Community Detection in the Setting of Generalized Random Dot Product Graphs

John Koo

Submitted to the faculty of the Univesity Graduate School
in partial fulfillment of the requirements for the degree
Doctor of Philosophy
in the Department of Statistics,

Indiana University
December 2022

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Approved:

Michael W. Trosset, Ph.D.

Minh Tang, Ph.D.

Julia Fukuyama, Ph.D.

Roni Khardon, Ph.D.

Fangzheng Xie, Ph.D.

December 1, 2022

Acknowledgements

Abstract

Graph and network data, in which samples are represented not as a collection of feature vectors but as relationships between pairs of observations, are increasingly widespread in various fields ranging from analyzing data in the social sciences to training machine learning models for artificial intelligence tasks. One common goal of analyzing graph data is community detection or graph clustering, in which the graph is partitioned into disconnected subgraphs in an unsupervised yet meaningful manner (e.g., by optimizing an objective function or recovering unobserved labels). Because traditional clustering techniques were developed for data that can be represented as vectors, they cannot be applied directly to graphs. In this research, we investigate the use of a family of spectral decomposition based approaches for community detection in block models (random graph models with inherent community structure), first by demonstrating how under the generalized random dot product graph framework, all block models can be represented as a collection of feature vectors organized by community, applying clustering methods for these feature vector representations by exploiting the linear structures that the block models induce, and finally deriving the asymptotic properties of these methods. We further extend this connection between block models and community-organized generalized random dot product graphs to propose more flexible, nonlinear community structures, using real graphs with nonlinear structures as motivating examples.

Contents

1 Introduction

1.1 Graphs and Representations of Network Data

Graph and network data have become increasingly widespread in various fields including sociology, neuroscience, biostatistics, and computer science. This has resulted in challenges for researchers who rely on traditional statistical and machine learning methods that are incompatible with graph data and instead assume that the data exist as feature vectors. To illustrate this, consider the typical approach to building a statistical or machine learning model. Data are often represented as an $n \times p$ matrix $X = \begin{bmatrix} x_1 \mid \cdots \mid x_n \end{bmatrix}^\top$ in which each row $x_i \in \mathbb{R}^p$ is an observation of p features and each column is a set of n feature measurements. An analysis task for these data might be to come up with a classification model $\phi : \mathbb{R}^p \to \{1, 2, ..., K\}$ that uses the numerical values of each feature of a vector x_i to calculate a predicted label $z_i \in \{1, 2, ..., K\}$. For instance, $\phi(x)$ might first compute the distances from x to K points in $\mathbb{R}^p$ and then assign x to the label of the nearest point. Examples of this include linear discriminant analysis (in the case of supervised learning) and Lloyd's algorithm (Lloyd, 1982) or Gaussian mixture models (Fraley and Raftery, 2002) (in the case of unsupervised learning). However, it is not obvious how this method would translate to data that are represented as graphs, in which observations consist of relationships among a set of objects rather than numerical attributes associated with each object: Instead of feature vector $x_i = \begin{bmatrix} x_{i1} \mid \cdots \mid x_{ip} \end{bmatrix}^\top \in \mathbb{R}^p$, we observe $a_i = \begin{bmatrix} a_{i1} \mid \cdots \mid a_{in} \end{bmatrix}^\top \in \mathbb{R}^n$ in which each a_{ij} is object i's relationship to object j. For such data, it is often necessary to alter existing algorithms, transform the graph data into Euclidean data, often called *graph embedding* or *spectral clustering* (von Luxburg, 2007), and apply algorithms for Euclidean data on the embedding, or come up with new algorithms altogether.

Example 1.1 (Graph clustering)**.** K-means clustering (MacQueen, 1967) is a nonparametric

clustering method that minimizes the objective function

$$\underset{z_1,...,z_n}{\arg\min}\ W(z_1,...,z_n;x_1,...,x_n) = \underset{z_1,...,z_n}{\arg\min}\ \sum_{k=1}^{K}\sum_{x_i:z_i=k}\|x_i-\bar{x}_k\|_2^2, \tag{1}$$

in which $x_1,...,x_n \in \mathbb{R}^p$ are feature vectors of a sample, $z_1,...,z_n \in \{1,...,K\}$ are cluster assignments which we wish to optimize, and $\bar{x}_1,...,\bar{x}_K \in \mathbb{R}^p$ are the centroids of each cluster, which act as nuisance parameters. A popular implementation of K-means clustering is Lloyd's algorithm (Lloyd, 1982), which is a type of coordinate descent algorithm in which at each iteration the cluster labels are updated according to the label of its nearest centroid and centroids are updated as the sample means of the points in each cluster.

If we have a graph instead of vectors in $\mathbb{R}^p$, it is not obvious how to translate K-means clustering to these data. In particular, there is no inherent notion of centroid or sample mean for graph data. However, there are inherent notions of distances among vertices, such as shortest path distance, expected commute time, or resistance distance. One way to adapt K-means to graph data is to use an equivalent objective function as equation (1) that doesn't use centroids (which are just nuisance parameters) and is stated only in terms of distances:

$$\underset{z_1,...,z_n}{\arg\min}\ \tilde{W}(z_1,...,z_n;x_1,...,x_n) = \underset{z_1,...,z_n}{\arg\min}\ \sum_{k=1}^{K}\sum_{x_i,x_j:z_i=z_j=k} d^2(x_i,x_j).$$

Here, $d^2(x_i,x_j)$ is the squared distance between objects x_i and x_j. If they are vectors in Euclidean space, then we can use $d^2(x_i,x_j) = \|x_i-x_j\|_2^2$, or if they are vertices on a graph, then $d^2(x_i,x_j)$ represents some notion of graph distance. While this formulation of K-means clustering takes care of the lack of centroids, the implementation still requires some thought since Lloyd's algorithm includes the computation of centroids. An alternative algorithm for K-means clustering in this setting is MacQueen's exchange algorithm (MacQueen, 1967), which involves cycling through each vertex and then, for each vertex, cycling through the labels and choosing the label that minimizes the objective function.

Yet another approach to adapting K-means clustering to graph data is to embed the graph to Euclidean space and then apply Lloyd's algorithm on the embedding. In this context, a natural choice of embedding is one in which pairwise Euclidean distances between embedding vectors are (approximately) equal to pairwise graph distances between corresponding vertices. For example, the combinatorial Laplacian eigenmap is an embedding in which Euclidean distances between pairs of vectors in the embedding are equal to expected commute times between corresponding pairs of vertices in the graph (von Luxburg, 2007). This embedding is constructed by computing the combinatorial Laplacian matrix $L = D - A$ in which D is a diagonal matrix in which each D_{ii} is the degree of vertex i. Then the components of the embedding consist of $\left[v_{n-1}/\sqrt{\lambda_{n-1}} \mid \cdots \mid v_1/\sqrt{\lambda_1}\right]^\top$ in which λ_i is the i^{th} largest eigenvalue of L and v_i is its corresponding eigenvector ($\lambda_n = 0$ and is omitted). Then Lloyd's algorithm can be applied to this embedding. More often in practice, the d smallest eigenvalues are used to construct a lower dimensional embedding.

We now provide a more formal description of graph data: Suppose we observe a network of n objects and pairwise relationships between them. This network is represented by a graph $G = (V, E)$ with vertex set $V = \{v_1, ..., v_n\}$, representing the n objects, and edge set E, representing the up to $n(n-1)/2$ pairwise relationships (assuming that there are no self-loops). The numeric representation of these data is in the form of *affinity matrix* $A \in \mathbb{R}_+^{n \times n}$ in which each A_{ij} represents object i's relationship to object j. We assume that the entries of A represent affinities or similarities, i.e., the higher the value of A_{ij}, the stronger the relationship i has to j. If $A_{ij} = 0$, then i has no direct relationship to j. A is symmetric if it represents an undirected graph in which the relationship from i to j is the same as the relationship from j to i. A is binary, i.e., $A \in \{0, 1\}^{n \times n}$, if it represents an unweighted graph in which edges either exist or don't exist. If A is binary, we call it an *adjacency matrix*. In this work, we focus primarily on undirected and unweighted graphs without self loops. Oftentimes, graph data are paired with vertex attributes. For instance, in a social network graph, in which vertices correspond to individuals and edges correspond to

connections between pairs of individuals, each vertex (individual) might also have a set or vector of other observations (e.g., age, sex, location, etc.). This is often the setting for inference in the *exponential family of random graphs* (ERGM) (Kolaczyk and Csárdi, 2014). Another setting in which vertices have attributes is when the (oftentimes single) attribute is hidden or unobserved, and the inference task is to identify the attributes. If those hidden attributes are discrete labels, then the inference task is called *community detection*, which is the main interest in this work.

Example 1.2. Nepusz et al. (2008) constructed a friendship network among 81 faculty from various schools at a university in the UK (available in the `igraphdata` R package (Csardi and Nepusz, 2006)). These data are represented as a graph in which each vertex is a faculty member and edges between pairs of vertices indicate whether the two faculty are friends. The corresponding adjacency matrix $A \in \{0,1\}^{81\times 81}$ has zeros along the diagonal since there are no self-loops, and $A_{ij} = A_{ji} = 1$ if it is observed that the i^{th} and j^{th} faculty members are friends. The following is a visualization of this graph, with the vertices labeled by school affiliation.

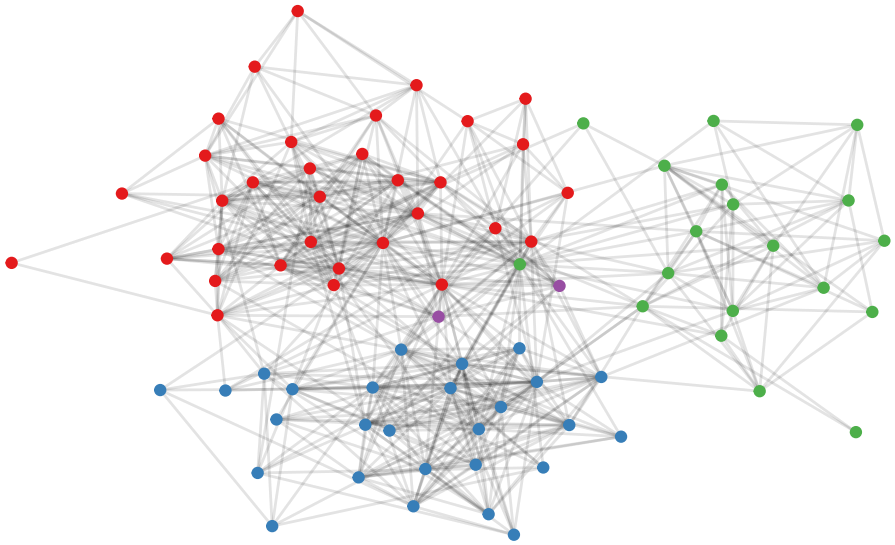

Figure 1: Friendship network of 81 faculty at a UK university. The vertices are labeled by school affiliation.

In the setting in which the labels are known and the edges are assumed to be sampled from some distribution conditioned on the labels, then one inference task is to fit an ERGM to these data. On the other hand, if the labels are unobserved and the task is to identify the labels (up to permutation) assuming that the labels are drawn from some distribution and the edges are also drawn from some distribution conditioned on the hidden labels, then the inference task is community detection.

1.2 Probabilistic Models for Graphs

Given a sample or dataset, a typical analysis task is statistical inference, or the estimation of various parameters under the assumption that the data come from a random distribution or process. These estimated parameters are often then used for making predictions or deriving insights about the population. For example, when fitting a Gaussian mixture model, the data are first assumed to come from a mixture of Gaussians. The model fitting process then involves estimating the means and standard deviations of each Gaussian component, along with the mixture weights. The resulting model provides insight into where each mixture component is located, how disperse each component is, and how the data are distributed between the components.

In order to perform a similar type of analysis for graphs, we must first define probability distributions from which such data can be sampled.

Within the scope of this work, we focus primarily on unweighted and undirected graphs without self-loops, with a brief discussion on generalizing these methods to weighted or directed graphs. The adjacency matrix that describes these graphs is binary, symmetric, and hollow. In this setting, a plausible model is to sample each edge independently from a Bernoulli distribution, i.e., $A_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(P_{ij})$ for some $P_{ij} \in [0, 1]$ for each $i < j$ (setting $A_{ji} = A_{ij}$ since A is symmetric, and $A_{ii} = 0$ since A is hollow). Then similar to how the edges are compiled into an adjacency matrix A, the edge probabilities can be compiled into an edge probability matrix $P \in [0, 1]^{n \times n}$. This type of graph model is defined as a *Bernoulli*

random graph (also called an inhomogeneous Erdös-Rényi graph). If A is the adjacency matrix of a Bernoulli random graph with edge probability matrix P, we denote $A \sim \text{BernoulliGraph}(P)$. If the vertices and edge probabilities are sampled as a sequence of n i.i.d. random variables from probability distribution F, then we denote $A \sim \text{BernoulliGraph}(F, n)$.

Statistical inference is not possible on a general Bernoulli random graph with arbitrary edge probabilities since the number of parameters (individual edge probabilities) is equal to the number of observations (presence or absence of an edge). Additional structure must be introduced. One such structured Bernoulli random graph model is the Erdös-Rényi graph, first proposed by Gilbert (1959), which is defined as follows:

Definition 1.1 (Erdös-Rényi graph)**.** Let P be an $n \times n$ matrix such that each $P_{ij} \equiv \theta \in [0, 1]$ is a constant. Then $A \sim \text{BernoulliGraph}(P)$ is an Erdös-Rényi graph.

Example 1.3 (Maximum likelihood estimator for the Erdös-Rényi graph)**.** One possible estimator for the lone parameter of an Erdös-Rényi graph, θ, is the maximum likelihood estimator, which is found by maximizing the log-likelihood function $\ell(\theta; A) = \log\theta \sum_{i<j} A_{ij} + \log(1-\theta) \sum_{i<j}(1 - A_{ij})$. This can be solved directly by setting $\frac{d\ell}{d\theta} = 0$, which yields $\hat{\theta} = \frac{2|E|}{n(n-1)}$, and $|E| = \sum_{i<j} A_{ij}$ is the number of edges in the graph.

Erdös-Rényi graphs lie on the opposite end of the spectrum in that they restrict the model to a single parameter, which does not result in a very interesting model or reflect many networks observed in real data. Much work has been done in developing various Bernoulli random graph models that strike a balance between structure and flexibility. Two common types of structure for graph models, that are of particular interest in this work, are edge probabilities based on latent communities (Lorrain and White, 1971, Airoldi et al., 2009, Karrer and Newman, 2011, Sengupta and Chen, 2018), called *block models*, and edge probabilities based on positions in a latent space (Young and Scheinerman, 2007, Rubin-Delanchy et al., 2022), called *latent space models*.

Example 1.4. Consider a graph in which the vertices represent a collection of n planets with intelligent life sending out radio waves, and the existence of an edge between vertices i and j represents whether planets i and j have established contact. Since radio waves decay according to the inverse-square law, a plausible model for the edge probability between two vertices is

$$P_{ij} = \frac{C\omega_i\omega_j}{d_{ij}^2}$$

where ω_i is the i^{th} planet's radio signal strength, d_{ij} is the Euclidean distance between planets i and j, and C is a normalizing constant. Let P be the $n \times n$ matrix of these edge probabilities. Then $A \sim \text{BernoulliGraph}(P)$ represents the network of planets that have made contact.

To estimate $\omega = \begin{bmatrix}\omega_1, \cdots, \omega_n\end{bmatrix}^\top$ after observing A and supposing that the relative locations of the planets as well as the normalizing constant C are known, one approach might be maximum likelihood maximization. For simplicity, set $C = 1$. Then the log-likelihood function can be written as:

$$\ell(\omega; A) = \sum_{i<j} A_{ij} \log\left(\frac{\omega_i\omega_j}{d_{ij}^2 - \omega_i\omega_j}\right) + \log(d_{ij}^2 - \omega_i\omega_j) + const.$$

The partial derivatives with respect to each ω_i are:

$$\frac{\partial \ell}{\partial \omega_i} = \sum_{j \neq i} \frac{A_{ij}}{\omega_i} - \frac{\omega_j(1 - A_{ij})}{d_{ij}^2 - \omega_i\omega_j}.$$

Setting the gradient to zero yields n equations with n unknowns, but it is not entirely obvious how to solve this system of equations. Alternatively, we can use gradient ascent or some other numerical optimization method, since this particular log-likelihood function is concave.

Definition 1.2 (Degree of a vertex)**.** Let $G = (V, E)$ be an undirected graph with vertex set $V = \{v_1, ..., v_n\}$. d_i, the degree of vertex v_i is the sum of the weights of edges that connect to

v_i. If G is unweighted, then v_i is equivalently the number of edges that connect to v_i. If G is represented by affinity or adjacency matrix A, then $d_i = \sum_j A_{ij}$.

If G is a Bernoulli random graph with edge probability matrix P, then the degree of each vertex v_i is a random variable, and its expected value (also called expected degree) is $E[d_i] = \sum_j P_{ij}$. For example, the expected degree of each vertex of an Erdös-Rényi graph with n vertices and constant edge probability p is $(n-1)p$.

1.3 Contributions of This Work

In this work, we explore three types of block models, the stochastic block model, the degree corrected block model, and the popularity adjusted block model, as well as a family of latent space models called generalized random dot product graphs.

The contributions of this work are as follows. In section 3, we show that, similar to how the stochastic block model and degree corrected block model are generalized random dot product graphs with specific latent structures, the popularity adjusted block model is also a generalized random dot product graph with a specific latent structure. More specifically, we show that the popularity adjusted block model is a generalized random dot product graph in which the latent vectors are organized by community and each community in the latent space is defined by a K-dimensional subspace, where K is the number of communities in the model. We then exploit this rigid and linear structure for community detection by applying sparse subspace clustering on the adjacency spectral embedding of the graph to infer the community labels. We also propose a novel algorithm called orthogonal spectral clustering, which constructs an embedding for the popularity adjusted block model which forces the communities to lie on *orthogonal* subspaces. Finally, we propose a spectral decomposition based parameter estimation algorithm for the popularity adjusted block model. All three algorithms have consistency properties that result in zero error with increasing graph size.

In section 4, we extend the latent structures of the stochastic block model, degree

corrected block model, and popularity adjusted block model to nonlinear latent structures. Instead of points, line segments, and subspaces, we propose the manifold block model which is a class of generalized random dot product graph in which the latent vectors lie on one of K manifolds. In the case of nonintersecting manifolds, we show that a clustering algorithm based on a nearest algorithms graph approach on an embedding is consistent, resulting in zero clustering error for large graphs, as long as the latent vectors are sampled from the manifolds densely In the case of intersecting manifolds, we propose the K-curves algorithm, which is a type of batch coordinate ascent algorithm which alternates between fitting manifolds to the embedding and reassigning clusters based on proximity to the manifolds.

We conclude with section 5 in which we compare the four block models.

2 Preliminaries

2.1 Block Models for Community Detection

Network analysis is often concerned with community detection. This has motivated statisticians to develop random graph models with inherent community structure in which the probability of an edge between vertices v_i and v_j depend on the community assignments z_i and z_j. More formally, one assigns each vertex v_i a community label $z_i \in \{1, ..., K\}$ and assumes a Bernoulli random graph in which $P_{ij} = g(z_i, z_j, \theta_i, \theta_j)$ for some function $g : \{1, ..., K\}^2 \times \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$ and $\theta_i \in \mathbb{R}^d$ is a vector of real-valued parameters associated with each vertex. While the graph is sampled conditioned on the labels z_i, it is assumed that the labels are unknown when the graph is observed. Within the context of this work, we will call such models *block models*. This type of model framework allows us to frame the goal of community detection as a statistical inference problem: construct estimators for the unobserved true community labels, up to permutation.

In this section, we explore the three main types of block models: the stochastic block model (Lorrain and White, 1971), the degree corrected block model (Karrer and Newman, 2011), and the popularity adjusted block model (Sengupta and Chen, 2018). The main focus is on how these models are connected, particularly in how they are nested models (Noroozi and Pensky, 2022), as well as community detection and parameter estimation via likelihood maximization. We will later return to these block models and the relationship to another class of random graph models in sections 2.6 and 3.

2.2 The Stochastic Block Model

The stochastic block model (SBM) (Lorrain and White, 1971) is the simplest of the three block models and assumes that each pair of communities (k, ℓ) has a constant edge probability $\theta_{k\ell}$. We now give the formal definition of the SBM within the context of Bernoulli random graphs:

Definition 2.1 (Stochastic block model)**.** Let $G = (V, E)$ be a Bernoulli random graph with n vertices, described by random adjacency matrix A. Let $K \geq 1$ be an integer and $\theta_{k\ell} \in [0, 1]$ for each $k, \ell \in \{1, ..., K\}$ (if G is undirected, then $\theta_{k\ell} = \theta_{\ell k}$). Let $z_1, ..., z_n \in \{1, ..., K\}$ be community labels associated with each vertex. If the edge probability matrix P for this graph is such that $P_{ij} = \theta_{z_i, z_j}$ and $A \sim \text{BernoulliGraph}(P)$, then G is a stochastic block model.

There are a few conventions for exactly how a graph is sampled as an SBM. Within the context of this work, we assume that for a particular SBM, the number of communities, K, and the community edge probabilities, $\theta_{k\ell}$, are fixed. Then some possible ways of sampling from an SBM are as follows:

1. Let the size of the graph, n, be fixed. Then each vertex v_i, and its corresponding label, z_i, are also fixed, in addition to the pairwise community edge probabilities $\{\theta_{k\ell}\}_K$, which are all treated as model parameters. Under this sampling scheme, there is no notion of increasing n.

2. The vertices and edges are sampled in sequence, and we suppose that each vertex v_i comes with a corresponding label, z_i, which determines its edge probabilities to the other vertices. Under this sampling scheme, there is no assumed distribution on the labels, which are treated as fixed parameters rather than random variables, along with $\{\theta_{k\ell}\}_K$, but there is a notion of increasing sample size n, which allows for asymptotics.

3. The vertices are sampled in sequence, and each label, z_i, is sampled as $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$, $\sum_k^K \alpha_k = 1$. The edge probabilities are then determined by the sample $z = [z_1 \cdots z_n]^\top$ and the parameters $\{\theta_{k\ell}\}_K$. Under this sampling scheme, both the edges and the labels are random, with fixed parameters $\alpha = [\alpha_1, \cdots, \alpha_K]^\top$ and $\{\theta_{k\ell}\}_K$.

We use the notation $A \sim \text{SBM}(z, \{\theta_{k\ell}\}_K)$ for the first two sampling schemes, and we use $A \sim \text{SBM}(\alpha, \{\theta_{k\ell}\}_K)$ for the third.

An extension to these include Bayesian models in which there are prior distributions for the parameters, e.g., each $\theta_{k\ell}$ has a prior distribution with support $[0, 1]$, such as $\theta_{k\ell} \sim \text{Beta}(a_{k\ell}, b_{k\ell})$, and the mixture parameters α has a prior distribution on the $K-1$ simplex, such as $\alpha \sim \text{Dirichlet}(\beta_1, ..., \beta_K)$. (See McDaid et al. (2013) and Zhang and Zhou (2016).)

Example 2.1 (Stochastic block model with $K = 2$ communities)**.** We construct an SBM with two communities in which $\theta_{11} = 1/2$, $\theta_{12} = 1/8$, and $\theta_{22} = 1/4$. Note that in this example, the within-community edge probabilities are greater than the between-community edge probability. A realization of this graph with $n_1 = n_2 = 32$ (where $n_k = |\{v_i : z_i = k\}|$) is illustrated in figure 2.

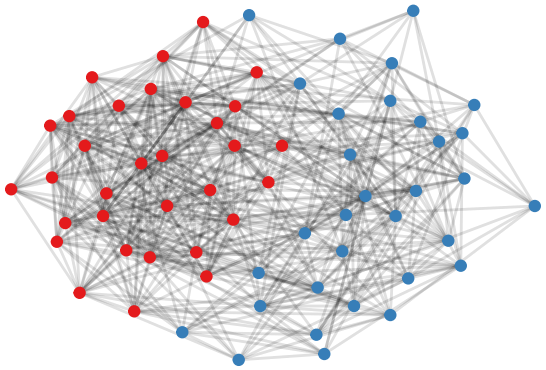

Figure 2: Two-community stochastic block model with $\theta_{11} = 1/2$, $\theta_{22} = 1/4$, and $\theta_{12} = \theta_{21} = 1/8$.

In most conceptions of block models, the within-community edge probability is greater than the between-community edge probability, as in example 2.1. This reflects the general intuition that pairs within the same community are more likely to be connected than pairs between two different communities. For instance, in the UK faculty friendship network in example 1.2, it is natural to expect that faculty within the same school are more likely to be friends than faculty from two different schools, and a cursory look at the visualization of the network appears to confirm this. However, SBMs are not necessarily restricted to this type

of community structure. We could just as easily construct an example in which θ_{11} and θ_{22} are less than θ_{12}. Within the context of block models, we say that a graph is *assortative* if P is positive semidefinite and *disassortative* otherwise, which roughly correspond to communities with higher within-community edge probabilities and lower between-community edge probabilities, respectively. For example, the SBM in example 2.1 is assortative and has edge probability matrix P that is rank 2 with nonzero eigenvalues $n(\frac{3}{8} \pm \frac{\sqrt{2}}{8}) > 0$. In the following example, we describe a network which can be modeled as a disassortative SBM.

Example 2.2 (Dating network as a disassortative stochastic block model)**.** Consider an undirected graph in which each vertex v_i represents users of an online dating service, each label z_i represents the user's gender, and each edge represents a successful match between pairs of users. For simplicity, we will restrict the labels to female and male (denoted as 1 and 2). If we model this as an SBM, then θ_{11} is the probability of a match between two female users, θ_{22} is the probability of a match between two male users, and $\theta_{12} = \theta_{21}$ is the probability of a match between a female user and a male user. Based on trends in the United States (Jones, 2022), a plausible set of edge probabilities is $\theta_{11} = \theta_{22} = 0.02$ and $\theta_{12} = 0.2$, which are used in the sample visualized in figure 3, along with $\alpha_1 = \alpha_2 = 1/2$ and $n = 64$.

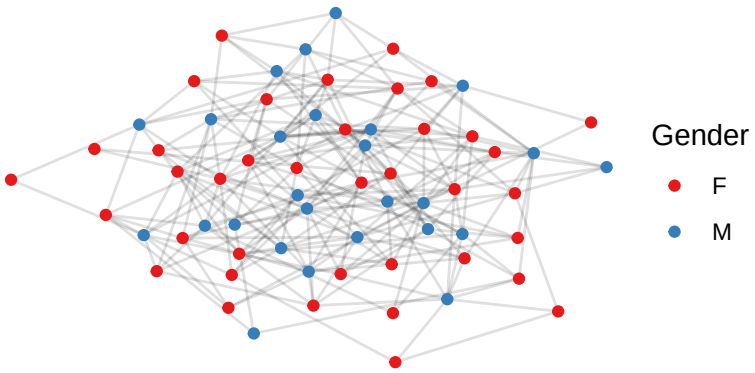

Figure 3: Stochastic block model of a dating network. This model is disassortative.

Example 2.3 (Assortativity and disassortativity of the two-community SBM)**.** Let P be the $n \times n$ probability matrix of a two-community SBM. Then $rank(P) = 2$, and its two nonzero

eigenvalues are:

$$\lambda = \frac{n}{2}\left(\theta_{11} + \theta_{22} \pm \sqrt{(\theta_{11} - \theta_{22})^2 + 4\theta_{12}^2}\right). \tag{2}$$

The first eigenvalue is always positive. The second eigenvalue is positive if $\theta_{11}\theta_{22} > \theta_{12}^2$. Thus, P describes an assortative SBM if $\theta_{11}\theta_{22} > \theta_{12}^2$ or a disassortative SBM if $\theta_{11}\theta_{22} < \theta_{12}^2$. This corresponds to the high-level understanding of assortativity and disassortativity based on whether within or between community edge probabilities are greater.

One approach to statistical inference for the SBM is likelihood maximization. The log-likelihood function can be written as:

$$\ell(z, \theta; A) = \sum_{i<j} \sum_{k,\ell} z_{ik} z_{j\ell} \Big(A_{ij} \log \theta_{k\ell} + (1 - A_{ij}) \log(1 - \theta_{k\ell}) \Big), \tag{3}$$

where $z_{ik} = I(z_i = k)$. If the labels are drawn from a multinomial distribution, then that can also be incorporated into the complete-data log-likelihood:

$$\ell(z, \theta, \alpha; A) = \sum_{k} \sum_{i} z_{ik} \log \alpha_k + \sum_{i<j} \sum_{k,\ell} z_{ik} z_{j\ell} \Big(A_{ij} \log \theta_{k\ell} + (1 - A_{ij}) \log(1 - \theta_{k\ell}) \Big). \tag{4}$$

The SBM is a type of mixture model, and like most classical mixture models and clustering problems, likelihood maximization is NP-hard. But because the SBM is a mixture model, a candidate algorithm for finding a (local) maximum of the likelihood is the widely studied expectation maximization (EM) algorithm (Dempster et al., 1977). However, due to the $z_{ik} z_{j\ell}$ interaction terms, the expectation step cannot be solved in closed form (Kolaczyk and Csárdi, 2014). While the labels are drawn independently, they are not necessarily conditionally independent. Nevertheless, making the conditional independence relaxation allows us to write the expectation and maximization steps in closed form. (The same type of approximation was used in the Bayesian context by Zhang and Zhou (2016).) As an example,

we write the EM algorithm for the likelihood function in equation (3) in algorithm 1.

Algorithm 1: Approximate EM algorithm for the SBM

Data: Adjacency matrix A, number of communities K

Result: Estimated community label probabilities $\{\pi_{ik}\}$ for which each $\pi_{ik} = P(z_i = k \mid A)$, estimated community edge probabilities $\{\hat{\theta}_{k\ell}\}_K$

Initialize $\{\pi_{ik}\}$, $\{\theta_{k\ell}\}$.
while $\|\nabla \ell\| > \epsilon$ **do**
  for $i = 1, ..., n$ **do**
    for $k = 1, ..., K$ **do**
      E-step: $\pi_{ik} \propto \exp\left(\sum_{j \neq i} \sum_{\ell} \pi_{j\ell} (A_{ij} \log \hat{\theta}_{k\ell} + (1 - A_{ij}) \log(1 - \hat{\theta}_{k\ell})) \right)$.
    end
  end
  for $k = 1, ..., K$ **do**
    for $\ell = 1, ..., K$ **do**
      M-step: $\hat{\theta}_{k\ell} = \frac{\sum_{i<j} A_{ij} \pi_{ik} \pi_{j\ell}}{\sum_{i<j} \pi_{ik} \pi_{j\ell}}$.
    end
  end
end

Applying algorithm 1 to example 2.1 with initial guesses of $\pi_{ik} = 0.5$ for each i, k, $\hat{\theta}_{11} = \hat{\theta}_{22} = 0.9$, and $\hat{\theta}_{12} = 0.1$ results in a community detection error rate of 4.7% and parameter estimates $\hat{\theta}_{11} = 0.503$, $\hat{\theta}_{22} = 0.241$, and $\hat{\theta}_{12} = 0.109$, compared to the true parameters $\theta_{11} = 0.5$, $\theta_{22} = 0.25$, and $\theta_{12} = 0.125$. The same algorithm applied to example 2.2, which is a disassortative model, using initial guesses of $\pi_{ik} = 0.5$, $\hat{\theta}_{11} = \hat{\theta}_{22} = 0.1$, and $\hat{\theta}_{12} = 0.9$, results in a community detection error rate of 0% and parameter estimates $\hat{\theta}_{11} = 0.034$, $\hat{\theta}_{22} = 0.018$, and $\hat{\theta}_{12} = 0.201$, compared to the true parameters $\theta_{11} = \theta_{22} = 0.02$ and $\theta_{12} = 0.2$.

Example 2.4. Recall the UK faculty friendship network from example 1.2. Suppose we

assume that this graph is an SBM. Then applying algorithm 1 to this network yields a community detection accuracy of 79.7%. For this analysis, the school with two faculty members is omitted (so $K = 3$ and $n = 79$), and the EM algorithm was initialized with random $\{\pi_{ik}\}$, $\hat{\theta}_{kk} = 0.9$, and $\hat{\theta}_{k\ell} = 0.1$ for $k \neq \ell$.

2.3 Random Graph Models and Sparsity

So far, we have described *dense* Bernoulli random graphs. For such models, as the sample size n increases, the expected degree of each node grows linearly. For example, consider the SBM with the third sampling setup described in section 2.2. Then the expected degree of vertex v_i is

$$\begin{aligned} E[d_i] &= \sum_j P_{ij} \\ &= \sum_{j \neq i} \theta_{z_i, z_j} \\ &= \Big(\sum_k n_k \theta_{z_i,k} \Big) - \theta_{z_i, z_i} \\ &= n \Big(\sum_k \alpha_k \theta_{z_i,k} \Big) - \theta_{z_i, z_i} \\ &= O(n). \end{aligned}$$

In many real networks, the degree of each vertex often does not grow proportionally with the size of the network. To account for this, a sparsity factor $\rho_n \in (0, 1]$ is introduced in the edge probabilities, i.e., $P_{ij} \leftarrow \rho_n P_{ij}$, for some sequence $\{\rho_n\}$ such that $\lim_{n\to\infty} \rho_n = 0$.

For example, a sparse SBM has edge probabilities $P_{ij} = \rho_n \theta_{z_i, z_j}$, for which we use the notation $A \sim \text{SBM}(z, \{\theta_{k\ell}\}_K; \rho_n)$ or $A \sim \text{SBM}(\alpha, \{\theta_{k\ell}\}_K; \rho_n)$, depending on whether we treat the labels as random or fixed. Then the expected degree grows as $O(n\rho_n)$ instead of linearly as $O(n)$. For the sake of unifying the sparse and dense regimes, we also allow for the special case $\rho_n = 1$ and include the sparsity factor throughout, unless otherwise stated. Finally, we also note that while ρ_n limits the rate of growth of the expected degree, our

theoretical results still require $n\rho_n$ to diverge to infinity, albeit at a slower rate than $O(n)$. In general, for consistency in estimation and inference, most results on Bernoulli random graphs require $n\rho_n = \omega\Big((\log n)^c\Big)$ for some $c > 1$. (See Abbe (2018), Xie (2021), and Rubin-Delanchy et al. (2022) for further discussion.)

2.4 Generalizations of the Stochastic Block Model: The Degree Corrected Block Model and the Popularity Adjusted Block Model

The SBM assumes homogeneity within each community. In particular, in the SBM, each vertex's expected degree and affinity toward each community depends only on its community label. The degree corrected block model (DCBM) and popularity adjusted block model (PABM) are generalizations of the SBM that address these limitations. In particular, the DCBM allows each vertex to have an expected degree independent of its label, and the PABM allows each vertex to have edge probabilities to each community independent of its own label. We also show that the DCBM is a special case of the PABM and that the SBM, DCBM, and PABM are nested models.

Definition 2.2 (Degree corrected block model)**.** Let $G = (V, E)$ be a Bernoulli random graph with n vertices, described by random adjacency matrix A. Let $K \geq 1$ be an integer that describes the number of communities and $\theta_{k\ell} \in [0, 1]$ for each $k, \ell \in \{1, ..., K\}$ (if G is undirected, then $\theta_{k\ell} = \theta_{\ell k}$), as in the SBM. Let $z_1, ..., z_n \in \{1, ..., K\}$ be community labels associated with each vertex. In addition, each vertex v_i has a degree correction parameter $\omega_i \in [0, 1]$. If the edge probability matrix P for this graph is such that $P_{ij} = \rho_n \theta_{z_i, z_j} \omega_i \omega_j$ and $A \sim \text{BernoulliGraph}(P)$, then G is a degree corrected block model. We use the notation $A \sim \text{DCBM}(z, \{\theta_{k\ell}\}_K, \omega; \rho_n)$ to denote a random adjacency matrix drawn from the DCBM with labels z, base edge probabilities $\{\theta_{k\ell}\}_K$ (of which there are $K(K+1)/2$), and degree correction factors $\omega \in \mathbb{R}^n$. Under the sampling scheme in which the labels are drawn from a

multivariate distribution with class probabilities α, we use the notation

$A \sim \text{DCBM}(\alpha, \{\theta_{k\ell}\}_K, \omega; \rho_n)$.

Example 2.5 (Dating network as a disassortative DCBM)**.** In example 2.2, we modeled an online dating network as an SBM in which the vertices correspond to individuals, the communities correspond to genders, and edges correspond to whether the online dating service successfully matched pairs of individuals. In this model, there is an implicit assumption that for each pair of genders, there is a constant probability of a match, e.g., each male member has a constant probability of matching with another male member as well as a (different) constant probability of matching with a female member. This does not take into account that some individuals are more likely to form connections than others, based on each individual's activity on the online service or popularity with other users. Modeling this as a DCBM allows us to take this into account by adjusting each user's expected degree by a factor of ω_i. Figure 4 is a visualization of the network from example 2.2 modified as a DCBM with parameters $\theta_{11} = \theta_{22} = 0.03$, $\theta_{12} = 0.3$, each ω_i drawn uniformly between $1/3$ and 1, and $\rho_n = 1$. The figure shows that some individuals (vertices) are a lot more popular than others in the form of having more matches with other individuals (edges).

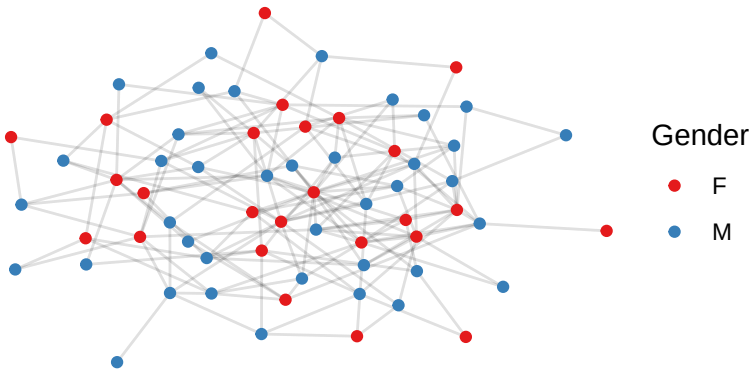

Figure 4: A dating network modeled as a DCBM. This model is disassortative.

The DCBM allows for each vertex to have its own expected degree while still maintaining community structure. However, it still assumes that each vertex within a

community has the same relative propensity to connect to each community. Referring again to the dating network example (examples 2.2 and 2.5), the model assume that each individual is ten times more likely to connect with a member of the opposite sex than with a member of the same sex, without regard to each individual's preferences. The popularity adjusted block model (Sengupta and Chen, 2018) allows each vertex to have its own affinity toward each community. In the dating network example, the PABM allows us to model each individual's sexual orientation instead of using population level averages.

Definition 2.3 (Popularity adjusted block model)**.** Let $G = (V, E)$ be an undirected Bernoulli random graph with n vertices, described by random adjacency matrix A. Let $K \geq 1$ be an integer that describes the number of communities, and let $z_1, ..., z_n \in \{1, ..., K\}$ be community labels associated with each vertex. Suppose that each vertex v_i has K popularity parameters $\lambda_{i1}, ..., \lambda_{iK}$ for which each λ_{ik} describes v_i's affinity toward community k. Then if the edge probability matrix P for this graph is such that $P_{ij} = \rho_n \lambda_{i,z_j} \lambda_{j,z_i}$ and $A \sim \text{BernoulliGraph}(P)$, then G is a popularity adjusted block model. We use the notation $A \sim \text{PABM}(z, \{\lambda_{i,k}\}_K; \rho_n)$ to denote a random adjacency matrix drawn from the PABM with labels z and popularity parameters $\{\lambda_{i,k}\}_K$. Under the sampling scheme in which the labels are drawn from a multivariate distribution with class probabilities α, we use the notation $A \sim \text{PABM}(\alpha, \{\lambda_{i,k}\}_K; \rho_n)$.

Example 2.6 (Dating network as a PABM)**.** In examples 2.2 and 2.5, we modeled an online dating network as a SBM and a DCBM, in which the vertices correspond to individuals, the community labels correspond to genders, and the edges represent whether the online dating service successfully matched pairs of individuals. In these models, it is assumed that all individuals have the same sexual orientation, i.e., the odds of matching with a member of the same gender vs. matching with a member of the opposite gender is constant across all members of each gender. Using the parameters specified in these examples, we assume that each individual is 10 times more likely to match with a member of the opposite gender

compared to a member of the same gender. The PABM allows us to model each individual's sexual orientation. Figure 5 is a visualization of the online dating network modeled as a PABM in which the popularity parameters are drawn as $\lambda_{ik} \sim \text{Beta}(1, 6)$ if $z_i = k$ and $\lambda_{ik} \sim \text{Beta}(2, 2)$ if $z_i \neq k$. Like in the SBM and DCBM examples, the average edge probability between genders is approximately ten times that of the average edge probability within genders, but unlike in the SBM and DCBM examples, we can see that some vertices have more connections to its own gender and some have more connections to the opposite gender.

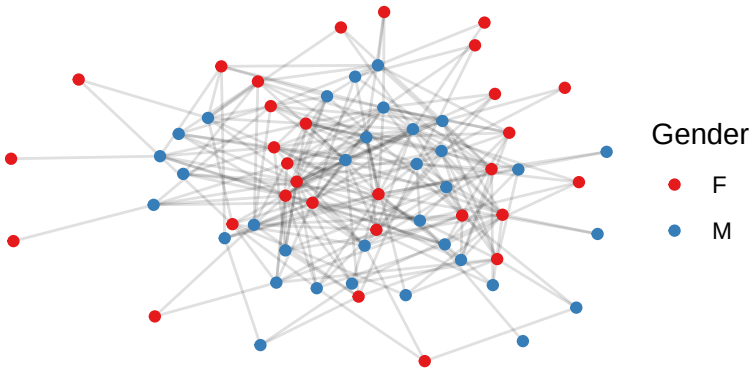

Figure 5: A dating network modeled as a PABM. This model is disassortative.

Like the SBM (and most mixture models), fitting a DCBM or PABM to a graph via direct likelihood maximization is NP-hard. However, like the SBM (and most mixture models), the EM algorithm is a candidate for (indirect) likelihood maximization. The EM algorithms for the DCBM and PABM are similar to algorithm 1, and the same independence relaxation is required. Note that it is required for each $\theta_{k\ell}\omega_i\omega_j \in (0, 1)$ (for the DCBM) or $\lambda_{ik}\lambda_{j\ell} \in (0, 1)$ (for the PABM), and some attention is required to maintain this constraint when implementing these algorithms.

Remark. Unlike the SBM (when excluding the sparsity factor), each DCBM and PABM is not unique and therefore non-identifiable. For example, in the case of the DCBM, doubling each $\theta_{k\ell}$ and dividing each ω_i by $\sqrt{2}$ results in the same edge probability matrix. Karrer and

Algorithm 2: Approximate EM algorithm for the DCBM.

Data: Adjacency matrix A, number of communities K

Result: Estimated community label probabilities $\{\pi_{ik}\}$ for which each $\pi_{ik} = P(z_i = k \mid A)$, estimated community edge probabilities $\{\hat{\theta}_{k\ell}\}_K$, estimated degree factors $\hat{\omega}$.

Initialize $\{\pi_{ik}\}$, $\{\hat{\theta}_{k\ell}\}_K$, $\hat{\omega}$.

while $\|\nabla \ell\| > \epsilon$ **do**

  for $i = 1, ..., n$ **do**

    for $k = 1, ..., K$ **do**

      E-step: $\pi_{ik} \propto \exp\left(\sum_{j \neq i} \sum_{\ell} \pi_{j\ell} (A_{ij} \log(\hat{\theta}_{k\ell} \hat{\omega}_i \hat{\omega}_j) + (1 - A_{ij}) \log(1 - \hat{\theta}_{k\ell} \hat{\omega}_i \hat{\omega}_j)) \right)$.

    end

  end

  for $k = 1, ..., K$ **do**

    for $\ell = 1, ..., K$ **do**

      M-step for $\theta_{k\ell}$: $\hat{\theta}_{k\ell} = \frac{\sum_{i<j} A_{ij} \pi_{ik} \pi_{j\ell}}{\sum_{i<j} \pi_{ik} \pi_{j\ell} \hat{\omega}_i \hat{\omega}_j}$.

    end

  end

  for $i = 1, ..., n$ **do**

    M-step for ω_i: $\hat{\omega}_i = \frac{\sum_{j \neq i} \sum_{k,\ell} \pi_{ik} \pi_{j\ell} A_{ij}}{\sum_{j \neq i} \sum_{k,\ell} \pi_{ik} \pi_{j\ell} \hat{\theta}_{k\ell} \hat{\omega}_j}$.

  end

end

Algorithm 3: Approximate EM algorithm for the PABM.

Data: Adjacency matrix A, number of communities K

Result: Estimated community label probabilities $\{\pi_{ik}\}$ for which each $\pi_{ik} = P(z_i = k \mid A)$, estimated popularity parameters $\{\hat{\lambda}_{ik}\}_K$.

Initialize $\{\pi_{ik}\}$, $\{\hat{\lambda}_{ik}\}_K$.

while $\|\nabla \ell\| > \epsilon$ **do**

  for $i = 1, ..., n$ **do**

    for $k = 1, ..., K$ **do**

      E-step: $\pi_{ik} \propto \exp\left(\sum_{j \neq i} \sum_{\ell} \pi_{j\ell} (A_{ij} \log(\hat{\lambda}_{i\ell}\hat{\lambda}_{jk}) + (1 - A_{ij}) \log(1 - \hat{\lambda}_{i\ell}\hat{\lambda}_{jk})) \right)$.

    end

  end

  for $i = 1, ..., n$ **do**

    for $\ell = 1, ..., K$ **do**

      M-step: $\hat{\lambda}_{i\ell} = \frac{\sum_{j \neq i} \sum_k \pi_{ik}\pi_{j\ell} A_{ij}}{\sum_{j \neq i} \sum_k \pi_{ik}\pi_{j\ell}\hat{\lambda}_{jk}}$.

    end

  end

end

Newman (2011) suggest imposing the constraint $\sum_{i:z_i=k} \omega_i = 1$ for each $k \in \{1, ..., K\}$, but other similar constraints can be used to enforce uniqueness, such as $\max_i \omega_i = 1$ and $\min_i \omega_i = 0$. Despite the lack of uniqueness without constraints, a naive EM type algorithm similar to the one in section 2.2 results in high community detection accuracy for most realizations of the DCBM and PABM, although it is not clear what its asymptotic properties are or whether there are any theoretical guarantees. Likewise, it is not immediately obvious how to approach the estimation of the $\theta_{k\ell}$'s, ω_i's, or λ_{ik}'s due to their non-uniqueness. Noroozi et al. (2019) suggest focusing estimation on the edge probability matrix P instead of the individual parameters to avoid this ambiguity. Nevertheless, the naive EM algorithms appear to work well, at least for community detection (estimating the labels z_i, e.g., by setting $\hat{z}_i = \arg\max_k \pi_{ik}$), based on empirical results. Applying algorithm 2 to example 2.5 results in 84% accuracy, and applying algorithm 3 to example 2.6 results in 95% accuracy, but they fail to capture the original (which are not necessarily "true") parameters.

Example 2.7. Recall the UK faculty friendship graph from example 1.2. In example 2.4, we fit an SBM to this graph using an EM algorithm (algorithm 1), yielding a community detection accuracy of 79.7%. Starting with the output of algorithm 1 and applying algorithm 2 yields a greater accuracy of 94.9%. This is expected, since the SBM is a special case of the DCBM, i.e., the SBM is the DCBM with some of the parameters restricted. Therefore, it is not immediately obvious from the improved accuracy whether the DCBM is a better fit for this graph than the SBM. Looking at the dispersion of the fitted degree factors $\hat{\omega}$, there does appear to be some indication that the DCBM is a better fit (figure 6).

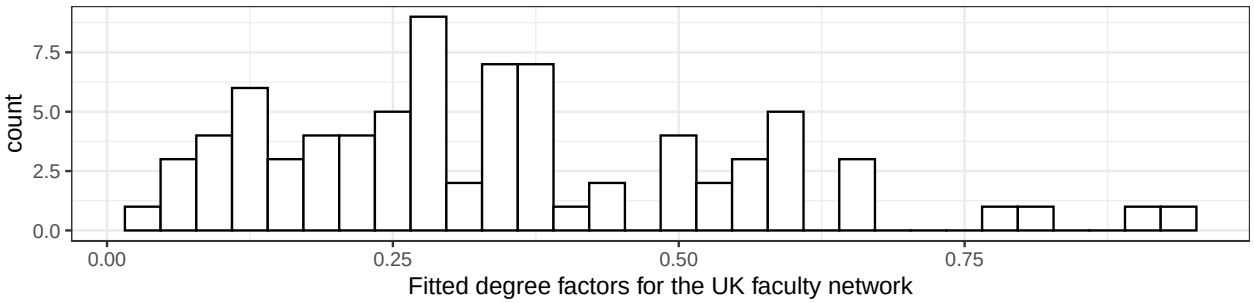

Figure 6: Histogram of the fitted degree factors for the UK faculty friendship network assuming that it is a DCBM. The DCBM was fit to this graph using algorithm 2. The dispersion of the degree factors suggests that the DCBM is a better fit than the SBM.

2.5 A Hierarchy of Block Models

The PABM, DCBM, and SBM are nested models, that is, the SBM is a special case of the DCBM which in turn is a special case of the PABM. In this section, these connections are made explicit, and we explore the implications for modeling. For ease of notation, throughout this section the rows and columns of P are assumed to be organized by community in increasing order, i.e., the first n_1 rows and columns of P correspond to community 1, the next n_2 rows and columns correspond to community 2, etc. We use the notation $P^{(k\ell)} = (P^{(\ell k)})^\top$ to denote the block of P that corresponds to the edges between communities k and ℓ. In section 3, we will deal with this notation and convention more carefully. For additional details on the connections among the SBM, DCBM, and PABM, refer to Noroozi and Pensky (2022).

To see that the SBM is a special case of the DCBM, we can simply restrict the degree parameters ω_i to all be a constant value, e.g., $\omega_i \equiv 1$. But in order to better understand the relationship between these two models, we will break down the models further.

Define B as the $K \times K$ matrix of edge probabilities between pairs of communities in the SBM. Then $B_{k\ell} = \theta_{k\ell}$ and $P_{ij} = \theta_{z_i,z_j} = B_{z_i,z_j}$. We can further show that block $P^{(k\ell)}$ can be written as $P^{(k\ell)} = e^{(k)} B_{k\ell} (e^{(\ell)})^\top$ where $e^{(k)}$ is the 1-vector of dimension k. Since $B_{k\ell}$ is a

scalar, $P^{(k\ell)} = e^{(k)} B_{k\ell} (e^{(\ell)})^\top$ is a singular value decomposition of a rank 1 matrix and thus each block $P^{(k\ell)}$ is rank 1. Then we can see that the entire edge probability matrix can be written as

$$P = ZBZ^\top$$

and Z is an $n \times K$ matrix in which $Z_{ik} = I(z_i = k) = z_{ik}$. Thus, if B is full rank, i.e., $\text{rank}(B) = K$, then P is also rank K.

To extend the SBM to the DCBM, we replace $e^{(k)}$ with the vector $\omega^{(k)}$ which contains the degree parameters of vertices in community k. Then for the DCBM, $P^{(k\ell)} = \omega^{(k)} B_{k\ell} (\omega^{(\ell)})^\top$. Similar to the SBM, this is a singular value decomposition of a rank 1 matrix. The full edge probability matrix can be written as

$$P = (\Omega Z) B (\Omega Z)^\top$$

for $\Omega = \text{diag}(\omega)$. Since Ω is diagonal, we again have that $\text{rank}(P) = K$ (again, assuming that B is full rank). Thus, the dimensionality of the DCBM is the same as the SBM, and to restrict the DCBM to the PABM, let $\Omega = I$.

To extend the DCBM to the PABM, we note that the $k\ell^{\text{th}}$ block of the edge probability matrix can be written as $P^{(k\ell)} = \lambda^{(k\ell)} (\lambda^{(\ell k)})^\top$ where $\lambda^{(k\ell)} \in \mathbb{R}^{n_k}$ is the vector of popularity parameters belonging to vertices in community k that point to community ℓ, i.e., $\lambda_i^{(k\ell)} = \lambda_{i+\sum_{s=1}^{k-1} n_s, \ell}$. Similar to how $\omega^{(k)}$ is a sub-vector of ω To match this decomposition with the one for the DCBM, we can fix each $B_{k\ell} \equiv 1$ and write $P^{(k\ell)} = \lambda^{(k\ell)} B_{k\ell} (\lambda^{(\ell k)})^\top$. Restricting each $\lambda^{(k\ell)} = \omega^{(k)}$ reduces the PABM to the DCBM. To write out the full edge probability matrix of the PABM, let $\Lambda^{(k)} = \begin{bmatrix} \lambda^{(k1)} & \cdots & \lambda^{(kK)} \end{bmatrix}$ and $X = \text{blockdiag}(\Lambda^{(1)}, ..., \Lambda^{(K)})$. Then for some $K^2 \times K^2$ permutation matrix Π,

$$P = X \Pi X^\top.$$

Table 1: Summary of an hierarchical or nested view of block momdels.

Model	Parameters	rank(P)	Matrix Decomposition	Reduction to simpler model
SBM	$K(K-1)/2$	K	$P = ZBZ^\top$	
DCBM	$K(K-1)/2 + n$	K	$P = (\Omega Z)B(\Omega Z)^\top$	$\omega = e$
PABM	nK	K^2	$P = X\Pi X^\top$	$\lambda_{ik} = \omega_i\sqrt{\theta_{z_i,k}}$

From this decomposition, we can see that unlike the SBM and DCBM, $\text{rank}(P) = \text{rank}(X) = K^2$. We will explore this decomposition of the edge probability matrix of the PABM in more detail in section 3.

The full hierarchy of nested block models is summarized in table 1.

2.6 The Generalized Random Dot Product Graph

In the Bernoulli random graph, the edge probability matrix P is of size $n \times n$, but when $K \ll n$, the structure of P is much simpler than its dimensions imply. In particular, P is rank K in the SBM and DCBM and rank K^2 in the PABM. This property lends these block models to be analyzed as another family of Bernoulli random graphs with low rank called the random dot product graph (RDPG) and generalized random dot product graph (GRDPG). In these models, P is decomposed into an outer product of two low-rank data matrices.

Definition 2.4 (Generalized random dot product graph)**.** Let $p \geq 1$ and $q \geq 0$ be integers. Define $I_{p,q}$ as the block diagonal matrix $I_{p,q} = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}$ where I_p and I_q are the identity matrices of dimensions $p \times p$ and $q \times q$ respectively. Denote $d = p + q$ and let $\mathcal{X}$ be the subset of $\mathbb{R}^d$ such that, for any $x, y \in \mathcal{X}$, we have $x^\top I_{p,q} y \in [0, 1]$. Let $X = \begin{bmatrix} x_1 \mid \cdots \mid x_n \end{bmatrix}^\top$ be an $n \times d$ matrix with rows $x_i \in \mathcal{X}$. A graph G with adjacency matrix A is said to be a generalized random dot product graph with latent positions X, sparsity parameter ρ_n, and signature (p, q) if $A \sim \text{BernoulliGraph}(P)$ where the edge probability matrix P is given by $P = \rho_n X I_{p,q} X^\top$, i.e., the entries of P are of the form $P_{ij} = \rho_n x_i^\top I_{p,q} x_j$.

We use the notation $A \sim \text{GRDPG}_{p,q}(X; \rho_n)$ to denote a random adjacency matrix A

drawn from latent positions X, sparsity parameter ρ_n, and signature (p, q). If the n vectors in X are drawn from probability distribution F on support $\mathcal{X}$, we use the notation $A \sim \text{GRDPG}_{p,q}(F; \rho_n)$.

If $q = 0$ and so $d = p$, (i.e., P is positive semidefinite), then we call this a random dot product graph. In this case, we use the notation $A \sim \text{RDPG}(X; \rho_n)$ and $A \sim \text{RDPG}(F; \rho_n)$.

Definition 2.5 (Indefinite orthogonal group)**.** The indefinite orthogonal group with signature (p, q) is the set $\{Q \in \mathbb{R}^{d\times d} \colon QI_{p,q}Q^\top = I_{p,q}\}$, denoted as $\mathbb{O}(p, q)$. Here $d = p + q$.

Remark. The latent vectors that produce $XI_{p,q}X^\top = P$ are not unique (Rubin-Delanchy et al., 2022). More specifically, if $P_{ij} = x_i^\top I_{p,q} x_j$, then for any $Q \in \mathbb{O}(p, q)$ we also have $(Qx_i)^\top I_{p,q}(Qx_j) = x_i^\top (Q^\top I_{p,q} Q)x_j = x_i^\top I_{p,q} x_j = P_{ij}$. Unlike in the RDPG case, transforming the latent positions via multiplication by $Q \in \mathbb{O}(p, q)$ does not necessarily maintain interpoint angles or distances.

Now that we defined the GRDPG as a generative model for Bernoulli random graphs, we need a method for estimation. We can use Adjacency Spectral Embedding (ASE) (Sussman et al., 2012) to recover the latent vectors of a GRDPG. This procedure consists of taking the spectral decomposition of A and keeping the $p + q$ largest eigenvalues (in modulus) and corresponding eigenvectors of A.

Definition 2.6 (Adjacency spectral embedding)**.** Let A be an $n \times n$ adjacency matrix and denote the eigendecomposition of A as

$$A = \sum_i \hat{\lambda}_i \hat{v}_i \hat{v}_i^\top, \quad \text{where } |\hat{\lambda}|_1 \geq |\hat{\lambda}_2| \geq \cdots \geq |\hat{\lambda}_n|.$$

Let $\hat{D}$ be a diagonal matrix whose diagonal entries are the eigenvalues $\{\hat{\lambda}_i\}_{i=1}^d$ arranged in decreasing values (not in decreasing modulus) and let $\hat{V}$ be the $n \times d$ matrix whose columns are the corresponding eigenvectors $\{\hat{v}_i\}_{i=1}^d$ arranged in the same order as the diagonal entries of $\hat{D}$. Now define $\hat{X} = \hat{V}|\hat{D}|^{1/2}$ where the $|\cdot|$ operation is applied elementwise to the entries

of $\hat{D}$. Then $\hat{X}$ serves as an estimate of X, up to some non-identifiable orthogonal transformation Q as described in remark 2.6 and definition 2.5; see Rubin-Delanchy et al. (2022) for further details.

Sussman et al. (2012) and Rubin-Delanchy et al. (2022) showed that the ASE is consistent in estimating the latent vectors, up to an indefinite orthogonal transformation. More specifically, for sparsity parameters such that $n\rho_n = \omega(\log^{4c} n)$ for some constant $c > 1$ and indefinite orthogonal transformation $Q \in \mathbb{O}(p, q)$,

$$\max_i \|Q\hat{x}_i - x_i\| = O_P\left(\frac{\log^c n}{n^{1/2}}\right)$$

where $x_i^\top$ and $\hat{x}_i^\top$ are the rows of $\hat{X}$ (from the ASE) and X, respectively.

Example 2.8. Suppose that latent vectors for an RDPG are drawn from the curve $x(t) = \left[t^2, 2t(t-1)\right]^\top$, $t \in [0, 1]$. Let $x_i = x(t_i)$. Then for each $i < j$, the edges are drawn as $A_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(x_i^\top x_j)$. If after observing A, we wish to estimate the curve, the ASE provides an estimator in which the estimated latent vectors converge to the curve, up to an orthogonal transformation (figure 7).

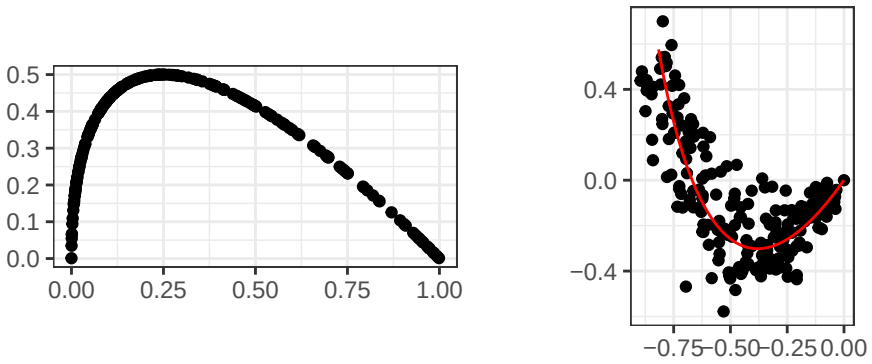

Figure 7: 200 points drawn from a curve (left), and the ASE of an adjacency matrix drawn from this curve as an RDPG (right). The red line is the original curve rotated to match the ASE.

2.7 Connecting the SBM and DCBM to the GRDPG

In this section, we connect the SBM and DCBM to the GRDPG and show how this connection is useful for community detection and parameter estimation. We begin by showing that *all* Bernoulli random graphs are GRDPGs.

Proposition 2.1. *Let* $A \sim \text{BernoulliGraph}(P; \rho_n)$. *Then there exists a* $d \leq n$ *and* $p + q = d$ *such that for some* $X \in \mathbb{R}^d$, $A \sim \text{GRDPG}_{p,q}(X; \rho_n)$. *If* P *is positive semidefinite, then* $A \sim \text{RDPG}(X; \rho_n)$.

Proof. P is the edge probability matrix of a GRDPG if and only if there exists an $X \in \mathbb{R}^{n \times d}$ such that $P = X I_{p,q} X^\top$. P is a square and symmetric matrix, so it has an eigendecomposition $P = V \Lambda V^\top$ where $V \in \mathbb{R}^{n \times d}$, Λ is a $d \times d$ diagonal matrix, and $d = \text{rank}(P)$. Let p and q respectively be the number of positive and negative eigenvalues of P. Then we can equivalently write $P = V|\Lambda|^{1/2} I_{p,q} |\Lambda|^{1/2} V^\top$ where $|\cdot|$ is applied element-wise. This is exactly the form of the GRDPG with latent vectors $V|\Lambda|^{1/2}$, i.e., if $A \sim \text{BernoulliGraph}(P; \rho_n)$, then $A \sim \text{GRDPG}_{p,q}(V|\Lambda|^{1/2}; \rho_n)$. If P is positive semidefinite, then $q = 0$ and we have $P = V\Lambda^{1/2}(V\Lambda^{1/2})^\top$, so $A \sim \text{RDPG}(V\Lambda^{1/2}; \rho_n)$. □

The SBM and DCBM are both Bernoulli random graphs, so they are also GRDPG. Recall that the SBM and DCBM are rank K. Therefore, we can find $X \in \mathbb{R}^{n \times K}$ that are the latent vectors of the SBM and DCBM. If we analyze the SBM and DCBM as GRDPGs, then we observe community-wise structures in the latent space that are useful for estimation. The latent vectors that generate the SBM consist of K point masses in $\mathbb{R}^K$, each of which correspond to a community, and similarly, the latent vectors that generate the DCBM consist of K line segments emitting from the origin in $\mathbb{R}^K$, each of which correspond to a community, and this property is observed in both assortative and disassortative models (Rubin-Delanchy et al., 2022). The ASE recovers these structures with noise (which vanishes almost surely as $n \to \infty$), so clustering applied to the ASE can recover the original

community labels. Rubin-Delanchy et al. (2022) showed that this approach to community detection for the SBM and DCBM is asymptotically consistent. More specifically, under certain sparsity conditions, clustering on the ASE of the SBM and DCBM results in zero misclustered vertices. Note that for a given P, X is not unique, i.e., for any $Q \in \mathbb{O}(p, q)$, XQ results in the same P (see remark 2.6). However, this does not change the general structure of the SBM and DCBM in the latent space. Multiplication by $Q \in \mathbb{O}(p, q)$ does not change the fact that the latent vectors of an SBM are point masses or that the latent vectors of a DCBM are line segments, since Q is a linear transformation.

Example 2.9. Consider a two-community assortative SBM. Recall that assortativity in SBMs is characterized by positive definite edge probability matrices. Therefore, this model is also an RDPG. To show this explicitly, suppose that P is organized by community such that the $P^{(k\ell)}$ block corresponds to edges from community k to ℓ, and consider

$$X = \begin{bmatrix} \sqrt{\theta_{11}} & 0 \\ \vdots & \vdots \\ \sqrt{\theta_{11}} & 0 \\ \sqrt{\theta_{12}^2/\theta_{11}} & \sqrt{\theta_{22} - \theta_{12}^2/\theta_{11}} \\ \vdots & \vdots \\ \sqrt{\theta_{12}^2/\theta_{11}} & \sqrt{\theta_{22} - \theta_{12}^2/\theta_{11}} \end{bmatrix}.$$

This describes two point masses in $\mathbb{R}^2$, one at $(\sqrt{\theta_{11}}, 0)$ and another at $(\sqrt{\theta_{12}^2/\theta_{11}}, \sqrt{\theta_{22} - \theta_{12}^2/\theta_{11}})$. Taking the dot product of two vectors in the first point mass results in θ_{11}, taking the dot product of two vectors in the second point mass results in θ_{22}, and taking the dot product of a vector from the first point mass with a vector from the second point mass results in θ_{12}, which correspond exactly to the edge probabilities between communities of the SBM. Therefore, $(XX^\top)_{ij} = \theta_{z_i, z_j}$. Note that this latent configuration implicitly assumes that $\theta_{11}\theta_{22} > \theta_{12}^2$, which is the condition we saw for the assortative two-community SBM in example 2.3.

If instead, we have a DCBM, consider the latent configuration ΩX where $\Omega = \text{diag}(\omega_1, ..., \omega_n)$ is the diagonal matrix of degree parameters. Then $P = \Omega X(\Omega X)^\top$ is the edge probability of a DCBM. Writing out ΩX explicitly yields:

$$X = \begin{bmatrix} \omega_1\sqrt{\theta_{11}} & 0 \\ \vdots & \vdots \\ \omega_{n_1}\sqrt{\theta_{11}} & 0 \\ \omega_{n_1+1}\sqrt{\theta_{12}^2/\theta_{11}} & \omega_{n_1+1}\sqrt{\theta_{22} - \theta_{12}^2/\theta_{11}} \\ \vdots & \vdots \\ \omega_n\sqrt{\theta_{12}^2/\theta_{11}} & \omega_n\sqrt{\theta_{22} - \theta_{12}^2/\theta_{11}} \end{bmatrix}.$$

The resulting rows of X are point masses that have been stretched out, turning into rays.

The latent structures of the SBM and DCBM when viewed as GRDPGs lend themselves to convenient community detection algorithms based on existing clustering algorithms for Euclidean data. The general strategy for these models is to construct the appropriate ASE, choosing the number of components based on the number of communities and whether to use positive or negative eigenvalues depending on assortativity or disassortativity, and then to apply the appropriate Euclidean-based clustering algorithm for the specific structure. In the case of the SBM, after embedding in K dimensions, since the original latent structure consists of K point masses, an appropriate clustering algorithm might be one based on centroids, such as K-means clustering or Gaussian mixture models. In the case of the DCBM, after embedding in K dimensions, since the original latent structure consists of K rays emanating from the origin, an appropriate clustering algorithm is one based on angles between points rather than distances, such as kernel K-means clustering using the cosine kernel. Another clustering algorithm for the ASE of a DCBM is to normalize the embedding vectors to norm 1 and then apply K-means to the normalized vectors.

Example 2.10. Recall the UK faculty network in example 1.2. In examples 2.4 and 2.7, we

fit an SBM and a DCBM to the graph using their corresponding EM algorithms (algorithms 1 and 2). While all three algorithms were able to correctly identify the labels for most of the vertices, it is not immediately clear which model is the most appropriate for this graph. Analyzing the graph as a GRDPG, however, provides a visual indication for which block model is the best fit. If we wish to analyze this as an SBM or DCBM under the GRDPG framework, we can take the ASE of the adjacency matrix and see whether the embedding vectors lie around point masses or rays. Based on figure 1, we assume that this model is assortative, so $p = K$ and $q = 0$ in constructing the ASE. Because the fourth community only contains two members, we omitted it from this analysis. Figure 8 suggests that the appropriate model here is the DCBM, since each community appears to be described by a line segment. K-means clustering on the normalized embedding vectors results in a community detection accuracy of 96.7%, which is an improvement over the EM algorithms while being substantially faster and easier to implement.

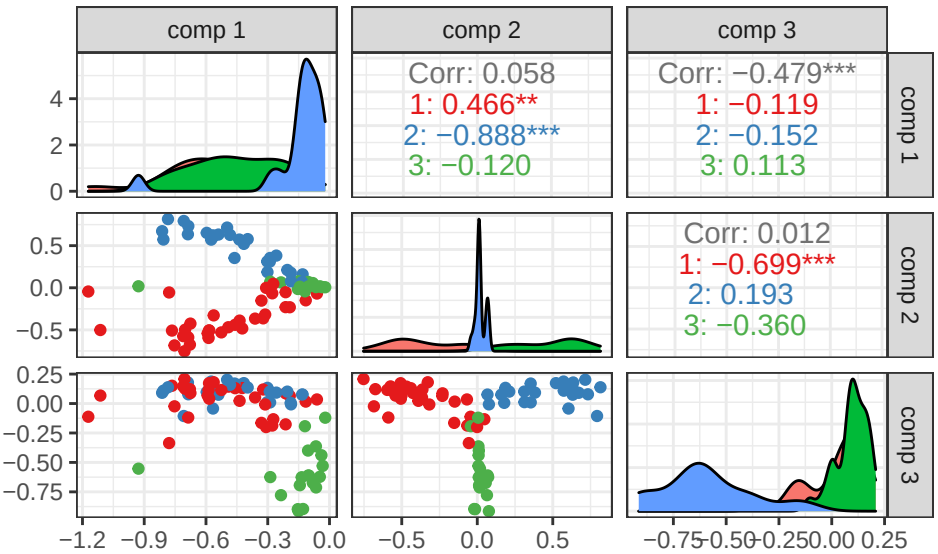

Figure 8: A three-dimensional ASE of the UK faculty network using the three most positive eigenvalues and their corresponding eigenvectors. The vertices are labeled by the school to which each faculty member belongs.

Example 2.11. Disassortative block models can also be characterized as GRDPGs.

Consider the dating network examples for the SBM (example 2.2) and DCBM (example 2.5). Since the edge probability matrices are not positive semidefinite (equation 2), these are disassortative models. More specifically, the edge probability matrix for each has one positive eigenvalue and one negative eigenvalue. Then the ASE for this model is $\hat{X} = \left[|\lambda_1|^{1/2} v_1 \mid |\lambda_n|^{1/2} v_n\right]$. A plot of the ASE of the graph in 2.2 (figure 9, left) suggests two (somewhat overlapping) ellipsoidal clusters, and fitting a GMM on the embedding results in a community detection accuracy of 96.9%, which is comparable to algorithm 1 while being simpler to implement. Likewise, a plot of the ASE of the graph in example 2.5 (figure 9, right) consists of points that lie along two lines that cross at the origin, and kernel K-means clustering using the cosine similarity kernel results in a community detection accuracy of 93.8%, which is an improvement over algorithm 2 while being simpler to implement. Also note that aside from the choice of the eigenvalues and eigenvectors (positive vs negative), the community detection algorithms for the SBM (K-means clustering or GMM on the ASE) and DCBM (kernel K-means clustering) are identical regardless of whether the model is assortative or disassortative.

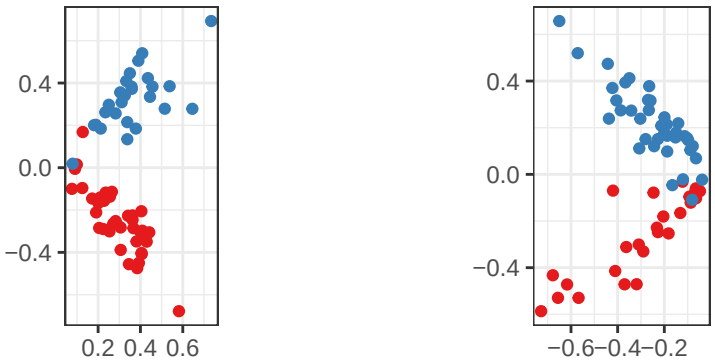

Figure 9: ASEs of the graphs in examples 2.2, which is a disassortative SBM (left), and 2.5, which is a disassortative DCBM (right). The colors correspond to vertex label. Note that the embedding vectors for the SBM cluster around two point masses while the embedding vectors for the DCBM cluster around two line segments.

The PABM is also a Bernoulli random graph, so it is also a GRDPG. The latent vectors

of the PABM when viewed as a GRDPG form K K-dimensional subspaces, which makes for an elegant parallel between the nested forms of the block models and the nested structures of the GRDPGs: The SBM consists of K zero-dimensional subspaces, and DCBM consists of K one-dimensional subspaces. We explore this connection further the next section (section 3).

3 Popularity Adjusted Block Models are Generalized Random Dot Product Graphs

In this section, we connect the PABM to the GRDPG and exploit that connection to develop algorithms for community detection and parameter estimation. In order to make the explicit connection between the PABM and the GRDPG, we make use of an alternative but equivalent definition of the PABM which parameterizes the model in terms of popularity vectors, which are collections of popularity parameters.

Remark. In a PABM, each vertex i has K popularity parameters $\lambda_{i1}, \ldots, \lambda_{iK}$, that describe its affinity toward each of the K communities. Another view of a PABM is as follows. Let $\tilde{P}$ be the matrix obtained by permuting the rows and columns of P so that the vertices are reorganized by community memberships $z_i \in \{1, 2, \ldots, K\}$ in increasing order. Denote by $\tilde{P}^{(k\ell)}$ the $n_k \times n_\ell$ submatrix of $\tilde{P}$ corresponding to the edge probabilities between vertices in communities k and ℓ. Note that $\tilde{P}^{(k\ell)} = (\tilde{P}^{(\ell k)})^\top$. Next let $\lambda^{(k\ell)} = \{\lambda_{i\ell} \colon z_i = k\} \in \mathbb{R}^{n_k}$; the elements of $\lambda^{(k\ell)}$ are the affinity parameters toward the ℓth community of all vertices in the k^{th} community. Define $\lambda^{(\ell k)}$ analogously. Then each block $\tilde{P}^{(k\ell)}$ can be written as the outer product of two vectors:

$$\tilde{P}^{(k\ell)} = \rho_n \lambda^{(k\ell)} (\lambda^{(\ell k)})^\top. \tag{5}$$

We will henceforth use the notation $A \sim \text{PABM}(\{\lambda^{(k\ell)}\}_K, \rho_n)$ to denote a random adjacency matrix A drawn from a PABM with K communities, popularity parameters $\{\lambda^{(k\ell)}\}$ and sparsity parameter ρ_n.

3.1 The Geometry of PABMs

In section 2.7, we showed how the SBM and DCBM are special cases of the GRDPG in which the latent vectors form a very particular geometry. Similarly, we now show the special geometry of the PABM when viewed as a GRDPG. For ease of exposition, and without loss

of generality, we drop the dependency on the sparsity parameter ρ_n and assume $\rho_n \equiv 1$ throughout this subsection.

Theorem 3.1 (The latent configuration of the PABM)**.** *Let $A \sim \text{PABM}(\{\lambda^{(k\ell)}\}_K)$ be an instance of a PABM with $K \geq 1$ blocks and latent vectors $\{\lambda^{(k\ell)} : 1 \leq k \leq K, 1 \leq \ell \leq K\}$. Then there exists a block diagonal matrix $X \in \mathbb{R}^{n \times K^2}$ defined by $\{\lambda^{(k\ell)}\}$ and a $K^2 \times K^2$ fixed orthonormal matrix U such that $A \sim \text{GRDPG}_{K(K+1)/2, K(K-1)/2}(\tilde{\Pi} X U)$. Here $\tilde{\Pi}$ is the permutation matrix such that $P = \tilde{\Pi}\tilde{P}\tilde{\Pi}^\top$ where the rows and columns of $\tilde{P}$ are arranged according to increasing values of the community labels (see remark 3).*

Proof. We will prove this theorem in two parts. First, for demonstration purposes, we focus on the case when $K = 2$ to build intuition. The general case of $K \geq 2$ is presented later.

For the $K = 2$ case, the proof is straightforward. We will first work with the matrix $\tilde{P}$. Note that $\tilde{P}$ has the form

$$\tilde{P} = \begin{bmatrix} P^{(11)} & P^{(12)} \\ P^{(21)} & P^{(22)} \end{bmatrix} = \begin{bmatrix} \lambda^{(11)}(\lambda^{(11)})^\top & \lambda^{(12)}(\lambda^{(21)})^\top \\ \lambda^{(21)}(\lambda^{(12)})^\top & \lambda^{(22)}(\lambda^{(22)})^\top \end{bmatrix}.$$

Now let

$$X = \begin{bmatrix} \lambda^{(11)} & \lambda^{(12)} & 0 & 0 \\ 0 & 0 & \lambda^{(21)} & \lambda^{(22)} \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1/\sqrt{2} & 1/\sqrt{2} \\ 0 & 0 & 1/\sqrt{2} & -1/\sqrt{2} \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

Then by straightforward matrix multiplication, we obtain

$$XUI_{3,1}U^\top X^\top = \begin{bmatrix} \lambda^{(11)}(\lambda^{(11)})^\top & \lambda^{(12)}(\lambda^{(21)})^\top \\ \lambda^{(21)}(\lambda^{(12)})^\top & \lambda^{(22)}(\lambda^{(22)})^\top \end{bmatrix} = \tilde{P}$$

and hence $\tilde{P}$ also corresponds to the edge probability matrix of GRDPG with latent vectors described by XU. As $P = \tilde{\Pi}\tilde{P}\tilde{\Pi}^\top$ we conclude that P has latent vectors described by $\tilde{\Pi}XU$.

It is nevertheless instructive to look at a few intermediate steps. More specifically, the product $UI_{3,1}U^\top$ yields a permutation matrix Π with fixed points at positions 1 and 4 and a cycle of order 2 swapping positions 2 and 3, i.e.,

$$\Pi = UI_{3,1}U^\top = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Furthermore, as U is orthonormal and $I_{3,1}$ is diagonal, $UI_{3,1}U^\top$ is also an eigendecomposition of Π where the fixed points of Π are mapped to the eigenvectors e_1 and e_4 while the cycles of order two are mapped to the eigenvectors $\frac{1}{\sqrt{2}}(e_2 + e_3)$ and $\frac{1}{\sqrt{2}}(e_2 - e_3)$; here e_i denote the i^{th} basis vector in $\mathbb{R}^4$. Thus, another way of decomposing the edge probability matrix is $\tilde{P} = X\Pi X^\top$ where the rows of X lie in the union of two 2-dimensional orthogonal subspaces and Π is a permutation matrix.

For the general case, we can extend $\tilde{P} = X\Pi X^\top$ to larger K. For a more concrete example of this, refer to Example 1. We once again consider $\tilde{P}$ as defined in remark 3. We first define the following matrices

$$\Lambda^{(k)} = \left[\lambda^{(k1)} \mid \cdots \mid \lambda^{(kK)}\right] \in \mathbb{R}^{n_k \times K}, \quad X = \text{blockdiag}(\Lambda^{(1)}, \ldots, \Lambda^{(K)}) \in \mathbb{R}^{n \times K^2}, \tag{6}$$

$$L^{(k)} = \text{blockdiag}(\lambda^{(1k)}, \ldots, \lambda^{(Kk)}) \in \mathbb{R}^{n \times K}, \quad Y = \left[L^{(1)} \mid \cdots \mid L^{(K)}\right] \in \mathbb{R}^{n \times K^2}. \tag{7}$$

It is then straightforward to verify that

$$XY^\top = \text{blockdiag}(\Lambda^{(1)}, \ldots, \Lambda^{(K)}) \begin{bmatrix} L_1^\top \\ \vdots \\ L_K^\top \end{bmatrix} = \begin{bmatrix} \Lambda^{(1)}(L^{(1)})^\top \\ \vdots \\ \Lambda^{(K)}(L^{(K)})^\top \end{bmatrix},$$

$$\Lambda^{(k)}(L^{(k)})^\top = \left[\lambda^{(k1)}(\lambda^{(1k)})^\top \mid \cdots \mid \lambda^{(kK)}(\lambda^{(Kk)})^\top\right] = \left[P^{(k1)} \mid P^{(k2)} \mid \cdots \mid P^{(kK)}\right].$$

We therefore have $\tilde{P} = XY^\top$. Similar to the $K = 2$ case, we also have $Y = X\Pi$ for some permutation matrix Π and hence $\tilde{P} = X\Pi X^\top$. The permutation described by Π now has K fixed points, which correspond to K eigenvalues equal to 1 with corresponding eigenvectors e_k where $k = r(K+1)+1$ for $0 \leq r \leq K-1$. It also has $\binom{K}{2}$ cycles of order 2. Each cycle corresponds to a pair of eigenvalues $\{-1, +1\}$ and a pair of eigenvectors $\{(e_s + e_t)/\sqrt{2}, (e_s - e_t)/\sqrt{2}\}$.

Let $p = K(K+1)/2$ and $q = K(K-1)/2$. We therefore have

$$\Pi = UI_{p,q}U^\top \tag{8}$$

where U is a $K^2 \times K^2$ orthogonal matrix and hence

$$\tilde{P} = XUI_{p,q}(XU)^\top. \tag{9}$$

In summary we can describe the PABM with K communities as a GRDPG with latent positions $\tilde{\Pi}XU$ and known signature $(p, q) = \left(\frac{1}{2}K(K+1), \frac{1}{2}K(K-1)\right)$. □

Example 3.1. Let A be a 3 blocks PABM with latent vectors $\{\lambda^{(k\ell)} : 1 \leq k \leq 3, 1 \leq \ell \leq 3\}$.

Using the same notation as in theorem 3.1, we can define

$$X = \begin{bmatrix} \lambda^{(11)} & \lambda^{(12)} & \lambda^{(13)} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda^{(21)} & \lambda^{(22)} & \lambda^{(23)} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda^{(31)} & \lambda^{(32)} & \lambda^{(33)} \end{bmatrix},$$

$$Y = \begin{bmatrix} \lambda^{(11)} & 0 & 0 & \lambda^{(12)} & 0 & 0 & \lambda^{(13)} & 0 & 0 \\ 0 & \lambda^{(21)} & 0 & 0 & \lambda^{(22)} & 0 & 0 & \lambda^{(23)} & 0 \\ 0 & 0 & \lambda^{(31)} & 0 & 0 & \lambda^{(32)} & 0 & 0 & \lambda^{(33)} \end{bmatrix}.$$

Then $Y = X\Pi$ and $\tilde{P} = XY^\top$ where Π is a 9×9 permutation matrix of the form

$$\Pi = \Big[e_1 \mid e_4 \mid e_7 \mid e_2 \mid e_5 \mid e_8 \mid e_3 \mid e_6 \mid e_9 \Big].$$

where e_i denotes the i^{th} basis vector in $\mathbb{R}^9$. The matrix Π corresponds to a permutation of $\{1, 2, \ldots, 9\}$ with the following decomposition.

1. Positions 1, 5, 9 are fixed.
2. There are three cycles of length 2, namely $(2, 4)$, $(3, 7)$, and $(6, 8)$.

We can thus write Π as $\Pi = UI_{6,3}U^\top$ where the first three columns of U consist of e_1, e_5, and e_9 corresponding to the fixed points, the next three columns are the eigenvectors $(e_k + e_\ell)/\sqrt{2}$, and the last three columns are the eigenvectors $(e_k - e_\ell)/\sqrt{2}$ for $(k, \ell) \in \{(2, 4), (3, 7), (6, 8)\}$.

The matrix $\tilde{P}$ is then the edge probabilities matrix for a Generalized Random Dot Product Graph whose latent positions are the rows of the matrix

$$XU = \begin{bmatrix} \lambda^{(11)} & 0 & 0 & \frac{\lambda^{(12)}}{\sqrt{2}} & \frac{\lambda^{(13)}}{\sqrt{2}} & 0 & \frac{\lambda^{(12)}}{\sqrt{2}} & \frac{\lambda^{(13)}}{\sqrt{2}} & 0 \\ 0 & \lambda^{(22)} & 0 & \frac{\lambda^{(21)}}{\sqrt{2}} & 0 & \frac{\lambda^{(23)}}{\sqrt{2}} & -\frac{\lambda^{(21)}}{\sqrt{2}} & 0 & \frac{\lambda^{(23)}}{\sqrt{2}} \\ 0 & 0 & \lambda^{(33)} & 0 & \frac{\lambda^{(31)}}{\sqrt{2}} & \frac{\lambda^{(32)}}{\sqrt{2}} & 0 & -\frac{\lambda^{(31)}}{\sqrt{2}} & -\frac{\lambda^{(32)}}{\sqrt{2}} \end{bmatrix}$$

and the latent positions for P is a permutation of the rows of XU.

3.2 Algorithms

Two inference objectives arise from the PABM:

1. Community membership identification (up to permutation).
2. Parameter estimation (estimating λ_{ik}'s).

In our methods, the data that are observed for estimation is the adjacency matrix, $A \sim \text{PABM}(\{\lambda^{(k\ell)}\}_K, \rho_n)$, along with an assumed number of communities, K. To motivate our methods, we first consider community detection and parameter estimation in the case where we know the edge probability matrix P beforehand, noting that community memberships and popularity parameters are not immediately discernible from P itself. After establishing methods for community detection and parameter estimation from P, we use the consistency property of the ASE (Sussman et al., 2012, Rubin-Delanchy et al., 2022) to demonstrate that the same methods work for A almost surely as $n \to \infty$.

3.2.1 Previous Work

Sengupta and Chen (2018) used Modularity Maximization (MM) and the Extreme Points (EP) algorithm (Le et al., 2016) for community detection and parameter estimation. They were able to show that as the sample size increases, the *proportion* of misclassified community labels (up to permutation) goes to 0.

Noroozi et al. (2019) used Sparse Subspace Clustering (SSC) (Elhamifar and Vidal, 2009) for community detection in the PABM. The SSC algorithm can be described as follows: Given $X \in \mathbb{R}^{n \times d}$ with vectors $x_i^\top \in \mathbb{R}^d$ as rows of X, the optimization problem $c_i = \arg\min_c \|c\|_1$ subject to $x_i = X^\top c$ and $c^{(i)} = 0$, where $c^{(i)}$ is the i^{th} entry of c, is solved for each $i \in [n]$. The solutions are collected into matrix $C = \begin{bmatrix} c_1 \mid \cdots \mid c_n \end{bmatrix}^\top$ to construct an affinity matrix $B = |C| + |C^\top|$. If each x_i lies exactly on one of K subspaces, B describes an

undirected graph consisting of *at least* K disjoint subgraphs, i.e., $B_{ij} = 0$ if x_i, x_j lie on different subspaces. The intuition here is that vectors that lie on the same subspace can be described as linear combinations of each other, assuming the number of vectors in the subspace is greater than the dimensionality of the subspace. Then once sparsity is enforced, for each c_i, its j^{th} element $c_i^{(j)}$ is zero if x_j belongs to a subspace that doesn't contain x_i, resulting in $B_{ij} = 0$. If X instead represents points near K subspaces with some noise, then this property may only hold approximately and a final graph partitioning step may be required (e.g., edge thresholding or spectral clustering).

In practice, due to presence of noise, SSC is often done by solving the LASSO problems

$$c_i = \arg\min_c \frac{1}{2}\|x_i - X_{-i}^\top c\|_2^2 + \vartheta\|c\|_1 \tag{10}$$

for some sparsity parameter $\vartheta > 0$. The c_i vectors are then collected into C and B as before.

Definition 3.1 (Subspace Detection Property)**.** Let $X = \begin{bmatrix} x_1 \mid \cdots \mid x_n \end{bmatrix}^\top$ be noisy points sampled from K subspaces, i.e., $x_i = y_i + z_i$ where the y_i belongs to the union of K subspaces and the z_i are noise vectors. Let $\vartheta \geq 0$ be given and let C and B be constructed from the solutions of LASSO problems as described in equation (10) with this given choice of ϑ. Then X is said to satisfy the subspace detection property with sparsity parameter ϑ if each column of C has nonzero ℓ_2 norm and $B_{ij} = 0$ whenever y_i and y_j are from different subspaces.

Remark. One common approach to show that SSC works for a noisy sample X is to show that X satisfies the subspace detection property for some choice of ϑ; recall that ϑ is the sparsity parameter for the LASSO problems in equation (10). However, this is not sufficient to guarantee that SSC perfectly recovers the underlying subspaces. More specifically, if X satisfies the subspace detection property, then B describes a graph with *at least* K disconnected subgraphs, with the ideal case being that there are exactly K subgraphs which map onto each subspace. Nevertheless it is also possible that the K subspaces are represented by $K' > K$ multiple disconnected subgraphs and we cannot, at least without a

subsequent post-processing step, recover the K subspaces directly from B; see Nasihatkon and Hartley (2011) and Liu et al. (2013) for further discussions. Therefore in practice B is usually treated as an affinity matrix and, as we allude to earlier, the rows of B are partitioned using some clustering algorithm to obtain the final clustering.

Theorem 3.1 suggests that SSC is appropriate for community detection for the PABM, provided that we observe the edge probabilities matrix P. More precisely, given the matrix $\tilde{P}$ obtained by permuting the rows and columns of P as described in remark 3 we can recover XU up to some non-identifiability indefinite orthogonal transformation Q. Then using results from Soltanolkotabi and Candés (2012), it can be easily shown that the subspace detection property holds for XU. Indeed, the columns of XU from different communities correspond to mutually orthogonal subspaces. This then implies that the subspace detection property also holds for XUQ for all invertible transformation Q and hence the subspace detection property also holds for $\tilde{\Pi}XUQ$ for any $n \times n$ permutation matrix $\tilde{\Pi}$.

However, because we do not observe P but rather only the noisy adjacency matrix $A \sim \text{BernoulliGraph}(P)$, the natural approach then is to perform SSC on the rows of the spectral embedding of A, since the embedding of P consists of K subspaces (theore 3.1), and so the embedding of A will lie approximately on the K subspaces. We will show in theorem 3.4 that, with probability converging to one as $n \to \infty$, the rows of the ASE of A also satisfy the subspace detection property. Theorem 3.4 builds upon existing work by Rubin-Delanchy et al. (2022) who describe the convergence behavior of the ASE of A to that of $\tilde{\Pi}XU$, and Wang and Xu (2016) who show the necessary conditions for the subspace detection property to hold in noisy cases where the points lie near subspaces. Finally we emphasize that while Noroozi et al. (2019) also considered the use of SSC for community recovery in PABM, they instead applied SSC to the rows of A itself, foregoing the embedding step altogether. It is however much harder to show that the rows of A satisfy the subspace detection property and thus, to the best of our knowledge, there is currently no consistency result regarding the

application of SSC to the rows of A.

3.2.2 Algorithms for Community Detection

We previously stated in theorem 3.1 one possible set of latent positions that result in the edge probability matrix of a PABM, namely $P = \tilde{\Pi}(XU)I_{p,q}(XU)^\top\tilde{\Pi}^\top$ where X is block diagonal and $\tilde{\Pi}$ is a permutation matrix.

Furthermore, the explicit form of XU represents points in $\mathbb{R}^{K^2}$ such that points within each community lie on K-dimensional orthogonal subspaces, i.e. $\langle U^\top x_i, U^\top x_j\rangle = 0$ whenever i and j are in different communities. Thus if we have (or can estimate) XU directly, then both the community detection and parameter identification problem are trivial because U is orthonormal and fixed for each value of K. However, direct identification or estimation of XU is possibly difficult due to the non-identifiability of XU (see remark 2.6) when we are given only P. More specifically, suppose we find a matrix $Y \in \mathbb{R}^{n\times K^2}$ such that $P = YI_{p,q}Y^\top$. Then it is generally the case that $Y = \tilde{\Pi}XUQ$ for some indefinite orthogonal matrix $Q \in \mathbb{O}(p,q)$. However since Q is not necessarily an orthogonal matrix and hence, if y_i denote the i^{th} row of Y, then $\langle U^\top x_i, U^\top x_j\rangle \neq \langle y_i, y_j\rangle$. This prevents us from transferring the orthogonality property of XU directly to Y.

Nevertheless by using the special geometric structure of X we can circumvent the non-identifiability of Y and XU by using instead the rows of the matrix V of eigenvectors (corresponding to the non-zero eigenvalues) of P. In particular V is identifiable up to orthogonal transformations and furthermore, due to the block diagonal structure of X, the rows of V also lie on K distinct orthogonal subspaces and hence $v_i^\top v_j = 0$ whenever $z_i \neq z_j$.

Theorem 3.2. *Let $P = VDV^\top$ be the spectral decomposition of the edge probability matrix. Let $B = nVV^\top$. Assume $\lambda_{iz_i} > 0$ for each $i \in [n]$, i.e., each vertex's popularity parameter to its own community is nonzero. Then $B_{ij} = 0$ if and only if vertices i and j are in different communities.*

Proof. We first show that $VV^\top = \tilde{\Pi}X(X^\top X)^{-1}X^\top\tilde{\Pi}^\top$ where X is defined as in equation (6). Indeed, by Theorem 2, $P = \tilde{\Pi}XUI_{p,q}U^\top X^\top\tilde{\Pi}$ for $p = K(K+1)/2$ and $q = K(K-1)/2$. The eigendecomposition $P = VDV^\top$ also yields $P = V|D|^{1/2}I_{p,q}|D|^{1/2}V^\top$ where $|\cdot|^{1/2}$ is applied entry-wise. Now let $Y = \tilde{\Pi}XU$ and $\tilde{Y} = V|D|^{1/2}$; note that Y and $\tilde{Y}$ both have full column ranks. Because $P = YI_{p,q}Y^\top = \tilde{Y}I_{p,q}\tilde{Y}^\top$, we have

$$Y = \tilde{Y}I_{p,q}\tilde{Y}^\top Y(Y^\top Y)^{-1}I_{p,q}.$$

Let $Q = I_{p,q}\tilde{Y}^\top Y(Y^\top Y)^{-1}I_{p,q}$ and note that $Y = \tilde{Y}Q$. We then have

$$\begin{aligned} Q^\top I_{p,q}Q &= I_{p,q}(Y^\top Y)^{-1}Y^\top\tilde{Y}I_{p,q}I_{p,q}I_{p,q}\tilde{Y}^\top Y(Y^\top Y)^{-1}I_{p,q} \\ &= I_{p,q}(Y^\top Y)^{-1}Y^\top YI_{p,q}Y^\top Y(Y^\top Y)^{-1}I_{p,q} = I_{p,q} \end{aligned}$$

and hence Q is an indefinite orthogonal matrix.

Let $R = UQ|D|^{-1/2}$ and note that $V = \tilde{\Pi}XR$. Because R is invertible, we can write

$$\tilde{\Pi}X(X^\top X)^{-1}X^\top\tilde{\Pi}^\top = \tilde{\Pi}XR(R^\top X^\top XR)^{-1}R^\top X^\top\tilde{\Pi}^\top.$$

Furthermore, as V has orthonormal columns, $R^\top X^\top XR = V^\top\tilde{\Pi}\tilde{\Pi}^\top V = V^\top V = I$. We thus conclude

$$\tilde{\Pi}X(X^\top X)^{-1}X^\top\tilde{\Pi}^\top = V(V^\top V)^{-1}V^\top = VV^\top$$

as desired.

To complete the proof of theorem 3.2, recall that X is block diagonal with each block corresponding to one community, and hence $X(X^\top X)^{-1}X^\top$ is also a block diagonal matrix with each block corresponding to a community. As $B = nVV^\top = n\tilde{\Pi}X(X^\top X)^{-1}X^\top\tilde{\Pi}^\top$, we conclude that $B_{ij} = 0$ whenever vertices i and j belong to different communities. □

Theorem 3.2 provides perfect community detection from P. More specifically, let $|B|$ be

Algorithm 4: Orthogonal Spectral Clustering.

Data: Adjacency matrix A, number of communities K

Result: Community assignments $z_1, ..., z_n \in \{1, ..., K\}$

1 Compute the eigenvectors of A that correspond to the $K(K+1)/2$ most positive eigenvalues and $K(K-1)/2$ most negative eigenvalues. Construct V using these eigenvectors as its columns.

2 Compute $B = |nVV^\top|$, applying $|\cdot|$ entry-wise.

3 Construct graph G using B as its similarity matrix.

4 Partition G into K disconnected subgraphs (e.g., using edge thresholding or spectral clustering).

5 Map each partition to the community labels $1, ..., K$.

the affinity matrix for graph G', where $|\cdot|$ is applied entry-wise. Then G' consists of exactly K disjoint subgraphs, as G' has no edges between communities. All that is left to identify the communities is to assign each subgraph a distinct community label. In practice, we do not observe P and instead only observe the noisy $A \sim \text{BernoulliGraph}(P)$. A natural approach is then to use the affinity matrix $\hat{B} = n\hat{V}\hat{V}^\top$ where $\hat{V}$ is the matrix of eigenvectors (corresponding to the largest eigenvalues in modulus) of A. The resulting procedure, named Orthogonal Spectral Clustering, is presented in algorithm 4. The following result leverages existing theoretical properties of ASE for estimating of latent positions in a GRDPG (Rubin-Delanchy et al., 2022) to show that $\hat{B}$ converges almost surely to B; in particular $\hat{B}_{ij} \xrightarrow{a.s.} 0$ for each pair (i, j) in different communities.

Theorem 3.3. *Assume the setting of theorem 3.2. Let $\hat{B}$ with entries $\hat{B}_{ij}$ be the affinity matrix obtained from OSC as described in algorithm 4. Then for $n\rho_n = \omega(\log^4 n)$, we have*

$$\max_{i,j} |\hat{B}_{ij} - B_{ij}| = O\left(\frac{\log n}{\sqrt{n\rho_n}}\right) \tag{11}$$

with high probability. In particular $\hat{B}_{ij} - B_{ij} \xrightarrow{\text{a.s.}} 0$ where the convergence is uniform over all

i, j. Hence for all pairs (i, j) in different communities we have $\hat{B}_{ij} \xrightarrow{\text{a.s.}} 0$, while for all pairs (i, j) in the same community, $\liminf_{n\to\infty} |\hat{B}_{ij}| > 0$ almost surely.

Theorem 3.3 guarantees that for any $\epsilon > 0$, the number of edges of $\hat{B}$ between vertices of different communities that are larger than ϵ converges to zero with probability converging to one as n increases. We can always find an $\epsilon > 0$ such that $\hat{B}_{ij} > \epsilon$ with probability converging to one as n increases. Thus, by using $\hat{B}$, we can perfectly recover all the latent community assignments $z_1, z_2, \ldots, z_n$, i.e., the number of misclustered vertices is zero asymptotically almost surely. We note that theorem 3.3 is stronger than existing results in the literature; in particular theorem 1 of Sengupta and Chen (2018) (the paper that originally introduces the PABM model) only guarantees that the *proportion* of misclustered vertices converges to 0 as $n \to \infty$. Furthermore theorem 1 of Sengupta and Chen (2018) also requires the sparsity parameter ρ_n to satisfies $n\rho_n^2 = \omega(\log^2 n)$ which is a considerably stronger assumption than the assumption $n\rho_n = \omega(\log^4 n)$ used in theorem 3.3. Indeed, $n\rho_n^2 = \omega(\log^2 n)$ implies $n\rho_n = \omega(n^{1/2})$. We emphasize that the assumption $n\rho_n = \omega(\log^c n)$ for some constant $c > 1$ is commonly used in the context of graph estimation using spectral methods.

Theorems 3.1, 3.2, and 3.3 also provide a natural path toward using SSC for community detection. In particular we established in theorem 3.1 that an ASE of the edge probability matrix P can be constructed from a latent vector configuration consisting of orthogonal subspaces. Theorem 3.2 shows how this property can also be recovered from the eigenvectors of P. Then theorem 3.3 shows that, by replacing P with A, the rows of $\hat{V}$ also lie on asymptotically orthogonal subspaces. Motivated by theorem 3.3, theorem 3.4 below shows that the subspace detection property also holds for the rows of $\sqrt{n}\hat{V}$.

Theorem 3.4. *Let P describe the edge probability matrix of the PABM with n vertices, and let $A \sim \text{Bernoulli}(P)$. Let $\hat{V}$ be the matrix of eigenvectors of A corresponding to the K^2 largest eigenvalues in modulus. Then for any $\epsilon > 0$, there exists a choice of $\vartheta > 0$ and $N \in \mathbb{N}$*

Algorithm 5: Sparse Subspace Clustering using LASSO.

Data: Adjacency matrix A, number of communities K, hyperparameter λ

Result: Community assignments $z_1, ..., z_n \in \{1, ..., K\}$

Find V, the matrix of eigenvectors of A corresponding to the $K(K+1)/2$ most positive and the $K(K-1)/2$ most negative eigenvalues.

Normalize $V \leftarrow \sqrt{n}V$.

for $i = 1, ..., n$ **do**

  Assign $v_i^\top$ as the i^{th} row of V. Assign $V_{-i} = \left[v_1 \mid \cdots \mid v_{i-1} \mid v_{i+1} \mid \cdots \mid v_n\right]^\top$.

  Solve the LASSO problem $c_i = \arg\min_\beta \frac{1}{2}\|v_i - V_{-i}\beta\|_2^2 + \lambda\|\beta\|_1$.

  Assign $\tilde{c}_i = (c_i^{(1)}, \ldots, c_i^{(i-1)}, 0, c_i^{(i)}, \ldots, c_i^{(n-1)})^\top$ such that the superscript is the index of $\tilde{c}_i$.

end

Assign $C = \left[\tilde{c}_1 \mid \cdots \mid \tilde{c}_n\right]$.

Compute the affinity matrix $B = |C| + |C^\top|$.

Construct graph G using B as its similarity matrix.

Partition G into K disconnected subgraphs (e.g., using edge thresholding or spectral clustering).

Map each partition to the community labels $1, ..., K$.

such that for all $n \geq N$, $\sqrt{n}\hat{V}$ obeys the subspace detection property with probability at least $1 - \epsilon$.

3.2.3 Algorithm for Parameter Estimation

For ease of exposition we now assume in this subsection that the edge probability matrix P for the PABM had been arranged so that the rows and columns are organized by community so that $\tilde{P} = P$ (see remark 3). Then the $k\ell^{\text{th}}$ block is an outer product of two vectors, i.e., $P^{(k\ell)} = \lambda^{(k\ell)}(\lambda^{(\ell k)})^\top$. Therefore, given $P^{(k\ell)}$, $\lambda^{(k\ell)}$ and $\lambda^{(\ell k)}$ are solvable up to multiplicative constant using singular value decomposition. More specifically, let $P^{(k\ell)} = (\sigma^{(k\ell)})^2 u^{(k\ell)}(v^{(k\ell)})^\top$ be the singular value decomposition of $P^{(k\ell)}$ where $u^{(k\ell)} \in \mathbb{R}^{n_k}$

Algorithm 6: PABM parameter estimation.

Data: Adjacency matrix A, community assignments $1, ..., K$

Result: PABM parameter estimates $\{\hat{\lambda}^{(k\ell)}\}_K$.

Arrange the rows and columns of A by community such that each $A^{(k\ell)}$ block consists of estimated edge probabilities between communities k and l.

for $k, \ell = 1, ..., K,\ k \leq \ell$ **do**

  Compute $A^{(k\ell)} = U\Sigma V^\top$, the SVD of the $k\ell^{th}$ block.

  Assign $u^{(k\ell)}$ and $v^{(k\ell)}$ as the first columns of U and V. Assign $(\sigma^{(k\ell)})^2 \leftarrow \Sigma_{11}$.

  Assign $\hat{\lambda}^{(k\ell)} \leftarrow \pm\sigma^{(k\ell)}u^{(k\ell)}$ and $\hat{\lambda}^{(\ell k)} \leftarrow \pm\sigma^{(k\ell)}v^{(k\ell)}$.

end

and $v^{(k\ell)} \in \mathbb{R}^{n_\ell}$ are vectors and $\sigma^{(k\ell)}$ is a scalar. Then $\rho_n^{1/2}\lambda^{(k\ell)} = s_1 u^{(k\ell)}$ and $\rho_n^{1/2}\lambda^{(\ell k)} = s_2 v^{(k\ell)}$ for unidentifiable $s_1 s_2 = (\sigma^{(k\ell)})^2$. Because each $\lambda^{(k\ell)}$ is not strictly identifiable, we instead estimate each $\tilde{\lambda}^{(k\ell)} = \sigma^{(k\ell)}u^{(k\ell)}$. Given the adjacency matrix A instead of edge probability matrix P, we can simply use plug-in estimators by taking the SVD of each $A^{(k\ell)}$ to obtain $\hat{\lambda}^{(k\ell)} = \hat{\sigma}^{(k\ell)}\hat{u}^{(k\ell)}$ using the largest singular value of A and its corresponding singular vectors.

Theorem 3.5. *Let each $\tilde{\lambda}^{(k\ell)}$ be the popularity vector derived from its corresponding $P^{(k\ell)}$ and let $\hat{\lambda}^{(k\ell)}$ be its estimate obtained from $A^{(k\ell)}$ using algorithm 6. Then if $n\rho_n = \omega(\log^4 n)$,*

$$\max_{k,\ell\in\{1,...,K\}} \|\hat{\lambda}^{(k\ell)} - \tilde{\lambda}^{(k\ell)}\|_\infty = O\left(\frac{\log n_k}{\sqrt{n_k}}\right) \tag{12}$$

with high probability. Here $\|\cdot\|_\infty$ denotes the ℓ_∞ norm of a vector. Let $\hat{\Lambda}$ be the matrix

$$\hat{\Lambda} = \begin{bmatrix} \hat{\lambda}^{(11)} & \hat{\lambda}^{(12)} & \dots & \hat{\lambda}^{(1K)} \\ \hat{\lambda}^{(21)} & \hat{\lambda}^{(22)} & \dots & \hat{\lambda}^{(2K)} \\ \vdots & \vdots & \cdots & \vdots \\ \hat{\lambda}^{(K1)} & \hat{\lambda}^{(K2)} & \dots & \hat{\lambda}^{(KK)} \end{bmatrix}$$

and let $\hat{P} = \hat{X}UI_{p,q}U^\top\hat{X}^\top$*, where* $\hat{X}$ *is defined from* $\hat{\Lambda}$ *and* U *is defined from* K *as in theorem 3.1. Equation* (12) *then implies*

$$\frac{1}{n}\|\rho_n^{-1}\hat{P} - \rho_n^{-1}P\|_F = O((n\rho_n)^{-1/2}), \quad \max_{ij}|\rho_n^{-1}\hat{P}_{ij} - \rho_n^{-1}P_{ij}| = O((n\rho_n)^{-1/2}) \tag{13}$$

with high probability.

Equation (12) guarantees that $n^{-1/2}\|\rho_n^{-1/2}\hat{\Lambda} - \Lambda\|_F = O((n\rho_n)^{-1/2})$. Equation (13) then guarantees that the mean square error for $\rho_n^{-1}(\hat{P} - P)$ converges to 0 almost surely and furthermore the entries of $\rho_n^{-1}\hat{P}$ converge uniformly to the entries of $\rho_n^{-1}P$; recall that $\rho_n^{-1}P_{ij} = \lambda_{iz_j}\lambda_{jz_i}$. We note that these results are stronger than existing results in Sengupta and Chen (2018); for example theorem 2 in Sengupta and Chen (2018) only guarantees $n^{-1/2}\|\rho_n^{-1/2}\hat{\Lambda} - \Lambda\|_F = o(1)$ as $n \to \infty$.

3.3 Simulation Study

For each simulation, community labels are drawn from a multinomial distribution, the popularity vectors $\{\lambda^{(k\ell)}\}_K$ are drawn from two types of joint distributions depending on whether $k = \ell$ or $k \neq \ell$. The edge probability matrix P is constructed using the popularity vectors and finally the adjacency matrix A is drawn $A \sim \text{Bernoulli}(P)$. OSC (algorithm 4) is then used for community detection, and this method is compared against (1) SSC using the spectral embedding $\hat{V}$ (algorithm 5), (2) SSC using the rows of the observed adjacency matrix A as is done in Noroozi et al. (2019) and (3) modularity maximization (MM) as is done in Sengupta and Chen (2018). We denote the two SSC implementations using the rows of A and using the spectral embedding of A as SSC-A and SSC-ASE, respectively. The parameters ϑ that controls the sparsity for SSC-A and SSC-ASE were chosen via a preliminary cross-validation experiment. In practice, ϑ that guarantee SDP (if it is possible for the particular simulated data) often result in more than K disconnected subgraphs, so a smaller ϑ that does not guarantee SDP was chosen, and the final clustering step of SSC-A

and SSC-ASE was done by fitting a Gaussian mixture model to the normalized Laplacian eigenmap embeddings (Belkin and Niyogi, 2003) of the affinity matrix B. We also estimate the latent popularity vectors $\{\lambda^{(k\ell)}\}$ by assuming that the true community labels are known and then apply algorithm 6, and we compare this estimation method against an MLE-based estimator as described in Noroozi et al. (2019) and Sengupta and Chen (2018).

Modularity maximization is NP-hard, so Sengupta and Chen (2018) used the extreme points (EP) algorithm (Le et al. (2016)) as a greedy relaxation of the optimization problem; the EP algorithm has a running time of $O(n^{K-1})$ where n is the number of vertices in the graph and K is the number of communities. For these simulations we instead replace the EP algorithm with the Louvain algorithm for modularity maximization, as the implementation of the EP algorithm in Sengupta and Chen (2018) is too computationally expensive for $K > 2$. For $K = 2$, it was verified that the Louvain algorithm produces comparable results to EP-MM.

For comparing methods, we define the community detection error as:

$$L_c(\hat{\sigma}, \sigma; V) = \min_{\pi} \sum_i I(\pi \circ \hat{\sigma}(v_i) = \sigma(v_i))$$

where $\sigma(v_i)$ is the true community label of vertex v_i, $\hat{\sigma}(v_i)$ is the predicted label of v_i, and π is a permutation operator. This is effectively the "misclustering count" of clustering function $\hat{\sigma}$.

For parameter estimation, because the popularity parameters $\{\lambda_{ik}\}$ are unidentifiable, we instead estimate the edge probabilities $P_{ij} = \lambda_{iz_j}\lambda_{jz_i}$ via the quantities $\hat{P}_{ij} = \hat{\lambda}_{iz_j}\hat{\lambda}_{jz_i}$. The parameter estimation error is then given by the normalized Frobenius norm of P divided by the number of vertices, i.e.,

$$\text{RMSE}(\hat{P}, P) = \frac{1}{n}\|\hat{P} - P\|_F.$$

We also note that unlike the MLE-based method (Sengupta and Chen, 2018), the ASE method in algorithm 6 can be trivially modified so as to not require the community labels if we are only interested in estimating P. More specifically we first compute the ASE $\hat{Z}$ of A (see definition 2.6) and then compute $\hat{P} = \hat{Z} I_{p,q} \hat{Z}^\top$. The resulting estimate $\hat{P}$ will have the same convergence rate as that given in equation (13).

3.3.1 Balanced Communities

In each simulation, community labels $z_1, ..., z_n$ were drawn from a multinomial distribution with mixture parameters $\{\alpha_1, ..., \alpha_K\}$, then $\{\lambda^{(k\ell)}\}_K$ according to the drawn community labels, P was constructed using the drawn $\{\lambda^{(k\ell)}\}_K$, and A was drawn from P.

For these examples, we set the following parameters:

- Number of vertices $n = 128, 256, 512, 1024, 2048, 4096$
- Number of underlying communities $K = 2, 3, 4$
- Mixture parameters $\alpha_k = 1/K$ for $k = 1, ..., K$, (i.e., each community label has an equal probability of being drawn)
- Community labels $z_k \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$
- Within-group popularities $\lambda^{(kk)} \overset{\text{iid}}{\sim} \text{Beta}(2, 1)$
- Between-group popularities $\lambda^{(k\ell)} \overset{\text{iid}}{\sim} \text{Beta}(1, 2)$ for $k \neq \ell$

Fifty simulations were performed for each combination of n and K. The results for community recovery and parameter estimations are presented in figures 10 and 11, respectively.

Figure 10 shows that OSC recovers the community perfectly as n increases, i.e., the number of mislabeled vertices goes to 0. The performance of SSC-ASE is comparable to OSC for $K \geq 3$ but is noticeably worse when $K = 2$. Similarly, SSC on both the embedding and on the adjacency matrix produces similar trends for $K > 2$. The difference in performance between SSC-A and SSC-ASE for $K = 2$ can be attributed to the final spectral

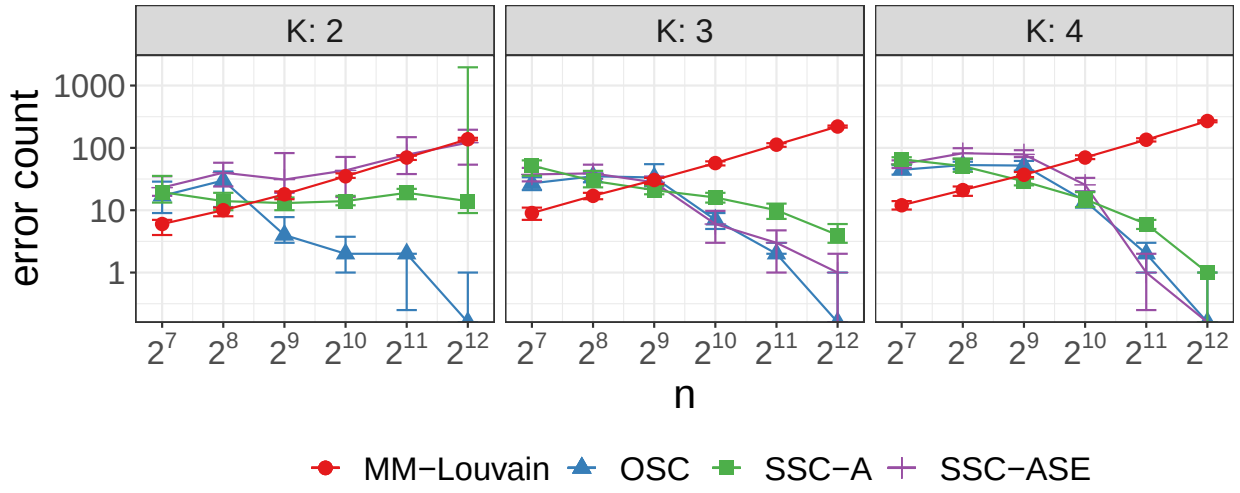

Figure 10: Median and IQR of community detection error. Communities are approximately balanced. Simulations were repeated 50 times for each sample size.

clustering step of the affinity matrix. While the subspace detection property is guaranteed for large n, in our simulations, setting the sparsity parameter ϑ to the required value usually resulted in more than K disconnected subgraphs in the affinity matrix $\hat{B}$. We instead chose a smaller sparsity parameter, necessitating a final clustering step. A GMM was fit to the normalized Laplacian eigenmap of $\hat{B}$, but visual inspection suggests that the communities are not distributed as a mixture of Gaussians in the eigenmap. A different choice of mixture distribution may result in better performance.

Given ground truth community labels, figure 11 shows that algorithm 6 and the MLE-based plug-in estimators perform comparably, with root mean square error decaying at rate approximately $n^{-1/2}$ as n increases.

3.3.2 Imbalanced Communities

Simulations performed in this section are the same as those in the previous section with the exception of the mixture parameters $\{\alpha_1, ..., \alpha_K\}$ used to draw community labels from the multinomial distribution. For these examples, we set the following parameters:

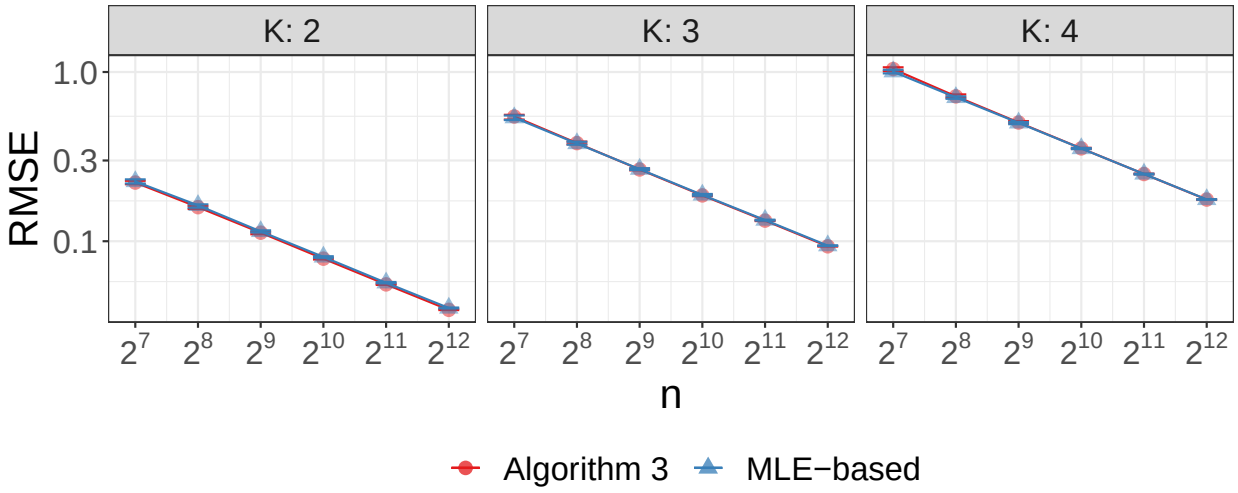

Figure 11: Median and IQR RMSE for edge probability matrices reconstructed from the outputs of algorithm 6 (red) compared against outputs of an MLE-based method (blue) proposed in Sengupta and Chen (2018). Simulations were repeated 50 times for each sample size. Communities were drawn to be approximately balanced.

- Number of vertices $n = 128, 256, 512, 1024, 2048, 4096$
- Number of underlying communities $K = 2, 3, 4$
- Mixture parameters $\alpha_k = \frac{k^{-1}}{\sum_{\ell=1}^{K} \ell^{-1}}$ for $k = 1, ..., K$
- Community labels $z_k \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$
- Within-group popularities $\lambda^{(kk)} \overset{\text{iid}}{\sim} \text{Beta}(2, 1)$
- Between-group popularities $\lambda^{(k\ell)} \overset{\text{iid}}{\sim} \text{Beta}(1, 2)$ for $k \neq \ell$

Fifty simulations were performed for each combination of n and K. The results for community recovery and parameter estimations are presented in figures 12 and 13, respectively.

From figure 12 we once again see that the number of mislabeled vertices trending to 0 for OSC. The performance of SSC-ASE is comparable to that of OSC for $K > 2$ but is worse when $K = 2$. Figure 13 indicates that the parameter estimation error also decays at rate $n^{-1/2}$ similar to that in the balanced communities setting.

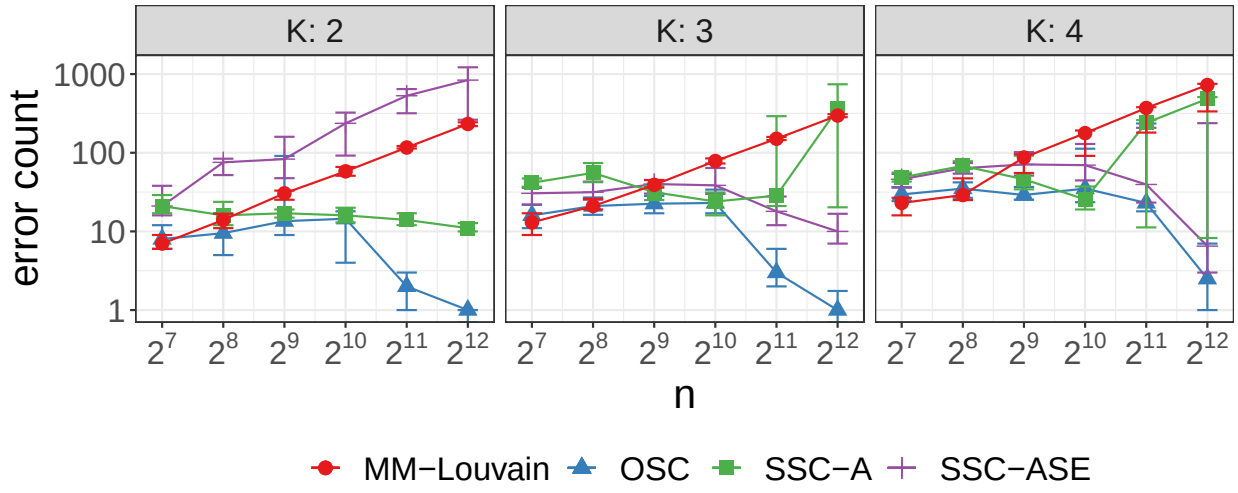

Figure 12: Median and IQR of community detection error. Communities are imbalanced. Simulations were repeated 50 times for each sample size.

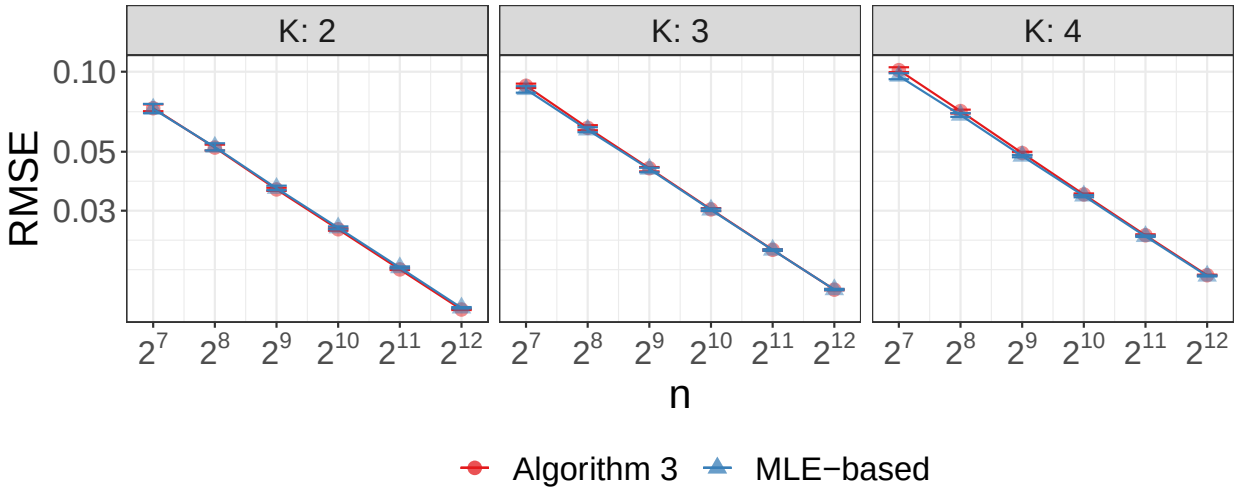

Figure 13: Median and IQR RMSE of edge probabilities derived from the outputs of Algorithm 6 (red) compared against an MLE-based method (blue) described in Sengupta and Chen (2018). Simulations were repeated 50 times for each sample size. Communities were drawn to be imbalanced.

3.3.3 Disassortative Models

Rubin-Delanchy et al. (2022) demonstrated the power of applying GRDPG-based approaches to disassortative block models. Likewise, demonstrate that GRDPG-based algorithms OSC and SSC-ASE perform well on disassortative PABMs. Simulations performed in this section are the same as those in section 3.3.1 with the exception of the distributions from which within and between-group popularity parameters are drawn. Here, we draw these parameters such that the expected value is 1/3 for the within-group popularity parameters and 2/3 for the between-group popularity parameters:

- Number of vertices $n = 128, 256, 512, 1024, 2048, 4096$
- Number of underlying communities $K = 2, 3, 4$
- Mixture parameters $\alpha_k = \frac{k^{-1}}{\sum_{\ell=1}^{K} \ell^{-1}}$ for $k = 1, ..., K$
- Community labels $z_k \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$
- Within-group popularities $\lambda^{(kk)} \overset{\text{iid}}{\sim} \text{Beta}(1, 2)$
- Between-group popularities $\lambda^{(k\ell)} \overset{\text{iid}}{\sim} \text{Beta}(2, 1)$ for $k \neq \ell$

OSC, SSC-ASE, and SSC-A perform similarly on disassortative PABMs compared to the simulations in sections 3.3.1 and 3.3.2 (figures 14 and 15).

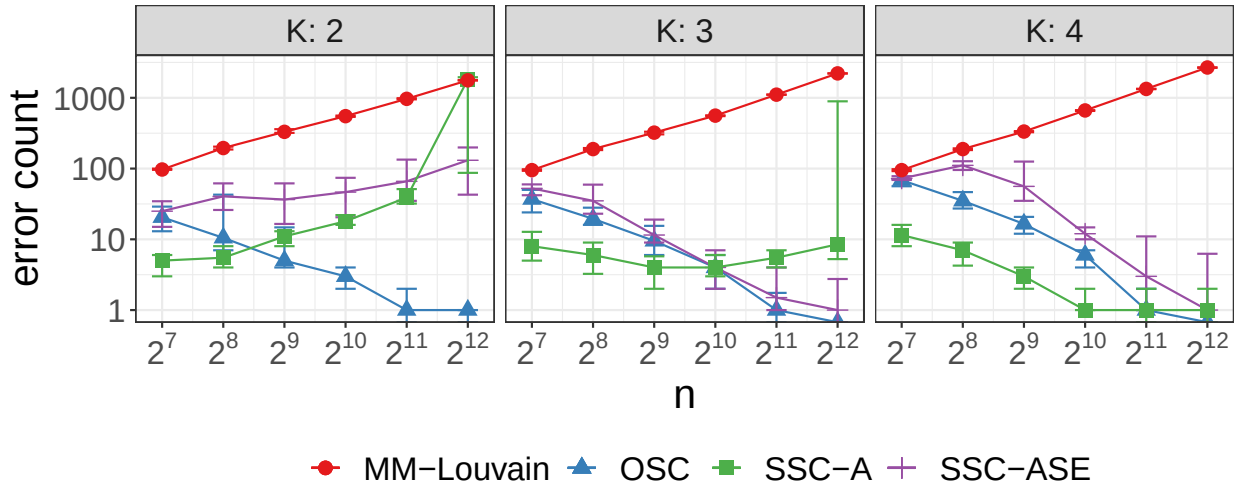

Figure 14: Median and IQR of community detection error. Communities are approximately balanced. Edge probabilities were drawn to be disassortative. Simulations were repeated 50 times for each sample size.

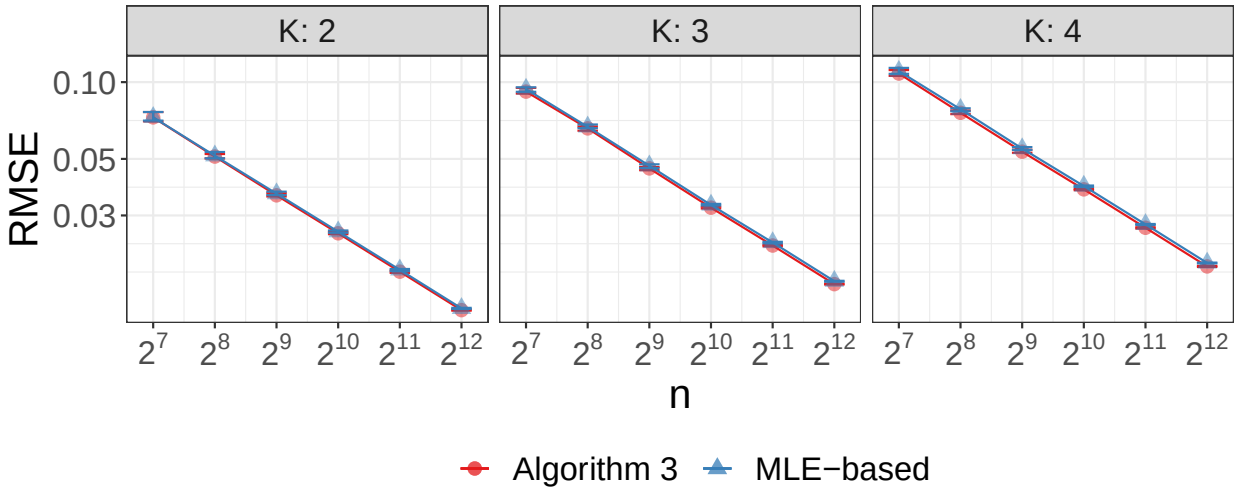

Figure 15: Median and IQR RMSE of edge probabilities derived from the outputs of algorithm 6 (red) compared against an MLE-based method (blue) described in Sengupta and Chen (2018). Simulations were repeated 50 times for each sample size. Communities were drawn to be approximately balanced. Edge probabilities were drawn to be disassortative.

Remark. While we call the models in this section "disassortative", it is our view that the assortative/disassortative distinction is not applicable to PABMs due to its flexibility. Unlike the SBM and DCBM, in the PABM, each vertex is free to have a higher affinity to its own community or to other communities, independent of each other. In other words, within the same community, v_i may have a larger popularity parameter to its own community whereas v_j may have a larger popularity parameter to a different community. Furthermore, when viewed as GRDPGs, the assortativity or disassortativity of SBMs and DCBMs affects whether P is positive semidefinite, which affects ASE-based approaches to analyzing the graph (Rubin-Delanchy et al., 2022), but in the full rank PABM, P is never positive semidefinite and will always have $K(K+1)/2$ positive eigenvalues and $K(K-1)/2$ negative eigenvalues.

3.4 Applications

In the first example, we applied OSC (algorithm 4) to the Leeds Butterfly dataset (Wang et al., 2018) consisting of visual similarity measurements among 832 butterflies across 10 species. The graph was modified to match the example from Noroozi et al. (2019): Only the $K = 4$ most frequent species were considered, and the similarities were discretized to $\{0, 1\}$ via thresholding. Figure 16 shows a sorted adjacency matrix sorted by the resultant clustering.

Comparing against the ground truth species labels, Noroozi et al. (2019) report that SSC on the adjacency matrix achieves an adjusted Rand index of 73% in their implementation, whereas OSC achieves 92% and SSC on the ASE achieves 96%.

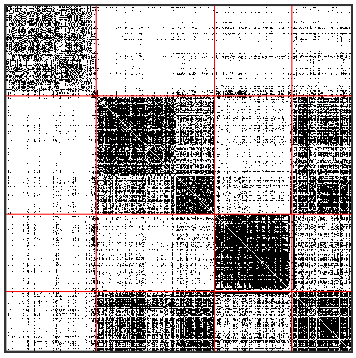

Figure 16: Adjacency matrix of the Leeds Butterfly dataset after sorting by the clustering outputted by OSC.

In the second example (figure 17 and table 2), we applied OSC to the British MPs Twitter network (Greene and Cunningham, 2013), the Political Blogs network (Adamic and Glance, 2005), and the DBLP network (Gao et al., 2009, Ji et al., 2010). For this data analysis, we subsetted the data as described in Sengupta and Chen (2018) for their analysis of the same networks. Our methods slightly underperformed compared to modularity maximization, although performance is comparable. The run time of OSC is however much smaller than that of modularity maximization.

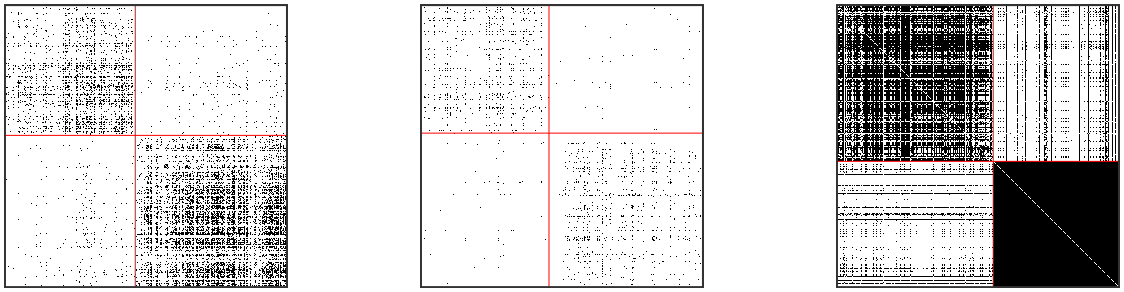

Figure 17: Adjacency matrices of (from left to right) the British MPs, Political Blogs, and DBLP networks after sorting by the clustering outputted by OSC.

Network	MM	SSC-ASE	OSC
British MPs	0.003	0.012	0.006
Political Blogs	0.050	0.187	0.062
DBLP	0.028	0.072	0.059

Table 2: Community detection error rates on the British MPs Twitter, Political Blogs, and DBLP networks using modularity maximization, sparse subspace clustering, and OSC.

In the third example (figure 18 and table 3), we analyzed the Karantaka villages data studied by Banerjee et al. (2013). We chose the `visitgo` networks from villages 12, 31, and 46 at the household level. In these networks, each node is a household and each edge is an interaction between members of pairs of households. The label of interest is the religious affiliation. The networks were truncated to religions "1" and "2", and vertices of degree 0 were removed. The villages were chosen based on there being an adequate number of nodes between households within each religion.

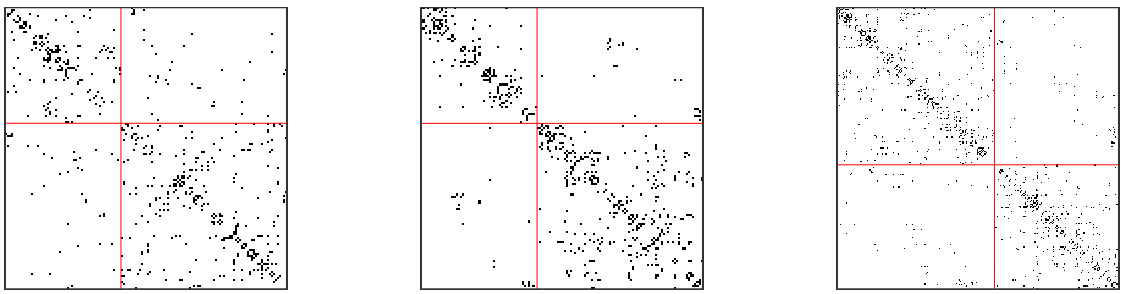

Figure 18: Adjacency matrix of the Karnataka villages data, arranged by the clustering produced by OSC (left). The villages studied here are, from left to right, 12, 31, and 46.

Network	MM	SSC-ASE	OSC
Village 12	0.270	0.291	0.227
Village 31	0.125	0.059	0.051
Village 46	0.052	0.069	0.056

Table 3: Community detection error rates for identifying household religion.

4 Generalized Random Dot Product Graphs with Nonlinear Community Structure

4.1 Community Detection as Clustering in the Latent Space

As established in section 2.7, it is possible to represent any Bernoulli random graph as a generalized random dot product graph. Whether this is useful for community detection in block models depends entirely on the configuration of the latent vectors. In the case of the Erdös-Rényi model, SBM, DCBM, and PABM, each community forms a linear structure, i.e., a subspace, and this rigid structure can be exploited for community detection. In this section, we explore GRDPGs with community-wise parameterized nonlinear latent structures. We begin with a motivating example (example 4.1).

Example 4.1 (Macaque visuotactile brain areas and connections)**.** Négyessy et al. (2006) constructed a graph based on observed 45 brain regions and 463 connections of the macaque monkey. The regions were then labeled by whether the brain region corresponds to a visual function or a sensorimotor function. This graph is visualized in figure 19. The data are available in the `igraphdata` package (Csardi and Nepusz, 2006).

Treating this graph as an RDPG with a latent space dimension of 2, we decomposed the adjacency matrix to construct its ASE (figure 20). The embedding vectors, which approximate the latent vectors, appear to approximately lie along two curves that intersect at the origin, with each curve corresponding to a label (visual vs sensorimotor). While the ASE suggests a community-wise structure in the latent space, the SBM, DCBM, and PABM are not suitable models for these data.

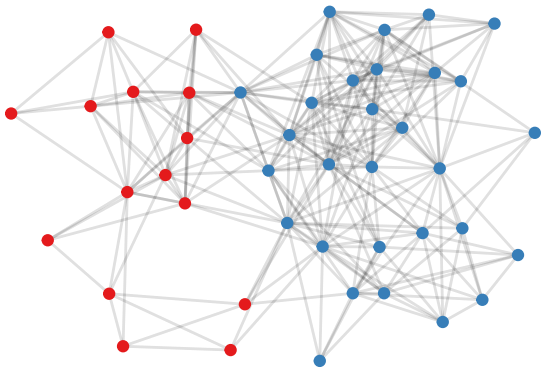

Figure 19: Macaque visuotactile brain areas and connections network. The vertices represent brain areas.

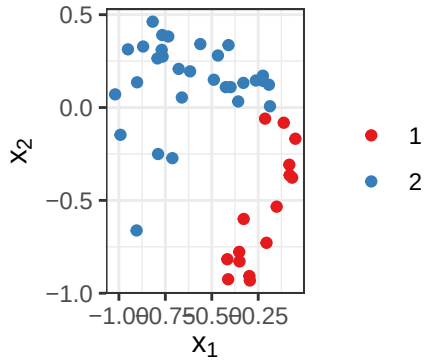

Figure 20: ASE on the Macaque brain areas network. The embedding suggests a community-wise manifold structure.

One way to model the data in example 4.1 as a GRDPG is to suppose that the latent vectors lie on one of K manifolds, and the label of each latent vector's corresponding vertex is the label of its manifold. The goal of community detection in this setting is to partition the graph that results from this latent configuration such that each partition maps onto each manifold. Before diving into the multi-manifold case, we can first consider the single manifold case as a generalization of the Erdös-Rényi model (example 4.2).

Example 4.2. In the Erdös-Rényi model, the edge probability matrix has a fixed value $P_{ij} \equiv p \in [0, 1]$. This corresponds to a one-dimensional RDPG with a latent structure

$X = \left[\sqrt{p}, \cdots, \sqrt{p}\right]^\top$. In another view, the RDPG with latent structure $X = \left[\sqrt{p}, \cdots, \sqrt{p}\right]^\top$ induces an Erdös-Rényi model in which the edge probabilities have a fixed value.

Suppose that we have RDPG in which the latent space is in $\mathbb{R}^2$ and the latent vectors are drawn uniformly from the quarter unit circle defined by $g(t) = \left[\cos\frac{\pi}{2}t, \sin\frac{\pi}{2}t\right]^\top$ for $t \in [0, 1]$. Then it can be shown that instead of a fixed $P_{ij} = p$, this RDPG induces a model in which the edge probabilities follow a distribution with density $f(p) = \frac{2}{\pi - 2}\left(\frac{1}{1-p^2} - 1\right)$.

By changing the latent structure from a point mass to a curve, we are able to come up with more flexible Bernoulli random graph models in which the edge probabilities follow more general probability distributions. Community structure can then be added by sampling latent vectors from multiple curves. The goal of this section is to explore GRDPGs with such structures and develop algorithms for community detection and parameter estimation.

4.2 The Manifold Block Model

We begin by defining the model:

Definition 4.1 (Manifold block model)**.** Let $p, q \geq 0$, $d = p + q \geq 1$, $1 \leq r < d$, $K \geq 2$, and $n > K$ be integers. Define manifolds $\mathcal{M}_1, ..., \mathcal{M}_K \in \mathcal{X}$ for $\mathcal{X} = \{x, y \in \mathbb{R}^d : x^\top I_{p,q} y \in [0, 1]\}$ each by continuous function $g_k : [0, 1]^r \to \mathcal{X}$. Define probability distributions $F_1, ..., F_K$ each with support $[0, 1]^r$. Then the following mixture model is a *manifold block model*.

1. Draw labels $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$.
2. Draw latent vectors by first taking each $t_i \overset{\text{ind}}{\sim} F_{z_i}$ and then computing each $x_i = g_{z_i}(t_i)$.
3. Compile the latent vectors into data matrix $X = \left[x_1 \mid \cdots \mid x_n\right]^\top$ and define the adjacency matrix as $A \sim \text{GRDPG}_{p,q}(X; \rho_n)$.

Definition 4.1 describes the most general version of the manifold block model (MBM). Some plausible constraints may be to fix one distribution for each underlying distribution $F_{z_i} \equiv F$ or to restrict the manifolds to be one-dimensional, i.e., $r \equiv 1$. We use these

constraints to construct the following example:

Example 4.3. Define two one-dimensional manifolds in $\mathbb{R}^2$ by $f_1(t) = \left[\cos(\frac{\pi}{3}t), \sin(\frac{\pi}{3}t)\right]^\top$ and $f_2(t) = \left[1 - \cos(\frac{\pi}{3}t), 1 - \sin(\frac{\pi}{3}t)\right]^\top$. Draw $t_1, ..., t_n \overset{\text{iid}}{\sim} \text{Uniform}(0, 1)$ and $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(\frac{1}{2}, \frac{1}{2})$, and compute latent vectors $x_i = f_{z_i}(t_i)$, which are collected in data matrix $X = \left[x_1 \mid \cdots \mid x_n\right]^\top$. Finally, let $A \sim \text{RDPG}(X)$. Figure 21 shows the latent configuration drawn from this latent distribution, a random dot product graph drawn from the latent configuration, and the ASE of the graph. Although there is no obvious community structure in the graph itself, the embedding shows a clear separation between the two communities. More specifically, similar to the latent vectors, the embedding vectors lie approximately on one of two curve segments (one-dimensional manifolds), suggesting that the appropriate community detection approach in this setting is to determine to which (unobserved) curve to which each embedding vector is closest.

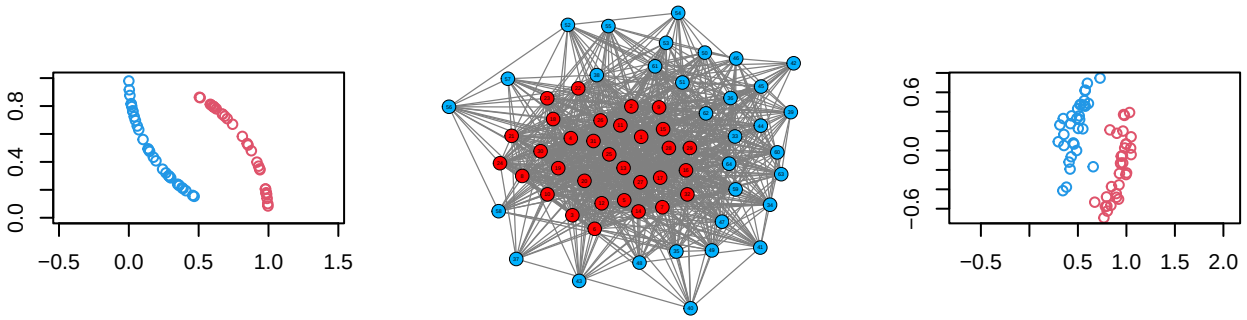

Figure 21: Manifold block model from two disjoint curves. The latent configuration is on the left, a random dot product graph drawn from the latent configuration is on the middle, and the ASE is on the right.

In the case of the SBM, a GRDPG-motivated community detection algorithm is to construct the ASE of the graph and then apply a point cloud based clustering algorithm such as K-means or GMM on the embedding. We saw that in the case of the DCBM and PABM, because the latent structures are not based on point masses, while the ASE does reveal the communities, the final clustering step requires other clustering algorithms that are

tailored to the latent community structure. Likewise, in the following sections, we develop algorithms specific to the structure of the MBM, one in the case of disjoint manifolds (section 4.3) and one in the general case in which the latent manifolds can intersect (section 4.4). We also derive theoretical consistency properties for both settings.

4.2.1 Previous Work

Manifolds have been observed in the ASEs of real observed graphs when they are studied as GRDPGs. The *Drosophila* connectome graph (Eichler et al., 2017) displays community-wise linear and quadratic structures in the ASE. Priebe et al. (2017) used GMMs on the ASE for community detection and found that the curve-like structures in the ASE were incompatible with centroid-based clustering methods. Sanna Passino and Heard (2022) used Gaussian processes to estimate curves in the embedding, which resulted in better empirical fits. Athreya et al. (2020) considered GRDPGs in which the vertices lie on curves or manifolds, but their focus was largely on estimating distributions on a single manifold rather than estimating the manifolds or identifying multiple manifolds. Rubin-Delanchy (2020) and Whiteley et al. (2022) noted that kernel GRDPGs (GRDPGs in which $P_{ij} = \kappa(x_i, x_j)$ for some kernel function κ) with community structure can have large dimensions and proposed using a lower dimension with manifold structure to approximate the nonlinearity of the kernel function.

Outside of random graph inference, there is a larger body of work on fitting parametric curves to vectors in $\mathbb{R}^d$ or identifying vectors belonging to different manifolds in $\mathbb{R}^d$. Manifold learning itself is a large area of research with a long history (Tenenbaum et al., 2000, Pastva, 1999, Trosset and Buyukbas, 2020, Asta, 2021), so we will focus primarily on the setting in which there are multiple manifolds and the goal is to identify which vector lies on which manifold (manifold clustering). The standard approach to manifold clustering is essentially a type of spectral clustering, involving spectral decomposition of some kernel matrix applied to the vectors lying on manifolds (or its graph Laplacian matrix). Unlike

standard manifold clustering techniques such as ratio cut, it has been observed that while the leading eigenvalues discern subspaces or approximations of subspaces, some of the other eigenvalues or a mixture of other eigenvalues help discern nonlinear structures (Whiteley et al., 2022, Li et al., 2022). Most of these methods also assume that the vectors lie exactly on the manifolds (i.e., the observed vector is $x_i \in \mathcal{M}_k$) instead of approximately on them (i.e., the observed vector is $x_i = y_i + \epsilon_i$ where $y_i \in \mathcal{M}_k$ and $\|\epsilon_i\|$ is small in some sense), which is the setting in which the clustering is done on the ASE.

4.3 Algorithm for Nonintersecting Manifolds

We begin with the case of GRDPGs in which communities are represented by nonintersecting manifolds. In this section, we consider the following setting: Suppose that each community C_k is represented by a closed manifold $\mathcal{M}_k$ in the latent space of a RDPG or GRDPG. Define $\delta = \min_{k \neq \ell} \min_{x \in \mathcal{M}_k, y \in \mathcal{M}_\ell} \|x - y\|$, the minimum distance between two manifolds. Furthermore, assume that $\delta > 0$, i.e., the manifolds do not intersect. This model can be viewed as a generalization of the SBM, which, when viewed as a GRDPG, consists of a latent space configuration in which each community corresponds to a point mass in the latent space. In this model, we generalize the point mass to a manifold. In the case of the DCBM in which each ω_i is bounded away from 0, i.e., $\omega_i \geq c > 0$, the nonintersecting MBM can be applied as well.

In the original latent space, it is possible to construct for each manifold $\mathcal{M}_k$ a connected η_k-neighborhood graph for some $\eta_k > 0$. If the subsample of vectors on each manifold is sufficiently dense such that $\max_k \eta_k = \eta < \delta$, then an η-neighborhood graph on the entire sample results in a graph with K disconnected subgraphs that map onto each manifold exactly. This is equivalent to applying single linkage clustering on the ASE and cutting the dendrogram at K clusters (see algorithm 7). The remainder of this section explores under which conditions these criteria are met for the latent configuration, in which the latent vectors lie exactly on manifolds, as well as for the ASE of the GRDPG sampled from these

latent configurations, in which the vectors lie approximately on manifolds with some noise that approaches zero asymptotically.

Algorithm 7: ASE clustering for nonintersecting communities.

Data: Adjacency matrix A, number of communities K, embedding dimensions (p, q).

Result: Community assignments $z_1, ..., z_n \in \{1, ..., K\}$.

1 Compute $\hat{X}$, the ASE of A using the p most positive and q most negative eigenvalues and their corresponding eigenvectors.

2 Apply single linkage clustering with K communities on $\hat{X}$.

Theorem 4.1 (Community detection for the GRDPG for which the communities come from nonintersecting manifolds)**.** *Let $x_1, ..., x_n$ be points sampled on K compact, connected manifolds $\mathcal{M}_1, ..., \mathcal{M}_K \subset \mathbb{R}^d$ each with probability measures $F_1, ..., F_K$, and the manifolds are separated by distance at least $\delta = \min_{k \neq \ell} \min_{x_i \in \mathcal{M}_k, x_j \in \mathcal{M}_\ell} \|x_i - x_j\| > 0$. Let $X = \begin{bmatrix} x_1 \mid \cdots \mid x_n \end{bmatrix}^\top$ and $A \sim \text{GRDPG}_{p,q}(X; \rho_n)$ for some $p, q \in \mathbb{N}_0$ such that $p + q = d$ and sparsity parameter ρ_n that satisfies $n\rho_n = \omega(\log^c n)$ for some $c > 1$. Define $A_n(\eta)$ as the event that an η-neighborhood graph constructed from the ASE of A consists of exactly K disconnected subgraphs that map exactly to each manifold. Then for some $C > 0$ and any $\eta \in (0, C\delta)$, $\lim_{n \to \infty} P(A_n(\eta)) = 1$.*

Example 4.4. Refer back to the RDPG in example 4.3 in which the latent vectors are drawn uniformly from two nonintersecting curves. Figure 21 shows that while the ASE introduces noise to the original latent vectors, there is still sufficient separation between the two communities. Applying single-linkage clustering on the ASE results in recovering the original community labels, up to permutation (figure 22).

Figure 22: Single-linkage clustering applied to the ASE of an MBM with two nonintersecting curves. The ASE is on the left and the dendrogram from single-linkage clustering, colored by the original community labels, is on the right.

4.4 Algorithm for Intersecting Manifolds

The consistency of the ASE provides asymptotic guarantees for perfect community label recovery for the nonintersecting MBM. This relies on the fact that single-linkage clustering and η-neighborhood clustering graphs result in disconnected components if clusters of points are far away from each other. However, many graphs do not behave so nicely. For example, the DCBM and PABM can both be viewed as MBMs in which the latent vectors come from (linear) manifolds that intersect at the origin (see section 2.7). Nonlinear intersecting manifolds have also been observed in ASEs of real graphs (Sanna Passino and Heard, 2022, Athreya et al., 2017). While the DCBM and PABM cases can be handled by algorithms specific to linear structures such as SSC and OSC, nonlinear intersecting manifolds require a more flexible approach. For the intersecting case, we restrict the MBM to the case in which each F_{z_i} is identical and univariate on the unit interval $[0, 1]$. The full mixture model is described as follows:

1. Draw $t_1, ..., t_n \overset{\text{iid}}{\sim} F$ for probability distribution F with support $[0, 1]$.
2. Draw $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(\alpha_1, ..., \alpha_K)$, the community labels.
3. Let each $x_i = g_{z_i}(t_i)$ be the latent vector for vertex v_i, and collect the latent vectors into matrix $X = \begin{bmatrix} x_1 \mid \cdots \mid x_n \end{bmatrix}^\top$.
4. Draw $A \sim \text{GRDPG}_{p,q}(X; \rho_n)$.

Example 4.5. Suppose we have a two-dimensional RDPG in which latent vectors lie on one of two curves defined by the parameterizations $g_1(t) = \left[t^2, 2t(1-t)\right]^\top$ and $g_2(t) = \left[2t(1-t), (1-t)^2\right]^\top$. This can be described as a two-community MBM in which the manifolds are both one-dimensional. For simplicity, we restrict $F_1 = F_2 = \text{Uniform}(0, 1)$. g_1 and g_2 intersect at two points, $t = 0$ and $t = 2/3$, which in the latent space correspond to $(0, 0)$ and $(4/9, 4/9)$. Clearly, single-linkage or η-neighborhood clustering does not work for this setting, both in the original latent space and in the ASE. The community structure is also not clear from the adjacency matrix. However, there still is a clear pattern in the ASE that reveals the community structure (figure 23).

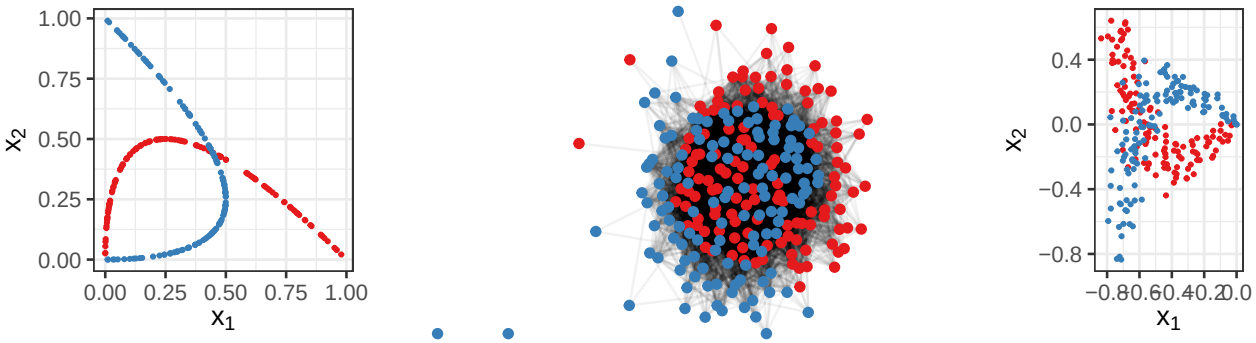

Figure 23: Latent vectors on intersecting curves (left), along with an RDPG drawn from this configuration (center) and its ASE (right).

In this setting, if we know the labels $z_1, ..., z_n$, then for each k^{th} subsample for which $z_i = k$, we can fit the curve $\hat{g}_k$ (up to an indefinite orthogonal transformation). On the other hand, if we know the curves $g_1, ..., g_K$, then we can assign each embedding vector $\hat{x}_i$ the predicted label $\hat{z}_i$ based on which curve it is closest to. This lends itself to an alternating coordinate descent algorithm in which we alternate between fitting curves and assigning labels at each iteration (algorithm 8). We call this algorithm K-curves clustering. In this algorithm, the loss function that we wish to minimize is

$$L(z_1, ..., z_n, g_1, ..., g_n; X) = \frac{1}{n} \sum_{k=1}^{K} \sum_{i:z_i=k} \|x_i - g_k(t_i)\|^2 \tag{14}$$

Algorithm 8: K-curves clustering.

Data: Adjacency matrix A, number of communities K, embedding dimensions p, q, stopping criterion ϵ

Result: Community assignments $1, ..., K$, curves $g_1, ..., g_K$

Compute X, the ASE of A using the p most positive and q most negative eigenvalues and their corresponding eigenvectors.

Initialize community labels $z_1, ..., z_n$.

repeat

  for $k = 1, ..., K$ **do**

    Define X_k as the rows of X for which $z_i = k$.

    Fit curve g_k and positions t_{k_i} to X_k by minimizing $\sum_{k_i} \|x_{k_i} - g_k(t_{k_i})\|^2$.

  end

  for $k = 1, ..., K$ **do**

    Assign $z_i \leftarrow \arg\min_\ell \|x_i - g_\ell(t_i)\|^2$.

  end

until *the change in* $\sum_k \sum_{i \in C_k} \|x_i - g_k(t_i)\|^2$ *is less than* ϵ

K-curves clustering either requires knowledge of the functional form of each g_k or a function that is flexible enough to approximate it. The choice of g_k affects the difficulty of the algorithm. As a balance between flexibility and ease of estimation, we consider the case where each g_k is a Bezier polynomial of degree R with coefficients p_k. Then we have $g_k(t) = g(t; p_k) = \sum_{r=0}^{R} p_k^{(r)} \binom{R}{r} (1-t)^{R-r} t^r$.

Example 4.6. Figure 24 shows one result of applying algorithm 8 on example 4.5. In this implementation, g_1 and g_2 are assumed to be Bezier curves in which $p_1^{(0)} = p_2^{(0)} = \begin{bmatrix} 0, 0 \end{bmatrix}^\top$, i.e., the curves meet at the origin. Here, an accuracy of 87.5% is achieved.

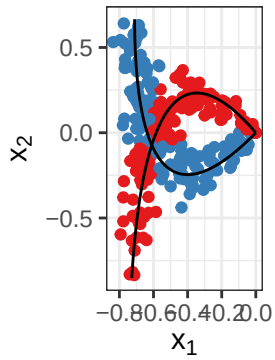

Figure 24: K-curves clustering applied to example 4.5. The embedding vectors are labeled by predicted community.

While example 4.6 achieved high accuracy, algorithm 8 often results in poor performance when it is paired with poor initialization. In example 4.6, it just so happened that we randomly chose initial curves that iterated to good fits, but this is not guaranteed. We also restricted the curves to meet at the origin, which appears to have helped guide the curves to good fits. In figures 26 and 25, we show the results of various initializations on fitting curves to example 4.5, such as random initialization and spectral clustering.

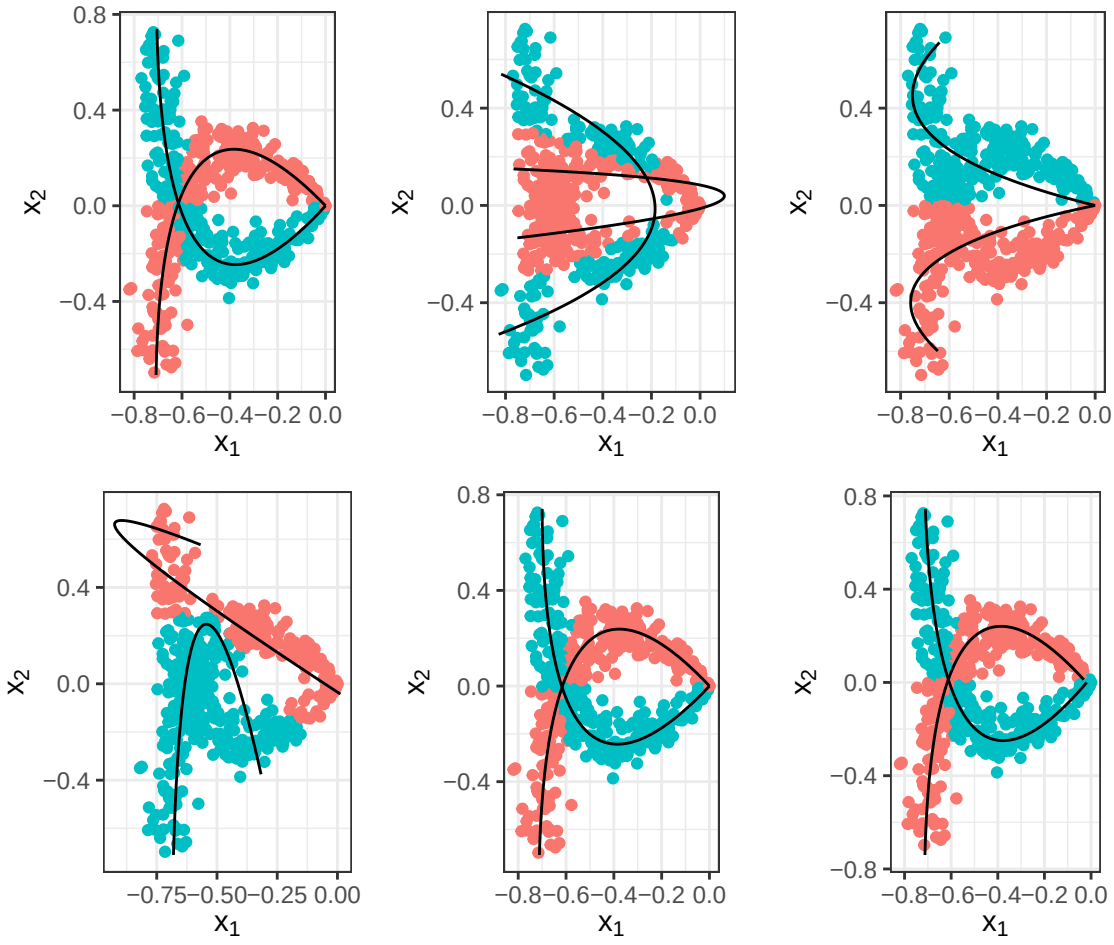

Figure 25: ASE labeled by estimated community labels for each initialization strategy. Upper right: random initialization assuming intersection at the origin. Upper middle: random initialization without assuming intersection at the origin. Upper right: initialization via spectral clustering assuming intersection at the origin. Lower left: initialization via spectral clustering without assuming intersection at the origin. Lower middle: initialization with ground truth labels, assuming intersection at the origin. Lower right: initialization with ground truth labels, without assuming intersection at the origin.

Alternatively, if we turn this problem into a semi-supervised problem in which we know a few of the community labels, then the convergence of the ASE guarantees high accuracy.

Theorem 4.2. *Let* $A \sim \text{MBM}(\{\alpha_1, ..., \alpha_K\}, \{F, ..., F\}, \{g_1, ..., g_K\}; \rho_n)$ *such that* F *has*

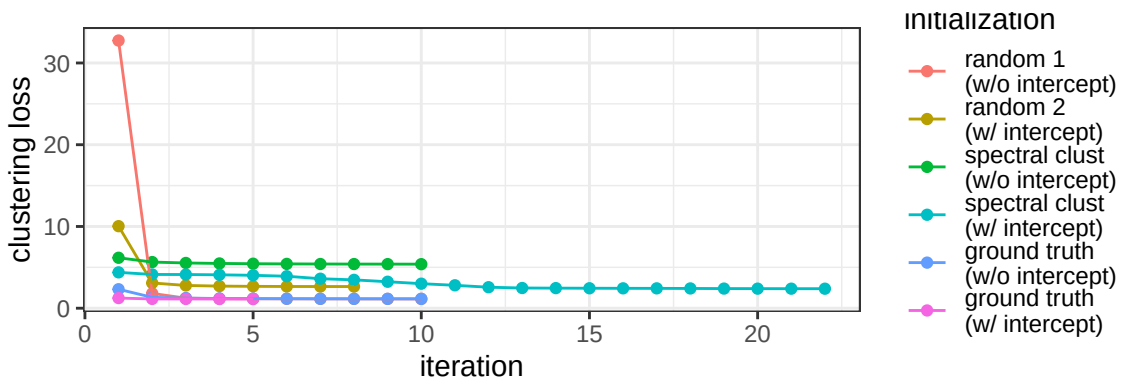

Figure 26: Clustering loss vs. iteration for each run of K-curve clustering.

support $[0,1]$*, and each* $g_k(t) = g(t; p_k)$ *is is a Bezier curve of order* R *that does not self-intersect (for any* $s \neq t$*,* $g_k(s) \neq g_k(t)$*). Suppose that for each community* k*, we have labels for at least* $R+1$ *vertices. Then if* $n\rho_n = \omega(\log^{4c} n)$*, as* $n \to \infty$*, the estimates outputted by* K*-curves clustering are such that*

$$L(\hat{z}_1, ..., \hat{z}_n, \hat{g}_1, ..., \hat{g}_K; X) \xrightarrow{p} 0,$$

where L *is the loss function defined in equation (14).*

4.5 Simulation Study

We performed two simulation experiments for algorithm 8. In the first simulation setup, we again turn to examples 4.5 and 4.6. The setup is as follows:

1. Draw $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(1/2, 1/2)$.
2. Draw $t_1, ..., t_n \overset{\text{iid}}{\sim} \text{Uniform}(0, 1)$.
3. Let each $x_i = g_{z_i}(t_i)$ where $g_1(t) = \left[t^2, 2t(1-t)\right]^\top$ and $g_2(t) = \left[2t(1-t), (1-t)^2\right]^\top$. Collect the latent vectors into matrix $X = \left[x_1 \mid \cdots \mid x_n\right]^\top$.
4. Draw $A \sim \text{RDPG}(X; \rho_n)$ (for these simulations we fix $\rho_n \equiv 1$).

For this simulation study, we set $n = 128, 256, 512, 1024, 2048$, and we start with completely random initialization as well as initializations in which we know 4 or 8 labels

from each community. For each setting, we simulate 50 replicates. Since the curves meet at the origin, we force the fitted Bezier curves to meet at the origin as well by fixing $p_1^{(0)} = p_2^{(0)} = (0, 0)$. Figure 27 shows the median and interquartile ranges for the error rates for each n and *a priori* known labels per community. The simulation results suggest that for sufficient n, we do not benefit from having more known labels, but having a few results in greatly improved performance.

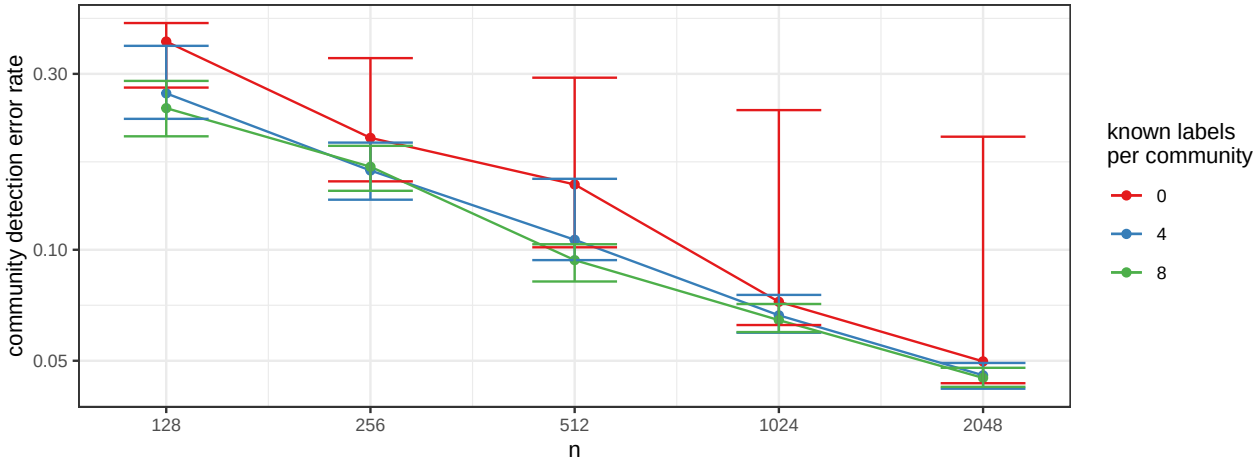

Figure 27: Median and IQR error rates over 50 replicates in the simulation setting for $K = 2$ curves.

We also explored the behavior of random initialization by setting $n = 512$ and compiling the loss values outputted by K-curves clustering for each replicate. Here, we repeated the simulations 256 times. Figure 28 suggests that in this particular setting, about half the time, random initialization of K-curves clustering results in the desired curves, while the rest of the time, K-curves clustering settled on a local minimum. Plotting each of the 256×2 fitted curves confirms this conclusion (figure 29).

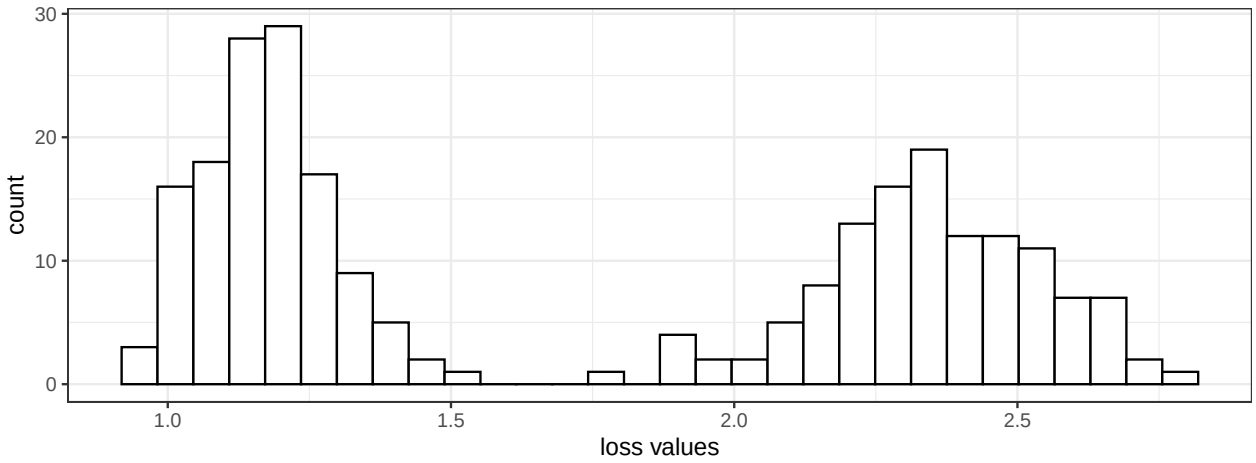

Figure 28: Histogram of loss values outputted by K-curves clustering over 256 replicates.

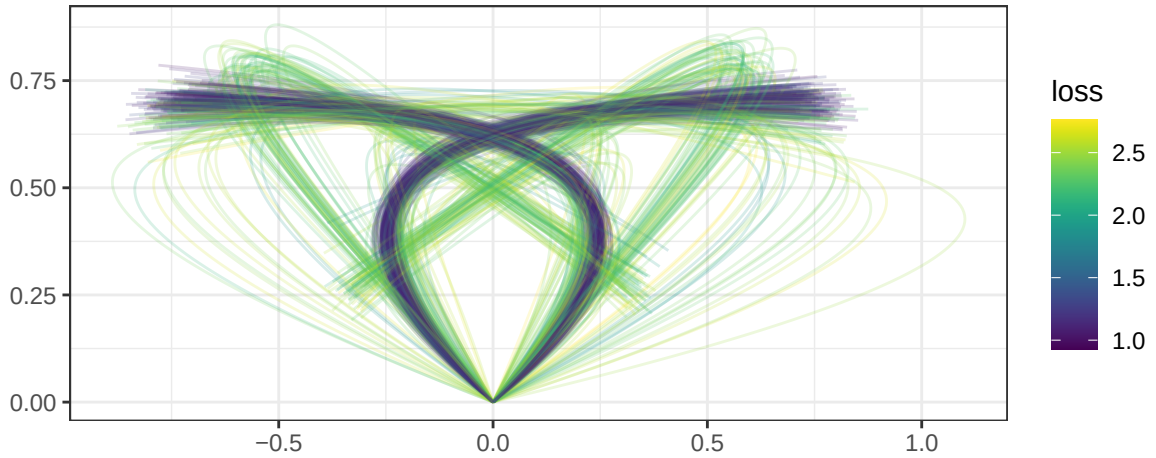

Figure 29: Fitted curves over 256 replicates. The curves are colored by the loss values.

In the second simulation, we extended the first simulation setup based on examples 4.5 and 4.6 into three dimensions and three curves. The mixture model is as follows (also see figure 30):

1. Draw $z_1, ..., z_n \overset{\text{iid}}{\sim} \text{Multinomial}(1/3, 1/3, 1/3)$.
2. Draw $t_1, ..., t_n \overset{\text{iid}}{\sim} \text{Uniform}(0, 1)$.
3. Let each $x_i = g_{z_i}(t_i)$ where
   i. $g_1(t) = \left[2t(t-1), t^2, 0\right]^\top$

ii. $g_2(t) = \left[0, t^2, 2t(t-1)\right]^\top$

iii. $g_3(t) = \left[2t(t-1), t^2, 2t(t-1)\right]^\top$.

4. Collect the latent vectors into matrix $X = \left[x_1 \mid \cdots \mid x_n\right]^\top$.
5. Draw $A \sim \text{RDPG}(X; \rho_n)$ (for these simulations we fix $\rho_n \equiv 1$).

Figure 30: Latent vectors (left) and ASE (right) of one realization of the mixture model in the simulation setup with $K = 3$ manifolds.

As in the previous setup, we set $n = 128, 256, 512, 1024, 2048$, and we started with completely random initializations as well as initializations in which we know 4 or 8 labels from each community. For each setting, we simulated 50 replicates. Since the curves meet at the origin, we forced the fitted Bezier curves to meet at the origin as well by fixing $p_k^{(0)} = (0, 0, 0)$ for each k. Figure 31 shows the median and interquartile ranges for the error rates for each n and *a priori* known labels per community. As in the previous $K = 2$ simulations, the results suggest that for sufficient n, we do not benefit from having more known labels, but having a few results in greatly improved performance. Unlike in the previous setup, random initialization results in low community detection error rates, close to the semi-supervised setups for larger n.

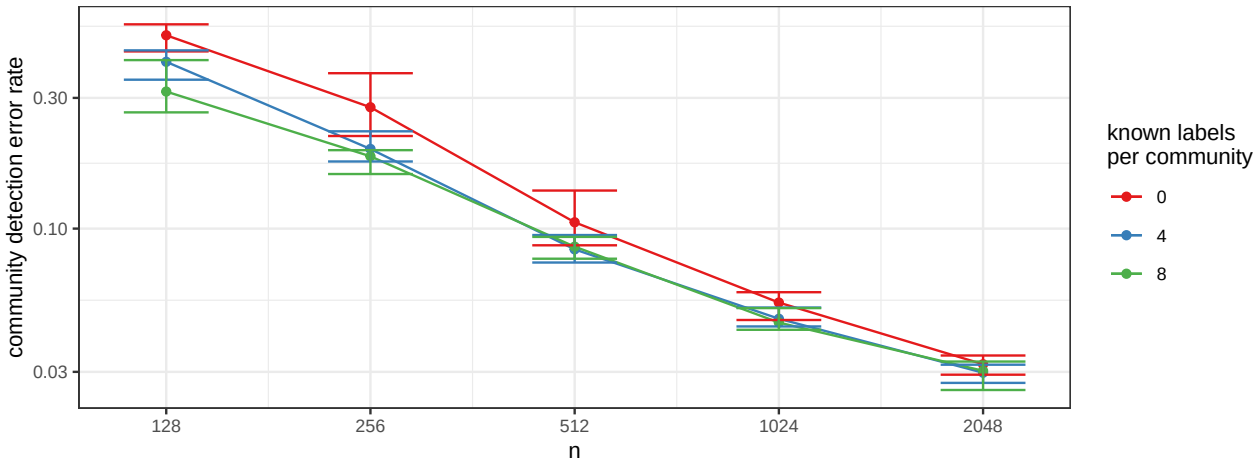

Figure 31: Median and IQR error rates over 50 replicates in the simulation setup for $K = 3$ curves.

4.6 Applications

In the first example, we applied K-curves clustering (algorithm 8) to the macaque visuotactile brain areas and connections network from example 4.1. As in example 4.1, we used $d = 2$ to construct the ASE, then used K-curves clustering to fit two curves to these data and predict the labels. Figure 32 shows the result of K-curves clustering on this graph, which yields a 4.4% error rate.

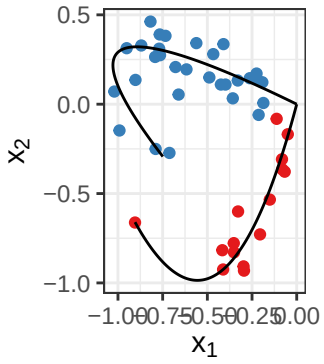

Figure 32: K-means clustering on the ASE of the Macaque brain areas network. The embedding vectors are colored by predicted label.

In the second example, we analyzed the Harry Potter enmity graph (Karakus and Pandey, 2014), visualized in figure 33. Here, the vertices represent the various characters in the *Harry Potter* series, and edges represent whether pairs of characters are enemies. This graph does not come with ground truth labels. Rubin-Delanchy et al. (2022) and Sanna Passino and Heard (2022) analyzed this graph as a GRDPG, and the ASE suggests that an appropriate model for this graph is a disassortative DCBM with $K = 2$ communities. Indeed, if we construct an ASE with $p = 1$ and $q = 1$ and label the points by character name, we see embedding vectors lie neatly along two rays emitting from the origin, with one ray populated by protagonists and another ray populated by antagonists (figure 34). Nevertheless, we tried fitting an MBM on this graph via K-curves clustering. Since the DCBM is a special case of the MBM, it should result in at least as good of a fit as the DCBM. Sanna Passino and Heard (2022) fit quadratic curves on the embedding and suggested that there was some information to be gained by the nonlinearity, so we fit quadratic Bezier curves in this example. The curves were initialized by specifying that Lord Voldemort (the main antagonist) is in one community and Harry Potter (the titular protagonist) is in another, with the other vertices not known *a priori*. The fitted quadratic curves are illustrated in figure 35, which indeed appear to be very close to linear.

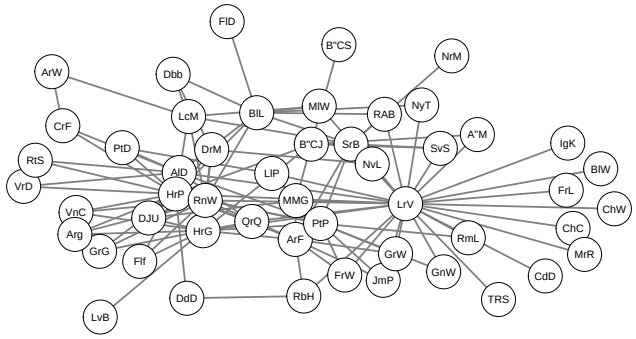

Figure 33: Harry Potter enmity graph. The vertices represent characters and the edges represent whether pairs of characters are enemies.

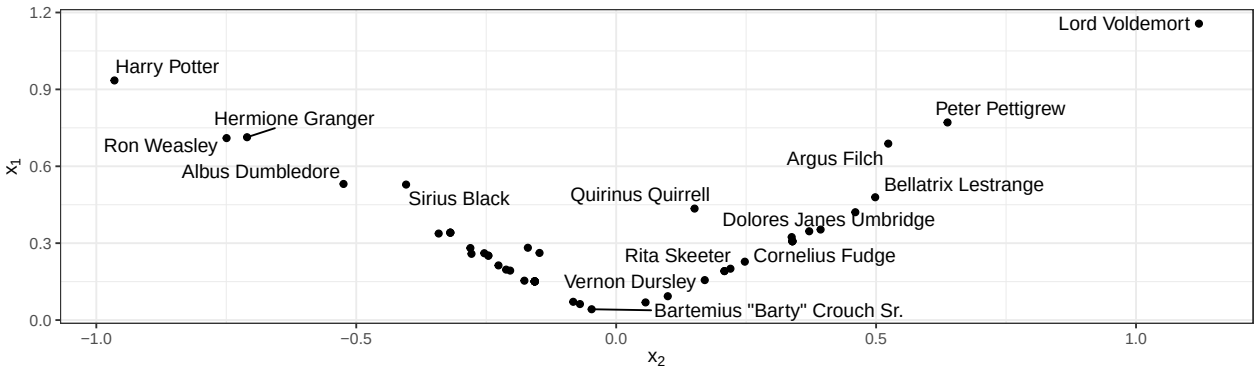

Figure 34: ASE of the Harry Potter emnity graph. The way in which the embedding vectors are organized suggest that a DCBM is an appropriate model for this graph.

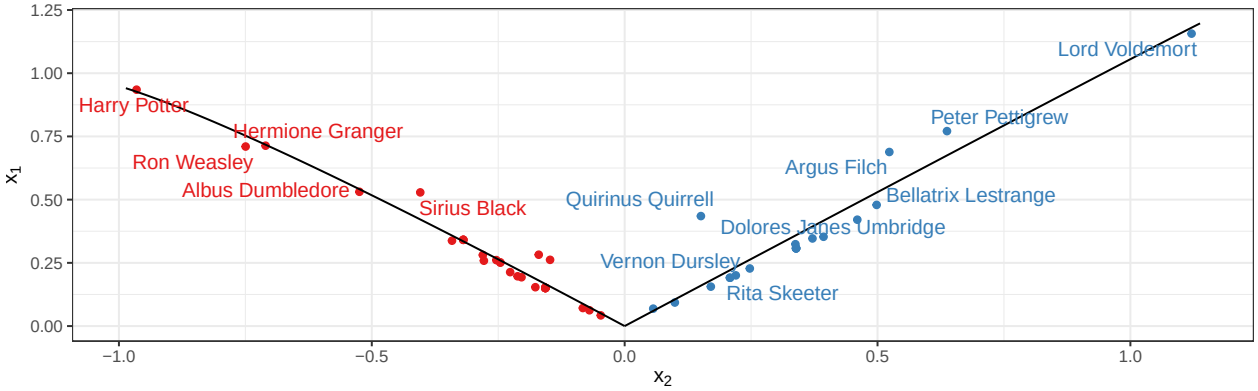

Figure 35: ASE of the Harry Potter emnity graph with quadratic Bezier curves fitted via K-curves clustering. The embedding vectors are colored by predicted label.

In the final example, we examined the *Drosophila* connectome graphs (Eichler et al., 2017), which has been studied as a GRDPG by Athreya et al. (2018), Priebe et al. (2017), and Sanna Passino and Heard (2022). This dataset consists of two graphs representing two networks of neurons, one for each hemisphere of the *Drosophila* brain. In these graphs, each vertex is a neuron, and the labels correspond to one of four neuron types (Kenyon Cells, Input Neurons, Output Neurons, and Projection Neurons). The resulting graphs are illustrated in figure 36.

The ASE in two dimensions suggest that the MBM with one dimensional quadratic

manifolds is an appropriate model for these graphs (figure 37). Fitting two quadratic Bezier curves to the ASEs (figure 38) results in 72.2% accuracy for the left hemisphere connectome and 83.6% accuracy for the right hemisphere connectome. The curves were fitted on three-dimensional ASEs with $p = 2$ and $q = 1$, which were chosen based on setting the threshold $|\lambda_i| > \sqrt{n}$ where λ_i is the i^{th} greatest eigenvalue. Figure 38 shows the embedding and curves projected to the first two components.

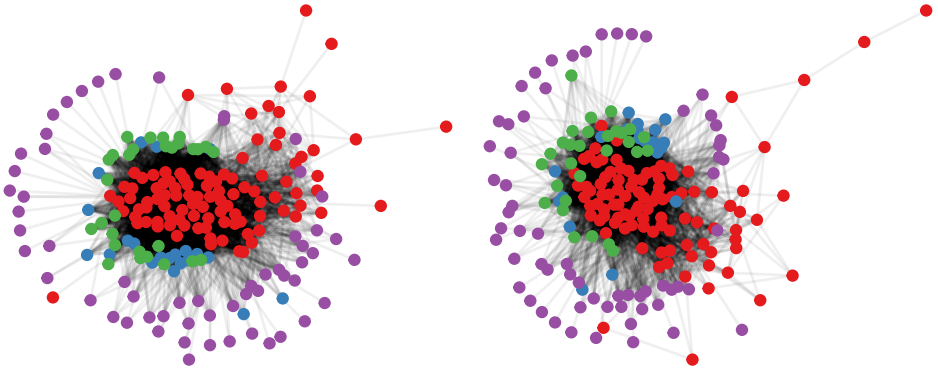

Figure 36: Graphs of the Drosophila connectomes. The left and right are of the left and right hemispheres, respectively. Each vertex represents a neuron, which are labeled by neuron type.

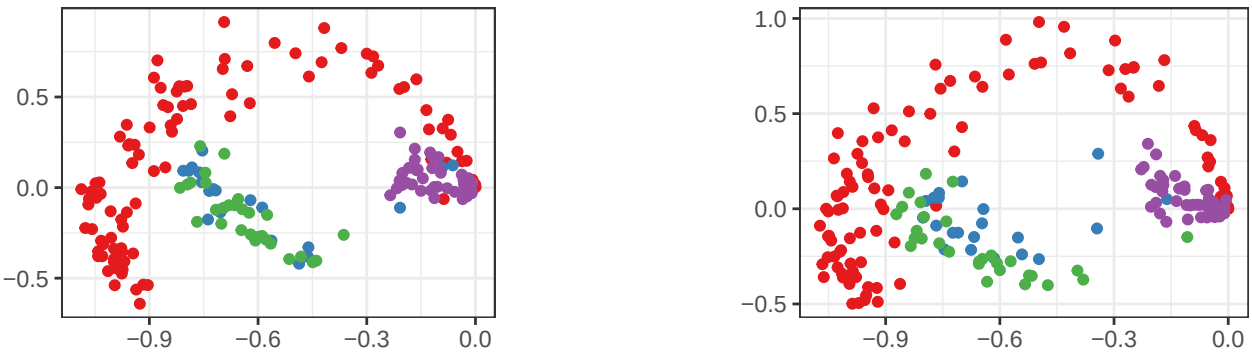

Figure 37: ASEs of the Drosophila connectome graphs. The embedding vectors are labeled by neuron type.

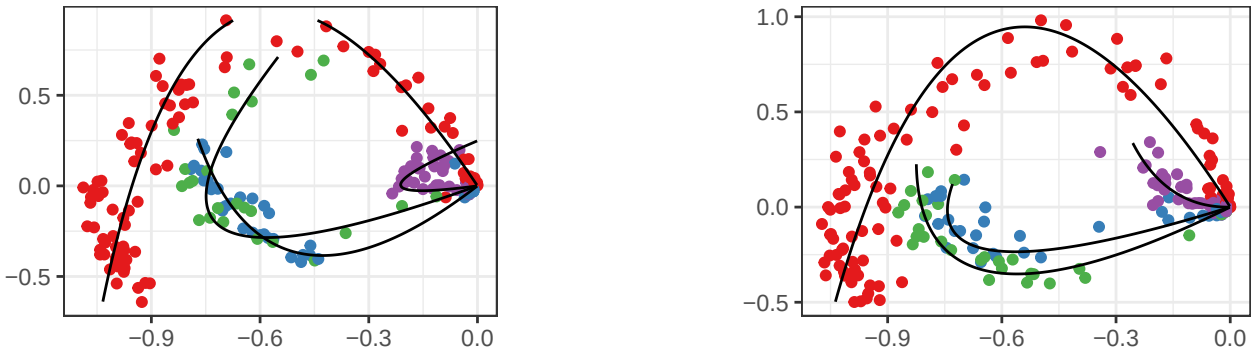

Figure 38: ASEs of the Drosophila connectome graphs with quadratic Bezier curves fitted via K-curves clustering. The embedding vectors are colored by predicted label.

5 Comparisons of Models and Conclusions

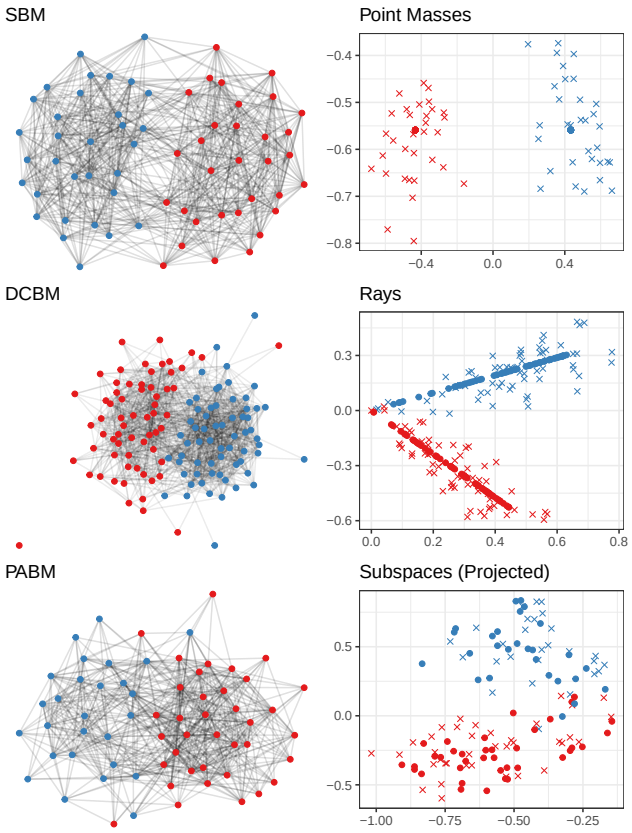

Figure 39: Graph visualizations (left) and ASEs (right) of the three main block models. Note the structures that the embedding vectors form for each model.

In this work, we showed the connection between block models and generalized random dot product graphs. In particular, block models are a type of generalized random dot product graph in which the latent vectors are organized by community. In the case of the stochastic block model, degree corrected block model, and popularity adjusted block model, the structures are linear. This makes community detection via clustering in the latent space a straightforward task: Construct a graph embedding and apply a well-known clustering algorithm that is appropriate for the (linear) latent structure. For the SBM, the appropriate choice of clustering algorithm is K-means clustering or Gaussian mixture models. For the DCBM, an appropriate choice is K-means clustering with the cosine kernel. For the PABM,

an appropriate choice is sparse subspace clustering, and we also developed orthogonal spectral clustering for this setting as well. Furthermore, by looking at how the embedding vectors are organized, the embedding makes visibly clear which of the three block models is the best fit for the graph, and in the case in which the number of communities or the assortativity or disassortativity of the model is unknown, there are existing heuristics for determining these based on the size of the eigenvalues. This is in contrast to direct likelihood maximization (e.g., via expectation maximization), which can be more of a black box in determining the appropriate model, number of communities, graph structure, etc., and results in slower, less interpretable algorithms. In conclusion, the graph embedding approach, which relies on making the connection between the block models and the GRDPG, is a powerful tool in both exploratory data analysis and statistical inference on graphs with community structure.

6 Appendix A: Proofs of Theorems from Section 3

Let V_n and $\hat{V}_n$ be the $n \times K^2$ matrices whose columns are the eigenvectors of P and A corresponding to the K^2 largest eigenvalues (in modulus), respectively. We first state an important technical lemma for bounding the maximum ℓ_2 norm difference between the rows of $\hat{V}_n$ and V_n. See Cape et al. (2019) and Rubin-Delanchy et al. (2022, Lemma 5) for a proof.

Lemma 6.1. *Let $A \sim \mathrm{PABM}(\{\lambda^{(k\ell\ell)}\}_K)$ be a K-blocks PABM graph on n vertices and let V and $\hat{V}$ be the $n \times K^2$ matrices whose columns are the eigenvectors of P and A corresponding to the K^2 largest eigenvalues in modulus, respectively. Let $v_i^\top$ and $\hat{v}_i^\top$ denote the ith row of V and $\hat{V}$, respectively. Then there exists a constant $c > 1$ and an orthogonal matrix W such that with high probability,*

$$\max_i \|W\hat{v}_i - v_i\| = O\Big(\frac{\log^c n}{n\sqrt{\rho_n}}\Big).$$

In particular we can take $c = 1 + \epsilon$ for any $\epsilon > 0$.

Proof of theorem 3.3. Recall the notations in lemma 6.1 and note that, under our assumption that the latent vectors $\lambda^{(k\ell)}$ are all homogeneous, we have $\max_i \|v_i\| = O(n^{-1/2})$.

Next recall theorem 3.2; in particular $B_{ij} = nv_i^\top v_j$. We therefore have

$$\begin{aligned}
\max_{ij} |\hat{B}_{ij} - B_{ij}| &= \max_{ij} n|\hat{v}_i^\top \hat{v}_j - v_i^\top v_j| \\
&\le n \max_{ij} |\hat{v}_i^\top W W^\top \hat{v}_j - v_i^\top v_j| \\
&\le n \max_{i,j} \Big(\|W^\top \hat{v}_i - v_i\| \times \|\hat{v}_j\| + \|W^\top \hat{v}_j - v_j\| \times \|v_i\| \Big) \\
&\le n \Big(\max_{ij} \|W\hat{v}_i - v_i\|^2 + \|W\hat{v}_i - v_i\| \times \|v_j\| + \|W\hat{v}_j - v_j\| \times \|v_i\| \Big) \\
&\le n \max_i \|W\hat{v}_i - v_i\|^2 + 2n \max_i \|W\hat{v}_i - v_i\| \times \max_j \|v_j\| \\
&= O\Big(\frac{\log^c n}{n^{1/2}\rho_n^{1/2}} \Big)
\end{aligned}$$

with high probability. Theorem 3.3 follows from the above bound together with the

conclusion in theorem 3.2 that $B_{ij} = 0$ whenever vertices i and j belongs to different communities. □

We now provide a proof of theorem 3.4. Our proof is based on verifying the sufficient conditions given in theorem 6 of Wang and Xu (2016) under which sparse subspace clustering based on solving the optimization problem in equation (10) yields an affinity matrix $B = |C| + |C^\top|$ satisfying the subspace detection property of definition 3.1. We first recall a few definitions used in Soltanolkotabi and Candés (2012) and Wang and Xu (2016); for ease of exposition, these definitions are stated using the notations of the current paper and we will drop the explicit dependency on n from our eigenvectors $\hat{V}$ of A and V of P.

Definition 6.1 (Inradius)**.** The inradius of a convex body $\mathcal{P}$, denoted by $r(\mathcal{P})$, is defined as the radius of the largest Euclidean ball inscribed in $\mathcal{P}$. Let X be a $n \times d$ matrix with rows $x_1, x_2, \ldots, x_n$. We then define, with a slight abuse of notation, $r(X)$ as the inradius of the convex hull formed by $\{\pm x_1, \pm x_2, \ldots, \pm x_n\}$.

Definition 6.2 (Subspace incoherence)**.** Let $\hat{V}$ be the eigenvectors of A corresponding to the K^2 largest eigenvalues in modulus. Let $\hat{V}^{(k)}$ denote the matrix formed by keeping only the rows of $\hat{V}$ corresponding to the k^{th} community and let $\hat{V}^{(-k)}$ denote the matrix formed by omitting the rows of $\hat{V}$ corresponding to the k^{th} community. Let $(\hat{v}_i^{(k)})^\top$ denote the ith row of $\hat{V}^{(k)}$ and $\hat{V}_{-i}^{(k)}$ be $\hat{V}^{(k)}$ with the i^{th} row omitted. Let V, $V^{(k)}$, $V^{(-k)}$, and $v_i^{(k)}$ be defined similarly using the eigenvectors V of P. Finally let $\mathcal{S}^{(k)}$ be the vector space spanned by the rows of $V^{(k)}$.

Now define $\nu_i^{(k)}$ for $k = 1, 2, \ldots, K$ and $i = 1, 2, \ldots, n_k$ as the solution of the following optimization problem

$$\nu_i^{(k)} = \max_{\eta} (\hat{v}_i^{(k)})^\top \eta - \frac{1}{2\lambda} \eta^\top \eta, \quad \text{subject to } \|V_{-i}^{(k)} \eta\|_\infty \leq 1.$$

Given $\nu_i^{(k)}$, let $\mathbb{P}_{\mathcal{S}^{(k)}}(\nu_i^{(k)})$ be the vector in $\mathbb{R}^{K^2}$ corresponding to the orthogonal projection of

$\nu_i^{(k)}$ onto the vector space $\mathcal{S}^{(k)}$ and define the projected dual direction $w_i^{(k)}$ as

$$w_i^{(k)} = \frac{\mathbb{P}_{\mathcal{S}^{(k)}}(\nu_i^{(k)})}{\|\mathbb{P}_{\mathcal{S}^{(k)}}(\nu_i^{(k)})\|}.$$

Now let $W^{(k)} = \left[w_1^{(k)} \mid \cdots \mid w_{n_k}^{(k)}\right]^\top$ and define the subspace incoherence for $\hat{V}^{(k)}$ by

$$\mu^{(k)} = \mu(\hat{V}^{(k)}) = \max_{v \in V^{(-k)}} \|W^{(k)} v\|_\infty.$$

Proof of theorem 3.4. For a given $k = 1, 2 \ldots, K$, let $r^{(k)} = \min_i r(V_{-i}^{(k)})$ be inradius of the convex hull formed by the rows of $V_{-i}^{(k)}$ and let $r_* = \min_k r^{(k)}$. Then Theorem 6 in Wang and Xu (2016) states that there exists a $\lambda > 0$ such that $\sqrt{n}\hat{V}$ satisfies the subspace detection property in definition 3.1 whenever the following two conditions are satisfied:

$$\mu^{(k)} < r^{(k)} \quad \text{for all } k = 1, 2, \ldots, K, \tag{15}$$

$$\max_i \|W\hat{v}_i - v_i\| \leq \min_k \frac{r_*(r^{(k)} - \mu^{(k)})}{2 + 7r^{(k)}}. \tag{16}$$

We now verify that for sufficiently large n, equation (15) and equation (16) holds with high probability.

Verifying equation (15). If n is sufficiently large then there are enough vertices in each community k so that $\text{span}(V_{-i}^{(k)}) = \mathcal{S}^{(k)}$ for all i and hence $r^{(k)} = \min_i r(V_{-i}^{(k)}) > 0$ for all $k = 1, 2, \ldots, K$.

Next, by theorem 3.2 we have that the subspaces $\{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(K)}\}$ are mutually orthogonal, i.e., $v^\top w = 0$ for all $v \in \mathcal{S}^{(k)}$ and $w \in \mathcal{S}^{(\ell)}$ with $k \neq \ell$. Now let $z \in \mathbb{R}^{K^2}$ be arbitrary and let $\tilde{z} = \mathbb{P}_{\mathcal{S}^{(k)}} z$ be the projection of z onto $\mathcal{S}^{(k)}$. We then have $v^\top \tilde{z} = 0$ for all $v \in V^{(-k)}$. Because z is arbitrary, this implies $\|W^{(k)} v\|_\infty = 0$ for all $v \in V^{(-k)}$ and hence $\mu^{(k)} = 0$ for all $k = 1, 2, \ldots, K$. Therefore $\mu^{(k)} < r^{(k)}$ for all $k = 1 = 2, \ldots, K$ as desired.

Verifying equation (16). Let $\delta = \max_i \sqrt{n}\|W\hat{v}_i - v_i\|$. Then from lemma 6.1, we have

$\delta \xrightarrow{a.s.} 0$ and hence

$$\delta < \min_k \frac{r_*(r^{(k)} - \mu^{(k)})}{2 + 7r^{(k)}}$$

asymptotically almost surely.

In summary $\sqrt{n}\hat{V}$ satisfies the subspace detection property with probability converging to 1 as $n \to \infty$. □

Remark. Theorem 6 of Wang and Xu (2016) assumes that each row v_i of V has unit norm, i.e., $\|v_i\| = 1$ for all i. This assumption has the effect of scaling the $r^{(k)}$ so that $r^{(k)} \leq 1$ for all $k = 1, 2, \ldots, K$. We emphasize that this assumption has no effect on the proof of Theorem 3.4. Indeed, because $\mu^{(k)} = 0$ for all k, as long as the rows of $V^{(k)}$ spans the subspace $\mathcal{S}^{(k)}$, then $ar^{(k)} > \mu^{(k)}$ for any scalar $a > 0$.

Proof of Theorem 3.5. Let P be organized by community such that $P^{(k\ell)}$ denote the $n_k \times n_\ell$ matrix obtained by keeping only the rows of P corresponding to vertices in community k and the columns of P corresponding to vertices in community ℓ. We define $A^{(k\ell)}$ analogously. Recall that $P^{(k\ell)} = \lambda^{(k\ell)}(\lambda^{(\ell k)})^\top$ for all k, ℓ. We now consider estimation of $P^{(k\ell)}$ for the cases when $k = \ell$ versus when $k \neq \ell$.

Case $k = l$. Let $P^{(kk)} = \sigma_{kk}^2 u^{(kk)}(u^{(kk)})^\top$ be the singular value decomposition of $P^{(kk)}$. We can then define $\tilde{\lambda}^{(kk)} = \sigma_{kk} u^{(kk)}$. Now let $\hat{U}^{(kk)}\hat{\Sigma}^{(kk)}(\hat{U}^{(kk)})^\top$ be the singular value decomposition of $A^{(kk)}$, and let $\hat{\sigma}_{kk}^2 \hat{u}^{(kk)}(\hat{u}^{(kk)})^\top$ be the best rank-one approximation of $A^{(kk)}$. Define $\hat{\lambda}^{(kk)} = \hat{\sigma}_{kk}\hat{u}^{(kk)}$. Then $\hat{\lambda}^{(kk)}$ is the adjacency spectral embedding approximation of $\lambda^{(kk)}$, and by Theorem 5 of Rubin-Delanchy et al. (2022), we have

$$\|\hat{\lambda}^{(kk)} - \lambda^{(kk)}\|_\infty = O\Big(\frac{\log n_k}{\sqrt{n_k}}\Big)$$

with high probability. Here $\|\cdot\|_\infty$ denote the ℓ_∞ norm of a vector.

Case $k \neq l$. Let $P^{(k\ell)} = \sigma_{k\ell}^2 u^{(k\ell)}(v^{(k\ell)})^\top$ and $P^{(\ell k)} = \sigma_{kl}^2 u^{(\ell k)}(v^{(\ell k)})^\top$ be the singular value

decompositions and note that $\sigma_{k\ell} = \sigma_{\ell k}$, $u^{(k\ell)} = v^{(\ell k)}$, and $v^{(k\ell)} = u^{(\ell k)}$. Now define $\lambda^{(k\ell)} = \sigma_{k\ell} u^{(k\ell)}$ and $\lambda^{(\ell k)} = \sigma_{k\ell} v^{(k\ell)}$.

Next consider the Hermitian dilation

$$M^{(k\ell)} = 2 \begin{bmatrix} 0 & P^{(k\ell)} \\ P^{(\ell k)} & 0 \end{bmatrix}$$

which is a symmetric $(n_k + n_\ell) \times (n_k + n_\ell)$ matrix. The eigendecomposition of $M^{(k\ell)}$ is then

$$M^{(k\ell)} = \begin{bmatrix} u^{(k\ell)} & -u^{(k\ell)} \\ v^{(k\ell)} & v^{(k\ell)} \end{bmatrix} \times \begin{bmatrix} \sigma_{kl}^2 & 0 \\ 0 & -\sigma_{kl}^2 \end{bmatrix} \times \begin{bmatrix} u^{(k\ell)} & -u^{(k\ell)} \\ v^{(k\ell)} & v^{(k\ell)} \end{bmatrix}^\top$$

Thus treating $M^{(k\ell)}$ as the edge probability matrix of a GRDPG, we have latent positions in $\mathbb{R}^2$ given by the $(n_k + n_\ell) \times 2$ matrix

$$\Lambda^{(k\ell)} = \begin{bmatrix} \sigma_{k\ell} u^{(k\ell)} & \sigma_{k\ell} u^{(k\ell)} \\ \sigma_{k\ell} v^{(k\ell)} & -\sigma_{k\ell} v^{(k\ell)} \end{bmatrix} = \begin{bmatrix} \lambda^{(k\ell)} & \lambda^{(k\ell)} \\ \lambda^{(\ell k)} & -\lambda^{(\ell k)} \end{bmatrix}.$$

Now consider

$$\hat{M}^{(k\ell)} = \begin{bmatrix} 0 & A^{(k\ell)} \\ A^{(\ell k)} & 0 \end{bmatrix}$$

We can then view $\hat{M}^{(k\ell)}$ as an adjacency matrix drawn from the edge probabilities matrix $M^{(k\ell)}$. Now suppose that the adjacency spectral embedding of $\hat{M}^{(k\ell)}$ is represented as the $(n_k + n_\ell) \times 2$ matrix

$$\hat{\Lambda}^{(k\ell)} = \begin{bmatrix} \hat{\lambda}^{(k\ell)} & \hat{\lambda}^{(k\ell)} \\ \hat{\lambda}^{(\ell k)} & -\hat{\lambda}^{(\ell k)} \end{bmatrix}$$

where each $\hat{\lambda}^{(k\ell)}$ is defined as in Algorithm 3. Then by Theorem 5 of Rubin-Delanchy et al. (2022), there exists an indefinite orthogonal transformation W^* such that, with high

probability,

$$\max_i |W^* \hat{\Lambda}_i^{(k\ell)} - \Lambda_i^{(k\ell)}\| = O\Big(\frac{\log(n_k + n_\ell)}{\sqrt{n_k + n_\ell}}\Big)$$

with high probability. Here $\Lambda_i^{(k\ell)}$ and $\hat{\Lambda}_i^{(k\ell)}$ denote the ith rows of $\Lambda^{(k\ell)}$ and $\hat{\Lambda}^{(k\ell)}$, respectively.

Furthermore, by looking at the proof of Theorem 5 in (Rubin-Delanchy et al., 2022), we see that W^* is also blocks diagonal with 2 blocks where the positive eigenvalues of $M^{(k\ell)}$ forming a block and the negative eigenvalues of $M^{(k\ell)}$ forming the remaining block. Because $M^{(k\ell)}$ has one positive eigenvalue and one negative eigenvalue, we see that W^* is necessarily of the form $W^* = \left[\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right]$ Using this form for W^*, we obtain

$$\max\{\|\hat{\lambda}^{(k\ell)} - \lambda^{(k\ell)}\|_\infty, \|\hat{\lambda}^{(\ell k)} - \lambda^{(\ell k)}\|_\infty\} = O\Big(\frac{\log(n_k + n_\ell)}{\sqrt{n_k + n_\ell}}\Big)$$

with high probability. Combining this bound with the bound for $\|\hat{\lambda}^{(kk)} - \lambda^{(kk)}\|_\infty$ given above yields equation (12) in theorem 3.5. □

7 Appendix B: Proofs of Theorems from Section 4

Lemma 7.1. *Let $x_1, ..., x_n$ be drawn from K compact, connected manifolds $\mathcal{M}_1$, ..., $\mathcal{M}_K$ each with probability measures F_1, ..., F_K, and the manifolds are separated by distance at least $\delta > 0$. Suppose that for any $\epsilon > 0$ and x drawn from each F_k, on $\mathcal{M}_k$, $F(B(x, \epsilon)) > 0$ where $B(x, \epsilon)$ is the open ball of radius ϵ centered at x. Let $E_n(\eta)$ denote the event that an η-neighborhood graph constructed from $x_1, ..., x_n$ is comprised of exactly K disjoint subgraphs that map to each of the K manifolds. Then if each $n_k \to \infty$ as $n \to \infty$, $\lim_{n\to\infty} P(E_n(\eta)) = 1$ for each $\eta \in (0, \delta)$.*

Proof. It is clear that if $\eta \in (0, \delta)$, an η-neighborhood graph constructed from the sample will always consist of at least K disjoint subgraphs for which no subgraph contains vertices belonging to points from two different manifolds. Then it is sufficient to show that for a

sufficiently large n, any η-neighborhood graph (where $\eta \in (0, \delta)$) will achieve E_n.

Define each $E_{n_k}^{(k)}(\eta)$ as the event that if a sub-sample of size n_k drawn from manifold $\mathcal{M}_k$, every $x \in \mathcal{M}_k$ is within distance η of some x_j of the sub-sample. Then if $E_{n_k}^{(k)}(\eta)$ is true, the η-neighborhood graph results in a connected subgraph for points within the k^{th} manifold. By lemma 2 of Trosset and Buyukbas (2020), $P((E_{n_k}^{(k)}(\eta))^c) \leq \ell_k(1-b_k)^{n_k}$ for some $\ell_k \in \mathbb{N}$ and $b_k \in (0, 1]$. If each $E_{n_k}^{(k)}(\eta)$ is true, then E_n is achieved, so $E_n(\eta) = \bigcap_k E_{n_k}^{(k)}(\eta)$).
$\bigcap_k E_{n_k}^{(k)}(\eta) = \Big(\bigcup_k (E_{n_k}^{(k)}(\eta))^c\Big)^c$, so it is sufficient to show $\lim_{n\to\infty} P\Big(\bigcup_k (E_{n_k}^{(k)}(\eta))^c\Big) \to 0$.

$$
\begin{aligned}
P\Big(\bigcup_k (E_{n_k}^{(k)})^c\Big) &\leq \sum_k P\Big((E_{n_k}^{(k)})^c\Big) \\
&\leq \sum_k \ell_k(1-b_k)^{n_k} \\
&\leq K\ell_{\max}(1-b_{\min})^{n_{\min}},
\end{aligned}
$$

which tends to 0 as $n \to \infty$. □

Proof of theorem 4.1. Define $E_n(\eta)$ as in lemma 7.1 for manifolds $Q_n(\mathcal{M}_1), ..., Q_n(\mathcal{M}_K)$ and each $e_i = \hat{x}_i - Q_n x_i$ where $\hat{x}_i$ is the i^{th} embedding vector and Q_n is some indefinite orthogonal transformation as in Rubin-Delanchy et al. (2022). Since Q_n is a linear map, for any $\eta \in (0, \|Q_n\|\delta)$, $P(E_n(\eta)) \to 1$ as $n \to \infty$. Let $\epsilon_i = \|e_i\|$, $\epsilon = \max_i \epsilon_i$, and $C_n = \|Q_n\|$.

$A_n(\eta)$ is true if $\eta < \min_{k,\ell} \min_{x_i \in \mathcal{M}_k, x_j \in \mathcal{M}_\ell} \|\hat{x}_i - \hat{x}_j\|$, which is defined as event $D_n(\eta)$, and $\eta \geq \max_k \max_{x_i, x_j \in \mathcal{M}_k} \|\hat{x}_i - \hat{x}_j\|$, which is defined as event $\hat{E}_n(\eta)$.

For any x_i, x_j from different manifolds, $\|\hat{x}_i - \hat{x}_j\| \geq C_n\delta - 2\epsilon$ if $2\epsilon \leq C_n\delta$. By theorem 3 of Rubin-Delanchy et al. (2022), for some finite $M > 0$, $P\Big(\epsilon < M\frac{\log^c n}{\sqrt{n}}\Big) \to 1$ as $n \to \infty$, so $P(C_n\delta < 2\epsilon) \leq P(C_n\delta < 2Mn^{1/2}\log^c n) \to 0$ since $C_n\delta > 0$. Then since $P(C_n\delta - 2\epsilon > 0) \to 1$, there is an $\epsilon \in (0, C_n\delta - 2\epsilon)$ with probability 1. Thus, $P(D_n(\eta)) \to 1$.

Then to show that $P(A_n) \to 1$:

$$
\begin{aligned}
P(A_n) &= P(\hat{E}_n(\eta) \cap D_n) \\
&= P((\hat{E}_n^c(\eta) \cup D_n^c)^c) \\
&= 1 - P(\hat{E}_n^c(\eta) \cup D_n^c) \\
&\geq 1 - P(\hat{E}_n^c(\eta)) - P(D_n^c) \\
&= P(\hat{E}_n(\eta)) + P(D_n) - 1,
\end{aligned}
$$

which tends toward 1 as $n \to \infty$ since both $P(E_n(\eta))$ and $P(D_n)$ tend toward 1 as $n \to \infty$. □

Proof of theorem 4.2. Let $\hat{X}$ with rows $\hat{x}_i^\top$ be the ASE of A drawn from an MBM in which each $g_k(t) = g(t; p_k)$ is a Bezier curve of order R. Define the ASE error as $\epsilon_i = \|Q\hat{x}_i - x_i\|$ and $\epsilon = \max_i \epsilon_i$ where Q is the appropriate indefinite orthogonal transformation for the ASE. By Rubin-Delanchy et al. (2022), $\epsilon = O_P\left(\frac{\log^c n}{\sqrt{n}}\right)$.

The statement of the theorem assumes that for each community k, there at least $R+1$ points with known label k. For simplicity, suppose we have exactly $R+1$ (if there are more, then discard the rest). Then for each community k, it is possible to fit a polynomial of order R that goes through each of the $R+1$ known points exactly, yielding $g(t; \hat{p}_k)$, where $\hat{p}_k$ are the fitted coefficients of the Bezier polynomial. Use the fitted Bezier curves to define $\delta_k(t) = \|Qg(t; \hat{p}_k) - g(t; p_k)\|$, $\delta_k = \max_t \delta_k(t)$, and $\delta = \max_k \delta_k$. Since $P(\epsilon < Mn^{-1/2} \log^c n) \to 1$ for constants $M, c > 0$, $P(\delta_k < Mn^{-1/2} \log^c n) \to 1$ for each k. Since ϵ_i describes the error between each embedding point and its corresponding latent vertex, and $\delta_k(t_i)$ describes the error between the fitted curve and its corresponding true curve in the latent space,

$$
L < \frac{1}{n} \sum_i \epsilon_i + \delta_{z_i}(t_i) < \frac{1}{n} n(\epsilon + \delta) = \epsilon + \delta.
$$

Then as $n \to \infty$,

$$P(L < 2Mn^{-1/2} \log^c n) > P(\epsilon + \delta < 2Mn^{-1/2} \log^c n) \to 1.$$

□

8 Appendix C: Details on Fitting Bezier Curves to Noisy Data

The curve fitting step in algorithm 8 requires estimation of some $g : [0, 1] \to \mathbb{R}^d$ for each subset of points corresponding to each label. One choice of curve is the Bezier polynomial. The Bezier polynomial of order R is defined by function g with parameters p such that

$$g(t; p) = \sum_{r=0}^{R} p^{(r)} \binom{R}{r} t^r (1-t)^{R-r}$$

where the Bezier coefficients $p^{(r)}$ are vectors in $\mathbb{R}^d$, and $p = \left[p^{(0)} \mid \cdots \mid p^{(R)}\right]^\top$.

The general idea behind Bezier curve fitting to noisy data is as follows: We wish to minimize the squared loss function

$$L(p, t; X) = \sum_i \|x_i - g(t_i; p)\|^2. \tag{17}$$

Suppose that we know the timepoints $t_1, ..., t_n$ which we wish to use to fit Bezier coefficients $p^{(0)}, ..., p^{(R)}$ to vectors $x_1, ..., x_n \in \mathbb{R}^d$ lying on or near a Bezier curve of order R. The squared loss then can be written as

$$L(p; t, X) = \|X - Tp\|^2,$$

where p is defined as before, T is an $n \times (R+1)$ matrix such that $T_{ir} = \binom{R}{r} t_i^r (1-t_i)^{R-r}$, and $X = \left[x_1 \mid \cdots \mid x_n\right]$. Then the least squares estimates for the Bezier coefficients p is simply the linear least squares estimate

$$\hat{p} = (T^\top T)^{-1} T^\top X. \tag{18}$$

On the other hand, suppose that we know the Bezier coefficients $p = \left[p^{(0)} \mid \cdots \mid p^{(R)}\right]^\top$, and we wish to fit timepoints $t_1, ..., t_n$ to vectors $x_1, ..., x_n$ lying on or near a Bezier polynomial of order R. In this setting, we can fit each individual t_i independently by minimizing $L(t_i; p, x_i) = \|x_i - g(t_i; p)\|^2$ for each t_i. If $g(t_i; p, x_i)$ is a polynomial of order R, then $\dot{L}(t_i; p, x_i)$ is also a polynomial, of order $2R - 1$. Minimizing $L(t_i; p, x_i)$ then simply involves finding the roots of $\dot{L}(t_i; p, x_i)$ and choosing the root with the smallest objective value (in practice, the endpoints $t_i = 0$ and $t_i = 1$ are also checked). For the Bezier curve of order R, the full polynomial is given by:

$$\dot{L}(t_i; p, x_i) = \left(\sum_{r=0}^{R} \binom{R}{r} (-1)^r c_r t^r\right)\left(\sum_{r=0}^{R-1} \binom{R-1}{r} (-1)^r c_{r+1} t^r\right), \tag{19}$$

where $c_R = \sum_{r=0}^{R} (-1)^{R-r} \binom{R}{r} p^{(r)}$. The full Bezier polynomial fitting method is detailed in algorithm 9.

Algorithm 9: Procedure for fitting a Bezier polynomial to noisy data.

Data: $x_1, ..., x_n \in \mathbb{R}^d$ lying on or near a Bezier curve of order R, stopping criterion ϵ.

Result: Bezier coefficients p, timepoint values $t_1, ..., t_n$.

```
Initialize timepoints t_1, ..., t_n (e.g., by sampling from a random distribution).
repeat
    Fit p̂ by equation 18.
    for i = 1, ..., n do
        Fit t_i by equation by finding the roots of equation 19 and choosing the roots that
         minimize equation 17.
    end
until the change in equation 17 is less than ε.
```

Algorithm 9 is sensitive to initialization. We propose two initialization strategies. The first involves using a one-dimensional Isomap (Tenenbaum et al., 2000) embedding normalized to the unit interval. Since the vectors lie approximately on a one-dimensional

manifold, a one-dimensional Isomap embedding should be able to approximate the manifold, and in particular the interpoint distances along the manifold, as long as the sample is sufficiently dense along the entire curve (Trosset and Buyukbas, 2020). The second initialization strategy involves choosing $R+1$ representative vectors. Then it is possible to fit a Bezier polynomial of order R exactly to these $R+1$ vectors. If the representative vectors lie close to the true curve, then the fitted curve is also close to the true curve, providing a good initial guess. In this strategy, we use the consistency of the ASE (Rubin-Delanchy et al., 2022) to ensure that this is true asymptotically of any embedding vector in the ASE (see theorem 4.2).

9 Appendix D: Supplementary Materials

The RMarkdown and LaTeXfiles to compile this dissertation can be found at https://github.com/johneverettkoo/dissertation. The code for simulations and data analyses in section 3 can be found at https://github.com/johneverettkoo/pabm-grdpg. An R package implementing orthogonal spectral clustering (algorithm 4) is currently in progress and can be found at https://github.com/johneverettkoo/osc. The code for simulations and data analyses in section 4 can be found at https://github.com/johneverettkoo/manifold-block-models.

References

Abbe, E. (2018). Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86.

Adamic, L. A. and Glance, N. (2005). The political blogosphere and the 2004 U.S. election: Divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, LinkKDD '05, page 36–43, New York, NY, USA. Association for Computing Machinery.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2009). Mixed membership stochastic blockmodels. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21*, pages 33–40. Curran Associates, Inc.

Asta, D. M. (2021). Non-parametric manifold learning.

Athreya, A., Fishkind, D. E., Levin, K., Lyzinski, V., Park, Y., Qin, Y., Sussman, D. L., Tang, M., Vogelstein, J. T., and Priebe, C. E. (2017). Statistical inference on random dot product graphs: a survey.

Athreya, A., Fishkind, D. E., Tang, M., Priebe, C. E., Park, Y., Vogelstein, J. T., Levin, K., Lyzinski, V., Qin, Y., and Sussman, D. L. (2018). Statistical inference on random dot product graphs: a survey. *Journal of Machine Learning Research*, 18(226):1–92.

Athreya, A., Tang, M., Park, Y., and Priebe, C. E. (2020). On estimation and inference in latent structure random graphs.

Banerjee, A., Chandrasekhar, A. G., Duflo, E., and Jackson, M. O. (2013). The Diffusion of Microfinance.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396.

Cape, J., Tang, M., and Priebe, C. E. (2019). Signal-plus-noise matrix models: eigenvector deviations and fluctuations. *Biometrika*, 106:243–250.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, Complex Systems:1695.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Eichler, K., Li, F., Litwin-Kumar, A., Park, Y., Andrade, I., Schneider-Mizell, C. M., Saumweber, T., Huser, A., Eschbach, C., Gerber, B., Fetter, R. D., Truman, J. W., Priebe, C. E., Abbott, L. F., Thum, A. S., Zlatic, M., and Cardona, A. (2017). The complete connectome of a learning and memory center in an insect brain. *bioRxiv*.

Elhamifar, E. and Vidal, R. (2009). Sparse subspace clustering. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797.

Fraley, C. and Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631.

Gao, J., Liang, F., Fan, W., Sun, Y., and Han, J. (2009). Graph-based consensus maximization among multiple supervised and unsupervised models. In Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22*, pages 585–593. Curran Associates, Inc.

Gilbert, E. N. (1959). Random Graphs. *The Annals of Mathematical Statistics*, 30:1141 – 1144.

Greene, D. and Cunningham, P. (2013). Producing a unified graph representation from multiple social network views. *CoRR*, abs/1301.5809.

Ji, M., Sun, Y., Danilevsky, M., Han, J., and Gao, J. (2010). Graph regularized transductive

classification on heterogeneous information networks. In Balcázar, J. L., Bonchi, F., Gionis, A., and Sebag, M., editors, *Machine Learning and Knowledge Discovery in Databases*, pages 570–586, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jones, J. M. (2022). LGBT identification in U.S. ticks up to 7.1%. *Gallup*.

Karakus, E. and Pandey, J. (2014). Potterverse. https://github.com/efekarakus/potter-network.

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

Kolaczyk, E. and Csárdi, G. (2014). *Statistical Analysis of Network Data with R*. Use R! Springer New York.

Le, C. M., Levina, E., and Vershynin, R. (2016). Optimization via low-rank approximation for community detection in networks. *Ann. Statist.*, 44(1):373–400.

Li, Z., Chen, Y., LeCun, Y., and Sommer, F. T. (2022). Neural manifold clustering and embedding.

Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., and Ma, Y. (2013). Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:171–184.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Lorrain, F. and White, H. C. (1971). Structural equivalence of individuals in social networks. *The Journal of Mathematical Sociology*, 1(1):49–80.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In Cam, L. M. L. and Neyman, J., editors, *Proc. of the fifth Berkeley*

Symposium on Mathematical Statistics and Probability, volume 1, pages 281–297. University of California Press.

McDaid, A. F., Murphy, T. B., Friel, N., and Hurley, N. J. (2013). Improved bayesian inference for the stochastic block model with application to large networks. *Computational Statistics and Data Analysis*, 60(C):12–31.

Nasihatkon, B. and Hartley, R. (2011). Graph connectivity in sparse subspace clustering. In *Computer Vision and Pattern Recognition*, pages 2137–2144.

Nepusz, T., Petróczi, A., Négyessy, L., and Bazsó, F. (2008). Fuzzy communities and the concept of bridgeness in complex networks. *Phys. Rev. E*, 77:016107.

Noroozi, M. and Pensky, M. (2022). The hierarchy of block models. *Sankhya A*, 84(1):64–107.

Noroozi, M., Rimal, R., and Pensky, M. (2019). Estimation and clustering in popularity adjusted block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

Négyessy, L., Nepusz, T., Kocsis, L., and Bazsó, F. (2006). Prediction of the main cortical areas and connections involved in the tactile function of the visual cortex by network analysis. *European Journal of Neuroscience*, 23(7):1919–1930.

Pastva, T. A. (1999). Bezier curve fitting.

Priebe, C. E., Park, Y., Tang, M., Athreya, A., Lyzinski, V., Vogelstein, J. T., Qin, Y., Cocanougher, B., Eichler, K., Zlatic, M., and Cardona, A. (2017). Semiparametric spectral modeling of the drosophila connectome.

Rubin-Delanchy, P. (2020). Manifold structure in graph embeddings.

Rubin-Delanchy, P., Cape, J., Tang, M., and Priebe, C. E. (2022). A statistical interpretation of spectral embedding: The generalised random dot product graph. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

Sanna Passino, F. and Heard, N. A. (2022). Latent structure blockmodels for bayesian spectral graph clustering. *Statistics and Computing*, 32(2):22.

Sengupta, S. and Chen, Y. (2018). A block model for node popularity in networks with community structure. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 80(2):365–386.

Soltanolkotabi, M. and Candés, E. J. (2012). A geometric analysis of subspace clustering with outliers. *Ann. Statist.*, 40(4):2195–2238.

Sussman, D. L., Tang, M., Fishkind, D. E., and Priebe, C. E. (2012). A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107:1119–1128.

Tenenbaum, J. B., de Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

Trosset, M. W. and Buyukbas, G. (2020). Rehabilitating isomap: Euclidean representation of geodesic structure.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

Wang, B., Pourshafeie, A., Zitnik, M., Zhu, J., Bustamante, C. D., Batzoglou, S., and Leskovec, J. (2018). Network enhancement as a general method to denoise weighted biological networks. *Nature Communications*, 9(1).

Wang, Y.-X. and Xu, H. (2016). Noisy sparse subspace clustering. *Journal of Machine Learning Research*, 17(12):1–41.

Whiteley, N., Gray, A., and Rubin-Delanchy, P. (2022). Discovering latent topology and geometry in data: a law of large dimension.

Xie, F. (2021). Entrywise limit theorems of eigenvectors for signal-plus-noise matrix models with weak signals.

Young, S. J. and Scheinerman, E. R. (2007). Random dot product graph models for social networks. In Bonato, A. and Chung, F. R. K., editors, *Algorithms and Models for the Web-Graph*, pages 138–149, Berlin, Heidelberg. Springer Berlin Heidelberg.

Zhang, A. Y. and Zhou, H. H. (2016). Minimax rates of community detection in stochastic block models. *The Annals of Statistics*, 44(5):2252–2280.