

Sam Cole*, Shmuel Friedland, and Lev Reyzin

A simple spectral algorithm for recovering planted partitions

<https://doi.org/10.1515/spma-2017-0013>

Received November 24, 2016; accepted July 14, 2017

Abstract: In this paper, we consider the *planted partition model*, in which $n = ks$ vertices of a random graph are partitioned into k “clusters,” each of size s . Edges between vertices in the same cluster and different clusters are included with constant probability p and q , respectively (where $0 \leq q < p \leq 1$). We give an efficient algorithm that, with high probability, recovers the clusters as long as the cluster sizes are at least $\Omega(\sqrt{n})$. Informally, our algorithm constructs the projection operator onto the dominant k -dimensional eigenspace of the graph’s adjacency matrix and uses it to recover one cluster at a time. To our knowledge, our algorithm is the first purely spectral algorithm which runs in polynomial time and works even when $s = \Theta(\sqrt{n})$, though there have been several non-spectral algorithms which accomplish this. Our algorithm is also among the simplest of these spectral algorithms, and its proof of correctness illustrates the usefulness of the Cauchy integral formula in this domain.

1 Introduction and previous work

In the Erdős-Rényi random graph model [13], graphs $G(n, p)$ on n vertices are generated by including each of the possible $\binom{n}{2}$ edges independently at random with probability $0 < p < 1$. A classical conjecture of Karp [24] states that there is no efficient algorithm for finding cliques of size $(1 + \epsilon) \log_{1/p} n$, though cliques of size at least $2 \log_{1/p} n$ will almost surely exist [7].

Jerrum [23] and Kučera [26] introduced a potentially easier variant called the *planted clique* problem. In this model, one starts with a random graph, but additionally, edges are added deterministically to an unknown set of s vertices (known as the “plant”) to make them form a clique. The goal then is to determine a.s. exactly which vertices belong to the planted clique, which should be easier when s becomes large.

When $s = \Omega(\sqrt{n \log n})$, the clique can be found by simply taking the s vertices with the largest degrees [26]. This bound was improved using spectral methods to $\Omega(\sqrt{n})$ by Alon et al. [2] and then others [5, 10, 12, 14, 15, 28]. These methods also handle a generalization of this problem in which edges within the plant are added merely with higher probability rather than deterministically.

A more general version of the problem is to allow for planting multiple disjoint cliques, sometimes called a *planted clustering*. In the most basic version, known as the *planted partition model* (also called the stochastic block model), n nodes are partitioned into k disjoint clusters of size $s = n/k$, which are “planted” in a random graph. Two nodes u and v get an edge with probability p if they are in the same cluster and with probability q if they reside in different clusters (with $p > q$ constant). The goal is now to recover the unknown clustering from the random graph generated according to the model, i.e., to determine exactly the vertices in each cluster a.s.

As in the planted clique case, a relatively simple algorithm can recover the clustering when the clique sizes are $\Omega(\sqrt{n \log n})$ —in this case pairs of vertices with the most common neighbors can be placed in the

*Corresponding Author: Sam Cole: Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, Chicago, Illinois 60607-7045, USA, E-mail: scole3@uic.edu

Shmuel Friedland, Lev Reyzin: Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, Chicago, Illinois 60607-7045, USA, E-mail: {friedlan, lreyzin}@uic.edu

same cluster [9]. However, when the cluster sizes are only required to be $\Omega(\sqrt{n})$, the problem, as in the planted clique case, becomes more difficult because a simple application of the Azuma-Hoeffding inequality no longer suffices.

Our main result is that this can, in fact, be done when the clusters are size $s = \Omega(\sqrt{n})$:

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 $c = O(1/(p - q)^2)$.*

Note that in this paper we consider only the setting in which p and q are constant and all clusters are the same size $s = n/k$.

Our algorithm is, to our knowledge, the first purely spectral algorithm which runs in polynomial time and recovers the planted partition a.s. even when all clusters are size $\Theta(\sqrt{n})$, though there have been several non-spectral algorithms which work in this setting [4, 8, 30]. In particular, the well-known spectral algorithms [28, 33] 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). On the other hand, Giesen and Mitsche’s algorithm [19] works when all clusters are size $\Theta(\sqrt{n})$ but has running time exponential in k . See Appendix A for comparison with previous work

Efficient algorithms for planted clustering typically rely on either convex optimization [4, 8, 30] or spectral techniques [19, 28, 33]. The latter, including ours, often involve looking at the projection operator onto the vector space spanned by the k eigenvectors corresponding to the k largest eigenvalues of the adjacency matrix \hat{A} of the randomly generated graph \hat{G} and showing that it is “not too far” from the projection operator of the expectation matrix $E[\hat{A}]$ onto its own k largest eigenvalues. Our algorithm is among the simplest of these spectral algorithms: we don’t randomly partition the vertices beforehand, and hence there is no messy “cleanup” step at the end.

A natural approach for identifying all the clusters would be to identify a single cluster, remove it, and recurse on the remaining vertices. This is hard to make work because the randomness of the instance \hat{G} is “used up” in the first iteration, and then subsequent iterations cannot be handled independently of the first. Existing spectral approaches bypass these difficulties by randomly splitting the input graph into parts, thus forcing independence in the randomness on the parts [19, 28, 33]. This partitioning trick works at the cost of complicating the algorithm. We, however, are able to make the natural recursive approach work by “preprocessing the randomness”: we show that certain (exponentially many) events *all* occur simultaneously with high probability, and as long as they all occur our algorithm *definitely* works.

$\Omega(\sqrt{n})$ cluster size is generally accepted to be the barrier for efficient algorithms for “planted” problems. Evidence for the difficulty of beating the \sqrt{n} barrier dates back to Jerrum [23], who showed a specific Markov chain approach will fail to find smaller cliques. Feige and Krauthgamer [14] showed that Lovász-Schrijver SDP relaxations run into the same barrier, while Feldman et al. [16] show that all “statistical algorithms” also provably fail to efficiently find smaller cliques in a distributional version of the planted clique problem. Recently, Ailon et al. [1] were able to recover planted clusterings in which some of the cluster sizes are $o(\sqrt{n})$, but their algorithm’s success depends on the simultaneous presence of clusters of size $\Omega(\sqrt{n} \log^2 n)$.

1.1 Outline

In Section 2 we formally define the planted partition model. In Section 3 we present our algorithm for identifying the clusters, and we briefly discuss its running time in Section 3.1. We prove its correctness in Section 7. Sections 4-6 are dedicated to developing the linear algebra tools necessary for the proof: in Section 4 we introduce tools from random matrix theory which we use in Section 5 to characterize the eigenvalues of the (unknown) expectation matrix A and the randomly generated adjacency matrix \hat{A} . This, in turn, allows us to bound the difference of their projections in Section 6. Showing that the projection operators of A and \hat{A} are “close” is the key ingredient in our proof.

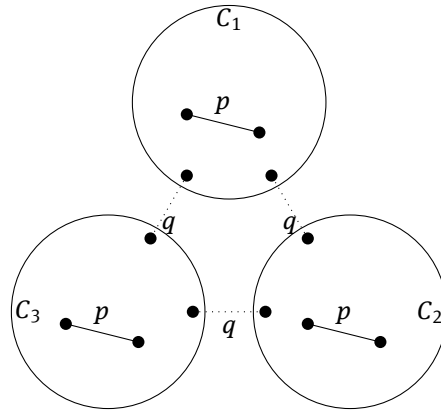


Figure 1: An illustration of the planted partition model. Edges between two vertices in the same cluster are added with probability p , while edges between two vertices in different clusters are added with probability q .

2 The planted partition problem

We now formally define the planted partition problem.

Definition 2 (Planted partition model). Let $\mathcal{C} = \{C_1, \dots, C_k\}$ be a partition of the set $[n] := \{1, \dots, n\}$ into k sets of size $s = n/k$, called clusters (assume $s|n$). For constants $0 \leq q < p \leq 1$, we define the planted partition model $\mathcal{G}(n, \mathcal{C}, p, q)$ to be the probability space of graphs with vertex set $[n]$, with edges ij (for $i \neq j$) included independently with probability p if i and j 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 $\mathcal{G}(n, p)$ [13], and the case $k = n$ gives $\mathcal{G}(n, q)$.

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

- $\hat{G} = ([n], \hat{E})$ – 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.
- $\hat{A} = (\hat{a}_{ij})_{i,j=1}^n \in \{0, 1\}^{n \times n}$ – the adjacency matrix of \hat{G} .
- $E[\hat{A}] := (E[\hat{a}_{ij}])_{i,j=1}^n$ – the entrywise expectation of \hat{A} .
- $A = (a_{ij})_{i,j=1}^n := E[\hat{A}] + pI_n$ – the expectation of the adjacency matrix \hat{G} with ps added to the diagonal (to make it a rank k matrix and simplify the proofs).

Problem 3 (Planted partition). Identify (or “recover”) the unknown partition C_1, \dots, C_k (up to a permutation of $[k]$) given only \hat{G} , or, equivalently, reproduce A given only \hat{A} .

In this paper we give an algorithm to recover the clusters which is based on the k largest eigenvalues of \hat{A} and the corresponding eigenspaces.

2.1 Graph and matrix notation

We will use the following notation throughout this paper:

- $N_G(v)$ – neighborhood of vertex v in a graph G . We will omit the subscript G when the meaning is clear.
- $G[S]$ – the induced subgraph of G on $S \subseteq V(G)$.
- $A[S]$ – the principal submatrix of A with row and column indices restricted to S .
- $\lambda_i(A)$ – the i th largest eigenvalue of a symmetric matrix A (recall that symmetric matrices have real eigenvalues).

- $\lambda_i(G)$ – the i th largest eigenvalue of G 's adjacency matrix.
- $P_k(A)$ – orthogonal projection operator onto the subspace of \mathbb{R}^n spanned by eigenvectors corresponding to the largest k eigenvalues of an $n \times n$ symmetric matrix A , represented in the standard basis for \mathbb{R}^n .
- $\|\cdot\|_2$ – the ℓ_2 - (“spectral”) norm of a vector or matrix.
- $\|\cdot\|_F$ – the Frobenius norm of a matrix.
- I_n – the $n \times n$ identity matrix.
- J_n – the $n \times n$ 1s matrix.
- $\mathbf{1}_S$ – 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. $\mathbf{1}_{[n]}$.
- $E[X]$ – the expectation of a random variable X . If X is matrix or vector valued, then the expectation is taken entrywise.
- a.s. – almost surely, i.e. with probability $1 - o(1)$ as $n \rightarrow \infty$.

3 The cluster identification algorithm

The main result of this paper is that Algorithm 1 below recovers clusters of size $c\sqrt{n}$:

Theorem 4. *For sufficiently large n with probability $\geq 1 - 2^{-\Omega(\sqrt{n})}$, Algorithm 1 correctly recovers planted partitions in which all clusters are size $s \geq c\sqrt{n}$, where $c := \max\left\{\frac{88}{p-q}, \frac{72}{(p-q)^2}\right\}$.*

Algorithm 1:

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 $\hat{p}_{i_1 j} \geq \dots \geq \hat{p}_{i_{s-1} j}$ be the entries other than \hat{p}_{jj} in nonincreasing order. Let $W_j := \{j, i_1, \dots, i_{s-1}\}$, 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 1 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 1 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 by a subset of the clusters, and that as long as they all hold Algorithm 1 will *definitely* succeed. See Section 7.

3.1 Running time

Let us analyze the running time of one iteration of Algorithm 1. 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 $O(n^2k)$ [20, 21]. Alternatively, one may utilize one of several recent randomized algorithms [21, 22, 25, 29] which allow this to be done faster, e.g. in time $O(n^2 \log k)$ [22].
- Step 4 can be done naively 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 7.1. 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 [11, 27], 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 1 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 $O(n^3)$.

4 Eigenvalues of random symmetric matrices

In Section 5 we will show that the eigenvalues of the random matrix \hat{A} are close to those of its expectation matrix A . To do so, we will need the following well-known result of Füredi and Komlós about the concentration of eigenvalues of random symmetric matrices [18, Theorem 2]:

Theorem 5. *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 $K, \sigma > 0$ so that the following conditions hold independent of n :*

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

Then

$$\max_{i=1}^n |\lambda_i(X)| \leq 2\sigma\sqrt{n} + 50Kn^{\frac{1}{3}} \log n \quad (4.1)$$

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

Note that the original paper by Füredi and Komlós assumes that $E[x_{ij}^2] = \sigma^2$ for all i, j , which in turn makes the bound (4.1) tight. However, if all we need is the upper bound in (4.1), as is the case in this paper, then the proof in [18] goes through with $E[x_{ij}^2] \leq \sigma^2$. (Actually, it was pointed out by Vu [32] that the proof in [18] contains a minor mistake, so we follow the corrected proof in [32].)

Unfortunately, the n^{-10} failure probability isn't small enough to guarantee our algorithm's success in every iteration, as we will need to apply Theorem 5 simultaneously to $2^{O(\sqrt{n})}$ submatrices of \hat{A} (see Section 7.3); however, we may combine it with the following concentration result to get exponentially small failure probability [3, Theorem 1]:

Theorem 6. *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 5 and 6, we get the following:

Theorem 7. *Let X be defined as in Theorem 5. Then*

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

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

Proof. By Theorem 5,

$$\Pr \left[\max_{i=1}^n |\lambda_i(X)| \geq 2\sigma\sqrt{n} + 50Kn^{\frac{1}{3}} \log n \right] < \frac{1}{n^{10}}. \quad (4.2)$$

For $n \geq n_0$, we have

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

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

$$|\lambda| \leq 2(\sigma + 0.1K)\sqrt{n}. \quad (4.3)$$

Indeed,

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

by (4.2). Now consider the random matrix $-X$. It satisfies the assumptions of Theorem 5. Therefore we have

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

As $\lambda_n(-X) = -\lambda_1(X)$, this is the same as $\Pr[\lambda_1(X) \leq -2(\sigma + 0.1K)\sqrt{n}]$. Hence (4.3) follows by definition of median.

We now ready to apply Theorem 6. Let $Y = \frac{1}{K}X$. So now each entry of Y is in $[-1, 1]$. Clearly the median of the random variable $\lambda_1(Y)$ is λ/K . By (4.3) and Theorem 6

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

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

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

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

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

as claimed. □

Note that there have been some recent results which give tight bounds on the spectra of more general random matrices [6], but the above are sufficient for our purposes.

Finally, we will need the following fact from linear algebra:

Proposition 8 (Weyl's inequalities). *Let $X, Y \in \mathbb{R}^{n \times n}$ be symmetric matrices. Then*

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

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

See, e.g., [17, Theorem 4.4.6] for proof.

5 Eigenvalues of A and \hat{A}

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 $C_1 = \{1, \dots, s\}, C_2 = \{s + 1, \dots, 2s\}, \dots, C_k = \{n - s + 1, \dots, n\}$. Then the expectation matrix A looks like:

$$A = \begin{pmatrix} p & \dots & p & q & \dots & q & & q & \dots & q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ p & \dots & p & q & \dots & q & & q & \dots & q \\ \hline q & \dots & q & p & \dots & p & & q & \dots & q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ q & \dots & q & p & \dots & p & & q & \dots & q \\ \hline & & \vdots & & & \vdots & \ddots & & & \vdots \\ \hline q & \dots & q & q & \dots & q & & p & \dots & p \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ q & \dots & q & q & \dots & q & & p & \dots & p \end{pmatrix} = qI_n + (p - q) \text{diag}(J_s, \dots, J_s),$$

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 9. *The eigenvalues of A are*

$$\begin{aligned} \lambda_1(A) &= (p - q)s + qn, \\ \lambda_i(A) &= (p - q)s \text{ for } i = 2, \dots, k, \\ \lambda_i(A) &= 0 \text{ for } i = k + 1, \dots, n. \end{aligned}$$

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 [17, Corollary 4.11.13]).

Lemma 10. *For sufficiently large n ,*

$$\|\hat{A} - A\|_2 \leq 8\sqrt{n} \tag{5.1}$$

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

Proof. Set $X = (x_{ij}) = \hat{A} - E[\hat{A}]$. 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 7, with

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

Thus,

$$\|X\|_2 = \max_{i=1}^n |\lambda_i(X)| \leq 2(\sigma + 3K)\sqrt{n} \leq 7\sqrt{n} \tag{5.2}$$

with probability $\geq 1 - e^{-n}$ by Theorem 7.

Observe that

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

\Rightarrow

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

From (5.2) 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:

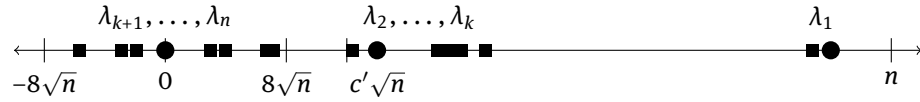


Figure 2: The distribution of eigenvalues of A (●) and \hat{A} (■).

Lemma 11. Assume \hat{A} satisfies (5.1) 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

$$c' := (p - q)c - 8. \tag{5.3}$$

Proof. Applying Weyl’s inequalities (Proposition 8) to Lemma 10 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 $i = 1, \dots, n$. Thus, by Lemma 9 we get

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

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 $\lambda_1(X) \leq \max_i \sum_j |x_{ij}|$. \square

Lemma 11 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

$$c > \frac{16}{p - q}. \tag{5.4}$$

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 $\lambda_2(\hat{A}), \dots, \lambda_k(\hat{A}) \leq (p - q)s + 8\sqrt{n} = O(s)$, but this information will not be needed hence. Figure 2 illustrates the distribution of eigenvalues of A and \hat{A} .

6 Deviations between the projectors $P_k(\hat{A})$ and $P_k(A)$

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 12.

$$P_k(A) = \frac{1}{s} \sum_{i=1}^k \mathbf{1}_{C_i} \mathbf{1}_{C_i}^\top = \frac{1}{s} H, \tag{6.1}$$

where $H \in \{0, 1\}^n$ is the “true” cluster matrix whose (i, j) th entry is 1 if and only if i and j are in the same cluster.

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

$$P_k(A) = P_{\mathbf{U}} = \sum_{i=1}^k \mathbf{u}_i \mathbf{u}_i^\top = \frac{1}{s} \sum_{i=1}^k \mathbf{1}_{C_i} \mathbf{1}_{C_i}^\top. \tag{6.2} \quad \square$$

If we assume $C_1 = \{1, \dots, s\}$, $C_2 = \{s + 1, \dots, 2s\}$, ..., $C_k = \{n - s + 1, \dots, n\}$ as in Section 5, then $P_k(A)$ looks like:

$$P_k(A) = \frac{1}{s} \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ 1 & \dots & 1 & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & \dots & 1 & \dots & 0 & \dots & 0 \\ \vdots & & \vdots & & & \vdots & & & & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & \dots & 1 & \dots & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & \dots & 1 & \dots & 1 \end{pmatrix} = \frac{1}{s} \text{diag}(J_s, \dots, J_s)$$

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, \dots, C_k . The central idea behind Algorithm 1 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.

6.1 The Cauchy integral formula for projections

To prove such bounds on $\|P_k(A) - P_k(\hat{A})\|_2$ and $\|P_k(A) - P_k(\hat{A})\|_F$ we will employ the Cauchy integral formula. Similar applications of the Cauchy integral formula are studied in matrix perturbation theory [20, 31] and could be adapted to obtain our bound on $\|P_k(A) - P_k(\hat{A})\|_2$, but we include the full proof for the sake of exposition.

Recall that an analytic function $f : \mathbb{C} \rightarrow \mathbb{C}$ can be extended to a function of matrices via its Taylor series [17]:

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

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 [17, Theorem 3.4.2]:

Theorem 13. *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 $\phi : \mathbb{C} \rightarrow \mathbb{C}$ analytic on Ω*

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

We get the following as a corollary [17, Problem 3.4.10]:

Theorem 14. *Let $B \in \mathbb{R}^{n \times n}$ be a symmetric matrix. Let $\gamma \subset \mathbb{C}$ be a simple, closed curve which is the boundary of an open set D such that $\lambda_1(B), \dots, \lambda_k(B) \in D$ and $\lambda_{k+1}(B), \dots, \lambda_n(B) \notin D \cup \gamma$. Then*

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

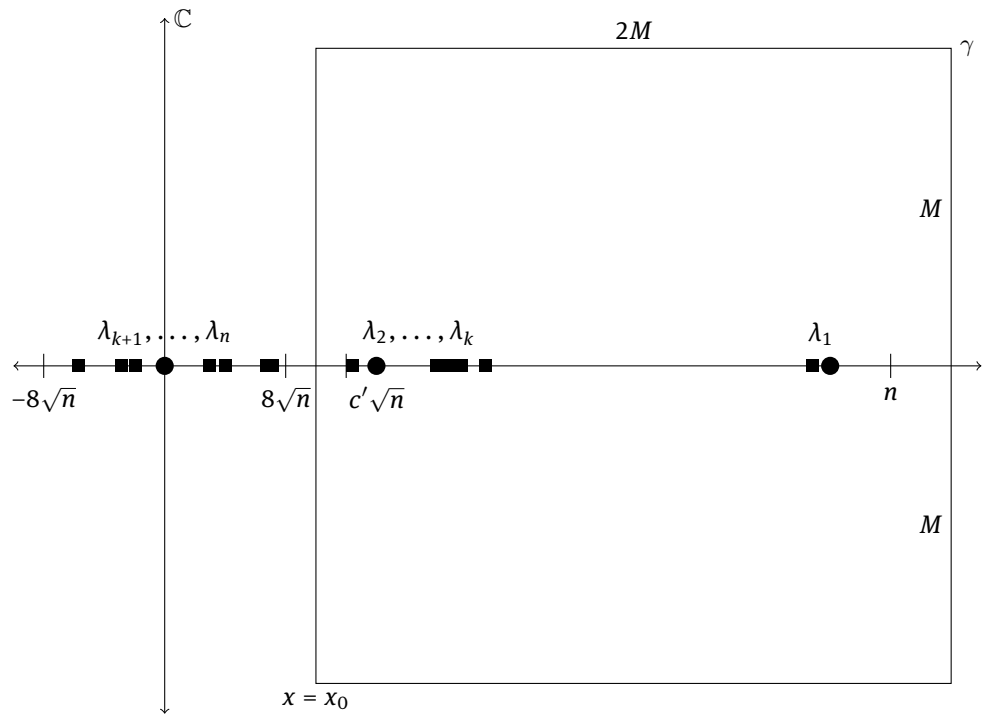


Figure 3: The largest k eigenvalues of both A (●) and \hat{A} (■) are in the interior of γ , while the remaining $n - k$ eigenvalues are in the exterior.

6.2 A bound on $\|P_k(\hat{A}) - P_k(A)\|_2$

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

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

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

Lemma 15. *Assume \hat{A} satisfies (5.1) and $s \geq c\sqrt{n}$. Then*

$$\|P_k(\hat{A}) - P_k(A)\|_2 \leq \epsilon \tag{6.2}$$

if c is sufficiently large.

Proof. Define γ to be a square in the complex plane with the length $2M \gg m$. Its sides are parallel to the x - and y -axes. The center of of the square is on the x -axis. The left and right sides of the square are on the lines $x = x_0 := \frac{(c'+8)\sqrt{n}}{2}$ and $x = x_0 + 2M$, respectively, where c' is defined in (5.3). The upper and lower sides of the square are on the lines $y = \pm M$. Note that by Lemma 11 the interior of γ contains the k largest eigenvalues of A and \hat{A} and the exterior of γ contains the other $n - k$ eigenvalues of A and \hat{A} (see Figure 3). To get our estimate (6.2) we will let $M \rightarrow \infty$.

Applying Theorem 14,

$$P_k(\hat{A}) = \frac{1}{2\pi i} \int_{\gamma} (zI_n - \hat{A})^{-1} dz,$$

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

Hence

$$\begin{aligned} P_k(\hat{A}) - P_k(A) &= \frac{1}{2\pi i} \int_{\gamma} (zI_n - \hat{A})^{-1} ((zI_n - A) - (zI_n - \hat{A}))(zI_n - A)^{-1} dz \\ &= \frac{1}{2\pi i} \int_{\gamma} (zI_n - \hat{A})^{-1} (\hat{A} - A)(zI_n - A)^{-1} dz, \end{aligned}$$

and so we get

$$\begin{aligned} \|P_k(\hat{A}) - P_k(A)\|_2 &\leq \frac{1}{2\pi} \int_{\gamma} \|(zI_n - \hat{A})^{-1} (\hat{A} - A)(zI_n - A)^{-1}\|_2 |dz| \\ &\leq \frac{1}{2\pi} \int_{\gamma} \|(zI_n - \hat{A})^{-1}\|_2 \|\hat{A} - A\|_2 \|(zI_n - A)^{-1}\|_2 |dz|. \end{aligned} \tag{6.3}$$

Observe that for each $z \in \mathbb{C}$ the matrices $zI_n - \hat{A}$, $zI_n - A$ are normal. Hence

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

Let us first estimate the contribution to the integral (6.3) on the left side of γ . Let $z = x_0 + yi$, $y \in \mathbb{R}$. That is, z lies on the line $x = x_0$. Therefore, by Lemma 11

$$|z - \lambda_j(A)|, |z - \lambda_j(\hat{A})| \geq \sqrt{\left(\frac{(c' - 8)\sqrt{n}}{2}\right)^2 + y^2} \text{ for } z = x_0 + yi.$$

Also recall from (5.1) that $\|\hat{A} - A\|_2 \leq 8\sqrt{n}$. Hence for $z = x_0 + yi$ one has the estimate:

$$\begin{aligned} &\frac{1}{2\pi} \int_{-M}^M \|(zI_n - \hat{A})^{-1}\|_2 \|\hat{A} - A\|_2 \|(zI_n - A)^{-1}\|_2 |dz| \\ &< \\ &\frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{8\sqrt{n}}{\left(\frac{(c' - 8)\sqrt{n}}{2}\right)^2 + y^2} dy = \frac{8}{\pi(c' - 8)} \int_{-\infty}^{\infty} \frac{du}{1 + u^2} = \frac{8}{(c' - 8)}. \end{aligned}$$

Next we estimate the contribution of the integral (6.3) on the other three sides. Consider first the side on the line $y = M$. Since the eigenvalues of \hat{A} and A are real it follows that

$$\|(x + Mi)I_n - \hat{A}\|_2, \|(x + Mi)I_n - A\|_2 \geq M$$

$$\begin{aligned} &\Rightarrow \\ &\|((x + Mi) - \hat{A})^{-1}\|_2, \|((x + Mi) - A)^{-1}\|_2 \leq \frac{1}{M}. \end{aligned}$$

Hence the contribution of (6.3) on the upper side of the square is bounded above by $\frac{8M\sqrt{n}}{2\pi M^2} = \frac{4\sqrt{n}}{\pi M}$. The same upper estimate holds for the lower side of the square on the line $y = -M$. We now estimate from above the contribution of (6.3) on the right side of the square on $x = x_0 + 2M$. Since the eigenvalues of \hat{A} and A are real and at most n it follows that

$$\begin{aligned} &\|(x_0 + 2M + yi)I_n - \hat{A}\|_2, \|(x_0 + 2M + yi)I_n - A\|_2 \geq 2M - n \\ &\Rightarrow \\ &\|((x_0 + 2M + yi)I_n - \hat{A})^{-1}\|_2, \|((x_0 + 2M + yi)I_n - A)^{-1}\|_2 \leq \frac{1}{2M - n}. \end{aligned}$$

Hence the contribution of (6.3) on the righthand side of the square is bounded above by $\frac{4M\sqrt{n}}{\pi(2M-n)^2}$. Therefore

$$\|P_k(\hat{A}) - P_k(A)\|_2 \leq \frac{8}{(c' - 8)} + 2 \cdot \frac{4\sqrt{n}}{\pi M} + \frac{4M\sqrt{n}}{\pi(2M - n)^2}.$$

Letting $M \rightarrow \infty$ we obtain $\|P_k(\hat{A}) - P_k(A)\|_2 \leq \frac{8}{c' - 8}$. We therefore get (6.2) if

$$\frac{8}{c' - 8} = \frac{8}{(p - q)c - 16} \leq \epsilon, \tag{6.4}$$

completing the proof. □

6.3 A bound on $\|P_k(\hat{A}) - P_k(A)\|_F$

Now we estimate the Frobenius norm of $P_k(\hat{A}) - P_k(A)$. Recall that for any matrix $B = (b_{ij}) \in \mathbb{R}^{m \times n}$

$$\|B\|_F := \sqrt{\sum_{i,j=1}^{m,n} b_{ij}^2}.$$

Moreover, if B is a symmetric, then

$$\|B\|_F^2 = \sum_{i=1}^n \lambda_i(B)^2. \tag{6.5}$$

Therefore we obtain the following lemma:

Lemma 16. $\|P_k(\hat{A}) - P_k(A)\|_F^2 \leq 2k\|P_k(\hat{A}) - P_k(A)\|_2^2$.

Proof. Recall that $P_k(\hat{A})$ and $P_k(A)$ have rank k . Hence $P_k(\hat{A}) - P_k(A)$ has rank at most $2k$. So $P_k(\hat{A}) - P_k(A)$ has at most $2k$ nonzero eigenvalues. The lemma thus follows from (6.5). □

7 Proof of algorithm’s corectness

The proof of Algorithm 1’s correctness goes roughly as follows. We will prove using the spectral analysis in Sections 5-6 that a.s. there is a column j for which $\|P_k(\hat{A})\mathbf{1}_{W_j}\|_2$ is “large” (Lemma 17). Next, we will show that any for any such j , the set W_j consists mostly of vertices from a single cluster (Lemma 18). Finally, we show how to recover this cluster exactly by looking at how many neighbors each vertex has in W_j (Lemmas 19-20).

This argument shows that Algorithm 1 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 by

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 [19, 28, 33]. The details of this approach, which we call “preprocessing the randomness,” are presented in Section 7.3

7.1 Notation

We will use the following notation in our proof:

- $H = (h_{ij})_{i,j=1}^n$ – the “true cluster matrix” as defined in (6.1), i.e., $h_{ij} = 1$ if i and j are in the same cluster, 0 else.
- W_1, \dots, W_n as defined as in step 3 of Algorithm 1.
- $\hat{H} = (\hat{h}_{ij})_{i,j=1}^n := (\mathbf{1}_{W_1}, \dots, \mathbf{1}_{W_n})$ – the “estimated cluster 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.

7.2 Technical lemmas

The proof of Theorem 4 relies on several additional lemmas. Lemmas 17-20 fit together roughly as follows:

- Lemma 17 says that a.s. there is a column j for which $\|P_k(\hat{A})\mathbf{1}_{W_j}\|_2$ is large.
- Lemma 18 says that for such a column j , W_j consists mostly of vertices from a single cluster C_i .
- Lemmas 19 and 20 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 17. *Assume \hat{A} satisfies (5.1). Then there exists a column j such that*

$$\|P_k(\hat{A})\mathbf{1}_{W_j}\|_2 \geq (1 - 8\epsilon^2 - \epsilon)\sqrt{s}. \quad (7.1)$$

Proof. Lemmas 15 and 16 together give

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

By definition of \hat{H} ,

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

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

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

Recall also (6.1) that $P_k(A) = \frac{1}{s}H$. Therefore

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

The triangle inequality then yields:

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

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

$$\sum_{i=1}^n (h_{ij^*} - \hat{h}_{ij^*})^2 \leq \frac{1}{n} \cdot 8k\epsilon^2 s^2 = 8\epsilon^2 s. \quad (7.2)$$

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

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

By (7.2) 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 15 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 18. *Assume \hat{A} satisfies (5.1) and j satisfies (7.1). 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

$$\|P_k(A)\mathbf{1}_W\|_2 \geq (1 - 8\epsilon^2 - 2\epsilon)\sqrt{s}. \quad (7.3)$$

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 (7.3).
- $t(W) = \tau$.

– $\|P_k(A)\mathbf{1}_W\|_2$ 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

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

We claim that

1. $t_2(W) < t_1(W)$. Suppose to the contrary. Maximize $\sum_{i=1}^k x_i^2$, such that $x_1 = x_2 \geq x_3 \geq \dots \geq x_k \geq 0$ and $\sum_{i=1}^k x_i = s$. It is easy to show that the maximum occurs when $x_1 = x_2 = \frac{s}{2}$ and $x_3 = \dots = x_k = 0$. Hence $\sum_{i=1}^k t_i(W)^2 \leq \frac{s^2}{2}$. By (7.3) 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. $t_3(W) = \dots = t_k(W) = 0$. 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 (7.3) 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. \square

Lemma 19. Consider cluster C_i and vertex $j \in [n]$. If $j \in C_i$, then

$$|N_{\hat{G}}(j) \cap C_i| \geq (p - \epsilon)s \tag{7.4}$$

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

$$|N_{\hat{G}}(j) \cap C_i| \leq (q + \epsilon)s \tag{7.5}$$

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

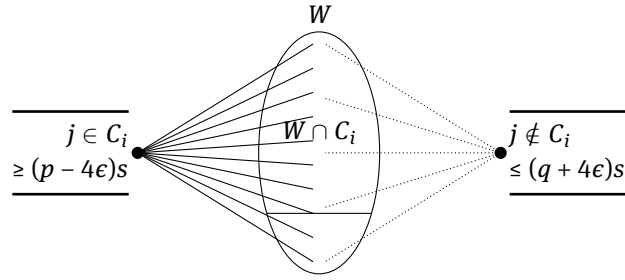


Figure 4: If W has large overlap with C_i , then a.s. vertices in C_i will have many neighbors in W , while vertices not in C_i will have relatively few neighbors in W .

Proof. Let $j \in C_i$. Then $E[|N(j) \cap C_i|] = p(s - 1)$, so Hoeffding’s inequality 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 $E[|N(j) \cap C_i|] = qs$, so

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

Lemma 20. 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 (7.4), then $|N_{\hat{C}}(j) \cap W| \geq (p - 4\epsilon)s$.
- b) If $j \in [n] \setminus C_i$ and j satisfies (7.5), then $|N_{\hat{C}}(j) \cap W| \leq (q + 4\epsilon)s$.

Proof. Assume $j \in C_i$ and j satisfies (7.4). 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

$$p - 4\epsilon \geq q + 4\epsilon. \tag{7.6}$$

7.3 Main proof

To prove Theorem 4, 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 1 *definitely* succeeds.
2. They all occur simultaneously a.s.

Therefore, Algorithm 1 succeeds a.s.

Before we define the events let us introduce some notation:

- 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. $\hat{G}_J := \hat{G} [\bigcup_{i \in J} C_i]$. Then for any fixed J we have

$$\hat{G}_J \sim \mathcal{G}(|J|s, \{C_i : i \in J\}, p, q). \quad (7.7)$$

- 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. $B_J := B [\bigcup_{i \in J} C_i]$.

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 (5.1) and that (7.4) and (7.5) 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 4. 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 1 succeeds in every iteration.

For the base case, take the original graph $\hat{G} = \hat{G}_{[k]}$ considered in the first iteration. Since $E_{[k]}$ is assumed to hold, (5.1) is satisfied. Thus, by Lemma 17, the column $j = j^*$ identified in step 4 satisfies (7.1). Then by Lemma 18 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 20.

Now assume Algorithm 1 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 11-18 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 17 the column $j = j^*$ identified in step 4 of Algorithm 1 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 7.1-7.2) are constructed from \hat{G}_J , not the original graph \hat{G} . Now by Lemma 18 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 20.

We have thus proved that Algorithm 1 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 10

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

By Lemma 19, for any i, j

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

Taking a union bound 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 1 succeeds with probability $\geq 1 - \left(\frac{2}{e}\right)^{\sqrt{n}} - n^{3/2} e^{-\sqrt{n}}$.

Note that we require (7.6) in order for step 5 of Algorithm 1 to correctly recover a cluster according to Lemma 20. In addition, the proof of Lemma 18 requires $\epsilon \leq .1$. By (6.4), we can satisfy both of these conditions by setting $c := \max\left\{\frac{88}{p-q}, \frac{72}{(p-q)^2}\right\}$. \square

Acknowledgement: We would like to thank the anonymous reviewers of this and previous versions of our paper for pointing us to relevant past work and for calling our attention to the fact that the iterations of Algorithm 1 cannot be handled independently. The latter issue fixed herein by “preprocessing the randomness” as discussed in Sections 3 and 7.

This research was partly supported by ARO grant 66497-NS and NSF grant IIS-1526379.

References

- [1] Nir Ailon, Yudong Chen, and Huan Xu. Breaking the small cluster barrier of graph clustering. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 995–1003, 2013.
- [2] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Struct. Algorithms*, 13(3-4):457–466, 1998.
- [3] Noga Alon, Michael Krivelevich, and Van H. Vu. On the concentration of eigenvalues of random symmetric matrices. *Israel Journal of Mathematics*, 131(1):259–267, 2002.
- [4] Brendan P. W. Ames. Guaranteed clustering and biclustering via semidefinite programming. *Mathematical Programming*, 147(1-2):429–465, 2014.
- [5] Brendan P. W. Ames and Stephen A. Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Math. Program.*, 129(1):69–89, 2011.
- [6] Afonso S. Bandeira and Ramon van Handel. Sharp nonasymptotic bounds on the norm of random matrices with independent entries. *Ann. Probab.*, 44(4):2479–2506, 07 2016.
- [7] Béla Bollobás and Paul Erdős. Cliques in random graphs. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 80, pages 419–427. Cambridge Univ Press, 1976.
- [8] Yudong Chen, S. Sanghavi, and Huan Xu. Improved graph clustering. *Information Theory, IEEE Transactions on*, 60(10):6440–6455, October 2014.
- [9] Yudong Chen and Jiaming Xu. Statistical-computational phase transitions in planted models: The high-dimensional setting. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 244–252, 2014.
- [10] Amin Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(02):227–284, 2010.
- [11] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
- [12] Yael Dekel, Ori Gurel-Gurevich, and Yuval Peres. Finding hidden cliques in linear time with high probability. In *Proceedings of ANALCO*, pages 67–75, 2011.
- [13] Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959 1959.
- [14] Uriel Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Struct. Algorithms*, 16(2):195–208, 2000.
- [15] Uriel Feige and Dorit Ron. Finding hidden cliques in linear time. In *Proceedings of AofA*, pages 189–204, 2010.
- [16] Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh Vempala, and Ying Xiao. 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.
- [17] Shmuel Friedland. *Matrices*. World Scientific, 2015.
- [18] Zoltán Füredi and János Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.
- [19] Joachim Giesen and Dieter Mitsche. Reconstructing many partitions using spectral techniques. In *Proceedings of the 15th International Symposium on Fundamentals of Computation Theory*, 2005.
- [20] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.
- [21] Ming Gu. Subspace iteration randomization and singular value problems. *SIAM Journal on Scientific Computing*, 37(3):A1139–A1173, 2015.
- [22] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011.
- [23] Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992.
- [24] Richard M. Karp. Probabilistic analysis of graph-theoretic algorithms. In *Proceedings of Computer Science and Statistics 12th Annual Symposium on the Interface*, page 173, 1979.
- [25] N. Kishore Kumar and J. Schneider. Literature survey on low rank approximation of matrices. *Linear and Multilinear Algebra*, pages 1–33, 2016.
- [26] Luděk Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.
- [27] François Le Gall. 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.

- [28] Frank McSherry. Spectral partitioning of random graphs. In *FOCS*, pages 529–537, 2001.
- [29] Nam H. Nguyen, Thong T. Do, and Trac D. Tran. 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.
- [30] Samet Oymak and Babak Hassibi. Finding dense clusters via “low rank + sparse” decomposition. *arXiv preprint arXiv:1104.5186*, 2011.
- [31] G.W. Stewart. *Introduction to matrix computations*. Computer science and applied mathematics. Academic Press, 1973.
- [32] Van Vu. Spectral norm of random matrices. *Combinatorica*, 27(6):721–736, 2007.
- [33] Van Vu. A simple SVD algorithm for finding hidden partitions. *arXiv preprint arXiv:1404.3918*, 2014.

A Comparison with previous results

The following table compares our work with previous algorithms for recovering planted partitions. Note that some of the algorithms apply to more general planted clustering settings, but here we list their performance *only in the setting considered in this paper*, i.e. for planted partition with constant edge probabilities and uniform cluster sizes. In particular, note that the the well-known results [28, 33] achieve the \sqrt{n} bound in certain settings, but only when some of the clusters are size $\omega(\sqrt{n})$.

Paper	Minimum cluster size for planted partition	Algorithm type
McSherry 2001 [28, Theorem 4]	$\Omega(n^{2/3})$	Spectral
Giesen & Mitsche 2005 [19]	$\Omega(\sqrt{n})$	Spectral
Oymak & Hassibi 2011 [30]	$\Omega(\sqrt{n})$	Convex programming
Ames 2014 [4]	$\Omega(\sqrt{n})$	Semidefinite programming
Chen et al. 2014 [8]	$\Omega(\sqrt{n})$	Convex programming
Vu 2014 [33, Theorem 2]	$\omega(\sqrt{n \log n})$	Spectral
Our result	$\Omega(\sqrt{n})$	Spectral

Thus, we see that while many have succeeded in recovering clusters of size $\Omega(\sqrt{n})$, prior to this paper, only Giesen and Mitsche [19] had done so using a purely spectral approach (to the best of our knowledge). While their proof techniques have much in common with our own, our algorithm is arguably much simpler. Furthermore, their algorithm takes $2^{\Theta(k)} \cdot \text{poly}(n)$ time and is thus inefficient when $k = \omega(\log n)$.