Statistical Tools for Network Data: Prediction and Resampling

by

Tianxi Li

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2018

Doctoral Committee:

Professor Liza Levina, Co-Chair
Professor Ji Zhu, Co-Chair
Assistant Professor Danai Koutra
Professor Kerby Shedden

Tianxi Li

tianxili@umich.edu

ORCID iD: 0000-0003-4595-1777

ACKNOWLEDGEMENTS

I want to give my biggest thanks to my advisors Prof. Liza Levina and Ji Zhu, for their warm and endless support, for the kindness and wisdom they passed on and for the moments they cheered me up from depression due to academic and personal difficulties. Five years is only a short period compared to the lifetime, but I learned so much from them that I will benefit from for the rest of my life. I am also extremely grateful to my wife, Meng, without whom I might need another three years to finish the work and my two little buddies, Jensen and Arthur, without whom I might finish the work one year earlier. Words can hardly express how happy I am with them and how much I love them. Thanks should also be attributed to my parents for their love and confidence in me. Their support is indispensable for everything that I accomplish. I also want to thank Prof. Kerby Shedden and Danai Koutra for their time to serve on the committee and their useful comments on my research. Finally, I feel very proud to be part of the big family - the Department of Statistics at University of Michigan. The time here is so precious and will be the most memorable period of my life. Go Blue!

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

Advances in data collection and social media have led to more and more network data appearing in diverse areas, such as social sciences, internet, transportation and biology. This thesis develops new principled statistical tools for network analysis, with emphasis on both appealing statistical properties and computational efficiency.

Our first project focuses on building prediction models for network-linked data. Prediction algorithms typically assume the training data are independent samples, but in many modern applications samples come from individuals connected by a network. For example, in adolescent health studies of risk-taking behaviors, information on the subjects' social network is often available and plays an important role through network cohesion, the empirically observed phenomenon of friends behaving similarly. Taking cohesion into account in prediction models should allow us to improve their performance. We propose a network-based penalty on individual node effects to encourage similarity between predictions for linked nodes, and show that incorporating it into prediction leads to improvement over traditional models both theoretically and empirically when network cohesion is present. The penalty can be used with many loss-based prediction methods, such as regression, generalized linear models, and Cox's proportional hazard model. Applications to predicting levels of recreational activity and marijuana usage among teenagers from the AddHealth study based on both demographic covariates and friendship networks are discussed in detail. We show that our approach to taking friendships into account can significantly improve predictions of behavior while providing interpretable estimates of covariate effects.

Resampling, data splitting, and cross-validation are powerful general strategies in statistical inference, but resampling from a network remains a challenging problem. Many statistical models and methods for networks need model selection and tuning parameters, which could be done by cross-validation if we had a good method for splitting network data; however, splitting network nodes into groups requires deleting edges and destroys some of the structure. Here we propose a new network cross-validation strategy based on splitting edges rather than nodes, which avoids losing information and is applicable to a wide range of network models. We provide a theoretical justification for our method in a general setting and demonstrate how our method can be used in a number of specific model selection and parameter tuning tasks, with extensive numerical results on simulated networks demonstrating its competitiveness with task-specific methods. We also apply the method to analysis of a citation network of statisticians and obtain meaningful research communities.

Finally, we consider the problem of community detection on partially observed networks. Communities are one important type of structure in networks and they have been widely studied. However, in practice, network data are often collected through sampling mechanisms, such as survey questionnaires, instead of direct observation. The noise and bias introduced by such sampling mechanisms can obscure the community structure and invalidate the assumptions of standard community detection methods. We propose a model to incorporate neighborhood sampling, through a model reflective of survey designs, into community detection for directed networks, since friendship networks obtained from surveys are naturally directed. We model the edge sampling probabilities as a function of both individual preferences and community parameters, and fit the model by a combination of spectral clustering and the method of moments. The algorithm is computationally efficient and comes with a theoretical guarantee of consistency. We evaluate the proposed model in extensive simulation studies and applied it to a faculty hiring dataset, discovering a meaningful hierarchy of communities among US business schools.

CHAPTER I

Introduction

Advances in data collection and social media have resulted in network data being collected in many applications and at the same time, networks have been widely used to describe relationships between individuals or interactions between units of complex systems in diverse fields, including but not limited to biology, computer science, sociology and economics. There has been significant amount of work in the past two decades on network analysis and modeling which have already provided salient insights about many mechanisms such as gene regulation, friendship formulation and eco-system evolution [Newman, 2010]. Some networks can be directly observed, such as the social networks from online social media or road networks to describe transportation systems, while others may be inferred from other analysis, such as the protein-to-protein interaction networks or brain connectomes. Moreoever, the network information is sometimes collected along with more traditional covariates on each unit of analysis such as characteristics of each person, gene expressions of each patient etc. [Michell and West, 1996, Pearson and Michell, 2000, Pearson and West, 2003, Harris, 2009, Ji et al., 2016]. One example of such network data is the survey data from the National Longitudinal Study of Adolescent Health (the AddHealth study) [Harris, 2009]. AddHealth was a major national longitudinal study of students in grades 7-12 during the school year 1994-1995, after which three further follow-ups were conducted in 1996, 2001-2002, and 2007-2008. In the Wave I survey, all students in the sample completed in-school questionnaires, and a subsample completed a follow-up in-home interview with more detailed questions. There are questions in both the in-school survey and the in-home interview asking students to name their friends (up to 10) so friendship networks connecting students can be constructed based on this information and one can analyze the network structures to obtain insights about the friendship relation between students. In addition to the information about friends, the survey also asked hundreds of questions about various aspects of the students personal and school life, collecting information about age, gender, race, socio-economic status, health, academic achievement, etc. Figure 1.1 shows the friendship network between students in one school from the AddHealth study as well as their race information.

In general, we can represent network data in the following way: given n nodes, indexed by $i = 1, 2, \cdots, n$, we have a network connecting the n nodes, represented by an adjacency matrix $A \in \mathbb{R}^{n\times n}$ such that

$$A_{ii'} = \mathbf{1}\{\text{there is an edge from } i \text{ to } i', \text{denoted by } i \to i'\}.$$

More generally, the network can be weighed, in which case we will define A as a real matrix instead where each entry represents the edge weight or when the network is undirected, we define A to be a symmetric matrix. In some situations, we may also observe $(\boldsymbol{x}_i, y_i), i = 1, 2 \cdots, n$, where $\boldsymbol{x}_i \in \mathbb{R}^p$ is the covariate vector and y_i is the response variable for the node i. We can denote the matrix stacking each $\boldsymbol{x}_i$ as the ith row by X and the corresponding vector stacking all y_i's by Y. When such X and/or Y is available, we can call the triple (A, X, Y) as a network-linked data set.

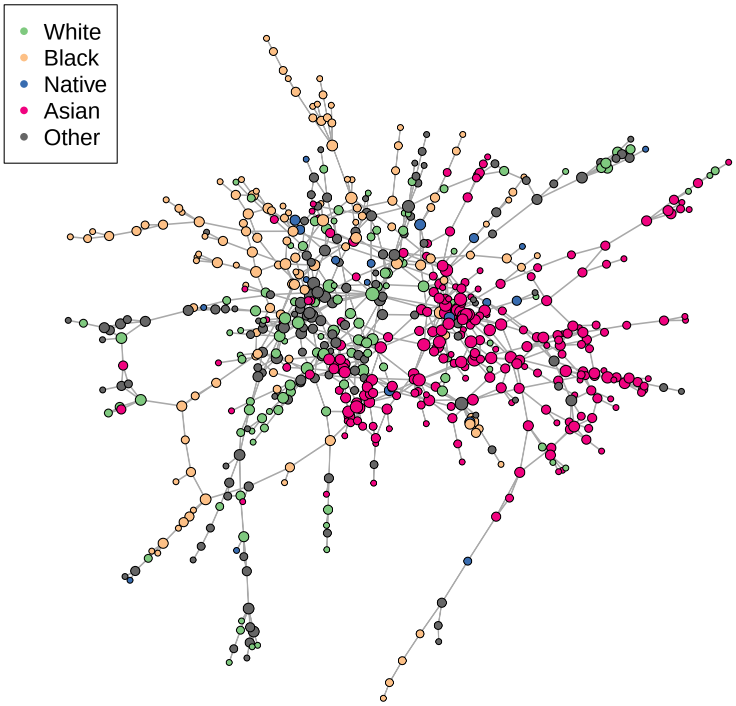


Figure 1.1: The friendship network between high school students from AddHealth study, where the edges indicate friend nominations. The nodes are colored according to the race information of students.

Many questions can be asked about analyzing a network dataset. For example, how can one build a prediction model on a network-linked data set and what are the benefits of the network information added to classical prediction setting? When the target is to understand the network structures, how can we build realistic models as well as make valid inference under a principled statistical framework? In the next a few chapters, some recent work to answer these questions will be introduced. However, we will first give a brief review for the classical setup of both predictive modeling and network analysis.

1.1 Notations

Given a positive integer n, define $[n] = \{1, 2, \cdots, n\}$. We will use the lower-case letters such as x to denote scalars while the bold version such as $\boldsymbol{x}$ to denote vectors, which we will treat as column vectors by default. Matrices are denoted by upper-case letters such as X. The transpose and trace of a matrix X is denoted by X^T and $\mathrm{tr}(X)$ respectively. For any matrix X, we use $\|X\|$ to denote its spectral norm, which is the largest singular value of X and $\|X\|_F$ to denote its Frobenius norm, defined by $\|X\|_F^2 = \sum_{ij} X_{ij}^2$. We use $\mathbf{1}_n$ to denote the column vector of n 1's and I_n to denote the $n \times n$ identity matrix. When it is clear in context, we may suppress the subscript.

1.2 Loss-based prediction models

Perhaps the most basic prediction model is the linear regression model. Given pairs of $(\boldsymbol{x}_i, y_i), i \in [n]$, where $[n] := \{1, 2, \cdots, n\}$, we assume

$$y_i = \alpha + \boldsymbol{x}_i^T \boldsymbol{\beta} + \epsilon_i.$$

In the model, $\boldsymbol{\beta} \in \mathbb{R}^p$ measures covariate effects, $\alpha \in \mathbb{R}$ is the intercept and $\epsilon_i \in \mathbb{R}$ is random noise, typically assumed to be i.i.d in the simple setting. The interpretation of $\boldsymbol{\beta}$ as covariate effects is one of the most important advantages of the linear model, giving a measure about how much change in the response one would expect due to the change of one of the covariates while fixing the rest. This interpretation admits scientific meanings and is the major target of using the model in many applications. For example, in medical studies, when x_1 is the indicator of a treatment while y is the health condition, the covariate effect of x_1 measures whether (and to what extent) the treatment is effective in changing the health condition, after accounting for the effects of other covariates.

In spite of the simplicity, the linear modeling idea is very power in the sense that it can extended to a large class of loss-based prediction models. Given a link function ϕ, a generalized linear model [Nelder and Baker, 1972] is defined through the relationship

$$\phi(\mathbb{E}y_i) = \boldsymbol{x}_i^T \boldsymbol{\beta} + \alpha$$

where is distribution of y_i is assumed to from exponential family and the covariate effects are still assumed in a linear form.

The linear forms are also widely used beyond generalized linear models. For instance, in survival regression problems, the Cox's proportional hazard model [Cox, 1972] assumes the hazard for ith subject is in the form of

$$h_0(t) \exp(\boldsymbol{x}_i^T \boldsymbol{\beta} + \alpha)$$

where $h_0(t)$ is some unspecified baseline hazard function at time t. In classification problems, there is a family of classifiers that are assumed to have the form of

$$C(\boldsymbol{x}_i) = f(\boldsymbol{x}_i^T \boldsymbol{\beta})$$

for some function f such that we expect to observe

$$y_i = \operatorname{sign}(C(\boldsymbol{x}_i)).$$

One popular method in this family of classifiers is the support vector machine (SVM) [Vapnik, 2013], which assumes f to be the identity function in its standard form.

We classify all the above models in the same family called *loss-based prediction models* due to the common strategy available for model estimation - the *M-estimation.* In particular, all of the above methods can be estimated by the following problem

$$\text{minimize}_{\boldsymbol{\beta}\in\mathcal{T}}\ \mathcal{L}(\{y_i, \boldsymbol{x}_i^T\boldsymbol{\beta}+\alpha\}_{i=1}^n)$$

where $\mathcal{L}$ is some general loss function and $\mathcal{T}$ is some parameter domain. For generalized linear models, $\mathcal{L}$ is the log likelihood of the observation when one uses the maximum likelihood framework for model estimation. For the Cox's proportional hazard model [Cox, 1972], $\mathcal{L}$ gives the partial likelihood function of the observations, based on which the estimation of $\boldsymbol{\beta}$ can be obtained. For the SVM, the $\mathcal{L}$ function is the hinge loss on all observations while the feasible region $\mathcal{T}$ is certain ℓ_2 ball of $\boldsymbol{\beta}$, such that the estimation is done by

$$\begin{aligned}\text{minimize}_{\boldsymbol{\beta}}\ &\sum_i[1-y_i(\boldsymbol{x}_i^T\boldsymbol{\beta}+\alpha)]_+\\ \text{such that }&\|\boldsymbol{\beta}\|_2^2\leq\lambda\end{aligned}$$

for some tuning parameter λ, according to the formulation in Hastie et al. [2009].

1.3 Random network modeling

Statistical methods for analyzing networks have received a lot of attention because of its wide-ranging applications in areas such as sociology, physics, biology and medical sciences. Statistical models provide an effective way to extract structural information about the network while filtering out noisy and uninformative details thus become popular in understanding the network structures and formulation. Perhaps the simplest statistical network model is the famous Erdös-Renyi model [Erds and Rényi, 1960], after which a large body of interesting models followed, including the stochastic block model (SBM) [Holland et al., 1983] and its variants such as the degree-corrected stochastic block model (DCSBM) [Karrer and Newman, 2011] or mixed membership block model (MMBM) [Airoldi et al., 2008], and the latent space model [Hoff et al., 2002], just to name a few. In this section, we introduce a generic probabilistic framework for statistical network models - the random network modeling framework, under which a few standard models will be introduced as well.

Let $\mathcal{V} = [n]$ denote the node set of a network, and let A be its $n \times n$ adjacency matrix. We view A as a single random realization of independent Bernoulli variables, such that each entry of A is generated independently according to

$$A_{ii'} \sim \text{Bernoulli}(P_{ii'})$$

where $P = (P_{ii'}) \in [0,1]^{n\times n}$. For undirected networks, we require both P and A to be symmetric and only the upper-triangular entries of A are generated from the defined model. In this framework, the model P admits the structural information that one is interested in extracting while A is assumed to be a noisy version of P. The statistical modeling procedure is then estimating (some aspects of) P from the given noisy observation A. Many interesting statistical models have been proposed under the random network modeling framework. Below we introduce a few popular ones.

Erdös-Renyi model (ER) The most widely known random network model is the Erdös-Renyi model [Erds and Rényi, 1960]. Specifically, the model assumes for some $p \in [0,1]$

$$P = p \cdot \mathbf{1}_n \mathbf{1}_n^T = (p)_{n\times n}.$$

Essentially, the model assumes all of node pairs are randomly connected in a uniform way. However, the ER model is completely homogeneous and has no interesting structure. In reality, there is seldom any real world networks that can be described well by the ER model.

Stochastic block model One interesting generalization of the ER model for undirected networks is the stochastic block model (SBM) [Holland et al., 1983]. In the SBM, we assume each node belongs to one of the K communities. We use $\boldsymbol{c} = \{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ to denote the vector of membership for all nodes, such that $\boldsymbol{c}_i \in [K]$. Then the probability of having an edge between nodes i and j is $P(A_{ii'} = 1) = B_{\boldsymbol{c}_i\boldsymbol{c}_{i'}}$ for some $K \times K$ symmetric probability matrix B. The probability matrix P can be written as $P = ZBZ^T$ where $Z \in \{0, 1\}^{n \times K}$ has exactly one "1" in each row, with $Z_{ik} = 1$ if node i belongs to community k. This model generalizes the ER model by assuming the nodes are inhomogeneous across groups but remain the same within the groups.

Degree-corrected stochastic block model (DCSBM) One well-known limitation of the SBM lies in it forces equal expected degrees for all the nodes in the same community, therefore ruling out "hubs" - nodes that have abnormally large number of connections compared to the rest in the population. The degree corrected stochastic block model corrects this homogenous degree problem of the SBM by associating each node i with an individual degree parameter θ_i. Let $\Theta = \text{diag}(\theta_1, \cdots, \theta_n)$. The DCSBM then assumes $P(A_{ii'} = 1) = \theta_i\theta_{i'}B_{\boldsymbol{c}_i\boldsymbol{c}_{i'}}$. Equivalently, the P matrix is assumed to be $P = \Theta ZBZ^T\Theta$ (a constraint is needed on Θ to ensure identifiability, with different authors choosing different versions; here we follow Karrer and Newman [2011] and assume $\sum_{Vc_i=k} \theta_i = 1$, for each $k \in [K]$).

Random dot product graph (RDPG) The RDPG [Young and Scheinerman, 2007] is another generalization of the SBM and one special class of latent space models. In RDPG, each node i is associated with a latent K dimensional vector $Z_i \in \mathbb{R}^K$, such that $P_{ii'} = Z_i^T Z_{i'}$. Essentially, RDPG assumes the connectivity of the random network only depends on K latent factors in a linear way through inner product. It has been shown that RDPG has a very good embedding properties in Euclidean space in various problems [Sussman et al., 2014, Tang et al., 2017] and its limiting behaviors can also be studied [Tang and Priebe, 2016]. More details about the model can be found in the review paper of Athreya et al. [2017].

Graphon model. Observe that random network model framework assumes that the nodes are exchangeable. According to Aldous [1981], the probability matrix of any exchangeable random graph can be represented by

$$P_{ii'} = f(\xi_i, \xi_{i'})$$

where function $f : [0,1] \times [0,1] \rightarrow [0,1]$ is symmetric in its two arguments and is called "graphon", while $\xi_i, i \in [n]$ are independent uniform random variables on $[0,1]$. The representation is determined only up to a measure-preserving transformation [Diaconis and Janson, 2007]. There is a substantial literature on estimating the graphon under various assumptions [Wolfe and Olhede, 2013, Choi and Wolfe, 2014, Gao et al., 2015].

Though in this thesis, we embed our discussion of network modeling in the theme of exchangeable random network modeling framework introduced above, which includes most of current random network models in statistics, there are other frameworks available, such as the one discussed by Crane and Dempsey [2015].

1.4 Outline of the thesis

The rest of the thesis is organized as follows:

Chapter II focuses on improving prediction models by incorporating network information A available from the network-linked data. The high-level questions we try to answer in Chapter II are “what are the reasonable assumptions one should assume for prediction models on network-linked data?” and “how can one incorporate network information wit both computationally efficiency and statistical principles?”. Specifically, we reply on one generic assumption called “network cohesion” to build prediction models, an empirically observed phenomenon of friends on social networks behaving similarly. Taking such cohesion into account in prediction models allows us to improve prediction and modeling performance. There we propose a network-based penalty on individual node effects to encourage similarity between predictions for linked nodes, and show that incorporating it into prediction leads to improvement over traditional models both theoretically and empirically when network cohesion is present. The penalty can be used with all the loss-based prediction methods introduced in this chapter. Applications to predicting levels of recreational activity and marijuana usage among teenagers from the AddHealth study based on both demographic covariates and friendship networks are discussed in detail and show that our approach to taking friendships into account can significantly improve predictions of behavior while providing interpretable estimates of covariate effects.

Chapter III and IV focus on the problems under the random network modeling framework.

While many statistical models and methods are now available for network analysis, resampling network data remains a challenging problem. Cross-validation is a useful general tool for model selection and parameter tuning, but is not directly applicable to networks since splitting network nodes into groups requires deleting edges and destroys some of the network structure. In Chapter III, we propose a new network resampling strategy based on splitting edges rather than nodes, applicable to both cross-validation and bootstrap for a wide range of network model selection tasks. We provide a theoretical justification for our method in a general setting and examples of how our method can be used in specific network model selection and parameter tuning tasks. Numerical results on simulated networks and on a citation network of statisticians show that this cross-validation approach works well for model selection.

Chapter IV addresses a commonly encountered practical difficulty in community detection for directed networks. Communities are an important type of structure in networks and they have been widely studied. In practice, network data are often collected through sampling mechanisms, such as survey questionnaires, instead of direct observation. The noise and bias introduced by such sampling mechanisms can obscure the community structure and invalidate the assumptions of standard community detection methods. In Chapter IV, we propose a model to incorporate neighborhood sampling, through a model reflective of survey designs, into community detection for directed networks, since friendship networks obtained from surveys are naturally directed. We model the edge sampling probabilities as a function of both individual preferences and community parameters, and fit the model by a combination of spectral clustering and the method of moments. The algorithm is computationally efficient and comes with a theoretical guarantee of consistency. We evaluate the proposed model in extensive simulation studies and applied it to a faculty hiring dataset, discovering a meaningful hierarchy of communities among US business schools.

CHAPTER II

Prediction model on network-linked data

2.1 Introduction

There is a large body of work extending over decades on predicting a response variable of interest from such covariates, via linear or generalized linear models, survival analysis, classification methods, and the like, which typically assume the training samples are independent and do not extend to situations where the samples are connected by a network. There has not been much focus on developing a general statistical framework for using network data in prediction, although there are methods available for specific applications [Wolf et al., 2009, Asur and Huberman, 2010, Vogelstein et al., 2013].

In the social sciences and especially in economics, on the other hand, there has been a lot of recent interest in causal inference on the relationship between a response variable and both covariates and network influences; see e.g., Shalizi and Thomas [2011] and references therein, and Manski [2013]. While in certain experimental settings such inference is possible [Rand et al., 2011, Choi, 2017, Phan and Airoldi, 2015], in most observational studies on networks establishing causality is substantially more difficult than in regular observational studies. While network cohesion (a generic term by which in this chapter we mean linked nodes acting similarly) is a well known phenomenon observed in numerous social behavior studies [Fujimoto and Valente, 2012, Haynie, 2001, Christakis and Fowler, 2007], explaining it causally on the basis of observational data is very challenging. An excellent analysis of this problem can be found in Shalizi and Thomas [2011], showing that it is in general impossible to distinguish network cohesion resulting from homophily (nodes become connected because they act similarly) and cohesion resulting from contagion (behavior spreads from node to node through the links), and to separate that from the effect of node covariates themselves. However, making good predictions of node behavior is an easier task than causal inference, and is often all we need for practical purposes. Our goal in this chapter is to take advantage of the network cohesion phenomenon in order to better predict a response variable associated with the network nodes, using both node covariates and network information. While we do not attempt to make causal inferences, we do focus on interpretable models where effects of individual variables can be explicitly estimated.

Using network information in predictive models has not yet been well studied. Most classical predictive models treat the training data as independently sampled from one common population, and, unless explicitly modeled, network cohesion violates virtually all assumptions that provide performance guarantees. More importantly, cohesion is potentially helpful in making predictions, since it suggests pooling information from neighboring nodes. In certain specific contexts, regression with dependent observations has been studied. For example, in econometrics, following the concepts initially discussed by Manski [1993], assuming some type of an autoregressive model on the response variables is common, such as the basic autoregressive model in Bramoullé et al. [2009] and its variants including group interactions and group fixed effects [Lee, 2007]. Such models assume specific forms of different types of network effects, namely, endogenous effects, exogenous effects and correlated effects, and most of this literature is focused on identifiability of such effects. In Bramoullé et al. [2009] and Lin [2010], these ideas were applied to the AddHealth data which we discuss in detail in Section 2.5. However, these methods have mainly been used to identify social effects defined within a very specific and difficult to verify model, without a focus on interpretability or good prediction performance. For instance, including neighbors' responses as covariates in linear regression makes interpretation of other covariate effects more difficult, and can make the distributional assumptions difficult to satisfy. This has been done carefully in spatial statistics literature, for example with the conditional autoregressive model (CAR) [Besag, 1974], but fitting these models typically requires MCMC and is very time-consuming. In addition, these methods do not extend easily beyond linear regression (for example, to generalized linear models and Cox's proportional hazard model).

Our approach is to introduce network cohesion using penalties built using the network information, and framing the problem as loss plus penalty; for simplicity, we will present the method for regression first, and then discuss extensions to general losses. At a high level, our network penalty parallels the ideas of fusion [Land and Friedman, 1997, Tibshirani et al., 2005]. Fusion penalties generally shrink the difference between either coefficients or predictions that are expected to be similar. Fusion penalties based on a network of variables have been used in variable selection [Li and Li, 2008, 2010, Pan et al., 2010, Kim et al., 2013], but this line of work is not directly relevant here since we are interested in using the network of observations, not variables. However, our approach can be viewed as a regression version of the point estimation problem discussed in Sharpnack et al. [2013] and Wang et al. [2016b]. Alternatively, it can be viewed in a Bayesian framework, as regression with a Gaussian Markov random field prior over the network.

We show that our method gives consistent estimates of covariate effects and derive explicit conditions on when enforcing network cohesion in regression can be expected to perform better than ordinary least squares. In contrast to previous work, we assume no specific form for the cohesion effects and require no information about potential groups. We also derive a computationally efficient algorithm for implementing our approach, which is efficient for both sparse and dense networks, the latter with an extra sparsification step which we prove preserves the relevant network properties. To the best of our knowledge, this is the first proposal of a general prediction framework with network cohesion among the observations that is computationally feasible and can retain covariate interpretations as well as make out-of-sample predictions.

The rest of this chapter is organized as follows. In Section 2.2, we introduce our approach in the setting of linear regression. We frame it as a penalized least squares problem which has a closed-form solution, and derive its Bayesian interpretation and connection to various other models. The idea is then extended to generalized linear models. Empirically, we show that our approach outperforms prediction without networks as well as an earlier modification intended to incorporate information from neighbors. Finite sample and asymptotic properties are discussed in Section 2.3. Brief simulation results demonstrating the theoretical bounds and comparisons to benchmarks are presented in Section 2.4. A detailed analysis and discussion of cohesion in the AddHealth data is presented in Section 2.5, where we apply our method to predict recreational activity and marijuana usage among teenagers. All algorithms in this chapter are implemented in the R package **netcoh** [Li et al., 2016a], available on CRAN.

2.2 Regression with network cohesion

2.2.1 Set-up and notation

We start from reviewing the setting up of the network-linked data and notations. The data consist of n observations $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_n, y_n)$, where $y_i \in \mathbb{R}$ is the response variable and $\boldsymbol{x}_i \in \mathbb{R}^p$ is the vector of covariates for observation i. We write $Y = (y_1, y_2, \cdots, y_n)^T$ for the response vector, and $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n)^T$ for the $n \times p$ design matrix. We treat X as fixed and assume its columns have been standardized to have mean 0 and variance 1. We also observe the network connecting the observations, $\mathcal{G} = (V, E)$, where $V = [n]$ is the node set of the graph, and $E \subset V \times V$ is the edge set. We represent the graph by its adjacency matrix $A \in \{0,1\}^{n\times n}$. We assume there are no loops so $A_{vv} = 0$ for all $v \in V$, and we assume the network is undirected, i.e., $A_{ii'} = A_{i'i}$. The (unnormalized) Laplacian of $\mathcal{G}$ is given by $L = D - A$, where $D = \mathrm{diag}(d_1, d_2, \cdots, d_n)$ is the degree matrix, with node degree d_i defined by $d_i = \sum_{i' \in V} A_{ii'}$.

2.2.2 Linear regression with network cohesion

Cohesion is a vague term that can be interpreted in several ways depending on whether it refers to the network itself or both the network and additional covariates. Cohesion defined on the network alone can be reflected in various properties, such as local density, connectivity and community structure; we refer the readers to Chapter 4 of Kolaczyk [2009] for details. In the context of prediction on networks, which is our focus, two types of cohesion are commonly discussed: homophily (also known as assortative mixing) and contagion. Homophily means nodes similar in their characteristics tend to connect, with the implication of a causal direction from sharing individual characteristics to forming a connection. In contrast, contagion means that nodes tend to behave similarly to their neighbors, with a casual direction from having a connection to exhibiting similar characteristics. Distinguishing these two phenomena in an observational study without additional strong assumptions is not possible [Shalizi and Thomas, 2011]. Nonetheless, both of these indicate a correlation between network connections and node similarities, observed empirically by many social behavior studies [Haynie, 2001, Pearson and West, 2003, Fujimoto and Valente, 2012], and that is all we need and assume in this chapter. We use the generic term "cohesion" in order to cover both possibilities of homophily and contagion, which we do not need to distinguish.

The general cohesion penalty idea is simplest to present in the context of linear regression, so we start from this setting. Assume that

$$Y = \boldsymbol{\alpha} + X\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.1}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_n)^T \in \mathbb{R}^n$ is the vector of individual node effects, and $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_p)^T \in \mathbb{R}^p$ is the vector of regression coefficients. At this stage, no assumption on the distribution of the error $\boldsymbol{\epsilon}$ is needed, but we assume $\mathbb{E}\boldsymbol{\epsilon} = \mathbf{0}$ and $\mathrm{Var}(\boldsymbol{\epsilon}) = \sigma^2 I_n$. For simplicity, we further assume that $n > p$ and $X^T X$ is invertible. If $p > n$ and this is not the case, the usual penalties on $\boldsymbol{\beta}$, such as a lasso and ridge, can be applied; our focus here, however, is on regularizing the individual effects, and so we will not focus on additional regularization on $\boldsymbol{\beta}$ that may be necessary.

Including the individual node effects $\boldsymbol{\alpha}$ instead of a common shared intercept turns out to be key to incorporating network cohesion. In general $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, which add up to $n + p$ unknown parameters, cannot be estimated from n observations without additional assumptions. One well-known example of such assumptions is the simple fixed effects model (see e.g. Searle et al. [2009]), when n samples come from K known groups (typically $K \ll n$), and within each group individuals share a common intercept. Here, we regularize the problem through a network cohesion penalty on $\boldsymbol{\alpha}$ instead of making explicit assumptions about any structure in $\boldsymbol{\alpha}$.

The regression with network cohesion (RNC) estimator we propose is defined as the minimizer of the objective function

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \|Y - X\boldsymbol{\beta} - \boldsymbol{\alpha}\|^2 + \lambda \boldsymbol{\alpha}^T L \boldsymbol{\alpha}, \tag{2.2}$$

where $\|\cdot\|$ is the L_2 vector norm and $\lambda > 0$ is a tuning parameter. An equivalent and more intuitive form of the penalty, which follows from a simple property of the graph Laplacian, is

$$\boldsymbol{\alpha}^T L \boldsymbol{\alpha} = \sum_{(i,i')\in E} (\alpha_i - \alpha_{i'})^2. \tag{2.3}$$

Thus, we penalize differences between individual effects of nodes connected by an edge in the network. We call this term the *cohesion penalty* on $\boldsymbol{\alpha}$. We assume that the effect of covariates X is the same across the network; as with any linear regression, two nodes with similar covariates will have similar values of $\boldsymbol{x}^T\boldsymbol{\beta}$, and the cohesion penalty ensures the neighboring nodes have similar individual effects α. Note that this is different from imposing network homophily (which would require nodes with similar covariates to be more likely to be connected).

The minimizer of (2.2) can be computed explicitly (if it exists) as

$$\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}) = (\tilde{X}^T\tilde{X} + \lambda M)^{-1}\tilde{X}^T Y. \tag{2.4}$$

Here, $\tilde{\boldsymbol{X}} = (I_n, X)$ and

$$M = \begin{bmatrix} L & 0_{n\times p} \\ 0_{p\times n} & 0_{p\times p} \end{bmatrix}$$

where $\mathbf{0}_{a\times b}$ is an $a \times b$ matrix of all zeros. The estimator exists if $\tilde{X}^T\tilde{X} + \lambda M$ is invertible. Note that

$$\tilde{X}^T\tilde{X} + \lambda M = \begin{bmatrix} I_n + \lambda L & X \\ X^T & X^TX \end{bmatrix}, \tag{2.5}$$

so it is positive definite if and only if the Schur complement $I_n + \lambda L - X(X^TX)^{-1}X^T = P_{X^\perp} + \lambda L$ is positive definite. From (2.3), we can see that L is positive semi-definite but singular since $L\mathbf{1}_n = 0$ and thus in principle the estimator may not be computable. In Section 2.3, we will give an interpretable theoretical condition for the estimator to exist. In practice, a natural solution is to ensure numerical stability by replacing L with the regularized Laplacian $L + \gamma I$, where γ is a small positive constant. Then the estimator always exists, and in fact the regularized Laplacian may better represent certain network properties, as discussed by Chaudhuri et al. [2012], Amini et al. [2013], Le et al. [2017] and others. The resulting penalty is

$$\sum_{(i,i')\in E} (\alpha_i - \alpha_{i'})^2 + \gamma \sum_i \alpha_i^2, \tag{2.6}$$

which one can also interpret as adding a small ridge penalty on α for numerical stability.

Remark II.1. The penalty (2.6) suggests a natural baseline comparison for our model which can be used to assess whether cohesion is in fact present in the data. If the graph has no edges. i.e., no information about network connections is available, the penalty (with $\gamma = 1$) reduces to a ridge penalty on the individual effects α. The parameter estimates are then obtained by minimizing

$$L_n(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \|Y - X\boldsymbol{\beta} - \boldsymbol{\alpha}\|^2 + \lambda\|\boldsymbol{\alpha}\|^2 \ . \tag{2.7}$$

We call this the *null model* for RNC, as it still incorporates individual node effects which in themselves can improve performance compared to OLS with a common intercept. As discussed later in Section 2.2.4 and 2.2.7, this null model can also be viewed as a random effects model with i.i.d Gaussian intercepts. Comparing the fit of the null model to that of RNC can in fact provide qualitative evidence of cohesion. For linear regression, the null model can improve the fit to training data, but it gives exactly the same estimate of $\boldsymbol{\beta}$ as the OLS (Lemma A.3 in Appendix A), and thus cannot improve predictions on test data, since without network information individual effects on test data cannot be estimated; see more on this in Section 2.4.

Remark II.2. A possible alternative to our cohesion penalty is the network lasso penalty, $\sum_{(i,i')\in E}|\alpha_i - \alpha_{i'}|$ [Hallac et al., 2015]. However, this penalty introduces piecewise constants on the network, a rather stronger assumption than we make about cohesion which may not be always realistic. It is also much more computationally demanding, requiring a sophisticated algorithm and implementation even for moderate size networks.

Remark II.3. It is also possible to assume different but cohesive covariate effects $\boldsymbol{\beta}$ for each individual, which can be implemented in exactly the same way as our idea of the individual intercepts α. As usual, there is a trade-off between including more parameters for better fit and parsimony of the model. We set $\boldsymbol{\beta}$ to be shared among all individual to represent the universal treatment effect, which seems to be reasonable and easy to interpret in many situations.

2.2.3 Network cohesion for general loss functions

The RNC methodology extends naturally to generalized linear models and many other regression or classification models, such as Cox's proportional hazard model [Cox, 1972] for survival analysis, and support vector machines [Vapnik, 2013] for classification using the formulation of Wahba et al. [1999]. Here we will explicitly write out two extensions, to generalized linear models (GLMs) and Cox's model. For any GLM with a link function $\phi(\mathbb{E}Y) = X\boldsymbol{\beta} + \boldsymbol{\alpha}$, where $\boldsymbol{\alpha} \in \mathbb{R}^n$ are the individual effects, suppose the log-likelihood (or partial log-likelihood) function is $\ell(\boldsymbol{\alpha}, \boldsymbol{\beta}; X, Y)$. Then if the observations are linked by a network, to induce network cohesion one can fit the model by maximizing the penalized likelihood

$$\ell(\boldsymbol{\alpha} + X\boldsymbol{\beta}; Y) - \lambda\boldsymbol{\alpha}^T(L + \gamma I)\boldsymbol{\alpha}. \tag{2.8}$$

When ℓ is concave in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, which is the case for exponential families, the optimization problem can be solved via Newton-Raphson or another appropriate convex optimization algorithm. Note that the quadratic approximation to (2.8) is the quadratic approximation to the log-likelihood plus the penalty, and thus the problem can be efficiently solved by iteratively reweighed linear regression with network cohesion, just like the GLM is fitted by iteratively reweighed least squares. The ridge penalty term γI helps with numerical stability and for logistic regression avoids fitted probabilities of 0 and 1 for isolated nodes, which may cause the iterative algorithm to diverge; as discussed in the previous section, adding this term to the Laplacian also improves its representation of the underlying network structure.

RNC can be similarly generalized to Cox's proportional hazard model [Cox, 1972]. In this setting, we observe times until some event occurs, called survival times, which may be censored (unobserved) if the event has not occurred for a particular node. Cox's model assumes the hazard function $h_v(y)$ for each individual v is

$$h_i(y) = h_0(y)\exp(\boldsymbol{x}_i^T\boldsymbol{\beta}), i \in V,$$

where y is the survival time, $\boldsymbol{x}_v$ is the vector of p observed covariates for individual i, $\boldsymbol{\beta} \in R^p$ is the coefficient vector and h_0 is an unspecified baseline hazard function. When we have observations connected by a network, we can model the individual effects and then encourage network cohesion. Thus we will assume the hazard for each node v is given by

$$h_i(y) = h_0(y)\exp(\boldsymbol{x}_i^T\boldsymbol{\beta} + \alpha_i), i \in V, \tag{2.9}$$

where α_i is the individual effect of node i. The appropriate loss function in terms of the parameters $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$ is the partial log-likelihood

$$\ell(\boldsymbol{\theta}; \boldsymbol{i}) = \sum_i \delta_i \Big[\boldsymbol{x}_i^T\boldsymbol{\beta} + \alpha_i - \log\big(\sum_{u: y_{i'} \geq y_i} \exp(\boldsymbol{x}_{i'}^T\boldsymbol{\beta} + \alpha_{i'})\big)\Big] \tag{2.10}$$

where y_i is the observed survival time for node i, and δ_i is the censoring indicator, which is 0 if the observation is right-censored and 1 otherwise. Note that the partial log-likelihood is invariant under a shift in $\boldsymbol{\alpha}$ since such a shift can always be absorbed into h_0. Thus for identifiability, we require $\sum \alpha_i = 0$. For fixed covariates $\boldsymbol{x}_i$, α_i is the individual deviation from the population average log-hazard. The sum-to-zero constraint can be automatically enforced by replacing the network Laplacian L in the network cohesion penalty with its regularized version $L + \gamma I$, or equivalently adding a ridge penalty on α's. Thus we maximize the following objective function, adding a regularized cohesion penalty to the partial log-likelihood:

$$\ell(\boldsymbol{\theta}) - \lambda\boldsymbol{\alpha}^T(\boldsymbol{L} + \gamma I)\boldsymbol{\alpha}.$$

2.2.4 A Bayesian interpretation

The RNC estimator can also be framed as a Bayesian regression model. Consider the model

$$Y|\boldsymbol{\alpha}, \boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\alpha} + X\boldsymbol{\beta}, \sigma^2 I), \quad \boldsymbol{\beta} \sim \pi_{\boldsymbol{\beta}}(\phi), \quad \boldsymbol{\alpha} \sim \pi_{\boldsymbol{\alpha}}(\Phi),$$

where $\pi_{\boldsymbol{\beta}}(\phi)$ is the prior for $\boldsymbol{\beta}$ with hyperparameter ϕ, $\pi_{\boldsymbol{\alpha}}(\Phi)$ is the prior for $\boldsymbol{\alpha}$ with hyperparameter Φ, and σ^2 is assumed to be known. Suppose we take $\pi_{\boldsymbol{\beta}}(\phi)$

to be the non-informative Jeffrey's prior, reflecting lack of prior knowledge about the coefficients, and set $\pi_{\boldsymbol{\beta}}(\phi) \propto 1$. For $\boldsymbol{\alpha}$, assume a Gaussian Markov random field (GMRF) prior $\pi_{\boldsymbol{\alpha}} = \mathcal{N}_{\mathcal{G}}(\mathbf{0}, \Phi)$, where $\Phi = \Omega^{-1} = \zeta^2(L + \gamma I)^{-1}$. Note that when $\gamma = 0$, Ω is not invertible, and $\pi_{\boldsymbol{\alpha}}$ is an improper prior called intrinsic GMRF [Rue and Held, 2005].

If the posterior modes are used as the estimators for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, then this is equivalent to (2.2) with $\lambda = \sigma^2/\zeta^2$ and the Laplacian replaced by the regularized Laplacian $L+\gamma I$. Thus the estimator of (2.2) is the Bayes estimator with the improper intrinsic GMRF prior over the network on $\boldsymbol{\alpha}$. Note that this Bayesian interpretation is also valid for the generalized linear models.

2.2.5 Prediction and choosing the tuning parameter

To compute fitted values on the training data (in-sample prediction), we simply use $\hat{\boldsymbol{\alpha}} + X\hat{\boldsymbol{\beta}}$. The out-of-sample prediction task in this setting is to make predictions on a group of new subjects whose covariates as well as network connections (but not responses) become available after the model is fitted on training data. Since we have a different α_v for each node v, predicted individual effects are needed for new samples. Suppose we have a total of n training samples and n' test samples, resulting in a new network with $n + n'$ nodes where the first n nodes are from training and the last n' are the test nodes. Write the associated Laplacian as

$$L' = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix},$$

where L_{11} corresponds to the original n training samples and L_{22} corresponds to the n' test samples. Similarly write the individual effect vector as $(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2)$, where $\boldsymbol{\alpha}_1 = \hat{\boldsymbol{\alpha}}$ is estimated from training data, and $\boldsymbol{\alpha}_2$ needs to be predicted.

To take advantage of cohesion, we predict $\boldsymbol{\alpha}_2$ by minimizing the overall cohesion penalty, letting

$$\hat{\boldsymbol{\alpha}}_2 = \arg\min_{\boldsymbol{\alpha}_2} (\hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha}_2)^T L' (\hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha}_2) \ .$$

This gives

$$\hat{\boldsymbol{\alpha}}_2 = -L_{22}^{-1} L_{21} \hat{\boldsymbol{\alpha}}.$$

This corresponds to a supervised prediction setting, our focus in this chapter, which assumes only the training data are available at the time of fitting. Our method can also be used in a semi-supervised setting, where the entire network is available at the time of training. In this case, the cohesion penalty at the fitting stage can include all the individual effects for all data points and the entire network so $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ are jointly optimized simultaneously.

The tuning parameter λ can be selected by cross-validation. Randomly splitting or sampling from a network is not straightforward; however, we found that the usual "naive" cross-validation finds very good tuning parameters for our method, perhaps because it is fundamentally a regression problem and we are not attempting to make any inferences about the structure of the network. We tune using regular 10-fold cross-validation, randomly splitting the samples into 10 folds, leaving each fold out in turn, and training the model using the remaining nine folds and the corresponding induced subnetwork. The cross-validation error is computed as the average of the prediction errors on the fold that was left out, and the tuning parameter is picked to minimize the cross-validation error.

2.2.6 An efficient computation strategy

Computing the estimator (2.4) involves solving a $(n + p) \times (n + p)$ linear system so a naive implementation would require $O((n + p)^3)$ operations. For GLMs, such a system has to be solved in each Newton step. This computational burden can be reduced significantly by taking advantage of the fact that most networks in practice have sparse adjacency matrices as well as sparse Laplacians, which allows for using block elimination. A general description of this strategy can be found in many standard texts (see e.g. Boyd and Vandenberghe [2004], Ch. 4). Here we give the details in our setting.

The linear system we need to solve is

$$(\tilde{X}^T\tilde{X} + \lambda M)\boldsymbol{a} = \boldsymbol{b}.$$

From (2.5), we can rewrite this system with the following block structure:

$$\begin{bmatrix} I+\lambda L & X \\ X^T & X^TX \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}.$$

The top row gives

$$(I+\lambda L)\boldsymbol{a}_1 = (\boldsymbol{b}_1 - X\boldsymbol{a}_2)$$

and substituting this into the bottom row, we have

$$(X^TX - X^T(I+\lambda L)^{-1}X)\boldsymbol{a}_2 = \boldsymbol{b}_2 - X^T(I+\lambda L)^{-1}\boldsymbol{b}_1.$$

Note that $I+\lambda L$ is a symmetric diagonal dominant (SDD) matrix, and is sparse most of the time in practice, so $(I+\lambda L)^{-1}\boldsymbol{b}_1$ and $(I+\lambda L)^{-1}X$ can be efficiently computed [Koutis et al., 2010, Cohen et al., 2014]. The cost of this step is roughly $O(p(n+2|E|)(\log n)^{1/2})$, where $|E|$ is the number of edges in the network and c is some absolute constant. The cost of the remaining computations is dominated by the cost of inverting the $p \times p$ matrix $X^TX - X^T(I+\lambda L)^{-1}X$, which is of the same order as the cost of solving a standard least squares problem.

When A and L are dense matrices, with $|E| = O(n^2)$, the strategy above has the cost of $O(pn^2((\log n)^{1/2})$, which is still better than naively solving the system, but we do not gain anything from block elimination unless L is sparse. However, we can first apply a graph sparsification algorithm to A and use the sparsified A^* as input for RNC. For instance, the algorithm of Spielman and Teng [2011] can find A^* with $O(\epsilon^{-2} n \log n)$ edges at the cost of $O(|E| \log^2 n)$ operations such that its sparsified Laplacian L^* satisfies

$$(1-\epsilon)L \preceq L^* \preceq (1+\epsilon)L,$$

for a given constant $\epsilon > 0$. After this sparsification step, the complexity of solving the linear system reduces to to $O(pn \log^c n)$ for $c \leq 3$. In Section 2.3, we will provide theoretical guarantees for the accuracy of the RNC estimator based on L^* compared to that based on L.

When the number of edges is on the order of $O(n^2)$, the sparsification step itself has complexity of $O(n^2 \log^c n)$, which is not necessarily cheaper than directly solving the original dense linear system using the SDD property. However, the advantage of sparsification becomes obvious when one has to iteratively solve the linear system for the GLM or Cox's model, and/or compute a solution path for a sequence of λ values. In such situations, sparsificaiton only has to be done once and the average complexity of solving the linear system can be close to $O(n \log^c n)$ for the whole estimation procedure. Details of complexity calculations for the RNC are given in Appendix A.2; a more comprehensive discussion of the computational trade-off of sparsification can be found in Sadhanala et al. [2016].

2.2.7 Connection to other models

Fixed group effects models The fixed effects regression model with subjects divided into groups is a special case of RNC. If the graph $\mathcal{G}$ represents the groups as cliques (everyone within the same group is connected), there are no connections between groups, and we let $\lambda \to \infty$, then all nodes in one group will share a common intercept.

Mixed effects models. A mixed model, like ours, has individual effects viewed as random ($\boldsymbol{\alpha}$) and fixed covariate effects ($\boldsymbol{\beta}$), but no network effects. Our null model is a standard mixed model. The Bayesian interpretation of our method suggests we are inducing correlations between the random effects, $\boldsymbol{\alpha} \sim \mathcal{N}_{\mathcal{G}}(0, \Phi)$. The estimator (2.4) is then the mixed model equation in Henderson [1953] for estimating fixed effects and predicting random effects simultaneously (see Searle et al. [2009]). However, the framework of mixed models requires stronger assumptions on the form of variance components. Moreover, (generalized) mixed models are not designed for predictions conceptually, and we will show in the simulation study as well as theoretically in Lemma A.3 in Appendix A that the null model is not able to improve on out-of-sample predictions.

Spatial models In spatial statistics, data points are typically indexed by their locations. A weight matrix A can be computed as a function of distance between locations and can be used as a weighted analogue of our network adjacency matrix. This leads to natural connections between RNC and methods used in spatial statistics. In particular, ignoring the covariates X, RNC reduces to the Laplacian smoothing point estimation procedure in Sharpnack et al. [2013] and Wang et al. [2016b], which is equivalent to krigging in spatial statistics [Cressie, 1990]. It has been shown that a class of semi-supervised learning methods based on Laplacian smoothing can be viewed as "graph krigging" [Xu et al., 2010] . From this perspective, RNC can be viewed as a generalization of graph krigging of Xu et al. [2010] to incorporate covariates and general loss functions. With covariates X included, the Bayesian interpretation of RNC assumes the same Gaussian Markov random field distribution for α as the conditional autoregressive model (CAR) [Besag, 1974] and its GLM generalization (Chapter 9 of Waller and Gotway [2004]) assume for errors in spatial regression. However, ζ^2 and σ^2 in our Bayesian interpretation are treated as parameters in the CAR, while $\lambda = \sigma^2/\zeta^2$ is treated as a tuning parameter in RNC. Further, the CAR model is fitted either by maximum likelihood involving computationally expensive integration steps, or by posterior inference via Markov chain Monte Carlo after assuming a full Bayesian model with additional priors on $\boldsymbol{\beta}$ and ζ^2, etc. Both ways require much heavier computations than RNC, especially for GLM where the Gaussian Markov random field is no longer the conjugate prior. More importantly, CAR models cannot be applied to general loss functions that are not a well-defined likelihood, for example, for Cox's model and SVM. Also, CAR models suffer from conceptual difficulties in making out-of-sample predictions [Waller and Gotway, 2004]. In contrast, RNC provides a universal strategy under general loss functions and comes with a natural out-of-sample predictor, discussed in Section 2.2.5.

Manifold embeddings Our Laplacian-based penalty has connections to the large literature on manifold embeddings and semi-supervised learning. The general task of manifold embeddings is to embed data points, typically observed in some high-dimensional space equipped with a potentially non-Euclidean similarity measure, into a low-dimensional Euclidean space, while preserving dissimilarity between the points as much as possible. Finding the "right" embedding space is expected to help with downstream analysis tasks, such as visualization [Tenenbaum et al., 2000] or clustering [Shi and Malik, 2000]. Perhaps the algorithm most closely related to ours is Laplacian Eigenmaps [Belkin and Niyogi, 2003], which proposed using k eigenvectors of the constructed graph Laplacian L corresponding to the smallest eigenvalues as the Euclidean embedding of the graph in order to obtain a low-dimensional representation of the data, and its kernel version with a regularization penalty [Belkin et al., 2006]. There are multiple semi-supervised learning approaches to prediction on manifolds, where it is assumed that all the similarities (corresponding to the network in our case) are observed but only some of the data points are labelled [Zhou et al., 2004, 2005]. Later out-of-sample extensions [Bengio et al., 2004, Cai et al., 2007, Vural and Guillemot, 2016] were developed by assuming the embedding coordinates take certain specific forms as functions of the original data points, and in general the manifold literature relies on an underlying Euclidean space where distance and smoothness are well defined, an assumption we do not make.

Supervised manifold embeddings have also been proposed when class labels are available in training data, including for the Laplacian Eigenmaps [Yang et al., 2011, Raducanu and Dornaika, 2012, Vural and Guillemot, 2016]. The basic idea is to learn a low-dimensional embedding of the data that also corresponds to a good separation of classes, and then use the coordinates in this embedding as predictors instead of the original variables. For general response variables instead of class labels, there is no supervised variant of Laplacian Eigenmaps as far as we are aware. More importantly, the embedding coordinates are typically complicated implicit functions of all the variables, and their coefficients cannot be interpreted in any meaningful way. Our method, on the other hand, has the original variables as predictors in the model (and nothing else), and thus their regression coefficients are readily interpretable.

2.3 Theoretical properties of the RNC estimator

Recall the RNC estimator is given by

$$\hat{\boldsymbol{\theta}} = (\tilde{X}^T\tilde{X} + \lambda M)^{-1}\tilde{X}^T Y, \tag{2.11}$$

where

$$M = \begin{bmatrix} L & 0 \\ 0 & 0 \end{bmatrix}.$$

We continue to assume that X has centered columns and full column rank. Intuitively, we expect the network cohesion effect to improve prediction only when the network provides "new" information that is not already contained in the predictors X. We formalize this intuition in the following assumption:

Assumption II.4. *For any $\boldsymbol{u} \neq 0$ in the column space of X, $\boldsymbol{u}^T L\boldsymbol{u} > 0$.*

This natural and fairly mild assumption is enough to ensure the existence of the RNC estimator. Write $\text{col}(X)$ for the linear space spanned by columns of X and $\text{col}(X)^{\perp}$ for its orthogonal complement. Then the projection matrix onto $\text{col}(X)^{\perp}$ is $P_{X^{\perp}} = I_n - P_X$, where $P_X = X(X^TX)^{-1}X^T$. Write $\lambda_{\min}(M)$ for the minimum eigenvalue of any matrix M. Then we have the following lemma:

Proposition II.5. *Whenever $\lambda > 0$, we have $0 \le \nu = \lambda_{\min}(P_{X^{\perp}} + \lambda L) \le 1$. Under Assumption II.4 the RNC estimator* (2.11) *exists.*

Lemma II.5 in the Appendix shows that when the network is connected and X is centered, the RNC estimator always exists since in a connected graph, L has rank $n-1$, and an eigenvector $\mathbf{1}$.

Theorem II.6. *Under Assumption II.4, the RNC estimator $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})$ defined by*

(2.11) *satisfies*

$$\begin{aligned}
&(2.12) & \mathrm{MSE}(\hat{\boldsymbol{\alpha}}) &\leq \frac{\lambda^2}{\nu^2}\|L\boldsymbol{\alpha}\|^2 + \frac{n}{\nu}\sigma^2, \\
&(2.13) & \mathrm{MSE}(\hat{\boldsymbol{\beta}}) &\leq \frac{\lambda^2}{\nu^2\mu}\|L\boldsymbol{\alpha}\|^2 + \sigma^2(\frac{1}{\nu}+1)\mathrm{tr}((X^TX)^{-1}), \\
&(2.14) & \mathbb{E}\|\hat{Y} - \mathbb{E}Y\|^2 &\leq \frac{\lambda^2}{\nu}\|L\boldsymbol{\alpha}\|^2 + \sigma^2\|S_\lambda\|_F^2,
\end{aligned}$$

where the minimum eigenvalue of X^TX is denoted by μ and $\|S_\lambda\|_F$ is the Frobenius norm of the shrinkage matrix $S_\lambda = \tilde{X}(\tilde{X}^T\tilde{X}+\lambda L)^{-1}\tilde{X}^T$. In particular, when $\|L\boldsymbol{\alpha}\| = 0$, and therefore $\boldsymbol{\alpha}$ is constant over each connected component of the network, RNC is unbiased.

The proof is given in the Appendix where the expressions for exact errors are also available. Theorem II.6 applies to any fixed n. The asymptotic results as the size of the network n grows are presented next in Theorem II.7. We add the subscript n to previously defined quantities to emphasize the asymptotic nature of this result.

Theorem II.7. *If Assumption II.4 holds, $\mu_n = O(n)$, $\|L_n\boldsymbol{\alpha}_n\|^2 = o(n^c)$ for some constant $c < 1$, and there exists a sequence of λ_n and a constant $\rho > 0$ such that $\liminf_n \nu_n > \rho$, then*

$$\mathrm{MSE}(\hat{\boldsymbol{\beta}}) \leq O(\lambda_n^2 n^{-(1-c)}) + O(n^{-1}).$$

Therefore if $\lambda_n^2 = o(n^{1-c})$, $\hat{\boldsymbol{\beta}}$ is an L_2-consistent estimator of $\boldsymbol{\beta}$.

Remark II.8. Note that the quantity $L\boldsymbol{\alpha}$ appearing in the assumptions is the gradient of the cohesion penalty with respect to $\boldsymbol{\alpha}$, $\nabla_{\boldsymbol{\alpha}}\boldsymbol{\alpha}^TL\boldsymbol{\alpha} = 2L\boldsymbol{\alpha}$. We call $L\boldsymbol{\alpha}$ the cohesion gradient. In physics, cohesion gradient is used to measure heat diffusion on graphs when $\boldsymbol{\alpha}$ is a heat function:

$$(L\boldsymbol{\alpha})_i = |\mathrm{ne}(i)|\left(\alpha_i - \frac{\sum_{i'\in\mathrm{ne}(i)}\alpha_{i'}}{|\mathrm{ne}(i)|}\right).$$

where ne(i) is the set of neighbors of i defined by the graph. Thus $\|L\boldsymbol{\alpha}\|$ represents the difference between nodes' individual effects and the average of their neighbors' effects. The condition of Theorem II.7 requires that the norm of the vector $L\boldsymbol{\alpha} \in \mathbb{R}^n$ grows slower than $O(\sqrt{n})$. This condition is satisfied by a large set of $n-$dimensional vectors defined on many networks; the following proposition gives an example.

Proposition II.9. *Assume the network is a $\sqrt{n} \times \sqrt{n}$ lattice. Then $\|L\boldsymbol{\alpha}\|^2 \leq n^c$ as long as $\boldsymbol{\alpha}$ is in the subspace spanned by k smallest eigenvalues of L for some $k \leq Cn^{\frac{1+c}{2}}$, where C and c are some constants and $c < 1$.*

It is instructive to compare the MSE of our estimator with the MSE of the ordinary least squares (OLS) estimator, as well as the null model (which is what our estimator gives when the network has no edges). For OLS, we have

$$\hat{\boldsymbol{\beta}}_{OLS} = (X^TX)^{-1}X^TY, \ \hat{\boldsymbol{\alpha}}_{OLS} = \bar{y}\mathbf{1},$$

where $\hat{\boldsymbol{\alpha}}_{OLS}$ is the common intercept. Compared to OLS, the RNC estimator reduces bias caused by the network-induced dependence among samples, and as a trade-off increases variance; thus intuitively, one would expect that the signal-to-noise ratio and the degree of cohesion in the network will determine which estimator performs better. From Theorem II.6 and the basic properties of the OLS estimator (stated as Lemma A.1 in the Appendix), it is easy to see that if

$$\left(\frac{n}{\nu} - 1\right)\sigma^2 \leq V(\boldsymbol{\alpha}) - \frac{\lambda^2}{\nu^2}\|L\boldsymbol{\alpha}\|^2 \tag{2.15}$$

where $V(\boldsymbol{\alpha}) = \sum_i(\boldsymbol{\alpha}_i - \bar{\boldsymbol{\alpha}})^2$, then the RNC estimator of the individual effects $\hat{\boldsymbol{\alpha}}$ has a lower MSE than that of $\hat{\boldsymbol{\alpha}}_{OLS}$. The left hand side of (2.15) represents the increase in variance induced by adding the individual effects, whereas the right hand size is the corresponding reduction in squared bias. When $\boldsymbol{\alpha}$ is sufficiently smooth over the network, $\|L\boldsymbol{\alpha}\|$ is negligible compared to other terms, and the condition essentially requires that the total variation of α_v around its average is larger than the total noise level. Similarly, for the coefficients β, if

$$\operatorname{tr}((X^TX)^{-1})\frac{\sigma^2}{\nu} \leq \|(X^TX)^{-1}X^T\boldsymbol{\alpha}\|^2 - \frac{\lambda^2}{\mu}\|L\boldsymbol{\alpha}\|^2 \qquad (2.16)$$

then the RNC estimator $\hat{\boldsymbol{\beta}}$ has a lower MSE than $\hat{\boldsymbol{\beta}}_{OLS}$. Again, the two sides of the inequality represent the increase in variance and the reduction in squared bias, respectively. The null model gives an estimate for $\boldsymbol{\beta}$ identical to $\hat{\boldsymbol{\beta}}_{OLS}$, so the same comparison applies. The null model estimate of $\boldsymbol{\alpha}$ involves more terms and the corresponding tuning parameter and does not result in clear comparison. However, we demonstrate the difference numerically by the next example and by our simulation study in Section 2.4. The exact formula for the null model estimation error is given by Lemma A.3 in Appendix A.

Example II.10. We illustrate the bias-variance trade-off on a simple example. Suppose we have a network with $n = 300$ nodes which consists of three disconnected components G_1, G_2, G_3, of 100 nodes each. Each component is generated as an Erdos-Renyi graph, with each pair of nodes forming an edge independently with probability 0.05. Individual effects α_i are generated independently from $\mathcal{N}(\eta_{c_i}, 0.1^2)$, where $c_i \in \{1, 2, 3\}$ is the component to which nodes i belongs, $\eta_1 = -1$, $\eta_2 = 0$, $\eta_3 = 1$. We set $\lambda = 0.1$. Substituting the expectation $\mathbb{E}A$ for A, we have $\nu \approx 0.5$, $\|L\boldsymbol{\alpha}\|^2 \approx 105$, and $V(\boldsymbol{\alpha}) \approx 203$. Then as long as the noise variance $\sigma < 0.57$, (2.15) will be satisfied. Similarly, $X^TX \approx nI_2$, and $\|X^T\boldsymbol{\alpha}\|^2 \approx 406$ in expectation. Thus (2.16) holds and the RNC is beneficial if $\sigma < 0.54$ (approximately). The bias-variance trade-off in the mean squared prediction errors (MSPE) can be demonstrated explicitly when varying λ; Figure 2.1 shows this trade-off between bias and variance together with the OLS baseline when $\sigma = 0.5$. The MSPEs of OLS and the null model are also shown. Note that this calculation for RNC is based on conservative bounds. In reality the RNC is going to be beneficial for a larger range of σ values.

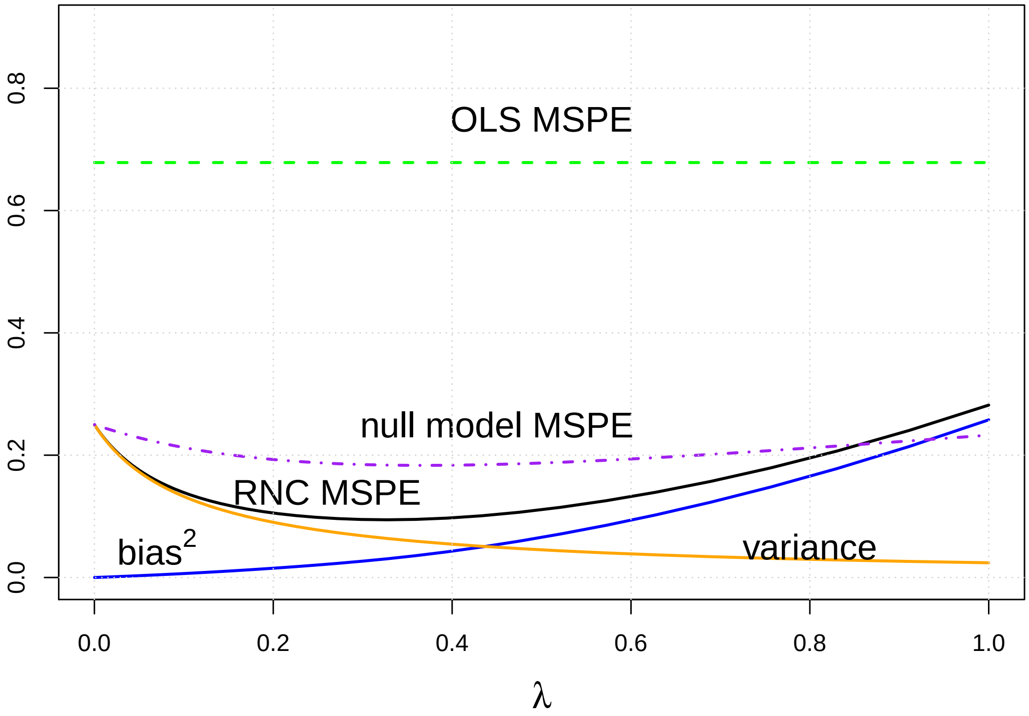


Figure 2.1: Mean squared prediction error $\mathbb{E}\|\hat{Y} - \mathbb{E}Y\|^2/n$ and the bias-variance trade-off of the RNC estimator (based on the upper bound (2.14) in Theorem II.6), in the setting of Example II.10 with $\sigma = 0.5$.

Remark II.11. If we use (2.6) and are willing to make strong assumptions about the distribution as in the Bayesian interpretation, it can be shown (see Searle et al. [2009], Ch. 7 for details) that $\hat{\boldsymbol{\alpha}}$ is the best linear unbiased predictor (BLUP) of $\boldsymbol{\alpha}$ and $\hat{\boldsymbol{\beta}}$ is the best linear unbiased estimator (BLUE) of β. However, these are strong assumptions which we prefer to avoid.

Finally, we investigate the effects of graph sparsification, proposed in Section 2.2.6 to reduce computational cost. For any $\epsilon > 0$, let L^* be the Laplacian of a network on the same nodes satisfying

$$(1-\epsilon)L \preceq L^* \preceq (1+\epsilon)L. \tag{2.17}$$

In addition, let $\hat{\boldsymbol{\theta}}$ be the minimizer of

$$f(\boldsymbol{\theta}) = \ell(\boldsymbol{\alpha} + X\boldsymbol{\beta}; Y) + \lambda\boldsymbol{\alpha}^T L\boldsymbol{\alpha}, \tag{2.18}$$

and $\hat{\boldsymbol{\theta}}^*$ be the minimizer of

$$f^*(\boldsymbol{\theta}) = \ell(\boldsymbol{\alpha} + X\boldsymbol{\beta}; Y) + \lambda\boldsymbol{\alpha}^T L^*\boldsymbol{\alpha}, \tag{2.19}$$

where ℓ can be a general loss function, such as the sum of squared errors in linear model or the negative log-likelihood in GLM.

Theorem II.12. *Given two Laplacians L and L^* satisfying* (2.17) *for $0 < \epsilon < 1/2$, assume ℓ in* (2.18) *is twice differentiable and f is strongly convex with $m > 0$, such that for any $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{R}^{n+p}$,*

$$\bigtriangledown^2 f(\boldsymbol{\theta}) \succeq mI_{n+p}.$$

Then $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}^*$ minimizing (2.18) *and* (2.19) *respectively, with the same λ, satisfy*

$$\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \leq \frac{2\epsilon\lambda}{m}\min\Big(2\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}} + |\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}} - \hat{\boldsymbol{\alpha}}^{*T} L^*\hat{\boldsymbol{\alpha}}^*| + 2\epsilon\hat{\boldsymbol{\alpha}}^{*T} L^*\hat{\boldsymbol{\alpha}}^*\ ,\ \frac{2\epsilon\lambda}{m}\lambda_1(L)^2\|\hat{\boldsymbol{\alpha}}\|^2\Big). \tag{2.20}$$

The proof is given in the Appendix. Theorem II.12 can be seen as a generalization of the result of Sadhanala et al. [2016] for point estimation by Laplacian smoothing (or kriging) for Gaussian and binary data. Our bound is slightly better than that of Sadhanala et al. [2016].

Remark II.13. The term $\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}}$ is the cohesion penalty and is expected to be small for estimated $\hat{\alpha}$. Further, we can expect both $|\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}} - \hat{\boldsymbol{\alpha}}^{*T} L^*\hat{\boldsymbol{\alpha}}^*|$ and $\epsilon\hat{\boldsymbol{\alpha}}^{*T} L^*\hat{\boldsymbol{\alpha}}^*$ to be much smaller than $\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}}$, and the first bound in (2.20) is typically much smaller than the second. Therefore, the bound is essentially

$$\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \lesssim \frac{4\epsilon\lambda}{m}\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}}. \tag{2.21}$$

Remark II.14. The theorem shows that the squared error in estimation with an ϵ-approximated Laplacian is decreasing linearly in ϵ. In particular, it is easy to check that for the linear regression case, we have

$$\bigtriangledown^2 \ell(\boldsymbol{\theta}) = 2(\tilde{X}^T\tilde{X} + \lambda M).$$

Strong convexity always holds whenever RNC estimate exists, and the bound becomes

$$\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \lesssim \frac{2\epsilon\lambda\hat{\boldsymbol{\alpha}}^T L\hat{\boldsymbol{\alpha}}}{\lambda_n(\tilde{X}^T\tilde{X} + \lambda M)}. \tag{2.22}$$

Remark II.15. Theorem II.12 can also be viewed as a result on network misspecification. If the true network is observed with errors, but its Laplacian L^* satisfies (2.17) and is close enough to the correct L, we have the same error bound for the estimate from the mispecified network. Another way to make the method more robust to errors in the network is to replace L by a low-rank approximation to it, if we have reasons to believe a low-rank structure describes the underlying network well.

2.4 Numerical performance evaluation

In this section, we investigate the effects of including network cohesion on simulated data, in linear and logistic regression.

The simulated networks are generated from the SBM with $n = 300$ nodes and $K = 3$ blocks. Under the stochastic block model, the nodes are assigned to blocks independently by sampling from a multinomial distribution with parameters $(\pi_1, \dots, \pi_K)$. We set $\pi_1 = \pi_2 = \pi_3 = 1/3$, $B_{kk} = p_w = 0.2$, and $B_{kl} = p_b$ for all $k \neq l$. As in Example II.10, the individual effects α_i's are generated independently from a normal distribution with the mean determined by the node's block label, $\mathcal{N}(\eta_{c_i}, s^2)$, where $\eta_1 = -1$, $\eta_2 = 0$, $\eta_3 = 1$, and the parameter s controls how "cohesive" the α_i's within each block are. The predictor coefficients $\boldsymbol{\beta}$ are drawn independently from $\mathcal{N}(1, 1)$.

This simulation setting is not especially favorable to RNC since it does not satisfy the smoothness requirement of Theorem II.7 except when $s = 0$. Moreover, because edges connecting different blocks give false information and edges within the same block are all exchangeable, an edge between two nodes does not give direct evidence of them having similar α's (except when $p_b = 0$). However, there is cohesion on the network in the sense that some $alpha$s are more similar to each other than to others, and we can vary the strength of cohesion by varying s; varying p_b allows us to test robustness against "false" edges, meaning edges that do not indicate similarity.

We compare RNC to four other methods on these simulated networks: a baseline (OLS for continuous response and logistic regression for binary response), the null model, where the graph is empty and we simply add a ridge penalty on the individual effects, a fixed effects "oracle" model which knows the true blocks and uses the same α for all the nodes in the same block, and a mixed effects model which adds Gaussian random effects to the fixed effect model, fitting exactly the model that was used to generate the data. The tuning parameters are always selected by 10-fold cross-validation; however, the linear null model always makes the same out-of-sample predictions as OLS (Lemma A.3 in the Appendix), for any value of λ, and thus cross-validation cannot be used to select the tuning parameter. This is a side effect of the bigger problem for the null model, which is its inability to make non-trivial out-of-sample predictions. Instead of cross-validation, we use the restricted maximum likelihood (REML) estimate under the corresponding linear mixed model framework for $\lambda = \sigma^2/\zeta^2$. The mixed effects model is also estimated by REML.

Four performance metrics are reported: the average mean squared error (MSE) of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and in-sample and out-of-sample mean squared prediction errors (MSPEs). Figure 2.2 shows results as the variance parameter s changes from 0 to 1 with $p_b = 0.02$. All methods get worse as s increases and the signal-to-noise ratio goes down, as one would expect. The OLS is the worst on all measures since the other models incorporate the individual effects α and thus provide a better fit. However, incorporating α in the null model only helps with the in-sample error; for estimating β and out-of-sample prediction, the null model is exactly the same as OLS. The RNC and the two oracle models generally perform much better and are fairly close to each other, with the oracle fixed effects model performing somewhat better on the in-sample error when s is small and the oracle is close to the true model, and both the RNC and oracle mixed effects model outperforming the oracle fixed effects model for larger s since they can adapt to the changing amount of cohesion over the network. Instead of using known blocks we could have also fitted them by one of the many available community detection methods, but that would only help if the underlying model does indeed have communities. The RNC, on the other hand, does not require an assumption of communities and can adapt to cohesion over many different types of underlying graphs.

Figure 2.3 shows how the four performance metrics respond to an increase in p_b, the probability of “false” edges, with fixed $s = 0.1$. As expected, the performance

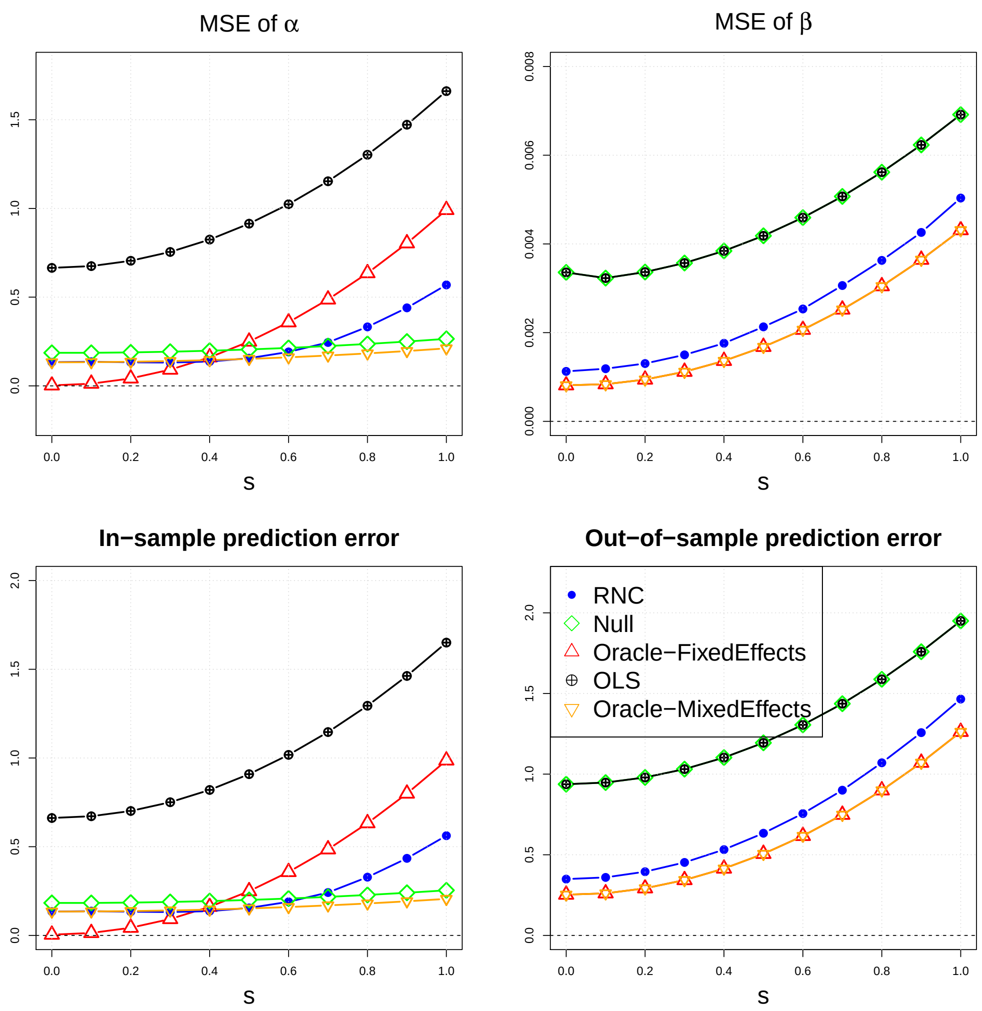


Figure 2.2: Linear regression with varying s and $p_b = 0.02$. Performance is evaluated by the MSEs of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and in-sample and out-of-sample mean squared prediction errors.

of RNC degrades as p_b increases. However, even when $p_b = 0.05$, when the ratio of within-block "true" edge probability to between-block "false" edge probability is only 4/3, RNC still does much better than OLS and the null model in estimating $\boldsymbol{\beta}$ and out-of-sample prediction.

Next, we use the same setting for generating the network, covariates, and parameters, but generate $\boldsymbol{Y}$ from the Bernoulli distribution with probabilities of success given by the logit function of $X^T\boldsymbol{\beta} + \boldsymbol{\alpha}$. We then estimate the parameters by fitting standard logistic regression and also logistic regression with our proposed network cohesion penalty. We fix a small value of the ridge regularization tuning parameter, $\gamma = 0.01$, as it is only added for numerical stability. We evaluate the methods by computing the average MSE of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and the vector of n Bernoulli probabilities estimated as

$$\hat{p}_i = \frac{\exp(\boldsymbol{x}_i^T \hat{\boldsymbol{\beta}} + \hat{\alpha}_i)}{1 + \exp(\boldsymbol{x}_i^T \hat{\boldsymbol{\beta}} + \hat{\alpha}_i)},$$

as well as the probabilities on 50 hold-out samples. The latter two are analogues to in-sample and out-of-sample prediction errors in linear regression.

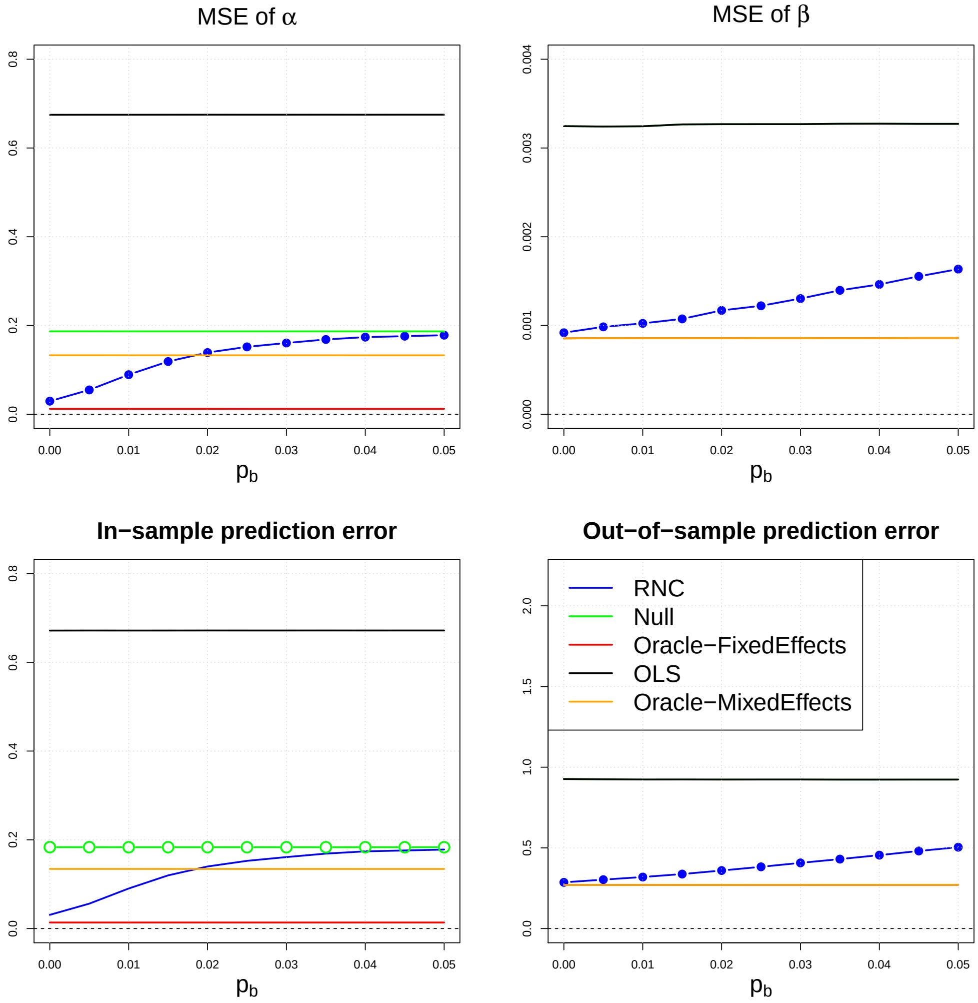


Figure 2.3: Linear regression with varying p_b and $s = 0.1$. Performance is evaluated by the MSEs of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and in-sample and out-of-sample mean squared prediction errors.

Figure 2.4 shows the average MSE of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and in-sample and out-of-sample probabilities as s varies. The general pattern remains similar to linear regression. Although in this case the null model is no longer identical to regular logistic regression, it still gives nearly the same out-of-sample error. The oracle mixed effects model is the best, as it assumes the true model. In general, all methods deteriorate with increasing s, and while the logistic RNC does not perform quite as well as the oracle, it gets much closer to it than any other method. Figure 2.5 shows the metrics when varying p_b from 0 to 0.05 with fixed $s = 0.1$. Again, the RNC outperforms regular logistic regression and the null model even when $p_b = 0.05$.

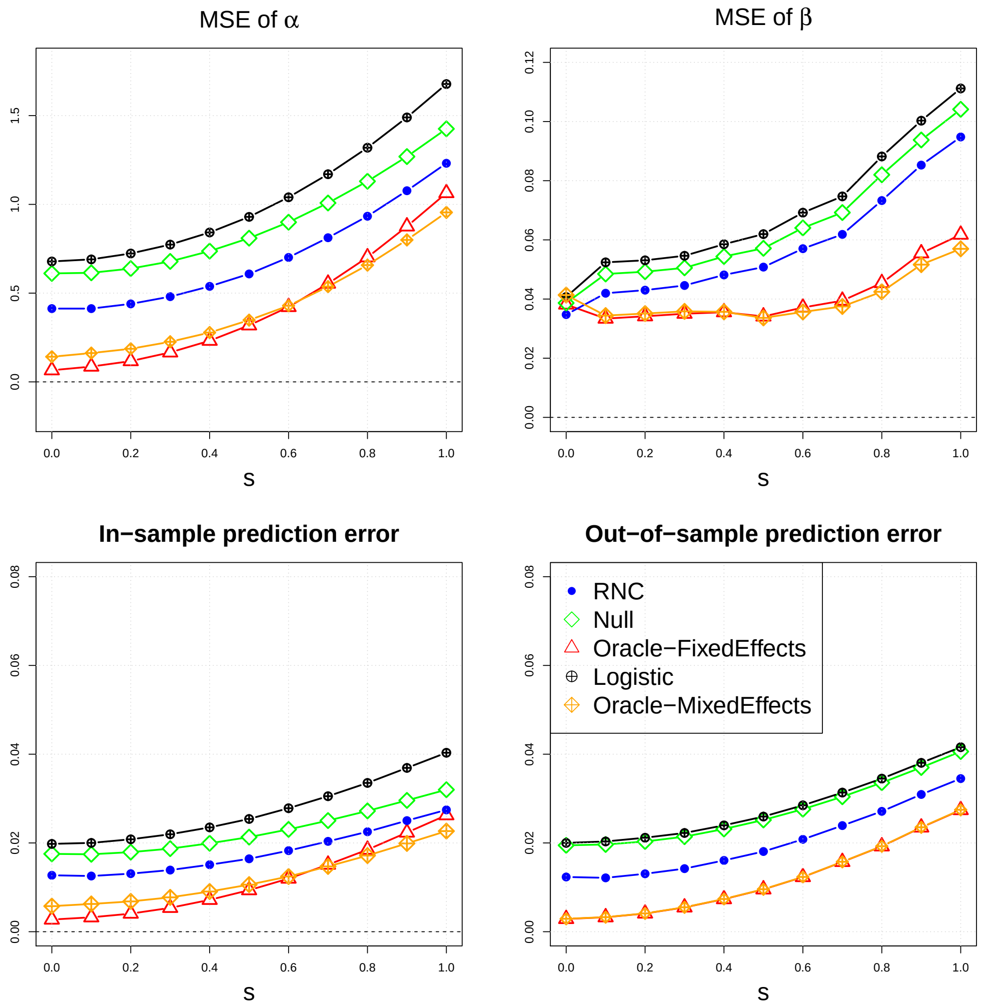


Figure 2.4: Performance logistic regression methods when varying s and fixing $p_b = 0.02$, measured by the MSE of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, in-sample and out-of-sample mean squared probability estimation errors.

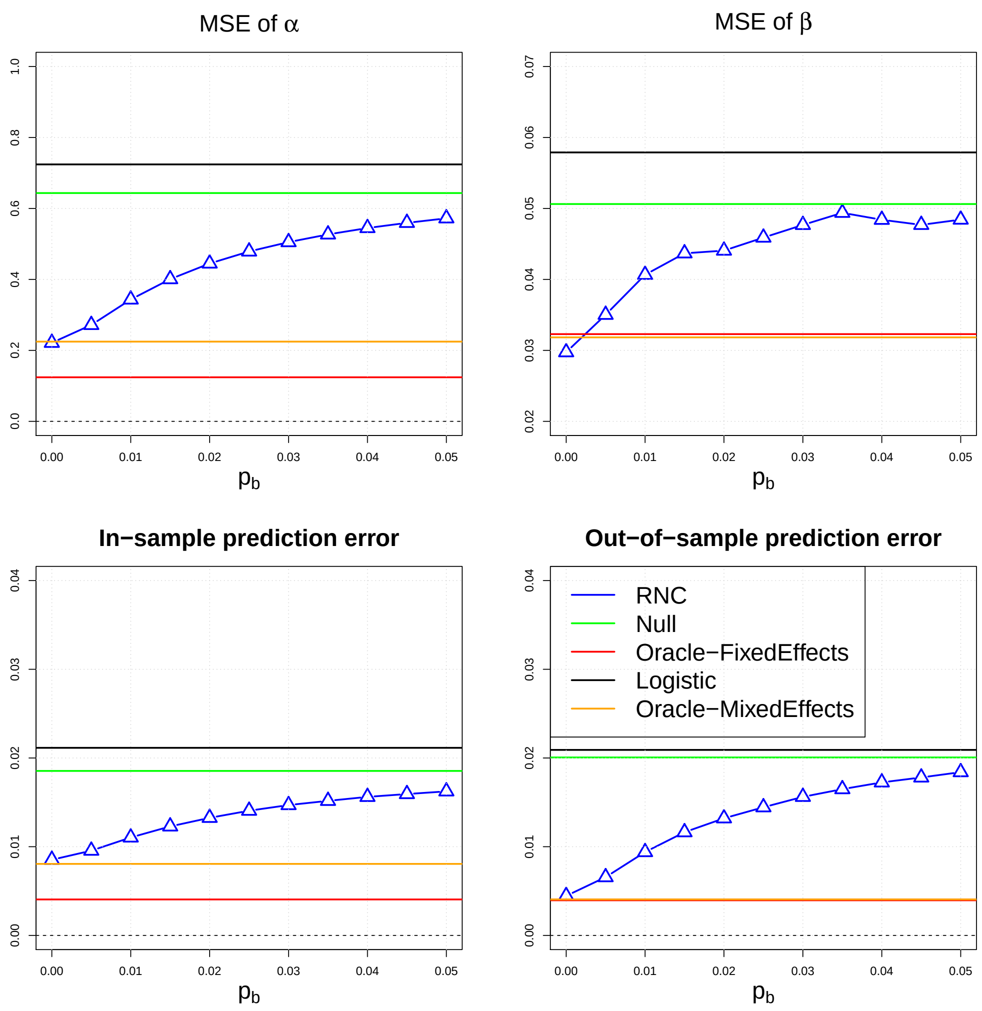


Figure 2.5: Performance of five logistic regression methods when varying p_b and fixing $s = 0.1$, measured by the MSE of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, in-sample and out-of-sample mean squared probability estimation errors.

We conclude this section with a simple example illustrating the graph sparsification approach to dense networks. We generate a weighted network with $n = 3000$ nodes, divided into three blocks of 1000 nodes each. All the within-block entries of the weighted adjacency matrix are 1 and the other entries are 0.1. Thus the network matrix is a fully dense matrix. The other settings are the same as in the linear regression simulation, and we compare the linear RNC estimator estimated using the original Laplacian L to the one based on the sparsified L^*. Figure 2.6 shows the results as a function for different values of the approximation accuracy ϵ's, defined in (2.17). The top left plot shows the the sparsified matrix corresponding to $\epsilon = 0.15$, which has around 52% of all elements set to 0. The top right plot shows the observed approximation error $\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2$ and its theoretical upper bound (2.22). The theoretical bound is conservative but follows the same trend. Finally, the bottom plots of the difference in estimation errors for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ show that the difference between the sparsified and the original estimators goes to 0 as $\epsilon \to 0$, as it should, and that for moderate values of ϵ the differences are small and can go in either direction, which suggests an increase in variance but not much change in bias. Overall, in this example sparsification provides a reliable approximation to the original RNC estimator, and is a useful tool to save computational time for large dense networks.

2.5 Analysis of the AddHealth Data

We investigate the ability of our method to capture network effects and improve prediction in two applications using data from the AddHealth study [Harris, 2009]. We will only use Wave I data in which both covariates and friendship networks are available. Our first test task is predicting students' recreational activity from their demographic covariates and their friendship networks; this was done via a network autoregressive model in Bramoullé et al. [2009], who used the in-school survey data. In order to be able to compare with their results directly, we also use the in-school data only for this task. The students were asked about friends at both in-school and in-home interviews, and the resulting networks are somewhat different. Our second application is predicting the age of first marijuana use, and the data on marijuana use are only available from the in-home interviews; thus for the second task we use the friendship network constructed from the in-home interviews. Prediction performance

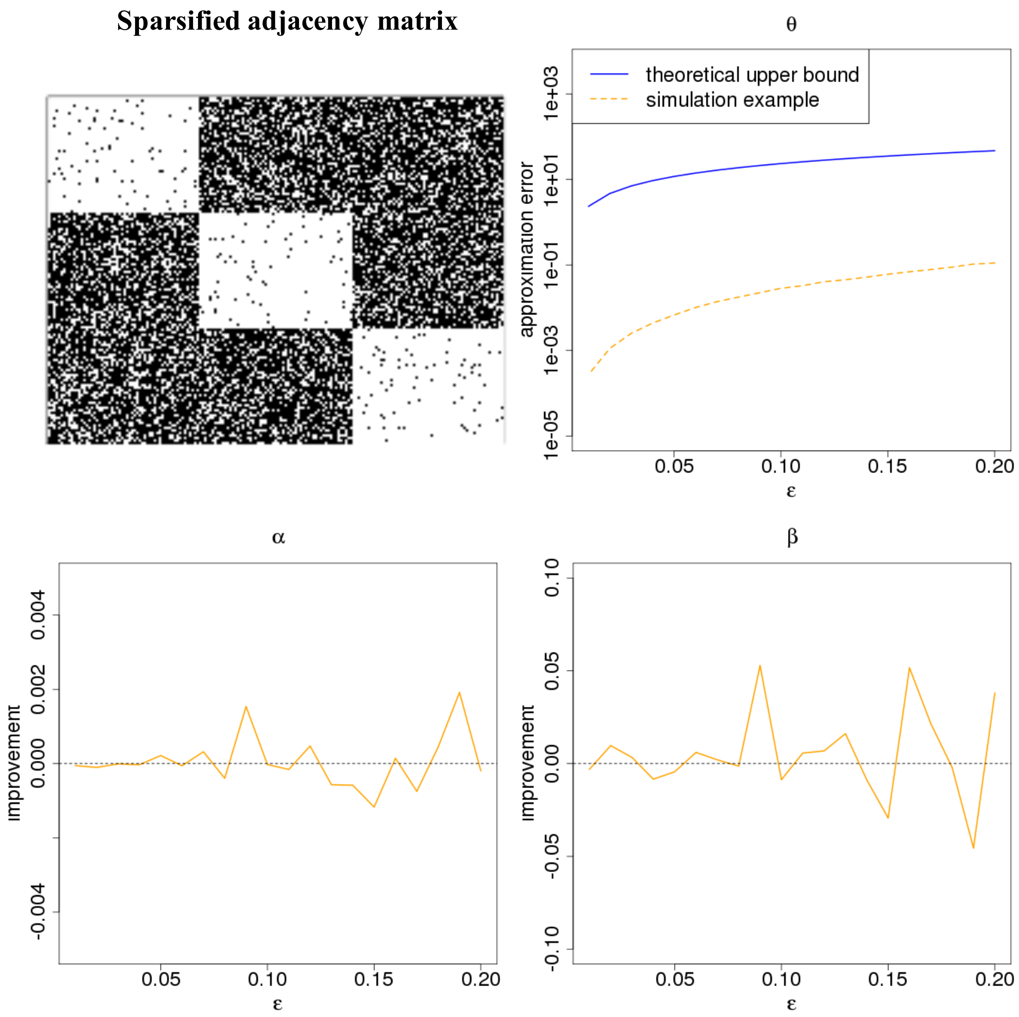


Figure 2.6: Top left: the adjacency matrix of the sparsified network for $\epsilon = 0.15$ (white indicates a nonzero entry, black is a zero entry); Top right: $\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2$ and the bound (2.22); Bottom left: relative improvement of the sparsified estimator $\boldsymbol{\alpha}^*$ over the original estimator $\hat{\boldsymbol{\alpha}}$, that is, $1 - \text{MSE}_{\boldsymbol{\alpha}^*}/\text{MSE}_{\hat{\boldsymbol{\alpha}}}$; Bottom right: relative improvement of the sparsified estimator $\boldsymbol{\beta}^*$ over the original estimator $\hat{\boldsymbol{\beta}}$.

on these two tasks is presented in this section. Additional results on sensitivity to missing data are presented in Appendix A.4.

2.5.1 Predicting recreational activity in adolescents: a linear model example

This exact task on the AddHealth data was considered by Bramoullé et al. [2009], who incorporated peer effects into ordinary linear regression in via the auto-regressive model

$$y_v = \frac{1}{|N(v)|} \sum_{u \in N(v)} (\gamma y_u + \boldsymbol{x}_u^T \boldsymbol{\tau}) + \boldsymbol{x}_v^T \boldsymbol{\beta} + \epsilon_v, v \in V \ , \tag{2.23}$$

or, equivalently, in matrix form

$$Y = (I - \gamma D^{-1} A)^{-1} (D^{-1} A X \boldsymbol{\tau} + X \boldsymbol{\beta} + \boldsymbol{\epsilon}). \tag{2.24}$$

The authors called this the social interaction model (SIM), also sometimes called a “linear-in-means” model. In econometric terminology, the local average of responses models endogenous effects, and the local averages of predictors are the exogenous effects. This model generally requires multiple additional assumptions to be identifiable and distributionally compatible across different equations, an issue not considered by Bramoullé et al. [2009]. It also loses the interpretation of predictor coefficients as the change in the predicted value corresponding to a unit increase in one predictor with all others fixed. When there are known groups in the data, fixed effects can be added to this model [Lee, 2007]. In Bramoullé et al. [2009], SIM was applied to the AddHealth data to predict levels of recreational activity from a number of demographic covariates as well as the friendship network. The covariates used are age, grade, sex, race, whether born in the U.S. living with the mother, living with the father, mother’s education, father’s education, and parents’ participation in the labor market. For some of the categorical variables, some of the levels were merged; refer to Bramoullé et al. [2009] for details. Recreational activity was measured by the number of clubs or organizations to which the student belongs, with “4 or more” recorded as 4. The histogram as well as the mean and standard deviation of recreational activity are shown in Figure 2.7. We used exactly the same variables with the same level merging.

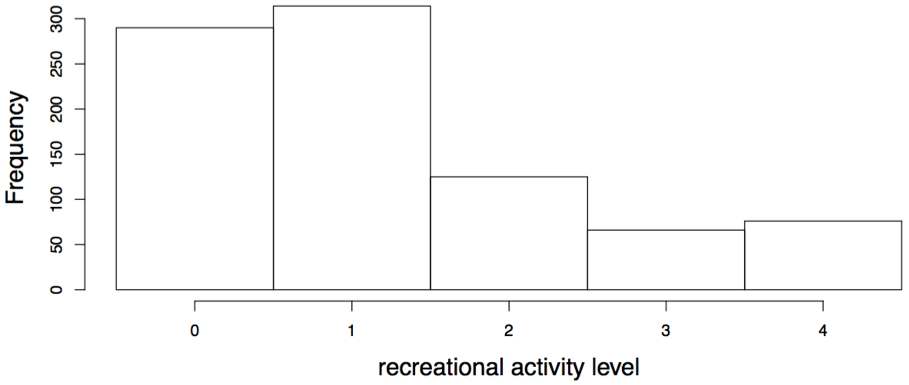


Figure 2.7: Histogram of the response, recreational activity level, from the data set used in the linear regression example. The mean recreational activity is 1.22, with standard deviation 1.23.

We compare performance of our proposed RNC method with the SIM model (2.23) from Bramoullé et al. [2009], and to regular linear regression without network effects implemented by ordinary least squares (OLS), with the same response and predictors as in Bramoullé et al. [2009]. As discussed, the null model always gives out-of-sample predictions identical to OLS, so we do not distinguish between them in this example. As an additional comparison with SIM, we also fit RNC with local averages of predictors as additional variables in the model (RNC-LA). We also apply the Bayesian model from Section 2.2.4, which is equivalent to the CAR model as discussed in Section 2.2.7. However, such a model can only make out-of-sample predictions if the entire network, including that of the test data, is available before training. Therefore, we implement this model as an oracle, including the entire network connecting the test and the training data at the training stage. We call this method "oracle-Bayes" to indicate it is using oracle information that is not available to all the other methods, and thus is not a fair competitor. The Bayesian estimates are computed as posterior medians from MCMC samples using the implementation of Lee [2013].

We use the largest school in the dataset, with 2350 students. For 1223 records with some missing values, we implemented conditional imputation, using random forests trained on all the variables without missing values. In Appendix A.4, we include a sensitivity analysis to the proportion of missing data, showing that our analysis is very robust. To order predictors, we randomly split the network in two connected subgraphs with similar sizes. We use one of these connected networks, with 898 data points, to perform variable selection, and the other network with 940 points for evaluating the models. The remaining 512 samples are not connected to either of the two networks and mostly consist of isolated nodes or isolated pairs; we remove them from the analysis since those are not going to be able to demonstrate peer effects.

We perform forward variable selection on the variable selection set, and then add variables in the selected order to the model fitted on the other dataset. Doing variable selection and model evaluation on two separate data sets avoids introducing model selection bias into our estimated prediction error. The forward selection procedure starts with fitting an RNC model without any covariates, obtaining an estimate of $\hat{\boldsymbol{\alpha}}$ from this model, and then running standard forward selection adding one variable at a time to $\hat{\boldsymbol{\alpha}}$ which always remains in the model with a fixed coefficient 1. This ensures that selected variables are not acting as proxies to peer effects but are adding as much new information as possible.

To evaluate predictive performance, we randomly hold out 90 students (about 10%) from the model evaluation dataset as test data, and fit all the models on the rest. The variables are added to all the models one at a time in the order determined by the variable selection procedure. The procedure is repeated for 50 independent random data splits into training and test sets. The root mean squared errors (RMSEs) over these 50 splits are shown in Table 2.1. In each row, we report the results from a paired t-test over the 50 random splits for each model compared with RNC. It is clear that both SIM and RNC are able to improve predictions by using information from the network, but RNC is more effective at this in all models. Including local averages of predictors does not help RNC at all, indicating that the network effects it picks are distinct from and perhaps more informative than the ones reflected in local average. The oracle Bayes method does not perform as well as RNC either, though it uses more information. A potential explanation for this may be that the specific distribution assumptions that the Bayesian model imposes are not satisfied for this dataset; in particular, it might be a stretch to model the 4-level ordinal recreation activity variable as Gaussian.

None of the demographic predictors are particularly strong, and network information is relatively more helpful: the RNC error using only network cohesion and none of the predictors is lower than the error of *any* model fitted by either OLS, SIM or oracle Bayes. As with any other prediction task, adding unhelpful covariates tends to corrupt performance, and RNC achieves the best performance with only one predictor in the model (father's education). Finally, the coefficients from both OLS and RNC regressions are reported in Table A.1 of Appendix A.3. They are generally similar, suggesting that the network cohesion penalty does not fundamentally change interpretation of the coefficients, but improves prediction.

Remark II.16. For fair comparison with Bramoullé et al. [2009], we formulate the problem of predicting recreational activity level as a linear regression problem. However, given the ordinal nature of the response, an alternative option may be using ordinal regression with network cohesion.

model	OLS & Null	SIM	RNC	RNC-LA	oracle-Bayes
no covariates	1.217 **	1.177 **	1.157	1.157	1.165 *
+ father's education	1.215 **	1.180 **	1.156	1.160 *	1.165
+ race	1.213 **	1.178 **	1.158	1.164 *	1.163
+ age	1.214 **	1.177 **	1.158	1.163	1.161
+ mother's education **	1.216 **	1.179	1.160	1.167	1.171
+ born in the US	1.217 **	1.179 **	1.161	1.169	1.168
+ sex	1.211 **	1.174 *	1.157	1.161	1.167
+ parents in labor market	1.214 **	1.179 **	1.159	1.165	1.172 *
+ living with mother	1.216 **	1.182 **	1.161	1.169	1.174 *
+ living with father	1.218 **	1.186 **	1.163	1.174 *	1.172
+ grade	1.219 **	1.188 **	1.163	1.176 *	1.175 *

Table 2.1: Root mean squared errors for predicting recreational activity, over 50 independent data splits into test (90 samples) and training sets. All methods are compared to RNC by a paired two-sample t-test, where ** indicates $p \leq 10^{-4}$ and * indicates $10^{-4} < p < 10^{-2}$. Each row adds the variable listed to the model in the previous row, in the order determined on a separate set by forward selection with network cohesion effects included.

2.5.2 Predicting the risk of adolescent marijuana use

While many prediction tasks can be addressed with linear or logistic regression, there are settings where survival analysis is the only appropriate tool. In the AddHealth survey, the students were asked "How old were you when you tried marijuana for the first time?", and the answer can either be age (an integer up to 18) or "never", which is a censored observation of age of first use. A survival model is thus the appropriate prediction tool. Here we apply Cox's proportional hazard model, with network cohesion, to the largest community in the dataset with 1862 students from the Wave I in-home interview (this question was only asked in the in-home interviews). The friendship network is also based on in-home data for consistency; there are 2820 additional covariates on each student collected from the in-home surveys.

As before, missing values are imputed with conditional imputation using random forests, with covariates without missing values as predictors. However, we deleted variables that had missing values due to questionnaire design, and variables with more than half the values missing. This left us with 218 variables in total (since there are so many variables in the in-home survey, there are many missing values). As in the previous example, we split the data randomly into two connected components of roughly equal sizes, 645 observations for variable selection and 647 observations for model evaluation. The remaining isolated nodes or pairs are removed. The variable selection step is implemented as in the previous example, with network cohesion effects in the model. Five strongest predictors are selected, with the additional requirement that each survey category (survey questions were grouped) has no more than one variable selected. We then use a regular forward selection algorithm to determine the order in which these five variables should be added to the model.

Given the selected variables and the order in which to add them, we fit the regular Cox's model, the null model, and the RNC for survival on the model evaluation data set. The null model is numerically nearly identical to the regular Cox's model. We also include a naive extension of the social interaction model (SIM) (2.23) to survival analysis, including the neighborhood averages of x's as extra covariates. However, the neighborhood averages of y's cannot be computed here, since many of the y's are censored and it is not clear how to extend the autoregressive component of the model to survival data. We also include "RNC-LA" again, which adds all the local averages of predictors to the RNC model. In the survival model, RNC can be fitted with no covariates, but this is not possible for the regular Cox's model or SIM since partial likelihood is not defined without covariates.

Evaluating predictive performance of survival models is not straightforward; we use the survival ROC curve suggested by Song and Zhou [2008]. We calculate the prediction ROC curve for each age between 14 and 17 (most age values fall into this range), then integrate the area under curve (AUC) over age to get a measure of overall prediction performance. We randomly select 60 nodes (about 10%) as the test set and use the remaining nodes and their induced sub-network as the training set. This is independently repeated 50 times and the average integrated AUC (iAUC) over the 50 replications is used to evaluate performance. For simplicity of comparisons, we fixed the tuning parameter $\lambda = 0.005$ for all models, based on validation on a different school, and set $\gamma = 0.1$. This results in a conservative comparison of our method to Cox's model, since tuning each RNC fit separately can only improve its performance.

model	Cox & Null	SIM	RNC	RNC-LA
no covariates	–	–	0.606	0.606
+ ever tried cigarette smoking	0.657 **	0.663 **	0.709	0.703 **
+ deliberately damaged others' property	0.700 **	0.707 **	0.735	0.736
+ times of being jumped in past 12 months	0.713 **	0.733 *	0.740	0.758 **
+ how often to wear seatbelt in a car	0.721 **	0.743	0.745	0.765 **
+ received school suspension	0.727 **	0.743	0.748	0.766 **

Table 2.2: Average integrated AUC (iAUC) for survival prediction ROC curves for age 14-17, over 50 random splits of the data into training and test sets. All methods are compared with RNC by a paired two-sample t-test. ** indicates $p \leq 10^{-4}$ and * indicates $10^{-4} < p < 10^{-2}$. Each row adds the variable listed to the model in the previous row, in the order determined on a separate set by forward selection with network cohesion effects included.

Table 2.2 shows the average iAUC results. All models improve or stay the same with additional predictors. All methods that use the network information always do better than the regular Cox's model with the same covariates. RNC always outperforms SIM, and RNC-LA improves upon RNC for models with more covariates, but not for the smaller ones. This may suggest that some predictors' local averages are more helpful than others; however, including any local averages distorts the meaning of the coefficients. Overall, the network cohesion effect in predicting marijuana usage is clearly useful.

The estimated individual hazards $\exp(\hat{\alpha}_i)$'s are shown in Figure 2.8, represented by node size, together with the friendship network and the observed age represented by node color. The cohesion effect can be seen in both the data itself and in the estimated hazards.

Table 2.3 shows the coefficients of the regular Cox's model and the RNC model. They are overall similar, though it appears that for most variables the coefficient is slightly reduced with the addition of network effects. This makes sense since some of the covariates are also likely cohesive Michell and West [1996], Pearson and Michell [2000], Pearson and West [2003] and can serve as proxies to peer effects, thus appearing to be more influential than they really are by themselves.

covariate	Cox & Null	RNC	p-value (from Cox)
ever tried cigarette smoking	1.627	1.370	$< 10^{-6}$
deliberately damaged others' property	0.348	0.367	$< 10^{-4}$
times of being jumped in past 12 months	-0.122	-0.191	0.077
how often wears seatbelt	0.288	0.283	0.007
received school suspension	0.633	0.473	$< 10^{-6}$

Table 2.3: Estimated covariate coefficients from regular Cox's model and RNC for first age of marijuana use prediction.

2.6 Summary and future work

We have proposed a general computationally efficient framework for introducing network cohesion effects into prediction problems, without losing the interpretability and meaning of the original predictors. For the regression setting, we also derived conditions for when this approach outperforms regular regression and have shown the proposed estimator is consistent. In general, we can view RNC as another example of benefits of regularization when there are more parameters than one can estimate with the data available. Encouraging network cohesion implicitly reduces the number of free parameters, somewhat in the same spirit as the fused lasso penalty [Tibshirani et al., 2005]. There are important differences, however; we have a computationally efficient way to use the available network data whereas the fused lasso optimization problem is hard to solve, and we can explicitly assess the trade-off in bias and variance that results from encouraging cohesion.

A future direction to explore is understanding the behavior of network cohesion on different kinds of networks. The large literature on random graph models for networks gives many options for modeling the network as random rather than treating it as fixed, as we did here; we would expect that some types of networks spread cohesion over the network faster than others. While we focused on prediction in this chapter, the cohesion penalty may also turn out to be useful in causal inference on networks when such inference is possible. Formal inference under cohesion, such as confidence intervals and hypothesis tests, are also left for future work.

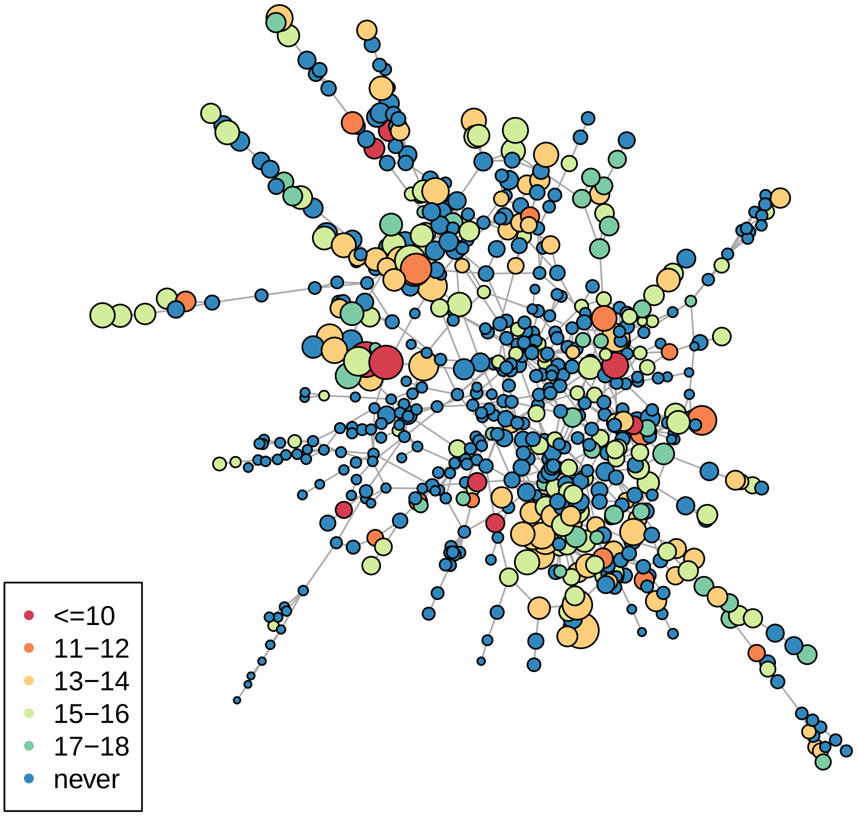


Figure 2.8: Age of first marijuana risk use shown on the friendship network. Node size represents the individual's hazard, and node color represents the observed age of first use.

CHAPTER III

Network cross-validation by edge-sampling

3.1 Introduction

Statistical methods for analyzing networks have received a lot of attention because of their wide-ranging applications in areas such as sociology, physics, biology and medical sciences. Statistical network models provide a principled approach to extracting salient information about the network structure while filtering out the noise. Perhaps the simplest statistical network model is the famous Erdös-Renyi model [Erds and Rényi, 1960], which served as a building block for a large body of more complex models, including the stochastic block model (SBM) [Holland et al., 1983], its variants such as the degree-corrected stochastic block model (DCSBM) [Karrer and Newman, 2011] or mixed membership block model (MMBM) [Airoldi et al., 2008], and the latent space model [Hoff et al., 2002], to name a few.

While there has been plenty of work on models for networks and algorithms for fitting them, inference frameworks for these models are commonly lacking, making it hard to take advantage of the full power of statistical modeling. Resampling methods provide a general and relatively model-free inference framework and are commonly used in modern statistics, with bootstrap and cross-validation being the tools of choice for a large number of inference tasks. Neither of these procedures are directly applicable to network, because, while they differ in details, they both face the challenge of sampling multiple networks which are "similar" to the observed network but not the same; formally, they need to be sampled from the same distribution, but the distribution is unknown and we only have one network we observed from it. This seems an impossible problem, except there is often structure in the network that can be estimated and exploited to create a sampling mechanism. The method we propose in this paper is equally applicable to creating bootstrap samples or to performing cross-validation. For simplicity of presentation, we present the method in the context of cross-validation, but the multiple noisy version of the original network it creates can be equally well used for bootstrap.

Cross-validation is important in most network modeling situations – while there are plenty of models to choose from, it is a lot less clear how to select the best model for the data, and how to choose tuning parameters for the selected model, which is often necessary in order to fit it.

In classical settings where the data points are assumed to be an i.i.d. sample, cross-validation (CV) is one of the most general and appealing ways for model selection and parameter tuning. In general, cross-validation works by splitting the data into multiple parts (folds), holding out one fold at a time as a test set, fitting the model on the remaining folds and computing its error on the held out-fold, and finally averaging the errors across all folds to obtain the cross-validation error. The tuning parameter is then chosen to minimize this error. To explain the challenge of applying this idea to networks, we first introduce a probabilistic framework.

Let $\mathcal{V} = \{1, 2, \cdots, n\} =: [n]$ denote the node set of a network, and let A be its $n \times n$ adjacency matrix, where $A_{ij} = 1$ if there is an edge from node i to node j and 0 otherwise. For undirected networks, A is a symmetric matrix. We view the elements of A as realizations of independent Bernoulli variables, with $\mathbb{E}A = M$, where M is a matrix of probabilities. For undirected networks, we further assume M is symmetric and the unique edges A_{ij}, $i < j$ are independent Bernoulli variables, and $A_{ji} = A_{ij}$. The general task is to estimate M from the data A, under various structural assumptions we might make to address the difficulty of having a single realization of A.

To perform cross-validation on networks, one has to decide how to split the data contained in A, and how to treat the resulting partial data which is not a complete network any more. To the best of our knowledge, there is little work available on the topic. Cross-validation was used by Hoff [2008] under a particular latent space model, and Chen and Lei [2017] propose a novel cross-validation strategy for model selection under the stochastic block model and its variants. In this paper, we do not assume any specific model for the network, but instead make a more general structural assumption of M being approximately low rank, which holds for most popular network models. We propose a new general *edge* cross-validation (ECV) strategy for networks, splitting node pairs rather than nodes into different folds, a natural yet crucial choice. Treating the network after removing the entries of A for some node pairs as a partially observed network, we apply low rank matrix completion to "complete" the network and then fit the relevant model. This reconstructed network has the same rate of concentration around the true model as the full network adjacency matrix, allowing for valid analysis. Our method is valid for many types of network models, including directed and undirected, binary and weighted networks. As concrete examples, we show how ECV can be applied to determine the latent space dimension for random dot product models, select between block model variants, tune regularization for spectral clustering, and tune neighborhood smoothing for graphon models.

The rest of this chapter is organized as follows. Section 3.2 introduces the new edge-based cross-validation algorithm (ECV) as a generic framework, as well as a general error bound on ECV. Section 3.3 presents several specific network model selection and parameter tuning problems and demonstrates how ECV can be used for these tasks, including selecting rank of a generic low-rank network model, model selection in block models, tuning regularized spectral clustering, and tuning graphon estimation. Section 3.4 presents extensive simulation studies of ECV and its competitors for the tasks introduced in Section 3.3. Section 3.5 presents an application to a weighted statistics citation network, and Section 3.6 concludes with discussion. The proofs and additional numerical results are given in the Appendix B.

3.2 The edge cross-validation (ECV) algorithm

3.2.1 Notation and model

For simplicity of presentation, we derive everything for binary networks, but it will be clear that our framework is directly applicable to weighted networks, which are prevalent in practice. In Section 3.5, we apply the method to a real weighted network.

Recall n is the number of nodes in the network and A is its $n \times n$ adjacency matrix. For undirected networks, A is a symmetric matrix. Let $D = \mathrm{diag}(d_1, d_2, \cdots, d_n)$ be the diagonal matrix with node degrees $d_i = \sum_j A_{ij}$ on the diagonal. The (normalized) Laplacian of a network is defined as $L = D^{-1/2}AD^{-1/2}$. Finally, we write I_n for the $n \times n$ identity matrix and $\mathbf{1}_n$ for $n \times 1$ column vector of ones, suppressing the dependence on n when it is clear from the context. For any matrix M, we use $\|M\|$ to denote its spectral norm and $\|M\|_F$ to denote its Frobenius norm.

We follow the exchangeable random graph model framework [Aldous, 1981] which includes most current random network models in statistics. Generically, all such models assume that given a $n \times n$ matrix of probabilities M, the A_{ij}'s are independent Bernoulli variables with $\mathbb{P}(A_{ij} = 1) = M_{ij}$, and the model assumptions are made on the structure of M. As we have discussed, without any assumptions on M inference is impossible since we only have one observation. On the other hand, we would like to avoid assuming a specific parametric model, since one of the goals of cross-validation is exactly to choose between models, and thus we would rather not assume we know exactly how the network was generated. As a compromise, we make a weak structural assumption on M, assuming it is low rank, which holds for many popular network models. Consider the following examples:

Random dot product graph model (RDPG). The RDPG [Young and Scheinerman, 2007] is a general low-rank network model and a special case of the latent space model. RDPG assumes each node of the network is associated with a latent K-dimensional vector $Z_i \in \mathbb{R}^K$, and $M_{ij} = Z_i^T Z_j$. RDPG has been successfully applied to a number of network problems [Sussman et al., 2014, Tang et al., 2017] and its limiting behaviors can also be studied [Tang and Priebe, 2016]. More details can be found in the review paper of Athreya et al. [2017].

Stochastic block model (SBM) and its generalizations. The SBM is perhaps the most widely used undirected network model with communities. The SBM assumes that $M = ZBZ^T$ where $B \in [0,1]^{K \times K}$ is a symmetric probability matrix and $Z \in \{0,1\}^{n \times K}$ has exactly one "1" in each row, with $Z_{ik} = 1$ if node i belongs to community k. Let $\boldsymbol{c} = \{c_1, \cdots, c_n\}$ be the vector of node membership labels c_i taking values in $1, \ldots, K$. The SBM assumes $P(A_{ij} = 1) = B_{c_i c_j}$, that is, the probability of edge between two nodes depends only on the communities they belong to. One of the commonly pointed out limitations of the SBM is that it forces equal expected degrees for all the nodes in the same community, therefore ruling out “hubs”. The degree corrected stochastic block model (DCSBM) corrects this by allowing nodes to have individual “degree parameters”, θ_i associated with each node i, and models $P(A_{ij} = 1) = \theta_i \theta_j B_{c_i c_j}$. The DCSBM needs a constraint to ensure identifiability, and here we enforce the constraint $\sum_{c_i = k} \theta_i = 1$, for each k, proposed by Karrer and Newman [2011].

Both the SBM and the DCSBM result in the probability matrix M of rank K, and are in fact special cases of the RDPG. There are other variants of SBM that are also low rank, for example the mixed membership block model proposed by Airoldi et al. [2008] where $P = \Gamma B \Gamma$ with $\Gamma \in \mathbb{R}^{n \times K}$ and each row of Γ_i is generated from a Dirichlet distribution. More about the recent developments on this class of models can be found in the review paper by Abbe [2017].

Graphon models. Aldous [1981] showed the probability matrix of any exchangeable random graph can be written as $M_{ij} = f(\xi_i, \xi_j)$ for a function $f : [0,1] \times [0,1] \to [0,1]$ (called the graphon) symmetric in its two arguments, and $\xi_i, i \in [n]$ are independent uniform random variables on $[0,1]$. The representation is determined only up to a measure-preserving transformation [Diaconis and Janson, 2007]. There is a substantial literature on estimating the graphon under various assumptions on f [Wolfe and Olhede, 2013, Choi and Wolfe, 2014, Gao et al., 2015]. In general, graphon models typically do *not* assume a low rank M. However, when the function f is smooth in certain sense, the corresponding matrix can typically be approximated reasonably well by a low rank matrix [Chatterjee, 2015], and f is not smooth, the sample size of 1 makes inference impossible once again. For this setting, even though the low rank assumption is not strictly correct, our proposal can be viewed as a cross-validation procedure based on the best low rank approximation (details discussed in Section 3.4.4).

3.2.2 The ECV procedure

For notational simplicity, we only present the algorithm for directed networks; the only modification needed for undirected networks is treating node pairs (i, j) and (j, i) as one pair. The key insight of ECV is to split node pairs rather than nodes, resulting in a partially observed network. We randomly sample node pairs (regardless of the value of A_{ij}) with a fixed probability $1 - p$ to be in the held-out set. By exchangeable model assumption, the values of A corresponding to held-out node pairs are independent of those corresponding to the rest. The leftover training network now has missing edge values, which means many models and methods cannot be applied to it directly. Our next step is to reconstruct a "complete" network $\hat{A}$ from the training node pairs. Fortunately, the missing entries are missing completely at random by construction, and this is the classic setting for matrix completion. Any low-rank based matrix completion algorithm can now be used to fill in the missing entries, for example Candes and Plan [2010], Davenport et al. [2014]. We postpone the algorithm details to Section 3.2.3.

Once we complete $\hat{A}$ through matrix completion, we can fit the candidate models on $\hat{A}$ and evaluate the relevant loss on the held-out entries of A, just as in standard cross-validation. There may be more than one way to evaluate the loss on $\hat{A}$ if the loss function itself is designed for binary input; we will elaborate on this in examples in Section 3.3. The general algorithm is summarized as Algorithm III.1 below. We present the version with many random splits into training and test pairs, but it is obviously applicable to K-fold cross-validation if the computational cost of many random splits is prohibitive.

Algorithm III.1 (The general ECV procedure). Input: an adjacency matrix A, a loss function L, a set $\mathcal{C}$ of Q candidate models or parameter values to select from, the training proportion p, and the number of replications N.

1. For $m = 1, \ldots, N$
   (a) Randomly choose a subset of node pairs $\Omega \subset \mathcal{V} \times \mathcal{V}$, selecting each pair independently with probability p.
   (b) Apply a low-rank matrix completion algorithm to (A, Ω) to obtain $\hat{A}$.
   (c) For each of the candidate models $q = 1, \ldots, Q$, fit the model on the $\hat{A}$, and evaluate its loss $L_q^{(m)}$ by averaging the loss function L with the estimated parameters over the held-out set $A_{ij}, (i,j) \in \Omega^{\perp}$.
2. Let $L_q = \frac{1}{N}\sum_{m=1}^{N} L_q^{(m)}$ and return $\hat{q} = \mathrm{argmin}_q L_q$ (the best model from set $\mathcal{C}$).

The two crucial parts of ECV are splitting node pairs at random and applying low-rank matrix completion to obtain a full matrix A. While we focus on cross-validation for model selection in this paper, it is clear that the exact same procedure can be used to create a bootstrap sample from A, of networks of the same size, which can be viewed as independent noisy versions of A.

3.2.3 Network recovery by matrix completion

There are many algorithms that can be used to recover $\hat{A}$ from the training pairs. Define operator $P_\Omega : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ by

$$(P_\Omega A)_{ij} = A_{ij}\mathbf{I}\{(i,j) \in \Omega\} ,$$

replacing held-out entries by zeros. A generic low-rank matrix completion procedure solves the problem

$$\begin{aligned} &\min_{Z} \;\; F(P_\Omega Z, P_\Omega A) \\ &\text{subject to} \;\; \operatorname{rank}(Z) \leq K \end{aligned} \tag{3.1}$$

where K is the rank constraint and F is a loss function measuring the discrepancy between Z and A on entries in Ω, for example, sum of squared errors or binomial deviance. The problem is non-convex due to the rank constraint, so many computationally feasible variants of (3.1) have been proposed for use in practice, obtained via convex relaxation and/or problem reformulation. While any such method can be used in ECV, for concreteness we follow the singular value thresholding procedure to construct a low rank approximation

$$\hat{A} = S_H\left(\frac{1}{p} P_\Omega A, K\right), \tag{3.2}$$

where $S_H(B, K)$ denotes rank K truncated SVD of a matrix B. That is, if the SVD of B is $A = UDV^T$ where $D = \operatorname{diag}(\sigma_1, \sigma_2, \cdots, \sigma_n)$, $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_n \geq 0$, then $S_H(A, K) = UD_K V^T$, where $D_K = \operatorname{diag}(\sigma_1, \sigma_2, \cdots, \sigma_K, 0, \cdots, 0)$.

This matrix completion procedure is similar to the universal singular value thresholding (USVT) method of Chatterjee [2015], except we fix K and always use top K eigenvalues and USVT uses a universal constant to threshold σ's. This method is very computationally efficient as it only requires a partial SVD of the adjacency matrix with held out edges replaced by zeros, which is typically sparse. It runs easily on a network of size $10^4 - 10^5$ on a laptop. There are more involved matrix completion algorithms, such as, for example, Keshavan et al. [2009] and Mazumder et al. [2010], which may sometimes give better accuracy. One can choose a more sophisticated method if the size of the network allows, but considering completion accuracy is not the ultimate goal here, since we expect and in fact need noisy versions of A, it may not be worth the extra computational cost: we tried the iterative method which appears as a primal version of the hardImpute algorithm in Mazumder et al. [2010], and while it improves matrix completion accuracy itself, the improvement in the model selection task is very small.

One could also consider binary rather than general matrix completion methods, also known as 1-bit matrix completion [Davenport et al., 2014, Cai and Zhou, 2013, Bhaskar and Javanmard, 2015], which are in fact more appropriate since A is a binary matrix. However, 1-bit matrix completion methods are generally much more computationally demanding than the Frobenius norm-based completion. For the rest of this paper, we use non-binary completion, which can also be thought of as estimating $\mathbb{E}A$, using truncated SVD (3.2) because of its low computational cost.

It remains to specify the rank K. In some situations, K itself is directly associated with the model to be selected, and thus there is an obvious choice, as in problems in Sections 3.3.1 and 3.3.3. In other situations, such as the graphon estimation problem in Section 3.3.4, K is not directly available, so we simply add K as an extra model selection parameter; see Section 3.3.1. One can also avoid selecting a K entirely by using the universal threshold proposed for USVT by Chatterjee [2015], but in practice we found this leads to lower model selection accuracy.

Remark III.2. If an upper bound on $\|M\|_\infty$ is available, say $\|M\|_\infty \leq \bar{d}/n$, an improved estimator $\tilde{A}$ can be obtained by truncating the entries of $\hat{A}$ onto the interval $[0, \bar{d}/n]$, as in Chatterjee [2015]. A trivial option of truncating to the interval $[0, 1]$ is always available, ensuring $\tilde{A}$ is a better estimator of M in Frobenius norm than $\hat{A}$. We did not observe any substantial improvement in model selection performance from truncation, however. In some applications, a binary adjacency matrix may be required for subsequent model fitting; if that is the case, a binary matrix can be obtained from $\tilde{A}$ by using one of the standard link prediction methods, for example, by thresholding at 0.5

Remark III.3. An alternative to matrix completion is to simply replace all of the held out entries by zeros and use the resulting matrix A^0 for model estimation. The resulting model estimate M^0 of the probability matrix EA^0 is a biased estimator of M, but since we know the missing probability p, we can remove this bias by setting $M^* = M^0/p$ as in Chatterjee [2015] and Gao et al. [2016], then use M^* for prediction and calculating the cross-validation error. This method is valid as long as the adjacency matrix is binary and probably the simplest of all (though, surprisingly, we did not find any explicit references to this in the literature). In particular, for the stochastic block model it is equivalent to our general ECV procedure when using (3.2) for matrix completion. However, in applications beyond block models these two approaches will give different results, and we have empirically observed that ECV with matrix completion works better and is much more robust to the choice of p. Moreover, filling in zeros instead of doing matrix completion does not work for weighted networks, since that would clearly change the weight distribution which cannot be fixed by a simple rescaling by p. We do not pursue this version further.

3.2.4 Theoretical justification

Intuitively, ECV should be valid as long as $\hat{A}$ reflects relevant properties of the true underlying model. The following theorem formalizes this intuition. We make two assumptions:

Assumption III.4. $\mathrm{rank}(M) = K$.

Assumption III.5. $\max_{ij} M_{ij} \leq d/n$ *for some positive* d.

Assumption III.5 can be satisfied trivially by setting $d = n$. However, in many network models the entries of M are assumed to be $o(1)$ in order to avoid a dense graph, and our bounds can be improved if additional information about d is available.

Theorem III.6. *Let M be a probability matrix satisfying III.4 and III.5. Let A be an adjacency matrix with edges sampled independently and $\mathbb{E}(A) = M$. Let Ω be an index matrix for a set of edges selected independently with probability $p \geq C_1 \log n/n$ for some absolute constant C_1, with $\Omega_{ij} = 1$ if the edge (i,j) is selected and 0 otherwise. If $d \geq C_2 \log(n)$ for some absolute constant C_2, then with probability at least $1 - 3n^{-\delta}$ for some $\delta > 0$, the completed matrix $\hat{A}$ defined in* (3.2) *satisfies*

$$\|\hat{A} - M\| \leq \tilde{C} \max\left(\sqrt{\frac{Kd^2}{np}}, \sqrt{\frac{d}{p}}, \frac{\sqrt{\log n}}{p}\right) \tag{3.3}$$

where $\tilde{C} = \tilde{C}(\delta, C_1, C_2)$ is a constant that only depends on C_1, C_2 and δ. This also implies

$$\frac{\|\hat{A} - M\|_F^2}{n^2} \leq \frac{\tilde{C}^2}{2} \max\left(\frac{K^2 d^2}{n^3 p}, \frac{Kd}{n^2 p}, \frac{K \log n}{n^2 p^2}\right). \tag{3.4}$$

This theorem holds for both directed and undirected networks. The Frobenius error bound (3.4) can be directly compared with other bounds in the matrix completion literature. For binary matrix completion, Davenport et al. [2014] give

$$\frac{\|\hat{A} - M\|_F^2}{n^2} = O\left(\sqrt{\frac{K}{np}}\right) \tag{3.5}$$

using nuclear norm relaxation of the rank constraint. The same bound was obtained by Chatterjee [2015] for the USVT without using a pre-defined K. Since both Davenport et al. [2014] and Chatterjee [2015] assume $\|M\|_\infty$ is bounded, for comparison we take $d = n$ in III.5. This gives

$$\frac{\|\hat{A} - M\|_F^2}{n^2} = O\left(\max\left(\frac{K^2}{np}, \frac{K \log n}{n^2 p^2}\right)\right).$$

Our bound and (3.5) differ by a factor of $O(\max(\sqrt{\frac{K^3}{np}}, \sqrt{\frac{K}{np}\frac{\log n}{np}}))$, which is dominated by $O(\sqrt{\frac{K^3}{np}})$since we require $p \geq C_1 \frac{\log n}{n}$ in Theorem III.6. Therefore as long as $K = o((np)^{1/3})$, our bound is better. Moreover, in ECV we control p and can treat it as a constant, which makes our bound better as long as $K = o(n^{1/3})$. The gain comes from pre-defining K in the matrix completion procedure, as opposed to the universal threshold used by the USVT of Chatterjee [2015]

Next, we compare Theorem III.6 with known rates for previously studied network problems. Again, we set p to be a constant, so the spectral norm error bound (3.3), taking into account the assumption $d \geq C_2 \log n$, becomes

$$\|\hat{A} - M\| \leq \tilde{C} \max\left(\sqrt{\frac{Kd}{n}}, 1\right) \sqrt{d}\ . \tag{3.6}$$

The bound (3.6) implies the rate of concentration of $\hat{M}$ around M is the same as the concentration of the full adjacency matrix A around its expectation [Lei and Rinaldo, 2014, Chin et al., 2015, Le et al., 2017], as long as $\frac{Kd}{n} \leq 1$. The sparser the network, the weaker our requirement for K. For instance, when the network is moderately sparse with $d = O(\log n)$, we only need $K \leq (n/\log n)$. This may seem counter-intuitive but this happens because the dependence on K in the bound comes entirely from M itself. A sparse network means that most entries of M are very small, thus replacing the missing entries in A with zeros does not contribute much to the overall error and the requirement on K can be less stringent. While for sparse networks the estimator is noisier, the noise bounds have the same order for the complete and the incomplete networks (when p is a constant), and thus the two concentration bounds still match.

Theorem III.6 essentially indicates

$$\|\hat{A} - M\| \approx \|A - M\|$$

if we assume $Kd \leq n$. Thus in the sense of concentration in spectral norm, we can treat $\hat{A}$ as a network sampled from the same model. Under the block models (see Section 3.3.2), such concentration of $\hat{A}$ is sufficient to ensure model estimation consistency at the same rate as can be obtained from using the original matrix A.

3.3 Examples of ECV for model selection

3.3.1 Model-free rank estimators

The rank constraint for the matrix completion problem is typically unknown, and in practice we need to choose or estimate it in order to apply ECV. When the true model is a generic low-rank model such as RDPG, selecting K is essentially selecting its latent space dimension. More generally, selection of K can itself be treated as a model selection problem, since the completed matrix $\hat{A}$ itself is a low rank approximation to the underlying probability matrix M. Since M is of course unknown, we will need to compare $\hat{A}$ to A in some way in order to select K.

One natural approach is to directly compare the values of $\hat{A}$ on the held-out entries of A. For instance, we can use the sum of squared errors,

$$\text{SSE} = \sum_{(i,j) \in \Omega^{\perp}} (A_{ij} - \hat{A}_{ij})^2,$$

or alternatively compute the binomial deviance (when the network is unweighted) on this set, and pick the value of K to minimize it.

Another possibility is to consider how well $\hat{M}$ performs on predicting links (for unweighted networks). We can predict $\hat{A}_{ij} = \mathbf{I}\{\hat{M}_{ij} > c\}$ for all entries in the hold-out set $\Omega^{\perp}$ for a threshold c, and vary c from 0 to 1 to obtain a sequence of link prediction results. A common measure of prediction performance is the area under the ROC curve (AUC), which compares false positive rates to true positive rates for all values of c, with perfect prediction corresponding to AUC of 1, and random guessing to 0.5. We can then select the K to maximize the AUC.

In practice, we have observed that both the imputation error and the AUC work well in general rank estimation tasks. For block models, they perform comparably to likelihood-based methods most of the time.

3.3.2 Model selection for block models

In this example, we show how to use ECV for model selection for SBM and DCSBM (referred to together for conciseness). The choice of fitting method is not crucial for model selection, and many methods are now available and known to be consistent for fitting the SBM and DCSBM [Karrer and Newman, 2011, Zhao et al., 2012, Bickel et al., 2013, Amini et al., 2013]. Here we use one of the simplest, fastest, and most common methods, spectral clustering on the Laplacian $L = D^{1/2}AD^{1/2}$, where D is the diagonal matrix of node degrees. For SBM, spectral clustering takes K leading eigenvectors of L, arranged in a $n \times K$ matrix U, and applies the K-means clustering algorithm to the rows of U to obtain cluster assignments for the n nodes. For DCSBM, the rows need to be normalized first.

Spectral clustering enjoys asymptotic consistency under the SBM when the average degree grows at least as fast as $\log n$ [Rohe et al., 2011, Lei and Rinaldo, 2014, Sarkar and Bickel, 2015]. The possibility of strong consistency for spectral clustering is recently discussed by Eldridge et al. [2017], Abbe et al. [2017] and Su et al. [2017]. Variants of spectral clustering are consistent under the DCSBM, for example, spherical spectral clustering [Qin and Rohe, 2013, Lei and Rinaldo, 2014] which normalizes the rows of U before applying K-means and the SCORE method [Jin, 2015] that divides each column of U by the first column of U.

Note that since both SBM and DCSBM are undirected network models, we use the undirected variant of ECV, selecting edges at random from the set of pairs (i, j)

with $i < j$ only and including the pair (j, i) whenever (i, j) is selected. Once node memberships are estimated, the other parameters are easy to estimate by conditioning on node labels. Specifically, for the SBM we simply take the MLE conditional on the node labels evaluated on the available node pairs. Let $\hat{C}_k = \{i : (i, j) \in \Omega, \hat{c}_i = k\}$ be the estimated member sets for each group $k = 1, \ldots, K$. Then we can estimate the entries of the probability matrix B as

$$\hat{B}_{kl} = \frac{\sum_{(i,j)\in\Omega} A_{ij} 1(\hat{c}_i = k, \hat{c}_j = l)}{\hat{n}^{\Omega}_{kl}} \tag{3.7}$$

where

$$\hat{n}^{\Omega}_{kl} = \begin{cases} |(i, j) \in \Omega : \hat{c}_i = k, \hat{c}_j = l| & \text{if } k \neq l \\ |(i, j) \in \Omega : i < j, \hat{c}_i = \hat{c}_j = k| & \text{if } k = l. \end{cases}$$

Under DCSBM, the probability matrix can be estimated similarly to Karrer and Newman [2011], Zhao et al. [2012] and Joseph and Yu [2016] via the Poisson approximation, letting

$$\hat{O}^*_{kl} = \sum_{(i,j)\in\Omega} A_{ij} 1(\hat{c}_i = k, \hat{c}_j = l) \tag{3.8}$$

and setting

$$\hat{\theta}_i = \frac{\sum_{j:(i,j)\in\Omega} A_{ij}}{\sum_{k=1}^{K} \hat{O}^*_{\hat{c}_i,k}} \ , \quad \hat{P}_{ij} = \hat{\theta}_i \hat{\theta}_j \hat{O}^*_{\hat{c}_i \hat{c}_j} / p \ . \tag{3.9}$$

The probability estimate $\hat{P}$ is scaled by p to reflect missing edges, which makes it slightly different from the estimator for the fully observed DCSBM [Karrer and Newman, 2011]. This rescaling happens automatically in the SBM estimator (3.7) since the sums in both the numerator and the denominator range over Ω only.

Finally, we need to specify a loss function to be evaluated on the held-out set. Natural loss functions for these models are either the squared error loss

$$L_2(A, \hat{A}) = \sum_{i<j,(i,j)\in\Omega^{\perp}} (A_{ij} - \hat{A}_{ij})^2,$$

or, to match the maximum likelihood estimators of parameters, the binomial deviance function

$$L_d(A, \hat{A}) = -\sum_{i<j,(i,j)\in\Omega^\perp} \left[A_{ij} \log(\hat{A}_{ij}) - (1 - A_{ij}) \log(1 - \hat{A}_{ij}) \right].$$

In practice, we observed that the L_2 loss works slightly better for model selection under both SBM and DCSBM.

The model selection question for block models includes the choice of SBM vs DCSBM and the choice of K. Suppose we want to select between SBM and DCSBM, with the number of communities ranging from 1 to $K_{\max}$. The candidate set of models in Algorithm III.1 is then $\mathcal{C} = \{\text{SBM-}K, \text{DCSBM-}K,\ K = 1, \ldots, K_{\max}\}$ where the number after the model name is the number of communities. The ECV algorithm for block model selection is summarized below as Algorithm III.7.

Algorithm III.7. Input: an adjacency matrix A, the largest number of communities to consider $K_{\max}\}$, the training proportion p, and the number of replications N.

1. For $m = 1, \ldots, N$
   (a) Randomly choose a subset of node pairs Ω, selecting each pair (i, j), $i < j$ independently with probability p, and adding (j, i) if (i, j) is selected.
   (b) For $K = 1, \ldots, K_{\max}$,
      i. Apply matrix completion to (A, Ω) with rank constraint K to obtain $\hat{A}_K$.
      ii. Run spectral clustering on $\hat{A}_K$ to obtain the estimated SBM membership vector $\hat{\boldsymbol{c}}^{(m)}_{1,K}$, and spherical spectral clustering to obtain the estimated DCSBM $\hat{\boldsymbol{c}}^{(m)}_{2,K}$.
      iii. Estimate the two models' probability matrices $\hat{A}^{(m)}_{1,K}$, $\hat{A}^{(m)}_{2,K}$ based on $\hat{\boldsymbol{c}}^{(m)}_{1,K}$, $\hat{\boldsymbol{c}}^{(m)}_{2,K}$ and evaluate the corresponding losses $L^{(m)}_{q,K}$, $q = 1, 2$ by applying

the loss function L with the estimated parameters to $A_{ij}, (i,j) \in \Omega^{\perp}$.

2. Let $L_{q,K} = \frac{1}{N}\sum_{m=1}^{N} L_{q,K}^{(m)}$. Return $(\hat{q}, \hat{K}) = \arg\min_{q=1,2} \min_{K=1,\ldots,K_{\max}} L_{q,K}$ as the best model (with $\hat{q} = 1$ indicating SBM and $\hat{q} = 2$ indicating DCSBM).

As a special case, one can also select just the number of communities K under the SBM (or DCSBM), a task recently considered by Latouche et al. [2012], McDaid et al. [2013], Bickel and Sarkar [2016], Lei [2016], Saldana et al. [2014], Wang et al. [2017], Chen and Lei [2017], Le and Levina [2015].

Theorem III.6 can be made more informative under the SBM and DCSBM, thanks to the many available results under these models. Specifically for SBM, we make the following standard assumption:

Assumption III.8. *The probability matrix $B^{(n)} = \rho_n B_0$, where B_0 is a fixed $K \times K$ symmetric nonsingular matrix with all entries in $[0,1]$ and K is a fixed number. Therefore the expected node degree is of the order $\lambda_n = n\rho_n$. Furthermore, there exists a constant $\gamma > 0$ such that $\min_k n_k > \gamma n$ where $n_k = |\{i : c_i = k\}|$.*

Many different versions of K-means can be used in spectral clustering. Here we state the result for the version of K-means used by Lei and Rinaldo [2014].

Proposition III.9 (Community recovery for each ECV split under the SBM)**.** *Let A be the adjacency matrix of a network generated from a SBM satisfying III.8 with K blocks, and $M = \mathbb{E}A$. Let $\hat{A}$ be the recovered adjacency matrix in (3.2). Assume the expected node degree $\lambda_n \geq C\log(n)$. Let $\hat{\mathbf{c}}$ be the output of spectral clustering on $\hat{A}$. Then $\hat{\mathbf{c}}$ coincides with the true $\mathbf{c}$ on all but $O(n\lambda_n^{-1})$ nodes (up to a permutation of block labels), with probability tending to one.*

To state an analogous result for the DCSBM, we need one more standard assumption on the degree parameters, similar to Jin [2015], Lei and Rinaldo [2014], Chen and Lei [2017].

Assumption III.10. $\min_i \theta_i \geq \theta_0$ *for some constant* $\theta_0 > 0$ *and* $\sum_{i:c_i=k} \theta_i = 1$ *for all* $k \in [K]$.

Proposition III.11 (Community recovery for each ECV split under the DCSBM)**.** *Let* A *be an adjacency matrix from a DCSBM satisfying III.8 and III.10 with* K *blocks, and* $M = \mathbb{E}A$. *Let* $\hat{A}$ *be the recovered adjacency matrix in* (3.2). *Assume the expected node degree* $\lambda_n \geq C \log(n)$. *Let* $\hat{\boldsymbol{c}}$ *be the output of spherical spectral clustering on* $\hat{A}$. *Then* $\hat{\boldsymbol{c}}$ *coincides with the true* $\boldsymbol{c}$ *on all but* $O(n\lambda_n^{-1/2})$ *nodes (up to a permutation of block labels), with probability tending to one.*

Comparison with cross-validation of Chen and Lei [2017]

The network cross-validation (NCV) algorithm by Chen and Lei [2017] was introduced explicitly for the purpose of model selection in block models, and thus it is of interest to compare with ours. The NCV algorithm first splits nodes at random into two groups $\mathcal{N}_1$ and $\mathcal{N}_2$, and then trains on pairs (i, j) corresponding to $i \in \mathcal{N}_1$ and $j \in \mathcal{N}_1 \cup \mathcal{N}_2$ are arranged into a rectangular matrix. The right singular vectors of this matrix are passed on to either spectral clustering for SBM or spherical spectral clustering for DCSBM to estimate node labels, with the same theoretical guarantees as ECV. The SBM model parameters can be estimated by standard estimators. However, standard estimators of DCSBM model parameters cannot be easily extended to a rectangular matrix, so a modified estimator is proposed in Chen and Lei [2017]. The node pairs (i, j) corresponding to $i, j \in \mathcal{N}_2$ are then used as a test set to evaluate the loss function and choose the best model.

The ECV is more general than the NCV, since it works with any low-rank approximation and does not rely on block structure in the data, and it also works for both directed and undirected networks, whereas NCV is for undirected networks only. As NCV does not recover the adjacency matrix, it cannot be used to evaluate methods that are based on certain transformations of the adjacency matrix, such as the problem in Section 3.3.3. Further, ECV is less likely than NCV to create isolated nodes in the training sample, which are useless in model fitting. To see this, consider the following simple calculation: assume that a given node i has degree d, and that all its d neighbors also have degree d. Suppose we apply NCV by deleting n/N rows of A, and hold out a matching number of entries at random via ECV. Let p_n and p_e be the probabilities that all neighbors of the given node i are assigned to the held-out set by NCV and ECV, respectively. Then a simple combinatorial calculation combined with Stirling's formula shows that for large n, the ratio of the two probabilities is approximately

$$p_e/p_n \approx e^{d/N^2}/N^d.$$

This ratio achieves its maximum 0.64 when $N = 2$ and $d = 1$ and can be much smaller if $N > 2, d > 1$. Table 3.1 shows p_e/p_n when $n = 300$ and $N = 3$, for different d.

Table 3.1: Ratio between p_e and p_n for $n = 300$, $N = 3$, and different d, where p_e and p_b are the probabilities that a node with d neighbors becomes isolated in the training set in ECV and NCV, respectively.

d	1	2	3	4	5
p_e/p_n	0.339	0.113	0.035	0.012	0.004

Although this example is a simplified calculation for one fixed node, it shows an important advantage of ECV over NCV under the block models, since isolated nodes are assigned to blocks randomly and decrease overall accuracy. In simulations, we also observed that ECV is much less likely to result in isolated nodes than NCV.

3.3.3 Parameter tuning in regularized spectral clustering

Regularized spectral clustering has been proposed to improve performance of spectral clustering in sparse networks, but regularization itself frequently depends on a tuning parameter that has to be selected correctly in order to achieve the improvement. Several different regularizations have been proposed and analyzed [Chaudhuri et al., 2012, Amini et al., 2013]. ECV can be used to tune all of them, but for concreteness here we focus on the proposal by Amini et al. [2013], who replace the usual normalized graph Laplacian $L = D^{-1/2}AD^{-1/2}$, where D is the diagonal matrix of node degrees, by the Laplacian computed from the regularized adjacency matrix

$$A_\tau = A + \tau \cdot \hat{d}/n\mathbf{1}\mathbf{1}^T \tag{3.10}$$

where $\hat{d}$ is the average node degree and τ is a tuning parameter, typically within $[0, 1]$. The scale of the multiplier is motivated by theoretical results under the SBM [Gao et al., 2017, Le et al., 2017]. This regularization is known to improve concentration [Le et al., 2017], but also the larger τ is, the more noise it adds, and thus we aim to select the best value of τ that balances these two effects. Joseph and Yu [2016] proposed a data-driven way to select τ called DKest based on theoretical bounds obtained under the SBM and the DCSBM. Using ECV is an alternative general data-driven way of selecting τ which does not rely on model assumptions.

Choosing a good τ is expected to give good clustering accuracy, defined as proportion of correctly clustered nodes under the best cluster matching permutation,

$$\max_{\hat{\boldsymbol{c}}^p \in \text{perm}(\hat{\boldsymbol{c}})} |\{i \in [n], \hat{c}_i^p = c_i\}|/n.$$

We can directly use Algorithm III.1 with the candidate set $\mathcal{C}$ being a grid of τ values and the matrix completion procedure applied to regularized partial adjacency matrices for each τ, as long as we can specify a loss function. Ideally, we would prefer a model-free loss function, applicable even when the block model does not hold. In general, choosing a loss function for cross-validation in clustering is difficult. While there is some work in the classical clustering setting [Tibshirani et al., 2001, Sugar and James, 2003, Tibshirani and Walther, 2005], it has not been discussed much in the network setting, and the loss function we propose next, one of a number of reasonable options, may be of independent interest.

For any cluster label vector $\boldsymbol{c}$, the set of node pairs $\mathcal{V} \times \mathcal{V}$ will be divided into $K(K+1)/2$ classes defined by $H(i,j) = (c_i, c_j)$. We treat each $H(i,j)$ as an unordered pair, since the network is undirected in spectral clustering. To compare two vectors of labels $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, we can compare their corresponding partitions H_1 and H_2 by computing co-clustering difference (CCD) or normalized mutual information (NMI) between them [Yao, 2003]. For instance, the co-clustering matrix for H_1 is defined to be the $n^2 \times n^2$ matrix G_1 such that $G_{1,(j-1)n+i,(q-1)n+p} = \mathbf{I}\{H_1(i,j) = H_1(p,q)\}$, reflecting whether or not two edges are in the same partition of H_1. Then the CCD between H_1 and H_2 is defined as the squared Frobenius norm of the difference between the two co-clustering matrices

$$\mathrm{CCD}(H_1, H_2) = \|G_1 - G_2\|_F^2/2.$$

We apply this measure to choose the tuning parameter τ as follows: for each split $m = 1, 2, \cdots, N$ of ECV and each candidate value of τ, we complete the adjacency matrix after removing the held-out entries and estimate cluster labels $\hat{c}_\tau^{(m)}$ and the corresponding $\hat{H}_\tau^{(m)}$ by regularized spectral clustering on the completed matrix with the candidate value of τ. We also compute $\hat{H}_\tau$, the partition corresponding to regularized spectral clustering on the full adjacency matrix with the same value of τ.

Then we choose τ by comparing these partitions constrained to the held-out set,

$$\hat{\tau} = \arg\min_{\tau \in \mathcal{C}} \sum_{m=1}^{N} \mathrm{CCD}(\hat{H}^{(m)}_{\tau,\Omega_m^\perp}, \hat{H}_{\tau,\Omega_m^\perp}).$$

Intuitively, if τ is a good value, the label vectors that generate $\hat{H}^{(m)}_{\tau,\Omega_m^\perp}$ and $\hat{H}_{\tau,\Omega_m^\perp}$ should both be close to the truth, and so the co-clustering matrices should be similar; if τ is a bad choice, then both label vectors will contain more errors, likely to be non-matching, and the corresponding CCD will be larger.

3.3.4 Tuning graphon model estimation method

Graphon (or probability matrix) estimation is another general task which often relies on tuning parameters that can be determined by cross-validation. Zhang et al. [2015] proposed a method called "neighborhood smoothing" to estimate M instead of f under the assumption that f is a piecewise Lipschitz function, avoiding the measure-preserving transformation ambiguity. They showed their method achieves a nearly optimal rate while requiring only polynomial complexity for computation (optimal methods are exponential). The method depends on a tuning parameter h which controls the degree of smoothing. The theory suggests $h = \tau\sqrt{\frac{\log n}{n}}$ for a constant τ.

This is a setting where we have no reason to assume a known rank of the true probability matrix and M does not have to be low rank. However, for a smooth graphon function a low rank matrix can approximate M reasonably well [Chatterjee, 2015]. The ECV procedure under the graphon model now has to select the best rank for its internal matrix completion step. Specifically, in each split, we can run the rank estimation procedure discussed in Section 3.3.1 to estimate the best rank for approximation and the corresponding $\hat{A}$ as the input for the neighborhood smoothing algorithm. The selected tuning parameter is again the one minimizing the average prediction error.

3.3.5 Stability selection

Stability selection [Meinshausen and Bühlmann, 2010] was proposed as a general method to reduce noise by repeating model selection many times over random splits of the data and keeping only the features that are selected in the majority of splits; any cross-validation procedure can benefit from stability selection since it relies on random data splits. An additional benefit of stability selection in our context is increased robustness to the choice of p and N (see Appendix B.2.3). Chen and Lei [2017] applied this idea to NCV as well, repeating the procedure multiple times and choosing the most frequently selected model. We use the same strategy for ECV (and NCV in comparisons), choosing the model selected most frequently out of 20 replications. When we need to select a numerical parameter rather than a model, we can also average the values selected over the 20 replications (and round to an integer if needed, say for the number of communities). Overall, picking the most frequent selection is more robust to different tasks, though picking the average may work better in some situations. More details are given in Section 3.4.

3.4 Numerical performance evaluation

In this section, we use extensive simulation studies to demonstrate the performance of ECV for the tasks discussed in Section 3.3: estimating rank for a general low-rank network model, model selection for block models (SBM vs DCSBM and the choice of K), tuning regularized spectral clustering and tuning neighborhood smoothing algorithm for graphon models.

The two internal parameters we need to set for the ECV are the selection probability p and the number of repetitions N. Our numerical experiments suggest (see

Appendix B.2.3) that the accuracy is stable for $p \in (0.85, 1)$ and the choice of N does not have much effect after applying stability selection, In all of our examples, we take $p = 0.9$ and $N = 3$, as a fair comparison with the recommended configuration for the NCV method of Chen and Lei [2017] under block models. This configuration seems to work well in all settings.

3.4.1 Rank estimation for general directed networks

Here we demonstrate the generality of ECV on the task of selecting the best rank for a network model for directed networks. There are no obvious competing methods for this task, since the NCV is designed for the block model family only. Assume $P = XY^T$ where $X, Y \in \mathbb{R}^{n\times K}$ are such that $P_{ij} \in [0, 1]$. This can be viewed as an instance of a directed random dot product graph model [Young and Scheinerman, 2007], with K the dimension of its latent space. We can use the ECV with either the AUC loos or the SSE loss for model selection in this case, again with either of the two stability selection methods. In simulations, we generate two $n \times K$ matrices S_1 and S_2 with each element drawn independently from the uniform distribution on $(0, 1)$, and set $P = S_1S_2^T$. We then normalize to [0, 1] by setting $P = (\max_{i,j} P_{ij})^{-1}P$ and generate the network adjacency matrix A with independent Bernoulli edges and $\mathbb{E}A = P$.

We fix $K = 3$ or $K = 5$ in the model and vary the number of nodes n. The candidate set is $K \in \{1, 2, \cdots, 8\}$. Table 3.2 show the distribution of estimated $\hat{K}$ under various settings. When the sample size is sufficiently large, all versions of ECV can estimate K well. The AUC-based ECV is always more accurate that the SSE-based ECV, and works better at smaller sample sizes. The estimation is quite stable for this task so stability selection does not offer much improvement.

K	n	method	$\hat{K}$: 1	2	3	4	5	6	7	8
3	600	ECV-AUC	42	61	97	-	-	-	-	-
		ECV-AUC-mode	40	61	99	-	-	-	-	-
		ECV-AUC-avg	42	61	97	-	-	-	-	-
		ECV-SSE	144	42	14	-	-	-	-	-
		ECV-SSE-mode	157	39	4	-	-	-	-	-
		ECV-SSE-avg	144	55	1	-	-	-	-	-
3	750	ECV-AUC	-	1	199	-	-	-	-	-
		ECV-AUC-mode	-	1	199	-	-	-	-	-
		ECV-AUC-avg	-	1	199	-	-	-	-	-
		ECV-SSE	11	59	130	-	-	-	-	-
		ECV-SSE-mode	6	52	142	-	-	-	-	-
		ECV-SSE-avg	5	67	128	-	-	-	-	-
3	900	ECV-AUC	-	-	3	-	-	-	-	-
		ECV-AUC-mode	-	-	3	-	-	-	-	-
		ECV-AUC-avg	-	-	3	-	-	-	-	-
		ECV-SSE	-	4	196	-	-	-	-	-
		ECV-SSE-mode	-	2	198	-	-	-	-	-
		ECV-SSE-avg	-	2	198	-	-	-	-	-
5	1500	ECV-AUC	39	20	26	33	82	-	-	-
		ECV-AUC-mode	31	20	28	33	88	-	-	-
		ECV-AUC-avg	39	20	26	33	82	-	-	-
		ECV-SSE	133	34	20	11	2	-	-	-
		ECV-SSE-mode	134	39	13	10	4	-	-	-
		ECV-SSE-avg	117	52	18	13	-	-	-	-
5	1800	ECV-AUC	-	-	1	3	196	-	-	-
		ECV-AUC-mode	-	-	1	3	196	-	-	-
		ECV-AUC-avg	-	-	1	3	196	-	-	-
		ECV-SSE	10	10	29	46	105	-	-	-
		ECV-SSE-mode	9	9	31	28	123	-	-	
		ECV-SSE-avg	4	13	30	47	106	-	-	-
5	2000	ECV-AUC	-	-	-	-	200	-	-	-
		ECV-AUC-mode	-	-	-	-	200	-	-	-
		ECV-AUC-avg	-	-	-	-	200	-	-	-
		ECV-SSE	-	-	5	14	181	-	-	-
		ECV-SSE-mode	-	-	6	11	183	-	-	
		ECV-SSE-avg	-	-	5	17	178	-	-	-

Table 3.2: Frequency of estimated rank values in 200 replications.

3.4.2 Model selection under block models

This task closely follows the evaluation setting for NCV from Chen and Lei [2017]. We investigate the performance of ECV and other relevant competing methods in two specific tasks:

1. Overall model selection: choosing the model (SBM or DCSBM) and the number of communities K simultaneously.

2. Estimating the number of communities: choosing K when the true model type (SBM or DCSBM) is known.

Overall model selection

The four methods compared on this task are ECV with L_2 loss (ECV-l2), the stable version of ECV where the most frequent selection of 20 independent repetitions of ECV-l2 is returned (ECV-l2-mode), and the corresponding versions of the NCV procedure (NCV-l2, NCV-l2-mode). We only show the results from using the L_2 loss for model selection since we observed it works better than binomial deviance for both ECV and NCV. The performance using binomial deviance as loss can be found in Appendix B.2.1.

The setting for all simulated networks in this section is as follows. For the DCSBM, we first sample 300 values from the power law distribution with the lower bound 1 and scaling parameter 5, and then set the node degree parameters θ_i, $i = 1, \cdots, n$ by randomly and independently choosing one of these 300 values. For the SBM, we set $\theta_i = 1$ for all i. Let $\pi \propto (1, 2^t, \cdots, K^t)$ be the proportions of nodes in the K communities; t controls the size balance (when $t = 0$ the communities have equal sizes). Let $B_0 = (1 - \beta)I + \beta \mathbf{1}\mathbf{1}^T$ and $B \propto \Theta B_0 \Theta$, so that β is the out-in ratio (the ratio of between-block probability and within-block probability of edge). The scaling is selected so that the average node degree is λ. We consider several configurations of size and the number of communities: $(n = 600, K = 3)$, $(n = 600, K = 5)$ and $(n = 1200, K = 5)$. For each configuration, we then vary three aspects of the model:

1. Sparsity: set the expected average degree λ to 15, 20, 30, or 40, fixing $t = 0$ and $\beta = 0.2$.
2. Community size: set t to 0, 0.25, 0.5, or 1, fixing $\lambda = 40$ and $\beta = 0.2$.
3. Out-in ratio: set β to 0, 0.25, or 0.5, fixing $\lambda = 40$ and $t = 0$.

All results are based on 200 replications.

The candidate model set contains both the SBM and the DCSBM with the number of communities varying from 1 to 8. Following Chen and Lei [2017], we evaluate performance on two different model selection tasks: choosing both the model (SBM vs. DCSBM) and the number of communities K simultaneously, and choosing K when the true model is known.

K	n	λ	t	β	ECV-l2	ECV-l2-mode	NCV-l2	NCV-l2-mode
3	600	15	0	0.2	0.73	0.87	0.00	0.00
		20	0	0.2	0.97	0.99	0.02	0.00
		30	0	0.2	1.00	1.00	0.43	0.40
		40	0	0.2	1.00	1.00	0.88	0.98
5	600	15	0	0.2	0.49	0.58	0.00	0.00
		20	0	0.2	0.90	0.95	0.00	0.00
		30	0	0.2	0.99	1.00	0.05	0.01
		40	0	0.2	0.99	1.00	0.27	0.24
5	1200	15	0	0.2	0.67	0.76	0.00	0.00
		20	0	0.2	0.99	0.99	0.00	0.00
		30	0	0.2	1.00	1.00	0.04	0.00
		40	0	0.2	1.00	1.00	0.41	0.33
3	600	40	0	0.2	1.00	1.00	0.88	0.98
		40	0.25	0.2	1.00	1.00	0.90	0.97
		40	0.5	0.2	1.00	1.00	0.92	0.97
		40	1	0.2	0.70	0.79	0.42	0.46
5	600	40	0	0.2	0.99	1.00	0.27	0.24
		40	0.25	0.2	0.98	1.00	0.28	0.29
		40	0.5	0.2	0.77	0.79	0.18	0.17
		40	1	0.2	0.11	0.06	0.05	0.00
5	1200	40	0	0.2	1.00	1.00	0.41	0.33
		40	0.25	0.2	1.00	1.00	0.44	0.39
		40	0.5	0.2	0.81	0.83	0.21	0.16
		40	1	0.2	0.10	0.06	0.00	0.06
3	600	40	0	0.1	1.00	1.00	0.99	1.00
		40	0	0.2	1.00	1.00	0.88	0.98
		40	0	0.5	0.95	0.97	0.00	0.00
5	600	40	0	0.1	1.00	1.00	0.79	0.96
		40	0	0.2	0.99	1.00	0.27	0.24
		40	0	0.5	0.00	0.00	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	0.90	0.99
		40	0	0.2	1.00	1.00	0.41	0.33
		40	0	0.5	0.00	0.00	0.00	0.00

Table 3.3: Overall model selection by ECV and NCV (fraction correct out of 200 replications). The true model is the DCSBM.

Table 3.3 shows the fraction (out of the 200 replications) of times the correct model was selected when the true model is the DCSBM. Over all settings, stability selection improves performance as long as the cross-validation itself is working reasonably well. This is expected since stability selection is only a variance reduction step, and it cannot help if the original procedure is not working. The NCV works well in easier settings (smaller number of communities, the network is relatively dense, communities are balanced, the out-in ratio is small). As the problem becomes harder, the NCV quickly loses accuracy in model selection. On the other hand, the ECV gives better selection than NCV in all cases, and in many settings the difference is very large. For instance, when $K = 5, n = 1200, \lambda = 30, \beta = 0.2$ with balanced communities, NCV completely fails (0% correct) while ECV gives the correct answer 100% of the time.

Table 3.4 shows the corresponding results when the underlying true model is the SBM. The task is easier under the SBM as the model is simpler, but the general pattern is very similar to the DCSBM setting. Stability selection clearly improves performance and the ECV performs better than NCV overall.

Selecting the number of communities

When the model (SBM or DCSBM) is known or assumed, there are multiple methods available for selecting the number of communities K which can be included in comparisons along with general cross-validation methods. For this task, we compare the following cross-validation procedures: the previously mentioned ECV-l2, NCV-l2, and the model-free ECV with the SSE and the AUC as loss functions, described in Section 3.3.1 (ECV-SSE and ECV-AUC, respectively). For any cross-validation method, we can further use stability selection by either picking the most frequent selection or picking the closest integer to the average selection. For instance, for ECV with the L_2 loss, we call the two stabilized versions ECV-l2-mode and ECV-l2-avg, respectively. Additionally, we include two methods specifically for choosing K under the block models, which we would expect to be at least as accurate as cross-validation considering that they use the true model and cross-validation does

K	n	λ	t	β	ECV-l2	ECV-l2-mode	NCV-l2	NCV-l2-mode
3	600	15	0	0.2	1.00	1.00	0.99	1.00
		20	0	0.2	1.00	1.00	1.00	1.00
		30	0	0.2	1.00	1.00	0.99	1.00
		40	0	0.2	1.00	1.00	1.00	1.00
5	600	15	0	0.2	0.81	0.88	0.71	0.86
		20	0	0.2	1.00	1.00	0.98	1.00
		30	0	0.2	1.00	1.00	0.98	1.00
		40	0	0.2	0.99	1.00	0.98	1.00
5	1200	15	0	0.2	0.98	0.98	0.91	0.96
		20	0	0.2	1.00	1.00	0.99	1.00
		30	0	0.2	1.00	1.00	0.96	1.00
		40	0	0.2	1.00	1.00	0.97	1.00
3	600	40	0	0.2	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	0.99	1.00
		40	0.5	0.2	1.00	1.00	0.97	1.00
		40	1	0.2	0.73	0.76	0.34	0.43
5	600	40	0	0.2	0.99	1.00	0.98	1.00
		40	0.25	0.2	1.00	1.00	0.95	1.00
		40	0.5	0.2	0.82	0.86	0.64	0.78
		40	1	0.2	0.06	0.01	0.17	0.10
5	1200	40	0	0.2	1.00	1.00	0.97	1.00
		40	0.25	0.2	1.00	1.00	0.98	1.00
		40	0.5	0.2	0.93	0.94	0.73	0.89
		40	1	0.2	0.01	0.01	0.21	0.06
3	600	40	0	0.1	1.00	1.00	0.98	1.00
		40	0	0.2	1.00	1.00	1.00	1.00
		40	0	0.5	0.92	0.96	0.83	0.96
5	600	40	0	0.1	0.99	1.00	0.97	1.00
		40	0	0.2	0.99	1.00	0.98	1.00
		40	0	0.5	0.00	0.00	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	0.98	1.00
		40	0	0.2	1.00	1.00	0.97	1.00
		40	0	0.5	0.00	0.00	0.00	0.00

Table 3.4: Overall model selection by ECV and NCV (fraction correct out of 200 replications). The true model is the SBM.

not. The method of Wang et al. [2017] is a BIC-type criterion (LR-BIC) based on an asymptotic analysis of the likelihood ratio statistic. Another BIC-type method proposed by Saldana et al. [2014] is based on the composite likelihood (CL-BIC) but it is computationally infeasible for networks with more than 1000 nodes (using the implementation on the authors' website) and it was less accurate than LR-BIC in our experiments on smaller networks, so we omit it from comparisons. From the class of eigenvalues-based methods proposed by Le and Levina [2015], we include the variant based on the Bethe-Hessian matrix with moment correction (BHmc). Due to the large number of methods, we first compare just the cross-validation methods, ECV-l2, NCV-l2, ECV-AUC, ECV-SSE, and their stabilized versions, and then we compare the best of the cross-validation methods with the other two. The results when the true model is the DCSBM are included in this section; the corresponding results for the SBM are given in Appendix B.2.2.

Table 3.5 shows the comparison between the cross-validation methods when we vary the average network degree with fixed $\beta = 0.2$ and balanced communities. Both stability selection methods improve the model selection accuracy of the ECV, and stabilization by average is better for all versions of the ECV. On the other hand, for the NCV the most frequently selected K is typically more accurate than the rounded average. Further, all the variants of the ECV work as well as or better than the NCV in all configurations. For example, when $n = 1200, K = 5, \lambda = 15$, the ECV accuracy is in the range 0.83-0.85 (for different versions), while the NCV completely fails. The model-free ECV-AUC has similar performance to ECV-l2 on this task, but it cannot be used to select between the SBM and the DCSBM. The ECV-SSE version is slightly inferior but still works much better than the NCV.

Tables 3.6 and 3.7 compare the same methods when we vary t fixing λ and β, and vary β while fixing λ and t, respectively. The pattern is very similar to Table 3.5.

Setting			ECV-l2			NCV-l2			ECV-AUC			ECV-SSE		
K	n	λ		mode	avg		mode	avg		mode	avg		mode	avg
3	600	15	0.99	1.00	1.00	0.82	0.99	0.94	0.99	1.00	1.00	0.99	1.00	1.00
		20	1.00	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		30	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		40	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5	600	15	0.57	0.60	0.72	0.01	0.00	0.00	0.55	0.59	0.68	0.33	0.34	0.68
		20	0.92	0.95	0.96	0.43	0.67	0.36	0.93	0.96	0.99	0.86	0.91	0.99
		30	0.99	1.00	1.00	0.76	0.99	0.91	1.00	1.00	1.00	1.00	1.00	1.00
		40	0.99	1.00	1.00	0.76	0.98	0.89	1.00	1.00	1.00	1.00	1.00	1.00
5	1200	15	0.74	0.79	0.85	0.01	0.00	0.00	0.73	0.79	0.83	0.22	0.26	0.83
		20	0.99	0.99	1.00	0.76	0.95	0.67	0.98	0.99	0.99	0.94	0.97	0.99
		30	1.00	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		40	1.00	1.00	1.00	0.96	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 3.5: The rate of correctly estimating the number of communities (out of 200 replications) when varying the network average degree and fixing $t = 0$, $\beta = 0.2$. The true model is the DCSBM.

Next, we compare the best of cross-validation methods (ECV-l2-avg, NCV-l2-mode, ECV-AUC-avg) with the model-based methods LR-BIC and BHmc, with

Setting			ECV-l2			NCV-l2			ECV-AUC			ECV-SSE		
K	n	t		mode	avg		mode	avg		mode	avg		mode	avg
3	600	0	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.25	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		1	0.70	0.79	0.67	0.49	0.48	0.49	0.85	0.86	0.90	0.82	0.83	0.90
5	600	0	0.99	1.00	1.00	0.76	0.98	0.89	1.00	1.00	1.00	1.00	1.00	1.00
		0.25	0.98	1.00	1.00	0.64	0.95	0.84	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	0.77	0.80	0.80	0.36	0.55	0.70	0.80	0.80	0.83	0.74	0.77	0.83
		1	0.11	0.06	0.07	0.06	0.01	0.01	0.03	0.01	0.01	0.01	0.00	0.01
5	1200	0	1.00	1.00	1.00	0.96	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.25	1.00	1.00	1.00	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	0.81	0.83	0.83	0.60	0.64	0.66	0.86	0.89	0.91	0.74	0.74	0.91
		1	0.10	0.06	0.07	0.01	0.00	0.00	0.04	0.01	0.01	0.00	0.00	0.01

Table 3.6: The rate of correctly estimating the number of communities (out of 200 replications) when varying t and fixing $\lambda = 40$, $\beta = 0.2$. The true model is the DCSBM.

Setting			ECV-l2			NCV-l2			ECV-AUC			ECV-SSE		
K	n	β		mode	avg		mode	avg		mode	avg		mode	avg
3	600	0.1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.2	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	0.96	1.00	1.00	0.00	0.00	0.00	0.96	1.00	1.00	0.12	0.16	1.00
5	600	0.1	1.00	1.00	1.00	0.85	1.00	0.99	0.98	1.00	0.99	1.00	1.00	0.99
		0.2	0.99	1.00	1.00	0.76	0.98	0.89	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5	1200	0.1	1.00	1.00	1.00	0.95	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00
		0.2	1.00	1.00	1.00	0.96	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		0.5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 3.7: The rate of correctly estimating the number of communities (out of 200 replications) when varying β and fixing $\lambda = 40$, $t = 0$. The true model is the DCSBM.

results shown in Table 3.8. The LR-BIC and BHmc perform perfectly most of the time, and outperform cross-validation when K is large and the network is sparse (harder settings). This is expected since cross-validation is a general method and the other two rely on the true model; they also cannot be applied to any other tasks. It is also not clear how they behave under model misspecification (important given that in the real world not many networks follow exactly the SBM or the DCSBM), while cross-validation can still be expected to give reasonable results; in particular, the ECV selection can be interpreted as the optimal model from the block model family in terms of link prediction for the observed network.

Setting					Method				
K	n	λ	t	β	ECV-l2-avg	NCV-l2-mode	ECV-AUC-avg	LR-BIC	BHmc
3	600	15	0	0.2	1.00	0.99	1.00	1.00	1.00
		20	0	0.2	1.00	1.00	1.00	1.00	1.00
		30	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
5	600	15	0	0.2	0.72	0.00	0.59	1.00	1.00
		20	0	0.2	0.96	0.67	0.96	1.00	1.00
		30	0	0.2	1.00	0.99	1.00	1.00	1.00
		40	0	0.2	1.00	0.98	1.00	1.00	1.00
5	1200	15	0	0.2	0.85	0.00	0.79	1.00	1.00
		20	0	0.2	1.00	0.95	0.99	1.00	1.00
		30	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
3	600	40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.5	0.2	1.00	1.00	1.00	1.00	1.00
		40	1	0.2	0.67	0.48	0.86	1.00	1.00
5	600	40	0	0.2	1.00	0.98	1.00	1.00	1.00
		40	0.25	0.2	1.00	0.95	1.00	1.00	1.00
		40	0.5	0.2	0.80	0.55	0.80	1.00	0.99
		40	1	0.2	0.07	0.01	0.01	0.46	0.10
5	1200	40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.5	0.2	0.83	0.64	0.89	1.00	1.00
		40	1	0.2	0.07	0.00	0.01	0.45	0.12
3	600	40	0	0.1	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.5	0.98	0.00	0.96	1.00	1.00
5	600	40	0	0.1	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	0.98	1.00	1.00	1.00
		40	0	0.5	0.00	0.00	0.00	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.5	0.00	0.00	0.00	0.00	0.00

Table 3.8: The rate of correctly estimating the number of communities (out of 200 replications) for the best variant of each method. The true model is the DCSBM.

3.4.3 Tuning regularized spectral clustering

Another application of ECV discussed in Section 3.3.3 is choosing the tuning parameter in regularized spectral clustering. Here we test the ECV on this task on networks generated from the DCSBM under the setting described in Section 3.4.2, with $n = 600$, $K = 3$, a power law distribution for θ_i, balanced community sizes $\pi = (1/3, 1/3, 1/3)$, out-in ratio $\beta = 0.2$, and average degree $\lambda = 5$, since regularization is generally only relevant when the network is sparse. The candidate set for the tuning parameter τ is $\mathcal{C} = \{0.1, 0.2, \cdots, 1.9, 2\}$. Without regularization, at this level of sparsity spectral clustering works very poorly. We use the ECV procedure described in Section 3.3.3 as well as its two stabilized versions to select τ. We also report the accuracy for each fixed value of τ in $\mathcal{C}$ as well as the DKest estimator of τ proposed by Joseph and Yu [2016].

In the sparse setting, spectral clustering may occasionally suffer from bad local optima found by K-means. Thus we report the median clustering accuracy out of 200 replications, as well as its 95% confidence interval calculated by bootstrap. Figure 3.1 shows the confidences intervals for the median accuracy of regularized spectral clustering for all tuning strategies out of 200. Without regularization, the clustering accuracy is below 0.5 (not shown). The accuracy jumps up with regularization for small τ values, and decreases slowly as τ increases. All data-driven methods give close to optimal performance, with DKest and ECV-avg giving the best result, closely followed by ECV without stability selection and ECV-mode. Again, considering that DKest is a model-based method designed specifically for this purpose, and ECV is a generic tuning method, this is a good result for the ECV.

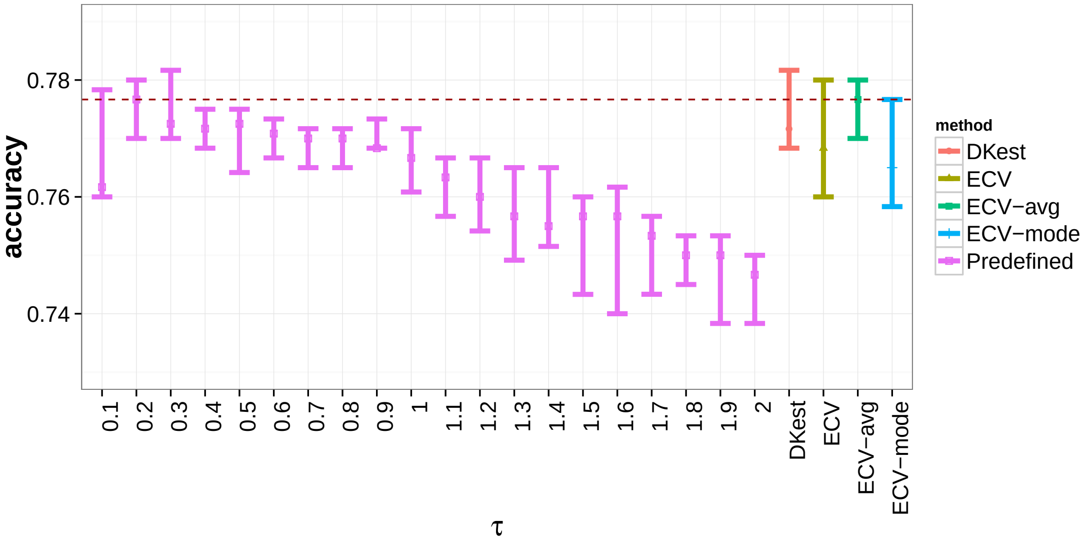
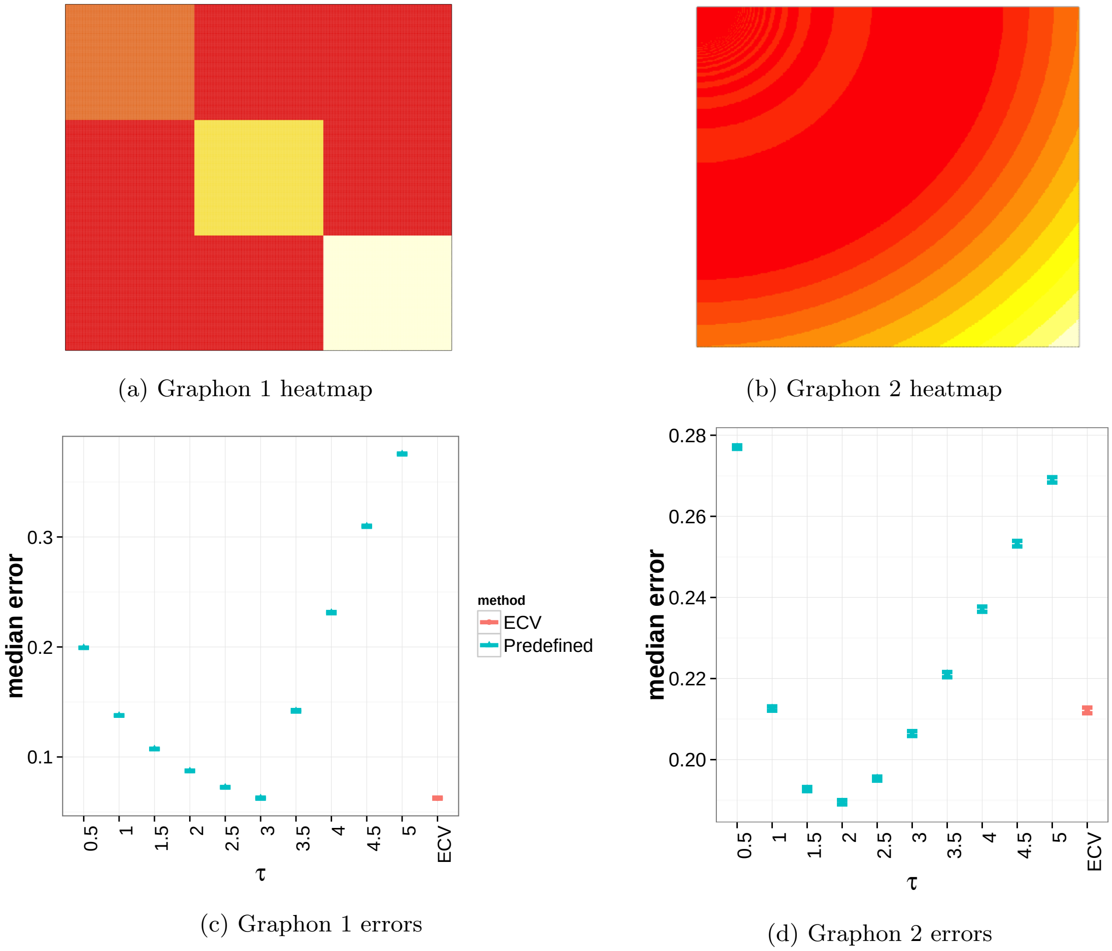


Figure 3.1: The median clustering accuracy for different fixed values of τ and for DKest and ECV tuning. The true model is DCSBM with $n = 600$, $K = 3$, $\lambda = 5$, $\beta = 0.2$ and $t = 0$.

3.4.4 Tuning nonparametric graphon estimation

In this example, we demonstrate the performance of ECV in tuning the neighborhood smoothing estimation for a graphon model. As discussed before, the theory in Zhang et al. [2015] suggests to use $h = \tau\sqrt{\frac{\log n}{n}}$ for a constant τ, but does not give a way to specify the value for τ. As we show next, the choice of the constant τ matters in practice and the ECV can be used to pick a good value.



(a) Graphon 1 heatmap

(b) Graphon 2 heatmap

(c) Graphon 1 errors

(d) Graphon 2 errors

Figure 3.2: Parameter tuning for piecewise constant graphon estimation.

The tuning procedure is very stable for the graphon problem and stability selection is unnecessary. Figure 3.2 shows the tuning results for two graphon examples taken from Zhang et al. [2015], both for networks with $n = 500$ nodes. Graphon 1 is a block model (though this information is never used), which is a piecewise constant function and P is low rank. Graphon 2 is a smoothly varying function which is not low rank; see Zhang et al. [2015] for more details. The errors are pictured as the median over 200 replications with a 95% confidence interval (calculated by bootstrap) of the normalized Frobenius error $\|\hat{P} - P\|_F / \|P\|_F$. For Graphon 1, which is low rank, the ECV works perfectly and picks the best τ of the candidate set most of the time. For Graphon 2, which is not low rank and therefore more challenging for a procedure that uses a low-rank approximation, the ECV does not always choose the very best τ, but still achieves a fairly competitive error rate by successfully avoiding the bad range for τ. This example illustrates that the constant can make a big difference for estimation error in this problem, and the ECV is successful at choosing it.

3.5 Community detection in a statistics citation network

This publicly available dataset provided by Ji et al. [2016] contains information about all papers (title, author, year, citations and DOI) published between 2003 and 2012 in four statistics journals considered top (Annals of Statistics, Biometrika, Journal of the American Statistical Association (Theory and Methods), and Journal of the Royal Statistical Society (Series B). This dataset was carefully constructed to resolve ambiguities and is relatively interpretable, at least to statisticians.

The dataset contains 3607 authors and 3248 papers. The citations of all the papers are available so we can construct the citation network between authors (as well as papers, but here we focus on authors as we are looking for research communities of people). We thus construct a weighted undirected network between authors, where the weight is the total number of their mutual citations. The largest connected component of the network contains 2654 authors. Thresholding the weight

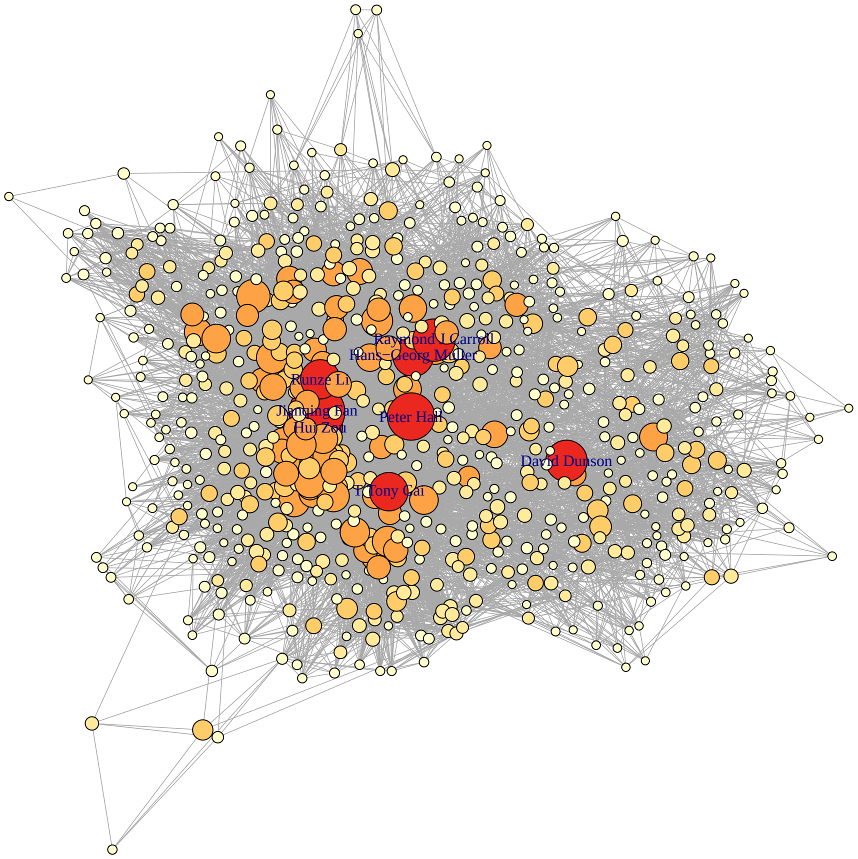


Figure 3.3: The core of statistician citation network. The network has 706 nodes with node citation count (ignoring directions) ranging from 15 to 703. The nodes sizes and colors indicate the citation counts and the nodes with larger citation counts are larger and darker.

to binary resulted in all methods for estimating K selecting an unrealistically large and uninterpretable value, suggesting the network is too complex to be adequately described by a binary block model. Since the weights are available and contain much more information than just the presence of an edge, we analyze the weighted network instead; seamlessly switching between binary and weighted networks is another advantage of the ECV.

Many real world networks have the core-periphery structure, and citation networks especially are likely to have this form. We focus on analyzing the core of the citation network, extracting it following the procedure proposed for the data set by Wang et al. [2016a]: delete nodes with less than 15 citations (in either direction) and their corresponding edges, and repeat until the network no longer changes. This results in a network with 706 authors shown in Figure 3.3. The individual node citation count ranges from 15 to 703 with a median 30.

	Interpretation [size]	Authors
1	high-dimensional inference (multiple testing, machine learning) [57]	T Tony Cai, Jiashun Jin, Larry Wasserman, Christopher Genovese, Bradley Efron, John D Storey, David L Donoho, Yoav Benjamini, Jonathan E Taylor, Joseph P Romano
2	high-dimensional inference (sparse penalties) [53]	Hui Zou, Ming Yuan, Yi Lin, Trevor J Hastie, Robert J Tibshirani, Xiaotong Shen, Jinchi Lv, Gareth M James, Hongzhe Li, Peter Radchenko
3	functional data analysis [52]	Hans-Georg Muller, Jane-Ling Wang, Fang Yao, Yehua Li, Ciprian M Crainiceanu, Jeng-Min Chiou, Alois Kneip, Hulin Wu, Piotr Kokoszka, Tailen Hsing
4	high-dimensional inference (theory and sparsity) [45]	Peter Buhlmann, Nicolai Meinshausen, Cun-Hui Zhang, Alexandre B Tsybakov, Emmanuel J Candes, Terence Tao, Marten H Wegkamp, Bin Yu, Florentina Bunea, Martin J Wainwright
5	high-dimensional covariance estimation [43]	Peter J Bickel, Ji Zhu, Elizaveta Levina, Jianhua Z Huang, Mohsen Pourahmadi, Clifford Lam, Wei Biao Wu, Adam J Rothman, Weidong Liu, Linxu Liu
6	Bayesian machine learning [41]	David Dunson, Alan E Gelfand, Abel Rodriguez, Michael I Jordan, Peter Muller, Gareth Roberts, Gary L Rosner, Omiros Papaspiliopoulos, Steven N MacEachern, Ju-Hyun Park
7	spatial statistics [41]	Tilmann Gneiting, Marc G Genton, Sudipto Banerjee, Adrian E Raftery, Haavard Rue, Andrew O Finley, Bo Li, Michael L Stein, Nicolas Chopin, Hao Zhang
8	biostatistics (machine learning) [40]	Donglin Zeng, Dan Yu Lin, Michael R Kosorok, Jason P Fine, Jing Qin, Guosheng Yin, Guang Cheng, Yi Li, Kani Chen, Yu Shen
9	suffiicent dimension reduction [39]	Lixing Zhu, R Dennis Cook, Bing Li, Chih-Ling Tsai, Liping Zhu, Yingcun Xia, Lexin Li, Liqiang Ni, Francesca Chiaromonte, Liugen Xue
10	high-dimensional inference (penalized methods) [38]	Jianqing Fan, Runze Li, Hansheng Wang, Jian Huang, Heng Peng, Song Xi Chen, Chenlei Leng, Shuangge Ma, Xuming He, Wenyang Zhang
11	Bayesian (general) [33]	Jeffrey S Morris, James O Berger, Carlos M Carvalho, James G Scott, Hemant Ishwaran, Marina Vannucci, Philip J Brown, J Sunil Rao, Mike West, Nicholas G Polson
12	high-dimensional theory and wavelets [33]	Iain M Johnstone, Bernard W Silverman, Felix Abramovich, Ian L Dryden, Dominique Picard, Richard Nickl, Holger Dette, Marianna Pensky, Piotr Fryzlewicz, Theofanis Sapatinas
13	mixed (causality + theory + Bayesian) [32]	James R Robins, Christian P Robert, Paul Fearnhead, Gilles Blanchard, Zhiqiang Tan, Stijn Vansteelandt, Nancy Reid, Jae Kwang Kim, Tyler J VanderWeele, Scott A Sisson
14	semiparametrics and nonparametrics [28]	Hua Liang, Naisyin Wang, Joel L Horowitz, Xihong Lin, Enno Mammen, Arnab Maity, Byeong U Park, Wolfgang Karl Hardle, Jianhui Zhou, Zongwu Cai
15	high-dimensional inference (machine learning) [27]	Hao Helen Zhang, J S Marron, Yufeng Liu, Yichao Wu, Jeongyoun Ahn, Wing Hung Wong, Peter L Bartlett, Michael J Todd, Amnon Neeman, Jon D McAuliffe
16	semiparametrics [24]	Peter Hall, Raymond J Carroll, Yanyuan Ma, Aurore Delaigle, Gerda Claeskens, David Ruppert, Alexander Meister, Huixia Judy Wang, Nilanjan Chatterjee, Anastasios A Tsiatis
17	mixed (causality + financial) [22]	Qiwei Yao, Paul R Rosenbaum, Yacine Ait-Sahalia, Yazhen Wang, Marc Hallin, Dylan S Small, Davy Paindaveine, Jian Zou, Per Aslak Mykland, Jean Jacod
18	biostatistics (survival, clinical trials) [22]	L J Wei, Lu Tian, Tianxi Cai, Zhiliang Ying, Zhezhen Jin, Peter X-K Song, Hui Li, Bin Nan, Hajime Uno, Jun S Liu
19	biostatistics - genomics [21]	Joseph G Ibrahim, Hongtu Zhu, Jiahua Chen, Amy H Herring, Heping Zhang, Ming-Hui Chen, Stuart R Lipsitz, Denis Heng-Yan Leung, Weili Lin, Armin Schwartzman
20	Bayesian (nonparametrics) [15]	Subhashis Ghosal, Igor Prunster, Antonio Lijoi, Stephen G Walker, Aad van der Vaart, Anindya Roy, Judith Rousseau, J H van Zanten, Richard Samworth, Aad W van der Vaart

Table 3.9: The 10 authors with largest total citation numbers (ignoring the direction) within 20 communities, as well as the community interpretations. The communities are ordered by size and authors within a community are ordered by mutual citation count.

Block models are not defined for weighted networks, but the Laplacian is still well-defined and so the spectral clustering algorithm for community detection can be applied. The model-free version ECV-SSE can be used to determine the number of communities. We apply the ECV-SSE procedure with $p = 0.9$ and $N = 3$ and repeat it 200 times, with the candidate values for K from 1 to 50. The stable version ECV-SSE-mode selects $K = 20$. We also used ECV to tune the regularization parameter for spectral clustering, as described in Section 3.3.3. It turns out the regularization does make the result more interpretable. We list the 20 communities in Table 3.9, with each community represented by 10 authors with the largest number of citations, along with subjective and tentative names we assigned to these communities. Note that the names are assigned based on the majority of authors' interests or area of contributions, and that it is based exclusively on data collected in the period 2003-2012, so people who have worked on many topics over many years tend to appear under the topic they devoted the most attention to in that time period. Many communities can be easily identified by their common research interests; high-dimensional inference, a topic that many people published on in that period of time, is subdivided into several sub-communities that are in themselves interpretable (communities 1, 2, 4, 5, 10, 12, 15). Overall, these groups are fairly easily interpretable to those familiar with the statistical literature of this decade.

3.6 Summary and future work

We have proposed a general framework for resampling networks based on, in a nutshell, leaving out adjacency matrix entries at random and using matrix completion to fill them back in before proceeding with the task at hand for the training data. While for specific problems like selecting the number of communities under the block models there are existing methods that work well, our proposal has the advantage of being general and competitive with specialized methods across the board. It relies on an approximately low rank assumption which we know to be reasonable for many real networks. However, if another structural assumption makes more sense for a given dataset, one can always replace the matrix completion method with something more appropriate for the situation, while the general principle remains the same. Under the low rank assumption on the underlying probability matrix, we showed that the completed matrix retains the same order of concentration around the truth as the full adjacency matrix; in practice, we expect the method to work well for approximately low rank structures as well.

The general scheme of leaving out entries at random followed by matrix completion can be useful in any resampling-based method, not just cross-validation. Establishing rigorous guarantees for bootstrap in this context is left for future work. Another direction we leave for future work is when there are additional node features available [Li et al., 2016c, Newman and Clauset, 2016]. The strategy we use for ECV could be modified to include resampling node features as well as edges in this context. Finally, the strategy may also prove useful in cross-validation on dynamic networks changing over time [Zhang et al., 2017, Rossetti and Cazabet, 2017], and in general any situation where one needs to create an artificial sample of networks based on a single observed network.

CHAPTER IV

A new community model for partially observed networks from surveys

4.1 Introduction

In this chapter, we will focus on community detection problem, which is only briefly mentioned in previous chapters. Community detection, the task of clustering nodes into groups with relatively homogeneous connection patterns, has been an intensively studied topic in network analysis [Fortunato, 2010, Goldenberg et al., 2010]. Community detection makes it possible to decompose a network into relatively homogenous parts and in many applications, can lead to new discoveries about the network nodes [Newman, 2006]. Many statistical network models with communities have now been proposed, from the simple stochastic block model [Holland et al., 1983] to complex extensions with mixed membership [Airoldi et al., 2008] or temporal evolution [Xu and Hero, 2013, Matias and Miele, 2017]. Such models can provide a rigorous statistical framework and theoretical performance guarantees [Rohe et al., 2011, Zhao et al., 2012], as well as lead to improved algorithms [Joseph and Yu, 2016, Gao et al., 2017, Le et al., 2017].

A practical difficulty in many empirical studies of networks arises from the data collection mechanism. Here we use the term network *survey* to refer to any situation where some edge information may be missing due to the data collection procedure.

This could include traditional surveys: in many social network studies, the network information is collected from a survey in which subjects are asked to name their friends [Michell and West, 1996, Harris, 2009]. Typically, a given number of slots is included for these nominations, which may cut off some friends from being named. More importantly, subjects may choose not to name all of their friends, for various reasons, and therefore we can think of observed edges as a subset of the true edges. More generally, a survey of a network in this general sense can result from networks collected by internet crawlers that only follow some of the paths for technical or time reasons, etc. In any of these situations, the missing edges may undermine validity or efficiency of standard network models.

In a sense, missing edges in networks can also be viewed as erroneous observations (a 0 instead of a 1), though the focus of the problem of denoising networks is slightly different. There has been a significant amount of work on denosing networks, which often considers both missing edges and falsely reported edges. Butts [2003] proposed a Bayesian method to evaluate how reliable an observed network is. Newman [2018a] proposed a link prediction framework to recover underlying networks without specific structures. Newman [2018b] further extends this path of work to a general framework to estimate networks under non-informative observational errors. For networks with communities, Guimerà and Sales-Pardo [2009] propose a Bayesian model and inference method to detect both missing and spurious edges. Martin et al. [2016] take a similar modeling strategy but assumed more flexible nonparametric error distributions for the potential observations errors.

The above models for noisy networks assume the missing mechanism is not related to the network structure, for instance communities. In some situations this assumption is reasonable, for instance for recording errors, but for a network resulting from a survey this assumption is hard to justify. For example, in a survey procedure, some individuals may prefer to nominate more of their friends from the same community while some others may randomly nominate some of their friends ignoring the community affiliations, even though the true underlying friendship distributed according to the community information. Generally, in a surveyed network the missing mechanism is potentially dependent on both of the communities and individual node characteristics, which needs different models from the network denoising methods. Recently, Le and Levina [2017] consider a related scenario where the missing mechanism depends on the community labels of the node pairs, but it uses a sample of networks from the same distribution, which is common in neuroimaging applications but not commonly available for friendship surveys.

Perhaps the most related work to ours is from Zhao et al. [2017], in which only one snapshot of the network is available and the observed network is a partially observed version of a true underlying network while the observation probability of an edge is a monotonic function of the true connection probability. However, Zhao et al. [2017] focus on the general link prediction task without specific structural assumptions such as communities and due to the generality of their framework, the method of Zhao et al. [2017] needs some additional node-wide similarity information on top of the network.

In this chapter, we propose a network model with communities we call *nomination stochastic block model* (NSBM) for directed networks. It can be used for community detection on a single network and aims to better model the networks resulting from surveys, with edge nomination mechanism driven by both communities and node-specific parameters. We propose computationally efficient algorithms for fitting this model, based on spectral clustering and the method of moments, and show statistical consistency for both community recovery and parameter estimation. We also propose a conditional surveyed network model, which can be more realistic in certain situations but is difficult for inference. However, our empirical results show that the proposed NSBM gives a good approximation to the conditional model. We use the NSBM to analyze a business school faculty hiring network in U.S. universities and obtain meaningful and interpretable results.

The rest of the chapter is organized as follows. In Section 4.2, we introduce the NSBM, as well as the community detection and model fitting algorithms. A conditional surveyed network model is also described and we discuss why the NSBM can be a potentially good approximation. In Section 4.3, the theoretical guarantee of consistency for community detection and parameter estimation under the NSBM are provided. In Section 4.4, we demonstrate how the model and model-fitting algorithms can be extended to weighted networks. The simulations studies under both NSBM and the conditional surveyed network model are included in Section 4.5 and Section 4.6 includes a detailed example of using the NSBM to analyze a faculty hiring network. The chapter is concluded in Section 4.7 with future work discussion.

4.2 The nomination stochastic block model

We start with a brief review of the directed extension of the standard SBM before presenting the new nomination model. While networks constructed by asking people about their friends are often treated as undirected by ignoring who nominated whom, they are directed by nature and in reality are often quite far from symmetric.

4.2.1 The directed stochastic block model

The stochastic block model (SBM) [Holland et al., 1983] is one of the most widely used and well-understood models for communities in a network. It has been shown to successfully discover meaningful communities in various problems, and can serve as a building block for more complicated models; see Abbe [2017] for a thorough recent review.

A network of n nodes can be represented by an $n \times n$ adjacency matrix A such that each entry $A_{ij} = \mathbf{1}(i \to j)$ is 1 if there is an edge from node i to node j and 0 otherwise. The standard SBM is defined for undirected networks, where $A_{ij} = A_{ji}$. For directed networks, a trivial extension of SBM can be defined as follows: given n nodes, a positive integer K and a $K \times K$ matrix of probabilities B, let $c_i \in \{1, \ldots, K\}$ be the community label of node i, and $\boldsymbol{c}$ be the vector of community labels. Here we treat $\boldsymbol{c}$ as fixed; a version with $\boldsymbol{c}$ sampled from a multinomial distribution is defined similarly. Let $G_k = \{i : c_i = k\}$ be the set of nodes in community k. The entries of the adjacency matrix A are then generated independently from the Bernoulli distribution with

$$P(A_{ij} = 1) = B_{c_i c_j} \tag{4.1}$$

Note that for the directed model we do not require B to be symmetric. This natural extension serves as a building block for the new model we propose.

4.2.2 The nomination stochastic block model

As discussed in the introduction, there are often missing edges in networks, and the pattern of missingness may be related to both community membership and individual node characteristics. Here we propose a new model that can reflect this. Let $\tilde{A}$ be the adjacency matrix we observe, where $\tilde{A}_{ij} = 1$ indicates node i reported that there is an edge from it to node j. In addition to previously defined $\boldsymbol{c}$ and B, we introduce two new node-specific parameters, given by n-dimensional vectors $\boldsymbol{\lambda} = (\lambda_i)$ and $\boldsymbol{\theta} = (\theta_i)$. The proposed nomination stochastic block model (NSBM) assumes the entries of $\tilde{A}$

are generated independently from the Bernoulli distribution with

$$P(\tilde{A}_{ij} = 1) = \tilde{P}_{ij} = \theta_i B_{c_i c_j}^{\lambda_i}, \tag{4.2}$$

which allows the probability of observing an edge to depend on both nodes' communities and the sender's individual characteristics.

If we enforce $\max_{1 \leq k,l \leq K} B_{kl} \leq 1$ and $\max_i \theta_i \leq 1$ (which we will show later can always be done with proper scaling), we can equivalently rewrite model (4.2) as a result of a generating process which takes an original network A generated from model (4.1) and applies a binary observation "mask" matrix R with Bernoulli entries generated independently with

$$P(R_{ij} = 1) = \theta_i B_{c_i c_j}^{\lambda_i - 1}.$$

The observed matrix is then given by

$$\tilde{A} = A \circ R,$$

where $\circ$ is the element-wise Hadamard matrix product. We can think of the parameter θ_i as measuring the overall propensity of node i to nominate friends, and of λ_i as a measure of their preference for nominating friends from their own or closely connected communities; both these factors may affect data collection in friendship surveys such as the AddHealth study [Harris, 2009].

As with any model involving products of multiple parameters, we need to ensure identifiability by determining conditions on B, $\boldsymbol{\lambda}$, $\boldsymbol{\theta}$, $\boldsymbol{c}$ that allow them to be uniquely identified from $\tilde{P}$, which controls the distribution of observed data $\tilde{A}$.

We need to ensure $\tilde{P}$ has no rows consisting entirely of zeros, and thus we require $\theta_i > 0$ for all i and that each row of B contains at least one positive entry. We also need scaling constraints on B and $\boldsymbol{\lambda}$ to avoid invariance to multiplicative constants.

In addition, if $B_{kl} = 0$ for all $l \neq k$, then community k will not send edges to other communities and it will be impossible to identify λ_i's for nodes in community k. On the other hand, if $B_{kl} = B_{kk}$ for all l, then community k is not identifiable. Putting all these together leads to the following identifiability conditions.

Proposition IV.1. *If the following conditions hold, then model* (4.2) *is identifiable.*

1. $B_{kk} = 1$ *for all* $k = 1, \ldots, K$.
2. *For each* k*, there exists at least one* $l \neq k$ *such that* $B_{kl} \in (0, 1) \cup (1, \infty)$.
3. $\theta_i > 0$ *for all* $i = 1, \ldots, n$.
4. $\frac{1}{n_k} \sum_{i \in G_k} \lambda_i = 1$ *for all* $k = 1, \ldots, K$*, where* $n_k = |G_k|$.
5. *If* $B_{kl} = 0$*, then* $\lambda_i > 0$ *for all* $i \in G_k$.

The proof can be found in Appendix C.1.

4.2.3 Community detection under the NSBM

The community label vector $\boldsymbol{c}$ is typically the main quantity of interest in community detection problems. For the standard SBM and its degree corrected version [Karrer and Newman, 2011], spectral clustering algorithms are among the most popular methods for estimating community labels [Rohe et al., 2011, Lei and Rinaldo, 2014, Jin, 2015]. Spectral methods have many advantage: easy implementation, computational efficiency and good theoretical properties. Generally speaking, spectral clustering relies on community information being represented in the eigenvectors of the population matrix $\mathbb{E}A$, and on good concentration of A around its expectation.

Under the NSBM, even though each row of the expectation of $\tilde{P}$ has community information confounded with individual preferences, the following result shows that community information can still be recovered from the columns of $\tilde{P}$.

Proposition IV.2. *Let $\tilde{P} = \tilde{U}\tilde{D}\tilde{V}^T$ be the SVD of $\tilde{P}$. Then there exists a matrix $X \in \mathbb{R}^{K \times K}$ such that*

$$\tilde{V} = ZX$$

where Z is the $n \times K$ community membership matrix, defined by $Z_{ik} = 1(c_i = k)$. In addition, $\|X_{k\cdot} - X_{l\cdot}\|_2 = \sqrt{n_k^{-1} + n_l^{-1}}$ for any $1 \le k, l \le K$.

The proof can be found in Appendix C.1. Proposition IV.2 suggests the right singular vectors of $\tilde{A}$ can be used to recover communities, as long as $\tilde{A}$ concentrates around $\tilde{P}$. This is formalized in the following algorithm, which we call Right singular vectors Spectral Clustering (Right SC).

Algorithm IV.3 (Right SC). Given an adjacency matrix $\tilde{A}$ of a directed network and the number of communities K:

1. Compute the singular value decomposition $\tilde{A} = \widehat{U}\widehat{D}\widehat{V}^T$.
2. Set $\widehat{X} = \widehat{V}_{\cdot,1:K}$ be the K leading right singular vectors, i.e., the first K columns of $\widehat{V}$.
3. Run the K-means clustering algorithm on $\widehat{X}$ to assign each node to a community.

While optimizing the K-means loss is NP-hard, there are many efficient algorithms that find approximate solutions. For theoretical developments, we will assume the K-means algorithm finds a value of the objective function that is at most $(1 + \epsilon)$ of the global minimum; this can be found efficiently for a small positive constant ϵ [Kumar et al., 2004].

4.2.4 Parameter estimation under the NSBM

Given community labels $\boldsymbol{c}$, it is relatively straightforward to estimate other parameters in the model (4.2), under identifiability constraints of Proposition IV.1. We use the method of moments to estimate the parameters. Specifically, if $B_{kl} > 0$, then for any arbitrary $i \in G_k$ and $j \in G_l$, we have

$$\log(\tilde{P}_{ij}) = \mu_{il} = \log(\theta_i) + \lambda_i \log(B_{kl}). \tag{4.3}$$

Combining the conditions in Proposition IV.1 and (4.3), we obtain the following identities:

$$\theta_i = \tilde{P}_{ij} \text{ for any } j \in G_k \ , \tag{4.4}$$

$$B_{kl} = \exp(-\frac{1}{n_k}\sum_{i\in G_k}(\mu_{ik} - \mu_{il})) \ , \tag{4.5}$$

$$\lambda_i = \frac{\mu_{ik} - \mu_{il}}{\sum_{j\in G_k}(\mu_{jk} - \mu_{jl})/n_k}, \text{ if } B_{kk} \neq B_{kl}. \tag{4.6}$$

Moreover, we also observe that

$$\exp(\mu_{il}) = \frac{1}{n_l}\mathbb{E}\sum_{j\in G_l}\tilde{A}_{ij} \tag{4.7}$$

Thus we can match the moment in (4.7) and estimate parameters using identities (4.4)-(4.6), with some modifications to handle boundary cases. For simplicity, we add the following assumption to those made in Proposition IV.1.

Assumption IV.4. *Assume $B_{kl} \neq B_{kk}$ for any $1 \leq k \neq l \leq K$.*

Under IV.4 and the conditions of Proposition IV.1, we propose the following algorithm to estimate the parameters in NSBM.

Algorithm IV.5 (Parameter estimation by the method of moments)*.* Given the network $\tilde{A}$ and community labels $\boldsymbol{c}$, for $k = 1, 2, \cdots, K$:

1. Set $T_{il} = \frac{\sum_{j\in G_l}\tilde{A}_{ij}}{n_l}$ for each $i \in G_k$ and $1 \leq l \leq K$.

2. Estimate θ_i by

$$\hat{\theta}_i = T_{ik} \ . \tag{4.8}$$

3. Find set $\Psi_k = \{l : 1 \leq l \leq K, T_{il} = 0 \ \forall i \in G_k\}$. Set $\hat{B}_{kl} = 0$ for each $l \in \Psi_k$.

4. (a) Define $Y_{il} = \log(T_{il} \vee \frac{1}{n_l})$, where the $\frac{1}{n_l}$ is used to avoid overflow for the pathological case of $T_{il} = 0$ for some $i \in G_k$.

   (b) For each $l \in \{1, 2, \cdots, K\}/(\Psi_k \cup \{k\})$

$$\hat{B}_{kl} = \exp(-\frac{1}{n_k} \sum_{i \in G_k} (Y_{ik} - Y_{il})). \tag{4.9}$$

   (c) Pick any $l \in \{1, 2, \cdots, K\}/(\Psi_k \cup \{k\})$, set

$$\hat{\lambda}_i = \frac{Y_{ik} - Y_{il}}{\sum_{j \in G_k} (Y_{jk} - Y_{jr})/n_k} \tag{4.10}$$

Remark IV.6. It is not difficult to remove Assumption IV.4 and estimate the parameters under the conditions of Proposition IV.1 alone. We only need to modify the last step (4.10) by summing up across l in both the numerator and the denominator. However, since we will need a stronger version of IV.4 for theoretical developments, we keep it here for the sake of conciseness.

Remark IV.7. In the current setting, the estimators are coincide with the MLE, as T_{il} is the MLE of $\exp(\mu_{il})$. However, in more general settings such as the ones introduced in Section 4.2.5 and Section 4.4, the MLE may be hard to obtain while the method of moments still remains a computationally feasible option as it only requires the conditions on first-order moments. Therefore, we introduce our estimators from the perspective of moment matching.

4.2.5 The conditional NSBM

One interpretation of the proposed NSBM is combining a directed SBM with an independent edge nomination procedure. The assumption of independence between the two can be restrictive in some applications. An alternative model would allow nominating edges conditioning on the presence of a true edge, that is, R would depend on A. Specifically, consider the following conditional NSBM (cNSBM): given the parameters $B, \boldsymbol{c}$, a nomination quota vector $\boldsymbol{d} = (d_i)$, and the nomination preference vector $\boldsymbol{\alpha} = (\alpha_i)$, we generate the observed network $\tilde{A}$ as follows:

1. Generate A from the directed SBM;
2. Given A, the i-th row of the observed network $\tilde{A}$, denoted by $\tilde{A}_{i\cdot}$, is generated by
    - If $\sum_j A_{ij} \leq d_i$, set $\tilde{A}_{i\cdot} = A_{i\cdot}$. That is, nominate all neighbors.
    - If $\sum_j A_{ij} > d_i$, subsample d_i of the existing neighbors in A sequentially with probability proportional to $B_{c_i c_j}^{\alpha_i}$ for neighbor node j. That is, the probability of choosing the next neighbor j is proportional to the weights $B_{c_i c_j}^{\alpha_i}$ amongst the remaining neighbors.

The cNSBM defined assumes the nominating procedure depends on the realized values of A, and that each node is limited in how many nominations they can make. For example, if $\alpha_i = 0$, node i uniformly chooses d_i of its neighbors to nominate, and if $\alpha_i = \infty$, node i nominates d_i neighbors with the highest connection probabilities (or if there are more than d_i such neighbors, then it randomly selects d_i of them). The cNSBM makes the entries of $\tilde{A}$ dependent in a complicated way, and even the marginal distribution of $\tilde{A}$ is no longer readily available. As a consequence, both fitting the model and investigating its properties becomes challenging. However, it appears that empirically the NSBM provides a good approximation to the expectation of $\tilde{A}$ under the cNSBM, even though it ignored dependence between entries.

Consider the following example of networks generated from the cNSBM. The underlying network A with $n = 300$ nodes is generated from the directed SBM with $K = 10$ communities of equal sizes. We focus on node 1 from community 1 without loss of generality. Let $\tilde{P}$ be the marginal distribution of $\tilde{A}$ and P be the distribution of A, we want to learn the relation between $\tilde{P}_{1\cdot}$ and $P_{1\cdot}$ where $\tilde{P}$ is calculated by Monte-Carlo average of 10,000 replications. We set the values in $B_{1\cdot}$ to range from 0.3 to 0.05 uniformly in log scale for $B_{1l}, 1 \leq l \leq K$.

Figure 4.1 shows the relationship between the values in $\log(P_{1\cdot})$ and $\log(P_{1\cdot})$ where the α is set to be $-1, -0.5, 1, 2, 5$, and ∞ and the nomination quota d_1 is 10 and 20 respectively. It can be seen that the log-log relationship is close to linear in all of the configurations of α and d. Such relationship indicates that each row of the marginal probability matrix $\tilde{P}$ may be approximated well by a power function of the same row in P. Note that the power function is exactly the assumption of NSBM since

$$\tilde{P}_{ij} = \theta_i B_{c_i c_j}^{\lambda_i}$$

is a power function of $B_{c_i c_j}$. Therefore, intuitively we expect the NSBM to be a good approximation for the conditional nomination procedure.

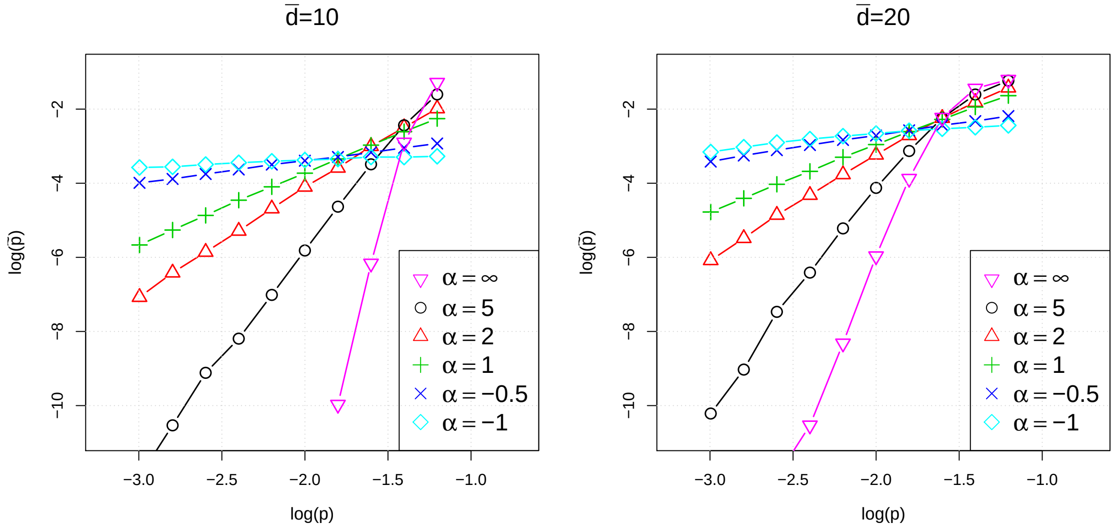


Figure 4.1: The log-log relationship between $\log(P_{1\cdot})$ and $\log(P_{1\cdot})$ under the conditional NSBM. The figures indicate an approximately linear relationship in the log scale.

4.3 Consistency under the NSBM

Here we investigate asymptotic properties of community detection and parameter estimation under the NSBM. In particular, in Section 4.3.1, we show that the Right SC algorithm will mis-cluster at most a vanishing proportion of nodes with high probability, as long as the network is not so sparse that it no longer concentrates. In Section 4.3.2, we show consistency for parameter estimators of B, $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}$.

4.3.1 Consistency of community detection

We first introduce an additional assumption we need for considering asymptotic behavior, which is that none of the communities vanish relative to the size of others when n grows.

Assumption IV.8. $n_{\min} := \min_k n_k \geq \kappa' n$ *for some constant* $\kappa' > 0$*. Also define* $n_{\max} = \max_k n_k$

Assumption IV.9. *There exists a constant* $\eta > 0$ *such that* $\max_i |\lambda_i| \leq \eta$ *and matrix* B *is a fixed matrix for all* n*.*

Theorem IV.10 (Consistency of community detection by the Right SC algorithm)**.** *Let* $\hat{\mathbf{c}}$ *be the output of the Right SC algorithm with* $(1+\epsilon)$ *optimal solution,* $T_n = \sigma_K(\tilde{P})$ *the* K*th largest singular value of* $\tilde{P}$ *and* $\theta_{\max} = \max_i \theta_i$*. Define sets* $S_k = G_k / \hat{G}_k, 1 \leq k \leq K$*. Assume identifiability assumptions of Proposition IV.1 and IV.8=IV.9 hold while* $\theta_{\max} \geq C_0 \frac{\log n}{n}$ *for some constant* C_0*. If there exists a constant* C_1 *depending on* C_0*,* ϵ *and* κ', η*, such that*

$$\frac{K n \theta_{\max}}{T_n^2} \leq \frac{1}{C_1},$$

then with probability at least $1 - n^{-1}$*, there exists a permutation of labels* $\hat{\mathbf{c}}$*, such that*

$$\sum_k \frac{|S_k|}{n_k} \leq C_1 \frac{K n \theta_{\max}}{T_n^2}.$$

Theorem IV.10 depends on $\sigma_K(\tilde{P})$, a quantity without an obvious interpretation. Under a particular parameterization, we can state a simpler form of this result.

Assumption IV.11 (Simplified Parameterization)**.** *Write $\theta_i = \rho_n \bar{\theta}_i$ where $\bar{\theta}_i$'s are independently sampled from a fixed multinoulli distribution on m_1 different positive values of which the maximum value is 1. Also, assume the values of λ_i's are obtained by i.i.d sample from a fixed multinoulli distribution with mean value 1 on m_2 values and then rescaled to ensure the identifiability constraint in Proposition IV.1.*

The above parameterization explicitly assume that essentially, ρ_n is the only quantity whose scale varies with n. The notation $\theta_i = \rho_n \bar{\theta}_i$ allows us to parameterize the sparsity of the network by a single parameter ρ_n. Specifically, under IV.8 and IV.11, we have

$$\kappa' n \rho_n \bar{\theta}_i \le n_{c_i} \rho_n \bar{\theta}_i \le \mathbb{E}(\sum_j \tilde{A}_{ij})$$

thus $\kappa' n \rho_n$ gives a lower bound on the minimum expected degree of the network as

$$\kappa' n \rho_n \le \min_i \frac{\mathbb{E}(\sum_j \tilde{A}_{ij})}{\bar{\theta}_i} \le \mathbb{E}(\sum_j \tilde{A}_{ij}).$$

We have the following corollary of Theorem IV.10:

Corollary IV.12. *Let $\hat{\mathbf{c}}$ be the clustering labels found by the Right SC algorithm with $(1+\epsilon)$ optimal solution and define sets $S_k = G_k / \hat{G}_k, 1 \le k \le K$. If assumptions of Proposition IV.1, IV.8,IV.11 hold and $n\rho_n \ge C_0 \log n$ for some constant C_0, then for sufficiently large n, with probability at least $1 - 2n^{-1}$, there exists a permutation of labels $\hat{\mathbf{c}}$, such that*

$$\sum_k \frac{|S_k|}{n_k} \le C' \frac{1}{n\rho_n} \tag{4.11}$$

for some constant C' depending on $C_0, \kappa', \epsilon, \eta, K$ and the multinoulli distributions for $\bar{\theta}_i$'s and λ_i's.

4.3.2 Parameter estimation consistency

Now we investigate theoretical properties of parameter estimation for $B, \boldsymbol{\lambda}$ and $\boldsymbol{\theta}$. For simplicity, we assume the true community labels $\boldsymbol{c}$ are known, and we make one additional regularity assumption.

Assumption IV.13. *There exists a constant $\kappa > 0$, such that for any $k \neq l$, either $B_{kl} \leq \exp(-\kappa)$ or $B_{kl} \geq \exp(\kappa)$ is true.*

IV.13 essentially requires the connection strength from every community to a different community to be distinct from the connection strength within the community itself (recall that for identifiability we assume $B_{kk} = 1$). Then we have the following result.

Theorem IV.14. *Let $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\lambda}}$ and $\hat{B}$ be the estimators for $\boldsymbol{\theta}, \boldsymbol{\lambda}$ and B respectively, obtained by Algorithm IV.5. Assume conditions of Proposition IV.1, IV.8 and IV.11 hold and $\min_i \theta_i \geq c_0 n^{-1/4}$. Then there exists a constant c, such that for any $\rho \in (0,1)$ and sufficiently large n we have:*

$$\max_i \frac{|\hat{\theta}_i - \theta_i|}{\theta_i} \leq \frac{1}{c_0} n^{-\frac{1}{12}}, \tag{4.12}$$

with probability at least $1 - 2\exp(-\frac{c}{2}n^{1/3})$;

$$\max_{k,r} |\log(\hat{B}_{kl}) - \log(B_{kl})| \leq 2n^{-\frac{1-\rho}{4}}, \tag{4.13}$$

with probability at least $1 - 8\exp(-\frac{c\kappa' c_0^2}{10}n^{\rho/2})$;

$$\max_{i\in[n]} |\hat{\lambda}_i - \lambda_i| \leq \frac{2\sqrt{2}}{\kappa}\sqrt{4 + 16(\eta + 1/2)^2} n^{-\frac{1-\rho}{4}}, \tag{4.14}$$

with probability at least $1 - 16\exp(-\frac{c\kappa' c_0^2}{10}n^{\rho/2})$.

4.4 Extension to weighted networks

Networks with edge weights are frequently encountered in practice, and even though methods for binary networks can be applied to weighted networked after thresholding edge weights, this results in substantial loss of information. For NSBM, it turns out to be straightforward to model the weighted network directly.

Given community labels $\boldsymbol{c}$, assume each edge weight $\tilde{A}_{ij}$ is independently generated as

$$\tilde{A}_{ij} \sim \pi(\theta_i B_{c_i c_j}^{\lambda_i})$$

where π is a probability distribution satisfying

$$\mathbb{E}_\pi \tilde{A}_{ij} = \theta_i B_{c_i c_j}^{\lambda_i}. \tag{4.15}$$

The specific choice of π will depend on the problem at hand. For instance, the Poisson distribution has often been used to model network edge weights, and is a good choice for non-negative integer weights without a heavy tail Karrer and Newman [2011]. The distribution π can depend on multiple parameters, but we only require one constraint directly on its expectation. Similarly to the binary edge setting, we can interpret (4.15) as a combination of generating and nominating procedures. Since the model is specified through the expectation, we can still apply the right spectral clustering and method of moments algorithms, and similar theoretical guarantees can be obtained as long as the generating distribution π is sub-Gaussian.

4.5 Simulation studies

In this section, we demonstrate the effectiveness of the NSBM for community detection and network modeling using simulation studies. We first show that under the NSBM, the Right SC performs best on detecting communities, outperforming several other natural choices of spectral clustering algorithms. Next, we generate networks from the conditional model introduced in Section 4.2.5 and fit several computationally feasible network models with communities, among which NSBM turns out to provide the best approximation.

4.5.1 Community detection under NSBM

In this section, we evaluate several commonly used spectral clustering algorithms for community detection under the NSBM. Treating the network as a directed network, one alternative to the Right SC algorithm we proposed is to use the left singular vectors for clustering, which we call “Left SC”. However, it is clear that under NSBM Left SC will fail given it does not take node heterogeneity between nodes within the same community into account. Therefore we instead consider left spherical SC (Left SSC),which first normalizes each row of the matrix of left singular vectors before applying the K-means algorithm. This is a standard way to deal with row heterogeneity [Lei and Rinaldo, 2014]. We do not consider the spherical version of Right SC, since under our model they are very similar.

Another common approach is to treat the directed network as undirected and transform $\tilde{A}$ to a symmetric matrix. A commonly used transformation is the “OR” operation, i.e., connecting two nodes in the undirected network if there is an edge in either direction in the directed network. Applying SC and SSC to the symmetrized network gives two more options, “Symmetric SC” and “Symmetric SSC”. This is essentially equivalent to treating the network as generated from the SBM or the degree-corrected SBM, respectively.

Networks are generated as follows: $n = 300$ nodes are randomly assigned to $K = 3$ communities with equal probability. The matrix B has all diagonal elements equal to 1 and all off-diagonal elements equal to β. The parameters λ_i's are generated independently with $\log(\lambda)$ sampled uniformly from the interval $(-t, t)$, and then rescaled to satisfy the constraint $\sum_{\boldsymbol{c}_i=k} \lambda_i = n_k$ for each k. Each θ_i is independently set either to c or $0.05c$, with probability 0.5 each, with the value c chosen so that the resulting average degree of the network is 40.

First, we vary t from 0.2 to 2 and fix $\beta = 0.2$. Clustering accuracy measure d by $\text{Ham}(\hat{\boldsymbol{c}}, \boldsymbol{c})/n$ and averaged over 100 replications is shown in Figure 4.2. For small t, the nomination step does not change the network much, and thus spectral clustering based on the standard SBM (or DCSBM) still works. As t increases and the nomination preferences become more heterogeneous across nodes, ignoring the edge direction dilutes the block structure and symmetric clustering methods fail. The Left SSC is even worse, since it relies entirely on the senders information, where the community structure is masked by heterogeneity in the nomination preferences.

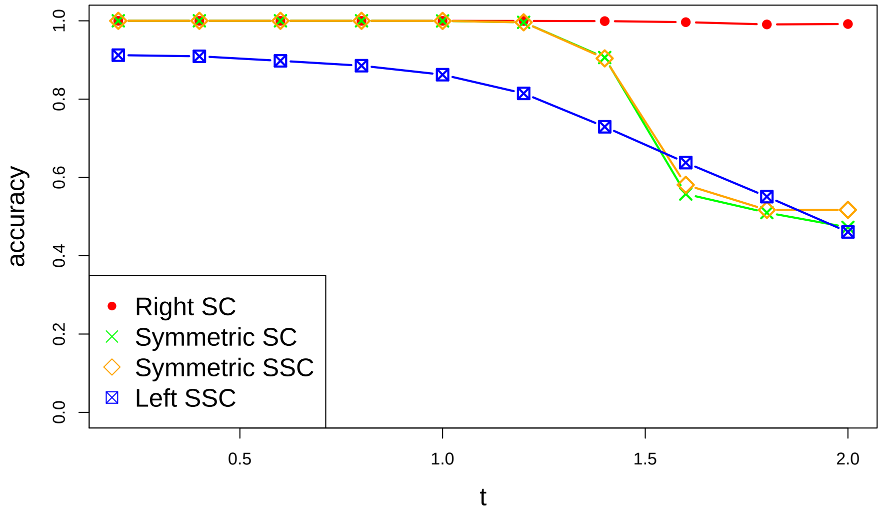


Figure 4.2: Community detection accuracy of spectral methods under NSBM as a function of t, with $\beta = 0.2$.

Next, we compare different methods while varying the signal-to-noise ratio. Specifically, we vary β from 0.1 to 0.5 and fix $t = 1.5$. The average clustering accuracy is shown in Figure 4.3. In this scenario, alternative clustering algorithms fail even for small β. The Right SC is the only algorithm effective for a wide range of β values.

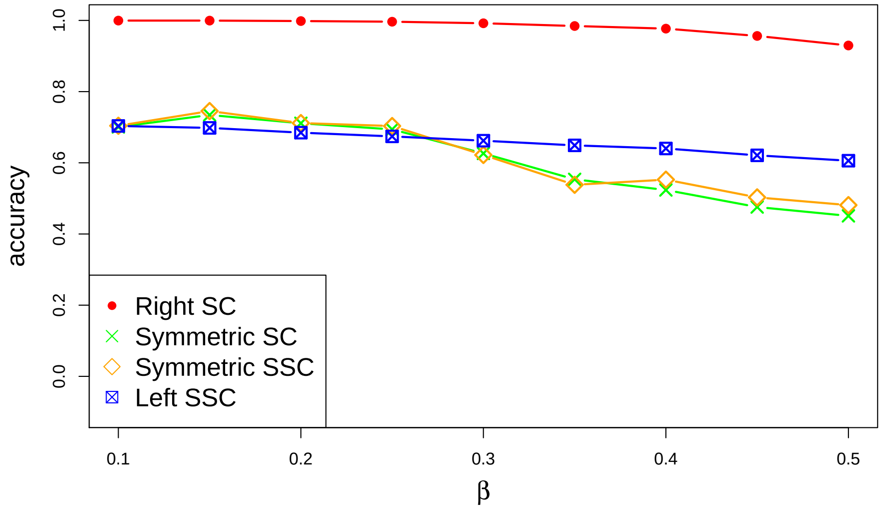


Figure 4.3: Community detection accuracy of spectral methods under NSBM as a function of β, with $t = 1.5$.

4.5.2 NSBM as an approximation to the conditional model

Instead of generating networks from the NSBM, here we generate networks from the conditional NSBM introduced in Section 4.2.5. We first generate A from the directed SBM with $n = 300$ nodes and $K = 3$ communities, where the community labels are uniformly assigned to the nodes. The off-diagonal values of the block connection matrix are sampled uniformly from $U(0.6\beta, 1.4\beta)$ for some β to be specified. The average degree of A is 90, which is quite dense. However, we do not observe A, and in generating $\tilde{A}$, we randomly choose 40% of the nodes to have $\bar{d}_i = 5$ (so these nodes are allowed to nominate up to five connections), 40% of the nodes to have $\bar{d}_i = 10$, and the rest to $\bar{d}_i = 120$, so the resulting network has an average degree around 24. Given a value α_0, we randomly choose 50% of the nodes to have

$\alpha_i = \alpha_0$, 10% of the nodes to have $\alpha_i = 0.1\alpha_0$, 10% of the nodes to have $\alpha_i = 0.5\alpha_0$ and 30% of the nodes to have $\alpha_i = -0.8\alpha_0$.This means some of the nodes prefer to report connections from communities close to themselves, while other nodes have a preference for reporting connections to communities they are less connected to, on average.

Since we do not have a direct method to fit the conditional model, we set out to see how well the NSBM approximates it. We compare its performance to three other computationally feasible network models with communities: the directed SBM and its degree-corrected version directed DCSBM Karrer and Newman [2011], and the stochastic co-clustering block model (SCBM) of Rohe et al. [2016]. The SCBM is also based on the idea of directed DCSBM but assumes different community memberships for senders and receivers, therefore it is not directly comparable in the sense of community detection. However, it can also provide an approximation to our conditional nomination model.

First, we compare community detection performance of the four models. The community detection step for the four models is carried out by Right SC, symmetric SC, symmetric SSC and the co-clustering (Right+Left) SSC in Rohe et al. [2016] respectively. Parameter estimation for the three competitor models is done by maximum likelihood after community detection, as in Karrer and Newman [2011]. In addition to community detection accuracy, we also report the relative error of estimating the underlying probability matrix,

$$\frac{\|\hat{P} - \tilde{P}\|_F^2}{\|\tilde{P}\|_F^2}.$$

Figure 4.4 shows the community detection accuracy and log relative estimation error for α_0 ranging from 0 to 5 and fixed $\beta = 0.35$. The Right SC is still effective under the conditional model, while other clustering methods fail to detect communities. When

α_0 becomes large, the nodes tend to only nominate from their own communities, thus the community structure becomes stronger and the two symmetric algorithms improve slightly. The Left SC again does not work well since it does not access the relevant information. Neither the SBM nor the DCSBM estimate the true probability matrix well, and the SCBM works somewhat better, due to its higher flexibility. The NSBM approximates the probability matrix well and obtains the lowest error among the methods compared. It also shows very little sensitivity to the value of α_0.

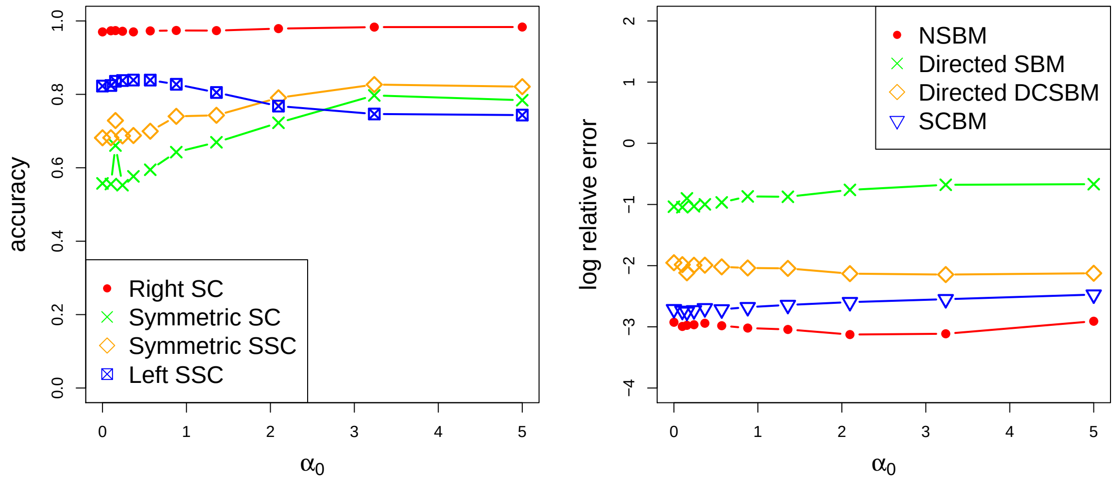


Figure 4.4: Community detection accuracy and log relative errors of estimating the probability matrix under conditional model, for NSBM, directed SBM, directed DCSBM, and SCBM, as a function of α_0 with fixed $\beta = 0.35$.

Next, we fix $\alpha_0 = 1$ and vary the value of β to investigate the impact of signal-to-noise ratio. The results are shown in Figures 4.5. In this setting, the Left SSC is again not competitive, even for small β. When $\beta < 0.3$, the two symmetric methods again work similarly to Right SC, but their accuracy drops quickly once $\beta \geq 0.3$. For probability matrix estimation, the NSBM is always better than the other three for $\beta \leq 0.4$, providing a good approximation to the general model. As β increases, the signal-to-noise ratio becomes lower, and all methods become similar.

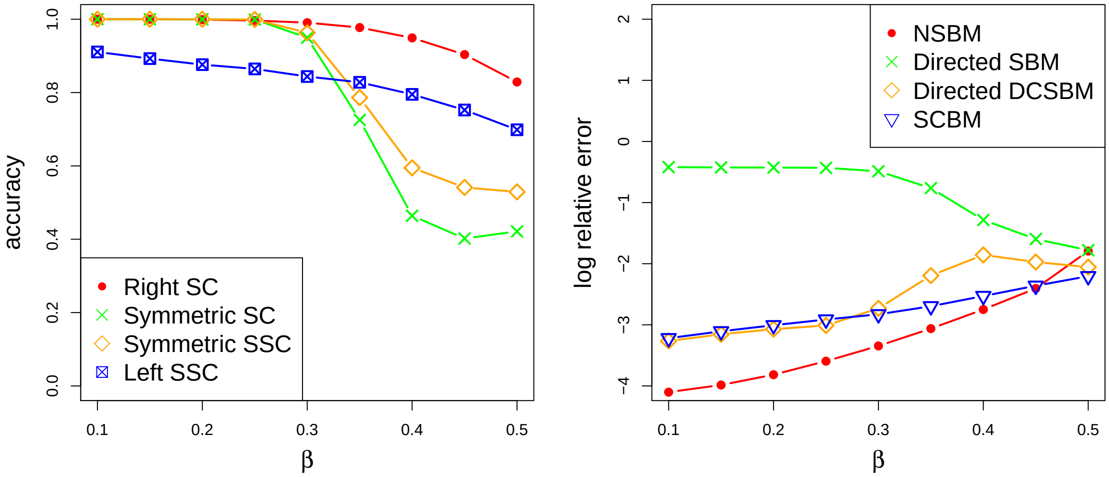


Figure 4.5: Community detection accuracy and log relative errors of estimating the probability matrix under the conditional model, for NSBM, directed SBM, directed DCSBM, and SCBM, as a function of β with fixed $\alpha_0 = 1$.

4.6 Business faculty hiring network analysis

Here we apply the proposed NSBM model to analyze a faculty hiring network between US Business schools. The data were collected by Clauset et al. [2015] via web crawling, and records information on 18,924 tenure or tenure-track faculty members, recording the institution from which they obtained their PhD and the institution by which they were hired. The original dataset covers faculty in the fields of business, computer science and history; here we focus on the business hiring network.

The data from business schools covers 7856 faculty members from 112 institutions. To reduce noise, we removed institutions with either receiver or sender degree of 3 or less, resulting in 87 institutions remaining. We construct a network by creating an edge from i to j with weight 1 if institution i has hired one faculty with a Ph.D. from institution j. If institution i has hired more than one graduate of institution j, we set the edge weight to 2. We found empirically that truncating the weights improves

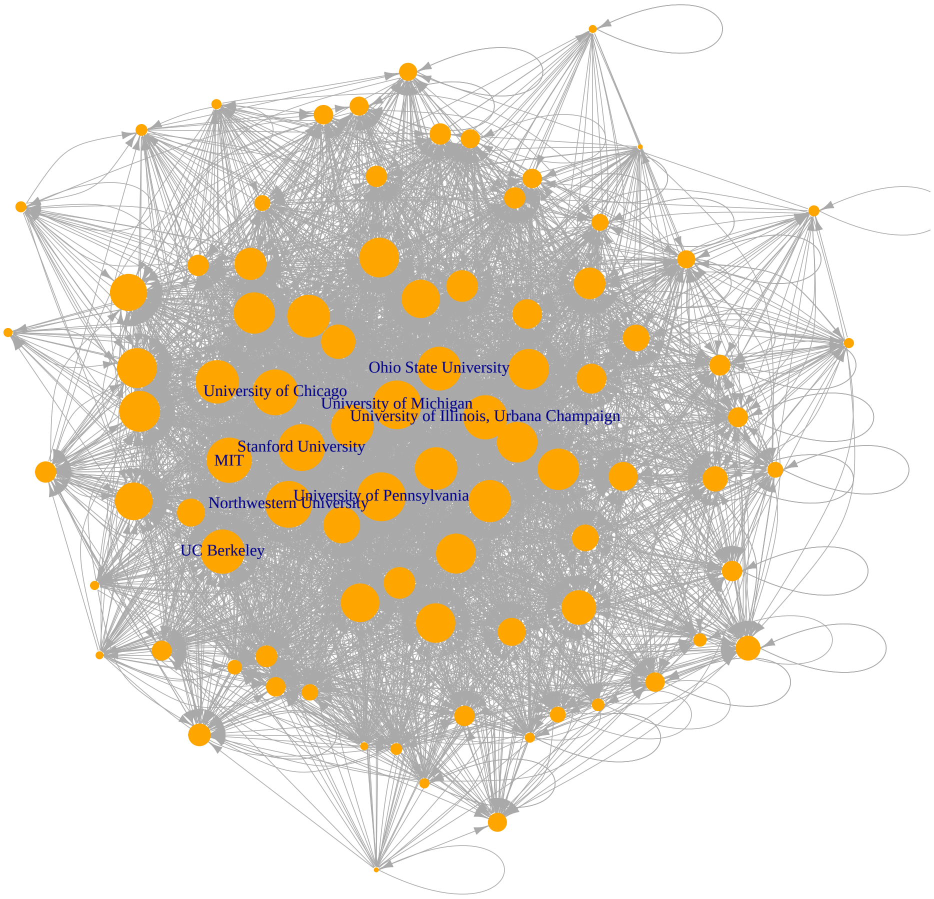


Figure 4.6: The hiring network between 87 U.S. business schools. An edge from i to j indicates that institution i has hired Ph.D. graduates from institution j. The node size is proportional to the receiver degree.

stability, whereas setting all edge weights to 0 or 1 loses too much information. The resulting directed network with 87 nodes is shown in Figure 4.6, where the node size is proportional to the receiver degree, i.e., the number of institutions that institution i has sent its graduates to. We are interested in finding communities of institutions as well as investigating whether there are hiring "inequalities" between these communities. The NSBM suits the hiring network well, because we do not observe job offers that did not result in a hire, and we can assume that most institutions made some offers that were declined.

To determine the number of communities, we apply the edge cross-validation method with average stability selection proposed by Li et al. [2016b], which has been shown to be very effective for network model selection. The procedure suggests $K = 4$ communities for this dataset. We then fit the NSBM to the network with $K = 4$. Table 4.1 shows the communities as well as their average rankings from two sources, the US NEWS graduate school rankings from 2012 (from the data set) and the π-ranking proposed by Clauset et al. [2015]. The π-ranking is designed to measure hiring advantage, where a higher ranked institution tends to be more successful in hiring competitive candidates. We list up to 15 institutions with the highest π-ranking within each community. Overall, the communities match both ranking systems very well, showign a clear ordering, with the first community mostly consisting of top business schools, the second one with good but slightly lower ranked schools, and so on.

	size	USNews (avg./med.)	π-ranking (avg./med.)	Institutions
1	12	7.7/8	8.3/8	Stanford, MIT, Harvard, UC Berkeley, U Chicago, Cornell, U Michigan, Columbia, Yale, U Penn., NYU, Duke
2	12	29.8/32.5	17.7/17.5	U Rochester, Northwestern, Carnegie Mellon, U Wisconsin Madison, UCLA, U Minnesota-Twin Cities, UIUC, Purdue, U Florida, UT Austin, U Washington
3	19	53.1/54	45/45	Ohio State, UNC Chapel Hill, U Pittsburgh, Penn. State, Indiana U., Michigan State, Georgia Tech, U Arizona, SUNY Buffalo, Texas A&M, U Georgia, Arizona State, U South Carolina, Virginia Tech, Florida State
4	44	63.7/63	61.4/61.5	Washington U St. Louis, U Maryland College Park, U Colorado Boulder, UC Irvine, U Utah, U Oregon, U Southern California, UT Dallas, U Virginia, Boston U., UMass Amherst, Emory, Case Western, UC Davis, Vanderbilt

Table 4.1: Communities of business schools found by NSBM and their average and median rankings from US News 2012 and Clauset et al. [2015]. Up to 15 institutions with the highest π-ranking are shown for each community.

The parameters of NSBM can be directly interpreted to see if we observe a hierarchy in hiring, which was reported by Clauset et al. [2015]. Based on the weighted

NSBM in Section 4.4, we define connection strength from community k to community l as the expectation of average connection weights from nodes $i \in G_k$ to nodes $j \in G_l$,

$$M_{kl} = \frac{1}{n_k n_l} \sum_{i \in G_k, j \in G_l} \theta_i B_{ij}^{\lambda_i}.$$

	Group 1	Group 2	Group 3	Group 4
Group 1	1.86	0.93	0.15	0.06
Group 2	1.56	1.31	0.42	0.13
Group 3	0.86	1.29	0.98	0.32
Group 4	0.76	0.86	0.51	0.22

Table 4.2: Estimated strengths of connections between communities.

Table 4.2 shows the estimated connection strengths for the business hiring network. It shows that Group 1 institutions tend to hire the most from their own group, and about half as many from Group 2. They are not very likely to hire from Groups 3 and 4. Group 2 institions hire roughtly equally from Groups 1 and 2, and a fraction from Group 3, but very few from 4. Interestingly, Group 3 institutions hire the most from Group 2, not Group 1. Community 4 follows a similar pattern, hiring more from groups closer to itself. The model parameters thus indicate a strong hierarchy in hiring relationships between the groups, which is validated by the rankings in Table 4.1,

In addition to community parameters, NSBM allows us to estimate hiring preferences of individual institutions as represented by parameters λ. Table 4.3 shows the estimated λ_i's for Group 1, indicating how strongly each institution follows the community level preferences. For instance, we see that Yale and Cornell show a stronger preference for hiring graduates from their own group, while U Michigan and U Penn are relatively less stringent. However, all the institutions have $\hat{\lambda}_i > 0.5$, indicating that they all follow the community-level preference order in Table 4.2 reasonably well.

Institution	$\hat{\lambda}_i$	USN ranking	π-ranking
Yale	2.17	10	11
Cornell	1.88	16	7
Columbia	1.10	9	10
Harvard	0.97	2	3
MIT	0.96	3	2
UC Berkeley	0.85	7	4
U of Chicago	0.75	5	6
Stanford	0.74	1	1
New York U	0.70	10	16
Duke	0.69	12	19
U Michigan	0.61	14	9
U Pennsylvania	0.57	3	12

Table 4.3: Estimated λ_i's for Group 1 institutions.

Overall, by fitting the proposed NSBM model to the faculty hiring network, we discover clear hierarchical structures reflecting profound social inequalities in the hiring relationship between institutions, which matches the observation of Clauset et al. [2015].

We briefly report community detection results on this network obtained by spectral clustering on the undirected version of the hiring network. The four communities are shown in Table C.1 with average and median ranking by US News and π-ranking, and up to 20 institutions with the highest π-ranking in each community. The first group is still higher ranked even though it no longer includes universities like Yale, Cornell, and Columbia, but the following three groups all show similar average rankings, suggesting these communities are not especially interpretable and likely do not correspond to real hiring patterns. This confirms the importance of using the correct spectral information for obtaining meaningful results.

4.7 Summary and future work

We have proposed a directed network model that admits community structures as well as the potential survey procedures in data-collecting stage. Computationally efficiently algorithms are available for model fitting and the theoretical guarantees are

	size	USN (avg./med.)	π-ranking (avg./med.)	Institutions
1	19	19.2/14	17.8/13	Stanford, MIT, Harvard, UC Berkeley, U Rochester, U Chicago, Northwestern, U Michigan, U Penn., Carnegie Mellon, NYU, U Minnesota Twin Cities, Duke, UNC Chapel Hill, U Washington St. Louis, U Maryland, College Park, U Southern California, Case Western Reserve U, Boston College
2	20	55.1/56.5	44.6/42	Cornell, Columbia, U Wisconsin-Madison, UIUC, Ohio State, U Florida, U Pittsburgh, Penn State, Michigan State, SUNY Buffalo, U Mass Amherst, Syracuse, Tulane, U Connecticut, U Cincinnati, Rutgers U, Temple U, SUNY Binghamton, St. Louis U, Northeastern U
3	24	52.7/40	54/49	Yale, UCLA, U Washington, U Colorado Boulder, UC Irvine, U Utah, U Oregon, UT Dallas, U Virginia, Boston U, UC Davis, Vanderbilt, Claremont Graduate U, U Houston, Rice U, Southern Methodist U, George Washington U, CUNY Baruch College, U Hawaii
4	24	63.8/63	56/56.5	Purdue, U Iowa, UT Austin, Indiana U, Georgia Tech, U Arizona, Texas A&M, U Georgia, Arizona State, U South Carolina, Virginia Tech, Florida State, U Oklahoma, U Kansas, Louisiana State, U Arkansas, U Tennesse, U Kentucky, U Alabama, Oklahoma State

Table 4.4: Communities of business school institutions detected by symmetric spectral clustering.

provided for these algorithms. By both simulation examples and a real world application, we have demonstrated the effectiveness of the proposed model in discovering communities and the underlying data generating mechanism.

Though we can use the NSBM to approximate the conditional nomination model, it would be interesting in the future to investigate how we can directly make inference under the model. This may involve Bayesian inference framework and rely on MCMC methods, for which the computation efficiency for large networks might be a concern. Another direction is to consider other structural assumptions of the network other than communities and investigate how the potential survey procedure interacts with the specific network structure.

APPENDICES

APPENDIX A

Appendix for Chapter II

A.1 Proofs

Proof of Proposition II.5. The first claim follows directly from the fact that $\mathbf{1}$ is an eigenvector of $P_{X^\perp} + \lambda L$ with eigenvalue 1, since $\mathbf{1} \in \mathrm{col}(X)^\perp$ and $L\mathbf{1} = 0$. To show the second claim, note that the minimum eigenvalue of $P_{X^\perp} + \lambda L$ is the solution of the optimization problem

$$\min_{\|\boldsymbol{u}\|=1} \boldsymbol{u}^T (P_{X^\perp} + \lambda L)\boldsymbol{u}.$$

Assume $\boldsymbol{u} = \boldsymbol{u}_1 + \boldsymbol{u}_2$, where $\boldsymbol{u}_1 \in \mathrm{col}(X)^\perp$, $\boldsymbol{u}_2 \in \mathrm{col}(X)$ and $\|\boldsymbol{u}_1\|^2 + \|\boldsymbol{u}_2\|^2 = 1$. Then the objective function can be rewritten as

$$\lambda \boldsymbol{u}^T L \boldsymbol{u} + \|\boldsymbol{u}_1\|^2.$$

This is zero if and only if $\|\boldsymbol{u}_1\| = 0$ and $\boldsymbol{u}^T L \boldsymbol{u} = 0$, but these two contradict Assumption II.4. As discussed in Section 2.2.2, the RNC estimator exists whenever $P_{X^\perp} + \lambda L$ is invertible, which shows that the RNC estimate exists.

□

One formula that will be used frequently later is the decomposition of MSE for a vector estimation:

$$\mathbb{E}\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|^2 = \|\mathbb{E}\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|^2 + \mathrm{tr}(\mathrm{Var}(\hat{\boldsymbol{\theta}})),$$

in which we call the second term total variance of $\hat{\boldsymbol{\theta}}$.

We first derive the bias and variance of both the OLS and the RNC estimators. We use $b(\cdot)$ to denote the bias of an estimator. The bias, variance and MSE of the OLS estimator are standard. We state the MSE here for completeness without proof.

Lemma A.1. *For the OLS estimator given by*

$$\hat{\boldsymbol{\beta}}_{OLS} = (X^T X)^{-1} X^T Y,\ \hat{\boldsymbol{\alpha}}_{OLS} = \bar{y}\mathbf{1},$$

we have

$$\begin{aligned}
\mathrm{MSE}(\hat{\boldsymbol{\alpha}}_{OLS}) &= \|\bar{\boldsymbol{\alpha}}\mathbf{1} - \boldsymbol{\alpha}\|^2 + \frac{\sigma^2}{n},\\
\mathrm{MSE}(\hat{\boldsymbol{\beta}}_{OLS}) &= \|(X^T X)^{-1} X^T \boldsymbol{\alpha}\|^2 + \sigma^2 \mathrm{tr}((X^T X)^{-1}),
\end{aligned}$$

$$\mathbb{E}\|\hat{Y}_{OLS} - \mathbb{E}Y\|^2 = \|(\frac{1}{n}\mathbf{1}\mathbf{1}^T + X(X^T X)^{-1} X^T)\boldsymbol{\alpha} - \boldsymbol{\alpha}\|^2 + \sigma^2 \|\frac{1}{n}\mathbf{1}\mathbf{1}^T + X(X^T X)^{-1} X^T\|_F^2.$$

Lemma A.2. *The bias of the RNC estimator is given by*

$$b(\hat{\boldsymbol{\theta}}) = -\lambda(\tilde{X}^T \tilde{X} + \lambda M)^{-1} M\boldsymbol{\theta}. \tag{A.1}$$

Equivalently, one can write it in the following decomposed form:

$$b(\hat{\boldsymbol{\theta}}) = (b(\hat{\boldsymbol{\alpha}})^T, ((X^T X)^{-1} X^T b(\hat{\boldsymbol{\alpha}}))^T)^T, \tag{A.2}$$

where $b(\hat{\boldsymbol{\alpha}}) = -(\frac{1}{\lambda} P_{X^\perp} + L)^{-1} L\boldsymbol{\alpha}$, *and* $P_X = X(X^T X)^{-1} X^T$ *is the projection matrix onto* $col(X)$.

The variance of the RNC estimator is given by

$$\mathrm{Var}(\hat{\boldsymbol{\theta}}) = \sigma^2 (\tilde{X}^T \tilde{X} + \lambda M)^{-1} \tilde{X}^T \tilde{X} (\tilde{X}^T \tilde{X} + \lambda M)^{-1} \preceq \sigma^2 (\tilde{X}^T \tilde{X} + \lambda M)^{-1}.$$

Proof. For the bias term,

$$\begin{aligned}
b(\hat{\boldsymbol{\theta}}) &= \mathbb{E}(\tilde{X}^T \tilde{X} + \lambda M)^{-1} \tilde{X}^T Y - \boldsymbol{\theta}\\
&= (\tilde{X}^T \tilde{X} + \lambda M)^{-1} \tilde{X}^T \tilde{X}\boldsymbol{\theta} - \boldsymbol{\theta}\\
&= -\lambda(\tilde{X}^T \tilde{X} + \lambda M)^{-1} M\boldsymbol{\theta}.
\end{aligned}$$

Note that we have $M\boldsymbol{\theta} = \begin{bmatrix} L\boldsymbol{\alpha} \\ 0 \end{bmatrix}$. By the block matrix inverse formula, we have

$(\tilde{X}^T\tilde{X} + \lambda M)^{-1} =$

$$\begin{bmatrix} (P_{X^\perp} + \lambda L)^{-1} & (P_{X^\perp} + \lambda L)^{-1}X(X^TX)^{-1} \\ (X^TX)^{-1}X^T(P_{X^\perp} + \lambda L)^{-1} & (X^TX)^{-1} + (X^TX)^{-1}X^T(P_{X^\perp} + \lambda L)^{-1}X(X^TX)^{-1} \end{bmatrix}.$$

Then (A.2) follows directly from decomposing the bias vector into the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ parts.

The variance can be calculated by the standard OLS formula taking $\tilde{X}$ as the design matrix. The positive semi-definiteness follows from the fact that

$$X^TX \preceq X^TX + \lambda M$$

whenever M is positive semi-definite. □

From Lemma A.2 and the bias-variance decomposition, we can directly get the closed form expressions for the MSE of RNC estimation. In particular,

$$\begin{aligned} \text{MSE}(\boldsymbol{\theta}) =& \|\lambda(P_{X^\perp} + \lambda L)^{-1}L\boldsymbol{\alpha}\|^2 + \|\lambda(X^TX)^{-1}X^T(P_{X^\perp} + \lambda L)^{-1}L\boldsymbol{\alpha}\|^2 \\ &+ \sigma^2\text{tr}((\tilde{X}^T\tilde{X} + \lambda M)^{-1}\tilde{X}^T\tilde{X}(\tilde{X}^T\tilde{X} + \lambda M)^{-1}). \end{aligned} \tag{A.3}$$

Proof of Theorem II.6. Note that $P_{X^\perp} + \lambda L \succeq \nu I$. Thus the squared bias term for $\boldsymbol{\alpha}$ is

$$\|\lambda(P_{X^\perp} + \lambda L)^{-1}L\boldsymbol{\alpha}\|^2 \le \frac{\lambda^2}{\nu^2}\|L\boldsymbol{\alpha}\|^2.$$

The total variance of $\hat{\boldsymbol{\alpha}}$ can be upper bounded by

$$\text{tr}(\sigma^2(P_{X^\perp} + \lambda L)^{-1}) \le \frac{\sigma^2}{\nu}\text{tr}(I) = \frac{n\sigma^2}{\nu}.$$

Thus the bound (2.12) on MSE$(\hat{\boldsymbol{\alpha}})$ follows.

From Lemma A.2, we have

$$\begin{aligned}\|b(\hat{\boldsymbol{\beta}})\|^2 =& b(\hat{\boldsymbol{\alpha}})^T X(X^TX)^{-1}(X^TX)^{-1}X^T b(\hat{\boldsymbol{\alpha}})\\ \leq& \frac{1}{\mu} b(\hat{\boldsymbol{\alpha}})^T X(X^TX)^{-1}(X^TX)(X^TX)^{-1}X^T b(\hat{\boldsymbol{\alpha}})\\ =& \frac{1}{\mu} b(\hat{\boldsymbol{\alpha}})^T X(X^TX)^{-1}X^T b(\hat{\boldsymbol{\alpha}}) = \frac{1}{\mu} b(\hat{\boldsymbol{\alpha}})^T (P_X b(\hat{\boldsymbol{\alpha}}))\\ =& \frac{1}{\mu}\|P_X b(\hat{\boldsymbol{\alpha}})\|^2 \leq \frac{1}{\mu}\|b(\hat{\boldsymbol{\alpha}})\|^2 \leq \frac{\lambda^2}{\nu^2\mu}\|L\boldsymbol{\alpha}\|^2. \qquad \text{(A.4)}\end{aligned}$$

By Lemma A.2 and Schur complement, the covariance matrix of $\hat{\boldsymbol{\beta}}$ is

$$\begin{aligned}\mathrm{Var}(\hat{\boldsymbol{\beta}}) &\preceq \sigma^2(X^TX)^{-1} + \sigma^2(X^TX)^{-1}X^T(P_{X^\perp}+\lambda L)^{-1}X(X^TX)^{-1}\\ &\preceq \sigma^2(X^TX)^{-1} + \frac{\sigma^2}{\nu}(X^TX)^{-1}X^TX(X^TX)^{-1} = \sigma^2(\frac{1}{\nu}+1)(X^TX)^{-1}. \qquad \text{(A.5)}\end{aligned}$$

Combining the squared bias (A.4) and variance (A.5) gives the bound (2.13) on $\mathrm{MSE}(\hat{\boldsymbol{\beta}})$. The mean squared prediction error can be similarly derived. With $\hat{\boldsymbol{V}} = \tilde{X}\hat{\boldsymbol{\theta}}$, we have

$$b(\hat{\boldsymbol{V}}) = \tilde{X}b(\hat{\boldsymbol{\theta}}) = -\lambda\tilde{X}(\tilde{X}^T\tilde{X}+\lambda M)^{-1}M\boldsymbol{\theta},$$

and

$$\mathrm{Var}(\hat{\boldsymbol{V}}) = \sigma^2\tilde{X}(\tilde{X}^T\tilde{X}+\lambda M)^{-1}\tilde{X}^T\tilde{X}(\tilde{X}^T\tilde{X}+\lambda M)^{-1}\tilde{X}^T.$$

Thus

$$\begin{aligned}\mathbb{E}\|\hat{\boldsymbol{V}} - \mathbb{E}Y\|^2 &= \|b(\hat{\boldsymbol{V}})\|^2 + \mathrm{tr}(\mathrm{Var}(\hat{\boldsymbol{V}}))\\ &\leq \lambda^2(L\boldsymbol{\alpha})^T(P_{X^\perp}+\lambda L)^{-1}(L\boldsymbol{\alpha}) + \sigma^2\mathrm{tr}(S_\lambda^TS_\lambda)\\ &\leq \frac{\lambda^2}{\nu}\|L\boldsymbol{\alpha}\|^2 + \sigma^2\|S_\lambda\|_F^2.\end{aligned}$$

This completes the proof of Theorem II.6. □

Proof of Proposition II.9. Let τ_i be the ith largest eigenvalue of L with the associated eigenvector $\boldsymbol{u}_k$. Assume $\boldsymbol{\alpha} = \sum_{i=n-k+1}^n \rho_i\boldsymbol{u}_i$. Without loss of generality, assume

$\|\boldsymbol{\alpha}\|^2 = n$ thus $\sum_{i=n-k+1}^{n} \rho_i^2 = n$. In this situation, we need

$$\|L\boldsymbol{\alpha}\|^2 = \sum_{i=n-k+1}^{n} \rho_i^2 \tau_i^2 \leq n^c.$$

Since $\sum_{i=n-k+1}^{n} \rho_i^2 \tau_i^2 \leq \tau_{n-k+1}^2 \sum_{i=n-k+1}^{n} \rho_i^2 = n\tau_{n-k+1}^2$, it is sufficient to have $\rho_{n-k+1}^2 \leq n^{-(1-c)}$. By basic graph spectral theory [Edwards, 2013], we can see that all of the eigenvalues of the lattice network can be written as

$$4\sin^2(\frac{\pi}{2}\frac{i}{\sqrt{n}}) + 4\sin^2(\frac{\pi}{2}\frac{j}{\sqrt{n}})$$

for some $(i, j) \in [\sqrt{n}] \times [\sqrt{n}]$. Thus it is sufficient to ensure

$$4\sin^2(\frac{\pi}{2}\frac{i}{\sqrt{n}}) + 4\sin^2(\frac{\pi}{2}\frac{j}{\sqrt{n}}) \leq 4(\frac{\pi}{2}\frac{i}{\sqrt{n}})^2 + 4(\frac{\pi}{2}\frac{j}{\sqrt{n}})^2 \leq n^{-\frac{1-c}{2}}.$$

For reasonably large n, the proportion of pairs (i, j) satisfying the condition in $[\sqrt{n}] \times [\sqrt{n}]$ is approximately the area ratio between a 1/4 sphere and a square, which is $\frac{1}{4\pi}\frac{n^{\frac{1+c}{2}}}{n}$. Therefore, the number of eigenvalues that satisfies the requirement is at least $Cn^{\frac{1+c}{2}}$ for some constant C.

□

For the easiness of comparison, we also give similar error bounds for the linear null model estimate, which is obtained as

$$\tilde{\boldsymbol{\theta}} = (\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}) = \text{argmin}_{\boldsymbol{\alpha},\boldsymbol{\beta}} \|Y - X\boldsymbol{\beta} - \boldsymbol{\alpha}\|^2 + \lambda\|\boldsymbol{\alpha}\|^2 \tag{A.6}$$

The following proposition shows that in the case of linear regression, the null model gives the same estimate of $\boldsymbol{\beta}$ as OLS.

Lemma A.3. *Let $\tilde{\boldsymbol{\beta}}$ be the estimate from null model. Then we have*

$$\tilde{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{OLS} = (X^TX)^{-1}X^TY.$$

As a result, we have $\tilde{\boldsymbol{\alpha}} = \frac{1}{1+\lambda}(Y - X\hat{\boldsymbol{\beta}}_{OLS})$. *Moreover, the estimation errors for the null model satisfy*

$$\begin{aligned}
\mathrm{MSE}(\tilde{\boldsymbol{\alpha}}) &= \|\boldsymbol{\alpha} - \frac{1}{1+\lambda}P_{X^\perp}\boldsymbol{\alpha}\|^2 + \frac{(n-p)\sigma^2}{(1+\lambda)^2} \\
&\leq \frac{\lambda^2}{(1+\lambda)^2}\|\boldsymbol{\alpha}\|^2 + \frac{1}{(1+\lambda)^2}\|P_{X^\perp}\boldsymbol{\alpha}\|^2 + \frac{(n-p)\sigma^2}{(1+\lambda)^2}, \\
\mathrm{MSE}(\tilde{\boldsymbol{\beta}}) &= \|(X^TX)^{-1}X^T\boldsymbol{\alpha}\|^2 + \sigma^2\mathrm{tr}((X^TX)^{-1}), \\
\mathbb{E}\|\tilde{Y} - \mathbb{E}Y\|^2 &= \frac{\lambda^2}{(1+\lambda)^2}\|P_{X^\perp}\boldsymbol{\alpha}\|^2 + \sigma^2(p + \frac{n-p}{(1+\lambda)^2}).
\end{aligned}$$

In particular, the optimal MSPE is

$$\mathbb{E}\|\tilde{Y} - \mathbb{E}Y\|^2 = \frac{(n-p)\sigma^2\|P_{X^\perp}\boldsymbol{\alpha}\|^2}{(n-p)\sigma^2 + \|P_{X^\perp}\boldsymbol{\alpha}\|^2}$$

which is achieved when $\lambda = \frac{(n-p)\sigma^2}{\|P_{X^\perp}\boldsymbol{\alpha}\|^2}$.

Proof of Lemma A.3. Notice that X is column centered, so we always have $\mathbf{1}^TX = 0$, which ensures

$$\hat{\boldsymbol{\beta}}_{OLS} = (X^TX)^{-1}X^TY.$$

The solution of the null model is given by

$$\begin{bmatrix} \tilde{\boldsymbol{\alpha}} \\ \tilde{\boldsymbol{\beta}} \end{bmatrix} = \begin{bmatrix} (1+\lambda)I_n & X \\ X^T & X^TX \end{bmatrix}^{-1} \begin{bmatrix} Y \\ X^TY \end{bmatrix}. \tag{A.7}$$

By block matrix inverse formula, we have

$$\begin{aligned}
\tilde{\boldsymbol{\beta}} &= -\frac{1+\lambda}{\lambda}\frac{1}{1+\lambda}(X^TX)^{-1}X^TY + \frac{1+\lambda}{\lambda}(X^TX)^{-1}X^TY \\
&= (X^TX)^{-1}X^TY = \hat{\boldsymbol{\beta}}_{OLS}.
\end{aligned}$$

The formula for $\tilde{\boldsymbol{\alpha}}$ and all the error bounds can then be obtained similarly as in Theorem II.6. □

Proof of Theorem II.12. Denote $\ell(\boldsymbol{\alpha} + X\boldsymbol{\beta}; Y)$ by $\ell(\boldsymbol{\theta})$. Define

$$M = \begin{bmatrix} L & 0_{n\times p} \\ 0_{p\times n} & 0_{p\times p} \end{bmatrix}.$$

The matrix M^* is defined similarly. Then by the optimality of $\hat{\boldsymbol{\theta}}^*$ under f^*, we have

$$\begin{aligned} \ell(\hat{\boldsymbol{\theta}}^*) + \lambda\hat{\boldsymbol{\theta}}^{*T} M^* \hat{\boldsymbol{\theta}}^* &= f^*(\hat{\boldsymbol{\theta}}^*) \\ &\leq f^*(\hat{\boldsymbol{\theta}}) \\ &= \ell(\hat{\boldsymbol{\theta}}) + \lambda\hat{\boldsymbol{\theta}}^T M^* \hat{\boldsymbol{\theta}} \\ &\leq \ell(\hat{\boldsymbol{\theta}}) + \lambda(1+\epsilon)\hat{\boldsymbol{\theta}}^T M \hat{\boldsymbol{\theta}}, \end{aligned} \tag{A.8}$$

in which the last inequality can be easily derived from (2.17) by noticing that M^* has all zeros except in the upper left corner. By Taylor expansion of ℓ at $\hat{\boldsymbol{\theta}}$, we have

$$\begin{aligned} \ell(\hat{\boldsymbol{\theta}}^*) &= \ell(\hat{\boldsymbol{\theta}}) + \bigtriangledown\ell(\hat{\boldsymbol{\theta}})^T(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) + \frac{1}{2}(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}})^T \bigtriangledown^2 \ell(\bar{\boldsymbol{\theta}})(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) \\ &= \ell(\hat{\boldsymbol{\theta}}) + \bigtriangledown\ell(\hat{\boldsymbol{\theta}})^T(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) + \frac{1}{2}(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}})^T(\bigtriangledown^2\ell(\bar{\boldsymbol{\theta}}) + 2\lambda M)(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) \\ &\quad - \lambda(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}})^T M(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) \\ &\geq \ell(\hat{\boldsymbol{\theta}}) + \bigtriangledown\ell(\hat{\boldsymbol{\theta}})^T(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) + \frac{m}{2}\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 - \lambda(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}})^T M(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}). \end{aligned} \tag{A.9}$$

In (A.9), $\bar{\boldsymbol{\theta}}$ is some point between $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}^*$ and the last inequality comes from the strong convexity assumption on f. Substituting (A.9) into (A.8) yields

$$\begin{aligned} \frac{m}{2}\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 &\leq - \bigtriangledown \ell(\hat{\boldsymbol{\theta}})^T(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) + \lambda(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}})^T M(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) \\ &\quad + \lambda(1+\epsilon)\hat{\boldsymbol{\theta}}^T M\hat{\boldsymbol{\theta}} - \lambda\hat{\boldsymbol{\theta}}^{*T} M^*\hat{\boldsymbol{\theta}}^* \\ &= - \bigtriangledown \ell(\hat{\boldsymbol{\theta}})^T(\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}) + \lambda(2+\epsilon)\hat{\boldsymbol{\theta}}^T M\hat{\boldsymbol{\theta}} \\ &\quad + \lambda\hat{\boldsymbol{\theta}}^{*T} M\hat{\boldsymbol{\theta}}^* - \lambda\hat{\boldsymbol{\theta}}^{*T} M^*\hat{\boldsymbol{\theta}}^* - 2\lambda\hat{\boldsymbol{\theta}}^T M\hat{\boldsymbol{\theta}}^*. \end{aligned} \tag{A.10}$$

Since $\hat{\boldsymbol{\theta}}$ is the minimizer of f, we have the stationary condition

$$\bigtriangledown\ell(\hat{\boldsymbol{\theta}}) + 2\lambda M\hat{\boldsymbol{\theta}} = \mathbf{0}. \tag{A.11}$$

Substituting (A.11) into (A.10) gives

$$\begin{aligned}
\frac{m}{2}\|\hat{\boldsymbol\theta}^* - \hat{\boldsymbol\theta}\|^2 &\le 2\lambda\hat{\boldsymbol\theta}^T M(\hat{\boldsymbol\theta}^* - \hat{\boldsymbol\theta}) + \lambda(2+\epsilon)\hat{\boldsymbol\theta}^T M\hat{\boldsymbol\theta} + \lambda\hat{\boldsymbol\theta}^{*T} M\hat{\boldsymbol\theta}^* \\
&\quad - \lambda\hat{\boldsymbol\theta}^{*T} M^*\hat{\boldsymbol\theta}^* - 2\lambda\hat{\boldsymbol\theta}^T M\hat{\boldsymbol\theta}^* \\
&= \epsilon\lambda\hat{\boldsymbol\theta}^T M\hat{\boldsymbol\theta} + \lambda\hat{\boldsymbol\theta}^{*T} M\hat{\boldsymbol\theta}^* - \lambda\hat{\boldsymbol\theta}^{*T} M^*\hat{\boldsymbol\theta}^* \\
&\le \epsilon\lambda\hat{\boldsymbol\theta}^T M\hat{\boldsymbol\theta} + \epsilon\lambda\hat{\boldsymbol\theta}^{*T} M\hat{\boldsymbol\theta}^* \\
&\le \epsilon\lambda\hat{\boldsymbol\theta}^T M\hat{\boldsymbol\theta} + \frac{\epsilon}{1-\epsilon}\lambda\hat{\boldsymbol\theta}^{*T} M\hat{\boldsymbol\theta}^* \\
&= \epsilon\lambda\hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha} + \frac{\epsilon}{1-\epsilon}\lambda\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^*, \qquad \text{(A.12)}
\end{aligned}$$

where we use (2.17) again. This gives the bound we need. However, it would be better to have a bound with a dominant term that only depends on $\hat{\boldsymbol\alpha}$ and L. Thus we rearrange the terms as

$$\begin{aligned}
\frac{m}{2}\|\hat{\boldsymbol\theta}^* - \hat{\boldsymbol\theta}\|^2 &\le \epsilon\lambda\hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha} + \frac{\epsilon}{1-\epsilon}\lambda\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^* \\
&\le \epsilon\lambda\hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha} + (1+2\epsilon)\lambda\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^* \\
&= \epsilon\lambda[2\hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha} + (\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^* - \hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha}) + 2\epsilon\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^*] \\
&\le \epsilon\lambda[2\hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha} + |\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^* - \hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha}| + 2\epsilon\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^*], \qquad \text{(A.13)}
\end{aligned}$$

in which the second inequality comes from the fact that $\frac{1}{1-\epsilon} < 1+2\epsilon$ for $\epsilon < 1/2$. Note that we expect $|\hat{\boldsymbol\alpha}^{*T} L\hat{\boldsymbol\alpha}^* - \hat{\boldsymbol\alpha}^T L\hat{\boldsymbol\alpha}|$ to be negligible compared to the first term.

We now proceed to proving the second bound that only involves $\|\hat{\boldsymbol\alpha}\|$. By Taylor expansion, we have, for any $\boldsymbol\theta, \boldsymbol\theta_0 \in \mathbb{R}^n$,

$$\begin{aligned}
f^*(\boldsymbol\theta) &= f^*(\boldsymbol\theta_0) + \bigtriangledown f^*(\boldsymbol\theta_0)^T(\boldsymbol\theta - \boldsymbol\theta_0) + \frac{1}{2}(\boldsymbol\theta - \boldsymbol\theta_0)^T \bigtriangledown^2 f^*(\tilde{\boldsymbol\theta})(\boldsymbol\theta - \boldsymbol\theta_0) \\
&\ge f^*(\boldsymbol\theta_0) + \bigtriangledown f^*(\boldsymbol\theta_0)^T(\boldsymbol\theta - \boldsymbol\theta_0) + \frac{m}{2}\|\boldsymbol\theta - \boldsymbol\theta_0\|^2,
\end{aligned}$$

where the inequality follows from strong convexity. In particular, taking $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}_0 = \hat{\boldsymbol{\theta}}^*$ and noticing that $\bigtriangledown f^*(\hat{\boldsymbol{\theta}}^*) = \mathbf{0}$, we get

$$\|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \leq \frac{2}{m}(f^*(\hat{\boldsymbol{\theta}}) - f^*(\hat{\boldsymbol{\theta}}^*)).$$

Strong convexity also implies (equation (9.9) of [Boyd and Vandenberghe, 2004]) that

$$(f^*(\hat{\boldsymbol{\theta}}) - f^*(\hat{\boldsymbol{\theta}}^*)) \leq \frac{1}{2m}\|\bigtriangledown f^*(\hat{\boldsymbol{\theta}})\|^2.$$

Combining the two parts, we have

$$\text{(A.14)} \qquad \|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \leq \frac{1}{m^2}\|\bigtriangledown f^*(\hat{\boldsymbol{\theta}})\|^2 = \frac{1}{m^2}\|\bigtriangledown f^*(\hat{\boldsymbol{\theta}}) - \bigtriangledown f(\hat{\boldsymbol{\theta}})\|^2,$$

in which the last equality comes from the fact that $\bigtriangledown f(\hat{\boldsymbol{\theta}}) = \mathbf{0}$. From (2.18), the gradients of f and f^* are

$$\bigtriangledown f(\hat{\boldsymbol{\theta}}) = \bigtriangledown \ell + 2\lambda M\hat{\boldsymbol{\theta}}, \quad \bigtriangledown f^*(\hat{\boldsymbol{\theta}}) = \bigtriangledown \ell + 2\lambda M^*\hat{\boldsymbol{\theta}}.$$

Thus the difference between $\hat{\boldsymbol{\theta}}^*$ and $\hat{\boldsymbol{\theta}}$ can be bounded by

$$\text{(A.15)} \qquad \|\hat{\boldsymbol{\theta}}^* - \hat{\boldsymbol{\theta}}\|^2 \leq \frac{1}{m^2}\|2\lambda(M - M^*)\hat{\boldsymbol{\theta}}\|^2.$$

Finally, from (2.17), we obtain

$$\begin{aligned} \|2\lambda(M - M^*)\hat{\boldsymbol{\theta}}\|^2 &= \|2\lambda(L - L^*)\hat{\boldsymbol{\alpha}}\|^2 \\ &\leq 4\lambda^2\|L - L^*\|_2^2\|\hat{\boldsymbol{\alpha}}\|^2 \\ \text{(A.16)} \qquad &\leq 4\lambda^2\epsilon^2\|L\|_2^2\|\hat{\boldsymbol{\alpha}}\|^2. \end{aligned}$$

Combining (A.15) and (A.16) yields the second bound and completes the proof. □

A.2 Complexity of solving RNC estimator by block elimination

We calculate the complexity of solving RNC estimator here assuming the block elimination strategy described in Section 2.2.6 is used. The first major part is solving an $n \times n$ sparse symmetric diagonal dominant system to obtain $(I + \lambda L)^{-1}X$ and $(I + \lambda L)^{-1}\boldsymbol{b}_1$ in the estimator. Using the linear system notations, we want to solve

$$A\boldsymbol{x} = \boldsymbol{b}$$

where $A = I + \lambda L$. Naively solve it by Cholesky decomposition ignoring special structures would result in $O(n^3)$ operations. When A is sparse as in a great many of applications, we can first find a permutation matrix P to permute A and then find sparse factorization for the resulting permuted matrix

$$PAP^T = LL^T.$$

The operation counts in this step depends on the heuristic algorithm to find a good permutation, the number of nonzero elements in A (which is $n + 2|E|$ in our setting) and the positions of these nonzeros (depicted by the network). Roughly speaking, it depends on $\sum_i d_i^2$ [Spielman, 2010]. Though the general complexity is not available, it is shown in Lipton et al. [1979] that the complexity for the network transformed from a $\sqrt{n} \times \sqrt{n}$ grid is $O(n^{3/2})$ by using an algorithm called George's Nested Dissection. Solving both $(I + \lambda L)^{-1}X$ and $(I + \lambda L)^{-1}\boldsymbol{b}_1$ thus requires $O(n^{3/2} + pn)$ and when n dominates p, we just have $O(n^{3/2})$ there. We refer readers to Lipton et al. [1979] for details.

Alternatively, one can solve the system approximately by iterative methods [Spielman, 2010, Koutis et al., 2010]. In particular, Koutis et al. [2010] propose an iterative algorithm with preconditioning such that for any n-node network, an approximate solution $\hat{\boldsymbol{x}}$ of accuracy

$$\|\hat{\boldsymbol{x}} - A^{-1}\boldsymbol{b}\|_A < \epsilon\|A^{-1}\boldsymbol{b}\|_A$$

can be computed in expected time $O(m \log^2 n \log(1/\epsilon))$ where $m = n + 2|E|$ and the A-norm is defined by

$$\|\boldsymbol{x}\|_A = \sqrt{\boldsymbol{x}^T A \boldsymbol{x}}.$$

To solve both $(I+\lambda L)^{-1}X$ and $(I+\lambda L)^{-1}\boldsymbol{b}_1$, this is expected to takes $O(pm \log^2 n \log(1/\epsilon))$ operations. Notice that even if A is fully dense with n^2 nonzero entries, the cost is still much lower than the naive solving.

The rest steps in the block elimination only involve matrix multiplications and general solving for a $p \times p$ symmetric positive definite system. The order is then $O(np^2 + p^3)$, the same as OLS procedure.

In summary, if one tries to compute the estimator exactly, the order depends on the network connecting the samples. When the network is from a $\sqrt{n} \times \sqrt{n}$ grid, the complexity is in the order of $O(n^{3/2} + pn + np^2 + p^3)$. If approximate methods are used instead, the order is expected to be $O(p(n + 2|E|) \log^3 n + np^2 + p^3)$ for general networks with high accuracy (taking approximation tolerance $\epsilon = O(1/n)$).

Both of dense and sparse Cholesky factorizations can be further parallelized on modern distributed systems [Bosilca et al., 2012, Faverge and Ramet, 2008, Lacoste et al., 2014], when high computational performance is needed. The complexity in such settings heavily depends the systems.

A.3 Coefficients of recreational activity linear models

In the example of Section 2.5.1, we use linear regression to predict recreational activity level from nine demographic covariates. The covariate coefficients from OLS and RNC regressions are shown in Table A.1. Most of the coefficients are similar for the two models, suggesting that most of the variables do not contain (or mask) network structural information. The only covariate that is relatively significant in OLS but has a substantially smaller effect in RNC is the indicator variable "race-black". This suggests that race follows a network cohesion pattern, and thus is not as important for RNC since it is already getting network information elsewhere.

category (contrast)	covariate	OLS	RNC	p-value (OLS)
	age	-0.086	-0.088	0.065
sex (male)	female	0.229	0.241	0.003
grade (other)	grade11to12	0.206	0.212	0.078
race (other)	white	0.023	-0.029	0.733
	black	0.539	0.426	0.007
	Asian	0.346	0.512	0.081
	native	0.369	0.252	0.407
born in U.S. (no)	yes	-0.059	0.090	0.290
living with mother (no)	yes	0.095	0.162	0.251
living with father (no)	yes	-0.089	-0.047	0.620
parents in labor market (no)	yes	-0.193	0.018	0.957
mother education (no high school)	high school	-0.039	-0.021	0.861
	more than high school	-0.116	-0.000	1.000
	college	0.108	0.163	0.226
	unknown	-0.061	-0.051	0.681
father education (no high school)	high school	-0.132	-0.127	0.326
	more than high school	0.012	-0.040	0.814
	college	-0.049	-0.026	0.853
	unknown	-0.330	-0.336	0.006

Table A.1: Estimated covariate coefficients from OLS and RNC linear regression on the recreational activity example. The p-values are for the OLS estimate.

A.4 Sensitivity to missing data

The AddHealth data set contains many records with missing values, and we used imputation in both examples to handle the missing data. Here we report results of a sensitivity analysis to the amount of missing data.

In the recreational activity example, we remove an additional fraction p_m of records in each column at random, where p_m varies from 0 to 0.5; if the original column was missing m values, it will now be missing $m(1 + p_m)$ records. When $p_m = 0$, the results match the ones reported in Section 2.4. Table A.2 shows the corresponding RMSEs for the full model with all 10 predictors, calculated in the same way as in Section 2.4, for a range of values of p_m. The relative rankings of the five models never change, although there are some small numerical changes in the errors. This very robust performance suggests that our results are not sensitive to proportion of missing data.

p_m	OLS & Null	SIM	RNC	RNC-LA	oracle-Bayes
0%	1.219 **	1.188 **	1.163	1.176 *	1.175 *
10%	1.219 **	1.186 **	1.163	1.174 *	1.171
20%	1.216 **	1.184 **	1.160	1.172 *	1.169 *
30%	1.218 **	1.188 **	1.164	1.174 *	1.174 *
40%	1.220 **	1.198 **	1.167	1.186 **	1.179 *
50%	1.216 **	1.185 **	1.163	1.175 *	1.172

Table A.2: Prediction errors of five models with missing data imputation, with varying proportion of additional missing values. All other columns are compared with RNC by a paired two-sample t-test. ** indicates a p-value $\leq 10^{-4}$ and * indicates a p-value $\in (10^{-4}, 10^{-2})$.

In the marijuana usage example, we conduct the same experiment. However, the number of records with missing values is much smaller in the home-survey data (used for marijuana example) than the school survey data (used in the recreational activity example). Therefore, we take a larger range for p_m from 0 to 2. The results of the corresponding prediction iAUC for the four models (with all the five variables) are shown in Table A.3. Again, the relative ranking is the same as in Section 2.4 for all different values of p_m. Moreover, the iAUCs are also very stable across different settings of p_m.

p_m	Cox & Null	SIM	RNC	RNC-LA
0	0.727 **	0.743	0.748	0.766 **
0.5	0.727 **	0.743	0.748	0.766 **
1	0.726 **	0.742	0.747	0.765 **
1.5	0.726 **	0.742	0.747	0.765 **
2	0.729 **	0.745	0.749	0.767 **

Table A.3: Average integrated AUC (iAUC) for survival prediction ROC curves for age 14-17 with artificially increased missing values (by p_m). The average is taken over 50 random splits of the data into 60 test samples and 587 training samples. All values are compared with the columns of RNC by a paired two-sample t-test. ** indicates a p-value $\leq 10^{-4}$ and * indicates a p-value $\in (10^{-4}, 10^{-2})$.

APPENDIX B

Appendix for Chapter III

B.1 Proofs

We start with additional notation. For any vector $\boldsymbol{x}$, we use $\|\boldsymbol{x}\|$ to denote its Euclidean norm. We denote the singular values of a matrix P by $\sigma_1(P) \geq \sigma_2(P) \geq \cdots \sigma_K(P) > \sigma_{K+1}(P) = \sigma_{K+2}(P) \cdots \sigma_n(P) = 0$, where $K = \text{rank}(P)$. Recall the Frobenius norm $\|P\|_F$ is defined by $\|P\|_F^2 = \sum_{ij} P_{ij}^2 = \sum_i \sigma_i(P)^2$, the spectral norm $\|P\| = \sigma_1(P)$, the infinity norm $\|P\|_\infty = \max_{ij} |P_{ij}|$, and the nuclear norm $\|P\|_* = \sum_i \sigma_i(P)$ be the nuclear norm. In addition, the max norm of P [Srebro and Shraibman, 2005] is defined as

$$\|P\|_{\max} = \min_{P=UV^T} \max(\|U\|_{2,\infty}^2, \|V\|_{2,\infty}^2),$$

where $\|U\|_{2,\infty} = \max_i (\sum_j U_{ij}^2)^{1/2}$.

We will need the following well-known inequalities:

$$\|P\| \leq \|P\|_F \leq \sqrt{K}\|P\|, \tag{B.1}$$

$$\|P\|_F \leq \|P\|_* \leq \sqrt{K}\|P\|_F \tag{B.2}$$

$$|\text{tr}(P_1^T P_2)| \leq \|P_1\|\|P_2\|_* \tag{B.3}$$

$$\max(\|P^T\|_{2,\infty}, \|P\|_{2,\infty}) \leq \|P\| \tag{B.4}$$

$$\|P\|_{\max} \leq \sqrt{K}\|P\|_\infty. \tag{B.5}$$

Relationship (B.3), which holds for any two matrices P_1, P_2 with matching dimensions, is called norm duality for the spectral norm and the nuclear norm [Boyd and Vandenberghe, 2004]. Relationship (B.5) can be found in Srebro and Shraibman [2005]. The last one we need is the variational property of spectral norm:

$$\|P\| = \max_{\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^n:\|\boldsymbol{x}\|=\|\boldsymbol{y}\|=1} \boldsymbol{y}^T P\boldsymbol{x}. \tag{B.6}$$

Our proof will rely on a concentration result for the adjacency matrix. To the best of our knowledge, Lemma C.3 stated next is the best concentration bound currently available, proved by Lei and Rinaldo [2014]. The same concentration was also obtained by Chin et al. [2015] and Le et al. [2017].

Lemma B.1. *Let A be the adjacency matrix of a random graph on n nodes with independent edges. Set $\mathbb{E}(A) = P = [p_{ij}]_{n\times n}$ and assume that $n\max_{ij} p_{ij} \leq d$ for $d \geq C_0 \log n$ and $C_0 > 0$. Then for any $\delta > 0$, there exists a constant $C = C(\delta, C_0)$ such that*

$$\|A - P\| \leq C\sqrt{d}$$

with probability at least $1 - n^{-\delta}$.

Another tool we need is the discrepancy between a bounded matrix and its partially observed version given in Lemma B.2, which can be viewed as a generalization of Theorem 4.1 of Bhojanapalli and Jain [2014] and Lemma 6.4 of Bhaskar and Javanmard [2015] to the more realistic uniform missing mechanism in the matrix completion problem. Let $G \in \mathbb{R}^{n\times n}$ be the indicator matrix associated with the hold-out set Ω, such that if $(i,j) \in \Omega$, $G_{ij} = 0$ and otherwise $G_{ij} = 1$. Note that under the uniform missing mechanism, G can be viewed as an adjacency matrix of an Erdös-Renyi random graph where all edges appear independently with probability p. Note that $P_\Omega A = A \circ G$ where $\circ$ is the Hadamard (element-wise) matrix product.

Lemma B.2. *Let G an adjacency matrix of an Erdös-Renyi graph with the probability of edge $p \geq C_1 \log n/n$ for a constant C_1. Then for any $\delta > 0$, with probability at least $1 - n^{-\delta}$, the following relationship holds for any $Z \in \mathbb{R}^{n\times n}$ with* $\operatorname{rank}(Z) \leq K$

$$\left\|\frac{1}{p}Z \circ G - Z\right\| \leq 2C\sqrt{\frac{nK}{p}}\|Z\|_\infty$$

where $C = C(\delta, C_1)$ is the constant from Lemma C.3 that only depends on δ and C_1.

Proof of Lemma B.2. Let $Z = UV^T$, where $U \in \mathbb{R}^{n\times K}$ and $V \in \mathbb{R}^{n\times K}$ are the matrices that achieve the minimum in the definition of $\|Z\|_{\max}$. Denote the ℓth column of U by $U_{\cdot\ell}$ and the ℓth row by $U_{\ell\cdot}$.

Given any unit vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we have

$$\begin{aligned}\boldsymbol{y}^T\left(\frac{1}{p}Z \circ G - Z\right)\boldsymbol{x} &= \sum_\ell \left[\frac{1}{p}\boldsymbol{y}^T(U_{\cdot\ell}V_{\cdot\ell}^T) \circ G\boldsymbol{x} - (\boldsymbol{y}^T U_{\cdot\ell})(\boldsymbol{x}^T V_{\cdot\ell})\right] \\ &= \sum_\ell \left[\frac{1}{p}(\boldsymbol{y} \circ U_{\cdot\ell})^T G(\boldsymbol{x} \circ V_{\cdot\ell}) - (\boldsymbol{y}^T U_{\cdot\ell})(\boldsymbol{x}^T V_{\cdot\ell})\right]. \qquad \text{(B.7)}\end{aligned}$$

Let $\tilde{\mathbf{1}} = \mathbf{1}_n/\sqrt{n}$ be the constant unit vector. For any $1 \leq \ell \leq n$, let $\boldsymbol{y} \circ U_{\cdot\ell} = \alpha_\ell \tilde{\mathbf{1}} + \beta_\ell \tilde{\mathbf{1}}_\perp^\ell$ in which $\tilde{\mathbf{1}}_\perp^\ell$ is a vector that is orthogonal to $\tilde{\mathbf{1}}$. It is easy to check that

$$\alpha_\ell = (\boldsymbol{y} \circ U_{\cdot\ell})^T \tilde{\mathbf{1}} = \frac{1}{\sqrt{n}}\boldsymbol{y}^T U_{\cdot\ell}.$$

Similarly, we also have

$$(\boldsymbol{x} \circ V_{\cdot\ell})^T \tilde{\mathbf{1}} = \frac{1}{\sqrt{n}}\boldsymbol{x}^T V_{\cdot\ell}.$$

Let $\bar{G} = p\mathbf{1}\mathbf{1}^T$ be the expectation of G with respect to the missing mechanism. Then

$$\begin{aligned}(\boldsymbol{y} \circ U_{\cdot\ell})^T G(\boldsymbol{x} \circ V_{\cdot\ell}) &= \frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T G(\boldsymbol{x} \circ V_{\cdot\ell}) + \beta_\ell \tilde{\mathbf{1}}_\perp^{\ell\,T} G(\boldsymbol{x} \circ V_{\cdot\ell}) \\ &\text{(B.8)} \\ &= \frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T \bar{G}(\boldsymbol{x} \circ V_{\cdot\ell}) + \frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T (G - \bar{G})(\boldsymbol{x} \circ V_{\cdot\ell}) + \beta_\ell \tilde{\mathbf{1}}_\perp^{\ell\,T} G(\boldsymbol{x} \circ V_{\cdot\ell})\ .\end{aligned}$$

Notice that $\tilde{\mathbf{1}}^T\bar{G} = np\tilde{\mathbf{1}}^T$, and therefore

$$\frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T\bar{G}(\boldsymbol{x}\circ V_{\cdot\ell}) = \frac{np}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T(\boldsymbol{x}\circ V_{\cdot\ell})\ . = p(\boldsymbol{y}^T U_{\cdot\ell})(\boldsymbol{x}^T V_{\cdot\ell})$$

Further, since $\bar{G}\tilde{\mathbf{1}}_\perp^\ell = 0$ for any ℓ, we can rewrite (B.8) as

$$\begin{aligned}(\boldsymbol{y}\circ U_{\cdot\ell})^T G(\boldsymbol{x}\circ V_{\cdot\ell}) &= p(\boldsymbol{y}^T U_{\cdot\ell})(\boldsymbol{x}^T V_{\cdot\ell})\\ &+ \frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T(G-\bar{G})(\boldsymbol{x}\circ V_{\cdot\ell}) + \beta_\ell \tilde{\mathbf{1}}_\perp^{\ell\,T}(G-\bar{G})(\boldsymbol{x}\circ V_{\cdot\ell})\ . \end{aligned} \tag{B.9}$$

Substituting (B.9) into (B.7) and applying (B.6) and the Cauchy-Schwarz inequality leads to

$$\begin{aligned}\boldsymbol{y}^T(\frac{1}{p}P_\Omega Z - Z)\boldsymbol{x} &= \frac{1}{p}\sum_\ell\left[\frac{1}{\sqrt{n}}(\boldsymbol{y}^T U_{\cdot\ell})\tilde{\mathbf{1}}^T(G-\bar{G})(\boldsymbol{x}\circ V_{\cdot\ell}) + \beta_\ell\tilde{\mathbf{1}}_\perp^{\ell\,T}(G-\bar{G})(\boldsymbol{x}\circ V_{\cdot\ell})\right]\\ &\le \frac{1}{p}\|G-\bar{G}\|\left[\sum_\ell \frac{1}{\sqrt{n}}|\boldsymbol{y}^T U_{\cdot\ell}|\|\boldsymbol{x}\circ V_{\cdot\ell}\| + \sum_\ell |\beta_\ell|\|\boldsymbol{x}\circ V_{\cdot\ell}\|\right]\\ &\le \frac{1}{p}\|G-\bar{G}\|\left[\frac{1}{\sqrt{n}}\sqrt{\sum_\ell(\boldsymbol{y}^T U_{\cdot\ell})^2}\sqrt{\sum_\ell\|\boldsymbol{x}\circ V_{\cdot\ell}\|^2} + \sqrt{\sum_\ell\beta_\ell^2}\sqrt{\sum_\ell\|\boldsymbol{x}\circ V_{\cdot\ell}\|^2}\right].\end{aligned} \tag{B.10}$$

Using Cauchy-Schwarz inequality, the definition of max norm and the relationship (B.5), we get

$$\sum_\ell(\boldsymbol{y}^T U_{\cdot\ell})^2 \le \sum_\ell\|\boldsymbol{y}\|^2\|U_{\cdot\ell}\|^2 = \|U\|_F^2 \le n\|U\|_{2,\infty}^2 \le n\|Z\|_{\max} \le n\sqrt{K}\|Z\|_\infty\ . \tag{B.11}$$

Similarly,

$$\begin{aligned}\sum_\ell\beta_\ell^2 &= \sum_\ell(\tilde{\mathbf{1}}_\perp^{\ell\,T}(\boldsymbol{y}\circ U_{\cdot\ell}))^2 \le \sum_\ell\|\boldsymbol{y}\circ U_{\cdot\ell}\|^2\\ &= \sum_\ell\sum_i y_i^2 U_{il}^2 \le \|U\|_{2,\infty}^2\sum_i y_i^2 \le \|Z\|_{\max} \le \sqrt{K}\|Z\|_\infty.\end{aligned} \tag{B.12}$$

We also have

$$\sum_\ell\|\boldsymbol{x}\circ V_{\cdot\ell}\|^2 \le \sqrt{K}\|Z\|_\infty. \tag{B.13}$$

Combining (B.11), (B.12) and (B.13) with (B.10), we get

$$\boldsymbol{y}^T(\frac{1}{p}P_\Omega Z - Z)\boldsymbol{x} \leq \frac{2\sqrt{K}}{p}\|G-\bar{G}\|\|Z\|_\infty. \tag{B.14}$$

From (B.6), we have

$$\|\frac{1}{p}P_\Omega Z - Z\| = \sup_{\|\boldsymbol{x}\|=\|\boldsymbol{y}\|=1} \boldsymbol{y}^T(\frac{1}{p}P_\Omega Z - Z)\boldsymbol{x} \leq \frac{2\sqrt{K}}{p}\|G-\bar{G}\|\|Z\|_\infty.$$

Finally, Lemma C.3 implies

$$\|G-\bar{G}\| \leq C(\delta, C_1)\sqrt{pn} \tag{B.15}$$

with probability at least $1-n^{-\delta}$ defined in Lemma C.3. Therefore, with probability at least $1-n^{-\delta}$,

$$\|\frac{1}{p}P_\Omega Z - Z\| \leq 2C(\delta, C_1)\sqrt{\frac{nK}{p}}\|Z\|_\infty.$$

□

The following lemma is from Klopp [2015]. See also Corollary 3.3 of Bandeira et al. [2016] for a more general results in expectation form.

Lemma B.3 (Proposition 13 of [Klopp, 2015])**.** *Let X be an $n \times n$ matrix with each entry X_{ij} being independent and bounded random variables, such that* $\max_{ij}|X_{ij}| \leq \sigma$ *with probability 1. Then for any $\delta > 0$,*

$$\|X\| \leq C' \max(\sigma_1, \sigma_2, \sqrt{\log n})$$

in which $C' = C'(\sigma, \delta)$ is a constant that only depends on δ and σ,

$$\sigma_1 = \max_i \sqrt{\mathbb{E}\sum_j X_{ij}^2}$$

and

$$\sigma_2 = \max_j \sqrt{\mathbb{E}\sum_i X_{ij}^2}.$$

Proof of Theorem III.6. Our proof is valid weather or not the network is undirected, as Lemma C.3 holds for both directed and undirected networks. So we would proceed ignoring that P can be potentially symmetric. Let $W = A - P$, so $\mathbb{E}W = 0$. It is known that

$$S_H(\frac{1}{p}P_\Omega A, K) = \text{argmin}_{P:\text{rank}(P)\leq K}\|\frac{1}{p}P_\Omega A - P\|. \tag{B.16}$$

By this property, we have

$$\begin{aligned}
\|\hat{A} - P\| &= \|\hat{A} - \frac{1}{p}P_\Omega A + \frac{1}{p}P_\Omega A - P\| \\
&\leq \|\frac{1}{p}P_\Omega A - \hat{A}\| + \|\frac{1}{p}P_\Omega A - P\| \\
&\leq 2\|\frac{1}{p}P_\Omega A - P\| \\
&\leq 2\|\frac{1}{p}P_\Omega P - P + \frac{1}{p}P_\Omega W\| \\
&\leq 2\|\frac{1}{p}P_\Omega P - P\| + \frac{2}{p}\|P_\Omega W\| \\
&= 2\|\frac{1}{p}G \circ P - P\| + \frac{2}{p}\|G \circ W\| := \mathcal{I} + \mathcal{II}.
\end{aligned}$$

Since $\text{rank}(P) \leq K$, by Lemma B.2, we have

$$\mathcal{I} \leq 4C(\delta, C_1)\sqrt{\frac{nK}{p}}\|Z\|_\infty \leq 4C(\delta, C_1)\sqrt{\frac{Kd^2}{np}} \tag{B.17}$$

with probability at least $1 - n^{-\delta}$ for any $\delta > 0$.

We want to apply the result of Lemma B.3 to control $\mathcal{II}$, by conditioning on W. Notice that $(G \circ W)_{ij} = \eta_{ij}W_{ij}$ where $\eta_{ij} \sim B(p)$. Clearly we can set $\sigma = 1$ in the lemma. Also,

$$\begin{aligned}
\sigma_1 &= \max_i \sqrt{\mathbb{E}(\sum_j \eta_{ij}^2 W_{ij}^2 | W)} = \max_i \sqrt{\sum_j W_{ij}^2 \mathbb{E}(\eta_{ij}^2 | W)} \\
&= \max_i \sqrt{p}\sqrt{\sum_j W_{ij}^2} = \max_i \sqrt{p}\sqrt{\|W_{i\cdot}\|_2^2} \\
&= \sqrt{p}\sqrt{\|W\|_{2,\infty}^2} \leq \sqrt{p}\|W\|
\end{aligned}$$

in which the last inequality comes from (B.4). Similarly, we have

$$\sigma_2 = \max_j \sqrt{\mathbb{E}(\sum_i \eta_{ij}^2 W_{ij}^2 | W)} \leq \sqrt{p}\|W\|.$$

Now by Lemma B.3, we know that given W,

$$\mathcal{II} = \frac{2}{p}\|G \circ W\| \leq \frac{2}{p}C'(\delta)(\sqrt{p}\|W\| \vee \sqrt{\log n}) \tag{B.18}$$

with probability at least $1 - n^{-\delta}$ where $C'(\delta)$ is the $C'(1, \delta)$ in Lemma B.3.

Finally, applying Lemma C.3 to (B.18), we have for any $\delta_2, \delta_3 > 0$

$$\mathcal{II} \leq \frac{2}{p}C'(\delta)\max(C(\delta, C_2)\sqrt{p}\sqrt{d}, \sqrt{\log n}) \leq C''(\delta, C_2)\frac{\max(\sqrt{pd}, \sqrt{\log n})}{p} \tag{B.19}$$

with probability at least $1 - 2n^{-\delta}$ where $C''(\delta, C_2) = 2C'(\delta)\max(C(\delta, C_2), 1)$.

Combining (B.17) and (B.19) gives

$$\|\hat{A} - P\| \leq \mathcal{I} + \mathcal{II} \leq \tilde{C}\max(\sqrt{\frac{Kd^2}{np}}, \sqrt{\frac{d}{p}}, \frac{\sqrt{\log n}}{p})$$

with probability at least $1 - 3n^{-\delta}$ where $\tilde{C}(\delta, C_1, C_2) = 4C(\delta, C_1) + C''(\delta, C_2)$.

The bound about Frobenius norm (3.4) directly comes from (B.1) since $\text{rank}(\hat{A} - P) \leq 2K$.

□

Proof of Proposition III.9 and III.11. A direct consequence of Theorem III.6 is the concentration bound

$$\|\hat{A} - P\| \leq C\sqrt{d}$$

with high probability. Then the conclusion of Proposition III.9 can be proved following the strategy of Corollary 3.2 of Lei and Rinaldo [2014]. The same concentration bound also holds for DCSBM. To prove Proposition III.11, recall that

$n_k = |\{i : c_i = k\}|$. Following Lei and Rinaldo [2014], define $\boldsymbol{\theta}_k = \{\theta_i\}_{c_i=k}$ and

$$\nu_k = \frac{1}{n_k^2} \sum_{i:c_i=k} \frac{\|\boldsymbol{\theta}_k\|^2}{\theta_i^2}.$$

Let $\tilde{n}_k = \|\boldsymbol{\theta}_k\|^2$ be the "effective size" of the kth community. Under III.10, we have

$$\nu_k \leq \frac{1}{n_k^2} \sum_{i:c_i=k} \frac{n_k}{\theta_0^2} = \frac{1}{\theta_0^2}.$$

Furthermore, when III.8 and III.10 hold, we have

$$\text{(B.20)} \qquad \frac{\sum_k n_k^2 \nu_k^2}{\min_k \tilde{n}_k^2} \leq \frac{\sum_k n_k^2 \nu_k^2}{\min_k n_k^2 \theta_0^4} \leq \frac{\sum_k n_k^2}{\gamma^2 \theta_0^8} \leq \frac{K}{\gamma^2 \theta_0^8} = O(1).$$

Proposition III.11 can then be proved by following the proof of Corollary 4.3 of Lei and Rinaldo [2014] and applying (B.20).

□

B.2 Additional simulation results for model selection under the block models

B.2.1 Using binomial deviance loss function for overall block model selection

As discussed in the paper, we can use both L_2 loss and binomial deviance as loss functions in selecting between different block models. Empirically we found the L_2 loss gives better results, shown in Section 3.4. For completeness, we include overall block model selection correct rate using binomial deviance for both ECV (ECV-dev) and NCV (NCV-dev) in Table B.1 (when the true model is DCSBM) and Table B.2 (when the true model is SBM). The pattern is the same as for the L_2 loss; both methods benefit from stability selection and ECV always dominates NCV. The difference between the two methods is very large under the DCSBM and smaller under the SBM.

K	n	λ	t	β	ECV-dev	ECV-dev-mode	NCV-dev	NCV-dev-mode
3	600	15	0	0.2	0.47	0.45	0.00	0.00
		20	0	0.2	0.89	0.96	0.00	0.00
		30	0	0.2	1.00	1.00	0.26	0.14
		40	0	0.2	1.00	1.00	0.84	0.96
5	600	15	0	0.2	0.34	0.40	0.00	0.00
		20	0	0.2	0.82	0.93	0.00	0.00
		30	0	0.2	0.97	1.00	0.01	0.00
		40	0	0.2	0.99	1.00	0.13	0.10
5	1200	15	0	0.2	0.45	0.53	0.00	0.00
		20	0	0.2	0.94	0.98	0.00	0.00
		30	0	0.2	1.00	1.00	0.00	0.00
		40	0	0.2	1.00	1.00	0.23	0.15
3	600	40	0	0.2	1.00	1.00	0.86	0.95
		40	0.25	0.2	1.00	1.00	0.89	0.94
		40	0.5	0.2	1.00	1.00	0.89	0.95
		40	1	0.2	0.64	0.71	0.29	0.41
5	600	40	0	0.2	0.97	1.00	0.16	0.10
		40	0.25	0.2	0.98	1.00	0.12	0.09
		40	0.5	0.2	0.72	0.79	0.07	0.04
		40	1	0.2	0.12	0.07	0.07	0.03
5	1200	40	0	0.2	1.00	1.00	0.21	0.17
		40	0.25	0.2	1.00	1.00	0.21	0.15
		40	0.5	0.2	0.81	0.82	0.09	0.04
		40	1	0.2	0.09	0.07	0.02	0.01
3	600	40	0	0.1	1.00	1.00	0.99	1.00
		40	0	0.2	1.00	1.00	0.85	0.96
		40	0	0.5	0.96	0.97	0.00	0.00
5	600	40	0	0.1	0.99	1.00	0.59	0.82
		40	0	0.2	0.98	1.00	0.18	0.10
		40	0	0.5	0.00	0.00	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	0.93	1.00
		40	0	0.2	0.99	1.00	0.22	0.15
		40	0	0.5	0.00	0.00	0.00	0.00

Table B.1: Overall block model selection correct rate of ECV and NCV in 200 replications when binomial deviance is used as the loss function. The underlying true model is DCSBM.

K	n	λ	t	β	ECV-dev	ECV-dev-mode	NCV-dev	NCV-dev-mode
3	600	15	0	0.2	1.00	1.00	0.98	1.00
		20	0	0.2	1.00	1.00	0.99	1.00
		30	0	0.2	1.00	1.00	0.99	1.00
		40	0	0.2	1.00	1.00	0.99	1.00
5	600	15	0	0.2	0.82	0.89	0.71	0.87
		20	0	0.2	0.99	1.00	0.97	1.00
		30	0	0.2	0.99	1.00	0.98	1.00
		40	0	0.2	1.00	1.00	0.97	1.00
5	1200	15	0	0.2	0.97	0.98	0.92	0.96
		20	0	0.2	1.00	1.00	1.00	1.00
		30	0	0.2	1.00	1.00	0.96	1.00
		40	0	0.2	1.00	1.00	0.96	1.00
3	600	40	0	0.2	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	0.99	1.00
		40	0.5	0.2	1.00	1.00	0.95	1.00
		40	1	0.2	0.67	0.73	0.26	0.38
5	600	40	0	0.2	0.99	1.00	0.94	1.00
		40	0.25	0.2	1.00	1.00	0.95	1.00
		40	0.5	0.2	0.80	0.86	0.58	0.74
		40	1	0.2	0.12	0.04	0.21	0.12
5	1200	40	0	0.2	1.00	1.00	0.97	1.00
		40	0.25	0.2	1.00	1.00	0.96	1.00
		40	0.5	0.2	0.93	0.94	0.68	0.84
		40	1	0.2	0.05	0.01	0.23	0.11
3	600	40	0	0.1	1.00	1.00	0.98	1.00
		40	0	0.2	1.00	1.00	0.99	1.00
		40	0	0.5	0.94	0.97	0.85	0.97
5	600	40	0	0.1	0.99	1.00	0.96	1.00
		40	0	0.2	0.99	1.00	0.93	1.00
		40	0	0.5	0.00	0.00	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	0.97	1.00
		40	0	0.2	1.00	1.00	0.97	1.00
		40	0	0.5	0.00	0.00	0.00	0.00

Table B.2: Overall block model selection correct rate of ECV and NCV in 200 replications when binomial deviance is used as the loss function. The underlying true model is SBM.

B.2.2 Selecting the number of communities under the SBM

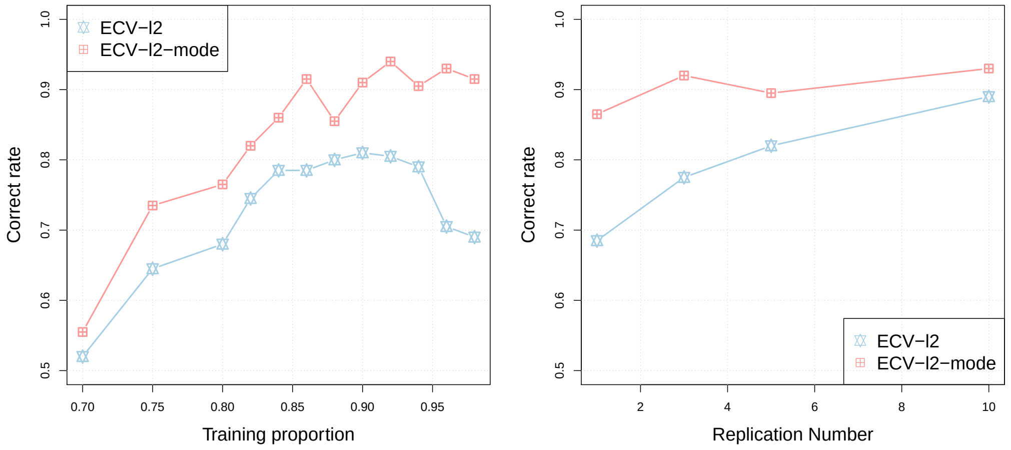
Table 3.8 in the paper shows the accuracy of selecting K from multiple methods under the DCSBM, and results under the SBM are given in Table B.3 below. The pattern is similar, except ECV-AUC has a problem with perfectly separated communities ($\beta = 0$, an unrealistic scenario, presumably due to many ties affecting the AUC). LR-BIC is more robust than BHmc to unbalanced community sizes but is the most vulnerable of all methods to high out-in ratio.

B.2.3 The impact of training proportion p and replication number N

This simulation study illustrates the impact of p and N on the performance of ECV on the task of block model selection considered in Section 3.4.2. The true model is the DCSBM with $K = 3$ equal-sized communities, $n = 600$, average degree 15, and the out-in ratio 0.2. The results are averaged over 200 replications. Figures B.1 and B.2 show the effects of varying p and N on model selection and estimation of K, respectively. Clearly, a small p will not produce enough data to fit the model accurately. A very large p is also not ideal since the test set will be very small so the validation becomes noisy. The larger the number of replications N, the better in general. The stability selection step makes our procedure much more robust to the choice of p and N, with similar performance for $p > 0.85$ and all values of N considered. In all our examples in the paper, we use $p = 0.9, N = 3$.

Configurations					Method				
K	n	λ	t	β	ECV-l2-avg	NCV-l2-mode	ECV-AUC-avg	LR-BIC	BHmc
3	600	15	0	0.2	1.00	1.00	1.00	1.00	1.00
		20	0	0.2	1.00	1.00	1.00	1.00	0.99
		30	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
5	600	15	0	0.2	0.89	0.86	0.89	0.99	1.00
		20	0	0.2	1.00	1.00	1.00	1.00	1.00
		30	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	0.99	1.00	1.00
5	1200	15	0	0.2	0.98	0.96	0.99	1.00	1.00
		20	0	0.2	1.00	1.00	1.00	1.00	1.00
		30	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
3	600	40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.5	0.2	1.00	1.00	1.00	1.00	1.00
		40	1	0.2	0.72	0.44	0.85	1.00	1.00
5	600	40	0	0.2	1.00	1.00	0.98	1.00	1.00
		40	0.25	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.5	0.2	0.86	0.78	0.88	1.00	0.99
		40	1	0.2	0.05	0.10	0.01	0.72	0.05
5	1200	40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.25	0.2	1.00	1.00	1.00	1.00	1.00
		40	0.5	0.2	0.95	0.89	0.96	1.00	1.00
		40	1	0.2	0.03	0.06	0.01	0.79	0.07
3	600	40	0	0.1	1.00	1.00	0.93	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.5	0.95	0.96	1.00	0.82	1.00
5	600	40	0	0.1	1.00	1.00	0.76	1.00	1.00
		40	0	0.2	1.00	1.00	0.99	1.00	1.00
		40	0	0.5	0.00	0.00	0.05	0.00	0.00
5	1200	40	0	0.1	1.00	1.00	0.88	1.00	1.00
		40	0	0.2	1.00	1.00	1.00	1.00	1.00
		40	0	0.5	0.00	0.00	0.04	0.00	0.00

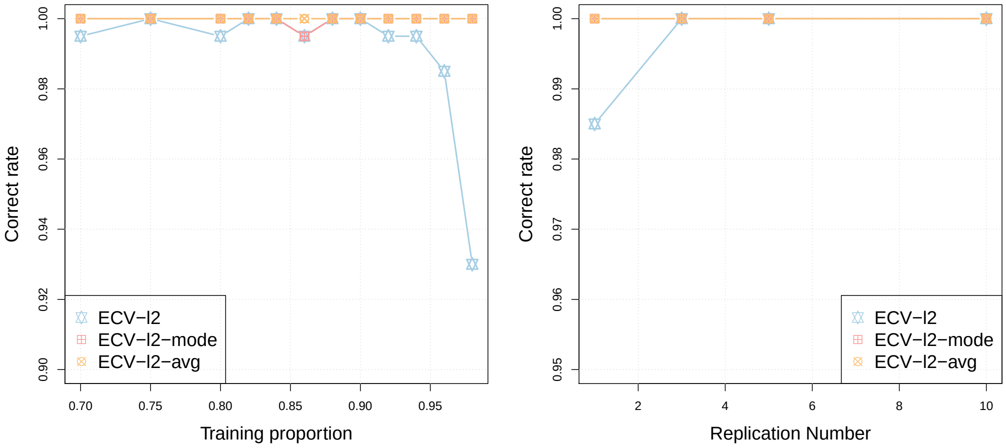
Table B.3: The correct rate for estimating the number of communities in 200 replications from the best variant of each method. The underlying true model is SBM.



(a) Varying p, $N = 3$.

(b) Varying N, $p = 0.9$.

Figure B.1: The rate of correctly selecting between the SBM and the DCSBM as a function of p and N.



(a) Varying p, $N = 3$.

(b) Varying N, $p = 0.9$.

Figure B.2: The rate of correctly selecting K under the true model as a function of p and N.

APPENDIX C

Appendix for Chapter IV

C.1 Proofs

Proof of Proposition IV.1. We only focus on identifying the parameter for **one arbitrary community** k. For any i such that $c_i = k$, we have

$$\mu_{il} = \log(\theta_i B_{kl}^{\lambda_i}) = \log(\theta_i) + \lambda_i \log(B_{kl}) \tag{C.1}$$

where we treat $\log(0)$ as $-\infty$. It can be seen that $\log(\theta_i) = \log(\tilde{P}_{ik})$ by setting $l = k$ and use the constraint $B_{kk} = 1$.

Write $b = (\log(B_{k1}), \cdots, \log(B_{k,K}))$. Notice that for any $1 \leq l \leq K$ such that $B_{kl} \neq 0$, we have

$$\mu_{ik} - \mu_{il} = \lambda_i(b_k - b_l). \tag{C.2}$$

When $B_{kl} \neq B_{kk}$, $b_k - b_l \neq 0$ thus we can uniquely determine the ratio between all nonzero λ_i's. Therefore, with the constraint $\sum_{i \in G_k} \lambda_k = n_k$, the identification of λ_i's is guaranteed.

Note that the constraint on λ_i also ensures that there exists at least one node i with nonzero λ_i. For such node i, we can see that b can be determined by (C.2) up to a shift. Since we constrain $b_k = B_{kk} = 1$, all the other entries of b are also identifiable.

The last condition is needed since if $\lambda_i < 0$ and $B_{c_i l} = 0$ for some $1 \le l \le K$, the model is not well-defined as $B_{c_i l}^{\lambda_i} = \infty$.

□

Proof of Proposition IV.2. It is easy to check that $\tilde{P} = FZ^T$ where F is the matrix obtained by applying the power function $f_i(x) = \theta_i x^{\lambda_i}$ to each element of the ith row in matrix ZB. Write $\Delta = \mathrm{diag}(\sqrt{n_1}, \cdots, \sqrt{n_K})$. Assume the SVD of $F\Delta$ is given by

$$F\Delta = UDV^T.$$

We have

$$\tilde{P} = UDV^T(Z\Delta^{-1})^T = UD(Z\Delta^{-1}V)^T.$$

Notice that $Z\Delta^{-1}$ is an orthonormal matrix and so is $Z\Delta^{-1}V$. Taking $X = \Delta^{-1}V$ gives the claimed result. □

We need the three following lemmas on spectral clustering.

Lemma C.1 (Lemma 7 of Chen and Lei [2017])**.** *Let $M, \widehat{M}$ be two matrices of size $n \times n$ and $V, \widehat{V}$ be the $n \times K$ orthogonal matrices of top K right singular vectors of M and $\widehat{M}$. Then there exists a $K \times K$ orthogonal matrix Q such that*

$$\|\widehat{V}Q - V\|_F \le \frac{2\sqrt{2K}\|\widehat{M} - M\|}{\sigma_K(M)}.$$

The orthogonal matrix Q is not material and will be ignored in the following discussion.

Lemma C.2 (Lemma 5.3 of Lei and Rinaldo [2014])**.** *Let $V, \widehat{V}$ be two $n \times K$ matrices with V having only K distinct rows, corresponding to K communities denoted by $\mathbf{c}$. Let $\hat{\mathbf{c}}$ be the output of a K-means clustering algorithm on $\widehat{V}$, with objective value no larger than $1 + \epsilon$ of the global optimum [Kumar et al., 2004]. Denote the community* *indices from $\boldsymbol{c}$ and $\hat{\boldsymbol{c}}$ by $\{G_k\}$ and $\{\hat{G}_k\}$. Define $S_k = \{i : i \in G_k, \hat{c}_i \neq k\}$. For any δ smaller than the minimum distance between any two distinct rows of V, if*

$$8(2+\epsilon)\|\widehat{V} - V\|_F^2 \leq n_{\min}\delta^2$$

where $n_{\min} = \min_k |G_k|$, then there exists a permutation of the K community labels in $\hat{\boldsymbol{c}}$, such that

$$\sum_{k=1}^{K} |S_k| \leq 8(2+\epsilon)\frac{\|\widehat{V} - V\|_F^2}{\delta^2}.$$

Another important result we need is the concentration of a random (directed) graph adjacency matrix from Le et al. [2017]. A similar argument is available from Lei and Rinaldo [2014].

Lemma C.3. *Let A be the adjacency matrix of a random graph on n nodes with independent edges. Set $\mathbb{E}(A) = P = [p_{ij}]_{n\times n}$ and assume that $n \max_{ij} p_{ij} \leq d$ for $d \geq C_0 \log n$ and $C_0 > 0$. Then there exists a constant C depending on C_0 such that*

$$\|A - P\| \leq C\sqrt{d}$$

with probability at least $1 - n^{-1}$.

Now with the three lemmas above, we can easily prove Theorem IV.10.

Proof of Theorem IV.10. Assume $\tilde{V}^*$ to be the matrix of right singular vectors for $\tilde{P}$ and assume $\tilde{V}$ is the right singular vectors of $\tilde{A}$. Notice that $\xi_{\max} = \max_{i,k,l} B_{kl}^{\lambda_i}$ is bounded so we pick an arbitrary constant upper bound $\xi_{\max}$ for it, which depends on B and η. The assumption $n\theta_{\max} \geq C_0 \log n$ implies the concentration requirement of Lemma C.3. From Lemma C.3, we have

$$\|\tilde{V} - \tilde{V}^*\|_F \leq \frac{2\sqrt{2K}}{T_n}\|\tilde{A} - \tilde{P}\| \leq \frac{2C\sqrt{2K}}{T_n}\sqrt{n\theta_{\max}\xi_{\max}}$$

with probability at least $1 - n^{-1}$.

To apply Lemma C.2, notice that from Proposition IV.2, the minimum distance between distinct rows in $\tilde{V}^*$ is at least $\sqrt{\frac{2}{n_{\max}}}$. Therefore, according to Lemma C.2,

$$\sum_k \frac{|S_k|}{n_k} \leq \frac{1}{n_{\min}} \sum_{k=1}^{K} |S_k| \leq \frac{1}{n_{\max}} 8(2+\epsilon) \frac{\|\tilde{V} - \tilde{V}^*\|_F^2}{\frac{2}{n_{\min}}}$$
$$\leq 32C^2(2+\epsilon)\xi_{\max} \frac{n_{\max} K n \theta_{\max}}{n_{\min} T_n^2} \leq \frac{32C^2(2+\epsilon)\xi_{\max}}{\kappa'} \frac{K n \theta_{\max}}{T_n^2}$$

as long as the condition of Lemma C.2 holds:

$$\frac{32C^2(2+\epsilon)\xi_{\max}}{\kappa'} \frac{K n \theta_{\max}}{T_n^2} \leq 1.$$

which can be guaranteed by the assumptions of Theorem IV.10 when setting $C_1 = \frac{32C^2(2+\epsilon)\xi_{\max}}{\kappa'}$. This completes the proof.

□

Proof of Corollary IV.12. Let f_1 and f_2 be the distribution of $\bar{\theta}_i$ and λ_i. Notice that with probability at least $1 - \exp(-\gamma_1 n)$, $\max_i \bar{\theta}_i = 1$ where γ_1 is a constant depending on f_1. Conditioning on this event to happen, we have $\theta_{\max} = \rho_n$ from A3. We now need a bound on T_n.

From Lemma IV.2, it can be seen that T_n is the Kth singular value of $\rho_n \cdot M$ where

$$M = \begin{pmatrix} \bar{\theta}_1 B^{\lambda_1}_{c_1,1} & \bar{\theta}_1 B^{\lambda_1}_{c_1,2} & \cdots & \bar{\theta}_1 B^{\lambda_1}_{c_1,K} \\ \bar{\theta}_2 B^{\lambda_2}_{c_2,1} & \bar{\theta}_2 B^{\lambda_2}_{c_2,2} & \cdots & \bar{\theta}_2 B^{\lambda_2}_{c_2,K} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\theta}_n B^{\lambda_n}_{c_n,1} & \bar{\theta}_n B^{\lambda_n}_{c_n,2} & \cdots & \bar{\theta}_n B^{\lambda_n}_{c_n,K} \end{pmatrix}.$$

and $\Delta = \operatorname{diag}(\sqrt{n_1}, \cdots, \sqrt{n_K})$. Under A3, there are only at most $m_1 m_2 K$ distinct rows of M. Denote the matrix with these $m_1 m_2 K$ rows as $\tilde{M} \in \mathbb{R}^{(m_1 m_2 K) \times K}$, then we can write

$$F = \rho_n \tilde{Z} \tilde{M}$$

where F is the same quantity in the proof of Proposition IV.2, $\tilde{Z} \in \mathbb{R}^{n\times(m_1m_2K)}$ with exactly one 1 in each row and zeros in the rest positions. $\tilde{Z}$ gives the correspondence between each row of M to the rows of $\tilde{M}$. Let $\tilde{n}_k$ be the number of times that the kth row of $\tilde{M}$ appears in rows in M, and define $\tilde{\Delta} = \text{diag}(\sqrt{\tilde{n}_1}, \cdots, \sqrt{\tilde{n}_{m_1m_2K}})$. It is easy to check $\tilde{Z}\tilde{\Delta}^{-1}$ is an orthogonal matrix. Therefore,

$$T_n = \sigma_K(\rho_n\tilde{\Delta}\tilde{M}\Delta) \geq \lambda\rho_n \min_{i,j,k} \sqrt{\tilde{n}_{ijk}} \min_k \sqrt{n_k}$$

in which $\lambda = \sigma_K(\tilde{M})$.

By IV.8-IV.11 and Hoeffding's inequality, we have

$$\min_{i,j,k} \tilde{n}_{ijk} \geq C_2 n$$

with probability at least $1 - \exp(-\gamma_2 n)$ for some constant $\gamma_2, C_2 > 0$ depending on κ', K and f_1, f_2. Under this event, we have

$$T_n \geq \sqrt{C_2\kappa'} n\rho_n.$$

Finally, applying Theorem IV.10 directly gives

$$\sum_k \frac{|S_k|}{n_k} \leq C_1 \frac{Kn\theta_{\max}}{T_n^2} \leq \frac{C_1}{C_2\kappa'}\frac{K}{n\rho_n}$$

with probability at least $1 - n^{-1} - e^{-\gamma_1 n} - e^{-\gamma_2 n} \geq 1 - 2n^{-1}$ for sufficiently large n. Setting $C' = \frac{C_1 K}{C_2\kappa'}$ gives the stated result.

□

Proof of Theorem IV.14. Without loss of generality, let us assume we are estimating the parameters in community 1 and that the first n_1 nodes are from community 1. Note that it is trivial to show the consistency for $B_{1l} = 0$. We now focus on the situation that $B_{1l} > 0$. We start from the fact that each A_{ij} is Bernoulli so it is trivially sub-Gaussian. For each $l \in [K]$, such that $B_{1l} > 0$, define

$$\tilde{P}_{il} = \theta_i B_{1l}^{\lambda_i}$$

We have

$$\mathbb{P}(|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})| > t) \leq 2\exp(-\tilde{c}n_l t^2) \leq 2\exp(-\tilde{c}\kappa' n t^2). \tag{C.3}$$

Setting $l = 1$ and $t = n^{-\frac{1}{3}}$ in (C.3) gives

$$\mathbb{P}(|\hat{\theta}_i - \theta_i| > n^{-\frac{1}{3}}) \leq 2\exp(-\tilde{c}\kappa' n^{1/3}).$$

Takes the union for all $i \in [n]$ gives

$$\mathbb{P}(\max_i |\hat{\theta}_i - \theta_i| > n^{-\frac{1}{3}}) \leq 2n\exp(-\tilde{c}\kappa' n^{1/3}) \leq 2\exp(-\frac{1}{2}\tilde{c}\kappa' n^{1/3})$$

for sufficiently large n. By noticing $\min_{\theta_i} \geq c_0 n^{-1/4}$ and writing $c = \tilde{c}\kappa'$, we get the bound in (4.12) thus finish the proof of Part 1.

Again from (C.3), we can see that

$$\mathbb{P}(|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})| > \frac{c_0}{2} n^{-1/4}) \leq 2\exp(-\frac{cc_0^2}{4} n^{1/2}). \tag{C.4}$$

Note that this also indicates that $\frac{\sum_{j \in G_l} A_{ij}}{n_l} > 0$ under the same event when n is sufficiently large.

Now we want to first show that for any $\rho \in (0, 1)$ and any i, l

$$\mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4}) \leq 4\exp(-\frac{cc_0^2}{4} n^{\rho/2}). \tag{C.5}$$

To see this, note that for any $\rho \in (0,1)$, we have

$$
\begin{aligned}
&\mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4})\\
&= \mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4}, \frac{\sum_{j \in G_l} A_{ij}}{n_l} \geq \frac{c_0}{2} n^{-1/4}) + \mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4}, \frac{\sum_{j \in G_l} A_{ij}}{n_l} < \frac{c_0}{2} n^{-1/4}\\
&\leq \mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4}, \frac{\sum_{j \in G_l} A_{ij}}{n_l} \geq \frac{c_0}{2} n^{-1/4}) + \mathbb{P}(\frac{\sum_{j \in G_l} A_{ij}}{n_l} < \frac{c_0}{2} n^{-1/4})\\
&\leq \mathbb{P}(|Y_{il} - \mu_{il}| > n^{-(1-\rho)/4}, \frac{\sum_{j \in G_l} A_{ij}}{n_l} \geq \frac{c_0}{2} n^{-1/4}) + \mathbb{P}(|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})| > \frac{c_0}{2} n^{-1/4})\\
&\leq \mathbb{P}(\frac{|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})|}{\frac{c_0}{2} n^{-1/4}} > n^{-(1-\rho)/4}, \frac{\sum_{j \in G_l} A_{ij}}{n_l} \geq \frac{c_0}{2} n^{-1/4}) + \mathbb{P}(|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})| > \frac{c_0}{2} n^{-1/4})\\
&\leq \mathbb{P}(\frac{|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})|}{\frac{c_0}{2} n^{-1/4}} > n^{-(1-\rho)/4}) + \mathbb{P}(|\frac{\sum_{j \in G_l} A_{ij}}{n_l} - \tilde{P}_{1l})| > \frac{c_0}{2} n^{-1/4})\\
&\leq 2\exp(-\frac{cc_0^2}{4} n^{\rho/2}) + 2\exp(-\frac{cc_0^2}{4} n^{1/2}) \leq 4\exp(-\frac{cc_0^2}{4} n^{\rho/2})
\end{aligned}
$$

in which the third last line comes from the fact that

$$|\log(x) - \log(y)| \leq \frac{1}{\min(x,y)} |x-y|, x, y > 0.$$

From (C.5), we then know that for a fixed $l \neq 1$

$$\mathbb{P}(|(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})| > 2n^{-(1-\rho)/4}) \leq 8\exp(-\frac{cc_0^2}{4} n^{\rho/2}) \tag{C.6}$$

and

$$
\begin{aligned}
&\mathbb{P}(|\frac{1}{n_1} \sum_{i:\boldsymbol{c}_i = 1} [(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})]| > 2n^{-(1-\rho)/4})\\
&= \mathbb{P}(|\frac{1}{n_1} \sum_{i:\boldsymbol{c}_i = 1} (Y_{i1} - Y_{il}) - (\log(B_{11}) - \log(B_{1l}))| > 2n^{-(1-\rho)/4})\\
&\leq 8n_1 \exp(-\frac{cc_0^2}{4} n^{\rho/2}) \leq 8\exp(-\frac{cc_0^2}{8} n^{\rho/2}) \qquad \text{(C.7)}
\end{aligned}
$$

for sufficiently large n. Part 2 of the theorem comes directly from (C.7) after taking the union of K^2 events.

For part 3, define $b_l = \log(B_{1l})$ for $B_{1l} > 0$. We discuss the estimation in two cases according to IV.13.

Case 1: If $b_l - b_1 \leq -\kappa$.

Applying Taylor's theorem to the function $F(x, y) = \frac{x}{y}, x, y > 0$, we have

$$|F(x,y) - F(x_0, y_0)| = |\nabla F(\tilde{x}, \tilde{y}) \cdot (x - x_0, y - y_0)| \leq \|\nabla F(\tilde{x}, \tilde{y})\| \|(x - x_0, y - y_0)\|$$

in which $\tilde{x}$ lies between x and x_0 and $\tilde{y}$ lies between y and y_0. Notice that

$$\nabla F(x,y) = \begin{bmatrix} \frac{1}{y} \\ \\ -\frac{x}{y^2} \end{bmatrix}$$

So we further have

$$\text{(C.8)} \quad (F(x,y) - F(x_0, y_0)) \leq \left(\frac{1}{\min(y, y_0)^2} + \frac{\max(x, x_0)^2}{\min(y, y_0)^4} \right) \|(x - x_0, y - y_0)\|^2.$$

Assume n is sufficiently large such that $2n^{-\frac{1-\rho}{n}} < \min(\frac{\kappa}{2}, \frac{\kappa}{2\eta}) \leq \frac{b_1 - b_l}{2}$. Define the event

$$E := \{|(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})| \leq 2n^{-(1-\rho)/4}, |\frac{1}{n_1} \sum_{i:\boldsymbol{c}_i = 1} [(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})]|$$

$$\leq 2n^{-(1-\rho)/4}, Y_{i1} - Y_{il} > 0, \sum_{i:\boldsymbol{c}_i = 1} (Y_{i1} - Y_{il}) > 0\}$$

$$= \{|(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})| \leq 2n^{-(1-\rho)/4}, |\frac{1}{n_1} \sum_{i:\boldsymbol{c}_i = 1} [(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})]| \leq 2n^{-(1-\rho)/4}\}.$$

Under the event E, we have

$$|\hat{\lambda}_i - \lambda_i|^2$$

$$= |\frac{(Y_{i1} - Y_{il})}{\sum_{i:\boldsymbol{c}_i=1} (Y_{i1} - Y_{il})/n_1} - \frac{(\mu_{i1} - \mu_{il})}{\sum_{i:\boldsymbol{c}_i=1} (\mu_{i1} - \mu_{il})/n_1}|^2$$

$$\leq \left(|(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})|^2 + |\frac{1}{n_1} \sum_{i:\boldsymbol{c}_i = 1} [(Y_{i1} - Y_{il}) - (\mu_{i1} - \mu_{il})]|^2 \right) \times$$

$$\left(\frac{1}{\min(\sum_{i:\boldsymbol{c}_i=1} (Y_{i1} - Y_{il})/n_1, \sum_{i:\boldsymbol{c}_i=1} (\mu_{i1} - \mu_{il})/n_1)^2} + \frac{\max(Y_{i1} - Y_{il}, \mu_{i1} - \mu_{il})^2}{\min(\sum_{i:\boldsymbol{c}_i=1} (Y_{i1} - Y_{il})/n_1, \sum_{i:\boldsymbol{c}_i=1} (\mu_{i1} - \mu_{il})} \right.$$

$$\leq \left(\frac{1}{((b_1 - b_l) - 2n^{-(1-\rho)/4})^2} + \frac{(\lambda_i (b_1 - b_l) + 2n^{-(1-\rho)/4})^2}{((b_1 - b_l) - 2n^{-(1-\rho)/4})^4} \right) 8n^{-\frac{1-\rho}{2}}$$

$$\leq \left(\frac{4}{(b_1 - b_l)^2} + \frac{16(\lambda_i + 3/2)^2}{(b_1 - b_l)^2} \right) 8n^{-\frac{1-\rho}{2}}$$

$$\leq \frac{32}{\kappa^2} \left(1 + 4(\eta + 3/2)^2\right) n^{-\frac{1-\rho}{2}}.$$

Finally, according to (C.6) and (C.7), the event E happens with probability at least

$$1 - 8\exp(-\frac{cc_0^2}{4}n^{\rho/2}) - 8\exp(-\frac{cc_0^2}{8}n^{\rho/2}) \geq 1 - 16\exp(-\frac{cc_0^2}{8}n^{\rho/2}).$$

Case 2: If $b_l - b_1 \geq \kappa$. This is the symmetric version of the previous case. The only thing we need to change is to use the Taylor's expansion of $F(x, y) = \frac{x}{y}$ for $x, y < 0$ which still gives the same bound.

Combining Case 1 and 2 shows the error bound for one λ_i. Applying the same procedure for any i in any community then taking the union of the events, we know that

$$\max_i |\hat{\lambda}_i - \lambda_i| \leq \frac{4\sqrt{2}}{\kappa}\sqrt{1 + 4(\eta + 3/2)^2}n^{-(1-\rho)/4}$$

with probability at least $1 - 16n\exp(-\frac{cc_0^2}{8}n^{\rho/2})$. For sufficiently large n, it can be shown that this probability is larger or equal to

$$1 - 16\exp(-\frac{cc_0^2}{10}n^{\rho/2}).$$

This completes the proof for part 3.

□

C.2 Community detection of business schools on the undirected hiring network

In Section 4.6, we show the community analysis by using the directed hiring network between business schools. In this section, we list the community detection result by spectral clustering if we treat the network as undirected. The four communities are shown in Table C.1 with average and median ranking by US News and π-ranking as well the 20 institutions with the highest π-ranking in each community. It can be seen that the overall tie of the 3 communities are not so clear according to the reference ranking systems of US News and π-ranking. The first community is still overall better but the exclusion of institutions such as Yale, Cornell and Columbia with inclusion of some other state universities make it much less interpretable compared with the results in Table 4.1 of Section 4.6. This indicates the importance of using the correct spectral information, since making the network symmetric hides the interpretable community structures.

	size	USN (avg./med.)	π-ranking (avg./med.)	Institutions
1	19	19.2/14	17.8/13	Stanford, MIT, Harvard, UC Berkeley, U. Rochester, U. Chicago, Northwestern, U. Michigan, U. Penn., Carnegie Mellon, NYU, U. Minnesota-Twin Cities, Duke, UNC-Chapel Hill, U. Washington St. Louis, U. Maryland, College Park, U. Southern California, Case Western Reserve U., Boston College
2	20	55.1/56.5	44.6/42	Cornell, Columbia, U. Wisconsin-Madison, UIUC, Ohio State, U. Florida, U. Pittsburgh, Penn State, Michigan State, SUNY-Buffalo, U. Mass-Amherst, Syracuse, Tulane, U. Connecticut, U. Cincinnati, Rutgers U., Temple U., SUNY-Binghamton, St. Louis U., Northeastern U.
3	24	52.7/40	54/49	Yale, UCLA, U. Washington, U. Colorado-Boulder, UC Irvine, U. Utah, U. Oregon, UT-Dallas, U. Virginia, Boston U., UC Davis, Vanderbilt, Claremont Graduate U., U. Houston, Rice U., Southern Methodist U., George Washington U., CUNY Baruch College, U. Hawaii
4	24	63.8/63	56/56.5	Purdue, U. Iowa, UT-Austin, Indiana U., Georgia Tech, U. Arizona, Texas A&M, U. Georgia, Arizona State, U. South Carolina, Virginia Tech, Florida State, U. Oklahoma, U. Kansas, Louisiana State, U. Arkansas, U. Tennesse, U. Kentucky, U. Alabama, Oklahoma State

Table C.1: Communities of business school institutions detected by symmetric spectral clustering.

Bibliography

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

E. Abbe, J. Fan, K. Wang, and Y. Zhong. Entrywise eigenvector analysis of random matrices with low expected rank. *arXiv preprint arXiv:1709.09565*, 2017.

E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008.

D. J. Aldous. Representations for partially exchangeable arrays of random variables. *Journal of Multivariate Analysis*, 11(4):581–598, 1981.

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

S. Asur and B. A. Huberman. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE, 2010.

A. Athreya, D. E. Fishkind, K. Levin, V. Lyzinski, Y. Park, Y. Qin, D. L. Sussman, M. Tang, J. T. Vogelstein, and C. E. Priebe. Statistical inference on random dot product graphs: a survey. *arXiv preprint arXiv:1709.05454*, 2017.

A. S. Bandeira, R. van Handel, et al. Sharp nonasymptotic bounds on the norm of random matrices with independent entries. *The Annals of Probability*, 44(4): 2479–2506, 2016.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7(Nov):2399–2434, 2006.

Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. *Advances in Neural Information Processing Systems*, 16:177–184, 2004.

J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 192–236, 1974.

S. A. Bhaskar and A. Javanmard. 1-bit matrix completion under exact low-rank constraint. In *Information Sciences and Systems (CISS), 2015 49th Annual Conference on*, pages 1–6. IEEE, 2015.

S. Bhojanapalli and P. Jain. Universal matrix completion. In *Proceedings of The 31st International Conference on Machine Learning*, pages 1881–1889, 2014.

P. Bickel, D. Choi, X. Chang, H. Zhang, et al. Asymptotic normality of maximum likelihood and its variational approximation for stochastic blockmodels. *The Annals of Statistics*, 41(4):1922–1943, 2013.

P. J. Bickel and P. Sarkar. Hypothesis testing for automated community detection in networks. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(1):253–273, 2016.

G. Bosilca, A. Bouteiller, A. Danalis, T. Herault, P. Lemarinier, and J. Dongarra. Dague: A generic distributed dag engine for high performance computing. *Parallel Computing*, 38(1):37–51, 2012.

S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge University Press, 2004.

Y. Bramoullé, H. Djebbari, and B. Fortin. Identification of peer effects through social networks. *Journal of Econometrics*, 150(1):41–55, 2009.

C. T. Butts. Network inference, error, and informant (in) accuracy: a bayesian approach. *social networks*, 25(2):103–140, 2003.

D. Cai, X. He, and J. Han. Spectral regression: A unified approach for sparse subspace learning. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 73–82. IEEE, 2007.

T. Cai and W.-X. Zhou. A max-norm constrained minimization approach to 1-bit matrix completion. *The Journal of Machine Learning Research*, 14(1):3619–3647, 2013.

E. J. Candes and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

S. Chatterjee. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2015.

K. Chaudhuri, F. C. Graham, and A. Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *COLT*, volume 23, pages 35–1, 2012.

K. Chen and J. Lei. Network cross-validation for determining the number of communities in network data. *Journal of the American Statistical Association*, pages 1–11, 2017.

P. Chin, A. Rao, and V. Vu. Stochastic block model and community detection in sparse graphs: A spectral algorithm with optimal rate of recovery. In *Conference on Learning Theory*, pages 391–423, 2015.

D. Choi. Estimation of monotone treatment effects in network experiments. *Journal of the American Statistical Association*, pages 1–9, 2017.

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

N. A. Christakis and J. H. Fowler. The spread of obesity in a large social network over 32 years. *New England Journal of Medicine*, 357(4):370–379, 2007.

A. Clauset, S. Arbesman, and D. B. Larremore. Systematic inequality and hierarchy in faculty hiring networks. *Science advances*, 1(1):e1400005, 2015.

M. B. Cohen, R. Kyng, G. L. Miller, J. W. Pachocki, R. Peng, A. B. Rao, and S. C. Xu. Solving sdd linear systems in nearly m log 1/2 n time. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 343–352. ACM, 2014.

D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

H. Crane and W. Dempsey. A framework for statistical network modeling. *arXiv preprint arXiv:1509.08185*, 2015.

N. Cressie. The origins of kriging. *Mathematical geology*, 22(3):239–252, 1990.

M. A. Davenport, Y. Plan, E. van den Berg, and M. Wootters. 1-bit matrix completion. *Information and Inference*, 3(3):189–223, 2014.

P. Diaconis and S. Janson. Graph limits and exchangeable random graphs. *arXiv preprint arXiv:0712.2749*, 2007.

T. Edwards. The discrete laplacian of a rectangular grid, 2013.

J. Eldridge, M. Belkin, and Y. Wang. Unperturbed: spectral analysis beyond davis-kahan. *arXiv preprint arXiv:1706.06516*, 2017.

P. Erds and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5:17–61, 1960.

M. Faverge and P. Ramet. Dynamic scheduling for sparse direct solver on numa architectures. In *PARA'08*, 2008.

S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

K. Fujimoto and T. W. Valente. Social network influences on adolescent substance use: disentangling structural equivalence from cohesion. *Social Science & Medicine*, 74(12):1952–1960, 2012.

C. Gao, Y. Lu, and H. H. Zhou. Rate-optimal graphon estimation. *The Annals of Statistics*, 43(6):2624–2652, 2015.

C. Gao, Y. Lu, Z. Ma, and H. H. Zhou. Optimal estimation and completion of matrices with biclustering structures. *Journal of Machine Learning Research*, 17 (161):1–29, 2016.

C. Gao, Z. Ma, A. Y. Zhang, and H. H. Zhou. Achieving optimal misclassification proportion in stochastic block models. *The Journal of Machine Learning Research*, 18(1):1980–2024, 2017.

A. Goldenberg, A. X. Zheng, S. E. Fienberg, and E. M. Airoldi. A survey of statistical network models. *Foundations and Trends® in Machine Learning*, 2(2):129–233, 2010.

R. Guimerà and M. Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52):22073–22078, 2009.

D. Hallac, J. Leskovec, and S. Boyd. Network lasso: Clustering and optimization in large graphs. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 387–396. ACM, 2015.

K. M. Harris. *The National Longitudinal Study of Adolescent to Adult Health (Add Health), Waves I & II, 1994-1996; Wave III, 2001-2002; Wave IV, 2007–009 [machine-readable data file and documentation].* Carolina Population Center, University of North Carolina at Chapel Hill, 2009.

T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

D. L. Haynie. Delinquent peers revisited: Does network structure matter? *American Journal of Sociology*, 106(4):1013–1057, 2001.

C. R. Henderson. Estimation of variance and covariance components. *Biometrics*, 9 (2):226–252, 1953.

P. Hoff. Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, pages 657–664, 2008.

P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

P. Ji, J. Jin, et al. Coauthorship and citation networks for statisticians. *The Annals of Applied Statistics*, 10(4):1779–1812, 2016.

J. Jin. Fast community detection by SCORE. *The Annals of Statistics*, 43(1):57–89, 2015.

A. Joseph and B. Yu. Impact of regularization on spectral clustering. *The Annals of Statistics*, 44(4):1765–1791, 2016.

B. Karrer and M. E. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107, 2011.

R. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. In *Advances in Neural Information Processing Systems*, pages 952–960, 2009.

S. Kim, W. Pan, and X. Shen. Network-based penalized regression with application to genomic data. *Biometrics*, 69(3):582–593, 2013.

O. Klopp. Matrix completion by singular value thresholding: sharp bounds. *Electronic Journal of Statistics*, 9(2):2348–2369, 2015.

E. D. Kolaczyk. *Statistical Analysis of Network Data: Methods and Models*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 038788145X, 9780387881454.

I. Koutis, G. L. Miller, and R. Peng. Approaching optimality for solving sdd linear systems. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 235–244. IEEE, 2010.

A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time (1+ ε)-approximation algorithm for geometric k-means clustering in any dimensions. In *Proceedings-Annual Symposium on Foundations of Computer Science*, pages 454–462. IEEE, 2004.

X. Lacoste, M. Faverge, P. Ramet, S. Thibault, and G. Bosilca. Taking advantage of hybrid systems for sparse direct solvers via task-based runtimes. In *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*, pages 29–38. IEEE, 2014.

S. R. Land and J. H. Friedman. Variable fusion: A new adaptive signal regression method. Technical report, Technical Report 656, Department of Statistics, Carnegie Mellon University Pittsburgh, 1997.

P. Latouche, E. Birmele, and C. Ambroise. Variational bayesian inference and complexity control for stochastic block models. *Statistical Modelling*, 12(1):93–115, 2012.

C. M. Le and E. Levina. Estimating the number of communities in networks by spectral methods. *arXiv preprint arXiv:1507.00827*, 2015.

C. M. Le and E. Levina. Estimating a network from multiple noisy realizations. *arXiv preprint arXiv:1710.04765*, 2017.

C. M. Le, E. Levina, and R. Vershynin. Concentration and regularization of random graphs. *Random Structures & Algorithms*, 2017.

D. Lee. CARBayes: An R package for Bayesian spatial modeling with conditional autoregressive priors. *Journal of Statistical Software*, 55(13):1–24, 2013. URL `http://www.jstatsoft.org/v55/i13/`.

L.-f. Lee. Identification and estimation of econometric models with group interactions, contextual factors and fixed effects. *Journal of Econometrics*, 140(2): 333–374, 2007.

J. Lei. A goodness-of-fit test for stochastic block models. *The Annals of Statistics*, 44(1):401–424, 2016.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237, 2014.

C. Li and H. Li. Network-constrained regularization and variable selection for analysis of genomic data. *Bioinformatics*, 24(9):1175–1182, 2008.

C. Li and H. Li. Variable selection and regression analysis for graph-structured covariates with an application to genomics. *The Annals of Applied Statistics*, 4(3): 1498, 2010.

T. Li, E. Levina, and J. Zhu. *netcoh: Statistical Modeling with Network Cohesion*, 2016a. URL `http://CRAN.R-project.org/package=netcoh`. R package version 0.11.

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. *arXiv preprint arXiv:1612.04717*, 2016b.

T. Li, E. Levina, and J. Zhu. Prediction models for network-linked data. *arXiv preprint arXiv:1602.01192*, 2016c.

X. Lin. Identifying peer effects in student academic achievement by spatial autoregressive models with group unobservables. *Journal of Labor Economics*, 28(4): 825–860, 2010.

R. J. Lipton, D. J. Rose, and R. E. Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16(2):346–358, 1979.

C. F. Manski. Identification of endogenous social effects: The reflection problem. *The Review of Economic Studies*, 60(3):531–542, 1993.

C. F. Manski. Identification of treatment response with social interactions. *The Econometrics Journal*, 16(1):S1–S23, 2013.

T. Martin, B. Ball, and M. E. Newman. Structural inference for uncertain networks. *Physical Review E*, 93(1):012306, 2016.

C. Matias and V. Miele. Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(4):1119–1141, 2017.

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322, 2010.

A. F. McDaid, T. B. Murphy, N. Friel, and N. J. Hurley. Improved Bayesian inference for the stochastic block model with application to large networks. *Computational Statistics & Data Analysis*, 60:12–31, 2013.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

L. Michell and P. West. Peer pressure to smoke: the meaning depends on the method. *Health Education Research*, 11(1):39–49, 1996.

J. A. Nelder and R. J. Baker. *Generalized linear models*. Wiley Online Library, 1972.

M. Newman. *Networks: an introduction*. Oxford university press, 2010.

M. Newman. Network structure from rich but noisy data. *Nature Physics*, page 1, 2018a.

M. Newman. Network reconstruction and error estimation with noisy network data. *arXiv preprint arXiv:1803.02427*, 2018b.

M. E. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

M. E. Newman and A. Clauset. Structure and inference in annotated networks. *Nature Communications*, 7, 2016.

W. Pan, B. Xie, and X. Shen. Incorporating predictor network in penalized regression with application to microarray data. *Biometrics*, 66(2):474–484, 2010.

M. Pearson and L. Michell. Smoke rings: social network analysis of friendship groups, smoking and drug-taking. *Drugs: Education, Prevention, and Policy*, 7(1):21–37, 2000.

M. Pearson and P. West. Drifting smoke rings. *Connections*, 25(2):59–76, 2003.

T. Q. Phan and E. M. Airoldi. A natural experiment of social network formation and dynamics. *Proceedings of the National Academy of Sciences*, 112(21):6595–6600, 2015.

T. Qin and K. Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems*, pages 3120–3128, 2013.

B. Raducanu and F. Dornaika. A supervised non-linear dimensionality reduction approach for manifold learning. *Pattern Recognition*, 45(6):2432–2444, 2012.

D. G. Rand, S. Arbesman, and N. A. Christakis. Dynamic social networks promote cooperation in experiments with humans. *Proceedings of the National Academy of Sciences*, 108(48):19193–19198, 2011.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, pages 1878–1915, 2011.

K. Rohe, T. Qin, and B. Yu. Co-clustering directed graphs to discover asymmetries and directional communities. *Proceedings of the National Academy of Sciences*, 113(45):12679–12684, 2016.

G. Rossetti and R. Cazabet. Community discovery in dynamic networks: a survey. *arXiv preprint arXiv:1707.03186*, 2017.

H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

V. Sadhanala, Y.-X. Wang, and R. J. Tibshirani. Graph sparsification approaches for laplacian smoothing. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1250–1259, 2016.

D. F. Saldana, Y. Yu, and Y. Feng. How many communities are there? *ArXiv e-prints*, Dec. 2014.

P. Sarkar and P. J. Bickel. Role of normalization in spectral clustering for stochastic blockmodels. *Ann. Statist.*, 43(3):962–990, 06 2015. doi: 10.1214/14-AOS1285. URL `http://dx.doi.org/10.1214/14-AOS1285`.

S. R. Searle, G. Casella, and C. E. McCulloch. *Variance Components*, volume 391. John Wiley & Sons, 2009.

C. R. Shalizi and A. C. Thomas. Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods and Research*, 40(2): 211–239, 2011.

J. Sharpnack, A. Singh, and A. Krishnamurthy. Detecting activations over graphs using spanning tree wavelet bases. In *Artificial Intelligence and Statistics*, pages 536–544, 2013.

J. Shi and J. Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

X. Song and X.-H. Zhou. A semiparametric approach for the covariate specific roc curve with survival outcome. *Statistica Sinica*, pages 947–965, 2008.

D. A. Spielman. Algorithms, graph theory, and linear equations in laplacian matrices. In *Proceedings of the International Congress of Mathematicians*, volume 4, pages 2698–2722, 2010.

D. A. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.

N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. In *International Conference on Computational Learning Theory*, pages 545–560. Springer, 2005.

L. Su, W. Wang, and Y. Zhang. Strong consistency of spectral clustering for stochastic block models. *arXiv preprint arXiv:1710.06191*, 2017.

C. A. Sugar and G. M. James. Finding the number of clusters in a dataset. *Journal of the American Statistical Association*, 98(463), 2003.

D. L. Sussman, M. Tang, and C. E. Priebe. Consistent latent position estimation and vertex classification for random dot product graphs. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):48–57, 2014.

M. Tang and C. E. Priebe. Limit theorems for eigenvectors of the normalized laplacian for random graphs. *arXiv preprint arXiv:1607.08601*, 2016.

M. Tang, A. Athreya, D. L. Sussman, V. Lyzinski, C. E. Priebe, et al. A nonparametric two-sample hypothesis testing problem for random graphs. *Bernoulli*, 23 (3):1599–1630, 2017.

J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

R. Tibshirani and G. Walther. Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics*, 14(3):511–528, 2005.

R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Science & Business Media, 2013.

J. T. Vogelstein, W. G. Roncal, R. J. Vogelstein, and C. E. Priebe. Graph classification using signal-subgraphs: Applications in statistical connectomics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(7):1539–1551, 2013.

E. Vural and C. Guillemot. Out-of-sample generalizations for supervised manifold learning for classification. *IEEE Transactions on Image Processing*, 25(3):1410–1424, 2016.

G. Wahba et al. Support vector machines, reproducing kernel hilbert spaces and the randomized gacv. *Advances in Kernel Methods-Support Vector Learning*, 6:69–87, 1999.

L. A. Waller and C. A. Gotway. *Applied spatial statistics for public health data*, volume 368. John Wiley & Sons, 2004.

S. Wang, K. Rohe, et al. Discussion of "coauthorship and citation networks for statisticians". *The Annals of Applied Statistics*, 10(4):1820–1826, 2016a.

Y. R. Wang, P. J. Bickel, et al. Likelihood-based model selection for stochastic block models. *The Annals of Statistics*, 45(2):500–528, 2017.

Y.-X. Wang, J. Sharpnack, A. Smola, and R. J. Tibshirani. Trend filtering on graphs. *Journal of Machine Learning Research*, 17(105):1–41, 2016b.

T. Wolf, A. Schroter, D. Damian, and T. Nguyen. Predicting build failures using social network analysis on developer communication. In *Proceedings of the 31st*

International Conference on Software Engineering, pages 1–11. IEEE Computer Society, 2009.

P. J. Wolfe and S. C. Olhede. Nonparametric graphon estimation. *arXiv preprint arXiv:1309.5936*, 2013.

K. S. Xu and A. O. Hero. Dynamic stochastic blockmodels: Statistical models for time-evolving networks. In *International conference on social computing, behavioral-cultural modeling, and prediction*, pages 201–210. Springer, 2013.

Y. Xu, J. S. Dyer, and A. B. Owen. Empirical stationary correlations for semi-supervised learning on graphs. *The Annals of Applied Statistics*, pages 589–614, 2010.

W. Yang, C. Sun, and L. Zhang. A multi-manifold discriminant analysis method for image feature extraction. *Pattern Recognition*, 44(8):1649–1657, 2011.

Y. Yao. Information-theoretic measures for knowledge discovery and data mining. In *Entropy Measures, Maximum Entropy Principle and Emerging Applications*, pages 115–136. Springer, 2003.

S. J. Young and E. R. Scheinerman. Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 138–149. Springer, 2007.

X. Zhang, C. Moore, and M. E. Newman. Random graph models for dynamic networks. *The European Physical Journal B*, 90(10):200, 2017.

Y. Zhang, E. Levina, and J. Zhu. Estimating network edge probabilities by neighborhood smoothing. *Biometrika (In press)*, 2015.

Y. Zhao, E. Levina, and J. Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, pages 2266–2292, 2012.

Y. Zhao, Y.-J. Wu, E. Levina, and J. Zhu. Link prediction for partially observed networks. *Journal of Computational and Graphical Statistics*, 26(3):725–733, 2017.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328, 2004.

D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd international conference on Machine learning*, pages 1036–1043. ACM, 2005.