An Iterative Spectral Approach to Recovering Planted Partitions

by

Samuel Cole
B.A. (Oberlin College) 2009
M.S. (University of Illinois at Chicago) 2011

Thesis submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mathematics
in the Graduate College of the
University of Illinois at Chicago, 2018

Chicago, Illinois

Defense Committee:
Shmuel Friedland, Chair and Advisor
Benjamin Antieau
Lek-Heng Lim (University of Chicago)
Lev Reyzin
György Turán

To my best friends and parents,

Pat & Steve Cole

ACKNOWLEDGMENTS

Thanks to all my professors at UIC and Oberlin for teaching me almost everything I know about math and computer science. I would not have made it this far without each and every one of you. I would like to give a shout out to several of you in particular.

First and foremost, I would like to thank my advisor, Shmuel Friedland. You have been a great mentor, collaborator, and friend to me throughout my time at UIC. You have taught me a great deal about linear algebra, graph theory, optimization, fine wine, and living life to the fullest. I still have much to learn from you, and I hope we will continue to work together for many years. I cannot imagine a better advisor, and I will truly miss being your student!

I would also like to extend a special thank you to Lev Reyzin, for acting as my pseudo-advisor and helping me navigate the perils of academia. Collaborating with you and being in your classes were among the most rewarding experiences of my grad school years—even our ill-fated statistical dimension project. Thank you especially for being available over the past few months to answer my annoying questions about job applications, writing and defending my thesis, and academic life in general.

In addition to Shmuel and Lev, I would like to thank Dhruv Mubayi and Gyuri Turán. You are both excellent professors and played a major role in my combinatorial and computer science education. I would also like to thank my undergrad professors Alexa Sharp and Kevin Woods for introducing me to math and computer science research.

ACKNOWLEDGMENTS (Continued)

Thanks to my committee—Shmuel Friedland, Ben Antieau, Lek-Heng Lim, Lev Reyzin, and Gyuri Turán—for taking the time to read my thesis and participating in my defense. I would also like to thank Maureen Madden for helping me make sense of some of the administrative aspects of submitting and defending my thesis. Thanks to all of you for accommodating my time constraints.

I would like to thank/blame my friend Joe Kramer-Miller for inspiring me to leave my cushy industry job and pursue a math Ph.D., and I would like to thank my UIC friends, Alex Cameron, Ben Fish, Maxwell Levine, Yi Lin, Tasos Moulinos, Venu Tammali, and Sam Ziegler for getting me through it.

Finally, I would like to thank my parents for supporting me in everything I do.

CONTRIBUTION OF AUTHORS

Chapter 3 is based on collaboration with Shmuel Friedland and Lev Reyzin (1). See Appendix B for copyright information.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

a.s.	Almost surely
iff.	If and only if
iid	Independent, identically distributed
MLE	Maximum likelihood estimator
s.t.	Subject to
SVD	Singular value decomposition
UIC	University of Illinois at Chicago
w.h.p.	With high probability

SUMMARY

In the *planted partition problem*, the n vertices of a random graph are partitioned into k "clusters," and edges between vertices in the same cluster and different clusters are included with constant probability p and q, respectively (where $0 \leq q < p \leq 1$). In this work, we give an efficient spectral algorithm that recovers the clusters with high probability, provided that the sizes of any two clusters are either very close or separated by $\geq \Omega(\sqrt{n})$. The algorithm recovers the clusters one by one via *iterated projection*: it constructs the orthogonal projection operator onto the *dominant k-dimensional eigenspace* of the random graph's adjacency matrix, uses it to recover one of the clusters, then deletes it and recurses on the remaining vertices.

CHAPTER 1

INTRODUCTION

Copyright note: Parts of this introduction were taken from my papers (1; 2). See Appendix B for copyright information.

A central problem in data science is that of *community detection*. In this problem, one wishes to divide the nodes of a network into *communities* or *clusters* which are "more connected" than the network as a whole. For example, we can consider Facebook® to be a network in which the nodes are users, and two nodes are "connected" if those two users are friends. Our task is then to divide the users into groups within which there are more connections (i.e. friendships) relative to the size of the groups than in the network as a whole. In other words, we wish to partition the users into groups of high *density* compared to that of the network overall. See (3) for a survey of community detection.

A well-studied mathematical model for community detection is the *planted partition problem*, first introduced in the 1980s (4; 5; 6). In this model,the n vertices of a random graph are partitioned into k unknown clusters $C_1, \ldots, C_k$, and edges added independently with probabilities p and q between pairs of vertices in the same cluster and different clusters, respectively, with $0 \leq q < p \leq 1$ fixed. The problem is then to *recover* the unknown partition given only this randomly generated graph—i.e., to determine *exactly* which pairs of vertices are in the same cluster and which pairs are in different clusters with probability tending to 1 as $n \to \infty$.

In this thesis, which is based on previous work with Shmuel Friedland and Lev Reyzin (1; 2), we give an efficient spectral algorithm (i.e., based on the eigenvalues and eigenvectors of the random graph's adjacency matrix) which accomplishes this when the clusters are all size $s = \Omega(\sqrt{n})$ (Chapter 3), and then we show how to adapt the algorithm to a more general setting in which the clusters need not be the same size (Chapter 4). Our algorithm is far from the first algorithm for planted partition, but it compares favorably to other spectral algorithms. To the best of our knowledge, it is the first algorithm which:

- Is purely spectral. Many other polynomial-time algorithms utilize convex/semidefinite programming (7; 8; 9).
- Handles the case when all clusters are size $s = \Theta(\sqrt{n})$. The well-known spectral algorithms (10; 11; 12) require that $k = o(\sqrt{n})$ and hence do not work when all clusters are size $\Theta(\sqrt{n})$ (though they work in considerably more general settings than ours).
- Has worst-case running time which is polynomial in n. Giesen and Mitsche's algorithm (13) satisfies both of the previous criteria but has running time exponential in k.

In addition, our algorithm is very simple and intuitive compared to (11; 12; 13): all of these randomly bipartition the vertices of the graph before performing a projection, and hence must merge the parts back together at the end.

$\Omega(\sqrt{n})$ cluster size is generally accepted to be the barrier for efficient algorithms for "planted" problems. There is much evidence of this for the for the simpler problem *planted clique* (14;

15; 16), in which only a single large clique is planted in a random graph. Evidence for the difficulty of beating the $\sqrt{n}$ barrier dates back to Jerrum (15), who showed a specific Markov chain approach will fail to find smaller cliques. Feige and Krauthgamer (17) showed that Lovász-Schrijver SDP relaxations run into the same barrier, while Feldman et al. (18) show that all "statistical algorithms" also provably fail to efficiently find smaller cliques in a distributional version of the planted clique problem. These lower bounds all pertain to planted clique, while there seem to be relatively few such results for planted partition; this is a possible direction for future work.

1.1 Problem definition

We now formally define the planted partition problem. We refer the reader to Chapter 2 and Appendix A for relevant definitions and notation.

Copyright note: This is taken from (2, Section 2). See Appendix B for copyright information.

Definition 1 (Planted partition/stochastic block model)**.** *Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be a partition of the set $[n] := \{1, \ldots, n\}$ into k sets called* clusters*, with $|C_i| =: s_i$ for $i = 1, \ldots, k$. For constants $0 \leq q < p \leq 1$, we define the* planted partition model *(also known as the* stochastic block model*) to be the probability distribution $\mathcal{G}(n, \mathcal{C}, p, q)$ of graphs with vertex set $[n]$, with edges uv (for $u \neq v$) included independently with probability p if u and v are in the same cluster in $\mathcal{C}$ and probability q otherwise.*

See Figure 1. Note that the case $k = 1$ gives the standard Erdős-Rényi model $\mathcal{G}(n, p)$ (19), and the case $k = n$ gives $\mathcal{G}(n, q)$.

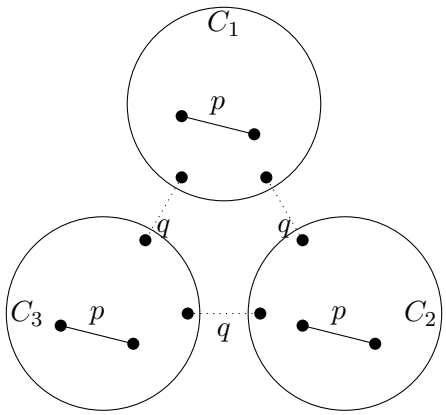

Figure 1. An illustration of the planted partition model. Edges between two vertices in the same cluster are added with probability p, while edges between two vertices in different clusters are added with probability q.

Problem 1 (Planted partition)**.** *Identify (or* recover*) the unknown partition* $\mathcal{C} = \{C_1, \ldots, C_k\}$ *(up to a permutation of* $[k]$*) given only a random graph* $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, p, q)$*. Equivalently, determine* $H(\mathcal{C})$ *(see Definition 4) given only* $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, p, q)$*.*

Observe that, by considering the adjacency matrix of $\hat{G}$, we can think of this as a problem about random symmetric matrices whose above-diagonal entries are independent Bernoulli random variables.

1.2 Notation

We will denote as follows the main quantities to consider in this thesis.

- n – the number of vertices.
- p, q – edge probabilities. Except when stated otherwise, we will assume that p and q are constant and $0 \leq q < p \leq 1$.
- $\mathcal{C} = \{C_1, \ldots, C_k\}$ – a partition of the vertex set $[n]$.
- $s_i := |C_i|$ for $i = 1, \ldots, k$. We will always assume without loss of generality that $s_1 \geq \ldots \geq s_k$. When $s_1 = \ldots = s_k$ (as will be the case in Chapter 3), we will denote the size of all clusters by s.
- $\hat{G} = ([n], \hat{E})$ – a random graph obtained from an *unknown* planted partition distribution $\mathcal{G}(n, \mathcal{C}, p, q)$. This is what the cluster identification algorithm receives as input.
- $c = c(p, q)$ – a constant which controls the minimum cluster size. We will assume that $s_i \geq c\sqrt{n}$ for all i. Its exact value will be specified later.
- $\epsilon = \epsilon(p, q)$ – a small tolerance depending on $p - q$. Its exact value will be specified later.
- $\hat{A} = (\hat{a}_{uv})_{u,v=1}^{n} \in \{0, 1\}^{n \times n}$ – the adjacency matrix of $\hat{G}$.
- $\mathrm{E}[\hat{A}] := (\mathrm{E}[\hat{a}_{uv}])_{u,v=1}^{n}$ – the entrywise expectation of $\hat{A}$.
- $A = (a_{uv})_{u,v=1}^{n} := \mathrm{E}[\hat{A}] + pI_n$ – the expectation of the adjacency matrix $\hat{G}$ with ps added to the diagonal (to make it a rank k). We will often refer to this as the "expectation matrix" of $\hat{G}$ rather than $\mathrm{E}[\hat{G}]$.

- $\hat{B} = (\hat{b}_{uv})_{u,v=1}^{n} := \hat{A} + pI_n - qJ_n$. We will refer to this as the *normalized adjacency matrix* of $\hat{G}$. This will be used in Chapter 4.
- $B = (b_{uv})_{u,v=1}^{n} := \mathrm{E}[\hat{B}] = A - qJ_n$. This will be used in Chapter 4.

See Appendix A for general notation.

1.3 Main results

The main contribution of this thesis is a polynomial time algorithm for recovering planted partitions in which all parts are size $\geq \Omega(\sqrt{n})$. In Chapter 3, we present an algorithm for recovering planted *equipartitions*, i.e. planted partitions in which all clusters are the same size. This yields the following result (1, Theorem 1):

Theorem 1. *There exists a deterministic, polytime algorithm which, for sufficiently large n, with probability $1 - o(1)$ correctly recovers planted partitions in which all clusters are size $s \geq c\sqrt{n}$, where $c = O(1/(p-q)^2)$.*

Our algorithm has the following basic structure:

1. Construct the *dominant rank-k projector* $P_k(\hat{A})$ of the input graph's adjacency matrix $\hat{A}$ (see Definition 14), where k is the number of clusters.
2. Let $\mathbf{v}$ be a certain (carefully chosen) column of $P_k(\hat{A})$, and let W be the indices of the "large" entries of $\mathbf{v}$. With high probability, W will consist mostly of vertices from a single cluster C_i.
3. C_i can be recovered exactly with high probability by taking the vertices v with "many" neighbors in W, i.e., with large $|W \cap N(v)|$.

Why does it make sense to use $P_k(\hat{A})$? If there are k clusters, then $\hat{A}$ is a random perturbation of a rank-k matrix (see Section 3.2). Intuitively, while $\hat{A}$ itself will usually have full rank, the projector $P_k(\hat{A})$ reveals its "true" underlying rank-k structure. This allows us to recover one of the clusters, delete it, and recurse on the remaining vertices.

Chapter 4, we show that, with minor modifications, the same algorithm can be used to recover planted partitions in a much more general setting: namely, the setting in which the clusters are partitioned into "superclusters," where clusters in the same supercluster are approximately the same size, while clusters in different superclusters have sizes separated by $\geq \Omega(\sqrt{n})$ (and, as in the equitable case, all clusters are size $\geq \Omega(\sqrt{n})$). Our main result is the following:

Theorem 2. *There exists a deterministic, polytime algorithm which, for sufficiently large n, with probability $1 - o(1)$ correctly recovers planted partitions $\mathcal{C}$ satisfying the assumptions specified in Section 4.1.*

These theorems are stated more precisely as Theorems 16 and 17, respectively. Finally, in Chapter 5, we discuss how to extend our results for planted partitions in random graphs to planted partitions in random symmetric matrices.

1.4 Previous algorithms

As discussed in the introduction, our algorithm is far from the first to provably recover planted partitions a.s. However, it compares favorably to previously known algorithms (see Chapter 7). In this section we present some of the best-known algorithms.

1.4.1 Trivial common neighbors algorithm

If the clusters are large enough, one can recover them by simply looking at the number of common neighbors each pair of vertices has: pairs of vertices in the same cluster will a.s. have more neighbors in common that pairs in different clusters, so this can be used to distinguish between the two cases. For simplicity, we assume that all clusters are size s, but similar algorithms exist for more general settings.

Algorithm 1 Common neighbors

Given a graph $\hat{G} = (\hat{V}, \hat{E})$, cluster size s:

1. For each pair of distinct vertices u, v, compute $N_{u,v} := |N(u) \cap N(v)|$.
2. For each pair of distinct vertices u, v, identify u and v as being in the same cluster iff. $N_{u,v} \geq (s-2)p^2 + (n-s)q^2 - \frac{3}{2}\sqrt{n \ln n}$.

This algorithm was first introduced by Dyer and Frieze in 1989 for the case $k = 2$ (20) and was extended to general k by Chen and Xu in 2014 (21, Section 2.4).

Theorem 3. *Algorithm 1 a.s. recovers planted partitions in which all parts are size* $s \geq \Omega\left(\frac{\sqrt{n \log n}}{(p-q)^2}\right)$.

Proof. Let $N_{u,v}$ denote the number of common neighbors of $u, v \in \hat{V}$, i.e. $N_{u,v} := |N(u) \cap N(v)|$. For $u \neq v$, this is the sum of $n-2$ independent Bernoulli random variables. Fix $t > 0$. Observe that if u and v are in the same cluster, then

$$\mathrm{E}[N_{u,v}] = (s-2)p^2 + (n-s)q^2,$$

and by Hoeffding's inequality (Theorem 5)

$$\Pr[N_{u,v} \leq \mathrm{E}[N_{u,v}] - t] \leq e^{-2t^2/(n-2)}.$$

Similarly, if u and v are in different clusters, then

$$\mathrm{E}[N_{u,v}] = 2(s-1)pq + (n-2s)q^2,$$

and

$$\Pr[N_{u,v} \geq \mathrm{E}[N_{u,v}] + t] \leq e^{-2t^2/(n-2)}.$$

If we take a union bound (Theorem 4) over all pairs u, v, with $u \neq v$, the probability that *any* of the events hold is $\binom{n}{2} e^{-2t^2/(n-2)}$. We want to make this probility $o(1)$. One can easily verify that setting $t = \frac{3}{2}\sqrt{n \ln n}$ accomplishes this.

Thus, a.s. the following are true:

- $N_{u,v} \geq (s-2)p^2 + (n-s)q^2 - \frac{3}{2}\sqrt{n \ln n}$ for all pairs u, v in the same cluster.

- $N_{u,v} \leq 2(s-1)pq + (n-2s)q^2 + \frac{3}{2}\sqrt{n \ln n}$ for all pairs u, v in different clusters.

Thus, we are able to distinguish between the two cases as long as

$$(s-2)p^2 + (n-s)q^2 - \frac{3}{2}\sqrt{n \ln n} > 2(s-1)pq + (n-2s)q^2 + \frac{3}{2}\sqrt{n \ln n}.$$

That is, we can identify all pairs with at least $(s-2)p^2+(n-s)q^2-\frac{3}{2}\sqrt{n \ln n}$ common neighbors as being in the same cluster, and all remaining pairs as being in different clusters (and there will a.s. be no conflicts). Requiring $s \geq \frac{5\sqrt{n \ln n}}{(p-q)^2}$ is sufficient to satisfy the above inequality. □

This simplistic algorithm requires the cluster sizes to be $\Omega(\sqrt{n \log n})$. More sophisticated algorithms (7; 8; 9; 11; 12; 1; 2) are able to recover planted partitions in which (at least some of) the clusters are size $\Theta(\sqrt{n})$ using spectral techniques or convex programming. Interestingly, the algorithms we present in Chapters 3 and 4 of this thesis use a similar Hoeffding argument as one of the ingredients in their proofs (Lemmas 8 and 17).

1.4.2 Spectral algorithms

Spectral algorithms for planted partition exploit properties of the *spectrum* (i.e. eigenvalues and eigenvectors) of the input graph's adjacency matrix in order to recover the unknown partition of the vertices. This technique was pioneered in the 1990s by Alon, Krivelevich, and Sudakov (14), who used the eigenvector corresponding to the second largest eigenvalue of the adjacency matrix to solve the simpler problem of *planted clique*. (In this problem, one wishes to recover an unknown high-density subset instead of an unknown high-density partition; see (15; 16).) Their algorithm works roughly as follows:

1. Construct the adjacency matrix $\hat{A}$ of the input graph $\hat{G}$, and let $\mathbf{v}$ be an eigenvector corresponding to its second largest eigenvalue.

2. Let W be the indices of the "large" entries of $\mathbf{v}$ (in absolute value). With high probability, W will consist mostly of vertices from the planted clique.

3. The planted clique can be recovered exactly with high probability by taking the vertices v with "many" neighbors in W, i.e., with large $|W \cap N(v)|$. This follows from a Hoeffding argument similar to the one in Section 1.4.1.

This elegant algorithm correctly recovers planted cliques of size $\geq \Omega(\sqrt{n})$. This demonstrates the power of spectral algorithms, as previously known degree based algorithms required the size of the clique to be $\geq \Omega(\sqrt{n \log n})$ (16).

For planted partitions with k clusters, rather than using a single eigenvector of $\hat{A}$, many spectral algorithms, including those we will present in Chapters 3 and 4, use the *dominant rank-k projector* $P_k(\hat{A})$ (Definition 14). This matrix encodes the eigenvalues corresponding to the k largest eigenvalues of $\hat{A}$ and thus captures $\hat{A}$'s "true" rank-k structure.

The most famous spectral algorithm for planted partition is due to McSherry (11, Section 1.2). Like our algorithms, McSherry's is based on projection matrices. The following is an outline of the algorithm, omitting the details:

1. Randomly partition the columns of $\hat{A}$ into two submatrices $\hat{A}_1$ and $\hat{A}_2$. This is done to enforce some regularity among the columns.

2. Compute the *combinatorial projections* $\hat{P}_1$ and $\hat{P}_2$ of $\hat{A}_1$ and $\hat{A}_2$. See (11, Section 3.2) for details. Let $\hat{\mathbf{p}}_v$ be the column of $\hat{P}_1$ or $\hat{P}_2$ corresponding to vertex v.

3. For each vertex v, mark all vertices u for which $||\hat{\mathbf{p}}_u - \hat{\mathbf{p}}_v||_2 \leq \tau$ as being in the same cluster as v, where $\tau > 0$ is some small tolerance.

If p and q are constant, this algorithm a.s. recovers the planted partition as long as

$$(p-q)^2(s_i + s_j) \geq \Omega\left(k\left(\frac{n}{s_k} + \log n\right)\right)$$

for any $i \neq j$ (11, Theorem 4). In particular, when all cluster sizes are within a constant factor of each other, i.e. $s_i = \Theta(n/k)$ for all i, this implies that the cluster sizes must all be $\geq \Omega(n^{2/3})$. Thus, our algorithm beats McSherry's in this regime, as it can handle the case when all clusters are size $\Theta(\sqrt{n})$.

On the other hand, MchSherry's algorithm has the advantage that it is faster. The most expensive operation in both algorithms is computing the appropriate projection operators, as this is done via SVD. Whereas our algorithm computes a projector in each of its k iterations, McSherry's only computes two projectors ($\hat{P}_1$ and $\hat{P}_2$) and is thus asymptotically faster than ours by a factor of $k \leq \sqrt{n}$. See Section 3.1.1 for analysis of our algorithm's running time.

Our algorithm, which we introduced in Seciton 1.3 and will present in detail in Chapters 3 and 4, can be thought of as a "spiritual successor" to the original spectral algorithm of Alon et al. (14), as its overall structure is very similar: it uses the "large" entries $P_k(\hat{A})$ to construct a set W consisting mostly of vertices from a single cluster, recovers the cluster exactly by taking

the vertices with the most neighbors in W, then deletes the recovered cluster and recurses on the remaining vertices.

See Chapter 7 for comparison of our algorithm to McSherry's and other well-known spectral algorithms.

1.4.3 Optimization-based algorithms

Suppose we get a random graph $\hat{G}$ from an unknown planted partition distribution $\mathcal{G}(n, \mathcal{C}, p, q)$. The best we could possibly do to recover the unknown partition $\mathcal{C}$ would be to choose the partition $\hat{\mathcal{C}}$ which maximizes the likelihood of producing $\hat{G}$ as a random sample. In other words,

$$\hat{\mathcal{C}} = \arg\max_{\mathcal{P}} \Pr_{\hat{F} \sim \mathcal{G}(n,\mathcal{P},p,q)} [\hat{F} = \hat{G}], \tag{1.1}$$

where the maximum is taken over all k-partitions $\mathcal{P}$ of $[n]$. This technique is known as *maximum likelihood estimation*, and is a common technique in statistics; see e.g. (22, Chapter 6). Observe that by identifying each partition $\mathcal{P}$ with its incidence matrix $H(\mathcal{P})$ (Definition 4), (Equation 1.1) is equivalent to determining

$$\hat{H} = \arg\max_{H} \Pr_{\hat{F} \sim \mathcal{G}(n,H,p,q)} [\hat{F} = \hat{G}], \tag{1.2}$$

where the maximum is taken over all k-partition matrices H (Definition 5). Note that we are abusing notation by writing H in place of its associated partition $\mathcal{P}$ in $\mathcal{G}(n, H, p, q)$.

Unfortunately, solving the above optimization problem is impractical: it would take exponential time to try all possible partition matrices H, and no polynomial-time method is known if

P $\neq$ NP. However, one way around this is to instead solve a *convex relaxation* of (Equation 1.2) and prove that, under certain conditions, the result is actually equal to the true maximizer. That is, instead of optimizing over all k-partition matrices H, we optimize over an appropriately chosen convex set *containing* all k-partition matrices. In general general convex optimization is hard, but in certain special cases it can be done efficiently; see (23) for an introduction to this topic.

This is the approach used by Chen et al. in (8; 21). Rather than optimizing over all k-partition matrices $H \in \{0,1\}^{n\times n}$, they optimize over the set of all $H = (h_{uv})_{u,v=1}^{n}$ such that

1. $||H||_* \leq n$, where $||\cdot||_*$ denotes the *trace norm* of a matrix, i.e. the sum of its singular values (24, Theorem 5.6.42),
2. $\sum_{u,v} h_{uv} = \sum_{i=1}^{k} s_i^2$,
3. $0 \leq h_{uv} \leq 1$ for all u, v.

It is easy to verify that this is indeed a convex set. Essentially, the first constraint ensures that the solution will have low rank (i.e., correct number of clusters), while the second ensures it will have the correct number of 1s. The authors prove that if $s_i \geq \Omega(\sqrt{n})$, then w.h.p. the solution to this convex optimization problem will, in fact, be the incidence matrix of the planted partition!

This is similar to the approach used by Oymak and Hassibi (9). They observe that the adjacency matrix $\hat{A}$ of $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, p, q)$ can be decomposed as the sum of a sparse matrix and a low-rank matrix. This leads to the convex optimization problem

$$\begin{aligned} \min_{L,S \in \mathbb{R}^{n \times n}} \quad & ||L||_* + \lambda ||S||_1 \\ \text{s.t.} \quad & L \in [0,1]^{n \times n} \\ & L + S = \hat{A}, \end{aligned}$$

where λ is an appropriately chosen constant. They show that w.h.p. the low-rank component L of the optimal solution will be the incidence matrix of the planted partition, as long as all custer sizes are $\geq \Omega(\sqrt{n})$. As in (8; 21), minimizing $||L||_*$ forces L to have low rank, and now minimizing $||S||_1$ forces S to be sparse.

CHAPTER 2

BACKGROUND

In this chapter we go over the mathematical background necessary to understand our results. We assume familiarity with basic graph theory, linear algebra, and probability theory. See, e.g., (25; 24; 22) for background. We review the most important concepts in the sections below. For notation, see Appendix A.

2.1 Set partitions

We begin by introducing the notion of a *set partition*:

Definition 2 (Partition)**.** *A* partition *of a finite set S is collection $\mathcal{P}$ of disjoint, nonempty subsets of S whose union is S. The subsets in this collection are known as the* parts *of the partition. If $|\mathcal{P}| = k$, then $\mathcal{P}$ is called a k-*partition *of S.*

An *equipartition* is a set partition in which all parts are approximately the same size.

Definition 3 (Equipartition)**.** *An* equitable partition *or* equipartition *of a finite set S is a partition $\mathcal{P}$ of S in which the sizes of any two parts differ by at most 1. If $|\mathcal{P}| = k$, then $\mathcal{P}$ is called a k-*equipartition *of S.*

We can identify a partition of a set S with its *incidence matrix* as follows:

Definition 4 (Incidence matrix)**.** *Let $\mathcal{P}$ be a partition of a set S of size n. The* incidence matrix *of $\mathcal{P}$, denoted $H(\mathcal{P})$, is the $n \times n$ 0-1 matrix with rows and columns indexed by S whose (u, v)th entry is 1 iff. u and v are in the same part of $\mathcal{P}$.*

Definition 5 (Partition matrix)**.** *An $n \times n$ matrix is a* partition matrix *iff. it is the incidence matrix of a partition of $[n]$. It is called a k-*partition matrix *iff. it is the incidence matrix of a k-partition matrix of $[n]$.*

Observe that each $n \times n$ partition matrix is the incidence matrix of a *unique* partition of $[n]$; thus there is a 1-1 correspondence between partitions and partition matrices over an index set S. In addition:

Observation 1. *The following are equivalent:*

1. *H is an $n \times n$ k-partition matrix.*
2. *H is the adjacency matrix of a graph G consisting of k disjoint cliques with a loop on each vertex.*
3. *H is permutation similar to $\mathrm{diag}(J_{n_1}, \ldots, J_{n_k})$ for some integers $n_1 \geq \ldots \geq n_k \geq 1$ such that $n_1 + \ldots + n_k = n$; i.e., H can be obtained by permuting the rows and columns of a block diagonal matrix with k ones matrices as the diagonal blocks.*

2.2 Probability theory

As this thesis concerns random graphs and matrices, we will need to make use of some basic results from probability theory. See (22) for introduction and (26) for a more formal treatment of the topic. In addition, (27) is a wonderful introduction to *the probabilistic method*—the use of probability theory in other areas of mathematics, particularly combinatorics.

We will occasionally use a "union bound" to upper bound the probability that any of a collection of events occur with the sum of their individual probabilities:

Theorem 4 (Union bound). *Let $A_1, \ldots, A_n$ be events in a probability space. Then*

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] \leq \sum_{i=1}^{n} \Pr[A_i],$$

and

$$\Pr\left[\bigcap_{i=1}^{n} A_i\right] \geq 1 - \sum_{i=1}^{n} \bar{A}_i.$$

The second inequality follows by applying the first inequality to the complements of $A_1, \ldots, A_n$.

The following well-known result shows that the sum of independent random variables is tightly concentrated about its mean (28):

Theorem 5 (Hoeffding's inequality). *Let $X_1, \ldots, X_n$ be independent random variables with support contained in $[0, 1]$. Define $S_n := X_1 + \ldots + X_n$. Then for any $t > 0$,*

$$\Pr[S_n \geq \mathrm{E}[S_n] + t] \leq e^{-2t^2/n},$$

$$\Pr[S_n \leq \mathrm{E}[S_n] - t] \leq e^{-2t^2/n},$$

and

$$\Pr[|S_n - \mathrm{E}[S_n]| \geq t] \leq 2e^{-2t^2/n}.$$

Note that the last inequality follows from the first two inequalities by taking a union bound.

We will also frequently use the term "almost surely" to describe events that occur with high probability.

Definition 6 (Almost surely/with high probability)**.** *Let* $(\Omega_n, \mathcal{A}_n, P_n)$ *be a probability space and* $A_n \in \mathcal{A}_n$ *an event in the probability space, for* $n = 1, 2, \ldots$. *Then* A_n *occurs* almost surely (a.s.) *or* with high probability (w.h.p.) *if* $\lim_{n\to\infty} \Pr[A_n] = 1$.

In our case, $(\Omega_n, \mathcal{A}_n, P_n)$ will be a probability space of random graphs on n vertices or random $n \times n$ matrices, and we will be concerned with events whose probabilities tend to 1 as the number of vertices or rows increases. Observe that A_n occurs a.s. iff. $\Pr[\bar{A}_n] \to 0$ as $n \to \infty$, where $\bar{A}_n$ denotes the complement of event A_n.

2.3 Graph theory

We assume the reader is familiar with basic graph theory. See (25) for a survey of the topic. For our purposes, we will assume all graphs are simple, undirected graphs with no loops.

2.3.1 Adjacency matrix

The field of *spectral graph theory* studies various matrices associated with graphs and the properties of their eigenvalues and eigenvectors. These matrices are introduced in (25) and covered in depth in (29). The most common of these matrices, and the one on which our planted partition algorithm and its analysis are based, is the *adjacency matrix* (25, Section 1.1):

Definition 7 (Adjacency matrix)**.** *Let* $G = (V, E)$ *be a simple graph on* n *vertices. The* adjacency matrix *of* G *is a symmetric* $n \times n$ *matrix* $A(G) = (a_{uv})_{uv \in V}$ *with rows and columns indexed by* V*, where*

$$a_{uv} := \begin{cases} 1 & \text{if } uv \in E \\ 0 & \text{else} \end{cases},$$

for $u, v \in V$.

2.3.2 Random graphs

The planted partition problem is largely studied because of its applications in network science, but at its core it is a problem about *random graphs*. A random graph is simply a graph drawn from some probability distribution over a set of graphs. The most common such model is the *Erdős-Rényi random graph* first introduced in (19):

Definition 8 (Erdős-Rényi random graph)**.** *The* Erdős-Rényi random graph distribution *is the probability distribution $\mathcal{G}(n,p)$ of graphs with vertex set $[n]$, with edges uv (for $u \neq v$) included independently with probability p. We will denote a graph drawn from this distribution as $G(n,p)$.*

We usually treat $p = p(n)$ as a function of n. If $p = \Omega(1)$ we call $G(n,p)$ as *dense* random graph, and if $p = o(1)$ we call $G(n,p)$ *sparse*. Erdős-Rényi random graphs are known to have a number of beautiful and surprising properties a.s. For example, if p is fixed then the size of the largest clique in $G(n,p)$ is $\Theta(\log n)$ a.s., and if $p = \Omega(\log n/n)$ then $G(n,p)$ connected a.s. See (25, Chapter 13) for an introduction to the Erdős-Rényi model and (30) for a formal, in-depth analysis of this an other models.

The planted partition model (or stochastic block model) introduced in Section 1.1 is a generalization of this model. Instead of including all edges with probability p, we include edges between pairs of vertices in the same cluster with probability p and edges between pairs of vertices in different clusters with probability q. See Definition 1.

2.4 Linear algebra

As the adjacency matrix of a simple graph is a symmetric, 0-1 matrix, we will largely be concerned with real symmetric matrices. Here we review some results that we will use in our analysis. See, e.g., (24; 31) for reference.

When we refer to the *spectrum* of a matrix, we are referring to its eigenvalues and corresponding eigenvectors. The following result characterizes the spectrum of a symmetric matrix:

Theorem 6 (Spectral theorem)**.** *Let $A \in \mathbb{R}^{n\times n}$. Then A is symmetric iff. all of the following are true:*

1. *A has real eigenvalues $\lambda_1(A) \geq \ldots \geq \lambda_n(A)$.*
2. *$\mathbb{R}^n$ has an orthonormal basis $\mathbf{u}_1, \ldots, \mathbf{u}_n$ of eigenvalues A, where $A\mathbf{u}_i = \lambda_i(A)\mathbf{u}_i$ for $i = 1, \ldots, n$.*
3. *A can be diagonalized as $A = U\Lambda(A)U^\top$, where $\Lambda(A) := \operatorname{diag}(\lambda_1(A), \ldots, \lambda_n(A))$ and $U = (\mathbf{u}_1 \ldots \mathbf{u}_n)$.*

See (24, Theorem 4.1.5). Note that the above diagonalization is the same as the singular value decomposition (SVD) of A, and that analogous results hold for complex Hermitian matrices.

We will also occasionally encounter *normal* matrices.

Definition 9 (Normal matrix)**.** *A matrix $A \in \mathbb{C}^{n\times n}$ is* normal *if $A^*A = AA^*$.*

We will make frequent use of the ℓ_2- and *Frobenius* matrix norms. We review their definitions and some basic results:

Definition 10 (ℓ_p-norm)**.** *Let* $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$ *and* $p > 0$*. Let* $\mathbf{v} = (v_1, \ldots, v_n)^\top \in \mathbb{F}^n$ *and* $A \in \mathbb{F}^{n \times n}$*. The* ℓ_p*-norms of* $\mathbf{v}$ *and* A *are defined as*

$$||\mathbf{v}||_p := \left(\sum_{i=1}^{n} |v_i|^p \right)^{1/p},$$

and

$$||A||_p := \sup_{\mathbf{x} \in \mathbb{F}^n \setminus \mathbf{0}} \frac{||A\mathbf{x}||_p}{||\mathbf{x}||_p}.$$

Theorem 7. *Let* $A \in \mathbb{C}^{n \times n}$ *be normal. Then* $||A||_2 = \max_{i=1}^{n} |\lambda_i(A)|$.

For this reason, the ℓ_2-norm of a matrix is referred to as its *spectral norm*. For general matrices, the ℓ_2-norm is equal to the largest singular value. Note that in particular the above theorem holds for real symmetric matrices.

Definition 11 (Frobenius norm)**.** *Let* $A = (a_{ij})_{i,j=1}^{n} \in \mathbb{C}^{n \times n}$*. The* Frobenius norm *of* A *is defined as*

$$||A||_F := \sqrt{\sum_{i,j=1}^{n} |a_{ij}|^2}.$$

This is simply the ℓ_2-norm of A if we treat A as a vector in $\mathbb{C}^{n^2}$. The next theorem follows immediately from the fact that $||A||_F = \sqrt{\mathrm{tr}(A^* A)}$:

Theorem 8. *Let* $A \in \mathbb{C}^{n \times n}$ *be Hermitian. Then*

$$||A||_F^2 = \sum_{i=1}^{n} \lambda_i(A)^2 \leq \mathrm{rk}(A) \cdot ||A||_2^2.$$

Again, note that in particular this theorem holds for real symmetric matrices.

Finally, the following theorem shows that if a matrix is perturbed by adding a matrix with small norm, then the eigenvalues of the perturbed matrix aren't too far off from those of the original matrix:

Theorem 9 (Weyl's inequalities)**.** *Let X, $\Delta \in \mathbb{C}^{n\times n}$ be Hermitian matrices and $Y = X + \Delta$. Then*

$$\lambda_i(X) + \lambda_n(\Delta) \leq \lambda_i(Y) \leq \lambda_i(X) + \lambda_1(\Delta)$$

for $i = 1, \ldots, n$.

See, e.g., (31, Theorem 4.4.6) for proof. Observe that by Theorem 7 this implies that

$$|\lambda_i(X) - \lambda_i(Y)| \leq ||\Delta||_2$$

for $i = 1, \ldots, n$.

2.4.1 Dominant eigenspaces and projectors

The notion of dominant eigenspaces and projectors will be extremely important to our planted partition algorithm and its analysis.

Definition 12 (Dominant eigenspace)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix such that $\lambda_k(A) > \lambda_{k+1}(A)$ for some $k < n$. The* dominant k-dimensional eigenspace *of A is the subspace of $\mathbb{R}^n$ spanned by eigenvectors corresponding to $\lambda_1(A), \ldots, \lambda_k(A)$.*

Note that as long as $\lambda_k(A) > \lambda_{k+1}(A)$ this subspace is unique and has dimension k.

Definition 13 (Orthogonal projection operator)**.** *Let* $\mathbf{U}$ *be a subset of* $\mathbb{R}^n$ *with orthonormal basis* $\mathbf{u}_1, \ldots, \mathbf{u}_r$*. The* orthogonal projection operator *or* orthogonal projector *onto* $\mathbf{U}$ *is*

$$P_{\mathbf{U}} := \sum_{i=1}^{k} \mathbf{u}_i \mathbf{u}_i^\top.$$

This matrix, when applied to a vector $\mathbf{v}$, gives the *projection* of $\mathbf{v}$ onto $\mathbf{U}$, i.e., the "closest" vector in $\mathbf{U}$ to $\mathbf{v}$. More precisely,

$$P_{\mathbf{U}}\mathbf{v} = \arg\min_{\mathbf{u} \in \mathbf{U}} ||\mathbf{u} - \mathbf{v}||_2.$$

The most important step in our planted partition algorithm is constructing the *dominant rank-k projector* of the input graph's adjacency matrix. This is defined as follows:

Definition 14 (Dominant rank-k projector)**.** *Let* $A \in \mathbb{R}^{n \times n}$ *be a symmetric matrix* $\lambda_k(A) > \lambda_{k+1}(A)$ *for some* $k < n$*. The* dominant rank-k projector *of* A*, denoted* $P_k(A)$*, is defined as* $P_{\mathbf{U}}$*, where* $\mathbf{U}$ *is the subspace of* $\mathbb{R}^n$ *spanned by eigenvectors of* A *corresponding to its* k *largest eigenvalues.*

2.4.2 The Cauchy Integral Formula for projectors

Copyright note: This section is taken from my previous work in (1, Section 6.1) and (2, Section 7). See Appendix B for copyright information.

Recall that an analytic function $f : \mathbb{C} \to \mathbb{C}$ can be extended to a function of matrices via its Taylor series (31):

$$f(Z) := f(a)I_n + \frac{f'(a)}{1!}(Z - aI_n) + \frac{f''(a)}{2!}(Z - aI_n)^2 + \dots.$$

In particular, if Z is diagonalizable as $Z = PDP^{-1}$ (as is any symmetric matrix), then $f(Z) = Pf(D)P^{-1}$, where $f(D)$ is evaluated by simply applying f to each diagonal entry.

Accordingly, we also get an extension of the Cauchy integral formula to matrices (31, Theorem 3.4.2):

Theorem 10 (Cauchy integral formula for matrices)**.** *Let Ω be an open set in $\mathbb{C}$. Assume that Γ is a finite set of disjoint simple, closed curves such that Γ is the boundary of an open set D, and $\Gamma \cup D \subset \Omega$. Assume that $Z \in \mathbb{C}^{n\times n}$ and $\lambda_i(Z) \in D$ for $i = 1, \dots, n$. Then for any $f : \mathbb{C} \to \mathbb{C}$ analytic on Ω*

$$f(Z) = \frac{1}{2\pi i}\int_\Gamma (zI_n - Z)^{-1} f(z)dz.$$

We get the following as a corollary (31, Problem 3.4.10):

Theorem 11 (Cauchy integral formula for projectors)**.** *Let $X \in \mathbb{R}^{n\times n}$ be a real symmetric matrix. Let $\gamma \subset \mathbb{C}$ be a simple, closed curve which is the boundary of an open set D such that $\lambda_1(X), \dots, \lambda_r(X) \in D$ and $\lambda_{r+1}(X), \dots, \lambda_n(X) \notin D \cup \gamma$. Then*

$$P_r(X) = \frac{1}{2\pi i}\int_\gamma (zI_n - X)^{-1}dz.$$

Proof. Let γ' be a simple, closed curve which is the boundary of an open set D' such that $\lambda_{r+1}(X), \ldots, \lambda_n(X) \in D'$ and $D \cup \gamma$ and $D' \cup \gamma'$ are separated. Let $\Gamma = \{\gamma, \gamma'\}$. Let Ω_1, Ω_2 be disjoint open sets containing $D \cup \gamma$ and $D' \cup \gamma'$, respectively (this is possible since the latter are separated), and let $\Omega = \Omega_1 \cup \Omega_2$.

Now define f to be a function analytic on Ω such that $f \equiv 1$ on Ω_1 and $f \equiv 0$ on Ω_2. Note that by the identity theorem (32, Corollary 4.9) f cannot be analytic on any connected, open set that intersects both Ω_1 and Ω_2. Observe that $P_r(X) = f(X)$; thus, by Theorem 10 we have

$$P_r(X) = f(X) = \frac{1}{2\pi i}\left(\int_\gamma (zI_n - X)^{-1} f(z) + \int_{\gamma'} (zI_n - X)^{-1} f(z)\right).$$

The theorem follows, as $f(z) = 1$ for $z \in \gamma$ and $f(z) = 0$ for $z \in \gamma'$. □

We can apply this theorem to bound the difference in norm of two projection operators if their largest eigenvalues are both "well-separated" from each other's smallest eigenvalues:

Theorem 12. *Let $X, Y \in \mathbb{R}^{n\times n}$ be symmetric. Suppose that the largest r eigenvalues of both X and Y are $\geq \beta$, and the remaining $n-r$ eigenvalues of both X and Y are $\leq \alpha$, where $\alpha < \beta$. Then*

$$||P_r(X) - P_r(Y)||_2 \leq \frac{||X - Y||_2}{\beta - \alpha} \tag{2.1}$$

and

$$||P_r(X) - P_r(Y)||_F \leq \frac{\sqrt{2r}||X - Y||_2}{\beta - \alpha}. \tag{2.2}$$

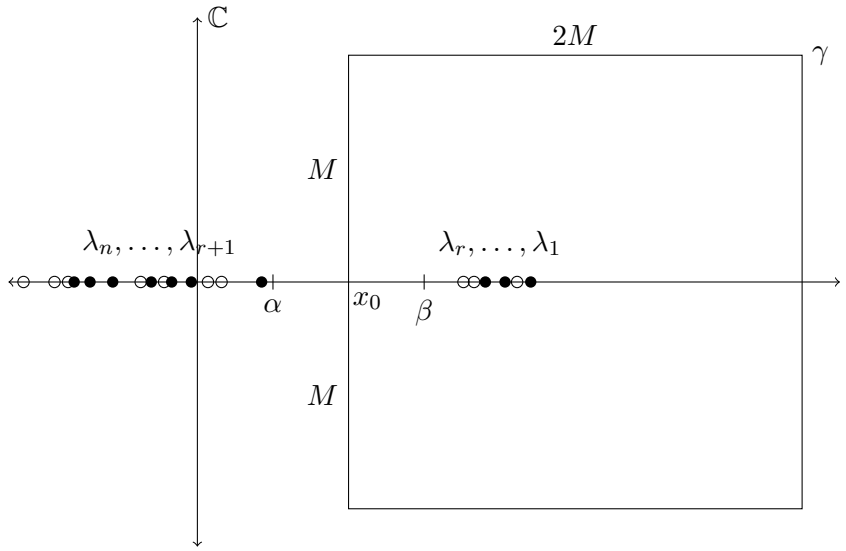

Figure 2. The largest r eigenvalues of both X ($\circ$) and Y ($\bullet$) are in the interior of γ, while the remaining $n - r$ eigenvalues are in the exterior.

Proof. We will prove (Equation 2.1) using Theorem 11. Define γ to be the boundary of a $2M \times 2M$ square in the complex plane with upper and lower sides are on the lines $y = \pm M$, left side on the line $x = x_0$, and right side on the line $x = x_0 + 2M$, where $x_0 := (\alpha + \beta)/2$ and we will let $M \to \infty$. Thus, the interior of γ contains exactly the largest r eigenvalues of both X and Y (see Figure 2).

Applying the Cauchy integral formula,

$$P_r(X) = \frac{1}{2\pi i} \int_\gamma (zI_n - X)^{-1} dz,$$
$$P_r(Y) = \frac{1}{2\pi i} \int_\gamma (zI_n - Y)^{-1} dz.$$

Hence

$$\begin{aligned} P_r(X) - P_r(Y) &= \frac{1}{2\pi i}\int_\gamma (zI_n - X)^{-1}\big((zI_n - Y) - (zI_n - X)\big)(zI_n - Y)^{-1}dz \\ &= \frac{1}{2\pi i}\int_\gamma (zI_n - X)^{-1}\big(X - Y\big)(zI_n - Y)^{-1}dz, \end{aligned}$$

and so we get

$$\begin{aligned} \|P_r(X) - P_r(Y)\|_2 &\leq \frac{1}{2\pi}\int_\gamma \|(zI_n - X)^{-1}\big(X - Y\big)(zI_n - Y)^{-1}\|_2|dz| \\ &\leq \frac{1}{2\pi}\int_\gamma \|(zI_n - X)^{-1}\|_2\|X - Y\|_2\|(zI_n - Y)^{-1}\|_2|dz|. \end{aligned} \tag{2.3}$$

Observe that for each $z \in \mathbb{C}$ the matrices $zI_n - X$ and $zI_n - Y$ are normal. Hence

$$\|(zI_n - X)^{-1}\|_2 = \frac{1}{\min_{j\in[n]}|z - \lambda_j(X)|}, \quad \|(zI_n - Y)^{-1}\|_2 = \frac{1}{\min_{j\in[n]}|z - \lambda_j(Y)|}.$$

Let us first estimate the contribution to the integral (Equation 2.3) on the left side of γ. Let $z = x_0 + yi, y \in \mathbb{R}$. That is, z lies on the line $x = x_0$. By our assumption about the eigenvalues of X and Y, all eigenvalues of X and Y are a horizontal distance of at least $d := (\beta - \alpha)/2$ from the line $x = x_0$; hence

$$|z - \lambda_j(X)|, |z - \lambda_j(Y)| \geq \sqrt{d^2 + y^2}$$

for $z = x_0 + yi$. Therefore, the contribution to (Equation 2.3) from the left side of γ is upper bounded by

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} \frac{||X-Y||_2}{d^2+y^2}dy = \frac{||X-Y||_2}{2d} = \frac{||X-Y||_2}{\beta-\alpha}.$$

The contributions from the other sides of γ go to 0 as $M \to \infty$. This completes the proof of (Equation 2.1).

To show (Equation 2.2), observe that $P_r(X)$ and $P_r(Y)$ both have rank r, so $P_r(X) - P_r(Y)$ has rank at most $2r$. Hence, $P_r(X) - P_r(Y)$ has at most $2r$ nonzero eigenvalues. Recall that for any real symmetric $n \times n$ matrix H

$$||H||_F^2 = \sum_{i=1}^{n} \lambda_i(H)^2 \leq \mathrm{rk}(H) \cdot ||H||_2^2.$$

The lemma thus follows from (Equation 2.1):

$$||P_r(X) - P_r(Y)||_F^2 \leq 2r||P_r(X) - P_r(Y)||_2^2 \leq \frac{2r||X-Y||_2^2}{(\beta-\alpha)^2}. \qquad \square$$

2.5 Eigenvalues of random symmetric matrices

Copyright note: This section is taken from my work in (1, Section 4). See Appendix B for copyright information.

By associating a graph with its adjacency matrix, we may consider a random graph from a planted partition distribution to be a random symmetric, 0-1 matrix in which the diagonal entries are 0 and the above-diagonal entries are independent Bernoulli random variables. We

advertise that our planted partition algorithm is "spectral," and its input is a random graph, so we had better be able to say something about the eigenvalues of a random symmetric matrix. We will show that the eigenvalues of such a random matrix X are close to those of its expectation matrix $\mathrm{E}[X]$. To do so, we will need the following well-known result of Füredi and Komlós about the concentration of eigenvalues of random symmetric matrices (33, Theorem 2):

Theorem 13. *Let $X = [x_{ij}] \in \mathbb{R}^{n \times n}$ be a random symmetric matrix where x_{ij} are independent random variables for $1 \leq i \leq j \leq n$. Assume that there exists $\kappa, \sigma > 0$ so that the following conditions hold independent of n:*

1. $\mathrm{E}[x_{ij}] = 0$ *for* $1 \leq i \leq j \leq n$.
2. $|x_{ij}| \leq \kappa$ *for* $1 \leq i \leq j \leq n$.
3. $\mathrm{E}[x_{ij}^2] \leq \sigma^2$ *for* $1 \leq i \leq j \leq n$.

Then

$$\max_{i=1}^{n} |\lambda_i(X)| \leq 2\sigma\sqrt{n} + 50\kappa n^{\frac{1}{3}} \log n \tag{2.4}$$

with probability $\geq 1 - n^{-10}$ for $n \geq n_0$.

Note that the original paper by Füredi and Komlós assumes that $\mathrm{E}[x_{ij}^2] = \sigma^2$ for all i, j, which in turn makes the bound (Equation 2.4) tight. However, if all we need is the upper bound in (Equation 2.4), as is the case in this thesis, then the proof in (33) goes through with $\mathrm{E}[x_{ij}^2] \leq \sigma^2$. (Actually, it was pointed out by Vu (34) that the proof in (33) contains a minor mistake, so we follow the corrected proof in (34).)

Unfortunately, the n^{-10} failure probability isn't small enough for our purposes, as we will need to apply Theorem 13 simultaneously to $2^{O(\sqrt{n})}$ submatrices of an $n \times n$ random matrix (see Section 3.4.3); however, we may combine it with the following concentration result to get exponentially small failure probability (35, Theorem 1):

Theorem 14. *Let $X = [x_{ij}] \in \mathbb{R}^{n\times n}$ be a random symmetric matrix where x_{ij} are independent random variables such that $|x_{ij}| \leq 1$ for $1 \leq i \leq j \leq n$. Then for every $1 \leq j \leq n$ the probability that $\lambda_j(X)$ deviates from its median by more than t is at most $4e^{-t^2/32j^2}$.*

Combining Theorems 13 and 14, we get the following:

Theorem 15. *Let X be defined as in Theorem 13. Then*

$$\max_{i=1}^{n} |\lambda_i(X)| \leq 2(\sigma + 3\kappa)\sqrt{n}$$

with probability $\geq 1 - e^{-n}$ for $n \geq n_0$.

Proof. By Theorem 13,

$$\Pr\left[\max_{i=1}^{n} |\lambda_i(X)| \geq 2\sigma\sqrt{n} + 50\kappa n^{\frac{1}{3}} \log n\right] < \frac{1}{n^{10}}. \tag{2.5}$$

For $n \geq n_0$, we have

$$50n^{\frac{1}{3}} \log n \leq 0.2\sqrt{n} \Rightarrow 2\sigma\sqrt{n} + 50\kappa n^{\frac{1}{3}} \log n \leq 2(\sigma + 0.1\kappa)\sqrt{n}.$$

Let λ be the median of the random variable $\lambda_1(X)$. We claim that

$$|\lambda| \leq 2(\sigma + 0.1\kappa)\sqrt{n}. \tag{2.6}$$

Indeed,

$$\Pr[\lambda_1(X) \geq 2(\sigma + 0.1\kappa)\sqrt{n}] \leq \frac{1}{n^{10}} \leq \frac{1}{2}$$

by (Equation 2.5). Now consider the random matrix $-X$. It satisfies the assumptions of Theorem 13. Therefore we have

$$\Pr[\lambda_n(-X) \geq 2(\sigma + 0.1\kappa)\sqrt{n}] \leq \frac{1}{n^{10}} \leq \frac{1}{2}.$$

As $\lambda_n(-X) = -\lambda_1(X)$, this is the same as $\Pr[\lambda_1(X) \leq -2(\sigma+0.1\kappa)\sqrt{n}]$. Hence (Equation 2.6) follows by definition of median.

We are now ready to apply Theorem 14. Let $Y = \frac{1}{\kappa}X$. So now each entry of Y is in $[-1, 1]$. Clearly the median of the random variable $\lambda_1(Y)$ is λ/κ. By (Equation 2.6) and Theorem 14

$$\Pr[\lambda_1(X) \geq 2(\sigma + 3\kappa)\sqrt{n}] \leq \Pr\left[\left|\lambda_1(Y) - \frac{\lambda}{\kappa}\right| \geq 5.8\sqrt{n}\right] \leq 4e^{\frac{-(5.8)^2 n}{32}} \leq \frac{1}{2}e^{-n}.$$

Similarly, we may apply the entire argument above to $-X$ to get

$$\Pr[\lambda_1(-X) \geq 2(\sigma + 3\kappa)\sqrt{n}] \leq \frac{1}{2}e^{-n}.$$

Noting that $\max_i |\lambda_i(X)|$ is either $\lambda_1(X)$ or $-\lambda_n(X) = \lambda_1(-X)$, we get

$$\Pr\left[\max_{i=1}^{n} |\lambda_i(X)| \geq 2(\sigma + 3\kappa)\sqrt{n}\right] \leq e^{-n},$$

as claimed. □

CHAPTER 3

ITERATED SPECTRAL RECOVERY OF PLANTED EQUIPARTITIONS

In this chapter we give an efficient spectral algorithm to recover planted equipartitions, i.e. partitions in which all parts are the same size. For the entirety of this chapter, we will assume that each cluster C_i has size $s := n/k$.

3.1 The cluster identification algorithm

Our main result is that Algorithm 2 below recovers clusters of size $c\sqrt{n}$:

Theorem 16. *For sufficiently large n with probability $\geq 1 - 2^{-\Omega(\sqrt{n})}$, Algorithm 2 correctly recovers planted partitions in which all clusters are size $s \geq c\sqrt{n}$, where $c := \max\left\{\frac{88}{p-q}, \frac{72}{(p-q)^2}\right\}$.*

We refer the reader back to Section 1.2 for a review of notation.

Algorithm 2 Equitable iterated projection

Given a graph $\hat{G} = (\hat{V}, \hat{E})$ and cluster size s:

1. Let $\hat{A}$ be the adjacency matrix of $\hat{G}$, $n := |\hat{V}|$, $k := n/s$.
2. Let $P_k(\hat{A}) =: (\hat{p}_{ij})_{i,j \in \hat{V}}$ be the orthogonal projection operator onto the subspace of $\mathbb{R}^n$ spanned by eigenvectors corresponding to the largest k eigenvalues of $\hat{A}$.
3. For each column j of $P_k(\hat{A})$, let $\hat{p}_{i_1 j} \geq \ldots \geq \hat{p}_{i_{n-1} j}$ be the entries other than $\hat{p}_{jj}$ in nonincreasing order. Let $W_j := \{j, i_1, \ldots, i_{s-1}\}$, i.e., the indices of the $s-1$ greatest entries of column j of $P_k(\hat{A})$, along with j itself.
4. Let j^* be the column j that maximizes $||P_k(\hat{A})\mathbf{1}_{W_j}||_2$, i.e. $j^* := \arg\max_{j \in \hat{V}} ||P_k(\hat{A})\mathbf{1}_{W_j}||_2$. It will be shown that W_{j^*} has large intersection with a single cluster $C_i \in \mathcal{C}$ a.s.
5. Let C be the set of s vertices in $\hat{G}$ with the *most* neighbors in W_{j^*}. It will be shown that $C = C_i$ a.s.
6. Remove C and repeat on $\hat{G}[\hat{V} \setminus C]$. Stop when there are $< s$ vertices left.

The overview of Algorithm 2 is as follows. The algorithm gets a random graph $\hat{G}$ generated according to $\mathcal{G}(n, \mathcal{C}, p, q)$. We first construct the projection operator which projects onto the subspace of $\mathbb{R}^n$ spanned by the eigenvectors corresponding to the largest k eigenvalues of $\hat{G}$'s adjacency matrix. This, we will argue, gives a fairly good approximation of at least one of

the clusters, which we can then find and "fix up." Then we remove the cluster and repeat the algorithm.

Note that we ensure that Algorithm 2 works in every iteration a.s. by "preprocessing the randomness"; more precisely, we will show that a.s. certain events occur simultaneously on *all* (exponentially many) subgraphs of $\hat{G}$ induced on a subset of the clusters, and that as long as they all hold Algorithm 2 will *definitely* succeed. See Section 3.4.

3.1.1 Running time

Let us analyze the running time of one iteration of Algorithm 2. Steps 2 and 4 are the most costly.

- In step 2, computing $P_k(\hat{A})$ can be done via classical subspace iteration methods in time $O(n^2k)$ (36; 37). Alternatively, one may utilize one of several recent randomized algorithms (37; 38; 39; 40) which allow this to be done faster, e.g. in time $O(n^2 \log k)$ (38).
- Step 4 can be done naïvely in $O(n^3)$ time. However, this can be improved to $O(n^2k)$ by instead multiplying $P_k(\hat{A})\hat{H}$ and taking the norm of each column, where $\hat{H}$ is defined as in Section 3.4.1. From step 2 we get an orthonormal decomposition of $P_k(\hat{A})$, i.e. an $n \times k$ orthogonal matrix U such that $UU^\top = P_k(\hat{A})$. Thus, we can compute $P_k(\hat{A})\hat{H} = UU^\top\hat{H}$ in $O(n^2k)$ time by first multiplying a $k \times n$ matrix and an $n \times n$ matrix, then an $n \times k$ matrix and a $k \times n$ matrix.

  In theory, this step can be sped up further using a fast matrix multiplication algorithm (41; 42), but such algorithms are rarely used in practice due to numerical instability and large constants hidden in their asymptotic running times.

Thus, each iteration of Algorithm 2 can be done in $O(n^2k)$ time. Since there are k iterations, the overall running time is $O(n^2k^2)$. In particular, as $k \leq \sqrt{n}$, this is $O(n^3)$.

3.2 Spectral properties of the adjacency matrix

The goal of this section is to prove a separation of the first k eigenvalues of both A and $\hat{A}$ from the remaining $n-k$. We begin by examining the eigenvalues of A.

Without loss of generality, we may assume $C_1 = \{1, \ldots, s\}, C_2 = \{s+1, \ldots, 2s\}, \ldots, C_k = \{n-s+1, \ldots, n\}$. Then the expectation matrix A looks like:

$$A = \left(\begin{array}{ccc|ccc|c|ccc} p & \cdots & p & q & \cdots & q & & q & \cdots & q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ p & \cdots & p & q & \cdots & q & & q & \cdots & q \\ \hline q & \cdots & q & p & \cdots & p & & q & \cdots & q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ q & \cdots & q & p & \cdots & p & & q & \cdots & q \\ \hline & \vdots & & & \vdots & & \ddots & & \vdots & \\ \hline q & \cdots & q & q & \cdots & q & & p & \cdots & p \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ q & \cdots & q & q & \cdots & q & & p & \cdots & p \end{array}\right) = qJ_n + (p-q)\,\mathrm{diag}(J_s, \ldots, J_s),$$

where J_m is the $m \times m$ ones matrix. Thus, A contains all the information about the unknown partition $\mathcal{C}$.

The following lemma is easily verified:

Lemma 1. *The eigenvalues of A are*

$$\lambda_1(A) = (p-q)s + qn,$$

$$\lambda_i(A) = (p-q)s \text{ for } i = 2, \ldots, k,$$

$$\lambda_i(A) = 0 \text{ for } i = k+1, \ldots, n.$$

So we see that the smallest positive eigenvalue is proportional to the size of the clusters.

We continue by bounding the spectral norm of $\hat{A} - A$ (recall that the spectral norm of a symmetric matrix $X \in \mathbb{R}^{n \times n}$ is $||X||_2 = \max_{i=1}^{n} |\lambda_i(X)|$; see (31, Corollary 4.11.13)).

Lemma 2. *For sufficiently large n,*

$$\|\hat{A} - A\|_2 \leq 8\sqrt{n} \tag{3.1}$$

with probability $\geq 1 - e^{-n}$.

Proof. Set $X = (x_{ij}) = \hat{A} - \mathrm{E}[\hat{A}]$. Let σ_{ij} be the standard deviation of x_{ij} and let $\sigma \geq \sigma_{ij}$ for $i, j \in [n]$. Hence, X satisfies the conditions of Theorem 15, with

$$\kappa = 1, \quad \sigma = \max(\sqrt{p(1-p)}, \sqrt{q(1-q)}) \leq \frac{1}{2}.$$

Thus,

$$||X||_2 = \max_{i=1}^{n} |\lambda_i(X)| \leq 2(\sigma + 3\kappa)\sqrt{n} \leq 7\sqrt{n} \tag{3.2}$$

with probability $\geq 1 - e^{-n}$ by Theorem 15.

Observe that

$$\hat{A} - A = \hat{A} - \mathrm{E}[\hat{A}] - pI_n = X - pI_n$$

$$\Rightarrow$$

$$\|\hat{A} - A\|_2 = \|X - pI_n\|_2 = \|X\|_2 + \|pI_n\|_2 \leq \|X\|_2 + p.$$

From (Equation 3.2) we deduce that

$$||\hat{A} - A||_2 \leq 7\sqrt{n} + p \leq 8\sqrt{n}$$

with probability $> 1 - e^{-n}$ for $n > n_0$. □

We can now use the lemma above to characterize the eigenvalues of $\hat{A}$ (and A) as follows:

Lemma 3. *Assume $\hat{A}$ satisfies* (Equation 3.1) *and $s \geq c\sqrt{n}$. Then the largest k eigenvalues of A and $\hat{A}$ are in the interval $[c'\sqrt{n}, n]$ and all other eigenvalues of A and $\hat{A}$ are in the interval $[-8\sqrt{n}, 8\sqrt{n}]$, where*

$$c' := (p - q)c - 8. \tag{3.3}$$

Proof. Applying Weyl's inequalities (Theorem 9) to Lemma 2 yields

$$|\lambda_i(\hat{A}) - \lambda_i(A)| \leq \max(|\lambda_n(A - \hat{A})|, |\lambda_1(A - \hat{A})|) = ||A - \hat{A}||_2 \leq 8\sqrt{n}$$

for $i = 1, \ldots, n$. Thus, by Lemma 1 we get

$$\begin{aligned} \lambda_i(\hat{A}) &\geq (p-q)s - 8\sqrt{n} \\ &\geq ((p-q)c - 8)\sqrt{n} \text{ for } i = 1, \ldots, k, \\ |\lambda_i(\hat{A})| &\leq 8\sqrt{n} \text{ for } i = k+1, \ldots, n. \end{aligned}$$

The lemma thus follows by definition of c'.

Note that the upper bound of n follows from the fact that for any $X = (x_{ij}) \in \mathbb{R}^{n \times n}$ we have $\lambda_1(X) \leq \max_i \sum_j |x_{ij}|$. □

Lemma 3 shows that a.s. we have a separation between the largest k eigenvalues and the remaining eigenvalues of both A and $\hat{A}$, provided that $c' > 8$, or equivalently

$$c > \frac{16}{p-q}. \tag{3.4}$$

We will assume this is the case from now on.

Note that the argument above shows that, in fact, $\lambda_1(\hat{A}) \geq qn + (p-q)s - 8\sqrt{n} = \Theta(n)$, while $\lambda_2(\hat{A}), \ldots, \lambda_k(\hat{A}) \leq (p-q)s + 8\sqrt{n} = O(s)$, but this information will not be needed hence. Figure 3 illustrates the distribution of eigenvalues of A and $\hat{A}$.

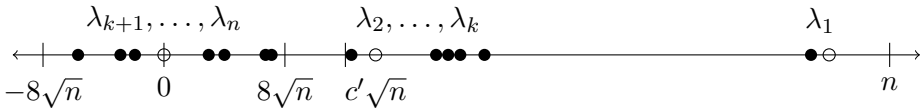

Figure 3. The distribution of eigenvalues of A ($\circ$) and $\hat{A}$ ($\bullet$).

3.3 Deviation between the projectors

In this section, we will prove bounds on $\|P_k(\hat{A}) - P_k(A)\|_2$ and $\|P_k(\hat{A}) - P_k(A)\|_F$, where $\|\cdot\|_2$ and $\|\cdot\|_F$ are the spectral and the Frobenius matrix norms, respectively. The following lemma characterizes $P_k(A)$:

Lemma 4.

$$P_k(A) = \frac{1}{s}\sum_{i=1}^{k} \mathbf{1}_{C_i}\mathbf{1}_{C_i}^\top = \frac{1}{s}H, \tag{3.5}$$

where $H := H(\mathcal{C})$ is the incidence matrix of $\mathcal{C}$.

Proof. Let $\mathbf{u}_i := \frac{1}{\sqrt{s}}\mathbf{1}_{C_i} \in \mathbb{R}^n$ for $i = 1, \ldots, k$, and let $\mathbf{U}$ be the subspace of $\mathbb{R}^n$ spanned by eigenvectors corresponding to $\lambda_1(A), \ldots, \lambda_k(A)$. It is easily verified that $\mathbf{u}_1, \ldots, \mathbf{u}_k$ are an orthonormal basis for $\mathbf{U}$. Thus, letting $P_{\mathbf{U}}$ denote the orthogonal projection operator onto $\mathbf{U}$, we get

$$P_k(A) = P_{\mathbf{U}} = \sum_{i=1}^{k} \mathbf{u}_i\mathbf{u}_i^\top = \frac{1}{s}\sum_{i=1}^{k} \mathbf{1}_{C_i}\mathbf{1}_{C_i}^\top. \qquad \square$$

If we assume $C_1 = \{1, \ldots, s\}, C_2 = \{s+1, \ldots, 2s\}, \ldots, C_k = \{n-s+1, \ldots, n\}$ as in Section 3.2, then $P_k(A)$ looks like:

$$P_k(A) = \frac{1}{s} \left(\begin{array}{ccc|ccc|c|ccc} 1 & \ldots & 1 & 0 & \ldots & 0 & & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ldots & \vdots & \ddots & \vdots \\ 1 & \ldots & 1 & 0 & \ldots & 0 & & 0 & \ldots & 0 \\ \hline 0 & \ldots & 0 & 1 & \ldots & 1 & & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ldots & \vdots & \ddots & \vdots \\ 0 & \ldots & 0 & 1 & \ldots & 1 & & 0 & \ldots & 0 \\ \hline & \vdots & & & \vdots & & \ddots & & \vdots & \\ \hline 0 & \ldots & 0 & 0 & \ldots & 0 & & 1 & \ldots & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ldots & \vdots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 & \ldots & 0 & & 1 & \ldots & 1 \end{array} \right) = \frac{1}{s} \operatorname{diag}(J_s, \ldots, J_s)$$

when represented in the standard basis for $\mathbb{R}^n$.

So we see that the columns of $P_k(A)$ are essentially the indicator vectors of the unknown clusters $C_1, \ldots, C_k$. The central idea behind Algorithm 2 is that if $||P_k(A) - P_k(\hat{A})||_F$ is sufficiently small, then some column of $P_k(\hat{A})$ is a good approximation to the corresponding column of $P_k(A)$ and can thus be used to recover the corresponding cluster.

As $P_k(\hat{A})$ and $P_k(A)$ are projection operators, we have

$$|P_k(\hat{A})\|_2 = \|P_k(A)\|_2 = 1 \quad \Rightarrow \quad \|P_k(\hat{A}) - P_k(A)\|_2 \leq 2.$$

In fact, we can make this difference arbitrarily small by increasing the cluster size appropriately, as shown in the following lemma.

Lemma 5. *Assume $\hat{A}$ satisfies* (Equation 3.1) *and $s \geq c\sqrt{n}$. Then*

$$\|P_k(\hat{A}) - P_k(A)\|_2 \leq \epsilon \tag{3.6}$$

and

$$\|P_k(\hat{A}) - P_k(A)\|_F \leq \sqrt{2k}\epsilon \tag{3.7}$$

provided that

$$\frac{8}{(p-q)c - 16} \leq \epsilon, \tag{3.8}$$

Proof. By (Equation 3.1) and Lemma 3, we can apply Theorem 12 with

$$X = A, \quad Y = \hat{A}, \quad \alpha = (p-q)c\sqrt{n} - 8\sqrt{n}, \quad \beta = 8\sqrt{n}$$

to get

$$\begin{aligned} ||P_k(A) - P_k(\hat{A})||_2 &\leq \frac{8\sqrt{n}}{(p-q)c\sqrt{n} - 16\sqrt{n}} \\ &= \frac{8}{(p-q)c - 16} \\ &\leq \epsilon, \end{aligned}$$

We get (Equation 3.7) similarly by applying (Equation 2.2). □

3.4 Proof of algorithm's corectness

The proof of Algorithm 2's correctness goes roughly as follows. We will prove using the spectral analysis in Sections 3.2-3.3 that a.s. there is a column j for which $||P_k(\hat{A})\mathbf{1}_{W_j}||_2$ is "large" (Lemma 6). Next, we will show that any for any such j, the set W_j consists mostly of vertices from a single cluster (Lemma 7). Finally, we show how to recover this cluster exactly by looking at how many neighbors each vertex has in W_j (Lemmas 8-9).

This argument shows that Algorithm 2 succeeds in iteration 1 a.s. To show that it succeeds in *every* iteration, we will apply the same argument to all "cluster subgraphs" of $\hat{G}$—i.e., those subgraphs induced on a subset of the clusters. We will prove that all 2^k such subgraphs have certain desirable properties a.s., in which case our algorithm *deterministically* succeeds in identifying a cluster. Therefore, when we remove it we are considering another cluster subgraph, so the algorithm again succeeds, and so on. Thus, we are able to restrict our analysis to these cluster subgraphs, bounding the number of events that need to occur in order to ensure the algorithm's success. This is how we avoid the need to randomly split the graph into parts, as in (13; 11; 12). The details of this approach, which we call "preprocessing the randomness," are presented in Section 3.4.3

3.4.1 Notation

We will use the following notation in our proof:

- $H = (h_{ij})_{i,j=1}^n := H(\mathcal{C})$ – the incidence matrix of $\mathcal{C}$ (Definition 4), i.e., $h_{ij} = 1$ if i and j are in the same cluster, 0 else.
- $W_1, \ldots, W_n$ as defined as in step 3 of Algorithm 2.

- $\hat{H} = (\hat{h}_{ij})_{i,j=1}^{n} := (\mathbf{1}_{W_1}, \ldots, \mathbf{1}_{W_n})$ – the "estimated incidence matrix." The idea is that at least one column of $\hat{H}$ will be a good approximation of the corresponding column of H, and we will give a way to find such a column. Note that each column of $\hat{H}$ has exactly s 1s and that $\hat{H}$ need not be symmetric.

3.4.2 Technical lemmas

The proof of Theorem 16 relies on several additional lemmas. Lemmas 6-9 fit together roughly as follows:

- Lemma 6 says that a.s. there is a column j for which $||P_k(\hat{A})\mathbf{1}_{W_j}||_2$ is large.
- Lemma 7 says that for such a column j, W_j consists mostly of vertices from a single cluster C_i.
- Lemmas 8 and 9 say that a.s. vertices in C_i will have many neighbors in W_j, while vertices outside C_i will have relatively few neighbors in W_j; hence, we can recover C_i by taking the s vertices with the most neighbors in W_j.

Lemma 6. *Assume $\hat{A}$ satisfies* (Equation 3.1)*. Then there exists a column j such that*

$$||P_k(\hat{A})\mathbf{1}_{W_j}||_2 \geq (1 - 8\epsilon^2 - \epsilon)\sqrt{s}. \tag{3.9}$$

Proof. Lemma 5 gives

$$||P_k(A) - P_k(\hat{A})||_F^2 \leq 2k\epsilon^2.$$

By definition of $\hat{H}$,

$$\mathrm{tr}(H^2) = \mathrm{tr}(\hat{H}^\top \hat{H}) = ns$$

and, letting $P_k(\hat{A}) = (\hat{p}_{ij})_{i,j=1}^n$, for each column $j \in [n]$ we have

$$(\hat{H}^\top P_k(\hat{A}))_{jj} = \sum_{i=1}^n \hat{h}_{ij}\hat{p}_{ij} \geq \sum_{i=1}^n h_{ij}\hat{p}_{ij} = (HP_k(\hat{A}))_{jj} \quad \Rightarrow \quad \mathrm{tr}(\hat{H}^\top P_k(\hat{A})) \geq \mathrm{tr}(HP_k(\hat{A})).$$

Recall also (Equation 3.5) that $P_k(A) = \frac{1}{s}H$. Therefore

$$\begin{aligned}
2k\epsilon^2 &\geq ||P_k(\hat{A}) - P_k(A)||_F^2 \\
&= \|\frac{1}{s}H - P_k(\hat{A})\|_F^2 \\
&= \frac{1}{s^2}\,\mathrm{tr}(H^2) + \mathrm{tr}(P_k(\hat{A})^2) - 2\frac{1}{s}\,\mathrm{tr}(HP_k(\hat{A})) \\
&\geq \frac{1}{s^2}\,\mathrm{tr}(\hat{H}^\top \hat{H}) + \mathrm{tr}(P_k(\hat{A})^2) - 2\frac{1}{s}\,\mathrm{tr}(\hat{H}^\top P_k(\hat{A})) \\
&= \|\frac{1}{s}\hat{H} - P_k(\hat{A})\|_F^2.
\end{aligned}$$

The triangle inequality then yields:

$$\|\frac{1}{s}H - \frac{1}{s}\hat{H}\|_F \leq \|\frac{1}{s}H - P_k(\hat{A})\|_F + \|\frac{1}{s}\hat{H} - P_k(\hat{A})\|_F \leq 2\epsilon\sqrt{2k}.$$

Thus

$$||H - \hat{H}||_F^2 = \sum_{j=1}^n \left(\sum_{i=1}^n (h_{ij} - \hat{h}_{ij})^2\right) \leq 8k\epsilon^2 s^2,$$

so by averaging there exists a column j^* such that

$$\sum_{i=1}^{n}(h_{ij^*} - \hat{h}_{ij^*})^2 \leq \frac{1}{n} \cdot 8k\epsilon^2 s^2 = 8\epsilon^2 s. \tag{3.10}$$

Now let C_{i^*} be the cluster containing j^*. Define $W = W_{j^*}$, $U = W \cap C_{i^*}$, $V = W \setminus U$. Thus we have

$$P_k(A)\mathbf{1}_U = \frac{|U|}{s}\mathbf{1}_{C_{i^*}},\ P_k(A)\mathbf{1}_V = \sum_{i \neq i^*} a_i \mathbf{1}_{C_i},\ 0 \leq a_i, \sum_{i \neq i^*} a_i = \frac{s - |U|}{s}.$$

By (Equation 3.10) we have $|U| \geq (1 - 8\epsilon^2)s$, so

$$||P_k(A)\mathbf{1}_W||_2^2 = ||P_k(A)\mathbf{1}_U||_2^2 + ||P_k(A)\mathbf{1}_V||_2^2 \geq ||P_k(A)\mathbf{1}_U||_2^2 = \frac{|U|^2}{s} \geq (1 - 8\epsilon^2)^2 s.$$

Finally, note that by Lemma 5 we have $||P_k(\hat{A}) - P_k(A)||_2 \leq \epsilon$, so the triangle inequality yields the desired result:

$$\begin{aligned} ||P_k(\hat{A})\mathbf{1}_W||_2 &\geq ||P_k(A)\mathbf{1}_W||_2 - ||(P_k(\hat{A}) - P_k(A))\mathbf{1}_W||_2 \\ &\geq ||P_k(A)\mathbf{1}_W||_2 - ||P_k(\hat{A}) - P_k(A)||_2||\mathbf{1}_W||_2 \\ &\geq (1 - 8\epsilon^2)\sqrt{s} - \epsilon\sqrt{s}. \end{aligned}$$

□

Lemma 7. *Assume $\hat{A}$ satisfies* (Equation 3.1) *and j satisfies* (Equation 3.9). *Then $|W_j \cap C_i| \geq (1 - 3\epsilon)s$ for some $i \in [k]$.*

Proof. For $W \subseteq [n]$ define

$$t(W) := \max_{i=1}^{k} |C_i \cap W|, \quad \tau := \min_{W} t(W),$$

where the minimum is taken over all $W \subseteq [n]$ such that $|W| = s$ and

$$||P_k(A)\mathbf{1}_W||_2 \geq (1 - 8\epsilon^2 - 2\epsilon)\sqrt{s}. \tag{3.11}$$

We will argue that

$$t(W_j) \geq \tau \geq (1 - 3\epsilon)s,$$

which proves the lemma.

The first inequality is easy: for $W = W_j$, the triangle inequality yields

$$\begin{aligned} ||P_k(A)\mathbf{1}_W||_2 &\geq ||P_k(\hat{A})\mathbf{1}_W||_2 - ||P_k(A) - P_k(\hat{A})||_2||\mathbf{1}_W||_2 \\ &\geq (1 - 8\epsilon^2 - \epsilon)\sqrt{s} - \epsilon\sqrt{s} \\ &= (1 - 8\epsilon^2 - 2\epsilon)\sqrt{s}. \end{aligned}$$

So $t(W_j) \geq \tau$. We just need to show that $\tau \geq (1 - 3\epsilon)s$.

For $W \subseteq [n], i \in [k]$, let $t_i(W) := |C_i \cap W|$. Construct $W \subseteq [n]$ such that

- $|W| = s$.
- W satisfies (Equation 3.11).

- $t(W) = \tau$.
- $||P_k(A)\mathbf{1}_W||_2$ is as large as possible.

We will argue that W must have a special structure: namely, it is split between two clusters.

By relabeling the clusters, we may assume without loss of generality that

$$\tau = t(W) = t_1(W) \geq \ldots \geq t_k(W).$$

We claim that

1. $t_2(W) < t_1(W)$. Suppose to the contrary. Maximize $\sum_{i=1}^k x_i^2$, such that $x_1 = x_2 \geq x_3 \geq \ldots \geq x_k \geq 0$ and $\sum_{i=1}^k x_i = s$. It is easy show that the maximum occurs when $x_1 = x_2 = \frac{s}{2}$ and $x_3 = \ldots = x_k = 0$. Hence $\sum_{i=1}^k t_i(W)^2 \leq \frac{s^2}{2}$. By (Equation 3.11) we have
$$(1 - 8\epsilon^2 - 2\epsilon)^2 s \leq ||P_k(A)\mathbf{1}_W||_2^2 = \frac{1}{s}\sum_{i=1}^k t_i(W)^2 \leq \frac{s}{2},$$
which is equivalent to $(1 - 8\epsilon^2 - 2\epsilon) \leq \frac{1}{\sqrt{2}}$. Choosing ϵ sufficiently small ($\epsilon \leq .1$ works) we get a contradiction.
2. $t_3(W) = \ldots = t_k(W) = 0$. Assume this is not the case. Then
$$\tau = t(W) = t_1(W) > t_2(W) \geq t_3(W) \geq 1, \quad \sum_{i=1}^k t_i(W) = s.$$

In particular, $C_3 \cap W$ and $C_2 \setminus W$ are both nonempty. Now construct $\tilde{W}$ from W by replacing a vertex from $C_3 \cap W$ with one from $C_2 \setminus W$. Clearly $|\tilde{W}| = s$, and $t(\tilde{W}) = \tau$ since only t_2 increases and $t_2(W) < t_1(W) = \tau$. But

$$\begin{aligned} ||P_k(A)\mathbf{1}_{\tilde{W}}||_2^2 - ||P_k(A)\mathbf{1}_W||_2^2 &= \frac{1}{s}\sum_{i=1}^{k} t_i(\tilde{W})^2 - \frac{1}{s}\sum_{i=1}^{k} t_i(W)^2 \\ &= \frac{1}{s}\left[(t_2(W)+1)^2 + (t_3(W)-1)^2 - t_2(W)^2 - t_3(W)^2\right] \\ &= \frac{2}{s}(t_2(W) - t_3(W) + 1) \\ &> 0, \end{aligned}$$

contradicting the maximality of $||P_k(A)\mathbf{1}_W||_2$.

Thus, W is split between two clusters C_1 and C_2; i.e., $W = U \cup V$, where $U := W \cap C_1$ and $V := W \cap C_2$. So by (Equation 3.11) we have

$$(1 - 8\epsilon^2 - 2\epsilon)^2 s \leq ||P_k(A)\mathbf{1}_W||_2^2 = ||P_k(A)(\mathbf{1}_U + \mathbf{1}_V)||_2^2 = \frac{|U|^2}{s} + \frac{(s-|U|)^2}{s}.$$

Solving the inequality for $|U|$ yields

$$\tau = \max\{|U|, |V|\} \geq (1 - 3\epsilon)s,$$

provided ϵ is small enough (again $\epsilon \leq .1$ is sufficient). This completes the proof. □

Lemma 8. *Consider cluster C_i and vertex $j \in [n]$. If $j \in C_i$, then*

$$|N_{\hat{G}}(j) \cap C_i| \geq (p - \epsilon)s \tag{3.12}$$

with probability $\geq 1 - e^{-\epsilon^2 s}$, and if $j \notin C_i$, then

$$|N_{\hat{G}}(j) \cap C_i| \leq (q + \epsilon)s \tag{3.13}$$

with probability $\geq 1 - e^{-\epsilon^2 s}$.

Proof. Let $j \in C_i$. Then $\mathrm{E}[|N(j) \cap C_i|] = p(s-1)$, so Hoeffding's inequality (Theorem 5) yields

$$\Pr[|N(j) \cap C_i| \leq (p - \epsilon)s] \leq e^{-2(\epsilon s - p)^2/(s-1)} \leq e^{-\epsilon^2 s}$$

for n (hence s) sufficiently large. On the other hand, if $j \notin C_i$. Then $\mathrm{E}[|N(j) \cap C_i|] = qs$, so

$$\Pr[|N(j) \cap C_i| \geq (q + \epsilon)s] \leq e^{-2\epsilon^2 s} \leq e^{-\epsilon^2 s}. \qquad \square$$

Lemma 9. *Let $W \subseteq [n]$ such that $|W| = s$ and $|W \cap C_i| \geq (1 - 3\epsilon)s$ for some $i \in [k]$. Then*

a) If $j \in C_i$ and j satisfies (Equation 3.12)*, then $|N_{\hat{G}}(j) \cap W| \geq (p - 4\epsilon)s$.*

b) If $j \in [n] \setminus C_i$ and j satisfies (Equation 3.13)*, then $|N_{\hat{G}}(j) \cap W| \leq (q + 4\epsilon)s$.*

Proof. Assume $j \in C_i$ and j satisfies (Equation 3.12). As $|C_i| = s$, we have $|C_i \setminus W| \leq 3\epsilon$. Therefore,

$$\begin{aligned} |N(j) \cap W| &\geq |N(j) \cap W \cap C_i| \\ &= |N(j) \cap C_i| - |(N(j) \cap C_i) \setminus W| \\ &\geq |N(j) \cap C_i| - |C_i \setminus W| \\ &\geq (p - \epsilon)s - 3\epsilon s \\ &= (p - 4\epsilon)s. \end{aligned}$$

Part b) follows by a similar argument. □

This lemma gives us a way to differentiate between vertices $j \in C_i$ and vertices $j \notin C_i$ as shown in Figure 4, provided

$$p - 4\epsilon \geq q + 4\epsilon. \tag{3.14}$$

3.4.3 Main proof

To prove Theorem 16, we will define certain (exponentially many) events on the probability space $\mathcal{G}(n, \mathcal{C}, p, q)$ and show that

1. As long as they all occur, Algorithm 2 *definitely* succeeds.
2. They all occur simultaneously a.s.

Therefore, Algorithm 2 succeeds a.s.

Before we define the events let us introduce some notation:

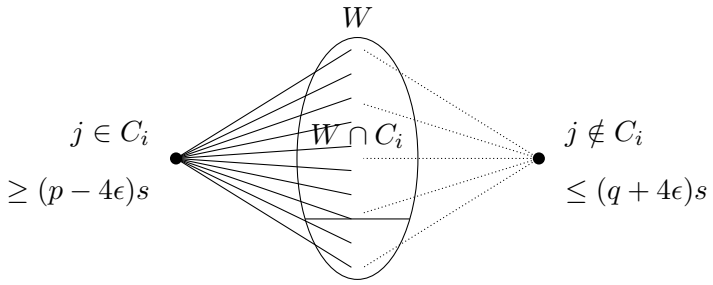

Figure 4. If W has large overlap with C_i, then a.s. vertices in C_i will have many neighbors in W, while vertices not in C_i will have relatively few neighbors in W.

- For $J \subseteq [k]$, define $\hat{G}_J$ to be the subgraph of $\hat{G}$ induced by clusters $C_i, i \in J$, i.e. $\hat{G}_J := \hat{G}\left[\bigcup_{i \in J} C_i\right]$. Then for any fixed J we have

$$\hat{G}_J \sim \mathcal{G}(|J|s, \{C_i : i \in J\}, p, q). \tag{3.15}$$

- For an $n \times n$ matrix B define B_J to be principal submatrix of B with row and column indices in the clusters $C_i, i \in J$, i.e. $B_J := B\left[\bigcup_{i \in J} C_i\right]$.

We will refer to these subgraphs and submatrices as *cluster subgraphs* and *cluster submatrices.*

Now we define two types of events in $\mathcal{G}(n, \mathcal{C}, p, q)$:

- *Spectral events*: for $J \subseteq [k]$, let E_J be the event that $||\hat{A}_J - A_J||_2 \leq 8\sqrt{|J|s}$.
- *Degree events*: for $1 \leq i \leq k, 1 \leq j \leq n$, let $D_{i,j}$ be the event that $|N_{\hat{G}}(j) \cap C_i| \geq (p - \epsilon)s$ if $j \in C_i$, or the event that $|N_{\hat{G}}(j) \cap C_i| \leq (q + \epsilon)s$ if $j \notin C_i$.

Thus, we have defined a total of $2^k + nk$ events. Essentially, these are the events that every $\hat{G}_J$ satisfies (Equation 3.1) and that (Equation 3.12) and (Equation 3.13) are satisfied for all $i \in [k], j \in [n]$. Note that the events are well-defined, as their definitions depend only on the underlying probability space $\mathcal{G}(n, \mathcal{C}, p, q)$ and not on the random graph $\hat{G}$ sampled from the space.

Now we are finally ready to prove the theorem:

Proof of Theorem 16. Assume E_J and $D_{i,j}$ hold for all $J \subseteq [k], i \in [k], j \in [n]$. We will prove by induction that Algorithm 2 succeeds in every iteration.

For the base case, take the original graph $\hat{G} = \hat{G}_{[k]}$ considered in the fist iteration. Since $E_{[k]}$ is assumed to hold, (Equation 3.1) is satisfied. Thus, by Lemma 6, the column $j = j^*$ identified in step 4 satisfies (Equation 3.9). Then by Lemma 7 we have $|W_{j^*} \cap C_i| \geq (1 - 3\epsilon)s$ for some $i \in [k]$. Finally, since $D_{i,j}$ is assumed to hold for all $j \in [n]$, step 5 correctly identifies $C = C_i$ by Lemma 9.

Now assume Algorithm 2 succeeds in the first t iterations, i.e., it correctly identifies a cluster and removes it in each of these iterations. Then the graph considered in the $(t+1)$st iteration is a cluster subgraph $\hat{G}_J$ for some $J \subseteq [k]$, $|J| = k - t$. Note that $\hat{G}_J$ has $|J|s = (k-t)s$ vertices. Now we apply Lemmas 3-7 with $\hat{A}_J$ instead of $\hat{A}$, A_J instead of A, $k - t$ instead of k, and $(k-t)s$ instead of n.

Since E_J is assumed to hold, by Lemma 6 the column $j = j^*$ identified in step 4 of Algorithm 2 satisfies $||P_{k-t}(\hat{A}_J)\mathbf{1}_{W_j}||_2 \geq (1 - 8\epsilon^2 - \epsilon)\sqrt{s}$. Note that $\hat{H}$ and W_j (Sections 3.4.1-3.4.2) are constructed from $\hat{G}_J$, not the original graph $\hat{G}$. Now by Lemma 7 we have $|W_{j^*} \cap C_i| \geq$

$(1-3\epsilon)s$ for some $i \in J$. Finally, since $D_{i,j}$ is assumed to hold for all $j \in [n]$, step 5 once again correctly identifies $C = C_i$ by Lemma 9.

We have thus proved that Algorithm 2 succeeds as long as E_J and $D_{i,j}$ hold for all $J \subseteq [k], i \in [k], j \in [n]$. Now, for any fixed nonempty $J \subseteq [k]$ we have $\hat{G}_J \sim \mathcal{G}(|J|s, \{C_i : i \in J\}, p, q)$, so by Lemma 2

$$\Pr[E_J] \geq 1 - e^{-|J|s} \geq 1 - e^{-s}.$$

By Lemma 8, for any i, j

$$\Pr[D_{i,j}] \geq 1 - e^{-\epsilon^2 s.}$$

Taking a union bound (Theorem 4) over all J, i, j, the probability that all E_J and $D_{i,j}$ hold is $\geq 1 - 2^k e^{-s} - nke^{-\epsilon^2 s}$. Therefore, as ϵ is constant and $k \leq \sqrt{n} \leq s$, Algorithm 2 succeeds with probability $\geq 1 - \left(\frac{2}{e}\right)^{-\sqrt{n}} - n^{3/2}e^{-\sqrt{n}}$.

Note that we require (Equation 3.14) in order for step 5 of Algorithm 2 to correctly recover a cluster according to Lemma 9. In addition, the proof of Lemma 7 requires $\epsilon \leq .1$. By (Equation 3.8), we can satisfy both of these conditions by setting $c := \max\left\{\frac{88}{p-q}, \frac{72}{(p-q)^2}\right\}$. □

CHAPTER 4

EXTENSION TO NON-EQUITABLE PARTITIONS

In this chapter we prove that, with minor modifications, the algorithm for recovering planted equipartitions can be used to recover more general planted partitions (subject to certain constraints).

4.1 The "superclusters" setting

Without much work, one can show that, in fact, Algorithm 2 works when all clusters are *almost* the same size—i.e., when $(1-\epsilon)\frac{n}{k} \leq |C_i| \leq (1+\epsilon)\frac{n}{k}$ for all i, where $\epsilon = O(p-q)$. A natural next step is to try to extend it to the case when the clusters are divided into K "superclusters," where clusters in the same supercluster have roughly the same size, while clusters in different superclusters have sizes separated by $\geq c\sqrt{n}$. This is the setting which we consider for the remainder of this chapter.

More precisely:

- Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the set of clusters.

- Let $s_i := |C_i|$ for $i = 1, \ldots, k$ and assume without loss of generality that

$$s_1 \geq \ldots \geq s_k \geq c\sqrt{n}. \tag{4.1}$$

- Assume $\mathcal{C}$ is partitioned into K "superclusters" $\mathcal{C} = \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_K$.

- Let $k_i := |\mathcal{C}_i|$ be the number of clusters in supercluster $\mathcal{C}_i$, for $i = 1, \ldots, K$.

- Assume that the sizes of clusters in different superclusters are separated by $\geq c\sqrt{n}$. Furthermore, we may assume that the $\mathcal{C}_i$ are arranged in decreasing order of their cluster sizes; i.e.,

$$\min_{C \in \mathcal{C}_i} |C| \geq \max_{C \in \mathcal{C}_{i+1}} |C| + c\sqrt{n} \tag{4.2}$$

  for $i = 1, \ldots, K-1$. Thus, by (Equation 4.1) we have

$$\mathcal{C}_1 = \{C_1, \ldots, C_{k_1}\}, \mathcal{C}_2 = \{C_{k_1+1}, \ldots, C_{k_1+k_2}\}, \ldots, \mathcal{C}_K = \{C_{k-k_K+1}, \ldots, C_k\}. \tag{4.3}$$

- Within the superclusters the sizes are approximately the same:

$$\max_{C \in \mathcal{C}_i} |C| \leq (1+\epsilon) \min_{C \in \mathcal{C}_i} |C| \tag{4.4}$$

  for $i = 1, \ldots, K$, where $\epsilon = \epsilon(p, q)$ will be specified later.

- We may sometimes abuse notation and use $\mathcal{C}_i$ to refer to the set of *indices* j such that $C_j \in \mathcal{C}_i$ or the set of *vertices* $u \in \bigcup_{C \in \mathcal{C}_i} C$.

We refer the reader back to Section 1.2 for a review of notation.

Our goal is still to recover the *individual clusters* exactly; we do not care about identifying which pairs of clusters belong to the same supercluster. Note that we assume that we know p, q, and $k_1, \ldots, k_K$ a priori. We will discuss how to determine these parameters empirically in Section 8.1.

A key difference between the equitable and non-equitable cases is that the algorithm and analysis are based on normalized versions of A and $\hat{A}$:

$$\hat{B} = (\hat{b}_{uv})_{u,v=1}^{n} := \hat{A} + pI_n - qJ_n, \quad B = (b_{uv})_{u,v=1}^{n} := \mathrm{E}[\hat{B}] = A - qJ_n.$$

This simplifies the spectral analysis considerably, since B is essentially a block diagonal matrix (after permuting the rows and columns). We will refer to $\hat{B}$ as the *normalized adjacency matrix* of $\hat{G}$. In order to compute $\hat{B}$, we assume that our algorithm has access to the exact values of p and q, or at least good approximations. We discuss this further in Section 8.1.2.

4.2 The algorithm

We now show how to adapt Algorithm 2 to the "superclusters" setting presented in Section 4.1. The key difference is that we will project onto the eigenspace of $\hat{B}$ corresponding to its largest k_1 (rather than k) eigenvalues. Because we have an $\Omega(\sqrt{n})$ separation between $\mathcal{C}_1$ and $\mathcal{C}_2$, this will allow us to recover one of the clusters in $\mathcal{C}_1$. When we have recovered all clusters in $\mathcal{C}_1$, we will move on to $\mathcal{C}_2$, then $\mathcal{C}_3$, and so on.

Another complication is that, since the clusters are not all the same size, it is no longer reasonable to assume we know the cluster sizes exactly. However, we will see that the eigenvalues of $\hat{B}$ give good approximations to the cluster sizes. More precisely, for $u \in C_i \in \mathcal{C}_j$, $\lambda_u(\hat{B})/(p-q)$ is a good approximation to s_i. In fact, since all clusters in $\mathcal{C}_j$ are approximately the same size, it is a good approximation to the size of *any* cluster in $\mathcal{C}_j$. This allows us to construct an approximate cluster of roughly the correct size, as in Step 4 of Algorithm 2.

Algorithm 3 Non-equitable iterated projection

Given a graph $\hat{G} = (\hat{V}, \hat{E})$, supercluster sizes $k_1, \ldots, k_K$:

1. Let $\hat{A}$ be the adjacency matrix of $\hat{G}$, $n := |\hat{V}|$, $\hat{B} := \hat{A} - qJ_n + pI_n$.

2. Let $P_{k_1}(\hat{B}) =: (\hat{p}_{uv})_{u,v\in\hat{V}}$ be the orthogonal projection operator onto the dominant k_1-dimensional eigenspace of $\hat{B}$.

3. Let $\hat{s} := (\lambda_1(\hat{B}) + 7\sqrt{n})/(p-q)$. We will see that this is approximately the size of the largest cluster.

4. For each column v of $P_{k_1}(\hat{B})$, let $W_v := \{u \in \hat{V} : \hat{p}_{uv} \geq \frac{1}{2\hat{s}}\}$, i.e., the indices of the "large" entries of column v of $P_{k_1}(\hat{B})$.

5. Let v^* be the column v such that $|W_v| \leq (1+\epsilon)s$ and $||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2$ is maximum, i.e. $v^* := \arg\max_{v:|W_v|\leq(1+\epsilon)\hat{s}} ||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2$. It will be shown that such a v^* exists and W_{v^*} has large intersection with a single cluster $C_i \in \mathcal{C}_1$ a.s.

6. Let C be the set of vertices in $\hat{G}$ with $\geq (p-10\epsilon)\hat{s}$ neighbors in W_{v^*}. It will be shown that $C = C_i$ a.s.

7. Remove C and repeat on $\hat{G}[\hat{V} \setminus C]$, with supercluster sizes $k_1 - 1, k_2, \ldots, k_K$. If $k_1 = 1$, instead use $k_2, \ldots, k_K$ as the supercluster sizes (i.e., k_2 becomes the "new" k_1, k_3 the "new" k_2, and so on). Stop when all supercluster sizes are 0.

The main result of this chapter is the following:

Theorem 17. *Let $\mathcal{C}$ be an unknown partition of $[n]$ satisfying the conditions in Section 4.1, with $\epsilon = O(p-q)$ and $c = \Omega\left(\frac{1}{(p-q)\epsilon}\right)$. Then Algorithm 3 recovers $\mathcal{C}$ given only $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, p, q)$ with probability $\geq 1 - 2^{-\Omega(\sqrt{n})}$.*

Sections 4.3-4.6 are devoted to proving the correctness of Algorithm 3, mirroring the analysis in (1). Sections 4.3 and 4.4 develop the linear algebra tools necessary for the proof, Section 4.5 uses these tools to prove that Steps 4-6 of Algorithm 3 successfully recover a single cluster a.s., while Section 4.6 shows that the algorithm as a whole successfully recovers *all* clusters a.s.

4.3 Spectral properties of the normalized adjacency matrix

Observe that by permuting the rows and columns of B we get $B \sim (p-q)\,\mathrm{diag}(J_{s_1}, \ldots, J_{s_k})$. Thus, its eigenvalues are trivial to compute:

Lemma 10. *B is a rank-k matrix with eigenvalues*

$$\lambda_i(B) = (p-q)s_i \text{ for } i = 1, \ldots, k,$$

$$\lambda_i(B) = 0 \text{ for } i = k+1, \ldots, n.$$

As in the equitable case, we a.s. get a deviation of at most $O(\sqrt{n})$ between the eigenvalues of B and $\hat{B}$ by applying a modified version of Füredi and Komlós's well-known result on the distribution of eigenvalues of random symmetric matrices (33):

Lemma 11. *With probability* $\geq 1 - e^{-n}$,

$$||B - \hat{B}||_2 \leq 7\sqrt{n} \tag{4.5}$$

for sufficiently large n.

Proof. Apply Theorem 15 to $X := \hat{B} - B$ with $\kappa = 1, \sigma = 1/2$ to get

$$||B - \hat{B}||_2 = \max_{i=1}^{n} |\lambda_i(X)| \leq 7\sqrt{n}$$

with probability $\geq 1 - e^{-n}$. □

By Weyl's inequalities (Theorem 9), we get

$$|\lambda_i(B) - \lambda_i(\hat{B})| \leq 7\sqrt{n} \tag{4.6}$$

for $i = 1, \ldots, n$; i.e., we can approximate the eigenvalues of B with those of $\hat{B}$ (and vice versa) with at most $O(\sqrt{n})$ error. This yields an $\Omega(\sqrt{n})$ separation in the eigenvalues of both B and $\hat{B}$ between different superclusters; i.e., for $i = 1, \ldots, K-1$,

$$\min_{C_j \in \mathcal{C}_i} \min\{\lambda_j(B), \lambda_j(\hat{B})\} \geq \max_{C_j \in \mathcal{C}_{i+1}} \max\{\lambda_j(B), \lambda_j(\hat{B})\} + \Omega(\sqrt{n}),$$

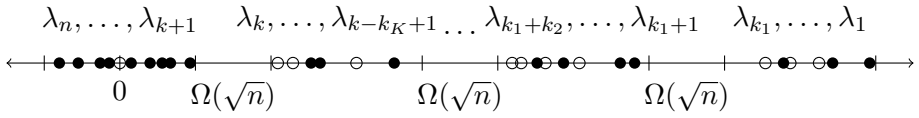

Figure 5. The distribution of eigenvalues of B ($\circ$) and $\hat{B}$ ($\bullet$)

as shown in Figure 5. However, such a separation between $\mathcal{C}_1$ and $\mathcal{C}_2$ will suffice, since Algorithm 3 computes $P_{k_1}(\cdot)$ in *each* iteration on a submatrix of the original $\hat{B}$ from the first iteration.

Lemma 12. *Assume* (Equation 4.5) *holds. Then*

$$(p-q)s_{k_1} - 7\sqrt{n} \leq \lambda_i(B), \lambda_i(\hat{B}) \leq (p-q)s_1 + 7\sqrt{n}$$

for $i = 1, \ldots, k_1$, *while*

$$\lambda_i(B), \lambda_i(\hat{B}) \leq (p-q)s_{k_1+1} + 7\sqrt{n}$$

for $i = k_1 + 1, \ldots, n$.

Proof. First we handle B. For $i \leq k_1$ we have

$$(p-q)s_{k_1} \leq \lambda_i(B) = (p-q)s_i \leq (p-q)s_1.$$

By definition of k_1, we have $\mathcal{C}_1 = \{C_1, \ldots, C_{k_1}\}$; thus, by (Equation 4.2) we have

$$\lambda_i(B) = (p-q)s_i \leq (p-q)s_{k_1+1}$$

for $i = k_1 + 1, \ldots, k$, and for $i > k$ we have

$$\lambda_i(B) = 0 \leq (p-q)s_{k_1+1}.$$

The lemma thus follows by (Equation 4.6) □

Thus, the eigenvalues of B and $\hat{B}$ corresponding to the clusters in $\mathcal{C}_1$ are separated from the remaining eigenvalues by at least $((p-q)c-14)\sqrt{n}$, since $s_{k_1} \geq s_{k_1+1} + c\sqrt{n}$ by (Equation 4.2). This quantity is positive as long as

$$c > \frac{14}{p-q}.$$

4.4 Deviation between the projection operators

We can apply Theorem 12 to B and $\hat{B}$ to get the following:

Lemma 13. *Assume* (Equation 4.5) *holds. Then we have*

$$||P_{k_1}(B) - P_{k_1}(\hat{B})||_2 \leq \epsilon \tag{4.7}$$

and

$$||P_{k_1}(B) - P_{k_1}(\hat{B})||_F \leq \sqrt{2k_1}\epsilon, \tag{4.8}$$

provided that

$$c \geq \frac{14}{(p-q)\epsilon}. \tag{4.9}$$

Proof. By (Equation 4.5) and Lemma 12, we can apply Theorem 12 with

$$X = B, \quad Y = \hat{B}, \quad \alpha = (p-q)s_{k_1} - 7\sqrt{n}, \quad \beta = (p-q)s_{k_1+1} + 7\sqrt{n}$$

to get

$$\begin{aligned} ||P_{k_1}(B) - P_{k_1}(\hat{B})||_2 &\leq \frac{7\sqrt{n}}{(p-q)(s_{k_1} - s_{k_1+1}) - 14\sqrt{n}} \\ &\leq \frac{7\sqrt{n}}{(p-q)c\sqrt{n} - 14\sqrt{n}} \\ &\leq \frac{14}{(p-q)c} \\ &\leq \epsilon, \end{aligned}$$

where the second inequality follows from (Equation 4.2). We get (Equation 4.8) similarly by applying (Equation 2.2). □

Thus, we see that c has an inverse dependence on ϵ and $p - q$. In order for the proofs in Section 4.5.2 to go through, we will require that $\epsilon = O(p - q)$. Thus, by (Equation 4.9) c must be $\Omega((p-q)^{-2})$.

Why is Lemma 13 useful? Observe that

$$P_{k_1}(B) = \sum_{i=1}^{k_1} \frac{1}{s_i} \mathbf{1}_{C_i} \mathbf{1}_{C_i}^\top.$$

Thus, the nonzero entries of $P_{k_1}(B)$ tell us precisely which pairs of vertices belong to the same cluster $C \in \mathcal{C}_1$. Of course our algorithm does not have access to this matrix, but by Lemma 13 $P_{k_1}(\hat{B}) \approx P_{k_1}(B)$ a.s., so we should be able to use $P_{k_1}(\hat{B})$ in place of $P_{k_1}(B)$ to recover the clusters. Sections 4.5 and 4.6 go over the details of this approach.

4.5 Recovering a single cluster

In this section we show how to use the spectral results in Sections 4.3 and 4.4 to recover a single cluster. In Section 4.5.1 we will show how to construct a.s. a set W with large intersection with a single cluster C_i (Steps 4-5 of Algorithm 3), and in Section 4.5.2 we will show how to recover C_i exactly a.s. by looking at the number of neighbors in W of each vertex (Step 6). In Section 4.6 we will show how to recover *all* clusters using this procedure.

4.5.1 Constructing an approximate cluster

We do not assume that our algorithm has access to the exact cluster sizes, so let us begin by showing that $\hat{s}$ as defined in Step 3 of Algorithm 3 is a good approximation to the size of the clusters in $\mathcal{C}_1$ (recall that by (4.1) they are all approximately the same size).

Lemma 14. *Assume* (Equation 4.5) *holds and define*

$$\hat{s} := \frac{\lambda_1(\hat{B}) + 7\sqrt{n}}{p - q}.$$

Then

$$s_{k_1} \leq \ldots \leq s_1 \leq \hat{s} \leq (1 + 2\epsilon)s_{k_1}. \tag{4.10}$$

Proof. By Lemma 10, $\lambda_1(B) = (p-q)s_1$, so by (Equation 4.5) and Weyl's inequalities we have

$$(p-q)s_1 - 7\sqrt{n} \leq \lambda_1(\hat{B}) \leq (p-q)s_1 + 7\sqrt{n}.$$

Thus, $\hat{s}$ is an upper bound on s_1. Finally, as $s_1 \leq (1+\epsilon)s_{k_1}$ by equation (Equation 4.4), we have

$$\begin{aligned}
\hat{s} &\leq \frac{\lambda_1(B) + 14\sqrt{n}}{p-q} \\
&= s_1 + \frac{14\sqrt{n}}{p-q} \\
&= s_{k_1}\left(\frac{s_1}{s_{k_1}} + \frac{14\sqrt{n}}{(p-q)s_{k_1}}\right) \\
&\leq s_{k_1}\left(1 + \epsilon + \frac{14}{(p-q)c}\right) \\
&\leq (1+2\epsilon)s_{k_1}.
\end{aligned}$$

Note that the last inequality follows from (Equation 4.9). □

We will now show how to use $P_{k_1}(\hat{B})$ to construct an "approximate cluster," (a set with small symmetric difference with one of the clusters) as in Steps 4-5 of Algorithm 3. The following lemma gives a way to produce such an approximate cluster using only $\hat{B}$:

Lemma 15. *Assume* (Equation 4.5) *holds. If* $|W| \leq (1+\epsilon)\hat{s}$ *and* $||P_{k_1}(\hat{B})\mathbf{1}_W||_2 \geq (1-3\epsilon)\sqrt{s}_{k_1}$, *then* $|W \cap C_i| \geq (1-6\epsilon)s_{k_1}$ *for some* $C_i \in \mathcal{C}_1$.

Proof. Observe that by (Equation 4.10) we have

$$|W| \leq (1+\epsilon)(1+2\epsilon)s_{k_1} \leq (1+4\epsilon)s_{k_1},$$

provided $\epsilon \leq 1/2$. By the triangle inequality,

$$\begin{aligned} ||P_{k_1}(B)\mathbf{1}_W||_2 &\geq ||P_{k_1}(\hat{B})\mathbf{1}_W||_2 - ||P_{k_1}(B) - P_{k_1}(\hat{B})||_2||\mathbf{1}_W||_2 \\ &\geq (1-3\epsilon)\sqrt{s_{k_1}} - \epsilon\sqrt{(1+4\epsilon)s_{k_1}} \\ &\geq (1-5\epsilon)\sqrt{s_{k_1}}. \end{aligned} \tag{4.11}$$

We will show that in order for this to hold, W must have large intersection with some cluster in $\mathcal{C}_1$.

Fix t such that $\frac{(1+4\epsilon)s_{k_1}}{2} \leq t \leq s_{k_1}$. Assume by way of contradiction that $|W \cap C_i| \leq t$ for all $i \leq k_1$. Observe that

$$||P_{k_1}(B)\mathbf{1}_W||_2^2 = \sum_{i=1}^{k} \frac{1}{s_i}|W \cap C_i|^2. \tag{4.12}$$

Consider the optimization problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{k_1} \frac{1}{s_i} x_i^2 \\ \text{s.t.} \quad & \sum_{i=1}^{k_1} x_i \leq (1+4\epsilon)s_{k_1}, \\ & 0 \leq x_i \leq t \text{ for } i = 1, \ldots, k_1, \end{aligned}$$

with variable x_i representing $|W \cap C_i|$. It is easy to see that the maximum occurs when $x_{k_1} = t, x_{k_1-1} = (1+4\epsilon)s_{k_1} - t$, $x_i = 0$ for all $i < k_1 - 1$, and the maximum is $\frac{t^2}{s_{k_1}} + \frac{((1+4\epsilon)s_{k_1}-t)^2}{s_{k-1}}$. Note that the value of of x_{k_1-1} is legal by our assumption that $t \geq \frac{(1+4\epsilon)s_{k_1}}{2}$. Thus, by (Equation 4.11) and (Equation 4.12) we have

$$(1-5\epsilon)^2 s_{k_1} \leq ||P_{k_1}(B)\mathbf{1}_W||_2^2 \leq \frac{t^2}{s_{k_1}} + \frac{((1+4\epsilon)s_{k_1}-t)^2}{s_{k_1-1}} \leq \frac{t^2}{s_{k_1}} + \frac{((1+4\epsilon)s_{k_1}-t)^2}{s_{k_1}}.$$

Solving for t, this implies

$$t \geq \left(\frac{1+4\epsilon}{2} + \frac{1}{2}\sqrt{1-28\epsilon+34\epsilon^2}\right) s_{k_1}.$$

If we make ϵ small enough ($\epsilon \leq .01$ suffices), then this is $> (1-6\epsilon)s_{k_1}$. Thus, if we pick $t = (1-6\epsilon)s_{k_1}$ we have a contradiction.

Therefore, it must be the case that $|W \cap C_i| > (1-6\epsilon)s_{k_1}$ for some $i \leq k_1$. Note that for the proof to go through we require $\frac{1+4\epsilon}{2} \leq 1-6\epsilon$, which is certainly satisfied if $\epsilon \leq .01$. □

This lemma shows that we can a.s. produce an approximate cluster by trying all sets $W \subseteq V$ with $|W| \leq (1+\epsilon)\hat{s}$ and taking the one which maximizes $||P_{k_1}(\hat{B})\mathbf{1}_W||_2$. However, this would take $\Omega(n^{s_{k_1}})$ time, so we need to narrow the search space. The next lemma shows that we can, in fact, produce such a W by defining

$$W_v := \left\{u : \text{the } (u,v) \text{ entry of } P_{k_1}(\hat{B}) \text{ is} \geq \frac{1}{2\hat{s}}\right\}$$

for each $v \in [n]$ and taking the W_v which maximizes $||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2$. This is exactly what is done in Steps 4-5 of Algorithm 3.

Lemma 16. *Assume* (Equation 4.5) *holds. Then there exists* $v \in [n]$ *such that* $|W_v| \leq (1+\epsilon)\hat{s}$ *and* $||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2 \geq (1-3\epsilon)\sqrt{s_{k_1}}$.

Proof. Let $H = (h_{uv})_{u,v=1}^n := H(\mathcal{C})$ be the incidence matrix of $\mathcal{C}_1$ (Definition 4. This is the matrix that results from rounding the nonzero entries of $P_{k_1}(B)$ to 1. Now we similarly define $\hat{H} = (\hat{h}_{uv})_{u,v=1}^n$ to be the matrix that results from rounding the "large" entries of $P_{k_1}(\hat{B})$ to 1:

$$\hat{h}_{uv} := \begin{cases} 1 & \text{if the } (u,v) \text{ entry of } P_{k_1}(\hat{B}) \text{ is } \geq \dfrac{1}{2\hat{s}} \\ 0 & \text{else} \end{cases} .$$

Observe that column v of $\hat{H}$ is $\mathbf{1}_{W_v}$.

Now consider the errors between H and $\hat{H}$. By definition of $\hat{H}$, each error contributes $\geq \frac{1}{4\hat{s}^2}$ to $||P_{k_1}(B) - P_{k_1}(\hat{B})||_F^2$. Thus, by Lemma 13 we have

$$\frac{1}{4\hat{s}^2} \cdot \begin{pmatrix} \#\text{ errors in} \\ \text{cols. } v \in \mathcal{C}_1 \end{pmatrix} \leq ||P_{k_1}(B) - P_{k_1}(\hat{B})||_F^2 \leq 2k_1\epsilon^2.$$

Let $n_1 := s_1 + \ldots + s_{k_1} =$ the number of vertices (columns) in $\mathcal{C}_1$. Averaging over the columns in $\mathcal{C}_1$, there must exist a vertex $v \in \mathcal{C}_1$ with at most $8k_1\epsilon^2\hat{s}^2/n_1$ errors. Let C_i be the cluster containing v. Then by (Equation 4.10)

$$\begin{aligned}
|W_v \setminus C_i| + |C_i \setminus W_v| &= \text{\# errors in column } v \text{ of } \hat{H} \\
&\leq \frac{8k_1\epsilon^2\hat{s}^2}{n_1} \\
&\leq \frac{8k_1\epsilon^2(1+2\epsilon)^2 s_{k_1}^2}{n_1} \\
&\leq \frac{8n_1\epsilon^2(1+2\epsilon)^2 s_{k_1}}{n_1} \\
&\leq 9\epsilon^2 s_{k_1} \\
&\leq \epsilon s_{k_1}.
\end{aligned}$$

Thus,

$$|W_v \cap C_i| = s_i - |C_i \setminus W_v| \geq s_i - \epsilon s_{k_1} \geq (1-\epsilon)s_i,$$

and

$$|W_v| = |W_v \cap C_i| + |W_v \setminus C_i| \leq s_i + \epsilon s_{k_1} \leq (1+\epsilon)s_i \leq (1+\epsilon)\hat{s}.$$

Finally, we must argue that $||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2 \geq (1-3\epsilon)\sqrt{s_{k_1}}$. First,

$$\begin{aligned}
||P_{k_1}(B)\mathbf{1}_{W_v}||_2^2 &= \sum_{j=1}^{k_1} \frac{|W_v \cap C_j|^2}{s_j} \\
&\geq \frac{|W_v \cap C_i|^2}{s_i} \\
&\geq (1-\epsilon)^2 s_i.
\end{aligned}$$

Then by the triangle inequality and (Equation 4.7)

$$\begin{aligned} ||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2 &\geq ||P_{k_1}(B)\mathbf{1}_{W_v}||_2 - ||P_{k_1}(B) - P_{k_1}(\hat{B})||_2||\mathbf{1}_{W_v}||_2 \\ &\geq (1-\epsilon)\sqrt{s_i} - \epsilon\sqrt{|W_v|} \\ &\geq (1-\epsilon)\sqrt{s_i} - \epsilon\sqrt{(1+\epsilon)s_i} \\ &\geq (1-3\epsilon)\sqrt{s_i}. \end{aligned}$$

This completes the proof. □

Lemmas 15 and 16 fit together as follows: Lemma 15 shows that any set W such that $W \leq (1+\epsilon)\hat{s}$ and $||P_{k_1}(\hat{B})\mathbf{1}_W||_2 \geq (1-3\epsilon)\sqrt{s_{k_1}}$ must come mostly from a single cluster, while Lemma 16 shows that there must be such a W among the W_v. Thus, we can a.s. produce an approximate cluster W by simply taking the W_v such that $|W_v| \leq (1+\epsilon)\hat{s}$ and $||P_{k_1}(\hat{B})\mathbf{1}_{W_v}||_2$ is maximum.

Note that this approach does not require any access to $s_1, \ldots, s_m$. However, we assume $k_1, \ldots, k_K$ are known so that we know how many eigenvectors to project onto (i.e., $k = k_1$).

4.5.2 Recovering the cluster exactly

Once we have a set W with small symmetric difference with a cluster C_i, we show how to recover C_i *exactly* a.s. by looking at the number of neighbors each vertex has in W. First, we show that vertices in C_i are distinguished from those outside C_i by their number of neighbors in C_i itself (Lemma 17). Then we show that using W in place of C_i does not throw things off by too much (Lemma 18).

Lemma 17. *Consider cluster C_i and vertex $u \in [n]$. If $u \in C_i$, then*

$$|N_{\hat{G}}(u) \cap C_i| \geq (p - \epsilon)s_i \tag{4.13}$$

with probability $\geq 1 - e^{-\epsilon^2 s_i}$, and if $u \notin C_i$, then

$$|N_{\hat{G}}(u) \cap C_i| \leq (q + \epsilon)s_i \tag{4.14}$$

with probability $\geq 1 - e^{-\epsilon^2 s_i}$.

The proof is essentially the same as that of Lemma 8 and is therefore left as an exercise.

Lemma 18. *Assume* (Equation 4.5) *holds. Suppose $|W| \leq (1+\epsilon)\hat{s}$ and $|W \cap C_i| \geq (1-6\epsilon)s_{k_1}$ for some $C_i \in \mathcal{C}_1$. Then*

a) If $u \in C_i$ and u satisfies (Equation 4.13), then $|N_{\hat{G}}(u) \cap W| \geq (p - 10\epsilon)\hat{s}$.

b) If $u \in [n] \setminus C_i$ and u satisfies (Equation 4.14), then $|N_{\hat{G}}(u) \cap W| \leq (q + 10\epsilon)\hat{s}$.

Proof. Assume $u \in C_i$ and u satisfies (Equation 4.13). We want to lower bound $|N(u) \cap W|$ in terms of $|N(u) \cap C_i|$. The worst case is when as many as possible of u's neighbors in C_i come from $C_i \setminus W$, i.e., when u is adjacent to all vertices in $C_i \setminus W$. Thus, we have

$$|N(u) \cap W| \geq |N(u) \cap C_i| - |C_i \setminus W| \geq (p - \epsilon)s_i - |C_i \setminus W|.$$

As $|C_i| = s_i$ and $|W \cap C_i| \geq (1-6\epsilon)s_{k_1}$, we have

$$|C_i \setminus W| \leq s_i - (1-6\epsilon)s_{k_1} \leq s_i - \frac{1-6\epsilon}{1+\epsilon}s_i \leq 7\epsilon s_i.$$

Therefore,

$$|N(u) \cap W| \geq (p-\epsilon)s_i - 7\epsilon s_i \geq (p-8\epsilon)s_{k_1}.$$

Therefore, by (Equation 4.10) we have

$$|N(u) \cap W| \geq \frac{p-8\epsilon}{1+2\epsilon}\hat{s} \geq (p-10\epsilon)\hat{s}.$$

This proves part a).

For part b), assume $u \notin C_i$ and u satisfies (Equation 4.14). Now we want to upper bound $|N(u) \cap W|$ in terms of $|N(u) \cap C_i|$. Now the worst case is when u has as many neighbors as possible in $W \setminus C_i$, i.e., when u is adjacent to all vertices in $W \setminus C_i$. In this case,

$$|N(u) \cap W| \leq |N(u) \cap C_i| + |W \setminus C_i| \leq (q+\epsilon)s_i + |W \setminus C_i|.$$

As $|W| \leq (1+\epsilon)\hat{s}$ and $|W \cap C_i| \geq (1-6\epsilon)s_{k_1}$, we have

$$|W \setminus C_i| \leq (1+\epsilon)\hat{s} - (1-6\epsilon)s_{k_1} \leq (1+\epsilon)\hat{s} - \frac{1-6\epsilon}{1+2\epsilon}\hat{s} \leq 9\epsilon\hat{s}.$$

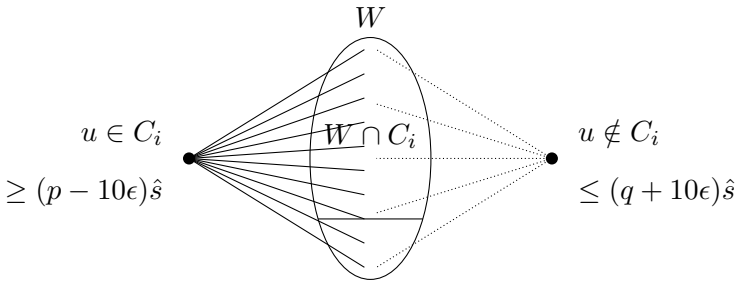

Figure 6. If W has large overlap with C_i, then a.s. vertices in C_i will have many neighbors in W, while vertices not in C_i will have relatively few neighbors in W.

Therefore,

$$|N(u) \cap W| \leq (q + \epsilon)s_i + 9\epsilon\hat{s} \leq (q + 10\epsilon)\hat{s}.$$

This completes the proof of b). □

Thus, if we have a set W which has large intersection with C_i, we can use $|N(u) \cap W|$ to distinguish between $u \in C_i$ and $u \notin C_i$ as shown in Figure 4, provided

$$p - 10\epsilon > q + 10\epsilon,$$

or, equivalently,

$$\epsilon < \frac{p - q}{20}.$$

Note that we apply Lemma 18 to the W_u, which themselves depend on the random sample $\hat{G}$, so we cannot simply treat $|N(u) \cap W|$ as the sum of $|W|$ independent random variables and follow a Hoeffding argument as in Lemma 17. This is why we need both Lemmas 17 and 18.

4.6 The "delete and recurse" step

After we have found one cluster, we cannot simply say that Algorithm 3 finds the remaining clusters by the same argument. Some care has to be taken because the iterations of Algorithm 3 cannot be handled independently: the event that iteration t correctly recovers a cluster certainly depends on whether or not iterations $1, \ldots, t-1$ correctly recovered clusters.

We can get around this by "preprocessing the randomness" as in Section 3.4.3. Essentially, we apply the analysis in Sections 4.3-4.5 to all 2^k *cluster submatrices* of $\hat{B}$ (principle submatrices induced by a subset of the clusters) and show via a union bound that the overall failure probability is still small.

Formally, we define the 2^k cluster submatrices as

$$\hat{B}_J := \hat{B}\left[\bigcup_{i \in J} C_i\right]$$

for $J \subseteq [k]$. We define cluster submatrices of B analogously Next, we define the following events on $\mathcal{G}(n, \mathcal{C}, p, q)$:

- *Spectral events*: for $J \subseteq [k]$, let E_J be the event that $||B_J - \hat{B}_J||_2 \leq 7\sqrt{\dim(B_J)} = 7\sqrt{\sum_{i \in J} s_i}$. These are the events that the eigenvalues of the cluster submatrices are close to their expectations.

- *Degree events*: for $1 \leq i \leq k, 1 \leq u \leq n$, let $D_{i,u}$ be the event that $|N_{\hat{G}}(u) \cap C_i| \geq (p-\epsilon)s_i$ if $u \in C_i$, or the event that $|N_{\hat{G}}(u) \cap C_i| \leq (q+\epsilon)s_i$ if $u \notin C_i$. These are the events that each vertex has approximately the expected number of neighbors in each cluster.

Thus, we have defined a total of $2^k + nk$ events. Essentially, these are the events that every $\hat{B}_J$ satisfies (Equation 4.5) and that (Equation 4.13) and (Equation 4.14) are satisfied for all $i \in [k], u \in [n]$. Note that the events are well-defined, as their definitions depend only on the underlying probability space $\mathcal{G}(n, \mathcal{C}, p, q)$ and not on the random graph $\hat{G}$ sampled from the space.

These final two lemmas prove that Algorithm 3 succeeds with probability at least

$$1 - \left(\frac{2}{e}\right)^{s_k} + \frac{n^{3/2}}{e^{\epsilon^2 s_k}} = 1 - 2^{-\Omega(\sqrt{n})} :$$

Lemma 19. *Assume E_J and $D_{i,u}$ hold for all $J \subseteq [k]$, $1 \leq i \leq k$, and $1 \leq u \leq n$. Then Algorithm 3 successfully recovers $\mathcal{C}$.*

Lemma 20. $\Pr\left[\bigcup_{J \subseteq [k]} \bar{E}_J \cup \bigcup_{i \in [k], j \in [n]} \bar{D}_{i,u}\right] \leq \left(\frac{2}{e}\right)^{s_k} + \frac{n^{3/2}}{e^{\epsilon^2 s_k}} = 2^{-\Omega(\sqrt{n})}.$

We omit the proofs, as they are essentially the same as the proof of the main theorem in Section 3.4.3. The main difference is that one argues that Algorithm 3 first recovers $\mathcal{C}_1$, then $\mathcal{C}_2$, etc. Thus, we really only have to take the union bound over $2^{k_1} + \ldots + 2^{k_K}$ cluster submatrices, not all 2^k.

CHAPTER 5

PLANTED PARTITIONS IN RANDOM SYMMETRIC MATRICES

Copyright note: This chapter is taken from my previous work in (2, Section 11). See Appendix B for copyright information.

We now attempt to push Algorithm 3 to the most general setting possible. In Chapters 3 and 4 we receive as input a random graph or, equivalently, a random symmetric matrix $\hat{A} = (\hat{a}_{uv})_{u,v=1}^{n}$ whose diagonal entries are 0 and whose off-diagonal entries are Bernoulli random variables with expectation p or q. More generally, we can assume that $\hat{A}$ is a random symmetric matrix whose entries come from arbitrary distributions (under certain assumptions) with expectations p and q.

Definition 15 (Planted partition model for random symmetric matrices)**.** *Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be a partition of the set $[n]$ into k clusters. For distributions D_1, D_2, D_3 on $\mathbb{R}$, we define the* planted partition model *$\mathcal{PP}(n, \mathcal{C}, D_1, D_2, D_3)$ to be the probability space of real symmetric $n \times n$ matrices $\hat{A} = (\hat{a}_{uv})_{u,v=1}^{n}$, where $\hat{a}_{uv}$ are distributed independently for $1 \leq u \leq v \leq n$ such that*

$$\hat{a}_{uv} \sim \begin{cases} D_1 & \text{if } u \neq v \text{ and } u, v \text{ in the same cluster} \\ D_2 & \text{if } u \neq v \text{ and } u, v \text{ in different clusters} \\ D_3 & \text{if } u = v \end{cases}.$$

Furthermore, we assume the following:

1. $\mathrm{E}[D_1] = p, \mathrm{E}[D_2] = q$, *and* $\mathrm{E}[D_3] = 0$, *where* $0 \leq q < p$.

2. $\mathrm{Var}[\hat{a}_{uv}] \leq \sigma^2$ *for all* u, v.

3. $|\hat{a}_{uv} - \mathrm{E}[\hat{a}_{uv}]| \leq \kappa$ *for all* u, v. *I.e., the support of each random variable* $\hat{a}_{uv}$ *is contained in an interval of length* 2κ *centered at its mean.*

Problem 2 (Planted partition in a random symmetric matrix)**.** *Identify (or "recover") the unknown partition* $C_1, \ldots, C_k$ *(up to a permutation of* $[k]$*) given only a random matrix* $\hat{A} \sim \mathcal{PP}(n, \mathcal{C}, D_1, D_2, D_3)$.

We will assume that $\mathcal{C}$ satisfies the superclusters assumptions of Section 4.1, with the following changes:

- We replace (Equation 4.1) and (Equation 4.2) with

$$s_1 \geq \ldots \geq s_k \geq \Delta$$

and

$$\min_{C \in \mathcal{C}_i} |C| \geq \max_{C \in \mathcal{C}_{i+1}} |C| + \Delta,$$

respectively.

- We now allow the parameters $p, q, \sigma, \kappa, \Delta$, and ϵ to depend on n.

- We assume without loss of generality that $p > q$, but we no longer require that $0 \leq q < p \leq 1$.

Our goal is thus to give conditions on the various parameters that are sufficient for Algorithm 3 to succeed a.s. (Section 5.4), noting that the algorithm now receives a random matrix $\hat{A} \sim \mathcal{PP}(n, \mathcal{C}, D_1, D_2, D_3)$ as input instead of a random graph $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, p, q)$.

We now indicate briefly the changes to the analysis in Sections 4.3-4.6 necessary to make Algorithm 3 work in this more general setting, leaving the details as an exercise.

5.1 Spectral results

We can define A, B, and $\hat{B}$ as in Section 1.2. The main difference in the spectral results of Section 4.3 is that we get

$$||B - \hat{B}||_2 \leq (2\sigma + 6\kappa)\sqrt{n} \tag{5.1}$$

a.s. in place of Lemma Equation 4.5. This dependence on σ and κ makes sense intuitively because these parameters control how concentrated the entries of $\hat{B}$ are about their means: if they are too spread out, we should not expect B and $\hat{B}$ to be "close." Conversely, if $\sigma, \kappa = o(1)$, then we should expect $\hat{B}$ to be closer to B than when these parameters are constant.

This yields a separation of $\geq (p-q)\Delta - (4\sigma + 12\kappa)\sqrt{n}$ between the eigenvalues of B and $\hat{B}$ corresponding to indices in different superclusters (cf. Lemma 12). This, in turn, allows us to bound $P_{k_1}(B) - P_{k_1}(\hat{B})$ in norm as in Lemma 13, provided that

$$\Delta \geq \frac{(4\sigma + 12\kappa)\sqrt{n}}{(p-q)\epsilon}. \tag{5.2}$$

5.2 Constructing an approximate cluster

In Step 3 of Algorithm 3, define

$$\hat{s} := \frac{\lambda_1(\hat{B}) + (2\sigma + 6\kappa)\sqrt{n}}{p - q}.$$

Then (Equation 4.10) holds as in the Bernoulli case, assuming (Equation 5.1) holds. Thus, Lemmas 15 and 16 remain exactly the same as in the Bernoulli case (except replace (Equation 4.5) with (Equation 5.1)).

5.3 Recovering the cluster exactly

In the Bernoulli case, we use the random variables

$$|N_{\hat{G}}(u) \cap C_i| = \sum_{v \in C_i} \hat{a}_{uv}$$

to distinguish between $u \in C_i$ and $u \notin C_i$ (Lemmas 17 and 18). Thus, for general distributions we define the random variable

$$S_{u,W} := \sum_{v \in W} \hat{a}_{uv}$$

for $u \in [n]$ and $W \subseteq [n]$. (Recall that a random variable is actually a measurable function from a probability space to $\mathbb{R}$; hence, $S_{u,W}$ is actually a function of the random matrix $\hat{A}$.)

Proceeding as in Lemma 17, for each cluster C_i we get

$$S_{u,C_i} \geq (p - \epsilon)s_i \tag{5.3}$$

for all $u \in C_i$, and

$$S_{u,C_i} \leq (q+\epsilon)s_i \tag{5.4}$$

for all $u \notin C_i$, each with probability $\geq 1 - \exp\left(-\frac{\epsilon^2 s_i}{3\kappa^2}\right)$.

We now argue that if W has large intersection with some C_i, then we get bounds on $S_{u,W}$ which are not far off from those on S_{u,C_i}. However, since the entries of $\hat{A}$ need not be 0 or 1, each element of $W \triangle C_i$ can throw off the bounds by as much as

$$\max_{u,v} |\hat{a}_{uv}| \leq \mu + \kappa,$$

where

$$\mu := \max\{|p|, |q|\}.$$

More precisely:

Lemma 21. *Assume* (Equation 5.1) *holds. Suppose* $|W| \leq (1+\epsilon)\hat{s}$ *and* $|W \cap C_i| \geq (1-6\epsilon)s_k$ *for some* $C_i \in \mathcal{C}_1$. *Then*

a) If $u \in C_i$ *and* u *satisfies (Equation 5.3), then* $S_{u,W} \geq (p - (18\mu + 16\kappa + 1)\epsilon)\hat{s}$.

b) If $u \in [n] \setminus C_i$ *and* u *satisfies (Equation 5.4), then* $S_{u,W} \leq (q + (16\mu + 16\kappa + 1)\epsilon)\hat{s}$.

We omit the proof, as it parallels that of Lemma 18.

Thus, we are able to recover the cluster a.s. as long as

$$p - (18\mu + 16\kappa + 1)\epsilon > q + (16\mu + 16\epsilon + 1)\epsilon,$$

or equivalently,

$$\epsilon < \frac{p-q}{34\mu + 32\kappa + 2}. \tag{5.5}$$

It may be possible to optimize these constants slightly by breaking the proof of Lemma 21 into cases based on whether p and q are positive or negative, but there will always be a dependence on p, q and κ.

5.4 Parameter dependencies

By (Equation 5.2), the above inequality (Equation 5.5) is satisfied if

$$\frac{(2\sigma + 6\kappa)\sqrt{n}}{(p-q)\Delta} \leq \frac{p-q}{34\mu + 32\kappa + 2}.$$

This can be accomplished if we require

$$\Delta = \Omega\left(\frac{\kappa(\max\{|p|,|q|\} + \kappa)\sqrt{n}}{(p-q)^2}\right). \tag{5.6}$$

In addition, observe that we need the failure probability of $\exp\left(-\frac{\epsilon^2 s_i}{3\kappa^2}\right)$ above to be $o((nk)^{-1})$, since we take a union bound over nk "degree events" (see Section 4.6). This can be accomplished if we require

$$\Delta = \omega\left(\frac{(\max\{|p|,|q|\} + \kappa)^2\kappa^2 \log n}{(p-q)^2}\right). \tag{5.7}$$

Thus, we get the following theorem:

Theorem 18. *Let $\mathcal{C}$ be defined as in the beginning of Chapter 5, and assume that* (Equation 5.5)-(Equation 5.7) *are satisfied. Then $\mathcal{C}$ can be recovered a.s. in polynomial time given only $\hat{A} \sim \mathcal{PP}(n, \mathcal{C}, D_1, D_2, D_3)$.*

Observe that since $\Delta \leq n$, (Equation 5.6) and (Equation 5.7) together imply

$$\kappa(\max\{|p|, |q|\} + \kappa\}) \ll \frac{\sqrt{n}}{\log n}.$$

So a necessary condition for Algorithm 3's success is that $|p|, |q|$, and κ aren't too big. On the other hand, if κ is small (i.e., the entries of $\hat{B}$ are highly concentrated), we can potentially get away with a smaller-than-$\sqrt{n}$ separation between the cluster sizes. Note also that our goal was to make Algorithm 3 work in the *most general* setting possible; it may be possible to obtain better conditions on the parameters in certain special cases. One can do this by mirroring the analysis in Sections 4.3-4.6.

CHAPTER 6

NUMERICAL RESULTS

In this chapter we show how our algorithms perform in practice. We performed the following experiment:

1. For $n = 100, 200, \ldots, 3600$, and $k = 2, 3, \ldots, k_{\max}(n)$, generate 20 random graphs from a planted partition distribution with $p = .99$, $q = .01$ and k parts of size $\lfloor n/k \rfloor$ or $\lfloor n/k \rfloor + 1$.
2. For each (n, k) pair, run Algorithm 3 on the graphs generated and record the number for which the planted partition was successfully recovered.
3. For each n, $k_{\max}(n)$ is the smallest k for which the proportion of successes was $< .6$.

The results are summarized in Figure 7. Note that for simplicity we are testing on planted equipartitions, but we implemented Algorithm 3 rather than Algorithm 2 because it works in a more general setting.

Essentially, for each n we want to find the largest $k = k_{\max}(n) - 1$ for which planted k-equipartitions can be reliably recovered. By Theorem 17, this k should be $\Theta(\sqrt{n})$. From Figure 7, $k_{\max}(n)$ certainly appears to be an increasing, concave function of n, though it is unclear whether it is $\Theta(\sqrt{n})$ due to the limited amount of data. Obtaining more convincing numerical results is a work in progress.

The main obstacle to obtaining more satisfying numerical results is the running time of our algorithm. See Section 3.1.1. One area where there is room for improvement is the computation

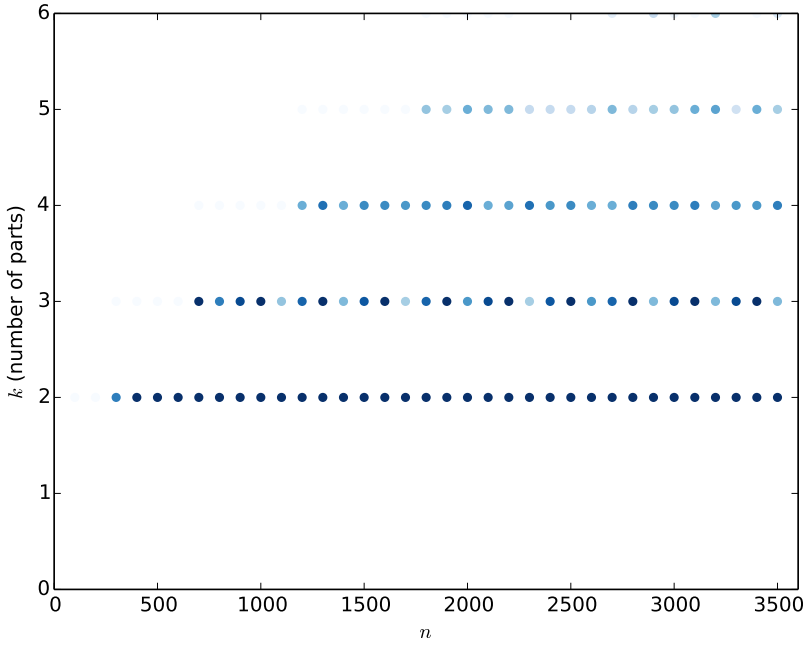

Figure 7. Success of Algorithm 3 on planted equipartitions with n vertices and k clusters, taken over 20 trials, with $p = .99, q = .01$

of the dominant rank-k projector $P_k(\hat{B})$. As noted in Section 3.1.1, this can be done in $O(n^2k)$ time by performing a truncated SVD (only computing the largest k singular values and singular vectors). However, our algorithm is implemented naïvely using the full SVD, which takes $\Theta(n^3)$ time. Thus, our the current implementation of our algorithm takes $\Theta(n^3k)$ time instead of $O(n^2k^2)$. This could represent a significant slowdown when n is large and k is small, as is the case in the above experiment.

Figure 8 shows the average running time of our algorithm on planted equipartitions with 2, 3, 4, 5, and 6 clusters. For each (n, k) pair, the algorithm was run on 20 random graphs from a planted partition distribution with $p = .99$, $q = .01$ and k parts of size $\lfloor n/k \rfloor$ or $\lfloor n/k \rfloor + 1$.

Algorithm 3 and associated experiments were implemented in Python™ using the SciPy and Matplotlib. The code can be found online at `https://github.com/smpcole/planted-partition`.

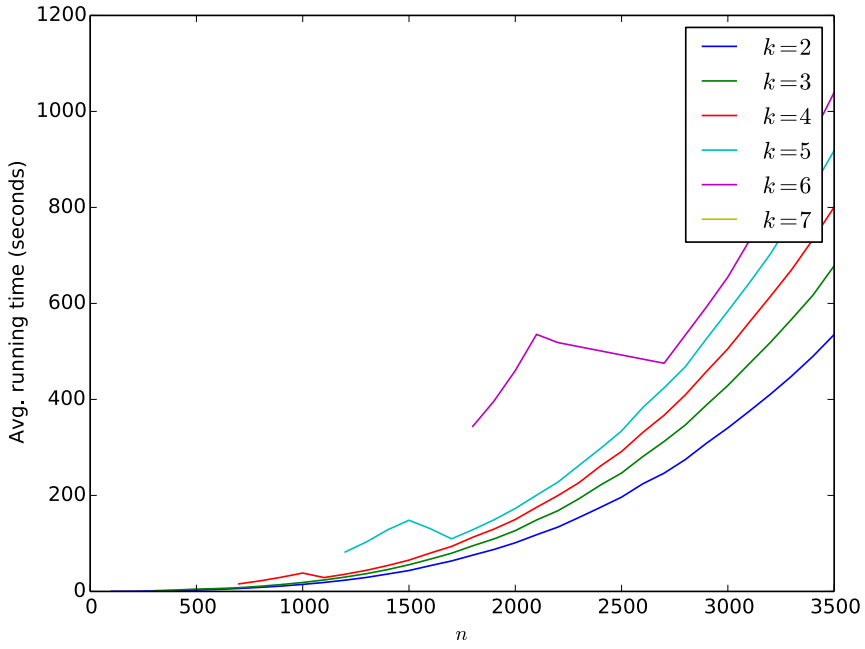

Figure 8. Average running time of Algorithm 3 on planted equipartitions with n vertices and k clusters, taken over 20 trials, with $p = .99, q = .01$

CHAPTER 7

COMPARISON WITH PREVIOUS RESULTS

In this section we compare the performance of Algorithm 3 with that of several well-known spectral algorithms on both dense and sparse random graphs. In both cases, we will assume for simplicity that all parts of the planted partition are the same size, i.e. $s_i = s := n/k$ for $i = 1, \ldots, k$.

Copyright note: This chapter is taken from my previous work in (2, Section 12). See Appendix B for copyright information.

7.0.1 Dense random graphs, many clusters

As noted in the introduction, Algorithm 3 compares favorably to well-known spectral algorithms on dense random graphs with many clusters of size $\Omega(\sqrt{n})$. More precisely, let us assume p and q are fixed constants and all clusters are size $s = n/k$. We would like to know the smallest s (i.e., largest k) for which each algorithm is guaranteed to recover the planted partition a.s. Table I summarizes the algorithms' guarantees:

To the best of the author's knowledge, Algorithm 3 is the first *spectral* algorithm which recovers planted partitions in which *all* clusters are size $\Theta(\sqrt{n})$ in polynomial time, though other algorithms (e.g. (7; 8; 9)) are able to achieve this using convex or semidefinite programming. Note that the above algorithms are able to recover planted partitions in which *some* of the clusters are size $\Theta(\sqrt{n})$, but only if there are also clusters of size $\omega(\sqrt{n})$.

Paper			Minimum cluster size
Coja-Oghlan	2010	(10, Theorem 2)	$\Omega(n^{4/5})$
McSherry	2001	(11, Theorem 4)	$\Omega(n^{2/3})$
Vu	2014	(12, Theorem 2)	$\omega(\sqrt{n \log n})$
This result		Theorem 17	$\Omega(\sqrt{n})$

TABLE I

Comparison of results in the dense regime

Also note that the above algorithms apply to more general planted partition distributions than ours. In particular, they allow for edge probabilities p_{ij} between pairs of vertices in clusters C_i and C_j, for $i, j = 1, \ldots, k$. In particular, this allows there to be "unclustered" vertices, i.e., clusters whose intra-cluster probability is no greater than the background edge probability. Extending Algorithm 3 to this setting remains an open problem.

7.0.2 Sparse random graphs, constant number of clusters of linear size

Planted partitions with many clusters of size $\Theta(\sqrt{n})$ are of theoretical interest because they approach the conjectured $\sqrt{n}$-barrier for efficient algorithms, but the setting in which there are few edges and a constant number of clusters of linear size is perhaps more realistic in practice. More precisely, we will assume $p, q = o(1)$ and $k = O(1)$ (hence $s = \Omega(n)$). For simplicity, we

Paper			Minimum edge probability
Coja-Oghlan	2010	(10, Theorem 2)	$\Omega(\log n/n)$
McSherry	2001	(11, Theorem 4)	$\Omega(\log^6 n/n)$
Vu	2014	(12, Theorem 2)	$\Omega(\sqrt{\log n/n})$
This result		Theorem 18	$\Omega(n^{-1/4})$

TABLE II

Comparison of results in the sparse regime

will further assume that $p = \Theta(q)$. We are now interested in the smallest p and q for which each algorithm is guaranteed to recover the planted partition a.s.

If we apply Theorem 18 with $\Delta = \Theta(n)$, $\kappa = 1$, and $p = \Theta(q) = o(1)$, then (Equation 5.6) implies

$$\frac{\sqrt{n}}{p^2} \leq \frac{\sqrt{n}}{(p-q)^2} = O(n).$$

Hence, we get $p, q \geq \Omega(n^{-1/4})$. Table II compares this to the guarantees of well-known spectral algorithms:

Thus, the above algorithms all outperform Algorithm 3 on sparse random graphs. Optimizing Algorithm 3 for the sparse regime is a possible direction for future work.

CHAPTER 8

FUTURE WORK

8.1 Parameter-free planted partition

Until this point, we have assumed that our algorithm has access to p, q, and $k_1, \ldots, k_K$, but not $s_1, \ldots, s_k$. In this section, we discuss what to do when we don't have access to these parameters' exact values.

Copyright note: This section is taken from my previous work in (2, Section 10). See Appendix B for copyright information.

8.1.1 Unknown supercluster sizes

Let us assume that *only* p and q are known. As it turns out, we can reduce this case to the case when $k_1, \ldots, k_K$ are known at the expense of slightly increasing c.

Assume (Equation 4.5) holds. Then by (Equation 4.2)

$$\begin{array}{l} C_i, C_{i+1} \text{ in different} \\ \text{superclusters} \end{array} \Rightarrow \lambda_i(\hat{B}) \geq \lambda_{i+1}(\hat{B}) + ((p-q)c - 14)\sqrt{n}. \tag{8.1}$$

Thus, let us go down the list of eigenvalues of $\hat{B}$ in decreasing order and whenever we see a separation of at least $((p-q)c-14)\sqrt{n}$ record the number of eigenvalues seen since the last such separation. Ignore the last group, as these eigenvalues correspond to the $n-k$ zero eigenvalues of B. Let $\hat{k}_1, \ldots, \hat{k}_L$ be this sequence of numbers.

Consider the clustering

$$\begin{aligned}
\hat{\mathcal{C}}_1 &:= \{C_1, \ldots, C_{\hat{k}_1}\}, \\
\hat{\mathcal{C}}_2 &:= \{C_{\hat{k}_1+1}, \ldots, C_{\hat{k}_1+\hat{k}_2}\}, \\
&\vdots \\
\hat{\mathcal{C}}_L &:= \{C_{\hat{k}_1+\ldots+\hat{k}_{L-1}+1}, \ldots, C_{\hat{k}_1+\ldots+\hat{k}_L}\}.
\end{aligned}$$

Observation 2. *If* (Equation 4.5) *holds, then* $\hat{\mathcal{C}}_1, \ldots, \hat{\mathcal{C}}_L$ *is a refinement of* $\mathcal{C}_1, \ldots, \mathcal{C}_K$ *(as partitions of* $\mathcal{C}$*).*

Proof. Consider $\hat{\mathcal{C}}_1$. By definition of $\hat{k}_1$,

$$\lambda_i(\hat{B}) < \lambda_{i+1}(\hat{B}) - ((p-q)c - 14)\sqrt{n}$$

for $i = 1, \ldots, \hat{k}_1 - 1$. By the contrapositive of (Equation 8.1), this means C_i, C_{i+1} are in the same $\mathcal{C}_j$ for $i = 1, \ldots, \hat{k}_1 - 1$. A similar argument applies for each $\hat{\mathcal{C}}_i$. Hence, each $\hat{\mathcal{C}}_i$ is a subset of some $\mathcal{C}_j$. □

Observation 3. *Assume* (Equation 4.5) *holds. Then* $s_i \geq s_{i+1} + \left(c - \frac{28}{p-q}\right)\sqrt{n}$ *whenever* C_i, C_{i+1} *are in different* $\hat{\mathcal{C}}_j$*. Equivalently,*

$$\min_{C \in \hat{\mathcal{C}}_j} |C| \geq \max_{C \in \hat{\mathcal{C}}_{j+1}} |C| + \left(c - \frac{28}{p-q}\right)\sqrt{n}$$

for $i = 1, \ldots, L-1$.

Thus, by these two observations, we a.s. have a separation into superclusters $\hat{\mathcal{C}}_1, \ldots, \hat{\mathcal{C}}_L$ such that

- $s_i \geq s_{i+1} + c'\sqrt{n}$ whenever C_i, C_{i+1} are in different $\hat{\mathcal{C}}_j$, where

$$c' := c - \frac{28}{p-q} \tag{8.2}$$

  (by Observation 3).

- $s_i \leq (1+\epsilon)s_{i+1}$ whenever C_i, C_{i+1} are in the same $\hat{\mathcal{C}}_j$ (by Observation 2 and (Equation 4.4)).

This is sufficient to recover $\mathcal{C}$, since we only care about recovering the individual clusters, not which supercluster each cluster belongs to.

Note that since $c = \Omega((p-q)^{-2})$, c' is still a large positive constant. Hence, in order for the analysis in Sections 4.3-4.6 to go through, we simply require

$$c \geq \frac{14}{(p-q)\epsilon} + \frac{28}{p-q} = \frac{14}{(p-q)\epsilon}(1+2\epsilon).$$

in place of (Equation 4.9). As $\epsilon \leq .01$, setting $c \geq \frac{15}{(p-q)\epsilon}$ suffices.

8.1.2 Unknown edge probabilities

Unfortunately, we still need to assume that Algorithm 3 has access to the exact values of p and q so that it can compute $\hat{B} := \hat{A} + pI_n - qJ_n$. There are two possible ways around this:

1. Obtain good estimates on the eigenvalues of A so that the algorithm can use $\hat{A}$ instead of $\hat{B}$, as in the equitable case. Since

$$A = B + qJ_n, \tag{8.3}$$

Weyl's inequalities give the bounds

$$(p-q)s_i \leq \lambda_i(A) \leq (p-q)s_{i-1}$$

for $i = 2, \ldots, k$. However, in order for the spectral results in Sections 4.3 and 4.4 to go through we need a separation between $\lambda_i(A)$ and $\lambda_{i-1}(A)$ when C_i and C_{i-1} are in different superclusters. Thus, the above bounds are not good enough. However, by (Equation 8.3) we may view A as a rank-1 perturbation of B (whose eigenvalues are known), so we may attempt to use perturbation results such as (43; 44; 45) to compute its eigenvalues.

2. Estimate p and q empirically. We must find a way to do so with $\leq O(1/\sqrt{n})$ error in order to overcome the $O(\sqrt{n})$ error introduced by the random noise (see Lemma 11). One promising approach is to use the theory of *graphons* (46). In (47; 48), the authors use the theory of large deviations (49) to show that the edge and triangle or l-star densities of "most" graphs together induce a multipodal (i.e. stochastic block model) structure in the limiting graphon. Thus, one might hope to estimate p, q, and $s_1, \ldots, s_k$ by looking at the graphon induced by these statistics on $\hat{G}$.

.

8.2 Lower bounds

In this section we discuss limitations of efficient algorithms for planted partition. In Section 8.2.1 we give an informal discussion of why planted partition becomes impossible when the cluster sizes are $O(\log n)$; however, it is only known how to efficiently recover planted partitions in which the cluster sizes are $\Omega(\sqrt{n})$. Closing the gap between $\log n$ and $\sqrt{n}$ is a major open problem in computer science. It is generally believed that the problem becomes intractable when the cluster sizes are $o(\sqrt{n})$, but proving this formally in the framework of complexity theory seems rather hopeless (Section 8.2.2); instead, one may attempt to show that specific algorithms or classes of algorithms provably fail when the cluster sizes are too small.

8.2.1 Information theoretic lower bound

In this section, we will show that if the sizes of the parts in a planted partition is $O(\log n)$, then there is no hope of recovering the planted partition with high probability. Essentially, an Erdős-Rényi random graph will a.s. have a partition of the vertices into cliques of size $\Omega(\log n)$; thus, a planted partition in which the parts are size $O(\log n)$ would be indistinguishable from a "naturally occurring" clique partition in a random graph.

Let us formalize this a bit. We will first need to define the notions of an *equitable coloring* and the *equitable chromatic number* of a graph (50):

Definition 16 (Equitable coloring)**.** *An* equitable k-coloring *of a graph $G = (V, E)$ is a vertex coloring of G with k colors such that sizes of any two color classes differ by at most 1.*

That is, an equitable coloring is a vertex coloring in which all color classes are approximately the same size.

Definition 17 (Equitable chromatic number)**.** *The* equitable chromatic number *of a graph $G = (V, E)$, denoted $\chi_=(G)$, is the smallest integer k such that G has a proper equitable k-coloring (i.e., no two vertices in the same color class share an edge).*

In (50), the authors prove that the equitable chromatic number of a random graph is approximately the same as its chromatic number:

Theorem 19. *Let $G(n, p)$ be an Erdős-Rényi random graph, where $n^{-1/5+\epsilon} < p < .99$ for some $\epsilon > 0$. Then a.s. $\chi(G(n,p)) \leq \chi_=(G(n,p)) \leq (1 + o(1))\chi(G(n,p))$.*

Recall the classic result of Bollobás (51) that the chromatic number of $G(n, p)$ is $\Theta(\log n)$:

Theorem 20. *For any fixed $p \in (0, 1)$, $\chi(G(n,p)) = \left(\frac{1}{2} + o(1)\right) \frac{n}{\log_{1/(1-p)} n}$ a.s.*

Thus, we get the following as a corollary:

Corollary 1. *Let $p \in (0, .99)$ be fixed. Then a.s. both of the following are true:*

1. $\chi_=(G(n,p)) = \left(\frac{1}{2} + o(1)\right) \frac{n}{\log_{1/(1-p)} n}$
2. *There exists a proper coloring of $G(n, p)$ in which all color classes are size $(2-o(1)) \log_{1/(1-p)} n$.*

Finally, observe that a partition of the vertices of a graph $G = (V, E)$ into disjoint cliques corresponds to a coloring of its complement $\bar{G}$. Thus, by part 2 of Corollary 1 we get the following:

Corollary 2. *Let $p \in (.01, 1)$ be fixed. Then a.s. the vertices $G(n, p)$ can be partitioned into cliques of size $(2 - o(1)) \log_{1/p} n$.*

Note that the size of the cliques in the above corollary is approximately the same as the clique number of $G(n, p)$ (52).

How does this relate to planted partition? Consider a planted partition distribution $\mathcal{G}(n, \mathcal{C}, 1, q)$ in which edges are added *deterministically* within the parts of the partition. Assume that all parts are size $s \leq (2-\epsilon)\log_{1/q} n$ for some constant $\epsilon > 0$. Then by Corollary 2 $\hat{G} \sim \mathcal{G}(n, \mathcal{C}, 1, q)$ will a.s. contain at least two distinct partitions into cliques of size s: the planted partition, and one guaranteed to exist by Corollary 2.

Formalizing this graph theoretic argument is a possible direction for future work. More formal information theoretic lower bounds have been proved in (21, Section 2.1) and (53; 54).

8.2.2 Average case complexity

Copyright note: Parts of this section are taken from the introduction of my paper (2). See Appendix B for copyright information.

We may view "planted" problems like planted clique as distributional versions of classic NP-hard problems like MAX-CLIQUE, MAX-CUT, or SPARSEST-CUT (55). Ideally, one would hope to prove such problems are distNP-complete if the cluster sizes are $o(\sqrt{n})$, where distP and distNP are distributional analogues of P and NP (56)). However, this goal seems unrealistic at present, as showing the existence of a "natural" distNP-complete problem is a major unresolved problem in complexity theory (see (57, Chapter 18)). Instead, various authors have shown that specific algorithmic techniques fail when the size of the parts is too small (17; 18; 15). We discuss one quite general class of algorithms below.

8.2.3 Statistical lower bound

A *statistical algorithm* is one which, instead of receiving a random graph $\hat{G}$ as input, can query an oracle for the values of statistics on the distribution of $\hat{G}$ within some tolerance τ (58). The authors of that paper show that for the related problem of *planted clique* (14; 15; 16), statistical algorithms require exponentially many queries when the size of the planted clique is $\leq n^{\frac{1}{2}-\delta}$. At present, no such result exists for planted partition.

The proof of the statistical lower bound for planted clique hinges on showing that any sufficiently large family of s-subsets of the vertices will have small average pairwise intersection. However, complications arise when trying to apply this technique to planted partition: it is not clear what the appropriate notion of "intersection" is for partitions, and there are far more partitions of a set of size n into sets of size $s = O(\sqrt{n})$ than there are subsets of size s. One possibility is to attempt to show that a large family of partitions must have small average pairwise *overlap*, where the overlap of two partitions $\mathcal{P}, \mathcal{Q}$ of a finite set S is defined as $\max_{P \in \mathcal{P}, Q \in \mathcal{Q}} |P \cap Q|$.

APPENDICES

Appendix A

NOTATION

Copyright note: Parts of this appendix were taken from (2, Section 4.1). See Appendix B for copyright information.

- $\sim$ – "sampled from"—i.e., if $\mathcal{P}$ is a probability space, then $\omega \sim \mathcal{P}$ means ω is a random sample drawn from $\mathcal{P}$.
- $\langle \cdot, \cdot \rangle$ – standard inner product of two vectors in $\mathbb{R}^n$ or $\mathbb{C}^n$.
- $|\cdot|$ – size of a finite set or absolute value of a real or complex number
- $||\cdot||_*$ – trace norm of a matrix (24, Theorem 5.6.42)
- $||\cdot||_F$ – the Frobenius norm of a matrix (Definition 11).
- $||\cdot||_p$ – the ℓ_p- norm of a vector or matrix (Definition 10).
- $\mathbf{0}$ – the 0 vector in a vector space.
- $\mathbf{1}_S$ – the indicator vector $\in \{0,1\}^n$ for the set $S \subseteq [n]$.
- $\mathbf{1}_n$ – the all 1s vector $\in \mathbb{R}^n$, i.e. $\mathbf{1}_{[n]}$.
- $A^\top$ – transpose of A
- A^* – conjugate transpose of A
- $A(G)$ – adjacency matrix of a graph G (Definition 7)
- $A[S]$ – the principal submatrix of A with row and column indices restricted to S.

Appendix A (Continued)

- a.s. – almost surely, i.e. with probability $1 - o(1)$ as $n \to \infty$ (Definition 6).
- $\mathbb{C}$ – the set of complex numbers
- $d_G(v) := |N_G(v)|$ – the degree of vertex v in G. Again, we will omit the subscript when the meaning is clear.
- $\dim(A)$ – the dimension of a square matrix A. If A is $n \times n$, then $\dim(A) = n$.
- $\bar{E}$ – complement of event E (the probability space is implied by context)
- $\mathrm{E}_D[X]$ – the expectation of a random variable X over a distribution D. If X is matrix or vector valued, then the expectation is taken entrywise. The distribution may be omitted if it is clear from context.
- $\mathbf{e}_1, \ldots, \mathbf{e}_n$ – standard basis vectors in $\mathbb{R}^n$.
- $\mathbb{F}^{m \times n}$ – vector space of $m \times n$ matrices over the field $\mathbb{F}$.
- $\mathbb{F}^n$ – vector space of dimension n over the field $\mathbb{F}$. Vectors are represented as ordered n-tuples in the standard basis.
- $G = (V, E)$ – a simple, undirected graph with vertex set V and edge set E.
- $\mathcal{G}(n, \mathcal{C}, p, q)$ – the planted partition (also known as stochastic block model) distribution (Definition 1)
- $G(n, p)$ – a single Erdős-Rényi random graph sampled from the distribution $\mathcal{G}(n, p)$ (Definition 8)
- $\mathcal{G}(n, p)$ – the Erdős-Rényi random graph distribution (Definition 8)

Appendix A (Continued)

- $G[S]$ – the induced subgraph of G on $S \subseteq V(G)$.
- $H(\mathcal{P})$ – incidence matrix of a partition $\mathcal{P}$ (Definition 4)
- I_n – the $n \times n$ identity matrix.
- J_n – the $n \times n$ 1s matrix.
- $N_G(v)$ – neighborhood of vertex v in a graph G. We will omit the subscript G when the meaning is clear.
- $[n] := \{1, \ldots, n\}$
- $O(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$, $o(\cdot)$ – asymptotic notation. Roughly speaking,
  - $f(n) = O(g(n))$ means $f(n)$ grows at most as fast as $g(n)$.
  - $f(n) = \Omega(g(n))$ means $f(n)$ grows at least as fast as $g(n)$.
  - $f(n) = \Theta(g(n))$ means $f(n)$ grows at "about the same rate" as $g(n)$.
  - $f(n) = o(g(n))$ means $f(n)$ grows "much slower" than $g(n)$.
  - $f(n) = \omega(g(n))$ means $f(n)$ grows "much faster" than $g(n)$.

  See, e.g., (59) for formal definitions.
- $P_k(A)$ – dominant rank-k projector of A (Definition 14)
- $\mathcal{PP}(n, \mathcal{C}, D_1, D_2, D_3)$ – the planted partition distribution for random symmetric matrices (Definition 15)
- $\Pr_D[A]$ - probability of event A over distribution D. The distribution may be omitted if it is clear from context.

Appendix A (Continued)

- $P_{\mathbf{U}}$ – orthogonal projection operator onto subspace $\mathbf{U}$ (Definition 13)
- $\mathbb{R}$ – the set of real numbers
- $\mathrm{rk}(\cdot)$ – rank of a matrix
- $\mathrm{tr}(\cdot)$ – trace of a matrix
- $\mathrm{Var}_D[X]$ – variance of a random variable X over distribution D. The distribution may be omitted if it is clear from context.
- w.h.p. – same as a.s. (Definition 6)
- $\lambda_i(A)$ – the ith largest eigenvalue of a symmetric matrix A (recall from Theorem 6 that symmetric matrices have real eigenvalues).
- $\lambda_i(G)$ – the ith largest eigenvalue of G's adjacency matrix.

Appendix B

COPYRIGHT INFORMATION

The following parts of this thesis are taken from taken from my paper with Shmuel Friedland and Lev Reyzin in the journal Special Matrices, published by DeGruyter (1):

- Chapter 3
- Sections 2.4.2 and 2.5
- Parts of the introduction and summary

This journal is fully open access, and therefore we retain the copyright to our article. A copy of the license is included below, and the Journal's open access statement can be found here: `http://degruyteropen.com/you/journal-author/open-access-statement/`.

The following parts are taken from my upcoming paper (2):

- Chapters 4, 5, and 7
- Sections 1.1, 1.2, 2.4.2, and 8.1
- Parts of the introduction and Section 8.2.2

This work was recently accepted to the journal Linear Algebra and its Applications, published by Elsevier. It is currently available online and is set to appear in print later this year. A copy of the license is included below.

LICENSE TO PUBLISH

Please read the terms of this agreement, print, sign, scan and send the document via online submission system Editorial Manager available for Non-Genetic Inheritance ("Journal") at http://www.editorialmanager.com/ngi/

De Gruyter Open Ltd. ("Journal Owner", also referred to as "You" in the Creative Commons license mentioned in section 1 below)

Article entitled ("Work" or "article"): A simple spectral algorithm for recovering planted partitions

Author/s: (also referred to as "Licensor/s") Sam Cole, Shmuel Friedland, Lev Reyzin

Corresponding author: (if more than one author) Sam Cole

1. License

The non-commercial use of the article will be governed by the Creative Commons Attribution-NonCommercial-NoDerivs license as currently displayed on http://creativecommons.org/licenses/by-nc-nd/3.0/., except that sections 2 through 8 below will apply in this respect and prevail over all conflicting provisions of such license model. Without prejudice to the foregoing, the author hereby grants the Journal Owner the exclusive license for commercial use of the article (for U.S. government employees: to the extent transferable) according to section 2 below, and sections 4 through 9 below, throughout the world, in any form, in any language, for the full term of copyright, effective upon acceptance for publication.

2. Author's Warranties

The author warrants that the article is original, written by stated author/s, is currently not under consideration, has not been published before, contains no unlawful statements, does not infringe the rights of others, is subject to copyright that is vested exclusively in the author and free of any third party rights, and that any necessary written permissions to quote from other sources have been obtained by the author/s.

3. User Rights

Under the Creative Commons Attribution-NonCommercial-NoDerivs license, the author(s) and users are free to share (copy, distribute and transmit the contribution) under the following conditions: 1. they must attribute the contribution in the manner specified by the author or licensor, 2. they may not use this contribution for commercial purposes, 3. they may not alter, transform, or build upon this work.

4. Rights of Authors

Authors retain the following rights:

- copyright, and other proprietary rights relating to the article, such as patent rights,
- the right to use the substance of the article in future own works, including lectures and books,
- the right to reproduce the article for own purposes, provided the copies are not offered for sale,
- the right to self-archive the article.

5. Co-Authorship

If the article was prepared jointly with other authors, the signatory of this form warrants that he/she has been authorized by all co-authors to sign this agreement on their behalf, and agrees to inform his/her co-authors of the terms of this agreement.

6. Termination

This agreement can be terminated by the author or the Journal Owner upon two months' notice where the other party has materially breached this agreement and failed to remedy such breach within a month of being given the terminating party's notice requesting such breach to be remedied. No breach or violation of this agreement will cause this agreement or any license granted in it to terminate automatically or affect the definition of Journal Owner. After the lapse of forty (40) years of the date of this agreement, this agreement can be terminated without cause by the author or the Journal Owner upon two years' notice. The author and the Journal Owner may agree to terminate this agreement at any time. This agreement or any license granted in it cannot be terminated otherwise than in accordance with this section 6.

7. Royalties

This agreement entitles the author to no royalties or other fees. To such extent as legally permissible, the author waives his or her right to collect royalties relative to the article in respect of any use of the article by the Journal Owner or its sublicensee.

8. Miscellaneous

The Journal Owner will publish the article (or have it published) in the Journal, if the article's editorial process is successfully completed and the Journal Owner or its sublicensee has become obligated to have the article published. Where such obligation depends on the payment of a fee, it shall not be deemed to exist until such time as that fee is paid. The Journal Owner may conform the article to a style of punctuation, spelling, capitalization and usage that it deems appropriate. The author acknowledges that the article may be published so that it will be publicly accessible and such access will be free of charge for the readers. The Journal Owner will be allowed to sublicense the rights that are licensed to it under this agreement. This agreement will be governed by the laws of England and Wales.

9. Scope of the Commercial License

The exclusive right and license granted under this agreement to the Journal Owner for commercial use is as follows:

- to prepare, reproduce, manufacture, publish, distribute, exhibit, advertise, promote, license and sub-license printed and electronic copies of the article, through the Internet and other means of data transmission now known or later to be developed; the foregoing will include abstracts, bibliographic information, illustrations, pictures, indexes and subject headings and other proprietary materials contained in the article,
- to exercise, license, and sub-license others to exercise subsidiary and other rights in the article, including the right to photocopy, scan or reproduce copies thereof, to reproduce excerpts from the article in other works, and to reproduce copies of the article as part of compilations with other works, including collections of materials made for use in classes for instructional purposes, customized works, electronic databases, document delivery, and other information services, and publish, distribute, exhibit and license the same.

Where this agreement refers to a license granted to Journal Owner in this agreement as exclusive, the author commits not only to refrain from granting such license to a third party but also to refrain from exercising the right that is the subject of such license otherwise than by performing this agreement.
The Journal Owner will be entitled to enforce in respect of third parties, to such extent as permitted by law, the rights licensed to it under this agreement.
If the article was written in the course of employment by the US or UK Government, and/or arises from NIH funding, please consult the Journal Owner for further instructions.

Author's Signature: Sam Cole

Name printed: Sam Cole

Date: 11/24/16

DE GRUYTER OPEN

RIGHTS & ACCESS

Elsevier Inc.

Article: Recovering Nonuniform Planted Partitions via Iterated Projection
Corresponding author: Mr. Sam Cole
E-mail address: smpcole@gmail.com
Journal: Linear Algebra and Its Applications
Our reference LAA14501
PII: S0024-3795(18)30119-8
DOI: 10.1016/j.laa.2018.03.009

YOUR STATUS

» I am the sole author of the manuscript

ASSIGNMENT OF COPYRIGHT

I hereby assign to Elsevier Inc. the copyright in the manuscript identified above (where Crown Copyright is asserted, authors agree to grant an exclusive publishing and distribution license) and any tables, illustrations or other material submitted for publication as part of the manuscript (the "Article"). This assignment of rights means that I have granted to Elsevier Inc., the exclusive right to publish and reproduce the Article, or any part of the Article, in print, electronic and all other media (whether now known or later developed), in any form, in all languages, throughout the world, for the full term of copyright, and the right to license others to do the same, effective when the Article is accepted for publication. This includes the right to enforce the rights granted hereunder against third parties.

SUPPLEMENTAL MATERIALS

"Supplemental Materials" shall mean materials published as a supplemental part of the Article, including but not limited to graphical, illustrative, video and audio material.

With respect to any Supplemental Materials that I submit, Elsevier Inc. shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Supplemental Materials in all forms and media (whether now known or later developed), and to permit others to do so.

RESEARCH DATA

"Research Data" shall mean the result of observations or experimentation that validate research findings and that are published separate to the Article, which can include but are not limited to raw data, processed data, software, algorithms, protocols, and methods.

With respect to any Research Data that I wish to make accessible on a site or through a service of Elsevier Inc., Elsevier Inc. shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Research Data in all forms and media (whether now known or later developed) and to permit others to do so. Where I have selected a specific end user license under which the Research Data is to be made available on a site or through a service, the publisher shall apply that end user license to the Research Data on that site or service.

REVERSION OF RIGHTS

Articles may sometimes be accepted for publication but later rejected in the publication process, even in some cases after public posting in "Articles in Press" form, in which case all rights will revert to the author (see https://www.elsevier.com/about/our-business/policies/article-withdrawal).

REVISIONS AND ADDENDA

I understand that no revisions, additional terms or addenda to this Journal Publishing Agreement can be accepted without Elsevier Inc.'s express written consent. I understand that this Journal Publishing Agreement supersedes any previous agreements I have entered into with Elsevier Inc. in relation to the Article from the date hereof.

AUTHOR RIGHTS FOR SCHOLARLY PURPOSES

I understand that I retain or am hereby granted (without the need to obtain further permission) the Author Rights (see description below), and that no rights in patents, trademarks or other intellectual property rights are transferred to Elsevier Inc..

The Author Rights include the right to use the Preprint, Accepted Manuscript and the Published Journal Article for Personal Use, Internal Institutional Use and for Scholarly Sharing.

In the case of the Accepted Manuscript and the Published Journal Article the Author Rights exclude Commercial Use (unless expressly

agreed in writing by Elsevier Inc.), other than use by the author in a subsequent compilation of the author's works or to extend the Article to book length form or re-use by the author of portions or excerpts in other works (with full acknowledgment of the original publication of the Article).

AUTHOR REPRESENTATIONS / ETHICS AND DISCLOSURE / SANCTIONS

I affirm the Author Representations noted below, and confirm that I have reviewed and complied with the relevant Instructions to Authors, Ethics in Publishing policy, Declarations of Interest disclosure and information for authors from countries affected by sanctions (Iran, Cuba, Sudan, Burma, Syria, or Crimea). Please note that some journals may require that all co-authors sign and submit Declarations of Interest disclosure forms. I am also aware of the publisher's policies with respect to retractions and withdrawal (https://www.elsevier.com/about/our-business/policies/article-withdrawal).
For further information see the publishing ethics page at https://www.elsevier.com/about/our-business/policies/publishing-ethics and the journal home page. For further information on sanctions, see https://www.elsevier.com/about/our-business/policies/trade-sanctions

Author representations

- The Article I have submitted to the journal for review is original, has been written by the stated authors and has not been previou
- The Article was not submitted for review to another journal while under review by this journal and will not be submitted to any oth
- The Article and the Supplemental Materials do not infringe any copyright, violate any other intellectual property, privacy or other r entity, or contain any libellous or other unlawful matter.
- I have obtained written permission from copyright owners for any excerpts from copyrighted works that are included and have cre the Article or the Supplemental Materials.
- Except as expressly set out in this Journal Publishing Agreement, the Article is not subject to any prior rights or licenses and, if my authors' institution has a policy that might restrict my ability to grant the rights required by this Journal Publishing Agreement (tal Author Rights permitted hereunder, including Internal Institutional Use), a written waiver of that policy has been obtained.
- If I and/or any of my co-authors reside in Iran, Cuba, Sudan, Burma, Syria, or Crimea, the Article has been prepared in a persona capacity and not as an official representative or otherwise on behalf of the relevant government or institution.
- If I am using any personal details or images of patients, research subjects or other individuals, I have obtained all consents requir and complied with the publisher's policies relating to the use of such images or personal information. See https://www.elsevier.co business/policies/patient-consent for further information.
- Any software contained in the Supplemental Materials is free from viruses, contaminants or worms.
- If the Article or any of the Supplemental Materials were prepared jointly with other authors, I have informed the co-author(s) of th Publishing Agreement and that I am signing on their behalf as their agent, and I am authorized to do so.

GOVERNING LAW AND JURISDICTION

This Agreement will be governed by and construed in accordance with the laws of the country or state of Elsevier Inc. ("the Governing State"), without regard to conflict of law principles, and the parties irrevocably consent to the exclusive jurisdiction of the courts of the Governing State.

For information on the publisher's copyright and access policies, please see http://www.elsevier.com/copyright.
For more information about the definitions relating to this agreement click here.

☑ **I have read and agree to the terms of the Journal Publishing Agreement.**

8th March 2018 T-copyright-v22/2017

CITED LITERATURE

1. Cole, S., Friedland, S., and Reyzin, L.: A simple spectral algorithm for recovering planted partitions. Special Matrices, 5(1):139–157, 2017.

2. Cole, S.: Recovering nonuniform planted partitions via iterated projection. Linear Algebra and its Applications, 2018.

3. Fortunato, S.: Community detection in graphs. Physics reports, 486(3):75–174, 2010.

4. Boppana, R. B.: Eigenvalues and graph bisection: An average-case analysis (extended abstract). In 28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987, pages 280–285, 1987.

5. Bui, T. N., Chaudhuri, S., Leighton, F. T., and Sipser, M.: Graph bisection algorithms with good average case behavior. Combinatorica, 7(2):171–191, Jun 1987.

6. Holland, P. W., Laskey, K. B., and Leinhardt, S.: Stochastic blockmodels: First steps. Social networks, 5(2):109–137, 1983.

7. Ames, B. P. W.: Guaranteed clustering and biclustering via semidefinite programming. Mathematical Programming, 147(1-2):429–465, 2014.

8. Chen, Y., Sanghavi, S., and Xu, H.: Improved graph clustering. Information Theory, IEEE Transactions on, 60(10):6440–6455, October 2014.

9. Oymak, S. and Hassibi, B.: Finding dense clusters via "low rank + sparse" decomposition. arXiv preprint arXiv:1104.5186, 2011.

10. Coja-Oghlan, A.: Graph partitioning via adaptive spectral techniques. Combinatorics, Probability and Computing, 19(02):227–284, 2010.

11. McSherry, F.: Spectral partitioning of random graphs. In FOCS, pages 529–537, 2001.

12. Vu, V.: A simple SVD algorithm for finding hidden partitions. arXiv preprint arXiv:1404.3918, 2014.

13. Giesen, J. and Mitsche, D.: Reconstructing many partitions using spectral techniques. In Proceedings of the 15th International Symposium on Fundamentals of Computation Theory, 2005.

14. Alon, N., Krivelevich, M., and Sudakov, B.: Finding a large hidden clique in a random graph. Random Struct. Algorithms, 13(3-4):457–466, 1998.

15. Jerrum, M.: Large cliques elude the metropolis process. Random Struct. Algorithms, 3(4):347–360, 1992.

16. Kučera, L.: Expected complexity of graph partitioning problems. Discrete Applied Mathematics, 57(2-3):193–212, 1995.

17. Feige, U. and Krauthgamer, R.: Finding and certifying a large hidden clique in a semirandom graph. Random Struct. Algorithms, 16(2):195–208, 2000.

18. Feldman, V., Grigorescu, E., Reyzin, L., Vempala, S., and Xiao, Y.: Statistical algorithms and a lower bound for detecting planted cliques. In Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013, pages 655–664, 2013.

19. Erdős, P. and Rényi, A.: On random graphs I. Publicationes Mathematicae (Debrecen), 6:290–297, 1959 1959.

20. Dyer, M. E. and Frieze, A. M.: The solution of some random np-hard problems in polynomial expected time. J. Algorithms, 10(4):451–489, December 1989.

21. Chen, Y. and Xu, J.: Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. arXiv preprint arXiv:1402.1267, 2014.

22. Hogg, R., McKean, J., and Craig, A.: Introduction to Mathematical Statistics. Pearson, 2013.

23. Boyd, S. and Vandenberghe, L.: Convex Optimization. Berichte über verteilte messysteme. Cambridge University Press, 2004.

24. Horn, R. A. and Johnson, C. R.: Matrix Analysis. Cambridge University Press, 2012.

25. Bondy, A. and Murty, U.: Graph Theory. Graduate Texts in Mathematics. Springer London, 2011.

26. Resnick, S.: A Probability Path. A Probability Path. Birkhäuser Boston, 2003.

27. Alon, N. and Spencer, J. H.: The probabilistic method. John Wiley & Sons, 2004.

28. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American statistical association, 58(301):13–30, 1963.

29. Chung, F.: Spectral Graph Theory. Number no. 92 in CBMS Regional Conference Series. Conference Board of the Mathematical Sciences, 1997.

30. Bollobás, B.: Random Graphs. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2 edition, 2001.

31. Friedland, S.: Matrices. World Scientific, 2015.

32. Stein, E. and Shakarchi, R.: Complex Analysis. Princeton lectures in analysis. Princeton University Press, 2010.

33. Füredi, Z. and Komlós, J.: The eigenvalues of random symmetric matrices. Combinatorica, 1(3):233–241, 1981.

34. Vu, V.: Spectral norm of random matrices. Combinatorica, 27(6):721–736, 2007.

35. Alon, N., Krivelevich, M., and Vu, V. H.: On the concentration of eigenvalues of random symmetric matrices. Israel Journal of Mathematics, 131(1):259–267, 2002.

36. Golub, G. H. and Van Loan, C. F.: Matrix Computations (3rd Ed.). Baltimore, MD, USA, Johns Hopkins University Press, 1996.

37. Gu, M.: Subspace iteration randomization and singular value problems. SIAM Journal on Scientific Computing, 37(3):A1139–A1173, 2015.

38. Halko, N., Martinsson, P. G., and Tropp, J. A.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev., 53(2):217–288, May 2011.

39. Kishore Kumar, N. and Schneider, J.: Literature survey on low rank approximation of matrices. Linear and Multilinear Algebra, pages 1–33, 2016.

40. Nguyen, N. H., Do, T. T., and Tran, T. D.: A fast and efficient algorithm for low-rank approximation of a matrix. In Proceedings of the forty-first annual ACM symposium on Theory of computing, pages 215–224. ACM, 2009.

41. Coppersmith, D. and Winograd, S.: Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation, 9(3):251 – 280, 1990.

42. Le Gall, F.: Powers of tensors and fast matrix multiplication. In Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation, ISSAC '14, pages 296–303, New York, NY, USA, 2014. ACM.

43. Ding, J. and Zhou, A.: Eigenvalues of rank-one updated matrices with some applications. Applied Mathematics Letters, 20(12):1223 – 1226, 2007.

44. Gu, M. and Eisenstat, S. C.: A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem. SIAM journal on Matrix Analysis and Applications, 15(4):1266–1276, 1994.

45. Mehl, C., Mehrmann, V., Ran, A. C., and Rodman, L.: Eigenvalue perturbation theory of classes of structured matrices under generic structured rank one perturbations. Linear Algebra and its Applications, 435(3):687 – 716, 2011. Special Issue: Dedication to Pete Stewart on the occasion of his 70th birthday.

46. Lovász, L.: Large networks and graph limits, volume 60. 2012.

47. Kenyon, R., Radin, C., Ren, K., and Sadun, L.: Multipodal structure and phase transitions in large constrained graphs. Journal of Statistical Physics, 168(2):233–258, 2017.

48. Kenyon, R., Radin, C., Ren, K., and Sadun, L.: The phases of large networks with edge and triangle constraints. arXiv preprint arXiv:1701.04444, 2017.

49. Chatterjee, S.: An introduction to large deviations for random graphs. Bulletin of the American Mathematical Society, 53(4):617–642, 2016.

50. Krivelevich, M. and Patkós, B.: Equitable coloring of random graphs. Random Structures and Algorithms, 35(1):83–99, 2009.

51. Bollobás, B.: The chromatic number of random graphs. Combinatorica, 8(1):49–55, 1988.

52. Bollobás, B. and Erdős, P.: Cliques in random graphs. In Mathematical Proceedings of the Cambridge Philosophical Society, volume 80, pages 419–427. Cambridge University Press, 1976.

53. Abbe, E., Bandeira, A. S., and Hall, G.: Exact recovery in the stochastic block model. IEEE Transactions on Information Theory, 62(1):471–487, 2016.

54. Banks, J., Moore, C., Neeman, J., and Netrapalli, P.: Information-theoretic thresholds for community detection in sparse networks. In Conference on Learning Theory, pages 383–416, 2016.

55. Garey, M. R. and Johnson, D. S.: Computers and intractability: a guide to the theory of NP-completeness. Freeman, 1979.

56. Levin, L. A.: Average case complete problems. SIAM Journal on Computing, 15(1):285–286, 1986.

57. Arora, S. and Barak, B.: Computational complexity: a modern approach. Cambridge University Press, 2009.

58. Feldman, V., Grigorescu, E., Reyzin, L., and Vempala, S.: The complexity of statistical algorithms. CoRR, abs/1201.1214, 2012.

59. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C.: Introduction to Algorithms, Third Edition. The MIT Press, 3rd edition, 2009.

Dept. of Mathematics,
Statistics, and Computer Science
Univ. of Illinois at Chicago
851 S. Morgan St. (MC 249)
Chicago, IL 60607
☎ (773) 818-4124
✉ scole3@uic.edu
www.homepages.math.uic.edu/~scole3

Sam Cole

Curriculum Vitae

Education

2013–present **Ph.D., mathematics (in progress)**, *Univ. of Illinois at Chicago*, Dept. of Mathematics, Statistics, and Computer Science.
Advisor: Shmuel Friedland
Research interests: random graphs, linear algebra methods in computer science and combinatorics

2010–2011 **M.S., mathematics**, *Univ. of Illinois at Chicago*, Dept. of Mathematics, Statistics, and Computer Science.
Concentration: mathematical computer science

2006-2009 **B.A., mathematics**, *Oberlin College*.
Minor: computer science

Papers

Sam Cole. Recovering nonuniform planted partitions via iterated projection. *Linear Algebra and its Applications*, 2018.

Sam Cole, Shmuel Friedland, and Lev Reyzin. A simple spectral algorithm for recovering planted partitions. *Special Matrices*, 5(1):139–157, 2017.

Sam Cole and Jamie Quadri. Selfish independent set. In *Midstates Conference for Undergraduate Research in Computer Science and Mathematics*, 2009.

Talks, conferences, & workshops

2017 **Meeting of the International Linear Algebra Society**, *Iowa State Univ.*, Mini-symposium on random matrix theory.
Invited talk: "A simple algorithm for spectral clustering of random graphs"

2017 **Gene Golub SIAM Summer School**, *Akademie Berlin-Schmöckwitz*.

2017 **SIU Mathematics Conference**, *Southern Illinois Univ. Carbondale*, Special session on matrix theory, computation, and applications.
Invited talk: "A simple spectral algorithm for recovering planted partitions"

2017 **Numerical Analysis Seminar**, *Univ. of Texas at Austin*.
Invited talk: "A simple algorithm for spectral clustering of random graphs"

2017 **Scientific and Statistical Computing Seminar**, *Univ. of Chicago*.
Invited talk: "A simple algorithm for spectral clustering of random graphs"

2016 **Graduate Research Workshop in Combinatorics**, *Univ. of Wyoming*.

2016 **Combinatorial and Additive Number Theory**, *CUNY Graduate Center*.
Contributed talk: "Planted partitions in random graphs"

2016 **Optimization and Parsimonious Modeling**, *Institute for Mathematics and its Applications, Univ. of Minnesota*.
Poster: "A Simple Spectral Algorithm for Recovering Planted Partitions" (presented by Shmuel Friedland)

Teaching

2010–2011; 2013–present **Teaching assistant**, *Univ. of Illinois at Chicago*.
Selected courses:
- Precalculus
- Calculus
- Introduction to programming in Python
- Data structures in C++

2016 **Graduate mentor**, *Univ. of Illinois at Chicago*, Mathematical Computing Laboratory.
Project: random geometric graphs

2015–2016 **Lecturer**, *Univ. of Illinois at Chicago*.
Courses:
- Multivariable calculus
- Mathematical Computing Laboratory summer high school workshop in graph theory

2008–2009 **Grader/lab assistant**, *Oberlin College*.
Courses:
- Linear algebra (grader)
- Introduction to computer science I & II (grader and lab assistant)

Awards

2011 **Graduate Student Teaching Award**, *Univ. of Illinois at Chicago*, Dept. of Mathematics, Statistics, and Computer Science.

Industry experience

2012 **Software engineer**, *Datalogics, Inc.*

Programming languages & software

C, C++, Git, HTML/CSS, Java, JavaScript, MATLAB, Python, SciPy