Joint Embedding of Graphs

Shangsi Wang, Joshua T. Vogelstein, Carey E. Priebe

Abstract—Feature extraction and dimension reduction for networks is critical in a wide variety of domains. Efficiently and accurately learning features for multiple graphs has important applications in statistical inference on graphs. We propose a method to jointly embed multiple undirected graphs. Given a set of graphs, the joint embedding method identifies a linear subspace spanned by rank one symmetric matrices and projects adjacency matrices of graphs into this subspace. The projection coefficients can be treated as features of the graphs. We also propose a random graph model which generalizes classical random graph model and can be used to model multiple graphs. We show through theory and numerical experiments that under the model, the joint embedding method produces estimates of parameters with small errors. Via simulation experiments, we demonstrate that the joint embedding method produces features which lead to state of the art performance in classifying graphs. Applying the joint embedding method to human brain graphs, we find it extract interpretable features that can be used to predict individual composite creativity index.

Index Terms—graphs, embedding, feature extraction, statistical inference

1 Introduction

In many problems arising in science and engineering, graphs arise naturally as data structure to capture complex relationships between a set of objects. Graphs have been used in various application domains as diverse as social networks [1], internet mapping [2], brain connectomics [3], political voting networks [4], and many others. The graphs are naturally high dimensional objects with complicated topological structure, which makes graph clustering and classification a challenge to traditional machine learning algorithms. Therefore, feature extraction and dimension reduction techniques are helpful in the applications of learning graph data. In this paper, we propose an algorithm to jointly embed multiple graphs into low dimensional space. We demonstrate through theory and experiments that the joint embedding algorithm produces features which lead to state of the art performance for subsequent inference tasks on graphs.

There exist a few unsupervised approaches to extract features from graphs. First, classical Principal Component Analysis can be applied by treating each edge of a graph as a raw feature [5]. This approach produces features which are linear combinations of edges, but it ignores the topological structure of graphs and the features extracted are not easily interpretable. Second, features can be extracted by computing summary topological and label statistics from graphs [6], [7]. These statistics commonly include number of edges, number of triangles, average clustering coefficient, maximum effective eccentricity, etc. In general, it is hard to know what intrinsic statistics to compute *a priori* and computing some statistics can be computationally expensive. Third, many frequent subgraph mining algorithms are developed [8]. For example, the fast frequent subgraph mining algorithm can identify all connected subgraphs that occur in a large fraction of graphs in a graph data set [9]. Finally, spectral feature selection can also be applied to graphs. It treats each graph as a node and constructs an object graph based on a similarity measure. Features are computed through the spectral decomposition of this object graph [10].

Adjacency Spectral Embedding (ASE) and Laplacian Eigenmap (LE) are proposed to embed a single graph observation [11], [12]. The inference task considered in these papers is learning of the block structure of the graph or clustering vertices. Given a set of graphs $\{G_i = (V_i, E_i)\}_{i=1}^m$, ASE and LE need to embed an adjacency matrix or Laplacian matrix of G_i individually, and there is no easy way to combine multiple embeddings. The joint embedding method considers the set of graphs together. It takes a matrix factorization approach to extract features for multiple graphs. The algorithm manages to simultaneously identify a set of rank one matrices and project adjacency matrices into the linear subspace spanned by this set of matrices. The joint embedding can be understood as a generalization of ASE for multiple graphs. We demonstrate through simulation experiments that the joint embedding algorithm extracts features which lead to good performance for a variety of inference tasks. In the next section, we review some random graph models and present a model for generating multiple random graphs. In Section 3, we define the joint embedding of graphs and present an algorithm to compute it. In Section 4, we perform some theoretical analyses of our joint embedding. The theoretical results and real data experiments are explored in Section 5. We conclude the paper with a brief discussion of implications and possible future work.

2 Setting

We focus on embedding unweighted and undirected graphs for simplicity, although the joint embedding algorithm

- *Shangsi Wang and Carey Priebe are with the Department of Applied Mathematics and Statistics, Johns Hopkins University. E-mail: swang127@jhu.edu, cep@jhu.edu*
- *Joshua Vogelstein is with the Department of Biomedical Engineering and Institute for Computational Medicine, Johns Hopkins University E-mail: jovo@jhu.edu*

works on weighted graphs, and directed graphs with some modifications. Let $\{G_i = (V_i, E_i)\}_{i=1}^m$ be m graphs, each with n vertices, and $\mathbf{A}_i$ be the adjacency matrix of graph G_i. The vertices in these graphs should be matched, which means that all the graphs have a common vertex set V. The joint embedding algorithm embeds all G_is simultaneously into $\mathbb{R}^d$ and represents G_i by a vector $\lambda_i \in \mathbb{R}^d$. Before discussing the joint embedding algorithm, we need a random graph model on multiple graphs, on which the theoretical analysis is based. Let us first recall a model on a single graph: Random Dot Product Graph [13].

Definition Random Dot Product Graph (RDPG). Let F be a distribution on a set $\mathcal{X} \in \mathbb{R}^d$ satisfying $x^T y \in [0,1]$ for all $x, y \in \mathcal{X}$. Let $\mathbf{X} = [x_1^T, x_2^T, ..., x_n^T] \in \mathcal{X}^n$. The notation is $(\mathbf{X}, \mathbf{A}) \sim RDPG(F)$, if the x_i are independent and identically distributed according to F, and conditioned on $\mathbf{X}$, the $\mathbf{A}_{st}$ are independent Bernoulli random variables,

$$\mathbf{A}_{st} \sim Bernoulli(x_s^T x_t).$$

Alternatively,

$$P(\mathbf{A}|\mathbf{X}) = \prod_{s<t} (x_s^T x_t)^{\mathbf{A}_{st}} (1 - x_s^T x_t)^{1-\mathbf{A}_{st}}.$$

Also, define $\mathbf{P} := \mathbf{X}\mathbf{X}^T$ to be edge probability matrix. When the latent positions $\mathbf{X}$ is regarded as parameter, the notation becomes $\mathbf{A} \sim RDPG(\mathbf{X})$.

The RDPG is a convenient model which is designed to capture more complex structures than Stochastic Block Model (introduced below). The RDPG can be further generalized to Latent Position Graph by replacing the inner product by a kernel [14]. The Adjacency Spectral Embedding of RDPG adjacency matrix is well studied [15]. Next, we propose a new random graph model which generalizes the RDPG to multiple graphs.

Definition Multiple Random Eigen Graphs (MREG). Let $\{h_k\}_{k=1}^d$ be a set of norm-1 vectors in $\mathbb{R}^n$, and F be a distribution on a set $\mathcal{X} \in \mathbb{R}^d$, satisfying $\sum_{k=1}^d \lambda[k] h_k h_k^T \in [0,1]^{n \times n}$ for all $\lambda \in \mathcal{X}$, where $\lambda[k]$ is the kth entry of vector λ. The m pairs $\{(\lambda_i, \mathbf{A}_i)\}_{i=1}^m$ follow a d-dimensional multiple random eigen graphs model, and the notation is

$$\{(\lambda_i, \mathbf{A}_i)\}_{i=1}^m \sim MREG(F, h_1, ..., h_d),$$

if $\{\lambda_i\}_{i=1}^m$ is independent and identically distributed according to distribution F, and conditioned on λ_i, the entries of A_i are independent Bernoulli random variables,

$$\mathbf{A}_i[s,t] \sim Bernoulli(\sum_{k=1}^d \lambda_i[k] h_k[s] h_k[t]).$$

$\mathbf{P}_i := \sum_{k=1}^d \lambda_i[k] h_k h_k^T$ is defined to be the edge probability matrix for graph i. In cases that $\{\lambda_i\}_{i=1}^m$ are of primary interest, they are treated as parameters, and it is said $\{\mathbf{A}_i\}_{i=1}^m$ follows a m-graph d-dimensional multiple random eigen graphs model with the notation:

$$\{\mathbf{A}_i\}_{i=1}^m \sim MREG(\lambda_1, ..., \lambda_m, h_1, ..., h_d).$$

Compared to the RDPG model, MREG is designed to model multiple graphs. The vectors $\{h_k\}_{k=1}^d$ are shared across graphs; a λ_i is sampled for each graph. On a single graph, they are equivalent if the edge probability matrix is positive semidefinite. In MREG, we allow self loops to happen. This is mainly for theoretical convenience. Next, we introduce another random graph model: Stochastic Block Model [16], which generalizes the Erdos-Renyi model [17]. SBM is a widely used model to study the community structure of a graph [18], [19].

Definition Stochastic Block Model (SBM). Let π be a prior probability vector for block membership which lies in the unit $K-1$-simplex. Denote by $\tau = (\tau_1, \tau_2, ..., \tau_n) \in [K]^n$ the block membership vector, where τ is a multinomial sequence with probability vector π. Denote by $\mathbf{B} \in [0,1]^{K \times K}$ the block connectivity probability matrix. Suppose $\mathbf{A}$ is a random adjacency matrix given by,

$$P(\mathbf{A}|\tau, \mathbf{B}) = \prod_{i<j} \mathbf{B}_{\tau_s, \tau_t}^{\mathbf{A}_{s,t}} (1 - \mathbf{B}_{\tau_s, \tau_t})^{(1-\mathbf{A}_{s,t})}$$

Then, $\mathbf{A}$ is an adjacency matrix of a K-block stochastic block model graph, and the notation is $\mathbf{A} \sim SBM(\pi, \mathbf{B})$. Sometimes, τ may also be treated as the parameter of interest, in this case the notation is $\mathbf{A} \sim SBM(\tau, \mathbf{B})$.

The top panel of Figure 1 shows the relationships between three random graph models defined above and the Erdos-Renyi (ER) model on 1 graph. The models considered are those conditioned on latent positions, that is τ, $\mathbf{X}$ and λ in SBM, RDPG and MREG respectively are treated as parameters; furthermore, loops are ignored in MREG. If an adjacency matrix $\mathbf{A} \sim SBM(\tau, \mathbf{B})$ and the block connectivity matrix $\mathbf{B}$ is positive semidefinite, $\mathbf{A}$ can also be written as an $RDPG(\mathbf{X})$ with $\mathbf{X}$ having at most K distinct rows. If an adjacency matrix $\mathbf{A} \sim RDPG(\mathbf{X})$, then it is also a 1-graph $MREG(\lambda_1, h_1, ..., h_d)$ with h_k being the normalized kth column of $\mathbf{X}$ and λ_1 being the vector containing the squared norms of columns of $\mathbf{X}$. However, a 1-graph $MREG(\lambda_1, h_1, ..., h_d)$ is not necessarily an RDPG graph since λ_1 could contain negative entries which may result in an indefinite edge probability matrix.

The bottom panel of Figure 1 shows the relationships between the models on multiple graphs. For ER, SBM and RDPG, the graphs are sampled i.i.d. with the same parameters. MREG has the flexibility to have λ differ across graphs, which leads to a more generalized model for multiple graphs. Actually, it turns out that if d is allowed to be as large as $\frac{n(n+1)}{2}$, MREG can represent any distribution on binary graphs, which includes distributions in which edges are not independent.

Theorem 2.1. Given any distribution $\mathcal{F}$ on graphs and a random adjacency matrix $\mathbf{A} \sim \mathcal{F}$, there exists a dimension d, a distribution F on $\mathbb{R}^d$, and a set of vectors $\{h_k\}_{k=1}^d$, such that $\mathbf{A} \sim MREG(F, h_1, ..., h_d)$.

Theorem 2.1 implies that MREG is really a semi-parametric model, which can capture any distribution on graphs. One can model any set of graphs by MREG with the guarantee that the true distribution is in the model with d being large enough. However, in practice, a smaller d may lead to better inference performance due to reduction in the

dimensionality. In the next section, we consider the joint embedding algorithm which can be understood as a parameter estimation procedure for MREG.

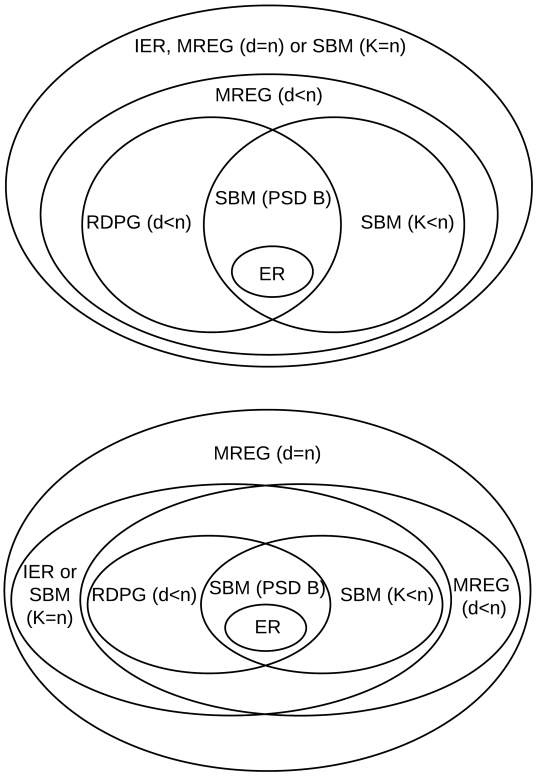


Figure 1: Relationships between random graph models on 1 graph and multiple graphs. The top panel shows the relationships between the random graph models on 1 graph. The models considered are those conditioned on latent positions, that is τ, $\mathbf{X}$ and λ in SBM, RDPG and MREG respectively are treated as parameters. ER is a 1-block SBM. If a graph follows SBM with a positive semidefinite edge probability matrix, it also follows the RDPG model. Any SBM and RDPG graph can be represented by a d-dimensional MREG model with d being less than or equal to the number of blocks or the dimension of RDPG. On one graph, inhomogeneous ER (IER), n-dimensional MREG and n-block SBM are equivalent. The bottom panel shows the relationships between the random graph models on multiple graphs. The models considered are those conditioned on latent positions, and for ER, SBM and RDPG graphs are sampled i.i.d. with the same parameters. In this case, MREG has the flexibility to have λ differ across graphs, which leads to a more generalized model for multiple graphs.

3 METHODOLOGY

3.1 Joint Embedding of Graphs

The joint embedding method considers a collection of vertex-aligned graphs, and estimates a common embedding space across all graphs and a loading for each graph. Specifically, it simultaneously identifies a subspace spanned by a set of rank one symmetric matrices and projects each adjacency matrix $\mathbf{A}_i$ into the subspace. The coefficients obtained by projecting $\mathbf{A}_i$ are denoted by $\hat{\lambda}_i \in \mathbb{R}^d$, which is called the loading for graph i. To estimate rank one symmetric matrices and loadings for graphs, the algorithm minimizes the sum of squared Frobenius distances between adjacency matrices and their projections as described below.

Definition Joint Embedding of Graphs (JE). Given m graphs $\{G_i\}_{i=1}^m$ with $\mathbf{A}_i$ being the corresponding adjacency matrix, the d-dimensional joint embedding of graphs $\{G_i\}_{i=1}^m$ is given by

$$(\hat{\lambda}_1, ..., \hat{\lambda}_m, \hat{h}_1, ..., \hat{h}_d) = \underset{\lambda_i, \|h_k\|=1}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \sum_{k=1}^{d} \lambda_i[k] h_k h_k^T\|^2. \quad (1)$$

Here, $\|\cdot\|$ denotes the Frobenius norm and $\lambda_i[k]$ is the kth entry of vector λ_i.

To make sure that the model is identifiable and avoid the problem scaling, h_k is required to have norm 1. In addition, $\{h_k h_k^T\}_{k=1}^d$ must be linearly independent to avoid identifiability issue in estimating λ_i; however, $\{h_k\}_{k=1}^d$ needs not to be linearly independent or orthogonal. To ease the notations, let us introduce two matrices $\mathbf{\Lambda} \in \mathbb{R}^{m \times d}$ and $\mathbf{H} \in \mathbb{R}^{n \times d}$, where λ_i is the ith row of Λ and h_k is the kth row of $\mathbf{H}$; that is, $\mathbf{\Lambda} = [\lambda_1^T, ..., \lambda_m^T]$ and $\mathbf{H} = [h_1, ..., h_d]$. The equation (1) can be rewritten using $\mathbf{\Lambda}$ and $\mathbf{H}$ as

$$(\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}}) = \underset{\mathbf{\Lambda}, \|h_k\|=1}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \sum_{k=1}^{d} \mathbf{\Lambda}_{ik} h_k h_k^T\|^2.$$

Denote the function on the left hand side of the equation by $f(\mathbf{\Lambda}, \mathbf{H})$ which is explicitly a function of λ_is and h_ks. There are several alternative ways to formulate the problem. If vector λ_i is converted into a diagonal matrix $\mathbf{D}_i \in \mathbb{R}^{d \times d}$ by putting entries of λ_i on the diagonal of $\mathbf{D}_i$, then solving equation (1) is equivalent to solving

$$\begin{aligned} \underset{\mathbf{D}_i, \|h_k\|=1}{\operatorname{argmin}} \quad & \sum_{i=1}^{m} \|\mathbf{A}_i - \mathbf{H}\mathbf{D}_i\mathbf{H}^T\|^2 \\ \text{subject to} \quad & \mathbf{D}_i \text{ being diagonal.} \end{aligned}$$

Equation (1) can be also viewed as a tensor factorization problem. If $\{\mathbf{A}_i\}_{i=1}^m$ are stacked in a 3-D array $\mathbb{A} \in \mathbb{R}^{m \times n \times n}$, then solving equation (1) is also equivalent to

$$\underset{\mathbf{\Lambda}, \|h_k\|=1}{\operatorname{argmin}} \|\mathbb{A} - \sum_{k=1}^{d} \mathbf{\Lambda}_{*k} \otimes h_k \otimes h_k\|^2,$$

where $\otimes$ denotes the tensor product and $\mathbf{\Lambda}_{*k}$ is the kth column of $\mathbf{\Lambda}$.

The joint embedding algorithm assumes the graphs are vertex-aligned, unweighted, and undirected. The vertex-aligned graphs are common in applications such as neuroimaging. In case that the graphs are not aligned, graph matching should be performed before the joint embedding [20], [21]. The mis-alignments of some vertices definitely have some adverse effects in estimating corresponding latent positions in H; however, a small amount of mis-aligned vertices should not have a big impact in estimating Λ. If the graphs have weighted edges, the joint embedding can still be applied. Also, the MREG model can be easily extended to weighted graphs by replacing the Bernoulli

distribution with other proper distributions. In fact, in the experiment of section 5.3, the graphs are weighted, where the edge weights are the log of fiber counts across regions of brains. In case of directed graph, to apply the joint embedding, one can symmetrize the graph by removing the direction of edges. Alternatively, $h_k h_k^T$ in equation (1) can be replaced by $h_k g_k^T$, with h_k and g_k representing the in and out latent positions respectively. With this modification, equation (1) becomes the tensor factorization problem [22].

The optimization problem in equation (1) is similar to Principal Component Analysis (PCA) in the sense of minimizing squared reconstruction error to recover loadings and components [5]. However, there are extra symmetries and rank constraints on the components. Specifically, if $h_k h_k^T$ is replaced by a symmetric matrix $\mathbf{S}_k$ in equation (1) and $\mathbf{S}_k$s are required to be orthogonal to each other, then the problem can be solved by applying PCA on vectorized adjacency matrices. The rank constrain can be also viewed as regularization. Compared to PCA, the joint embedding has fewer parameters to estimate, and h_k can be treated as latent positions for vertices, but the joint embedding yields a larger approximation error due to the extra rank constrain. Similar optimization problems have also been considered in the simultaneous diagonalization literature [23], [24]. The difference is that the joint embedding is estimating an n-by-d matrix $\mathbf{H}$ by minimizing reconstruction error instead of finding a n-by-n non-singular matrix by trying to simultaneously diagonalize all matrices. The problem in equation (1) has considerably fewer parameters to optimize, which makes it more stable and applicable with n being moderately large. In case of embedding only one graph, the joint embedding is equivalent to the Adjacency Spectral Embedding solved by singular value decomposition [11]. Next, we describe an algorithm to optimize the objective function $f(\mathbf{\Lambda}, \mathbf{H})$.

3.2 Alternating Descent Algorithm

The joint embedding of $\{G_i\}_{i=1}^m$ is estimated by solving the optimization problem in equation (1). There are a few methods proposed to solve similar problems. Carroll and Chang [25] propose to use an alternating minimization method that ignores symmetry. The hope is that the algorithm will converge to a symmetric solution itself due to symmetry in data. Gradient approaches have also been considered for similar problems [26], [27]. We develop an alternating descent algorithm to minimize $f(\mathbf{\Lambda}, \mathbf{H})$ that combines ideas from both approaches [28]. The algorithm can be also understood as a block coordinate descent method with $\mathbf{\Lambda}$ and $\mathbf{H}$ being the two blocks [29], [30]. The algorithm iteratively updates one of $\mathbf{\Lambda}$ and $\mathbf{H}$ while treating the other parameter as fixed. Optimizing $\mathbf{\Lambda}$ when fixing H is straight forward, since it is essentially a least squares problem. However, optimizing $\mathbf{H}$ when fixing $\mathbf{\Lambda}$ is hard due to the fact that the problem is non-convex and there is no closed form solution available. In this case, the joint embedding algorithm utilizes gradient information and take an Armijo backtracking line search strategy to update $\mathbf{H}$ [31].

Instead of optimizing all columns $\mathbf{\Lambda}$ and $\mathbf{H}$ simultaneously, we consider a greedy algorithm which solves the optimization problem by only considering one column of $\mathbf{\Lambda}$ and $\mathbf{H}$ at a time. Specifically, the algorithm fixes all estimates for the first $k_0 - 1$ columns of $\mathbf{\Lambda}$ and $\mathbf{H}$ at iteration k_0, and then the objective function is minimized by searching through only the k_0th column of $\mathbf{\Lambda}$ and $\mathbf{H}$. That is,

$$(\hat{\mathbf{\Lambda}}_{*k_0}, \hat{h}_{k_0}) = \underset{\mathbf{\Lambda}_{*k_0}, \|h_{k_0}\|=1}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \sum_{k=1}^{k_0-1} \hat{\mathbf{\Lambda}}_{ik} \hat{h}_k \hat{h}_k^T - \mathbf{\Lambda}_{ik_0} h_{k_0} h_{k_0}^T\|^2. \quad (2)$$

Let $f(\mathbf{\Lambda}_{*k_0}, h_{k_0})$ denote the sum on the left hand side of the equation. To compute a d-dimensional joint embedding $(\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}})$, the algorithm iteratively solves the one dimensional optimization problem above by letting k_0 vary from 1 to d.

There are a few advantages in iteratively solving one dimensional problems. First, there are fewer parameters to fit at each iteration, since the algorithm are only allowed to vary $\mathbf{\Lambda}_{*k_0}$ and h_{k_0} at iteration k_0. This makes initialization and optimization steps much easier compared to optimizing all columns of $\mathbf{H}$ simultaneously. Second, it implicitly enforces an ordering on the columns of $\mathbf{H}$. This ordering allows us to select the top few columns of $\mathbf{\Lambda}$ and $\mathbf{H}$ in cases where model selection is needed after the joint embedding. Third, it allows incremental computation. If d and d' dimensional joint embeddings are both computed, the first $\min(d, d')$ columns of $\hat{\mathbf{\Lambda}}$ and $\hat{\mathbf{H}}$ will be the same. Finally, based on numerical experiments, the difference between optimizing iteratively and optimizing all the parameters when d is small is negligible; however, the iterative algorithm yields a slightly smaller objective function when d is large. The disadvantage of optimizing each column separately is that the algorithm is more likely to end up at a local minimum when the objective function is structured not in favor of embedding iteratively. In practice, this problem can be mitigated by running the joint embedding algorithm several times with random initializations.

To find $\mathbf{\Lambda}_{*k_0}$ and h_{k_0} in equation (2), the algorithm needs to evaluate two derivatives: $\frac{\partial f}{\partial h_{k_0}}$ and $\frac{\partial f}{\partial \mathbf{\Lambda}_{ik_0}}$. Denote by $\mathbf{R}_{ik_0}$ the residual matrix after iteration $k_0 - 1$ which is $\mathbf{A}_i - \sum_{k=1}^{k_0-1} \hat{\mathbf{\Lambda}}_{ik} \hat{h}_k \hat{h}_k^T$. The gradient of the objective function with respect to h_{k_0} is given by

$$\frac{\partial f}{\partial h_{k_0}} = -4 \sum_{i=1}^{m} \mathbf{\Lambda}_{ik_0} (\mathbf{R}_{ik} - \mathbf{\Lambda}_{ik_0} h_{k_0} h_{k_0}^T) h_{k_0}. \quad (3)$$

The derivative of the objective function with respect to $\mathbf{\Lambda}_{ik_0}$ is given by

$$\frac{\partial f}{\partial \mathbf{\Lambda}_{ik_0}} = -2 \langle \mathbf{R}_{ik} - \mathbf{\Lambda}_{ik_0} h_{k_0} h_{k_0}^T, h_{k_0} h_{k_0}^T \rangle.$$

Setting the derivative to 0 yields

$$\hat{\mathbf{\Lambda}}_{ik_0} = \langle \mathbf{R}_{ik}, h_{k_0} h_{k_0}^T \rangle, \quad (4)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product.

The joint embedding algorithm alternates between updating $\hat{\mathbf{\Lambda}}_{ik_0}$ and $\hat{h}_{k_0}$ according to equation (3) and (4). Algorithm 1 describes the general procedure to compute the d-dimensional joint embedding of graphs $\{G_i\}_{i=1}^m$. The algorithm outputs two matrices: $\hat{\mathbf{\Lambda}}$ and $\hat{\mathbf{H}}$. The rows of $\hat{\mathbf{\Lambda}}$ denoted by $\{\hat{\lambda}_i\}_{i=1}^m$ can be treated as estimates of $\{\lambda_i\}_{i=1}^m$ in MREG and features for graphs. Columns of $\hat{\mathbf{H}}$ denoted by $\{\hat{h}_k\}_{k=1}^d$ are estimates of $\{h_k\}_{k=1}^d$. If a new graph G is observed with adjacency matrix $\mathbf{A}$, $\mathbf{A}$ can be projected into the linear space spanned by $\{\hat{h}_k\hat{h}_k^T\}_{k=1}^d$ to obtain features for the graph.

In case of $\mathbf{A}_i$ being large, the updating equations (3) and (4) are not practical due to $h_k h_k^T$ and $\mathbf{R}_{ik}$ being large and dense. However, they can be rearranged to avoid explicit computation of $h_k h_k^T$ and $\mathbf{R}_{ik}$. The equation (3) becomes

$$\begin{aligned}
\frac{\partial f}{\partial h_{k_0}} &= -4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}(\mathbf{R}_{ik} - \mathbf{\Lambda}_{ik_0}h_{k_0}h_{k_0}^T)h_{k_0} \\
&= -4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}\mathbf{R}_{ik}h_{k_0} + 4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}^2 h_{k_0} \\
&= -4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}(\mathbf{A}_i - \sum_{k=1}^{k_0-1}\mathbf{\Lambda}_{ik}h_k h_k^T)h_{k_0} + 4\sum_{i=1}^m \Lambda_{ik_0}^2 h_{k_0} \\
&= -4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}\mathbf{A}_i h_{k_0} + 4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}\sum_{k=1}^{k_0-1}\mathbf{\Lambda}_{ik}(h_k^T h_{k_0})h_k \\
&\quad + 4\sum_{i=1}^m \mathbf{\Lambda}_{ik_0}^2 h_{k_0}.
\end{aligned}$$

Similarly, the equation (3) can be rewritten as

$$\begin{aligned}
\hat{\mathbf{\Lambda}}_{ik_0} &= \langle \mathbf{R}_{ik}, h_{k_0}h_{k_0}^T\rangle \\
&= h_{k_0}^T\mathbf{R}_{ik}h_{k_0} \\
&= h_{k_0}^T(\mathbf{A}_i - \sum_{k=1}^{k_0-1}\mathbf{\Lambda}_{ik}h_k h_k^T)h_{k_0} \\
&= h_{k_0}^T\mathbf{A}_i h_{k_0} - \sum_{k=1}^{k_0-1}\mathbf{\Lambda}_{ik}(h_{k_0}^T h_k)^2.
\end{aligned}$$

Based on the rearranged equations, efficiently evaluating matrix vector product $\mathbf{A}_i h_{k_0}$ is needed to calculate the derivatives. This can be completed for a variety of matrices, in particular, sparse matrices [32].

The Algorithm 1 is guaranteed to converge to a stationary point. Specifically, at the termination of iteration k_0, $\frac{\partial f}{\partial h_{k_0}} \approx 0$ and $\frac{\partial f}{\partial \mathbf{\Lambda}_{ik_0}} \approx 0$. First, $\frac{\partial f}{\partial \mathbf{\Lambda}_{ik_0}} \approx 0$ is ensured due to exact updating by equation (4). Second notice that updating according to equation (3) and (4) always decreases the objective function. Due to the fact that h_{k_0} lies on the unit sphere and the objective is twice continuous differentiable, $\frac{\partial f}{\partial h_{k_0}}$ is Lipschitz continuous. This along with Armijo backtracking line search guarantees a "sufficient" decrease $c\|\frac{\partial f}{\partial h_{k_0}}\|^2$ each time when the algorithm updates h_{k_0} [31], where c is a constant independent of h_{k_0}. As a consequence, this implies $\|\frac{\partial f}{\partial h_{k_0}}\| \to 0$. In general, the objective function may have multiple stationary points due to non-convexity. Therefore, the algorithm is sensitive to initializations. In the experiment below, we initialize $\hat{\mathbf{\Lambda}}_{ik_0}$ and $\hat{h}_{k_0}$ through SVD of the average residual matrix $\sum \mathbf{R}_{ik_0}/m$. When time permits, we recommend running the joint embedding several times with random initializations.

The optimization algorithm described above may not be the fastest approach to solving the problem; however, numerical optimization is not the focus of this paper. Based on results from numerical applications, our approach works well in estimating parameters and extracting features for subsequent statistical inference. Next, we discuss some variations of the joint embedding algorithm.

Algorithm 1 Joint Embedding Algorithm

procedure FIND JOINT EMBEDDING $\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}}$ OF $\{\mathbf{A}_i\}_{i=1}^m$
  Set residuals: $\mathbf{R}_{i1} = \mathbf{A}_i$
  for $k = 1 : d$ **do**
    Initialize h_k and $\mathbf{\Lambda}_{*k}$
    while not convergent **do**
      Fixing $\mathbf{\Lambda}_{*k}$, update h_k by gradient descent (3)
      Project h_k back to the unit sphere
      Fixing h_k, update $\mathbf{\Lambda}_{*k}$ by (4)
      Compute objective $\sum_{i=1}^m \|\mathbf{R}_{ik} - \mathbf{\Lambda}_{ik}h_k h_k^T\|^2$
    end while
    Update residuals: $\mathbf{R}_{i(k+1)} = \mathbf{R}_{ik} - \mathbf{\Lambda}_{ik}h_k h_k^T$
  end for
  Output $\hat{\mathbf{\Lambda}} = [\mathbf{\Lambda}_{*1}, ..., \mathbf{\Lambda}_{*d}]$ and $\hat{\mathbf{H}} = [h_1, ..., h_d]$
end procedure

3.3 Variations

The joint embedding algorithm described in the previous section can be modified to accommodate several different settings.

Variation 1. When all graphs come from the same distribution, we can force estimated loadings $\hat{\lambda}_i$ to be equal across all graphs. This is useful when the primary inference task is to extract features for vertices. Since all graphs share the same loadings, with slightly abusing notations, let $\mathbf{\Lambda}$ be a vector in $\mathbb{R}^d$ and the optimization problem becomes

$$(\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}}) = \underset{\mathbf{\Lambda}, \|h_k\|=1}{\operatorname{argmin}} \sum_{i=1}^m \|\mathbf{A}_i - \sum_{k=1}^d \mathbf{\Lambda}_k h_k h_k^T\|^2.$$

In this case, the optimization problem can be solved exactly by finding the singular value decomposition of the average adjacency matrix $\frac{1}{m}\sum_{i=1}^m \mathbf{A}_i$.

Variation 2. When there is a discrete label $y_i \in \mathbb{Y}$ associated with G_i available, we may require all loadings $\hat{\lambda}_i$ to be equal within class. Let $\mathbf{\Lambda} \in \mathbb{R}^{|\mathbb{Y}|\times d}$, the optimization problem becomes

$$(\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}}) = \underset{\mathbf{\Lambda}, \|h_k\|=1}{\operatorname{argmin}} \sum_{i=1}^m \|\mathbf{A}_i - \sum_{k=1}^d \mathbf{\Lambda}_{y_i k} h_k h_k^T\|^2.$$

In this case, when updating $\mathbf{\Lambda}$ as in equation (4), the algorithm should average $\mathbf{\Lambda}_{y_i k}$ within the same class.

Variation 3. In some applications, we may require all $\mathbf{\Lambda}_{ik}$ to be greater than 0, as in non-negative matrix factorization.

One advantage of this constraint is that graph G_i may be automatically clustered based on the largest entry of $\hat{\lambda}_i$. In this case, the optimization problem is

$$(\hat{\mathbf{\Lambda}}, \hat{\mathbf{H}}) = \underset{\mathbf{\Lambda}\geq 0, \|h_k\|=1}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \sum_{k=1}^{d} \mathbf{\Lambda}_{ik} h_k h_k^T\|^2.$$

To guarantee nonnegativity, the algorithm should use nonnegative least squares in updating $\mathbf{\Lambda}$ [33]. Furthermore, a constrain on the number of non-zero elements in ith row of $\mathbf{\Lambda}$ can be added as in K-SVD [34], and a basis pursuit algorithm should be used to update $\mathbf{\Lambda}$ [35], [36]. Next, we discuss some theoretical properties of the joint embedding when treated as a parameter estimation procedure for the MREG model.

4 Theory

In this section, we consider a simple setting where graphs follow a 1-dimensional MREG model, that is $\{(\lambda_i, \mathbf{A}_i)\}_{i=1}^m \sim MREG(F, h_1)$. Under this MREG model, the joint embedding of graphs can be understood as estimators for parameters of the model. Specifically, $\hat{\lambda}_i$ and $\hat{h}_1$ are estimates of λ_i and h. We prove two theorems concerning the asymptotic behavior of estimator $\hat{h}_1$ produced by joint embedding.

Let $\hat{h}_1^m$ denote the estimates based on m graphs and define functions ρ, D_m and D as below:

$$\rho(\mathbf{A}_i, h) = \|\mathbf{A}_i - \langle \mathbf{A}_i, hh^T\rangle hh^T\|^2,$$

$$D_m(h, h_1) = \frac{1}{m}\sum_{i=1}^{m} \rho(\mathbf{A}_i, h),$$

$$D(h, h_1) = E(\rho(\mathbf{A}_i, h)).$$

One can understand D_m and D as sample and population approximation errors respectively. By equation (1),

$$\hat{h}_1^m = \underset{\|h\|=1}{\operatorname{argmin}} \underset{\lambda_i}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \lambda_i hh^T\|.$$

By equation (4),

$$\langle \mathbf{A}_i, hh^T\rangle = \underset{\lambda_i}{\operatorname{argmin}} \sum_{i=1}^{m} \|\mathbf{A}_i - \lambda_i hh^T\|.$$

Therefore,

$$\hat{h}_1^m = \underset{\|h\|=1}{\operatorname{argmin}} D_m(h, h_1).$$

The first theorem states that $\hat{h}_1^m$ converges almost surely to the global minimum of $D(h, h_1)$, given that the global minimum is unique. Alternatively, the theorem implies the sample minimizer converges to the population minimizer.

Theorem 4.1. If $D(h, h_1)$ has a unique global minimum at h', then $\hat{h}_1^m$ converges almost surely to h' as m goes to infinity. That is,

$$\hat{h}_1^m \overset{a.s.}{\to} h'.$$

Theorem 4.1 requires h' to be the unique global minimizer of $D(h, h_1)$. However, the global minimizer is definitely not unique due to the symmetry up to sign flip of h, that is $D(h, h_1) = D(-h, h_1)$ for any h. This problem can be addressed by forcing an orientation of $\hat{h}_1^m$ or stating that the convergence is up to a sign flip. It is also possible that there are multiple global minimizers of $D(h, h_1)$ which are not sign flips of each other. In this case, Theorem 4.1 does not apply. We are currently only certain that when all graphs are from the Erdos-Renyi random graph model, the global minimizer is unique up to a sign flip. The next theorem concerns the asymptotic bias of h'. It gives a bound on the difference between the population minimizer h' and the truth h_1.

Theorem 4.2. If h' is a minimizer of $D(h, h_1)$, then

$$\|h' - h_1\| \leq \frac{2E(\lambda_i)}{E(\lambda_i^2)(h_1^T h')^2}.$$

To see an application of Theorem 4.2, let us consider the case in which all graphs are Erdos-Renyi graphs with 100 vertices and edge probability of 0.5. Under this setting, Theorem 4.2 implies $\|h' - h_1\| \in [0, 0.04] \cup [1.28, 1.52]$. The second interval is disturbing. It is due to the fact that when $h_1^T h'$ is small, the bound is useless. We provide some insights as to why the second interval is there and how we can get rid of it with additional assumptions. In the proof of Theorem 4.2, we show that the global optimizer h' satisfies

$$h' = \underset{\|h\|=1}{\operatorname{argmax}} E(\langle \mathbf{A}_i, hh^T\rangle^2).$$

Taking a closer look at $E(\langle \mathbf{A}_i, hh^T\rangle^2)$,

$$\begin{aligned} E(\langle \mathbf{A}_i, hh^T\rangle^2) &= E(\langle \mathbf{P}_i, hh^T\rangle^2) + E(\langle \mathbf{A}_i - \mathbf{P}_i, hh^T\rangle^2) \\ &= E(\lambda_i^2)(h_1^T h)^4 + E((h^T(\mathbf{A}_i - \mathbf{P}_i)h)^2). \end{aligned}$$

Therefore,

$$h' = \underset{\|h\|=1}{\operatorname{argmax}} E(\lambda_i^2)(h_1^T h)^4 + E((h^T(\mathbf{A}_i - \mathbf{P}_i)h)^2).$$

We can see that $E(\lambda_i^2)(h_1^T h)^4$ is maximized when $h = h_1$; however, the noise term $E((h^T(\mathbf{A}_i - \mathbf{P}_i)h)^2)$ is generally not maximized at $h = h_1$. If n is large, we can apply a concentration inequality to $(h^T(\mathbf{A}_i - \mathbf{P}_i)h)^2$ and have an upper bound on $E((h^T(\mathbf{A}_i - \mathbf{P}_i)h)^2)$. If we further assume A_i is not too sparse, that is $E(\lambda_i^2)$ grows with n fast enough, then the sum of these two terms is dominated by the first term. This provides a way to have a lower bound on $h_1^T h'$. We may then replace the denominator of the bound in Theorem 4.2 by the lower bound. In general, if n is small, the noise term may cause h' to differ from h_1 by a significant amount. In this paper, we focus on the case that n is fixed. The case that n goes to infinity for Random Dot Product Graph is considered in [37].

The two theorems above concern only the estimation of h_1, but not λ_i. Based on equation (4), the joint embedding estimates λ_i by

$$\hat{\lambda}_i^m = \langle \mathbf{A}_i, \hat{h}_1^m \hat{h}_1^{mT}\rangle.$$

When m goes to infinity, we can apply Theorem 4.1,

$$\hat{\lambda}_i^m = \langle \mathbf{A}_i, \hat{h}_1^m \hat{h}_1^{mT}\rangle \overset{a.s.}{\to} \langle \mathbf{A}_i, h'h'^T\rangle = h'^T \mathbf{A}_i h'.$$

Then, applying the bound on $\|h' - h_1\|$ derived in Theorem 4.2 and utilizing the fact that $h^T \mathbf{A}_i h$ is continuous

in h, we can obtain an upper bound on $|\hat{\lambda}_i^m - h_1^T \mathbf{A}_i h_1|$. When $\mathbf{A}_i$ is large, $h_1^T \mathbf{A}_i h_1$ is concentrated around λ_i with high probability. As a consequence, with high probability $|\hat{\lambda}_i^m - \lambda_1|$ is small. In the next section, we demonstrate properties and utilities of the joint embedding algorithm through experiments.

5 Experiments

Before going into details of our experiments, we want to discuss how to select the dimensionality d of the joint embedding. Estimating d is an important model selection question that has been studied for years under various settings [38]. Model selection is not the focus of this paper, but we still face this problem in numerical experiments. In the simulation experiments of this section, we assume d is known to us and simply set the dimensionality estimate $\hat{d}$ equal to d. In the real data experiment, we recommend two approaches to determine $\hat{d}$. Both approaches require first running the d'-dimensional joint embedding algorithm, where d' is sufficiently large. We then plot the objective function versus dimension, and determine $\hat{d}$ to be where the objective starts to flatten out. Alternatively, we can plot $\{\hat{\mathbf{\Lambda}}_{ik}\}_{i=1}^m$ for $k = 1, ..., d'$, and select $\hat{d}$ when the loadings start to look like noise with 0 mean. These two approaches should yield a similar dimensionality estimate of $\hat{d}$.

5.1 Simulation Experiment 1: Joint Embedding Under a Simple Model

In the first experiment, we present a simple numerical example to demonstrate some properties of the joint embedding procedure as the number of graphs grows. We repeatedly generate graphs with 20 vertices from 3-dimensional MREG, where $\lambda_i[1] \sim \text{Uniform}(8, 16)$, $\lambda_i[2] \sim \text{Uniform}(0, 4)$ and $\lambda_i[3] \sim \text{Uniform}(0, 2)$, with

$$h_1 = [1, 1, 1, ..., 1]/\sqrt{20}$$
$$h_2 = [1, -1, 1, -1, 1, -1, ..., -1]/\sqrt{20}$$
$$h_3 = [1, 1, -1, -1, 1, 1, -1, -1, ..., -1]/\sqrt{20}.$$

We keep doubling the number of graphs m from 2^4 to 2^{12}. At each value of m, we compute the 3-dimensional joint embedding of graphs. Let the estimated parameters based on m graphs be denoted by $\hat{\lambda}_i^m$ and $\hat{h}_k^m$. Two quantities based on $\hat{h}_k^m$ are calculated. The first is the norm difference between the current h_k estimates and the previous estimates, namely $\|\hat{h}_k^m - \hat{h}_k^{m/2}\|$. This provides numerical evidence for the convergence of our principled estimation procedure. The second quantity is $\|\hat{h}_k^m - h_k\|$. This investigates whether $\hat{h}_k$ is an unbiased estimator for h_k. The procedure described above is repeated 20 times. Figure 2 presents the result.

Based on the plot, the norm of differences $\|\hat{h}_k^m - \hat{h}_k^{m/2}\|$ seem to converge to 0 as m increases. This suggests the convergence of $\hat{h}_1^m$. Second, we notice that the bias $\|\hat{h}_2^m - h_2\|$ and $\|\hat{h}_3^m - h_3\|$ do not converge to 0; instead, it stops decreasing at around 0.1 and 0.2 respectively. This suggests that $\hat{h}_k$ is an asymptotically biased estimator for h_k. Actually, this is as to be expected: when there are infinitely many nuisance parameters present, Neyman and

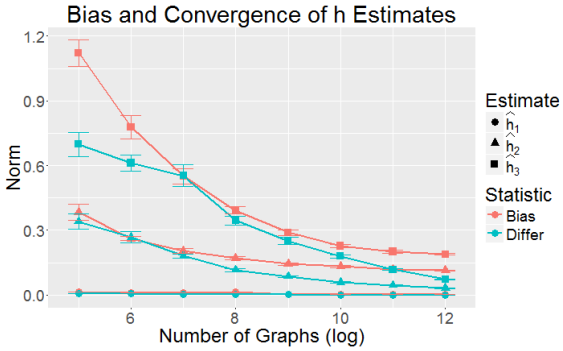


Figure 2: Mean bias ($\|\hat{h}_k^m - h_k\|$) and mean difference between estimates ($\|\hat{h}_k^m - \hat{h}_k^{m/2}\|$) across 20 simulations are shown. The standard errors are also given by error bars. The graphs are generated from a 3-dimensional MREG model as described in section 5.1. $\hat{h}_k^m$ has small asymptotic bias; however, it seems to converge as m increases.

Scott demonstrate that maximum likelihood estimator is inconsistent [39]. In our case, there are infinitely many λ_i as m grows; therefore, we do not expect the joint embedding to provide an asymptotic consistent estimate of h_k.

In applications such as clustering or classifying multiple graphs, we may be not interested in $\hat{h}_k$. $\hat{\lambda}_i$ is of primary interest, which provides information specifically about the graphs G_i. Here, we consider two approaches to estimate $\lambda_i[1]$. The first approach is estimating $\lambda_i[1]$ through joint embedding, that is

$$\hat{\lambda}_i[1] = \langle \mathbf{A}_i, \hat{h}_1^m \hat{h}_1^{mT} \rangle.$$

The second approach estimates λ_i by assuming h_1 is known. In this case, equation (4) gives

$$\hat{\lambda}_i[1] = \langle \mathbf{A}_i, h_1 h_1^T \rangle.$$

$\hat{\lambda}_i[1]$ calculated this way can be thought as the 'oracle' estimate. Figure 3 shows the differences in estimates provided by two approaches. Not surprisingly, the differences are small due to the fact that $\hat{h}_1^m$ and h_1 are close.

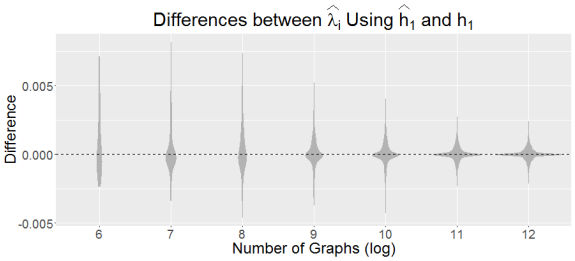


Figure 3: Distribution of differences between $\hat{\lambda}_i[1]$ estimated using $\hat{h}_1^m$ and h_1. The graphs are generated from the 3-dimensional MREG model as described in section 5.1. The differences are small due to the fact that $\hat{h}_1^m$ and h_1 are close.

5.2 Simulation Experiment 2: Joint Embedding to Classify Graphs

In this experiment, we consider the inference task of classifying graphs. We have m pairs $\{(\mathbf{A}_i, y_i)\}_{i=1}^m$ of observations.

Each pair consists of an adjacency matrix $\mathbf{A}_i \in \{0,1\}^{n\times n}$ and a label $y_i \in [K]$. Furthermore, all pairs are assumed to be independent and identically distributed according to an unknown distribution $\mathbb{F}_{\mathbf{A},y}$, that is

$$(\mathbf{A}_1, y_1), (\mathbf{A}_2, y_2), ..., (\mathbf{A}_m, y_m) \overset{i.i.d.}{\sim} \mathbb{F}_{A,y}.$$

The goal is to find a classifier g which is a function $g : \{0,1\}^{n\times n} \to [K]$ that has a small classification error $L_g = P(g(\mathbf{A}) \neq y)$.

We consider a binary classification problem where y takes value 1 or 2. 200 graphs with 100 vertices are independently generated. The graphs are sampled from a 2-dimensional MREG model. Let h_1 and h_2 be two vectors in $\mathbb{R}^{100}$, and

$$h_1 = [0.1, ..., 0.1]^T \text{ , and } h_2 = [-0.1, ..., -0.1, 0.1, ..., 0.1]^T.$$

Here, h_2 has -0.1 as its first 50 entries and 0.1 as its last 50 entries. Gaphs are generated according to the MREG model,

$$\{(\lambda_i, \mathbf{A}_i)\}_{i=1}^{200} \sim MREG(F, h_1, h_2), \tag{5}$$

where F is a mixture of two point masses with equal probability,

$$F = \frac{1}{2}\mathbb{I}\{\lambda = [25, 5]\} + \frac{1}{2}\mathbb{I}\{\lambda = [22.5, 2.5]\}.$$

We let the class label y_i indicate which point mass λ_i is sampled from. In terms of SBM, this graph generation scheme is equivalent to

$$A_i|y_i = 1 \sim SBM((1, ..., 1, 2, ..., 2), \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix})$$

$$A_i|y_i = 2 \sim SBM((1, ..., 1, 2, ..., 2), \begin{bmatrix} 0.25 & 0.2 \\ 0.2 & 0.25 \end{bmatrix}).$$

To classify graphs, we first jointly embed 200 graphs. The first two dimensional loadings are shown in Figure 4. We can see two classes are separated after being jointly embedded. Then, a 1-nearest neighbor classifier (1-NN) is

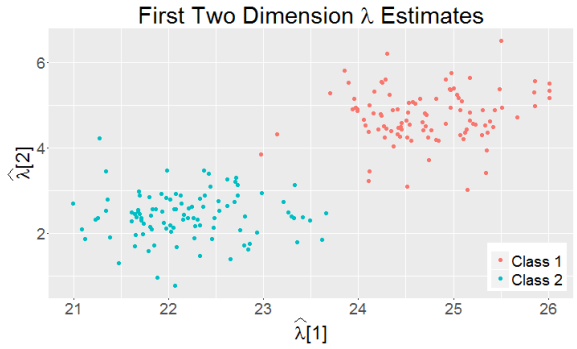


Figure 4: Scatter plot of loadings computed by jointly embedding 200 graphs. The graphs are generated from the 2-dimensional MREG model as described in equation (5). The loadings of two classes are separated after being jointly embedded.

constructed based on loadings $\{\hat{\lambda}_i\}_{i=1}^m$.

We compare classification performances of using the joint embedding to extract features to five other feature extraction approaches: Adjacency Spectral Embedding, Laplacian Eigenmap, Graph Statistics, Graph Spectral Statistics, and PCA. For Adjacency Spectral Embedding (ASE) and Laplacian Eigenmap (LE), we first embed each adjacency matrix or normalized Laplacian matrix and then compute the Procrustes distance between embeddings. For Graph Statistics (GS), we compute topological statistics of graphs considered by Park *et al.* in [7]. For Graph Spectral Statistics (GSS), we compute the eigenvalues of adjacency matrices and treat them as features [40]. For PCA, we vectorize the adjacency matrices and compute the factors through SVD. After the feature extraction step, we also apply a 1-NN rule to classify graphs. We let the number of graphs m increase from 4 to 200. For each value of m, we repeat the simulation 100 times. Figure 5 shows the result. The joint embedding takes advantage of increasing sample size and outperforms other approaches when given more than 10 graphs.

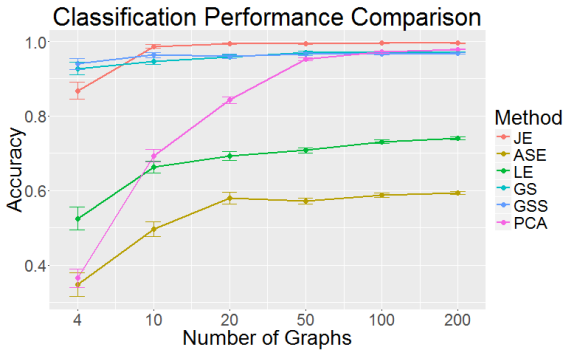


Figure 5: Mean classification accuracy of joint embedding, Adjacency Spectral Embedding, Laplacian Eigenmap, Graph Statistics, Graph Spectral Statistics, and PCA with their standard errors are shown. The graphs are generated from a 2-dimensional MREG model as described in the equation (5). The features are first extracted using methods described above; subsequently, we apply a 1-NN to classify graphs. For each value of m, the simulation is repeated 100 times. ASE, LE, GS and GSS do not take advantage of increasing sample size in the feature extraction step. PCA has poor performance when the sample sizes is small. Joint embedding takes advantage of increasing sample size and outperforms other approaches when given more than 10 graphs.

5.3 Real Data Experiment 1: Predict Composite Creativity Index

In this experiment, we study predicting individual composite creativity index (CCI) through brain connectomes obtained by Multimodal Magnetic Resonance Imaging [41]. Neuroimaging and creativity have been jointly investigated previously. Most studies utilize a statistical testing method and find CCI significantly related or inversely related to the activity of some regions of the brain. For a review, please see Arden *et al.* [42]. We embrace a different approach by directly building a prediction model for CCI. First, we jointly embed brain graphs of all subjects. Then, we

construct a linear regression model by treating the estimated loadings as explanatory variables and CCI as the response variable.

In total, 113 healthy, young adult subjects were scanned using a Siemens TrioTim scanner. 3D-MPRAGE and DTI in 35 directions of the subjects were acquired [43]. The images were then registered by Desikan-Killiany Atlas [44], and a graph of 70 vertices is constructed. The process of transforming MRI to graphs was completed by NeuroData's MRI Graphs pipeline [45]. The graphs derived have weighted edges. One example of a graph is shown in the top panel of Figure 6. For each subject, a divergent thinking measure was scored by independent judges using the Consensual Assessment Technique [46], from which the CCI is derived.

To predict the CCI, we first jointly embed 113 graphs with $d = 10$, and then fit a linear model by regressing CCI on $\hat{\lambda}_i$, that is

$$CCI_i \sim \beta_0 + \hat{\lambda}_i^T \beta + \epsilon_i.$$

We consider two linear regression models. One using only $\hat{\lambda}_i[1]$ as the explanatory variable, and another one using $\hat{\lambda}_i$ as the explanatory variables. If only the first dimensional loadings are used, the top panel of Figure 7 shows the result. There is a significant positive linear relationship between CCI and the first dimensional loadings. The first dimensional loadings generally capture the overall connectivity of graphs. In this case, the correlation between the first dimensional loadings and the sum of edge weights is around 0.98. This model implies that the individual tends to be more creative when there is more brain connectivity. The R-square of this model is 0.07248, and the model is statistically significantly better when compared to the null model with a p-value 0.0039, according to the F-test. This model suggests that individual is more creative if the brain is more connected.

If CCI is regressed on the 10 dimensional loadings, a summary of the linear model is provided in the appendix and a scatter plot of fitted CCI versus true CCI is provided in the bottom panel of Figure 7. The R-square is 0.2325 and the model is statistically significantly better than the null model with a p-value 0.0018 according to the F-test. It is also significantly better than the model with only $\hat{\lambda}_i[1]$. Although there is still a positive relationship between CCI and the first dimensional loadings, it is no longer significant due to the inclusion of more explanatory variables. In this model, there is a significant negative relationship between CCI and $\hat{\lambda}_i[6]$ based on the t-test. The scatter plot of CCI against $\hat{\lambda}_i[6]$ is given in the middle panel of Figure 7. We look into the rank one matrix $\hat{h}_6^T \hat{h}_6$, which is shown in the bottom panel of Figure 6. It has positive connectivity within each hemisphere of the brain, but negative connectivity across hemispheres. This suggests that compared to within-hemisphere connectivity, across-hemisphere connectivity tends to have a more positive impact on human creativity.

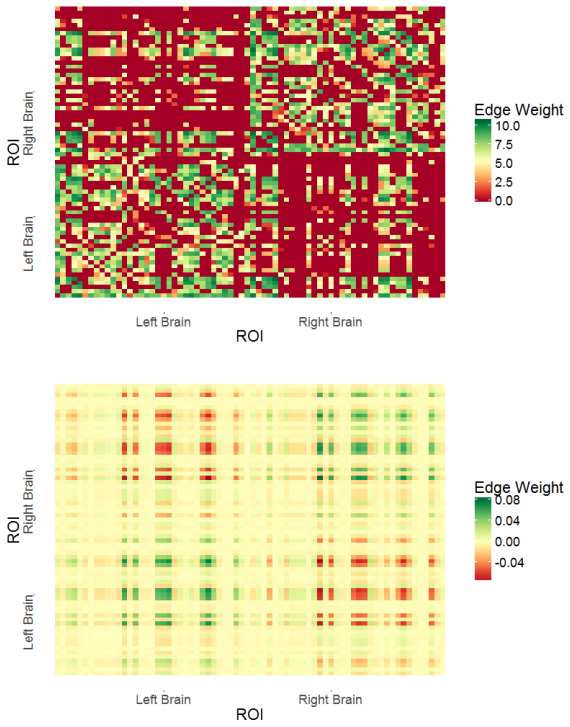


Figure 6: The top panel shows the graph derived from a typical subject. There is much more neural connectivity within each hemisphere. The bottom panel shows the rank one matrix $\hat{h}_6^T \hat{h}_6$, which has positive connectivity within each hemisphere, but negative connectivity across hemispheres.

5.4 Real Data Experiment 2: Joint Embedding to Cluster Vertices

In the previous experiments, we focus on feature extraction for graphs through the joint embedding. Here, we consider a different task, that is spectral clustering through the joint embedding. In general, spectral clustering first computes (generalized) eigenvalues and eigenvectors of adjacency matrix or Laplacian matrix, then clustering the latent positions into groups [11], [12]. The cluster identities of latent positions become the cluster identities of vertices of the original graph. When applied to one graph, the joint embedding is equivalent to Adjacency Spectral Embedding (ASE), which is one of the spectral clustering algorithms. When given multiple graphs, the joint embedding can estimate latent positions for graph i as $[\hat{\lambda}_i[1]^{\frac{1}{2}}\hat{h}_1, \hat{\lambda}_i[2]^{\frac{1}{2}}\hat{h}_2, ..., \hat{\lambda}_i[d]^{\frac{1}{2}}\hat{h}_d]$ or equivalently $\hat{\mathbf{H}}\hat{\mathbf{D}}_i^{\frac{1}{2}}$. Then, clustering algorithm can be applied to the latent positions.

We apply the spectral clustering approach to Wikipedia graphs [47]. The vertices of these graphs represent Wikipedia article pages. The two vertices are connected by an edge if either of the associated pages hyperlinks to the other. Two graphs are constructed based on English webpages and French webpages. The full graph has 1382 vertices which represents articles within 2-neighborhood of "Algebraic Geometry". Based on the content of the associated articles, they are grouped by hand into 6

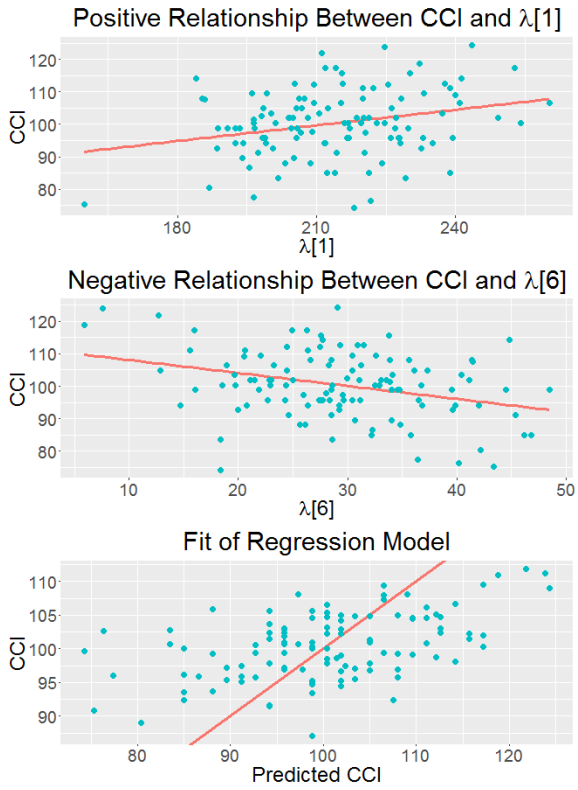


Figure 7: The top panel shows the scatter plot of CCI against $\hat{\lambda}_i[1]$ with the regression line. There is a positive relationship between CCI and first dimensional loadings. The middle panel shows the scatter plot of CCI against $\hat{\lambda}_i[6]$ with regression line. There is a negative relationship between CCI and sixth dimensional loadings. The bottom panel shows the predicted CCI versus true CCI with the identity line.

categories: People, Places, Dates, Things, Math Terms, and Categories.

We consider a subset vertices from 3 categories: People, Things, Math Terms. After taking the induced subgraph of these vertices and removing isolated vertices, there are $n = 704$ vertices left. Specifically, 326, 181, and 197 vertices are from People, Things and Math Terms respectively. We consider 4 approaches to embed the graphs to obtain latent positions: ASE on the English graph (ASE+EN), ASE on the French Graph (ASE+FR), joint embedding on the English graph (JE+EN), and joint embedding on the French Graph (JE+FR). The dimension d is set to 3 for all approaches, and the latent positions are scaled to have norm 1 for degree correction. Then, we apply 3-means to the latent positions.

The latent positions of English graph estimated based on the joint embedding is provided in Figure 8. The latent positions of Math Terms are separated from the other two clusters. However, the latent positions of People and Things are mixed. Table 1 shows the clustering results measured by adjusted rand index and purity [48], [49]. The English graph has clearer community structure than the French Graph. The clustering performance based on latent positions estimated through joint embedding is better. We expect joint embedding to be even better when given more graphs.

Method	ASE+EN	ASE+FR	JE+EN	JE+FR
ARI	0.147	0.115	**0.158**	0.156
Purity	0.549	0.520	**0.551**	0.549

TABLE 1: Clustering Performance on Wikipedia Graphs. The adjusted rand index (ARI) and purity of 4 spectral clustering approaches are shown. The best result is bolded. The joint embedding estimates latent positions which lead to better clustering performance than ASE.

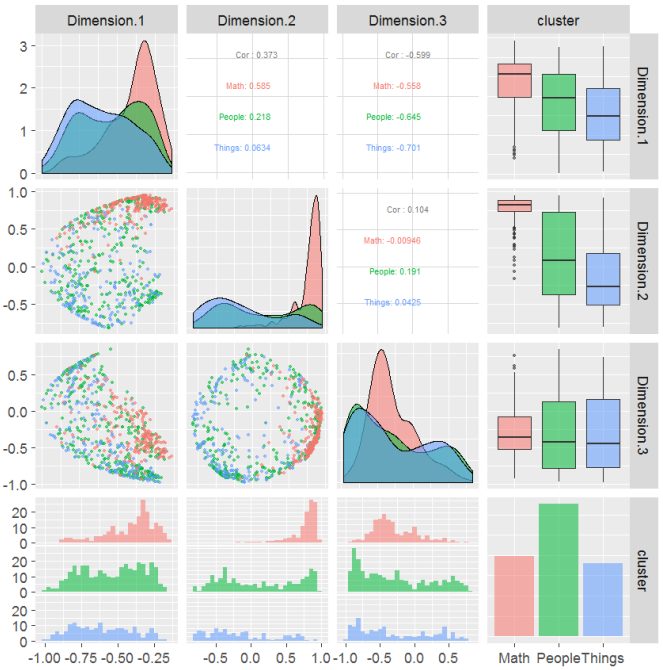


Figure 8: The latent positions of English Graph estimated by the joint embedding are shown. The first three plots on the diagonal are density estimates of latent positions for each dimension and category, and the last plot shows the number of points from each category. The first three plots of the last row show the histogram of latent positions for each dimension and category, and the first three plots of the last column are the corresponding box plot. The pairs plots of latent positions are given in the first three plots below the diagonal, the corresponding correlations are given above the diagonal. The latent positions of Math Terms are separated from the other two clusters. However, the latent positions of People and Things are mixed.

6 CONCLUSION

In summary, we developed a joint embedding method that can simultaneously embed multiple graphs into low dimensional space. The joint embedding can be utilized to estimate features for inference problems on multiple vertex matched graphs. Learning on multiple graphs has significant applications in diverse fields and our results have both theoretical and practical implications for the problem. As the real data experiment illustrates, the joint embedding is

a practically viable inference procedure. We also proposed a Multiple Random Eigen Graphs model. It can be understood as a generalization of the Random Dot Product Graph model or the Stochastic Block Model for multiple random graphs. We analyzed the performance of joint embedding on this model under simple settings. We demonstrated that the joint embedding method provides estimates with bounded error. Our approach is intimately related to other matrix and tensor factorization approaches such as singular value decomposition and CP decomposition. Indeed, the joint embedding and these algorithms all try to estimate a low dimensional representation of high dimensional objects through minimizing a reconstruction error. We are currently investigating the utility of joint embedding with more or less regularizations on parameters and under different set ups. We are optimistic that our method provides a viable tool for analyzing multiple graphs and can contribute to a deeper understanding of the joint structure of networks.

APPENDIX A

Proof of Theorem 2.1 Denote the probability of observing a particular adjacency matrix $\mathbf{A}_i$ under distribution $\mathcal{F}$ by p_i. It suffices to show that there is a set of parameters for MREG such that observing $\mathbf{A}_i$ under MREG is also p_i.

For undirected graphs with loops on n vertices, there are $\binom{n+1}{2}$ possible edges. Let $\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_{2^{\binom{n+1}{2}}}$ be all the possible adjacency matrices. Since real symmetric matrix of size n has $\binom{n+1}{2}$ free entries which lies in a linear space, if there exists $\binom{n+1}{2}$ linearly independent rank one symmetric matrices, they form a basis for this space. It turns out that the rank one symmetric matrices generated by vectors $\{e_i\}_{i=1}^n \cup \{e_i + e_j\}_{i<j}$ are linearly independent, where $\{e_i\}_{i=1}^n$ is the standard basis for n-dimensional Euclidean space.

Next, we construct parameters for the MREG. Let d be $\binom{n+1}{2}$ and

$$\{h_k\}_{k=1}^d = \{e_i\}_{i=1}^n \cup \{\frac{e_i + e_j}{\sqrt{2}}\}_{i<j}.$$

Since $\{h_k h_k^T\}_{k=1}^d$ forms a basis for real symmetric matrices, for each adjacency matrix $\mathbf{A}_i$, there exists a vector λ_i, such that

$$\mathbf{A}_i = \sum_k \lambda_i[k] h_k h_k^T.$$

Let F be a finite mixture distribution on points $\{\lambda_i\}_{i=1}^{2^{\binom{n+1}{2}}}$, that is

$$F = \sum p_i \mathbb{I}\{\lambda = \lambda_i\}.$$

Under this MREG model, for any adjacency matrix $\mathbf{A}_i$

$$P(\mathbf{A} = \mathbf{A}_i) = P(\lambda = \lambda_i) = p_i.$$

This concludes that the distribution $\mathcal{F}$ and $MREG(F, h_1, ..., h_d)$ are equal.

Proof of Theorem 4.1 First, we show that $|D_n(h, h_1) - D(h, h_1)|$ converges uniformly to 0. To begin with, notice three facts:

(1) the set $\{h : \|h\| = 1\}$ is compact;
(2) for all h, the function $\rho(\cdot, h)$ is continuous
(3) for all h, the function $\rho(\cdot, h)$ is bounded by n^2.

Therefore, by the uniform law of large numbers [50], we have

$$\sup_h |D_m(h, h_1) - D(h, h_1)| \overset{a.s.}{\to} 0.$$

To prove the claim of the theorem, we use a technique similar to that employed by Bickel and Doksum [51]. By definition, we must have $D_m(\hat{h}_1^m, h) \leq D_m(h', h)$ and $D(h', h) \leq D(\hat{h}_1^m, h)$. From these two inequalities,

$$\begin{aligned} D_m(h', h) - D(h', h) &\geq D_m(\hat{h}_1^m, h) - D(h', h) \\ &\geq D_m(\hat{h}_1^m, h) - D(\hat{h}_1^m, h). \end{aligned}$$

Therefore,

$$\begin{aligned} |D_m(\hat{h}_1^m, h) - D(h', h)| \leq \max(&|D_m(h', h) - D(h', h)|, \\ &|D_m(\hat{h}_1^m, h) - D(\hat{h}_1^m, h)|). \end{aligned}$$

This implies

$$|D_m(\hat{h}_1^m, h) - D(h', h)| \leq \sup_h |D_m(h, h_1) - D(h, h_1)|.$$

Hence, $|D_m(\hat{h}_1^m, h) - D(h', h)|$ must converge almost surely to 0, that is

$$|D_m(\hat{h}_1^m, h) - D(h', h)| \overset{a.s.}{\to} 0.$$

If $\hat{h}_1^m$ does not converge almost surely to h', then $\|\hat{h}_1^m - h'\| \geq \epsilon$ for some ϵ and infinitely many values of m. Since h' is the unique global minimum, $|D(\hat{h}_1^m, h) - D(h', h)| > \epsilon'$ for infinitely many values of m and some ϵ'. This contradicts with the previous equation. Therefore, $\hat{h}_1^m$ must converge almost surely to h'.

Proof of Theorem 4.2 The proof of theorem relies on two lemmas. The first lemma shows that h' is the eigenvector corresponding to the largest eigenvalue of $E(\langle \mathbf{A}_i, h'h'^T \rangle \mathbf{A}_i)$. The second lemma shows that $E(\langle \mathbf{A}_i, h'h'^T \rangle \mathbf{A}_i)$ is close to $E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T$ under Frobenius norm. Then, we apply Davis-Kahan theorem [52] to establish the result of theorem.

Lemma 6.1. The vector h' is the eigenvector corresponding to the largest eigenvalue of $E(\langle \mathbf{A}_i, h'h'^T \rangle \mathbf{A}_i)$.

We notice that

$$\begin{aligned} \min_{\|h\|=1} D(h, h_1) &= \min_{\|h\|=1} E(\|\mathbf{A}_i - \langle \mathbf{A}_i, hh^T \rangle hh^T\|^2) \\ &= \min_{\|h\|=1} E(\langle \mathbf{A}_i, \mathbf{A}_i \rangle - \langle \mathbf{A}_i, hh^T \rangle^2) \\ &= E(\langle \mathbf{A}_i, \mathbf{A}_i \rangle) - \max_{\|h\|=1} E(\langle \mathbf{A}_i, hh^T \rangle^2). \end{aligned}$$

Therefore,

$$h' = \operatorname*{argmin}_{\|h\|=1} D(h, h_1) = \operatorname*{argmax}_{\|h\|=1} E(\langle \mathbf{A}_i, hh^T \rangle^2). \quad (6)$$

Taking the derivative of $E(\langle \mathbf{A}_i, hh^T \rangle^2) + c(h^T h - 1)$ with respect to h,

$$\begin{aligned} \frac{\partial E(\langle \mathbf{A}_i, hh^T \rangle^2) + c(h^T h - 1)}{\partial h} &= E(\frac{\partial \langle \mathbf{A}_i, hh^T \rangle^2}{\partial h}) + 2ch \\ &= 4E(\langle \mathbf{A}_i, hh^T \rangle \mathbf{A}_i)h + 2ch. \end{aligned}$$

Setting this expression to 0 yields,

$$E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)h' = -\frac{1}{2}ch'.$$

Using the fact that $\|h'\| = 1$, we can solve for c:

$$c = -2h'^T E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)h' = -2E(\langle \mathbf{A}_i, h'h'^T\rangle^2).$$

Then, substituting for c,

$$E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)h' = E(\langle \mathbf{A}_i, h'h'^T\rangle^2)h'. \tag{7}$$

Therefore, we see that h' is an eigenvector of $E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)$ and the corresponding eigenvalue is $E(\langle \mathbf{A}_i, h'h'^T\rangle^2)$. Furthermore, $E(\langle \mathbf{A}_i, h'h'^T\rangle^2)$ must be the eigenvalue with the largest magnitude. For if not, then there exists an h'' with norm 1 such that

$$\begin{aligned}|h''^T E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)h''| &= |E(\langle \mathbf{A}_i, h'h'^T\rangle \langle \mathbf{A}_i, h''h''^T\rangle)| \\ &> E(\langle \mathbf{A}_i, h'h'^T\rangle^2);\end{aligned}$$

however, by Cauchy-Schwarz inequality we must have

$$\begin{aligned}E(\langle \mathbf{A}_i, h''h''^T\rangle^2)E(\langle \mathbf{A}_i, h'h'^T\rangle^2) > \\ |E(\langle \mathbf{A}_i, h'h'^T\rangle \langle \mathbf{A}_i, h''h''^T\rangle)|^2,\end{aligned}$$

implying $E(\langle \mathbf{A}_i, h''h''^T\rangle^2) > E(\langle \mathbf{A}_i, h'h'^T\rangle^2)$, which contradicts equation (6). Thus, we conclude that h' is the eigenvector corresponding to the largest eigenvalue of $E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)$.

Lemma 6.2.

$$\|E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i) - E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T\| \le 2E(\lambda_i).$$

We compute $E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)$ by conditioning on P_i.

$$\begin{aligned}&E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i|\mathbf{P}_i) \\ &= E(\langle \mathbf{A}_i - \mathbf{P}_i, h'h'^T\rangle (\mathbf{A}_i - \mathbf{P}_i)|\mathbf{P}_i) \\ &\quad + E(\langle \mathbf{A}_i - \mathbf{P}_i, h'h'^T\rangle \mathbf{P}_i)|\mathbf{P}_i) \\ &\quad + E(\langle \mathbf{P}_i, h'h'^T\rangle (\mathbf{A}_i - \mathbf{P}_i)|\mathbf{P}_i) + E(\langle \mathbf{P}_i, h'h'^T\rangle \mathbf{P}_i|\mathbf{P}_i) \\ &= E(\langle \mathbf{A}_i - \mathbf{P}_i, h'h'^T\rangle (\mathbf{A}_i - \mathbf{P}_i)|\mathbf{P}_i) + \lambda_i (h_1^T h')^2 \mathbf{P}_i \\ &= 2h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i) - DIAG(h_1 h_1^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i)) \\ &\quad + \lambda_i (h_1^T h')^2 \mathbf{P}_i.\end{aligned}$$

Here, DIAG() means only keep the diagonal of the matrix; $*$ means the Hadamard product, and $\mathbf{J}$ is a matrix of all ones. Using the fact that $\mathbf{P}_i = \lambda_i h_1 h_1^T$, we have

$$\begin{aligned}&E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i) - E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T \\ &= E(E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i|\mathbf{P}_i) - \lambda_i (h_1^T h')^2 \mathbf{P}_i) \\ &= E(2h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i) \\ &\quad - DIAG(h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i))).\end{aligned}$$

If we consider the norm difference between $E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)$ and $E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T$, we have

$$\begin{aligned}&\|E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i) - E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T\| \\ &= \|E(2h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i) \\ &\quad - DIAG(h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i)))\| \\ &\le E(\|2h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i) \\ &\quad - DIAG(h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i))\|) \\ &\le E(\|2h'h'^T * \mathbf{P}_i * (\mathbf{J} - \mathbf{P}_i)\|) \\ &\le E(\|2h'h'^T * \mathbf{P}_i\|) \\ &\le 2E(\lambda_i)\|h'h'^T * h_1 h_1^T\| \\ &= 2E(\lambda_i).\end{aligned}$$

This finishes the proof for the lemma.

Notice that the only non-zero eigenvector of $E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T$ is h_1 and the corresponding eigenvalue is $E(\lambda_i^2)(h_1^T h')^2$. We apply the Davis-Kahan theorem [52] to the eigenvector corresponding to the largest eigenvalue of matrices $E(\langle \mathbf{A}_i, h'h'^T\rangle \mathbf{A}_i)$ and $E(\lambda_i^2)(h_1^T h')^2 h_1 h_1^T$, yielding

$$\|h' - h_1\| \le \frac{2E(\lambda_i)}{E(\lambda_i^2)(h_1^T h')^2}.$$

APPENDIX B

Linear Regression Model Summary

```
> model<-lm(cci~Lambda+1)
> summary(model)

Call:
lm(formula = cci ~ Lambda + 1)

Residuals:
Min        1Q    Median        3Q       Max
-26.3432  -6.144  -0.7578  7.1032   16.9004

Coefficients: Estimate  Pr(>|t|)
(Intercept)  1.275e+02   0.000275  ***
Lambda1      2.421e-04   0.997981
Lambda2     -2.326e-01   0.070110  .
Lambda3     -3.716e-02   0.822592
Lambda4      8.049e-02   0.687628
Lambda5     -2.925e-01   0.421858
Lambda6     -4.285e-01   0.009088  **
Lambda7     -1.745e-01   0.590533
Lambda8     -3.465e-01   0.240093
Lambda9     -8.970e-01   0.007999  **
Lambda10    -8.955e-01   0.052839  .
---
Signif. codes:  0 *** 0.001 **
0.01 * 0.05 . 0.1   1

Residual standard error: 9.437
on 102 degrees of freedom
Multiple R-squared:  0.2325,
Adjusted R-squared:  0.1572
F-statistic:  3.09 on 10 and 102 DF,
p-value: 0.001795
```

REFERENCES

[1] E. Otte and R. Rousseau, "Social network analysis: a powerful strategy, also for the information sciences," *Journal of information Science*, vol. 28, no. 6, pp. 441–453, 2002.

[2] R. Govindan and H. Tangmunarunkit, "Heuristics for internet map discovery," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3. IEEE, 2000, pp. 1371–1380.

[3] E. T. Bullmore and D. S. Bassett, "Brain graphs: graphical models of the human brain connectome," *Annual review of clinical psychology*, vol. 7, pp. 113–140, 2011.

[4] M. D. Ward, K. Stovel, and A. Sacks, "Network analysis and political science," *Annual Review of Political Science*, vol. 14, pp. 245–264, 2011.

[5] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[6] G. Li, M. Semerci, B. Yener, and M. J. Zaki, "Graph classification via topological and label attributes," in *Proceedings of the 9th international workshop on mining and learning with graphs (MLG), San Diego, USA*, vol. 2, 2011.

[7] Y. Park, C. E. Priebe, and A. Youssef, "Anomaly detection in time series of graphs using fusion of graph invariants," *IEEE journal of selected topics in signal processing*, vol. 7, no. 1, pp. 67–75, 2013.

[8] C. Jiang, F. Coenen, and M. Zito, "A survey of frequent subgraph mining algorithms," *The Knowledge Engineering Review*, vol. 28, no. 01, pp. 75–105, 2013.

[9] J. Huan, W. Wang, and J. Prins, "Efficient mining of frequent subgraphs in the presence of isomorphism," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 2003, pp. 549–552.

[10] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 1151–1157.

[11] D. L. Sussman, M. Tang, D. E. Fishkind, and C. E. Priebe, "A consistent adjacency spectral embedding for stochastic blockmodel graphs," *Journal of the American Statistical Association*, vol. 107, no. 499, pp. 1119–1128, 2012.

[12] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[13] S. J. Young and E. R. Scheinerman, "Random dot product graph models for social networks," in *Algorithms and models for the web-graph*. Springer, 2007, pp. 138–149.

[14] M. Tang, D. L. Sussman, C. E. Priebe *et al.*, "Universally consistent vertex classification for latent positions graphs," *The Annals of Statistics*, vol. 41, no. 3, pp. 1406–1430, 2013.

[15] D. L. Sussman, M. Tang, and C. E. Priebe, "Consistent latent position estimation and vertex classification for random dot product graphs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 1, pp. 48–57, 2014.

[16] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social networks*, vol. 5, no. 2, pp. 109–137, 1983.

[17] P. ERDdS and A. R&WI, "On random graphs i," *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.

[18] B. Karrer and M. E. Newman, "Stochastic blockmodels and community structure in networks," *Physical Review E*, vol. 83, no. 1, p. 016107, 2011.

[19] V. Lyzinski, M. Tang, A. Athreya, Y. Park, and C. E. Priebe, "Community detection and classification in hierarchical stochastic blockmodels," *arXiv preprint arXiv:1503.02115*, 2015.

[20] J. Yan, Y. Li, W. Liu, H. Zha, X. Yang, and S. M. Chu, "Graduated consistency-regularized optimization for multi-graph matching," in *European Conference on Computer Vision*. Springer, 2014, pp. 407–422.

[21] H.-M. Park and K.-J. Yoon, "Encouraging second-order consistency for multiple graph matching," *Machine Vision and Applications*, vol. 27, no. 7, pp. 1021–1034, 2016.

[22] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[23] B. N. Flury and W. Gautschi, "An algorithm for simultaneous orthogonal transformation of several positive definite symmetric matrices to nearly diagonal form," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 1, pp. 169–184, 1986.

[24] A. Ziehe, P. Laskov, G. Nolte, and K.-R. MÃžller, "A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation," *Journal of Machine Learning Research*, vol. 5, no. Jul, pp. 777–800, 2004.

[25] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[26] W. Tang, Z. Lu, and I. S. Dhillon, "Clustering with multiple graphs," in *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE, 2009, pp. 1016–1021.

[27] T. G. Kolda, "Numerical optimization for symmetric tensor decomposition," *Mathematical Programming*, vol. 151, no. 1, pp. 225–248, 2015.

[28] J. C. Bezdek and R. J. Hathaway, "Convergence of alternating optimization," *Neural, Parallel & Scientific Computations*, vol. 11, no. 4, pp. 351–368, 2003.

[29] S. J. Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3–34, 2015.

[30] A. Beck and L. Tetruashvili, "On the convergence of block coordinate descent type methods," *SIAM journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.

[31] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[32] N. Bell and M. Garland, "Implementing sparse matrix-vector multiplication on throughput-oriented processors," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*. ACM, 2009, p. 18.

[33] H. Kim and H. Park, "Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method," *SIAM journal on matrix analysis and applications*, vol. 30, no. 2, pp. 713–730, 2008.

[34] M. Aharon, M. Elad, and A. Bruckstein, "rmk-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[35] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[36] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[37] A. Athreya, C. Priebe, M. Tang, V. Lyzinski, D. Marchette, and D. Sussman, "A limit theorem for scaled eigenvectors of random dot product graphs," *Sankhya A*, pp. 1–18, 2013.

[38] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, no. 2, 1995, pp. 1137–1145.

[39] J. Neyman and E. L. Scott, "Consistent estimates based on partially consistent observations," *Econometrica: Journal of the Econometric Society*, pp. 1–32, 1948.

[40] S. N. Dorogovtsev, A. V. Goltsev, J. F. Mendes, and A. N. Samukhin, "Spectra of complex networks," *Physical Review E*, vol. 68, no. 4, p. 046109, 2003.

[41] D. Koutra, J. T. Vogelstein, and C. Faloutsos, "D elta c on: A principled massive-graph similarity function," in *Proceedings of the SIAM International Conference in Data Mining. Society for Industrial and Applied Mathematics*. SIAM, 2013, pp. 162–170.

[42] R. Arden, R. S. Chavez, R. Grazioplene, and R. E. Jung, "Neuroimaging creativity: a psychometric view," *Behavioural brain research*, vol. 214, no. 2, pp. 143–156, 2010.

[43] M. Brant-Zawadzki, G. D. Gillan, and W. R. Nitz, "Mp rage: a three-dimensional, t1-weighted, gradient-echo sequence–initial experience in the brain." *Radiology*, vol. 182, no. 3, pp. 769–775, 1992.

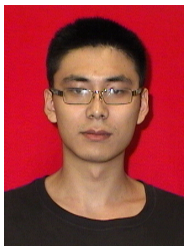
[44] R. S. Desikan, F. Ségonne, B. Fischl, B. T. Quinn, B. C. Dickerson, D. Blacker, R. L. Buckner, A. M. Dale, R. P. Maguire, B. T. Hyman *et al.*, "An automated labeling system for subdividing the human cerebral cortex on mri scans into gyral based regions of interest," *Neuroimage*, vol. 31, no. 3, pp. 968–980, 2006.

[45] G. Kiar, W. Gray Roncal, D. Mhembere, E. Bridgeford, R. Burns, and J. Vogelstein, "ndmg: Neurodata's mri graphs pipeline," Aug. 2016, open-source code. [Online]. Available: http://dx.doi.org/10.5281/zenodo.60206

[46] T. M. Amabile, "The social psychology of creativity: A componential conceptualization." *Journal of personality and social psychology*, vol. 45, no. 2, p. 357, 1983.

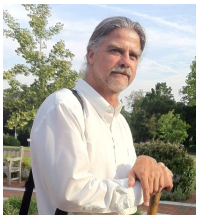
[47] S. Suwan, D. S. Lee, R. Tang, D. L. Sussman, M. Tang, C. E. Priebe *et al.*, "Empirical bayes estimation for the stochastic blockmodel," *Electronic Journal of Statistics*, vol. 10, no. 1, pp. 761–782, 2016.

[48] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004.
[49] E. Rendón, I. Abundez, A. Arizmendi, and E. Quiroz, "Internal versus external cluster validation indexes," *International Journal of computers and communications*, vol. 5, no. 1, pp. 27–34, 2011.
[50] R. I. Jennrich, "Asymptotic properties of non-linear least squares estimators," *The Annals of Mathematical Statistics*, vol. 40, no. 2, pp. 633–643, 1969.
[51] P. J. Bickel and K. A. Doksum, *Mathematical Statistics: Basic Ideas and Selected Topics, volume I*. CRC Press, 2015, vol. 117, ch. 6.
[52] C. Davis and W. M. Kahan, "The rotation of eigenvectors by a perturbation. iii," *SIAM Journal on Numerical Analysis*, vol. 7, no. 1, pp. 1–46, 1970.



Shangsi Wang received the BS degree in mathematics and actuarial science from University of Waterloo in 2012, the MA degree from the Department of Applied Mathematics and Statistics at Johns Hopkins University, in 2014. He is currently a graduate student at the Johns Hopkins University, working toward the PhD degree in statistics. His research interests include statistical inference on random netowrks and pattern recognition in graph datasets.

Joshua T. Vogelstein received the BS degree from the Department of Biomedical Engineering (BME) at Washington University in St. Louis, in 2002, the MS degree from the Department of Applied Mathematics and Statistics at Johns Hopkins University (JHU), in 2009, and the PhD degree from the Department of Neuroscience at JHU in 2009. He was a postdoctoral fellow in AMS at JHU from 2009 until 2011, when he was appointed an assistant research scientist, and became a member of the Institute for Data Intensive Science and Engineering. He spent two years at Information Initiative at Duke, before becoming Assistant Professor in BME at JHU, and core faculty in the Institute for Computational Medicine and the Center for Imaging Science. His research interests include computational statistics, focusing on ultrahigh-dimensional and non-Euclidean neuroscience data, especially connectomics.



Carey E. Priebe received the BS degree in mathematics from Purdue University, in 1984, the MS degree in computer science from San Diego State University, in 1988, and the PhD degree in information technology (computational statistics) from George Mason University, in 1993. From 1985 to 1994, he worked as a mathematician and scientist in the US Navy research and development laboratory system. Since 1994, he has been a professor in the Department of Applied Mathematics and Statistics, Johns Hopkins University (JHU). His research interests include computational statistics, kernel and mixture estimates, statistical pattern recognition, statistical image analysis, dimensionality reduction, model selection, and statistical inference for high-dimensional and graph data. He is a Senior Member of the IEEE, a Lifetime Member of the Institute of Mathematical Statistics, an Elected Member of the International Statistical Institute, and a Fellow of the American Statistical Association.