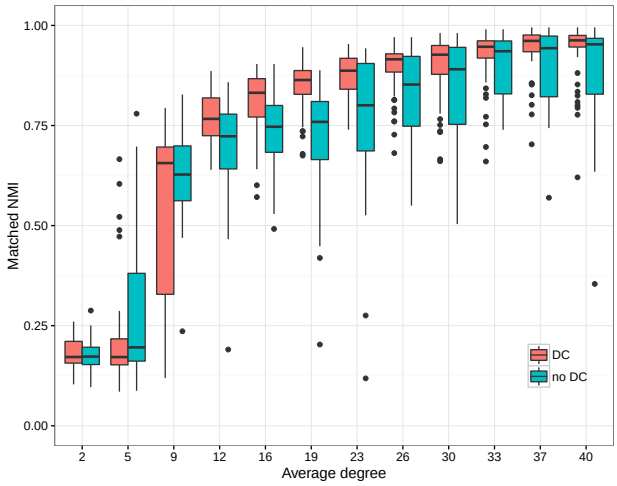


Figure 3.6: Effect of degree correction steps on variability, for a power-law network.

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 [47] using code inspired by [48]. For geo-data we used both the `ggmap` R package [49] and the API provided by [50].

In TOPARTICLES network, 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 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 $X_2 \in \mathbb{R}^{N_2 \times 2}$,

The page language was assigned differently for the two networks. For TOPARTICLES, it is the language in which the page was written. For CITIES, it is the country to which the city

	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 3.1: 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 3.2: Cities page–user network

(that the page is about) belongs. 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 3.7 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 3.1 and 3.2 show the break-down of pages/users based on community (i.e., language) 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 3.3 illustrates the result of the application of `biSC`, and the `mbiSBM` (`biSC`) algorithm with various combination of covariates. In all cases

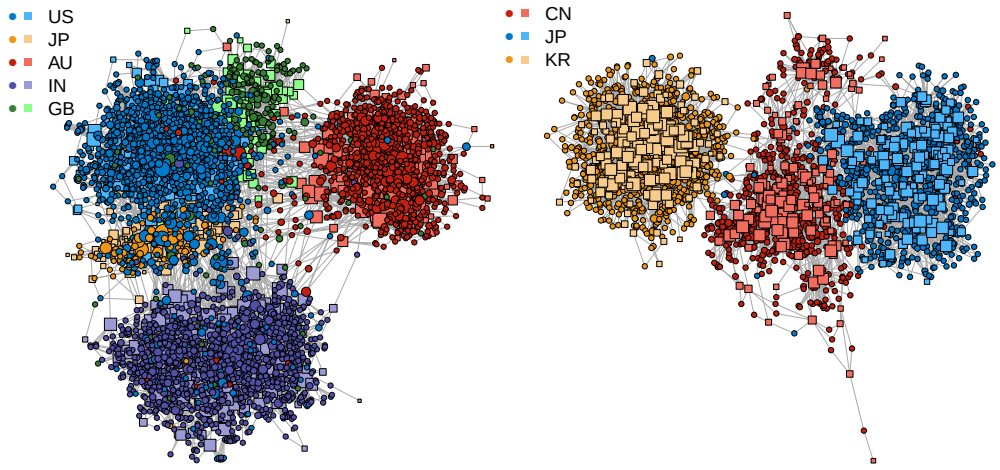


Figure 3.7: The two Wikipedia networks: (left) CITIES (right) TOPARTICLES. Nodes are colored according to true communities (assigned languages). Pages are denoted with squares and users with circles. Node sizes are proportional to log-degrees.

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

the degree-corrected (DC) version of `mbiSBM` is used. For CITIES, 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`.

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 3.8 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

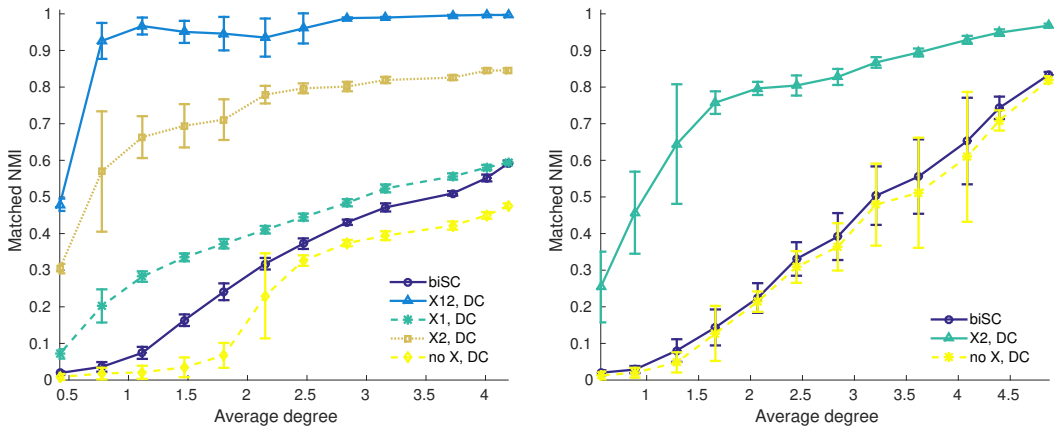


Figure 3.8: Effect of subsampling on Wikipedia networks: (left) Cities (right) TopArticles.

results are averaged over 50 replications and the interquartile range (IQR) is also shown as a measure of variability. The figures in Table 3.3 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 3.8(b), the NMI starts at 0.98 and remains much above 0.42 for most of the range of subsampling illustrates the ability of `mbiSBM` to

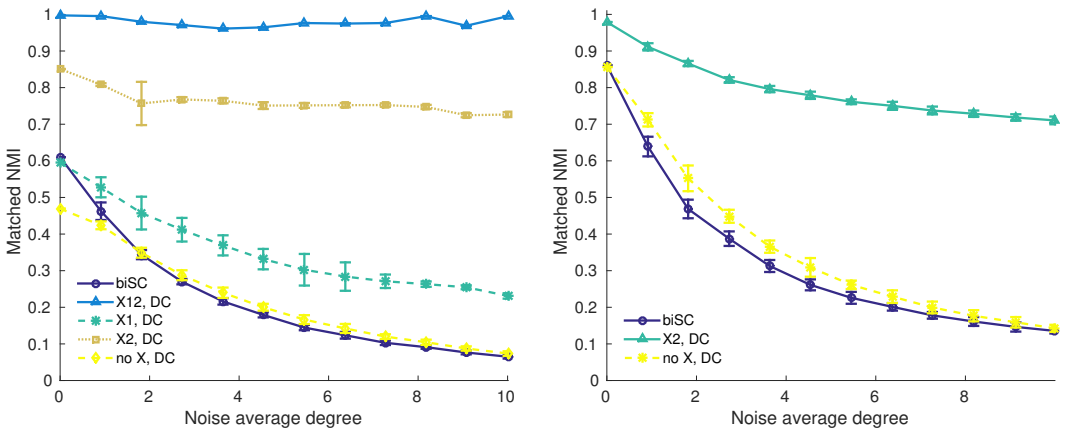


Figure 3.9: Effect of adding Erdös–Rényi noise on Wikipedia networks: (left) Cities (right) TopArticles.

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 3.9 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`.

Algorithm 1 Variational block coordinate ascent for fitting `mbiSBM`

Initialize τ_r using `biSC`, and $\theta_r = \mathbf{1}_{N_r}$ for $r = 1, 2$. Pick tolerance $\varepsilon \in (0, 1]$.

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 $r = 1, 2$.

while not `CONVERGED`, nor maximum iterations reached **do**

  Update (p, q) using (3.22) and $\pi_r, r = 1, 2$ using (3.20).

  Update $(\phi_0, \phi_1) \leftarrow (q - p, \log(p/q))$.

  if DC-version **then**

    Update θ_r by repeating (3.28) till convergence.

  end if

  Update $\beta_r, r = 1, 2$ using (3.13).

  $\tau_r^{\text{old}} \leftarrow \tau_r$, $r = 1, 2$.

  Update τ_1 by repeating (3.26) till convergence.

  Update τ_2 by repeating (3.26), with subscripts 1 and 2 switched and A replaced with A^T, till convergence.

  Update the following for for $r = 1, 2$ and $k \in [K]$: ▷ Update parameters

  $\bar{\tau}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik}$, and $D_k^{-1} \leftarrow \operatorname{diag}\left(\bar{\tau}_{1k} I_{d_1}/\sigma_1^2, \bar{\tau}_{2k} I_{d_2}/\sigma_2^2\right)$,

  $\bar{x}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$, and $\bar{\mu}_{rk} \leftarrow \bar{x}_{rk}/\bar{\tau}_{rk}$.

  Update $\widetilde{\Sigma}_k \leftarrow (D_k^{-1} + \Sigma^{-1})^{-1}$ and $\widetilde{\mu}_k \leftarrow \widetilde{\Sigma}_k(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$.

  Update $\mu \leftarrow \frac{1}{K}\sum_k \widetilde{\mu}_k$ and $\Sigma \leftarrow \frac{1}{K}\sum_{k=1}^K \left[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\right]$.

  Update $\sigma_r^2, r = 1, 2$ using (3.23).

  `CONVERGED` $\leftarrow \left[\max\{\delta_1, \delta_2\} < \varepsilon/K\right]$, where $\delta_r := \|\tau_r - \tau_r^{\text{old}}\|_\infty, r = 1, 2$

end while

Algorithm 2 Bipartite Spectral Clustering (`biSC`)

1: Input: bi-adjacency matrix $A \in \{0,1\}^{N_1 \times N_2}$.

2: Let $D_1 = \text{diag}(\sum_j A_{ij}, i = 1, \ldots, N_1)$ and $D_2 = \text{diag}(\sum_i A_{ij}, j = 1, \ldots, N_2)$.

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 ℓ_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.

APPENDIX

3.A Details of Section 3.3

3.A.1 Derivation of (3.12)

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 (3.11) into the variational likelihood (3.10). 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

$$T_1 = \sum_{i,j}\Big[\gamma_{ij}(\tau)\log(p^{A_{ij}}(1-p)^{1-A_{ij}}) + (1-\gamma_{ij}(\tau))\log(q^{A_{ij}}(1-q)^{1-A_{ij}})\Big] \tag{3.30}$$

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

$$\mathbb{E}[\varepsilon^T\Lambda\varepsilon] = \mathrm{tr}[\Lambda\widetilde{\Sigma}] + \widetilde{\mu}^T\Lambda\widetilde{\mu}, \tag{3.31}$$
$$\mathbb{E}[\log N(x; \varepsilon, \Sigma)] = \mathbb{E}[\log N(\varepsilon; x, \Sigma)] = -\frac{1}{2}\Big\{\log|\Sigma| + (x-\widetilde{\mu})^T\Sigma^{-1}(x-\widetilde{\mu}) + \mathrm{tr}(\Sigma^{-1}\widetilde{\Sigma})\Big\}$$
$$= -\frac{1}{2}\Big\{\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}\Psi)\Big\}, \tag{3.32}$$

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

Proof. Let us prove (3.32). 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 (3.31) 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 10, 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 10 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 10 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

$$T_4 = \mathbb{E}_{q_Z}\log q(Z) = \mathbb{E}_{q_Z}\sum_{r,i,k} z_{rik}\log\tau_{rik} = \sum_{r,i,k}\tau_{rik}\log\tau_{rik}. \tag{3.33}$$

Putting the pieces together we get expression (3.12).

3.A.2 Updates of $\widetilde{\Sigma}$ and $\widetilde{\mu}$

From (3.12), 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 $\beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2}[\operatorname{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2]$, and $S(\widetilde{\Gamma}, \mu) := \frac{1}{K}\sum_{k=1}^K \left[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\right]$ from their definitions, and looking at the result only as a function of $\widetilde{\Sigma}$, we obtain

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2}\sum_k \Big[\sum_{r,i} \tau_{rik}\frac{\operatorname{tr}\left((\widetilde{\Sigma}_k)_{rr}\right)}{\sigma_r^2} + \operatorname{tr}[\Sigma^{-1}\widetilde{\Sigma}_k] - \log|\widetilde{\Sigma}_k|\Big] \tag{3.34}$$

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

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

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 (3.18) follows by applying the following lemma.

Lemma 11 (Sum of quadratic forms)**.** *For symmetric matrices $Q_1, Q_2, \ldots,$*

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

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. □

3.A.3 Updates of σ^2, π, p and q

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

$$J((\sigma_r^2)) = -\frac{1}{2}\sum_r \Big[\frac{1}{\sigma_r^2}\sum_{i,k} \tau_{rik}\big[\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2\big] + d_r N_r \log \sigma_r^2\Big]. \tag{3.35}$$

The maximizer of the function $x \mapsto Ax^{-1} + B\log x$ is A/B (assuming $A, B > 0$), from which (3.19) 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 12. *For any nonnegative vector $(a_1, \ldots, a_K)$,*

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

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

3.B Details for degree-corrected algorithm

3.B.1 τ-update with degree restriction

In this section, we derive a dual ascent algorithm for the optimization problem (3.25) 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 (3.25) can be stated as

$$\min_{X=(x_{ik})} -\sum_{ik} x_{ik}(a_{ik} - \log x_{ik}), \quad \text{s.t. } X \in \mathcal{P}_{n,K}, \ \sum_i x_{ik}(\theta_i - 1) = 0, \ \forall k \tag{3.36}$$

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 (3.36), 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,

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

Solving for $X^*(\lambda)$ is an instance of the problem in Lemma 9. 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

$$x^*_{ik}(\lambda) \propto_k \exp(a_{ik} + \lambda_k(\theta_i - 1)). \tag{3.37}$$

Thus, the update for the dual descent can be written

$$\lambda_k^+ = \lambda_k - \mu\sum_i x^*_{ik}(\lambda)(\theta_i - 1), \quad \forall k. \tag{3.38}$$

3.B.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

$$\begin{aligned} \theta^+ &= \text{prox}_{th}(\xi) \\ \xi^+ &= \xi + P_V(2\theta^+ - \xi) - \theta^+ \end{aligned} \tag{3.39}$$

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 $[\text{prox}_{th}(\theta)]_i = \text{prox}_{td_i \log(\cdot)}(\theta_i)$. This univariate proximal operator can be easily shown to coincide with $[f_t(\theta, d)]_i$ as given in (3.27).

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 (3.39) simplifies to

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

which gives the claimed update.

3.C 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 (3.1). 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 (3.7): $\sum_{j\in C_{2\ell}} \theta_{2j} = |C_{2\ell}|$ 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 $\frac{2N_1N_2}{N_1+N_2} = 2/(N_1^{-1} + N_2^{-1})$ is the harmonic mean of N_1 and N_2.

3.D More simulations

Figure 3.D.1 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 3.4, 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.

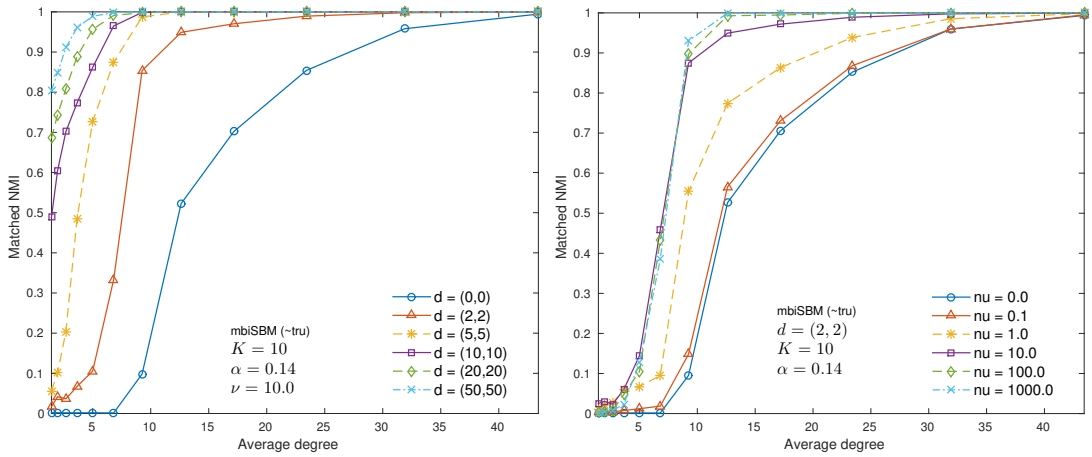


Figure 3.D.1: (Left) Effect of changing the dimension of covariates $d = (d_1, d_2)$ and (right) the parameter ν where $\Sigma = \nu I_{d_1+d_2}$.

CHAPTER 4

Unified Similarity

4.1 Introduction

In Chapter 3, we proposed a model-based approach (`mbiSBM`) to utilizing the information from both the bipartite network as well as the covariates. However, `mbiSBM` has some restrictive assumptions, that is, the normality and independence requirement for the covariates. In this chapter, we address these issues by proposing a new approach which we call *unified similarity clustering* (`USim`) that combines the information from the network with that from the node covariates into a single similarity matrix, which can then be input to a spectral clustering algorithm. `USim` can be used in both the unipartite as well as the bipartite settings.

The general idea is to use a kernel function to transform the node covariate information into a similarity matrix. Consider the unipartite case for example, where we have n nodes, and let $\{x_i, i = 1, \ldots, n\} \subset \mathbb{R}^d$ be the d-dimensional node covariate vectors. Given a suitably chosen kernel function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined on the covariate space ($\mathbb{R}^d$), we form the *kernel matrix* $K = (k(x_i, x_i)) \in \mathbb{R}^{n\times n}$, treating its (i, j) entry as a measure of similarity between covariates of nodes i and j . This approach to generating a similarity matrix is very versatile and commonly used in machine learning. It is well-known that by a proper choice of the kernel, one can often achieve good clustering of the data points in the so-called *feature space* even if they are not quite clustered in the original space.

Once we have the kernel matrix $K \in \mathbb{R}^{n\times n}$, we will treat it as a weighted adjacency matrix on the n nodes. On the other hand, we have the original network information which, in the unipartite case, comes as a (weighted) adjacency matrix $A \in [0, 1]^{n\times n}$. The problem is now reduced to combining two adjacency (or similarity) matrices. A simple approach is

to form a weighted average of the two: That is, given a tuning parameter $\tau > 0$, form the following similarity matrix:

$$W_\tau = W_\tau(A, K) := \tau A + K. \tag{4.1}$$

By varying τ one can tune in the contributions from each source of the information. This method of combining the two sources has been proposed, for example in [35], for the special case of the inner-product kernel $K = (\langle x_i, x_j \rangle)$. In the sequel, we will refer to (4.1) as the *average similarity*. The main drawback of this approach is that it does not naturally extend to the bipartite (or multi-partite) settings. The `USim` algorithm is motivated by the desire to have a unified approach to combining the two similarities that works equally well for the unipartite (both directed and undirected), as well as the bipartite settings (both matched and unmatched).

4.2 The `USim` Algorithm

The main idea behind the `USim` algorithm is to combine the two similarity (or adjacency) matrices in a multiplicative manner, as opposed to the additive approach in (4.1). Once we have a single similarity matrix, the rest of the algorithm follows standard spectral clustering procedures. Let us start with the unipartite case which is easier to compare with the average similarity approach.

4.2.1 Unified multiplicative similarity

Undirected Unipartite (UU) setting. Continuing with the unipartite setup of Section 4.1, the `USim` algorithm forms the following *unified similarity* matrix

$$S_\tau = S_{\tau,\alpha} := (K + \tau I)^\alpha A (K + \tau I)^\alpha \tag{4.2}$$

for some exponent $\alpha \in \mathbb{R}$ and tuning parameter $\tau \in [0, \infty)$. We often assume $\alpha > 0$—in fact, we are mostly interested in the choices $\alpha = 1$ and $\alpha = 1/2$—although we have stated the general form (4.2) since the algorithm is easily applicable for any $\alpha \in \mathbb{R}$ and $\tau > 0$. Note

that by letting $\alpha \to 0$, we recover the adjacency matrix of the network: $S_{\tau,\alpha} \to A$ (pure network information).

As in the case of the average similarity, we can use τ to tune in the contributions of the two sources of the information. For example, as $\tau \to \infty$, we have $S_\tau \sim \tau^{2\alpha} A$, that is, in the limit of large τ, the unified similarity only carries the network information. We note that with our choice of weighting scheme, the average similarity of (4.1) behaves similarity for large τ, that is, $W_\tau \sim \tau A$ as $\tau \to \infty$. Let us summarize these observations:

$$\frac{S_\tau}{\tau^{2\alpha}} \to A, \quad \text{and} \quad \frac{W_\tau}{\tau} \to A, \quad \text{as } \tau \to \infty.$$

The fact that both similarities approach, after normalization, the same limit (of no covariate information) is helpful when we compare them in simulations. Note, however, that in the limit of small τ, the two behave differently: As $\tau \to 0$, we have $W_\tau \to K$ (pure covariate information) while $S_\tau \to K^\alpha A K^\alpha$.

Bipartite setting (undirected). Now, consider the bipartite setting, with the adjacency matrix $A \in [0,1]^{n_1 \times n_2}$ and assume that we have covariates $\{x_{li}, i = 1, \ldots, n_l\} \subset \mathbb{R}^{d_l}$ for $l = 1, 2$. Using two (possibly different) kernel functions $k_l, l = 1, 2$, we can form the corresponding kernel matrices, for each of the two sides: $K_l = (k_l(x_{li}, x_{lj})) \in \mathbb{R}^{n_l \times n_l}$ for $l = 1, 2$. The bipartite multiplicative unified similarity is then given by

$$S_\tau = S_{\tau,\alpha} := (K_1 + \tau_1 I)^{\alpha_1} A (K_2 + \tau_2 I)^{\alpha_2} \tag{4.3}$$

where $\alpha = (\alpha_1, \alpha_2) \in \mathbb{R}^2$ is a set of exponents and $\tau = (\tau_1, \tau_2) \in [0, \infty)^2$ is a set of tuning parameters. We will assume throughout that α_1 and α_2 are the same, and with some abuse of notation write $\alpha_1 = \alpha_2 = \alpha$. It is clear that (4.2) can be thought of as a special case (4.3) in the undirected unipartite setting. (Even in the undirected unipartite case, one can use two different kernel functions, hence two kernel matrices on the two sides, though perhaps less justified in due to symmetry in that case.) One has a similar limiting behavior as $\tau_1, \tau_2 \to \infty$, that is, $S_\tau \sim \tau_1^{\alpha_1} \tau_2^{\alpha_2} A$, the pure network information case. As we will discuss below, the same similarity matrix (4.3) can be used to do both matched and unmatched bipartite clustering.

Directed Unipartite (DU) setting. In the directed case, A is an asymmetric $n \times n$ matrix where row i of A gives the sending pattern for node i and column j gives the receiving pattern for node j. Thus, in clustering applications, there are two labels associated with node i; one for the ith column and one for the ith row. One can view the adjacency matrix in this case as the bi-adjacency matrix of a bipartite network (with $n_1 = n_2 = n$). Thus, this case reduces in a straightforward manner to the (unmatched) bipartite setup. The general unified similarity in (4.3) can be used with obvious modifications. We have included this case, since some spectral clustering algorithms in the literature such as D-sim algorithm of [51], refer specifically to this case.

4.2.2 The spectral step

Once we have a single unified similarity matrix based on all the information, we can apply a standard spectral clustering algorithm to recover the clusters. We will focus the discussion on the bipartite setting which can be easily specialized to the unipartite setting as well.

Algorithm 3 `USim` Algorithm: bipartite (or directed unipartite) version

Input: Weighted bi-adjacency matrix $S \in \mathbb{R}_+^{n_1 \times n_2}$ and the number of clusters of the two sides: (r_1, r_2). In the matched case, we require $r_1 = r_2$.

1: Let $D_1 = \text{diag}(\sum_j S_{ij}, i \in [n_1])$ and $D_2 = \text{diag}(\sum_i S_{ij}, j \in [n_2])$ be the diagonal matrices of the degrees of the two sides.

2: Form the normalized Laplacian $L = D_1^{-1/2} S D_2^{-1/2}$.

3: Let $L = USV^T$ be the SVD of L truncated to $r = \min\{r_1, r_2\}$ largest singular values ($U \in \mathbb{R}^{n_1 \times r}$ and $V \in \mathbb{R}^{n_2 \times r}$).

4: Normalize each row of U and V to unit ℓ_2 norm to get $\widetilde{U}$ and $\widetilde{V}$, resp..

5: For the *matched setting*, perform k-means, with r clusters, on the rows of $[\widetilde{U}; \widetilde{V}] \in \mathbb{R}^{(n_1+n_2) \times r}$ (vertical concatenation).

6: For the *unmatched (or directed unipartite) setting*, separately perform k-means, with r_1 and r_2 clusters, on the rows of $\widetilde{U}$ and $\widetilde{V}$, respectively.

Algorithm 3 shows the Laplacian-based spectral clustering step of the `USim` algorithm for the two bipartite settings: matched and unmatched. Here, we are assuming that the overall unified similarity is nonnegative, so that the degree of a node is well-defined as a nonnegative number. The matched version of the algorithm obtains r clusters on each side which are in one to one correspondence. The matching is obtained because only r labels are used in assigning all the $n_1 + n_2$ rows of the concatenated matrix $[\widetilde{U}; \widetilde{V}]$. This approach is a modification of the algorithm of [22], and has been discussed in Section 3.5 as an initialization of for the `mbiSBM`. As opposed to the matched case, in the unmatched bipartite setting, we can potentially have two different number of clusters on the two sides: r_1 and r_2. It is also worth noting that exactly the same approach can be used for both (i) unmatched bipartite and (ii) directed unipartite settings as discussed earlier. This version of the algorithm thus resembles D-sim algorithm of [51] (derived for the network-only situation).

4.2.3 High-level analysis

The general approach in analyzing spectral clustering algorithms is based on the Davis-Kahan perturbation theory. The main ingredient is controlling the deviation of the similarity matrix under consideration from a noiseless target. Let us consider the unipartite setup, and assume that we have a stochastic model for the network adjacency matrix A, as well as the node covariates $\{x_i\}$. Then, the unified similarity S_τ given in (4.2) is a stochastic quantity and we expect it to concentrate, under suitable conditions, around a target matrix, say $\bar{S}_\tau = (\mathbb{E}K + \tau I)^\alpha (\mathbb{E}A)(\mathbb{E}K + \tau I)^\alpha$. We can then try to relate the perturbation $S_\tau - \bar{S}_\tau$ to that those in the kernel matrix $\Delta_K = K - \mathbb{E}K$ and the adjacency matrix $\Delta_A = A - \mathbb{E}A$. Consider the case $\alpha = 1$. Then, the following multiplicative perturbation result is useful in this regard (See the Appendix 4.A.2 for the proof)

Lemma 13. *Let A, A', B, B' be symmetric $n \times n$ matrices, and assume that*

$$\|A - A'\| \le \delta_1 \lesssim \|A'\|, \quad \|B - B'\| \le \delta_2 \lesssim \|B'\|.$$

Consider the matrix function $f(A, B) = BAB$. Then,

$$\|f(A, B) - f(A', B')\| \lesssim \delta_1 \|B'\|^2 + \delta_2 \|A'\| \|B'\|.$$

Assuming that the relative errors in the kernel matrix and the adjacency matrix are bounded above by constants, i.e., $\|\Delta_K\| \lesssim \|\mathbb{E}K\|$ and $\|\Delta_A\| \lesssim \|\mathbb{E}A\|$, we have

$$\begin{aligned} \|S_\tau - \bar{S}_\tau\| &\lesssim \|\Delta_A\| \, \|\mathbb{E}K + \tau I\|^2 + \|\Delta_K\| \, \|\mathbb{E}A\| \, \|\mathbb{E}K + \tau I\| \\ &= \|\Delta_A\| \, (\|\mathbb{E}K\| + \tau)^2 + \|\Delta_K\| \, \|\mathbb{E}A\| \, (\|\mathbb{E}K\| + \tau) \end{aligned}$$

where the second line uses fact that $\mathbb{E}K$ is PSD and that the eigenvalues of $\mathbb{E}K + \tau I$ are those of $\mathbb{E}K$ shifted up by $\tau > 0$.

4.2.4 Tuning the parameters

As discussed earlier, in the `USim` algorithm, we often set $\alpha = 1$ or $\alpha = 1/2$. However, we need to tune the τ parameter. The kernel matrix also usually has extra parameters (such as the bandwidth for distance kernels) that need to be tuned.

4.2.4.1 Tuning Distance kernel's scale parameter γ

Consider the distance kernels class, that is, given two samples x_i and x_j:

$$k(x_i, x_j) = g\Big(\frac{\|x_i - x_j\|}{\gamma}\Big) \tag{4.4}$$

Two popular kernels in this class are *Gaussian* and *exponential* kernels, where the Gaussian kernel is defined as $k(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{\gamma})$ and the exponential kernel is defined as $k(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|}{\gamma})$. As the kernel parameter γ increases, the structure of the kernel matrix tends to change from block diagonal to low-rank. One can verify that when $\gamma \to 0$, then the kernel matrix goes to the identity matrix and based on Ostrowski's theorem (See the Appendix 4.A.1), S_τ will behave like A.

Figure 4.1 shows the effect of changing the Gaussian kernel matrix parameter built from the covariate X_1 of Wikipedia data, that is, the user side of the Wikipedia data Cities.

It is known that spectral clustering can be sensitive to the choice of its parameters. Let r be the true number of clusters. Since $\mathbb{E}[K]$ has only r non-zero eigenvalues, it is reasonable to choose the parameter of the kernel matrix based on the eigengap between the r-th and $(r+1)$-th eigenvalues of the kernel matrix. Thus, to choose the parameter of the kernel, we do a grid search and choose the parameter that maximizes $\lambda_r - \lambda_{r+1}$.

4.2.4.2 Tuning τ

In the undirected unipartite case, the parameter τ is chosen to be the value which minimizes the k-means objective function, that is, the within cluster sum of squares,

$$\sum_{i=1}^{r} \sum_{\tilde{u}_j \in C_i} \|\tilde{u}_j(\tau) - m_i(\tau)\|^2 \tag{4.5}$$

where, $\tilde{u}_j$ is the j-th row of $\tilde{U}$.

In the bipartite and directed case, replace $\tilde{u}_j$ with z_j, where z_j is the j-th row of $[\widetilde{U}; \widetilde{V}]$.

4.3 Simulations

In this section, we show the effectiveness of `USim` in recovering the true labels in synthetic networks.

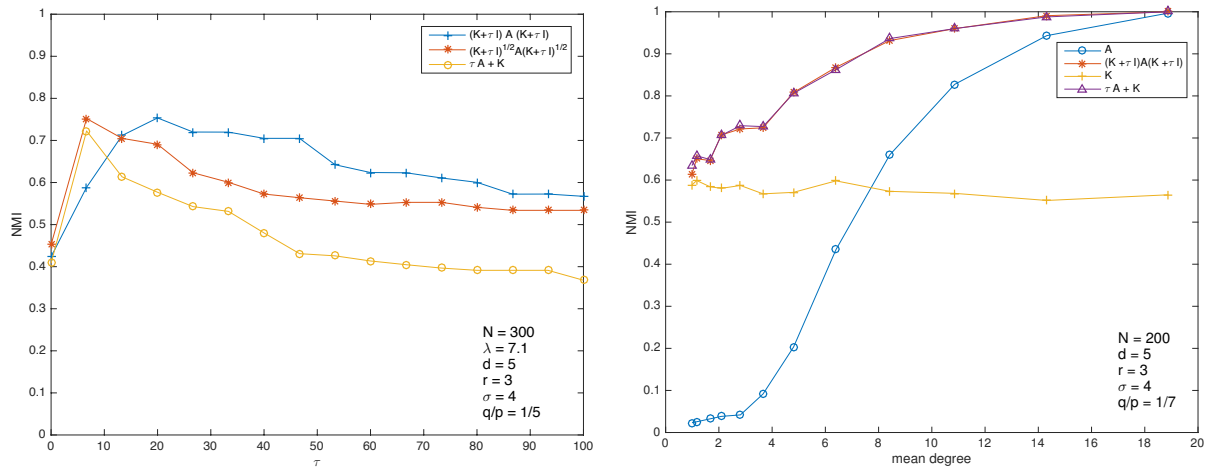
4.3.1 Undirected Unipartite (UU) Case

We simulate the $n \times n$ adjacency matrix A from the regular stochastic block model. This model postulates that the true node labels are drawn independently from the multinomial distribution with parameter π. More specifically, we consider the planted partition model where each node is assigned to one of the r 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. In other words, the connectivity matrix is: $\Psi = q11^T + (p-q)I \in [0,1]^{r \times r}$. Throughout this section, we fix $r = 3$ and $\pi = (1/3, 1/3, 1/3)$.We often reparametrize so we have a particular out-in-ratio q/p and ex-

pected degree λ. that We generate the covariates from the following Gaussian mixture model with $\sigma = 4$:

$$x_i = \mu_k + \sigma z_i, \quad \text{if } i \text{ belongs to cluster } k, \quad z_i \overset{\text{iid}}{\sim} N(0, I_d)$$

The cluster centers $\mu_k, k = 1, \ldots, r$ are chosen uniformly at random from the standard basis vector of $\mathbb{R}^d$, that is, from the uniform distribution on $\{e_1, \ldots, e_d\}$.



(a) (b)

Figure 4.1: (a) Effect of changing τ for a specific network with mean degree = 7.1. (b) Performance of Usim on Undirected Unipartite (UU) Case for different mean degrees

Figure 4.1(a) shows the effect of varying τ for the network with expected degree $\lambda = 7.1$ and out-in-ratio $q/p = 1/5$. As discussed earlier, both $S_{\tau,\alpha} = (K + \tau I)^\alpha A (K + \tau I)^\alpha$ and $W_\tau = K + \tau A$, behave similar to A for large τ, while for small τ, W_τ behaves like K. Thus, the performance of W_τ at the two extremes $\tau = 0, \infty$ corresponds to clustering based on covariates alone and based on network information alone. Based on Figure 4.1(a), combining the kernel with the adjacency matrix boosts the information, as the NMI is higher somewhere between these two extreme cases. It also shows that `USim` is more robust to the change of τ compared to its additive counterpart.

The Figure 4.1(b) shows that combining node covariates with the adjacency matrix will result in higher NMI compared to the cases where only A or the kernel K is used for the

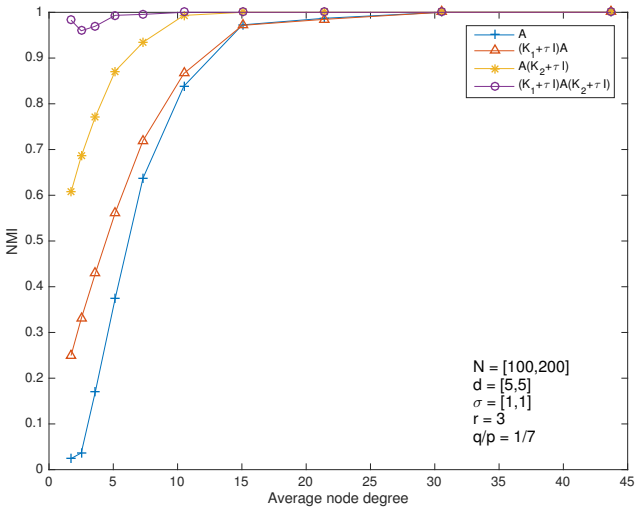


Figure 4.2: Performance of Usim on matched bipartite setting for different mean degrees

inference of node labels. In addition, the sparser the network is, the more significant the added benefit of covariates becomes.

4.3.2 Undirected Matched Bipartite Case

Figure 4.2 shows the performance of `USim` in the matched bipartite settings for various combination of the covariates. The network and covariates are generated from the `mbiSBM` model as in Section 3.2. We recall that biSC which is used as initialization in Section 3.5 corresponds to the matched version of Algorithm 3 in the limit of large τ. The plots show that `USim` with modest value of τ outperforms `biSC`, and the improvement is quite significant the sparser the network becomes.

4.4 Application

4.4.1 Wikipedia Examples Revisited

Let's revisit Wikipedia examples from Chapter 3 and apply `USim` on the two Wikipedia page–user networks, which we called TopArticles and Cities. Recall that each is a bipartite

Network	biSC	(USim)		
		X_1 & X_2	X_1	X_2
TopArticles	0.86	-	-	0.99
Cities	0.6	0.99	0.60	0.95

Table 4.1: Matched NMI for biSC and USim on the two Wikipedia networks.

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.

In TopArticles network, 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 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 $X_1 \in \mathbb{R}^{N_1 \times 2}$.

Results on Wikipedia networks. Table 4.1 illustrates the result of the application of biSC and USim algorithm with various combination of covariates. The table shows that using USim outperforms biSC.

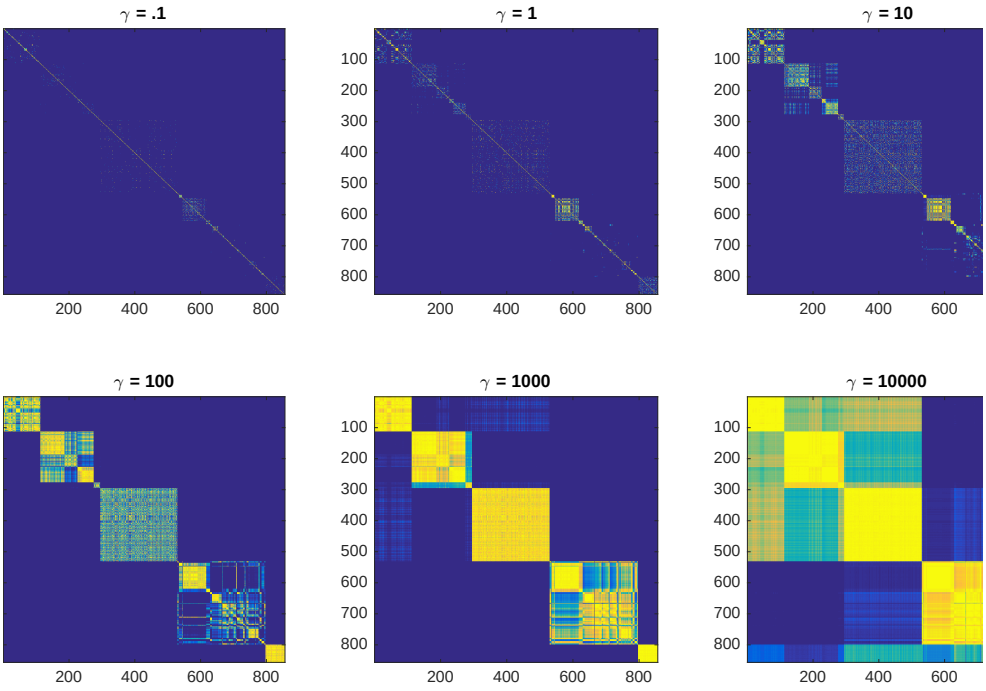


Figure 4.1: Effect of changing kernel parameter γ on the Wikipedia CITIES data

4.A Appendix

4.A.1 General form of Ostrowski's theorem

Let A be a Hermitian $n \times n$ matrix, and B an $n \times n$ matrix. Then, for each $1 \leq i \leq n$, there exists an nonnegative real θ_i with $\lambda_n(BB^*) \leq \theta_i \leq \lambda_1(BB^*)$, such that

$$\lambda_i(BAB^*) = \theta_i \lambda_i(A)$$

In particular, the number of positive (respectively, negative) eigenvalues of BAB^* is at most the number of positive(respectively, negative) eigenvalues of A. Consequently,

$$|\lambda_i(BAB^*) - \lambda_i(A)| \leq |\lambda_i(A)|\|B^*B - I\|$$

4.A.2 Proof of Lemma 13

Proof. We have

$$\|f(A, B) - f(A', B)\| = \|B(A - A')B\| \leq \|B\|^2\|A - A'\| \leq \delta_1\|B\|^2$$

and using Lemma 14,

$$\|f(A', B) - f(A', B')\| \leq 2\delta_2\|A'\|\|B'\| + \delta_2^2\|A'\|$$

We also have $\|B\|^2 \leq (\|B'\| + \delta_2)^2 \leq 2\|B'\|^2 + 2\delta_2^2$. Putting the pieces together using the triangle inequality

$$\|f(A, B) - f(A', B')\| \leq \|f(A, B) - f(A', B)\| + \|f(A', B) - f(A', B')\|.$$

we get the result. □

Lemma 14. *Let $g(B) = BAB$, and assume $\|B' - B\| \leq \delta$. Then,*

$$\|g(B) - g(B')\| \leq 2\delta\|A\|\|B'\| + \delta^2\|A\|$$

Proof. We have $\|B\| \leq \|B'\| + \delta$, and

$$\begin{aligned}
\|g(B) - g(B')\| &\leq \|BAB - B'AB\| + \|B'AB - B'AB'\| \\
&\leq \|B - B'\|\|AB\| + \|B'A\|\|B - B'\| \\
&\leq \delta\|A\|\|B\| + \|B'\|\|A\|\delta \\
&\leq \delta\|A\|(2\|B'\| + \delta).
\end{aligned}$$

□

CHAPTER 5

Discussion

In this dissertation, we first 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 as mixtures of multinomial distributions. Some care however is needed when deciding the distribution of the top layer to allow for proper information sharing among the lower level variables (v_{rk} in our notation). In addition, as mentioned (cf. Section 3.2.5), 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 [52]. 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 time-stamps 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.

Since the `mbiSBM` model has some restrictive assumptions, that is, the normality and independence requirement for the covariates, we also proposed a new approach *unified similarity clustering* (`USim`) that combines the information from the network with that from the node covariates into a single similarity matrix, which can then be input to a spectral clustering algorithm. `USim` can be used in both the unipartite as well as the bipartite settings.

Computation of kernel matrices can be quite expensive. Incorporating kernel approximation methods such as the Nyström approach into `USim` can speed up kernel computation. We would also like to consider the possibility of soft assignments of nodes to clusters in `USim` for future work.

Bibliography

[1] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks* 5.2 (1983), pp. 109–137.

[2] B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E* 83.1 (2011), pp. 1–11. arXiv: 1008.3926.

[3] P. K. Gopalan and D. 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 (2013), pp. 14534–9. arXiv: arXiv:1408.1149.

[4] P. J. Bickel and A. 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 (2009), pp. 21068–21073.

[5] A. Decelle et al. Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters* 107.6 (2011), pp. 1–5. arXiv: 1102.1182.

[6] K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *Annals of Statistics* 39.4 (2011), pp. 1878–1915. arXiv: 1007.1684.

[7] E. Mossel, J. Neeman, and A. Sly. Reconstruction and estimation in the planted partition model. *Probab. Theory Relat. Fields* 162(3) (2015), pp. 431–461. arXiv: 1202.1499.

[8] Y. Zhao, E. Levina, and J. Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *Annals of Statistics* 40.4 (2012), pp. 2266–2292. arXiv: 1110.3854.

[9] A. A. Amini et al. Pseudo-likelihood methods for community detection in large sparse networks. *Annals of Statistics* 41.4 (2013), pp. 2097–2122. arXiv: arXiv:1207.2340v3.

[10] T. Qin and K. Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. *Advances in Neural Information Processing Systems* (2013), pp. 1–9. arXiv: arXiv:1309.4111v1.

[11] E. Mossel, J. Neeman, and A. Sly. A Proof Of The Block Model Threshold Conjecture. *arXiv preprint arXiv:1311.4115* (2013). arXiv: 1311.4115.

[12] L. Massoulié. Community detection thresholds and the weak Ramanujan property. *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)* (2014), pp. 694–706. arXiv: 1311.3085.

[13] A. A. Amini and E. Levina. On semidefinite relaxations for the block model. *arXiv preprint arXiv:1406.5647* (2014). arXiv: 1406.5647.

[14] C. Gao et al. Achieving Optimal Misclassification Proportion in Stochastic Block Model. *arXiv preprint arXiv:1505.03772v5* (2015). arXiv: 1505.03772.

[15] L. Jing and A. Rinaldo. Consistency of spectral clustering in stochastic block models. *Annals of Statistics* 43.1 (2015), pp. 215–237. arXiv: arXiv:1312.2050v3.

[16] B. Hajek, Y. Wu, and J. Xu. Achieving Exact Cluster Recovery Threshold via Semidefinite Programming: Extensions. *IEEE Transactions on Information Theory* 62.10 (2016), pp. 5918–5937. arXiv: 1412.6156.

[17] E. Abbe, A. S. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory* 62.1 (2016), pp. 471–487. arXiv: 1405.3267.

[18] C. Gao et al. Community Detection in Degree-Corrected Block Models. *arXiv preprint arXiv:1607.06993* (2016). arXiv: 1607.06993.

[19] J. A. Hartingan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Society* 67.337 (1972), pp. 123–129.

[20] Y. Cheng and G. 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 (2000), pp. 93–103.

[21] S. C. Madeira et al. 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 (2010), pp. 153–165.

[22] I. 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* (2001), pp. 269–274.

[23] I. S. Dhillon. Information-Theoretic Co-clustering. *Proc. of 9th ACM SIGKDD Int'l Conf.* (2003), pp. 89–98.

[24] G. Bisson and F. Hussain. Chi-Sim: A new similarity measure for the co-clustering task. *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008* (2008), pp. 211–217.

[25] D. Choi and P. J. Wolfe. Co-clustering separately exchangeable network data. *Annals of Statistics* 42.1 (2014), pp. 29–63. arXiv: 1212.4093.

[26] C. J. Flynn and P. O. Perry. Consistent Biclustering. *arXiv preprint arXiv:1206.6927* (2012). arXiv: 1206.6927.

[27] K. Rohe, T. Qin, and B. 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 (2016), pp. 12679–12684.

[28] T. Zhou et al. Bipartite network projection and personal recommendation. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 76.4 (2007), pp. 1–7. arXiv: 0707.0540.

[29] D. B. Larremore, A. Clauset, and A. Z. Jacobs. Efficiently inferring community structure in bipartite networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 90.1 (2014), pp. 17–22. arXiv: 1403.2933.

[30] J. Wyse, N. Friel, and P. Latouche. Inferring structure in bipartite networks using the latent block model and exact ICL. 2013 (2014), p. 23. arXiv: 1404.2911.

[31] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B* 48.3 (1986), pp. 259–302.

[32] M. E. J. Newman and A. Clauset. Structure and inference in annotated networks. *Nature Communications* 7.May (2016), pp. 1–16. arXiv: 1507.04001.

[33] B. Yan and P. Sarkar. Convex Relaxation for Community Detection with Covariates. *arXiv preprint arXiv:1607.02675* (2016). arXiv: 1607.02675.

[34] Y. Zhang, E. Levina, and J. Zhu. Community Detection in Networks with Node Features. *Electronic Journal of Statistics* 10.2 (2016), pp. 3153–3178. arXiv: arXiv:1509.01173v1.

[35] N. Binkiewicz, J. T. Vogelstein, and K. Rohe. Covariate-assisted spectral clustering. *arXiv preprint arXiv:1411.2158* (2014). arXiv: 1411.2158.

[36] S. Günnemann et al. Spectral subspace clustering for graphs with feature vectors. *Proceedings - IEEE International Conference on Data Mining, ICDM* (2013), pp. 231–240.

[37] R. Vershynin. *High-Dimensional Probability with Applications in Data Science*.

[38] M. I. Jordan et al. Introduction to variational methods for graphical models. *Machine Learning* 37.2 (1999), pp. 183–233. arXiv: arXiv:1103.5254v3.

[39] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational Inference: A Review for Statisticians. *arXiv preprint arXiv:1601.00670* (2016). arXiv: 1601.00670.

[40] A. Gelman et al. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC Texts in Statistical Science, 2003.

[41] A. Dasgupta, J. Hopcroft, and F. McSherry. Spectral Analysis of Random Graphs with Skewed Degree Distributions. *45th Annual IEEE Symposium on Foundations of Computer Science* (2004), pp. 602–610.

[42] M. J. Wainwright and M. I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning* 1.1-2 (2008), pp. 1–305.

[43] J. Douglas and H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical society* 82 (1956), pp. 421–439.

[44] D. O'Connor and L. Vandenberghe. Primal-Dual Decomposition by Operator Splitting and Applications to Image Deblurring. *SIAM Journal on Imaging Sciences* 7.3 (2014), pp. 1724–1754.

[45] A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science* 286.October (1999), pp. 509–512. arXiv: 9910332 [cond-mat].

[46] F. M. Malvestuto. Statistical treatment of the information content of a database. *Information Systems* 11.3 (1986), pp. 211–223.

[47] *Wikimedia Statistics*. 2016. URL: https://stats.wikimedia.org.

[48] B. Keegan. *GitHub repository: 2014-On-Wikipedia*. 2014. URL: https://github.com/brianckeegan/2014-On-Wikipedia.

[49] D. Kahle and H. Wickham. ggmap : Spatial Visualization with. *The R Journal* 5.1 (2013), pp. 144–161.

[50] *IP lookup API*. 2016. URL: http://ipapi.co.

[51] K. Rohe. Co-clustering for directed graphs : the Stochastic co-Blockmodel and spectral algorithm Di-Sim (2015), pp. 1–39. arXiv: arXiv:1204.2296v2.

[52] M. Kivela et al. Multilayer networks. *Journal of Complex Networks* 2.3 (2014), pp. 203–271. arXiv: 1309.7233.