Abstract

Clustering Algorithms for Random and Pseudo-random Structures

Pradipta Mitra

2008

Partitioning of a set objects into a number of clusters according to a suitable distance metric is one of the major questions in data mining, information retrieval and other fields of computer science. This thesis describes and analyzes algorithms for clustering and partitioning data generated from random and pseudo-random models.

In random models, one assumes that the data matrix to be partitioned is generated from a simple distribution. A common feature of all the algorithms analyzed is that they are all spectral algorithms, as they employ information about the spectrum of the data matrix to perform clustering. We provide new results in a number of directions. A method based on the second singular vector is analyzed for a mixture model for graphs. The use of the notion of pseudo-randomness is another important aspect of our work. Pseudo-randomness, the idea to use deterministic definitions to capture properties of randomness, is used to extend the notion of probabilistic models, thus making it possible to model clustering problems for sparse (constant-degree) graphs. This work also proposes the first geometric, projection based algorithms known for discrete distributions, which allows a number of generalizations. In addition, entry-wise bounds for eigenvectors of adjacency matrices of random graphs are studied, and their implications for spectral algorithms are considered.
Clustering Algorithms for Random and
Pseudo-random Structures

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Pradipta Mitra

Dissertation Director: Ravindran Kannan

December, 2008
## Contents

1 Introduction 1

1.1 Linear Algebra Background .............................. 3
1.2 Random Graphs ........................................... 7
1.3 Models for Clustering .................................... 8
  1.3.1 Mixture models ...................................... 11
1.4 Spectral Clustering .................................... 12
1.5 Large Deviation Inequalities ............................ 14
1.6 Our contributions ...................................... 15
  1.6.1 Analysis of the second singular vector method ...... 15
  1.6.2 Quasi-random models for clustering ................. 15
  1.6.3 Entrywise bounds for eigenvectors .................. 16
  1.6.4 Projective algorithms for discrete distributions .... 17

2 Spectral Clustering by Recursive Partitioning 18

2.1 Introduction ........................................... 18
2.2 Model and Main Result .................................. 20
2.3 Related Work .......................................... 22
2.4 Algorithm .............................................. 23
  2.4.1 Proof ............................................... 24
# Quasi-randomness and Clustering

## Introduction

## Model

## Algorithm

- **3.3.1** A Spectral algorithm
- **3.3.2** The Clean-up Phase

# Entrywise Bounds on Eigenvectors of Random Graphs

## Introduction

## Model and Main Result

## Bound

## Lower Bound

## Application to Clustering

- **4.5.1** Model
- **4.5.2** Related work
- **4.5.3** The first analysis
- **4.5.4** The second analysis

# A Simple Algorithm for Discrete Distributions

## Introduction

## Model

## The Algorithm

## Analysis

# Conclusion
List of Figures

4.1 Entry wise bounds for eigenvectors of random graphs . . . . . . . . . . . 68
4.2 Tightness of entry wise bounds . . . . . . . . . . . . . . . . . . . . . . 68
5.1 Projections in mixture models . . . . . . . . . . . . . . . . . . . . . . 92
Chapter 1

Introduction

Clustering is the classification of objects into groups according to some notion of closeness among them. Clustering is an important problem in many fields, including information retrieval, machine learning, data mining, pattern recognition, and computational biology. Solving a clustering problem involves coming up with an appropriate distance metric between objects, and then designing an algorithm that will work well in theory, or practice, or both. With the explosion of information waiting to be analyzed, thanks to the World Wide Web and other sources, the problem has become all the more important in recent years. The clustering problem is a well-studied one (for a survey of the field, see [26]), but it is far from exhausted as a subject of research.

One of the main obstacles in theoretical research on Clustering is that almost all reasonable definitions of Clustering problems involve NP-hardness. There are many combinatorial optimization problems that can be said to define a clustering of the input data — k-center, k-median, k-means and Sparsest cut problems, to name a few, which all turn out to be NP-hard to optimize.

The Theoretical Computer Science community has dealt with NP-hardness in a
number of ways. One approach is the theory of approximation algorithms, where one only seeks an approximation, instead of the exact solution, in polynomial time. Other approaches include average case analysis, and its more recent relative — heuristic analysis. The central goal in Heuristic analysis is to explain why certain algorithms ("heuristics") work well in practice, even though the problems they seemingly solve are NP-hard, or there are worst case instances of the problem where the algorithm in question performs badly. The basic idea in Heuristic analysis is that the input is unlikely to be worst-case. An example of such an approach is Smoothed analysis pioneered by Spielman and Teng [42], that explained why the simplex algorithm for Linear programming works well in practice.

In this dissertation, we are concerned with the analysis of a well-known algorithm, or rather a family of algorithms. These algorithms generally are known as "spectral algorithms". Starting from the observation that most inputs for clustering problems can be viewed as matrices, these algorithms involve spectral analysis of the data matrix, by computing singular vectors of eigenvectors of the data. Spectral algorithms are known to perform well in a wide variety of situations [7, 18, 44]. Yet, as we have already mentioned, most natural clustering problems are NP-hard. This situation calls for an explanation of the success of spectral algorithms. A number of attempts have been made to do this. For example, the problem of finding the spectrum is known to be a relaxation of the problem of finding a sparse cut, which has been used as an explanation for the success of spectral algorithms [29]. In [41], it was shown that the spectral method finds good cuts in planar graphs. Another idea has been to show that spectral algorithms work when the data is generated from a "nice" probability distribution. This probabilistic approach is the focus of this work.

The rest of this Chapter is organized as follows. Sections 1.1 and 1.2 will respectively introduce basic linear algebraic concepts, and the important notion of random
graphs. We will discuss probabilistic clustering models in Section 1.3 and Spectral algorithms in Section 1.4. In Section 1.6, the main contributions of this dissertation will be outlined.

1.1 Linear Algebra Background

Vectors and Vector norms

Let $\mathbf{R}$ be the real line. Then a vector $x$ is an element in $\mathbf{R}^n$, for some $n$. A vector norm is a function $f : \mathbf{R}^n \to \mathbf{R}$ such that, for all $x \in \mathbf{R}^n$

$$f(x) \geq 0$$

$$f(x) = 0 \text{ iff } x = 0$$

$$f(x + y) \leq f(x) + f(y) \quad \text{[triangle inequality]}$$

$$f(\alpha x) = |\alpha| f(x), \quad \text{for all } \alpha \in \mathbf{R}$$

We use $x_i$ to denote the $i^{th}$ entry of $x$. The most common class of norms are the $p$-norms, defined for all $p \geq 0$ as:

$$\|x\|_p = \left( \sum_{i \in [n]} |x_i|^p \right)^{\frac{1}{p}}$$

And the cases of 1, 2 and $\infty$ require special mention, the second of which is the well
known euclidean norm:

\[
\|x\|_1 = \sum_{i \in [n]} |x_i|
\]

\[
\|x\|_2 = \left( \sum_{i \in [n]} |x_i|^2 \right)^{\frac{1}{2}}
\]

\[
\|x\|_{\infty} = \max_{i \in [n]} |x_i|
\]

The simplified notation \(\|x\|\) will be used to denote the euclidean (i.e. \(\| \cdot \|_2\)) norm.

The inner product of two vectors is defined by

\[
(x \cdot y) = \sum_{i \in [n]} x_i y_i
\]

Note that \((x \cdot x) = \|x\|^2\). Two vectors \(x\) and \(y\) are orthogonal if \((x \cdot y) = 0\).

Following are useful relations between norms

\[
\|x\|_2 \leq \|x\|_1 \leq \sqrt{n} \|x\|_2
\]

\[
\|x\|_{\infty} \leq \|x\|_2 \leq \sqrt{n} \|x\|_{\infty}
\]

And they are all special cases of what is a good candidate for the most important inequality in analysis:

**Cauchy-Schwartz Inequality** If \(x, y \in \mathbb{R}^n\)

\[
|(x \cdot y)| \leq \|x\| \|y\| \quad (1.1)
\]

4
Matrices and Matrix norms

Matrices are elements of $\mathbb{R}^{m \times n}$, usually understood as a collection of numbers arranged in an array with $m$ rows and $n$ columns. As matrices are simply collections of vectors, matrix norms are often defined in terms of vector norms.

The $p$-norm of a matrix $A \in \mathbb{R}^{m \times n}$ is defined as

$$\|A\|_p = \sup_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p}$$

It is clear that this is equivalent to

$$\|A\|_p = \max_{\|x\|_p = 1} \|Ax\|_p$$

Another important norm is the Frobenius norm

$$\|A\|_F = \sqrt{\sum_{i \in [m]} \sum_{j \in [n]} A_{ij}^2}$$

where $A_{ij}$ is the entry of $A$ in the $i^{th}$ row and $j^{th}$ column. A matrix is called orthogonal is its columns are orthogonal, and each column has euclidean norm 1.

Singular Value Decomposition

One of the most important fact about matrices that we will use is that there exists a Singular Value Decomposition (SVD) for each matrix, and it can be computed in polynomial time.

If $A$ is a $m \times n$ matrix, then there exist orthogonal matrices $U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}$
and a diagonal matrix $S \in \mathbb{R}^{m \times n}$ such that

$$A = USV^T$$

where the diagonals of $S$ are $\sigma_1 \geq \sigma_2 \ldots \sigma_p \geq 0$ ($p = \min(m,n)$), the columns of $U$ are $u_i$ and columns of $V$ are $v_i$. The $\sigma_i$ are known as the singular values of $A$, and the vectors $u_i$ and $v_i$ are called the left and right singular vectors, respectively. Without loss of generality, the singular values are often assumed to be sorted in non-increasing order of value, hence $\sigma_i$ is the first or top singular value, and so forth. Similarly, $v_i$ is known as the $i^{th}$ right singular vector.

Assume $r$ is the number of nonzero singular values of $A$. Then $\text{rank}(A) = r$ and

$$A = \sum_{i \in [r]} \sigma_i u_i v_i^T$$

This equation readily provides a number of useful estimates of the relation between matrix norms, and singular values:

$$
\|A\|_2 = \sigma_1 \\
\|A\|_F^2 = \sum_{i \in [r]} \sigma_i^2 \\
\|A\| \leq \|A\|_F \leq \sqrt{r} \|A\|
$$

**Eigen-decomposition**

In the special case where $A$ is a square symmetric matrix (such as an adjacency matrix of a undirected graph), we can get a decomposition where $U = V$. To be exact there
is a real orthogonal matrix $Q$ such that

$$A = QSQ^T$$  \hspace{1cm} (1.2)$$

where $S$ is a diagonal matrix. The columns of $Q$ are known as the eigenvectors and the diagonals of $S$ are known as the eigenvalues of $A$ (note that the eigenvalues are not necessarily positive). Let $q_i$ and $\lambda_i$ be the $i^{th}$ eigenvector and eigenvalue, respectively. Then following is a useful property of eigenvectors:

$$Aq_i = \lambda_i q_i$$

### 1.2 Random Graphs

A graph $G = G(V, E)$ is defined by a set of vertices $V$ and a set of edges $E$. Let $e(A, B)$ be the number of edges between two sets $A, B \in V$. We will use shorthands $e(A) \equiv e(A, V)$ and $e(v) \equiv e(\{v\}, V)$ [for a vertex $v \in V$].

First introduced by Gilbert [25] and independently by Erdos and Renyi [21], random graphs are probabilistic models for generating graphs. We will consider the well-known $\mathcal{G}_{n,p}$ model.

The probability space $\mathcal{G}_{n,p}$ is defined on graphs with $n$ vertices for $n \geq 2$, $0 \leq p \leq 1$. We generate a random element of this probability space in the following manner: each of $\binom{n}{2}$ possible edges of the graph are selected independently, each with probability $p$. Equivalently, for each possible graph $G$ on $n$ vertices, the probability of selecting $G$ from $\mathcal{G}_{n,p}$ is

$$\mathbb{P}(G) = p^m (1 - p)^{N - m}$$  \hspace{1cm} (1.3)$$

where $m = \text{number of edges in } G$ and $N = \binom{n}{2}$. 
The expected degree of each vertex is $\mathbb{E}(e(i)) \equiv d = (n-1)p$. For our purposes we will usually assume $p \geq \frac{\log n}{n}$. This will allow us to prove some useful concentration properties.

**Lemma 1** Let $G \in \mathcal{G}_{n,p}$. With probability at least $1 - \frac{1}{n^3}$, for all vertices $v \in G$

$$|e(v) - d| \leq 3\sqrt{d \log n}$$

where $e(v)$ is the degree of $v$.

**Proof** Elementary use of Chernoff inequality (see [35], or Theorem 3), followed by an union bound.

The random graphs also have a property known as quasi-randomness or pseudo-randomness, which says that the number of edges between every two subsets of a random graph is close to its expectation. This is a very useful property.

**Lemma 2** Assume $p \geq \frac{\log^2 n}{n}$. With probability $1 - o(1)$, for all sets $A, B \in V$

$$|e(A, B) - p|A||B| \leq 2\sqrt{np|A||B|}$$

**Proof** This also can be proved by applying the Chernoff inequality a few times [32].

### 1.3 Models for Clustering

This dissertation deals with two related models of clustering — one focusing on graphs and discrete matrices and the other on continuous high-dimensional distributions.
Planted Partition Models for Graphs

After the notion of random graphs was introduced, naturally many algorithmic questions were asked about them. This was particularly interesting as many combinatorial optimization problems on graphs are NP-hard. The idea was, if one could solve these problems on random graphs with high probability, one could claim that for “most graphs”, the problems had polynomial time algorithms to solve them. Significant amount of research has been done on finding Cliques, Colorings, Hamiltonian Paths and Bi-sections of Random graphs. We refer the reader to the survey by Frieze and McDiarmid [23] as a reference to such results.

With high probability, random graphs have large coloring numbers, very small cliques, etc. For example, the largest Clique in a random graph has size $\frac{2\log(np)}{\log(\frac{1}{1-p})}$ almost surely. Furthermore, an essentially greedy algorithm can find a clique half that size very easily [23]. Now, a distribution where the size of the largest clique can be at most logarithmic in the size of vertices does not seems to be a model of much practical interest. To address this problem, researchers have introduced models based on random graphs, yet where the edge distributions are not quite uniform. These distributions usually depend on the combinatorial problem one is trying to model. The idea is to skew the probabilities of edges appearing in a way such that the possibility of having a combinatorial object of a certain size increases. Well studied problems in this vein include $k$-coloring [3], max-clique [4], and min bi-section [10]. Here are the usual models for the first two problems:

- **$k$-Coloring**: Fix a partitioning of the $n$-vertices into $k$ sets of equal size. Let these sets be $T_1 \ldots T_k$. Now for each $r \neq s$ and all $u \in T_r$, $v \in T_s$, include the edge $e(u, v)$ independently with probability $p$. Intra-partition edges are not included. This defines a natural $k$-coloring on the graph. The problem is to
find the coloring given a graph thus generated.

• $k$-Clique: Fix a subset $C$ of the graph where $|C| = k$. Now select all edges within vertices in $C$ to be present (with probability 1). All other edges are selected independently with some (smaller) probability $p$. The graph clearly has a $k$-clique. The problem is to find that “hidden” clique.

For each problem, the goal of the algorithm is the recovery of the “hidden” structure, be it coloring, clique, or bi-section. The main question is for what parameters ($p$, $k$ etc) can the structure be found in polynomial time. These problems are often call “planted problems”, since the probabilities are tweaked to “plant” a certain combinatorial structure in the graph.

The planted $k$-clique problem was suggested by Karp in [31]. It is customary to study this problem for $p = \frac{1}{2}$. It was observed that if $k \geq 2\sqrt{n \log n}$, the clique members are simply the vertices of highest degree. The first (and only) non-trivial result for the problem was achieved by Alon et.al. [4] who showed that a planted clique of size $\Omega(\sqrt{n})$ can be found. Similarly, a lot of work has been done the problem of coloring random graphs which have $k$-colorings. Dyer and Frieze [20] presented an algorithm that optimally colors uniformly random $k$-colorable graphs for fixed $k$, with high probability. However, most $k$-colorable graphs are dense, and therefore easier to color than sparse graphs. Alon and Kahale [3] provided an algorithm that color random sparse graphs (where $p = \frac{c}{n}$ for a constant $c$) properly with high probability. McSherry [33] introduced the “Planted partition model” that generalizes many of these models. We will describe this model in fuller detail in Chapter 2.
1.3.1 Mixture models

Results for continuous distributions have focused on well-known “mixture models”. Here, instead of a graph, the input is an object-feature matrix. Typically, a collection of $m$ “objects” is analyzed, where each object is a vector in $n$-space. The input can thus be represented as a $n \times m$ matrix $A$, each column representing an object and each row representing a “feature”.

Here we start with $k$ probability distributions $D_1, D_2, \ldots, D_k$ where each is a distribution on $\mathbb{R}^n$. With each $D_r$ we can associate a center $\mu_r$ of the distribution, defined by $\mu_r = \int_{v \in \mathbb{R}^n} v D_r(v)$. A mixture of these probability distributions is a density of the form $w_1 D_1 + w_2 D_2 + \ldots + w_k D_k$, where the $w_i$ are fixed non-negative reals summing to 1. A clustering question in this case is to partition the vectors according to the underlying probability distributions. The most widely used distributions for these models is high-dimensional Gaussians, or as they are often known, multivariate normal distributions. These distributions are characterized by a center $\mu$ and a non-singular symmetric covariance matrix $\Sigma$. A $n$-dimensional Gaussian $N(\mu, \Sigma)$ has the density function:

$$p(x) = \frac{1}{(2\pi)^{n/2}\det(\Sigma)^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

where $\det(\Sigma)$ denotes the determinant of $\Sigma$. Possibly the most popular method for clustering mixture of Gaussian’s is the Expectation Minimization (EM) algorithm formalized by Dempster, Laird, and Rubin [17]. Inspite of it’s popularity, EM is a local search heuristic with not much analysis to support it, and it can be stuck at local minima. A more theoretically rigorous approach was introduced by Dasgupta [16] which used random projections to cluster a mixture of spherical Gaussians. More results have followed [5, 14, 28]. It is now known how to cluster log-concave distri-
butions [28], $f$-concentrated distributions [1] and other generalizations of Gaussians [14].

1.4 Spectral Clustering

Though a large variety of approaches have been used to solve clustering problems, one that has been consistently successful is the spectral approach. Here one usually uses spectral decomposition of a matrix to find a low-rank approximation of the data. Since the running times of many algorithms depend on the rank of the input matrix, it might be possible to use some exhaustive procedure to cluster the low-rank approximation of the data.

In another direction, there is reason to believe the second eigenvector of a graph might be used to find a good partition of a graph. This is because sparse cuts are related to the second smallest eigenvalue of the laplacian of graph (which is a simple transformation of the adjacency matrix) through a result often known as Cheeger’s inequality (See [41] for references). The result is not strong enough to imply soundness of spectral algorithms for all cases, but provides some support the case for using spectral methods for partitioning graphs.

Let us consider a rather simple example to provide a little bit of intuition to support the case for spectral methods. Here’s a matrix (indeed, an adjacency matrix of a graph) with two clear clusters:
The clusters are of course $T_1 = \{1,2,3,4\}$ and $T_2 = \{5,6,7,8\}$ (the numbers refer to the rows/columns). It is not too hard to see that

1. The first eigenvector of $X$ is $e_1 = \{1,1\ldots,1\}^T$ with eigenvalue 5.

2. The second eigenvector of $X$ is $e_2 = \{1,1,1,-1,-1,-1\}^T$ with eigenvalue 3.

The interesting thing is that the second eigenvector has revealed the right clustering. One hopes that some generalization of this will continue to hold for more complex situations.

For planted partition models, the study of spectral methods started with Boppana [10], but his approach required the solution to a convex optimization problem. Alon and Kahale [3] presented a spectral algorithm that used the last $k$ eigenvectors of the adjacency matrix of the graph to solve the planted $k$-coloring problem. In [4], Alon et al. presented a related algorithm to solve the $k$-Clique problem where $k = \Omega(\sqrt{n})$. These two papers contained a lot of ideas that recur in papers for planted models to date. McSherry [33] proposed a general model that would capture many optimization problems on graphs, and a single algorithm that would work for
them.

The first projection based algorithm for mixtures of Gaussians was Dasgupta’s
algorithm [16] that used random projections. Vempala and Wang [43] analyzed a
natural spectral algorithm and showed that it provided essentially optimal results.
These results were generalized in terms of the distributions involved in a number of
subsequent papers [1, 28].

1.5 Large Deviation Inequalities

We will use the Chernoff bound and some related inequalities many times in this
thesis. Here is the most popular form of Chernoff bound that we use [35]:

**Theorem 3** Let $X_1, \ldots, X_n$ be $n$ independent Bernoulli trials, where
$\mathbb{P}[X_i = 1] = p_i$, and $\mathbb{P}[X_i = 0] = q_i = 1 - p_i$. Let $X = \sum_i X_i$ and $\mu = \mathbb{E}(X)$ Then for any $\delta > 0$

$$
\mathbb{P}(X > (1 + \delta) \mu) \leq \left( \frac{e^\delta}{(1 + \delta)^{1+\delta}} \right)^\mu
$$

(1.4)

And

$$
\mathbb{P}(X < (1 - \delta) \mu) \leq \exp(-\mu \delta^2 / 2)
$$

(1.5)

The upper bound in somewhat complicated, which is why the following version
is often used:

**Corollary 4** Under the same assumptions as Theorem 3, for $\delta < 2e - 1$

$$
\mathbb{P}(X > (1 + \delta) \mu) \leq \exp(-\mu \delta^2 / 4)
$$

(1.6)

And for $\delta \geq 2e - 1$

$$
\mathbb{P}(X > (1 + \delta) \mu) \leq 2^{-\mu(1+\delta)}
$$

(1.7)
1.6 Our contributions

In this section, we provide the outline of our results, which will be described in detail in the next four chapters. A more technical summary of the results will be provided in Chapter 6. Our main focus is on the planted partition model, but we will present results for mixture models as well.

1.6.1 Analysis of the second singular vector method

In Chapter 2, we analyze a common heuristic for data partitioning. Assuming that the objects are arranged in columns, the idea is to look at the second right singular vector $v_2$ of the data matrix. The intuition is this: the top singular vector is often close to the $1$ vector (the “all ones” vector), and not very useful, hence $v_2$ is the most important of the interesting singular vectors.

Now, $v_2$ (or any singular vector) can be seen as a function from the objects to $\mathbb{R}$. We can sort the entries in $v_2$, and use this to bi-partition the data in one of many natural ways. One can, for example, split the objects into two equal sized parts, or define one part to be the objects with positive entries in $v_2$, and the other part to be the rest. We study an equally natural algorithm: split the data in the place where there is the largest gap in (sorted) $v_2$. We provide proof of its soundness for the “planted partition model” introduced by McSherry [33].

1.6.2 Quasi-random models for clustering

In Chapter 3, we focus on a modeling problem. Our main goal is to modify the planted partition model to allow constant degree graphs. This is not quite possible in the probabilistic model. Essentially, the problem is that if the expected degree is less than poly-logarithmic in the size of the graph, many properties (such as number
of edges of a vertex) are as concentrated around their mean as one needs to have well-defined clusters.

Our idea is to do away with randomness, and use the concept of pseudo-randomness or quasi-randomness instead. Pseudo-randomness in general refers to deterministic definitions of objects that are meant to mimic the behavior of random objects. Once we model the problem successfully, we show that a spectral algorithm works for this model. The main challenge is that, since we have no recourse to independence between objects, many tricks used in planted partition models have to be discarded. We use a local improvement algorithm, and show that it converges to the correct clustering in polynomial time.

1.6.3 Entrywise bounds for eigenvectors

Entry wise bounds for eigenvectors of Random graphs are studied in Chapter 4. Let $A$ be the adjacency matrix of a random graph $G \in \mathcal{G}_{n,p}$. The quantity $\|A - \mathbb{E}(A)\|_2$ (which is essentially the second eigenvalue of $A$) is well understood, and central to most results in spectral clustering. Here we are concerned with something a bit different, namely $\|v_1(A) - v_1(\mathbb{E}(A))\|_\infty$, where $v_1(M)$ is the top eigenvector of the matrix $M$. This is a different notion of spectral stability, and coupled with spectral norm bounds, gives a fuller picture of the spectrum of $A$.

Our study of this entity is motivated by a question in spectral clustering. Simply stated, the question is about whether or not use of randomness is necessary for spectral algorithms. We use our approach to prove that we can do without using randomness for certain models.
1.6.4 Projective algorithms for discrete distributions

The topic of chapter 5 is to extend spectral clustering results to more general distributions. A major underlying problem in many spectral clustering results to date is that they either work for continuous distributions (mixture of Gaussians, etc.) or for discrete distributions (Planted partition model), but not for both. In some sense, the hardness of finding a unified algorithm lies in the discrete side of things. Studying the literature one finds that algorithms for continuous distributions are more natural, while algorithms for discrete models are often ad-hoc and problem specific, and appeal to the combinatorial structure of the data.

Our primary motivation in studying this issue was a conjecture of McSherry [33]. He conjectured that for planted partition models, combinatorial methods are unnecessary. We solve this conjecture, and this allows us to extend spectral clustering results considerably. Our algorithm works for distributions with subgaussian tails, a fairly general distribution, as well as dependent distributions.
Chapter 2

Spectral Clustering by Recursive Partitioning

2.1 Introduction

The result presented in this Section is joint work with Anirban Dasgupta, John Hopcroft and Ravi Kannan and is adopted from a paper published in European Symposium on Algorithms, 2006 [13].

We will deal with the clustering problem on graphs generated by the versatile planted partition model (See [10, 33]). In this probabilistic model, one starts from a probability matrix $A$. Each edge $(u, v)$ is then a random variable that is independently chosen to be present with a probability $A_{uv}$, and absent with probability $1 - A_{uv}$. The vertex set of the graph is partitioned into $k$ subsets $T_1, T_2, \ldots, T_k$. The probabilities $A_{uv}$ depend only on the parts to which the two endpoints $u$ and $v$ belong. The adjacency matrix $\tilde{A}$ of the random graph so generated is presented as input. Our task then is to identify the latent clusters $T_1, T_2, \ldots, T_k$ from $\tilde{A}$.

Spectral methods have been widely used for clustering problems, both for theo-
retical analysis as well as empirical and application areas. The underlying idea is to use information about the eigenvectors of \( \hat{A} \) to extract structure. There are different variations to this basic theme of spectral clustering, which can be divided essentially into two classes of algorithms.

1. Projection heuristics, in which the top few eigenvectors of the adjacency matrix \( \hat{A} \) are used to construct a low-dimensional representation of the data, which is then clustered.

2. The second eigenvector heuristic, in which the coordinates of the second eigenvector of \( \hat{A} \) is used to find a split of the vertex set into two parts. This technique is then applied recursively to each of the parts obtained.

Experimental results claiming the goodness of both spectral heuristics abound. The worst case theoretical guarantees [29] that have been obtained, however, do not seem to match the stellar performance of spectral methods on many inputs, and thus it is still an open question to characterize the class of inputs for which spectral heuristics do work well. In order to be able to formalize the average case behavior of spectral analysis, researchers have analyzed its performance on graphs generated by random models with latent structure [6, 33]. These graphs are generated by choosing edges independently according to a low-rank probability matrix. The low rank of the probability matrix reflects the small number of vertex types present in the unperturbed data. The intuition developed by Azar et al. [6] is that in such models, the random perturbations may cause the individual eigenvectors to vary significantly, but the subspace spanned by the top few eigenvectors remains stable. From this perspective, however, the second eigenvector technique does not seem to be well motivated, and it is an interesting question as to whether we can claim anything better than the worst case bounds for the second eigenvector heuristic in this setting.
In this chapter, we prove the goodness of the second eigenvector partitioning for the planted partition random graph model [6, 15, 20, 33].

Our main aim in analyzing the planted partition model using the second eigenvector technique is to try to bridge the gap between the worst case analysis and the actual performance. However, in doing so, we achieve a number of other goals too. The most significant among these is that we can get tighter guarantees than [33] in terms of the dependence on the maximum variance. The required separation between the centers of clusters $T_r$ and $T_s$ can now be in terms of $\sigma_r + \sigma_s$, the maximum variances in each of these two clusters, instead of the maximum variance $\sigma_{\text{max}}$ in the entire matrix. This gain could be significant if the maximum variance $\sigma_{\text{max}}$ is due to only one cluster, and thus can potentially lead to identification of a finer structure in the data. Our separation bounds are however worse than [1, 33] in terms of dependence on the number of clusters.

### 2.2 Model and Main Result

$A$ is a matrix of probabilities where the entry $A_{uv}$ is the probability of an edge being present between the vertices $u$ and $v$. The vertices are partitioned into $k$ clusters $T_1$, $T_2, \ldots, T_k$. The probability $A_{uv}$ depends only on the two clusters that the vertices $u$ and $v$ belong to. In other words, there exist probabilities $P_{rs}(= P_{sr})$ for all $r, s \in [k]$, such that, if $u \in T_r$ and $v \in T_s$, then $A_{uv} = P_{rs}$

The size of the $r^{th}$ cluster $T_r$ is $n_r$. Define $w_r = \frac{n_r}{n}$ and $n_{\text{min}} = \min_r \{n_r\}$, $w_{\text{min}} = \min_r \{w_r\}$. The characteristic vector of the cluster $T_r$ is denoted by $\vec{I}(r)$ defined as $\vec{I}(r)(i) = 1$ for $i \in T_r$ and 0 elsewhere.

Given the probability matrix $A$, the random graph $\hat{A}$ is then generated by independently setting each $\hat{A}_{uv}(= \hat{A}_{vu})$ to 1 with probability $A_{uv}$ and 0 otherwise. In
what follows, we use the terms “column” and “vertex” interchangeably, noting that vertex $x$ corresponds to column $A_x$.

Now, the expectation of the random variable $\hat{A}_{uv}$ is equal to $A_{uv}$. The variance of $\hat{A}_{uv}$ is thus $A_{uv}(1 - A_{uv})$. The maximum variance of any entry of $\hat{A}$ is denoted $\sigma^2$, and the maximum variance for all columns belonging to a cluster $T_r$ is denoted by $\sigma_r^2$. We usually denote a matrix of random variables by $\hat{X}$ and the expectation of $\hat{X}$ as $X = \mathbb{E}\hat{X}$. We will also denote vectors by $\vec{x}, \vec{y}$ etc. $\vec{x}$ has the $i^{th}$ coordinate $\vec{x}(i)$. For a matrix $X$, $X_i$ denotes the column $i$. The number of vertices is $n$. We will use $\lambda_i(X)$ to denote the $i^{th}$ largest singular value of the matrix $X$. Define the matrix $J = I - \frac{1}{n}11^T$. As an operator, this matrix projects a vector to the hyperplane orthogonal to the $1$ vector. Note that for any vector $\vec{z}$ such that $\sum_i \vec{z}_i = 0$, $J\vec{z} = \vec{z}$.

We assume that each of the variances $\sigma_r$ satisfies $\sigma_r^2 \geq \log^6 n/n$. We also need to assume the following:

**Separation Condition:** There exists a large enough constant $c$ such that for vertices $u \in T_r$ and $v \in T_s$, the columns $A_u$ and $A_v$ of the probability matrix $A$ corresponding to different clusters $T_r$ and $T_s$ satisfy

$$
\|A_u - A_v\|^2_2 \geq 1024c^3k^7 (\sigma_r + \sigma_s)^2 \frac{\log(n)}{w_{\text{min}}} \tag{2.1}
$$

We say that a partitioning $(S_1, \ldots, S_l)$ respects the original clustering if the vertices of each $T_r$ lie wholly in any one partition $S_j$. We will refer to the parts $S_j$ as super-clusters, being the union of one or more clusters $T_r$. The aim is to prove the following theorem.

**Theorem 5** Given $\hat{A}$ that is generated as above, i.e. $A = \mathbb{E}\hat{A}$ satisfies condition 2.1, there is a polynomial time algorithm that can find the clusters $T_1, T_2 \ldots T_k$ with
probability at least $1 - \frac{1}{n^3}$.

2.3 Related Work

The second eigenvector technique has been analyzed before, but mostly from the viewpoint of constructing cuts in the graph that have a small ratio of edges cut to vertices separated. There has been a series of results [2, 10, 22, 40] relating the gap between the first and second eigenvalues, known as the Fiedler gap, to the quality of the cut induced by the second eigenvector. Spielman and Teng [41] demonstrated that the second eigenvector partitioning heuristic is good for meshes and planar graphs. Kannan et al. [29] gave a bicriteria approximation for clustering using the second eigenvector method. Cheng et al. [11] showed how to use the second eigenvector method combined with a particular cluster objective function in order to devise a divide and merge algorithm for spectral clustering. In the random graph setting, there have been results by Alon et al. [3], and Coja-Oghlan [12] in using the coordinates of the second eigenvector in order to perform coloring, bisection and other problems. In each of these algorithms, however, the cleanup phase is very specific to the particular clustering task at hand.

Experimental studies done on the relative benefits of the two heuristics mentioned in the introduction often show that the two techniques outperform each other on different data sets [44]. In fact results by Meila et al. [44] demonstrate that the recursive methods using the second eigenvector are actually more stable than the multiway spectral clustering methods if the noise is high. Another paper by Zhao et al. [47] shows that recursive clustering using the second eigenvector performs better than a number of other hierarchical clustering algorithms.
2.4 Algorithm

The main part of the algorithm is the bipartitioning step. We provide an outline of this step below, while details including proofs will be furnished in Section 2.4.1.

Given the original matrix $\hat{A}$ we will create sub matrices of $\hat{A}$ by partitioning the set of rows into $2k$ parts $B_1 \ldots B_{2k}$ randomly. First we will use $\hat{B}_1$ to bipartition the columns. If $V_2 \cup V_3$ is this bipartition, then will use two new $\hat{B}_i$’s (say, $\hat{B}_2$ and $\hat{B}_3$) to bipartition $V_2$ and $V_3$ respectively, and we continue in this fashion. To generate $k$ clusters, we will need no more than $2k$ matrices $B_i$. Hence we have reduced our problem to the following case: We are given $\hat{B}_j$ and a vertex set $V_j$ which is a union of two or more of the original clusters. That is, $V_j = \cup_{r \in I} T_r$ where $I$ is some index set. Our goal is produce a bipartitioning of $V_j = V'_1 \cup V'_2$ such that $V'_1$ and $V'_2$ are non-empty, disjoint, and each is a union of one or more of the original clusters $T_r$.

To this end, given such a sub-matrix $\hat{B} = \hat{B}_j$ and a vertex set $V_j$, we first discard all columns that don’t correspond to vertices in $V_j$. Let’s continue to call the resulting matrix $\hat{B}$ for convenience. We will further subdivide $B$ into $q = \frac{8ck \log n}{\mu_{\text{min}}}$ random submatrices. Suppose these submatrices are $\hat{C}_i$ for $i = 1$ to $q$. We will first find the top right singular vector $\vec{u}$ of the matrix $\hat{C}_iJ$, using any standard linear algebraic algorithm. The coordinates of this vector will induce a mapping from the columns (vertices) of $\hat{C}_i$ to the real numbers. We will find a large “gap” such that substantial numbers of vertices are mapped to both sides of the gap. This gives us a natural bipartition of the set of vertices of $\hat{C}_i$. We will prove that this classifies all vertices correctly, except possibly a small fraction. This will be shown in Lemmas 7 to 11. We next need to “clean up” this bipartitioning, and this will be done using a correlation graph construction along with a Chernoff bound. The algorithm and a proof will be furnished in Lemma 13.
Finally, in Section 2.4.2 we will describe how to apply the bipartitioning algorithm recursively to learn all the $k$ clusters.

### 2.4.1 Proof

Recall that each $\hat{C}^i$ is a $\frac{n_r w_{\text{min}}}{16k^2 \log n} \times n$ matrix where the set of rows are chosen randomly, and only columns corresponding to some $V_j$ are selected (where $V_j$ is the union of a number of clusters). In what follows, we will use $\hat{C} = \hat{C}^n$ for notational simplicity, since the results proven from Fact 6 to Lemma 11 will hold for each $\hat{C}^i$ with high probability. Denote the expectation of $\hat{C}$ by $E(\hat{C}) = C$, which consists of the entries of $A$ corresponding to the rows and columns chosen to be in $\hat{C}$. By $\vec{u}$ we denote the top right singular vector of $\hat{C}J$, i.e. the top eigenvector of $(\hat{C}J)^T \hat{C}J$. In what follows, we demonstrate that for each of random sub matrices $\hat{C}$, we can utilize the second right singular vector $\vec{u}$ to create a partitioning of the columns of $\hat{C}J$ that respects the original clustering, apart from some errors.

The following fact is intuitive.

**Fact 6** With probability $1 - \frac{1}{n^r}$, $\hat{C}$ has at least $\frac{n_r w_{\text{min}}}{24k^2 \log n}$ rows for each cluster $T_r \in V_j$

**Proof** The expected number of rows from $T_r$ is $\frac{n_r w_{\text{min}}}{16k^2 \log n}$. Using Chernoff bound (Theorem 3), the number of rows contributed by each cluster $T_r \in V_j$ to the matrix $\hat{C}$ is at least $\frac{n_r w_{\text{min}}}{24k^2 \log n}$ with probability $\geq 1 - \frac{1}{n^r}$ (as long as $n_r \geq \frac{64c^2k^2 \log^2 n}{w_{\text{min}}}$).

Let $\sigma = \max_r \{\sigma_r\}$, where the maximum is taken only over clusters present in $V_j$ (and therefore, potentially smaller than $\sigma_{\text{max}}$). The following result was originally proved by Furedi-Komlos [24], but there were some problems in the proof. Vu later provided an accurate proof with better bounds [45].

**Lemma 7 (Furedi, Komlos; Vu)** If $\hat{X}$ is a 0/1 random matrix with expectation $X = E\hat{X}$, and the maximum variance of the entries of $\hat{X}$ is $\sigma^2$ which satisfies $\sigma^2 \geq \frac{n_r w_{\text{min}}}{24k^2 \log n}$.
log^6 n/n then with probability 1 – o(1),

\[ \|X - \hat{X}\|_2 < 3\sigma\sqrt{n} \]

In particular, we have \( \|C - \hat{C}\|_2 < 3\sigma\sqrt{n} \).

The following lemmas will show that the top right singular vector \( \vec{u} \) of \( \hat{C}J \) gives us an approximately good bipartition.

**Lemma 8** The largest singular value of the expected matrix \( CJ \) satisfies \( \lambda_1(CJ) \geq 2c(\sigma_r + \sigma_s)k^{2.5}\sqrt{n}\frac{1}{w_{\min}} \) for each pair of clusters \( r \) and \( s \) that belong to \( C \). Thus, in particular, \( \lambda_1(CJ) \geq 2c\sigma k^{2.5}\sqrt{n}\frac{1}{w_{\min}} \).

**Proof** Let’s consider clusters \( T_r \) and \( T_s \), \( r \neq s \). Assume \( n_r \leq n_s \). Consider the vector \( \vec{z} \) defined as:

\[
\vec{z}_x = \begin{cases} 
\frac{1}{\sqrt{2n_r}} & \text{if } x \in T_r \\
-\sqrt{\frac{n_r}{n_s}} & \text{if } x \in T_s \\
0 & \text{otherwise}
\end{cases}
\]

Now, \( \sum_x \vec{z}(x) = \frac{n_r}{\sqrt{2n_r}} - \frac{n_r}{\sqrt{2n_s}}n_s = 0 \). So, \( J\vec{z} = \vec{z} \). Also, \( \|\vec{z}\|^2 = \frac{n_r}{2n_r} + \frac{n_rn_s}{2n_s^2} = \frac{1}{2} + \frac{1}{2} \leq 1 \). Clearly, \( \|\vec{z}\| \leq 1 \). Consider a vertex \( v \in T_t \) and \( C_v \) be the corresponding row in \( C \). Then \( C_v \cdot \vec{z} = \sqrt{\frac{n_r}{2}}(P_{rt} - P_{nt}) \). We also know from Fact 6 that there are
at least \( n_t w_{\text{min}} / (24ck^2 \log n) \) such rows. Now,

\[
\|CJ\tilde{z}\|^2 = \sum_v (C_v \cdot \tilde{z})^2 = \sum_t \sum_{j \in T_t} (C_v \cdot \tilde{z})^2
\]

\[
\geq \sum_t \frac{n_t}{2k \log n} \frac{n_r}{2} (P_{rt} - P_{st})^2
\]

\[
= \frac{n_r}{4k \log n} \sum_t n_t(P_{rt} - P_{st})^2
\]

\[
\geq 1024 \frac{n_r w_{\text{min}}}{24k^2 c \log n} c^3 k^7 (\sigma_r + \sigma_s)^2 \log (n) / w_{\text{min}}^6
\]

\[
\geq 16c^2 n k^5 (\sigma_r + \sigma_s)^2 \frac{1}{w_{\text{min}}^4}
\]

using the separation condition and the fact that \( n_r \) is at least \( w_{\text{min}} n \). Thus \( \lambda_1(CJ) \geq 4c(\sigma_r + \sigma_s)k^{2.5} \sqrt{n} \frac{1}{w_{\text{min}}^2} \).

We can now show that the norm of the matrix \( \tilde{C}J \) is large.

**Lemma 9** The top singular value of \( \tilde{C}J \) is at least \( \frac{c^2 k^2 \sqrt{n}}{w_{\text{min}}} \).

**Proof** Suppose \( \tilde{z} \) is defined as in Lemma 8. Then, \( J\tilde{z} = \tilde{z} \). Now,

\[
\lambda_1(\tilde{C}J) \geq \|\tilde{C}J\tilde{z}\| = \|(C - (C - \tilde{C}))J\tilde{z}\|
\]

\[
\geq \|CJ\tilde{z}\| - \|(C - \tilde{C})\tilde{z}\| \geq \|CJ\tilde{z}\| - \|C - \tilde{C}\|\|\tilde{z}\| \geq \|CJ\tilde{z}\| - \|C - \tilde{C}\|
\]

For the last step note that \( \|\tilde{z}\| \leq 1 \). Applying the results of Lemma 7 and Lemma 8, we have that \( \lambda_1(\tilde{C}J) \geq \frac{c^2 k^2 \sqrt{n}}{w_{\text{min}}} \).

**Lemma 10** Consider the vector \( \tilde{u} \), the top right singular vector of \( \tilde{C}J \). Let \( \tilde{v} \) be the projection of \( \tilde{u} \) on to the span of the indicator vectors \( \{\tilde{I}^{(r)}\} \) for clusters \( T_r \in V_j \). As
\[ \| \vec{I}^{(r)} \| = \sqrt{n_r}, \]
\[
\vec{v} = \sum_{r \in V_j} \frac{1}{n_r} (\vec{I}^{(r)} \cdot \vec{w}) \vec{I}^{(r)}
\]

And let, \( \vec{v}^\perp = \vec{u} - \vec{v} \). We claim, \( \vec{v}^\perp \) sums to zero on each \( T_r \), and

\[
\| \vec{v}^\perp \| \leq \frac{4w_{\min}^2}{ck^{2.5}} \tag{2.2}
\]

**Proof**  
By the definition of the vectors,

\[
\| \vec{v} \|^2 + \| \vec{v}^\perp \|^2 = \| \vec{u} \|^2 = 1
\]

Also, \( \vec{v}^\perp \) is orthogonal to each \( \vec{I}^{(r)} \), and that implies that \( \sum_{x \in T_r} \vec{v}^\perp(x) = 0 \) on every cluster \( T_r \in V_j \). Now,

\[
\lambda_1(\hat{C}J) = \| \hat{C}J\vec{u} \|
\]
\[
\leq \| \hat{C}J\vec{v} \| + \| \hat{C}J\vec{v}^\perp \|
\]
\[
\leq \lambda_1(\hat{C}J)\| \vec{v} \| + \| CJ\vec{v}^\perp \| + \| CJ - \hat{C}J\|\| \vec{v}^\perp \|
\]
\[
\leq \lambda_1(\hat{C}J)(1 - \| \vec{v}^\perp \|^2/2) + \| C - \hat{C}\|\| \vec{v}^\perp \|
\]

using the fact that \( (1 - x)^{1/2} \leq 1 - \frac{x}{2} \) for \( 0 \leq x \leq 1 \), and also noting that \( J\vec{v}^\perp = \vec{v}^\perp \), and therefore \( CJ\vec{v}^\perp = C\vec{v}^\perp = 0 \). Thus, from the above, \( \| \vec{v}^\perp \| \leq \frac{2\| C - \hat{C} \|}{\lambda_1(CJ)} \leq \frac{4w_{\min}^2}{\cos k^{2.5} n} \leq \frac{4w_{\min}^2}{ck^{2.5}} \) using Lemma 2 and Lemma 9.

We now show that in bipartitioning each \( \hat{C} \) using the vector \( \vec{u} \), we only make mistakes for a small fraction of the columns.

**Lemma 11**  
*Given the top right singular vector \( \vec{u} \) of \( \hat{C} \), there is a polynomial time algorithm that bipartitions the columns of \( \hat{C} \) based on \( \vec{u} \), such that all but \( \frac{40\min w_{\min}}{ck} \) columns respect the underlying clustering of the probability matrix \( C \).*
Proof Consider the following algorithm.

1. Find $\gamma$ such that at most $\frac{n_{\text{min}}w_{\text{min}}}{ck}$ of the entries of $\vec{u}$ lie in $(\gamma, \gamma + \frac{\sqrt{w_{\text{min}}}}{k^2\sqrt{n}})$, and that if $L = \{ x : \vec{u}(x) < \gamma + \frac{\sqrt{w_{\text{min}}}}{k^2\sqrt{n}} \}; R = \{ x : \vec{u}(x) \geq \gamma + \frac{\sqrt{w_{\text{min}}}}{k^2\sqrt{n}} \}$, then both $|L|$ and $|R|$ are at least $n_{\text{min}}/2$. Note that $\hat{C} = L \cup R$.

2. If we cannot find any such gap, don’t proceed (a cluster has been found that can’t be partitioned further). Otherwise, take $L \cup R$ as the bipartition.

The goal is to show that this algorithm returns a bipartition with the required properties.

We can write $v$ as

$$\vec{v} = \sum_{T, v_j} \frac{1}{n_r} (\vec{I}(r) \cdot \vec{u}) \vec{I}(r) = \sum_r \beta_r \vec{I}(r) \tag{2.3}$$

where we define $\beta_r = \frac{1}{n_r} (\vec{I}(r) \cdot \vec{u})$

We would like to show that there is an interval of $\Theta \left( \sqrt{w_{\text{min}}} \frac{1}{k^2\sqrt{n}} \right)$ on the real line such that no $\beta_r$ lies in this interval and at least one $\beta_r$ lies on each side of the interval. We will call such a gap a “proper gap”. Note that if a proper gap exists, $\vec{v}$ will partition the set of vertices into two parts such that there are at least $n_{\text{min}}$ vertices on each side of it.

Recall that $1 = \|\vec{u}\|^2 = \|\vec{v}\|^2 + \|\vec{v}^\perp\|^2$. This and Lemma 10 (Equation (2.2)) gives us

$$\|v\|^2 \geq 1 - \frac{16}{c^2k} \geq \frac{1}{2}$$

$$\Rightarrow \sum_{r=1}^k \beta_r^2 n_r \geq \frac{1}{2} \tag{2.5}$$
As \( \vec{v} \) is orthogonal to \( \mathbf{1} \)

\[
\sum_r n_r \beta_r = 0 \quad (2.6)
\]

Squaring Eqn 2.6

\[
(\sum n_r \beta_r)^2 = 0
\]

\[
\Rightarrow \sum r n_r^2 \beta_r^2 + 2 \sum_{r,s} n_r n_s \beta_r \beta_s = 0
\]

\[
\Rightarrow 2 \sum_{r,s} n_r n_s \beta_r \beta_s = -\sum_r n_r^2 \beta_r^2 \leq -n_{\min} \sum_r n_r \beta_r^2 \leq -\frac{n_{\min}}{2} \quad (2.7)
\]

using (2.5).

Eqn (2.7) implies that there exists \( r, s \) such that

\[
n_r n_s \beta_r \beta_s \leq -\frac{1}{4k^2 n_{\min}}
\]

\[
=> \beta_r \beta_s \leq -\frac{w_{\min}}{2k^2 n}
\]

Now,

\[
(\beta_r - \beta_s)^2 \geq \beta_r^2 + \beta_s^2 - 2 \beta_r \beta_s \geq \frac{1}{k^2 n} w_{\min}
\]

\[
=> |\beta_r - \beta_s| \geq \sqrt{\frac{w_{\min}}{k n}}
\]

(2.8)

(2.9)

Since there can be at most \( k - 2 \) more \( \beta \)'s in this interval, there exists \( q, t \) such that

\[
\beta_q - \beta_t \geq \frac{\sqrt{w_{\min}}}{k^2 \sqrt{n}}
\]

(2.10)
and $\beta_s \leq \beta_t$ or $\beta_s \geq \beta_q$ for all $s \neq q,t$. From equations (2.10) and (2.3), it can now be seen that there will be a proper gap of $\sqrt{\frac{w_{\min}}{k^2 \sqrt{n}}}$ in the vector $\vec{v}$.

Let there be $m$ numbers in $\vec{v}^\perp$ that are larger than $\sqrt{\frac{w_{\min}}{10k^2 \sqrt{n}}}$. We would like to show that $m$ is small. Now,

$$\left(\frac{\sqrt{w_{\min}}}{10k^2 \sqrt{n}}\right)^2 m \leq \|\vec{v}^\perp\|^2 \leq \frac{16w_{\min}^3}{c^2 k^5} \Rightarrow m \leq \frac{w_{\min}^2 n}{36ck} = \frac{n_{\min}w_{\min}}{36ck}$$

for large enough $c$. This shows that for most vertices $\vec{v}_1(x)$ is small and will not “spoil” the proper gap in $\vec{v}$. Thus, with probability $1 - o(1)$, the above algorithm for finding a gap in $\vec{u}$ succeeds.

We also have to show that any such gap that is found from $\vec{u}$ actually corresponds to a proper gap in $\vec{v}$. Since there must be at least $n_{\min}/2$ vertices on each side of the gap in $\vec{u}$, and since the values $\vec{u}(x)$ and $\vec{v}(x)$ are close except for $\frac{n_{\min}}{ck}$ vertices, it follows that a gap found in $\vec{u}$ must correspond to a proper gap in $\vec{v}$. Thus the only vertices that can be misclassified using this bipartition are the vertices that are either in the gap, or have $\vec{v}_1(x)$ larger than $\sqrt{\frac{w_{\min}}{k^2 \sqrt{n}}}$. Given this claim, it can be seen that using a proper gap, a bipartition of the vertices can be found with at most $(\frac{n_{\min}w_{\min}}{36ck})$ vertices on the wrong side of the gap.

We can derive the following corollary:

**Corollary 12** Let a bipartitioning of $\hat{C}$ has been computed using the algorithm described in the proof of Lemma 11. Then the probability of a vertex $v$ being misclassified if no more than $\frac{w_{\min}}{36ck}$.

**Proof** Assume $v$ is the vertex for which the misclassification probability is highest. For a vertex $v$ to be misclassified, the entry of $\vec{v}_\perp$ for $v$ has to be larger than
In Lemma 11, it was shown that this can happen for no more than \( \left( \frac{w_{\min} n}{16ck} \right) \) vertices. However, the probability of \( v \) being misclassified is identical to that of all other vertices of the same cluster, which has size at least \( n_{\min} \) (as that is the minimum cluster size, and all vertices of the same cluster have the same probability distributions). The statement of the corollary follows.

Now we have a bipartition of the columns of \( \hat{C} \) that is almost correct. However, \( \hat{C} \) is just one of the \( \frac{8ck \log n}{w_{\min}} \) matrices \( C^i \). Hence, we have \( \frac{8ck \log n}{w_{\min}} \) bipartitions of the same columns using matrices \( C^i \), each of which has entries independent of others. One plan to use these different partitions would be to claim that since the misclassification probability is small (Corollary 12), every vertex will be on the right side of the partition most of the time. This is true, but each \( C^i \) doesn’t necessarily induce the same bipartition of the clusters. For instance, if there are \( k' \) clusters in the \( \hat{C} \), then we could split the vertices in \( \binom{k'}{1} + \binom{k'}{2} + \ldots + \binom{k'}{k'/2} \approx 2^{k'-1} \) ways. This means a naive approach will need to account for all possible bipartitionings and hence require an extra factor of \( 2^{k-1} \) in the separation condition. The following lemma deals with this problem:

**Lemma 13** Suppose we are given set \( V_j \) that is the union of a number of clusters \( \cup_{r \in I} T_r \) (for some index set \( I \)). Given \( p = \frac{24k \log n}{w_{\min}} \) independent bipartitions of the set of columns \( V \), such that in each bipartition, a vertex has misclassification probability no more than \( \frac{w_{\min}}{4ck} \), there exists a polynomial time algorithm that, with high probability, will compute a partitioning of the set \( V_j \) such that

- The partitioning respects the underlying clusters of the set \( V_j \).
- The partitioning is non-trivial, that is, if the set \( V_j \) contains at least two clusters, then the algorithm finds at least two partitions.

**Proof** Consider the following algorithm. Denote \( \varepsilon = \frac{w_{\min}}{9ck} \).
1. Construct a (correlation) graph $H$ over the vertex set $V_j$ using this rule: two vertices $x, y \in V_j$ are adjacent if they are in the same $L$ or $R$ for at least $(1 - 2\varepsilon)$ fraction of the bipartitions.

2. Let $N_1, \ldots, N_t$ be the connected components of this graph. Return $N_1, \ldots, N_t$.

We now need to prove that the following claims hold with high probability:

1. $N_j$ respects the cluster boundaries, i.e. each cluster $T_r$ that is present in $V_j$ satisfies $T_r \subseteq N_{j_r}$ for some $j_r$; and

2. If there are at least two clusters present in $V$, i.e. $t \geq 2$, then there are at least two components in $H$.

For two vertices $x, y \in H$, let the support $s(x, y)$ be the fraction of tests such that $x$ and $y$ are on the same side of the bipartition.

By the choice of $\varepsilon$, the probability that $x$ is misclassified in the $i^{th}$ test is at most $\frac{\varepsilon}{4}$. The expected times that a vertex $x$ is misclassified is at most $\frac{\varepsilon q}{4}$. Supposing $Y_x^i$ is the indicator random variable for the vertex $x$ being misclassified in the $i^{th}$ test. Thus, $\Pr \left[ \sum_i Y_x^i > \varepsilon q \right] < \exp \left( -4 \log n \right) = \frac{1}{n^4}$ since $q = \frac{24ck\log n}{w_{\min}}$. For each pair of vertices in a cluster, they are on the same side of the bipartition for at least $(1 - 2\varepsilon)$ fraction of the tests. Clearly, the components $N_j$ always obey the cluster partitions.

Next, we have to prove the second claim. For contradiction, assume there is only one connected component. We know that if $x, y \in T_r$ for some $r$, the fraction of tests on which they landed on same side of partition is $s(x, y) \geq (1 - 2\varepsilon)$. Hence the subgraph induced by each $T_r$ is complete. With at most $k$ clusters in $V$, this means that any two vertices $x, y$ (not necessarily in the same cluster) are separated by a path of length at most $2k$. Clearly $s(x, y) \geq (1 - 4k\varepsilon)$. Hence, the total support of
inter-cluster vertex pairs is

\[
\sum_{r \neq s} \sum_{x \in T_r, y \in T_s} s(x, y) \\
\geq (1 - 4k\varepsilon) \sum_{r \neq s} n_r n_s \\
\geq \sum_{r \neq s} n_r n_s - 4k\varepsilon \sum_{r \neq s} n_r n_s. \quad (2.11)
\]

Let us count this same quantity by another method. From Lemma 11, it is clear that for each test at least one cluster was separated from the rest (apart from small errors). That is, at least \(n_{\text{min}}(1 - \varepsilon)\) vertices were separated from the rest. Hence the total support is

\[
\sum_{r \neq s} \sum_{x \in T_r, y \in T_s} s(x, y) \\
\leq \sum_{r \neq s} n_r n_s - n_{\text{min}}(1 - \varepsilon)(n - n_{\text{min}}(1 - \varepsilon)) \\
< \sum_{r \neq s} n_r n_s - n_{\text{min}} n/2
\]

But this contradicts equation 2.11 if \(4k\varepsilon \sum_{r \neq s} n_r n_s < n_{\text{min}} n/2\) i.e. \(\varepsilon < \frac{n_{\text{min}} n/8}{k \sum_{r \neq s} n_r n_s} < \frac{n_{\text{min}} n/2}{kn^2} \leq \frac{w_{\text{min}}}{8ck}\). With the choice of \(\varepsilon = w_{\text{min}}/(9ck)\), we get a contradiction. Hence the graph \(H\) satisfies the properties claimed.

\[\square\]

2.4.2 Final Algorithm

We now describe the complete algorithm. Basically, it is the bipartitioning technique presented in the previous section repeated (at most) \(2k\) times applied to the matrix \(\hat{A}\).

As the split in every level is “clean”, as we have shown above, the whole analysis
goes through for recursive steps without any problems. In order to de-condition the steps of the recursion, we have to first create $2k$ independent instances of the data by partitioning the rows of the matrix $\hat{A}$ into a $2k$ equally sized randomly chosen sets. This creates a collection of rectangular matrices $\hat{B}_1, \ldots, \hat{B}_k$. The module $\text{Bipartition}(\hat{X}, k)$ consists of two phases: an approximate partitioning by the singular vector ($\text{Split}$), followed by a clean-up phase.

**Algorithm 1** Cluster ($\hat{A}, k$)

Partition the set of rows into $k$ random equal parts, each part to be used in the corresponding step of recursion. Name the $i^{th}$ part to be $\hat{B}_i$.

Let $(S_1, \ldots, S_l) = \text{Bipartition} (\hat{B}_1, k)$.

Recursively call $\text{Bipartition}$ on each of $S_i$, and on each of the results, using the appropriate columns of a separate $\hat{B}_j$ for each call. The recursion ends when the current call returns only one $S_i$. Let $\hat{T}_1, \ldots, \hat{T}_k$ be the final groups.

**Algorithm 2** Bipartition ($\hat{X}, k$)

Partition the set of rows into $q = \frac{8ck \log n}{w_{\min}}$ equal parts randomly. The $i^{th}$ set of rows forms the matrix $\hat{C}^{(i)}$.

For each $\hat{C}^{(i)}$, find the right singular vector of $\hat{C}^{(i)}J$ and call it $\vec{u}_i$.

call $\text{Split}$

call $\text{Clean up}$
Algorithm 3 Split

Find a proper gap $\beta$, such that

1. $(\beta, \beta + \frac{\sqrt{w_{\min}}}{k^2 \sqrt{n}})$ has at most \( \frac{n_{\min} w_{\min}}{ck} \) vertices

2. Define

\[
L_i = \{ x : u_i(x) < \beta + \frac{\sqrt{w_{\min}}}{k^2 \sqrt{n}} \} \\
R_i = \{ x : u_i(x) \geq \beta + \frac{\sqrt{w_{\min}}}{k^2 \sqrt{n}} \}
\]

Then $|L_i| \geq n_{\min}/2; |R_i| \geq n_{\min}/2$

If no such gap exists, return a $\hat{C}^{(i)}$. Otherwise, return $L_i \cup R_i$

Algorithm 4 Clean up

Construct a (correlation) graph with the columns of $\hat{X}$ as the vertices. Connect two vertices $x$ and $y$ if they are on the same $L_i$ or $R_i$ for at least \( 1 - \frac{2w_{\min}}{9ck} \) times.

Let $N_1, \ldots, N_l$ be the connected components of this graph.

return $N_1, \ldots, N_l$. 
Chapter 3

Quasi-randomness and Clustering

3.1 Introduction

In this chapter, we will present results that extend the understanding of spectral algorithms for specific types of data, mainly sparse (constant-degree) graphs.

In spectral clustering results, typically one has to assume that the degree of each vertex of the graph is at least polylogarithmic in the size of the graph to get a good theoretical bound. In applications, however, data matrices (or graphs) are often much sparser. Intuitively this is clear when one considers, say, a document-term matrix where each document only contains few of the many possible terms. Another pertinent example would be partitioning of geometric data (for example, in computer vision) where given points in high dimensional space, one constructs a graph by making each point adjacent to its closest $d$ points (for some small $d$). Given this situation, it is an interesting question whether partitioning problems for sparse data can be modeled and solved.

An obvious way to handle low degree graphs would be to take the familiar $G_{n,p}$ random graph with $p = d/n$ where $d$ is the low expected degree. Unfortunately $d =$
$\Theta(\log n)$ is an important threshold for random graphs below which many properties needed for a successful clustering (e.g. tight concentration around expected degree) no longer hold. For a clustering application, it is possible that there is no longer a well defined solution.

We provide a model that uses the concept of pseudo-randomness or quasi-randomness to successfully encapsulate sparse graphs and sparse matrices. A quasi-random graph can be roughly defined as a graph where the number (or weight) of edges between every two subsets of vertices is close to the expected number of edges among those two subsets in a random graph.

The main technical difference from earlier results is the removal of randomness (from both the model and the algorithm). We use a recent bound on norm of quasi-random matrices proved in the theory of expanders [9] which allows us to show that a spectral algorithm provides an approximate clustering. Fixing the remaining error, the “clean-up phase” is done through a iterative procedure which doesn’t use randomness (thus different from [15, 33] etc). Our proof of convergence of this process provides a theoretical explanation for similar heuristics used in practice (see [7] and references therein).

Previous work on sparse clustering include a model for bi-partitioning random regular graphs, proposed in [13], but the model is much more restricted than here with its regularity condition, and an additional strong conditions on the degree of the vertices and size of the clusters is assumed to hold making the analysis much simpler to control. It should also be noted that in [41] and later papers, it was shown that a second eigenvector technique works for planar graphs and their generalizations. Those results are not directly comparable to the random graph line of research. Certain optimization problems on $G_{n,p}$ graphs with small $p$ has been handled in [3] and work following it, but those aren’t directly comparable our work.
3.2 Model

Before we introduce our model, let us define quasi-random graphs. This definition is one of a number of equivalent definitions [9, 32].

**Definition 14** A \(d\)-regular graph on \(n\) vertices is quasi-random with parameters \(d\) and \(\alpha\) (or \((d, \alpha)\)-jumbled), if for every two subsets of vertices \(A, B\)

\[
|e(A, B) - d|A||B|/n| \leq \alpha \sqrt{|A||B|}
\]

Where \(e(A, B)\) is the number of edges from \(A\) to \(B\).

Here is some notation. Let \([n] = \{1 \ldots n\}\). For a matrix \(X\) and two sets \(A, B \subset [n]\), let \(X(A, B)\) be the sub matrix of \(X\) with rows corresponding to \(A\) and columns corresponding to \(B\). We will use the shorthand \(X(A)\) to denote \(X(A, A)\). For \(A \cap B = \emptyset\) define \(e_X(A, B) = \sum_{i \in A, j \in B} X(A, B)_{ij}\), simply the sum of all entries in \(X(A, B)\) and will drop the subscript \(X\) when the matrix is clear from the context. Define \(e(A, A) = \frac{1}{2} \sum_{i \in A, j \in A} X(A)_{ij}\). These two definitions can be combined to define \(e(A, B)\) in a natural way when the sets overlap. When \(X\) is an adjacency matrix of an undirected graph, \(e(A, B)\) is simply the number of edges going from \(A\) to \(B\). We will use \(c_1, c_2\) etc as constants, and denote 2-norm and frobenius norms respectively as \(\| \cdot \|\) and \(\| \cdot \|_F\).

**Our model:** We are given a graph \(G(V, E)\) with \(|V| = n\) vertices. Let \(A\) be the adjacency matrix of the graph. There are \(k\) clusters \(T_r; r = 1 \ldots k\) such that \(T_r \cap T_s = \emptyset\) and \(\bigcup_r T_r = V\). Let, \(n_r = |T_r|, w_r = \frac{n_r}{n}, n_{\text{min}} = \min_r n_r,\) and \(w_{\text{min}} = \min w_r\). We assume that for a parameter \(\alpha\), for all pairs \(T_r, T_s\) (not necessarily different), there is \(d_{rs}\) and \(d_{rs}\) such that

1. \(d_{rs} n_r = d_{sr} n_s\)
2. \(|e(x, T_s) - d_{rs}| \leq c_1 \alpha \log d_{rs}\) for all \(x \in T_r\)

3. \(|e(x, T_r) - d_{sr}| \leq c_1 \alpha \log d_{sr}\) for all \(x \in T_s\)

4. Let \(X = A(T_r, T_s)\). Then,

\[
\left| e_X(S, R) - s r \frac{d_{rs}}{n_s} \right| \leq \alpha \sqrt{s r}
\]

for each \(S \subset T_r\) and \(R \subset T_s\), where \(s = |S|, r = |R|\).

Let \(A'\) be the \(n \times n\) matrix where \(A'(i, j) = \frac{d_{rs}}{n_s}\) if \(i \in T_r\) and \(j \in T_s\).

**Motivation and examples:** This model contains many well-known models (random graphs, random regular graphs etc). We provide here a few pertinent examples of quasi-random graphs that are different from random graphs.

- Alon provided an example (see [32]) of a quasi-random graph with high density but no triangles. This is clearly not the case for random graphs, where the number of triangles are known to be tightly concentrated around the (large) mean.

- A main ingredient of all spectral clustering papers is some kind of concentration inequality for eigenvalues of a graph. It is known (See Theorem 2.1, [32]) that concentration in eigenvalue implies quasirandomness. This means that our algorithm works for any graph (weighted/unweighted, directed/undirected) that exhibits such concentration inequalities, provided that the technical conditions hold.

**Separation Condition.** We assume the following separation condition among centers of clusters. Define the center of a cluster \(T_r\), denoted \(\mu_r\), to be \(\mu_r(x) = \frac{d_{rs}}{n_s}\) for
all $x \in T_s$. Now, we will assume, for all $r \neq s$

$$g_{rs} \equiv \|\mu_r - \mu_s\|^2 \geq 2c_1k^2\frac{\alpha^2}{w_{\min}n}\log^2 d$$  \hspace{1cm} (3.1)$$

for some constant $c_1$.

We seek to prove the following:

**Theorem 15** There is an polynomial time algorithm such that, given a graph $G(V, E)$ and $k$ that satisfies the conditions of the model described and condition 3.1, the algorithm finds the clusters $T_1 \ldots T_k$.

The main reference of comparison are results of [13, 33] where a similar model, but with random entries was investigated. Setting $\alpha$ to $\sqrt{d}$ and ignoring constants and other minor factors, the $\log^2 d$ factor in our separation bound had to be replaced by $\log n$, which is much higher for constant $d$. Also, by increasing the value of $\alpha$ we can easily describe models with less tightly knit clusters than the random case and solve the problem if the separation is also correspondingly larger. This cannot be modeled in the independent random case, as the only way to have weaker bounds will be to change the variance $\sigma^2 = \frac{d}{n}$. In other words, for fixed $d$, more clustering problems can be defined by our model.

### 3.3 Algorithm

#### 3.3.1 A Spectral algorithm

In this part we describe a SVD (singular value decomposition) based algorithm which finds an approximate clustering of the vertices.

We will use, as a part of our algorithm, an algorithm to find an approximate solution for the following problem:
**$l_2^2$ clustering problem:** Given a set of $n$ points $S = \{v_1, \ldots, v_n\}$ in $\mathbb{R}^d$ and a positive integer $k$, the problem is to find $k$ points $f_1, \ldots, f_k \in \mathbb{R}^d$ (called “centers”) so as to minimize the sum of squared distances from each point $v_i$ to its closest center. This naturally defines a partitioning of the $n$ points into $k$ clusters.

Quite efficient constant factor deterministic approximation algorithms (even PTAS’s) are available for this problem (See [19, 27, 30] and references therein). We will work with the factor 9 approximation algorithm by Kanungo et. al. [30].

Now we describe the algorithm **Cluster**($G, k$) that outputs $k$ approximate clusters.

**Algorithm 5 Cluster**($A, k$)

1. Find $A^{(k)}$, the best rank-$k$ approximation of the matrix $A$ (by computing the SVD of $A$)
2. Approximately solve the $l_2^2$ clustering problem with the following parameters
   1. The points are the columns of $A^{(k)}$
   2. $k$ is the number of centers we seek
3. Cluster all vertices by grouping each with the closest center

The analysis of this algorithm will require a recent result on norm of quasi-random graphs by Bilu and Linial [9] which takes the place of similar (but non-sparse) results used in previous works [24, 45]. First, a useful bound from [9]:

**Lemma 16** Let $X$ be an $n \times n$ real symmetric matrix such that the $l_1$ norm of each row in $X$ is at most $2d$, with all diagonal entries of $X$ at most $O(\alpha(\log(d/\alpha) + 1))$ in absolute value. Assume that for any two vectors $u, v \in \{0, 1\}^n$, with $\text{support}(u) \cap \text{support}(v) = \emptyset$:

$$\frac{|u^t X v|}{\|u\|\|v\|} \leq \alpha$$

(3.2)

Then the spectral radius of $X$ is $O(\alpha(\log(d/\alpha) + 1))$
Lemma 17 Assume $A$ and $A'$ are square symmetric matrices of dimension $n$ such that

1. $|\sum_j A_{ij}| \leq 2d; \forall i$

2. $|\sum_j A'_{ij}| \leq 2d; \forall i$

3. $0 \leq A_{ii} \leq 2\alpha \log d$, $0 \leq A'_{ii} \leq 2\alpha \log d$, $\forall i$

4. $|e_A(S, R) - e_{A'}(S, R)| \leq \alpha \sqrt{|S||R|}$ for every $S, R \subset [n]$.

Then

$$\|A - A'\|^2 \leq c_2\alpha^2 \log^2 d$$

for a fixed constant $c_2$

Proof Plug in $A - A'$ in Lemma 16 as $X$. The $l_1$ norm condition clearly holds.

Condition 3.2 boils down to

$$|u^t Av - u^t A'v| \leq \alpha\|u\|\|v\|$$

which is exactly the fourth condition in the statement of the Lemma 17. \qed

Remark This Lemma is quite close to Corollary 5.1 proved in [9] but not identical, as that result was proved in the context of $d$-regular expander graphs.

Here are a couple of useful facts.

Fact 18 For a matrix $X$ with rank $k$, we have that $\|X\|_F^2 \leq k\|X\|^2$.

Fact 19 Since $\|X\|_F^2 = \sum_i \|X_i\|^2$, we have that the number of columns $i$ such that $\|X_i\|^2$ is greater than $\|X\|_F^2/c$ is at most $c$.

The following lemma proves that $A^{(k)}$ and $A'$ are close in a specific sense.
Lemma 20  Assume $A$ is a square symmetric matrix quasi-random relative to a rank $k$ matrix $A'$. Then, $A^{(k)}$, the best rank-$k$ approximation of $A$, satisfies

$$\|A^{(k)} - A'\|^2_F \leq 8k\alpha^2 \log^2 d$$

Proof  As both $A^{(k)}$ and $A'$ are of rank $k$, $A^{(k)} - A'$ has rank at most $2k$. Then

$$\|A^{(k)} - A'\|^2_F$$
$$\leq 2k\|A^{(k)} - A'\|^2$$
$$= 2k\|A^{(k)} - A + A - A'\|^2$$
$$\leq 4k(\|A^{(k)} - A\|^2 + \|A - A'\|^2)$$
$$\leq 8k(\|A - A'\|^2) \leq 8k\alpha^2 \log^2 d$$

The second last inequality follows from the fact that $\|A^{(k)} - A\| \leq \|A - M\|$ for all rank $k$ matrices $M$ (by definition), and that $A'$ is a rank $k$ matrix.

Lemma 21  Given an instance of the clustering problem, consider any vector $y$ such that

$$\|y - \mu_r\|^2 \leq \frac{c\alpha^2 k^2 \log^2 d}{w_{\min} n}$$

for some $r$. Let, $B_s = \{x \in T_s : \|A^{(k)}(x) - y\|^2 \leq 2c\alpha^2 \log^2 d/(w_{\min} n)\}$. Then for all $s \neq r$,

$$|B_s| \leq \frac{128k\alpha^2 \log^2 d}{g_{rs}}$$
Proof From the conditions of the Lemma,

\[ \|y - \mu_s\| = \|y - \mu_r + \mu_r - \mu_s\| \geq \|\mu_r - \mu_s\| - \|y - \mu_r\| \geq \frac{1}{2} \sqrt{g_{rs}} \]

Now, for each \( x \in B_s \)

\[ \|A^{(k)}(x) - \mu_s\|^2 \geq (\|y - \mu_s\| - \|A^{(k)}(x) - y\|)^2 \geq \left(\frac{1}{4} \sqrt{g_{rs}}\right)^2 = \frac{1}{16} g_{rs} \]

By Lemma 20,

\[ \sum_{T_s} \|A^{(k)}(x) - \mu(s)\|^2 \leq 8k\alpha^2 \log^2 d \]
\[ \Rightarrow \sum_{B_s} \|A^{(k)}(x) - \mu(s)\|^2 \leq 8k\alpha^2 \log^2 d \]
\[ \Rightarrow |B_s| \frac{1}{16} g_{rs} \leq 8k\alpha^2 \log^2 d \]
\[ \Rightarrow |B_s| \leq \frac{128k\alpha^2 \log^2 d}{g_{rs}} \]

This proves the Lemma.

Lemma 21 implies, just by summing the terms:

**Corollary 22** Let \( B = \cup_{s \neq r} B_s \). Then \( |B| \leq \frac{1}{ck} n_{\min} \)

Now we prove that the clustering produced by the algorithm is approximately correct.
Lemma 23  Consider a clustering $P_1, P_2 \ldots P_k$ produced by the algorithm Cluster$(G, k)$.

We claim:

- Each $P_r$ can be identified with a unique and different $T_s$ such that $|P_r \cap T_s| \geq (1 - \frac{1}{ck}) n_s$. Without loss of generality, we shall assume $s = r$.

- We define

\[
\bar{P}_r = T_r \cap P_r \\
Q'_r = P_r - \bar{P}_r \\
Q'_{rs} = P_s \cap Q'_r
\]

Then for all $r, s \in [k]$ such that $r \neq s$

\[
q'_{rs} = |Q'_{rs}| \leq \frac{8k\alpha^2 \log^2 d}{g_{rs}} \tag{3.3}
\]

Proof  First we claim that there is a solution to the $l_2^2$ clustering problem so that the cost of the solution $C \leq 8k\alpha^2 \log^2 d$. Assign $f_r = \mu_r$ and let the cost of this solution be $C$. Let $a_r(i)$ be the vector corresponding to the vertices $A^{(k)}$

Now,

\[
C \leq \sum_r \sum_{i \in [n_r]} \|a_r(i) - \mu_r\|^2 \\
= \|A^{(k)} - A'\|^2_{F} \\
\leq 8k\alpha^2 \log^2 d
\]

By Lemma 20.

Remark  The solution calls for $f_r$ to be in the span of vectors in $A^{(k)}$, and $\mu_r$ might not be. But this simply strengthens the result.
Accordingly, Cluster\((A, k)\) gives us a solution with cost no more than \(72k\alpha^2 \log^2 d\).

We claim, for each \(P_r\) the center \(f_r\) will be such that

\[
\|\mu_r - f_r\|^2 \leq c k^2 \alpha^2 \frac{\log^2 d}{n_r}
\]

(3.4)

If this is not true for some \(r\), then the cost of the solution is at least

\[
\sum_{i \in [n_r]} \|a_r(i) - f_r\|^2 \\
\geq \sum_{i \in [n_r]} \|a_r(i) - \mu_r + f_r - \mu_r\|^2 \\
\geq \frac{1}{4} \sum_{i \in [n_r]} (\|f_r - \mu_r\|^2 - 3\|a_r(i) - \mu_r\|^2) \\
\geq \frac{1}{4} \sum_{i \in [n_r]} \|f_r - \mu_r\|^2 - 3 \sum_{i \in [n_r]} \|a_r(i) - \mu_r\|^2 \\
\geq \frac{c}{4} k^2 \alpha^2 \log^2 d - 300k\alpha^2 \log^2 d \\
\geq 63c_2k^2 \alpha^2 \log^2 d
\]

Which is a contradiction. We used the bound \(\|u + v\|^2 \geq \frac{1}{4}\|u\|^2 - 3\|v\|^2\) in line 3.

Assuming (3.4), Lemma 21 and Corollary 22 complete the proof of the Lemma.

\[\square\]

### 3.3.2 The Clean-up Phase

As is often the case in clustering algorithms, the spectral phase is followed by a "clean-up" phase where the remaining misclassified vertices are assigned to accurate clusters.

**The algorithm** The clean-up phase employs a very natural algorithm. Given a partition with most vertices classified, we will create a “profile” of every cluster by...
computing the average number of edges per vertex between every pair of clusters. Next, we will systematically check every vertex $v$ and do the following. Assume $v$ is now placed in $P_r$. Count the number of edges of $v$ to all clusters. In the event that these numbers are closer to the profile of $P_s$ for some $s \neq r$, move $v$ to $P_s$. Repeat this until no vertices can be moved anymore. The intuition behind the algorithm is that, among the misclassified vertices, there should be some whose degrees to various clusters will clearly identify it as out of place. We show that this process will converge in poly-time.

Recall that the clusters produced by the spectral algorithm are $P_1, \ldots P_k$. Let $d'_{rs} = \frac{e(P_r, P_s)}{p_r}$, where $p_r = |P_r|$. For all $r \in [k]$, define $\theta_r \in \mathbb{R}^n$ thus:

$$\theta_r(i) = \frac{d'_{rs}}{p_s}; \forall i \in P_s$$

We will use $d(v, S)$ to denote the number of edges from $v$ to $S$, and small letters for cardinality of sets, i.e. $q = |Q|$ etc.

**Algorithm 6** Clean-up($P_1 \ldots P_k$)

1: repeat  
2: for $1 \leq r \leq k$ do  
3: compute $\theta_r$  
4: end for  
5: $(P_1, \ldots P_k) = \text{Call Update}(P_1, \ldots P_k)$ [Algorithm 7]  
6: until the clusters remain unchanged

**Algorithm 7** Update($P_1, \ldots P_k$)

1: Assign $P'_1, \ldots P'_k = \emptyset$  
2: For each vector $v$  
3: Compute vector $x^*(v)$:  
4: $x(v)_i = \frac{d(v, P_s)}{p_s}$ for $i \in P_s$  
5: Let $r = \text{argmin}_s \| x(v) - \theta_s \|$  
6: Put $v$ in $P'_r$  
7: Return $(P'_1, \ldots P'_k)$
In Lemma 24 we will show that in spite of errors, we can construct an almost correct profile for degrees among clusters. Then Lemma 25 shows that among misclassified vertices, there are always some vertices with degrees closer to the profile of the correct cluster computed in Lemma 24 than the profile of the cluster they are in now. But some other vertices might get misclassified during the update phase, and Lemma 26 shows that that number is smaller than the number computed in Lemma 25. Together, these will be enough to prove our claim about the correctness of algorithm.

**Lemma 24** *In each step of the algorithm, for all pairs* \( r \neq t \in [k] \)

\[
\|\theta_r - \theta_t\|^2 > 0.75\|\mu_r - \mu_t\|^2 \geq c_1 \frac{\alpha^2 k^2 \log^2 d \min}{w_{\min} n}
\]

**Proof** First, we will prove the Lemma for the first round of execution of Algorithm Update(\(P_1 \ldots P_k\)).

As we know,

\[
\|\theta_r - \theta_t\|^2 = \sum_{s \in [k]} \left( \frac{d_r'_{rs} - d_t'_{ts}}{p_s} \right)^2 p_s
\]

\[
= \sum_{s \in [k]} \left( \frac{(d_{rs}' - d_{ts}')^2}{p_s} \right)
\]

Now,

\[
d_{rs}' = e(P_r, P_s)/p_r
\]

As \(P_r = \bar{P}_r + Q'_r\) and \(P_s = \bar{P}_s + Q'_s\)

\[
e(P_r, P_s) = e(\bar{P}_r, \bar{P}_s) + e(\bar{P}_r, Q'_s) + e(Q'_r, \bar{P}_s) + e(Q'_r, Q'_s)
\]
We claim,

\[ |c(\bar{P}_r, \bar{P}_s) - \frac{\bar{p}_rd_{rs}\bar{p}_s}{n_s}| \leq \alpha \sqrt{\bar{p}_s\bar{p}_r} \]  

(3.5)

\[ |c(\bar{P}_r, Q'_s) - \frac{\bar{p}_r d_{rs}q'_s}{n_s}| \leq 2\alpha k \sqrt{q'_s\bar{p}_r} \log d \]  

(3.6)

\[ |c(Q'_r, \bar{P}_s) - \frac{\bar{q}_r d_{rs}\bar{p}_s}{n_s}| \leq 2\alpha k \sqrt{q'_s\bar{p}_s} \log d \]  

(3.7)

\[ |c(Q'_r, Q'_s) - \frac{\bar{q}_r' d_{rs}q'_s}{n_s}| \leq 2\alpha k \sqrt{q'_r q'_s} \log d \]  

(3.8)

The first equation is true by a straightforward application of quasi-randomness property. The other three bounds are similar to each other. Let us prove equation (3.7).

For any \( u \in [k] \), by quasi-randomness,

\[ |c(Q'_u, \bar{P}_s) - \frac{q'_r d_{us}\bar{p}_s}{n_s}| \leq \alpha \sqrt{q'_r \bar{p}_s} \]

Then,

\[ |c(Q'_u, \bar{P}_s) - \frac{q'_r d_{rs}\bar{p}_s}{n_s}| \]

\[ \leq |c(Q'_u, \bar{P}_s) - \frac{q'_r d_{us}\bar{p}_s}{n_s}| + \frac{q'_r d_{rs}\bar{p}_s}{n_s} - \frac{q'_r d_{us}\bar{p}_s}{n_s} | \]

\[ \leq \alpha \sqrt{q'_r \bar{p}_s} + \left| \frac{q'_r d_{rs}\bar{p}_s}{n_s} - \frac{q'_r d_{us}\bar{p}_s}{n_s} \right| \]

\[ \leq \alpha \sqrt{q'_r \bar{p}_s} + \frac{q'_r \bar{p}_s}{n_s} (|d_{us} - d_{rs}|) \]

\[ \leq \alpha \sqrt{q'_r \bar{p}_s} + 2q'_r |d_{us} - d_{rs}| \]
But by Lemma 21, and noting that $g_{ur} \geq \frac{(d_{us} - d_{rs})^2}{n_s}$

\[
q'_{ru} \leq 128k \frac{\alpha^2 n_s \log^2 d}{(d_{us} - d_{rs})^2}
\]

\[
\Rightarrow \quad q'_{ru}|d_{us} - d_{rs}| \leq \alpha 12\sqrt{kq'_{ru}\hat{p}_s \log d}
\]

\[
\Rightarrow \quad |e(Q'_{ru}, \bar{P}_s) - \frac{q'_{ru}d_{rs}\bar{p}_s}{n_s}| \leq 12\alpha \sqrt{kq'_{ru}\hat{p}_s \log d}
\]

Summing over all $u$

\[
|e(Q'_r, \bar{P}_s) - \frac{q'_{r}\bar{p}_s}{n_s}| \leq 12\alpha k \sqrt{q'_{r}\bar{p}_s \log d}
\]

Combining the bounds of (3.5)—(3.8), we get

\[
|e(P_r, P_s) - \frac{p_r p_s d_{rs}}{n_s}| \leq 16\alpha k \sqrt{p_s \bar{p}_r \log d}
\]

\[
\Rightarrow \quad |d'_{rs} - \frac{p_s d_{rs}}{n_s}| \leq 16\alpha k \sqrt{\frac{p_s}{p_r} \log d}
\]

Hence,

\[
\|\theta_r - \theta_t\|^2
= \sum_{s \in [k]} \frac{1}{p_s} (d'_{rs} - d'_{ts})^2
\geq \sum_{s \in [k]} \frac{1}{p_s} \left( \frac{p_s}{n_s} |d_{rs} - d_{ts}| - 16\alpha k \sqrt{\frac{p_s}{p_r} + \frac{p_s}{p_t}} \log d \right)^2
\geq 0.8 \sum_{s \in [k]} \frac{1}{p_s} \left( \frac{p_s}{n_s} |d_{rs} - d_{ts}| \right)^2 - 320 \sum_{s \in [k]} \alpha^2 k^2 \left( \frac{1}{p_r} + \frac{1}{p_t} \right) \log^2 d
\geq 0.79 \sum_{s \in [k]} \frac{1}{n_s} (d_{rs} - d_{ts})^2 - 256 \sum_{s \in [k]} \alpha^2 k^2 \left( \frac{1}{p_r} + \frac{1}{p_t} \right) \log^2 d
\geq 0.79 \|\mu_r - \mu_t\|^2 - 256\alpha^2 k^2 \left( \frac{1}{p_r} + \frac{1}{p_t} \right) \log^2 d
\geq 0.75 \|\mu_r - \mu_t\|^2
\]
since \((A - B)^2 \leq 0.8A^2 - 20B^2\)

As we will see in Lemmas 25 and 26 the cluster quality only gets better with each round, hence the conditions for proof of the Lemma will continue to hold for all rounds.

**Lemma 25** Let \(Q_r = T_r - P_r\) and \(Q = \cup_r Q_r\). Then, for at least \(q/2\) vertices \(v \in Q\), if \(v \in T_r\)

\[
\|x(v) - \theta_r\| < \|x(v) - \theta_t\|
\]

for all \(t \neq r\). Here \(x(v)\) is defined as in Algorithm Update\((P_1 \ldots P_k)\). That means these vertices will get correctly classified during one round of Algorithm Update\((P_1 \ldots P_k)\).

**Proof** Since \(\|\theta_r - \theta_t\|\) is large (Lemma 24), is suffices to show that \(\|x(v) - \theta_r\|\)
is small, for a large number of vertices \(v \in Q\).

Let \(L = \{r \in [k] : q_r \geq \frac{1}{8k}q\}\). Clearly, \(\sum_{r \in L} q_r \geq \frac{7}{8} q\). Now consider any \(r \in L\).

Our goal is to show that a large fraction of \(Q_r\) have degrees that will identify it as belonging to \(T_r\). For any \(t\), let’s estimate \(e(Q_r, P_t)\)

Now

\[
e(Q_r, P_t) = e(Q_r, \bar{P}_t) + \sum_{s \neq t} e(Q_r, Q'_ts)\\
= e(Q_r, T_t) - e(Q_r, Q_t) + \sum_{s \neq t} e(Q_r, Q'_ts)
\]

By assumption about our model,

\[
|e(Q_r, T_t) - q_r d_{rt}| \leq q_r \alpha \log d
\]  \hspace{1cm} (3.9)
And by quasi-randomness,

\[
|e(Q_r, Q_t) - \frac{d_{rt}q_r q_t}{n_t}| \leq \alpha \sqrt{q_r q_t} \leq 4\alpha q_r \sqrt{k} \tag{3.10}
\]

where we use the fact that \( q_t \leq 8kq_r \), by choice of \( Q_r \).

Similarly,

\[
|e(Q_r, Q_{ts}) - \frac{q_t q_{ts} d_{rt}}{n_t}| = |e(Q_r, Q_{ts}) - \frac{q_t q_{ts} d_{rt}}{n_r}| \\
\leq |e(Q_r, Q_{ts}) - \frac{q_t q_{ts} d_{sr}}{n_r}| + \frac{q_t q_{ts}}{n_r} |d_{sr} - d_{tr}| \\
\leq \alpha q_r \sqrt{k} + \frac{q_t q_{ts}}{n_r} |d_{sr} - d_{tr}|
\]

We have already seen that (in Lemma 24):

\[
q_{ts} |d_{tr} - d_{sr}| \leq \alpha \sqrt{kq_{ts} \bar{p}_r \log d} \\
\Rightarrow \frac{q_t q_{ts}}{n_r} |d_{sr} - d_{tr}| \leq \frac{q_r}{n_r} \alpha \sqrt{kq_{ts} \bar{p}_r \log d} \\
\Rightarrow |e(Q_r, Q_{ts}) - \frac{q_t q_{ts} d_{rt}}{n_t}| \leq q_r \alpha \sqrt{\frac{1}{k} \log d} \tag{3.11}
\]

Combining equations (3.9)—(3.11)

\[
|e(Q_r, P_t) - \frac{q_r P_t d_{rt}}{n_t}| \leq 8\alpha q_r \log d \tag{3.12}
\]

From (3.12), we can show that there is a set \( B_t \subset Q_r \) such that

1. \( |B_t| \geq (1 - \frac{1}{10k})q_r \)

2. \( |e(v, P_t) - \frac{v d_{rt}}{n_t}| \leq 80\alpha k \log d \) for all \( v \in B_t \)
Then, combining over all $t$, we can find $Q_r^{\text{good}}$ of size at least $\frac{q}{10} q_r$, such that for all $v \in Q_r^{\text{good}}$ and all $t$

$$|e(v, P_t) - \frac{p_t d_{rt}}{n_t}| \leq 80\alpha k \log d$$

Converting this bound to a bound on $\|x(v) - \theta_r\|$ follows along the same lines as Lemma 24. □

**Lemma 26** During one round of Algorithm 7, less than $q/8$ vertices get misclassified (among all vertices). Here, $q_r$ is defined as in Lemma 25.

**Proof** The proof is essentially the same as that of Lemma 25. □

Now the proof of Theorem 15 is straightforward. More than a net of $q/2$ vertices get correctly classified at each round. This process will converge in logarithmic steps to the correct partition.
Chapter 4

Entrywise Bounds on Eigenvectors of Random Graphs

4.1 Introduction

Spectral graph theory has been extensively used to study properties of graphs, and the results from this theory have found many applications in algorithmic graph theory as well. A field where the study of spectral properties has been particularly useful is the study of random graphs and matrices. Starting from Wigner’s celebrated semi-circle law [46], a number of results on random matrices and random graphs have been proved (For example [24, 45]). These results are of the following flavor. Given a random matrix $A$, one shows that $\| A - \mathbb{E}A \|_2$ is small. For example, the well known result by Furedi and Komlos [24, 45] states that, if $A$ is adjacency matrix of a graph $G \in G_{n,p}$, then with probability $1 - o(1)$

$$\| A - \mathbb{E}A \|_2 \leq (2 + o(1)) \sqrt{np}$$
For random graphs, $\mathbb{E}(A)$ is (almost) a rank-1 matrix, hence the concentration bound proves that the $A$ can be well approximated (in the $\| \cdot \|_2$ norm) by a rank-1 matrix. This statement can be easily converted to a statement about the first eigenvectors $v_1(A)$ and $v_1(\mathbb{E}(A))$, i.e. we can show that $\|v_1(A) - v_1(\mathbb{E}(A))\|_2$ is small.

In this chapter, we shall study the entry wise perturbation for eigenvectors of random graphs, i.e. $\|v_1(A) - v_1(\mathbb{E}(A))\|_\infty$. Our result is not directly implied by spectral norm concentration results, though are related through the use of expansion properties in our proofs. We are unaware of earlier work on this particular property of random graphs. Perturbation of eigenvectors and eigenspaces have been classically studied for unitarily invariant norms, in particular for the spectral and Hilbert-Schmidt norms [8]. Perturbation in the $\| \cdot \|_\infty$ norm has been studied in the Markov Chain literature [36] to investigate stability of steady state distributions, however, the error model in those works do not seem to carry over to random graphs in any useful way. Other standard results on entrywise bounds for eigenvectors [34] also seem powerless in this case.

As we mentioned, the study of perturbation of spectral subspaces has traditionally focused on the $\| \cdot \|_2$ norm. This does convert to a statement about $\| \cdot \|_\infty$ norm, but it is much weaker than our bounds. Taking $G_{n,\frac{1}{2}}$ as an example, on the same scale spectral norm bounds only imply $\Omega(1)$ entrywise differences, whereas our results show that the differences are no larger than $O(\sqrt{\frac{\log n}{n}})$. Indeed, for $p = o(1)$, the spectral norm provides no bounds stronger than what the Perron-Frobenius theorem implies, while our bounds continue to be non-trivial for smaller values. Our bounds are useful for $p \geq \frac{\log^6 n}{n}$.

In Section 4.2 we present the statement of the Theorem and in Section 4.3 we present the proof of the result. The main technical tool used in that section is quasirandom properties of random graphs. We prove tightness of the bounds for
quasi-random graphs in Section 4.4.

In Section 4.5 we explore the connection of the problem of entry-wise bounds with algorithms for spectral clustering for probabilistic models. The spectral norm bounds discussed above have been successfully used for analyzing spectral heuristics for probabilistic data [4, 10, 33]. In these models, the data is generated from an underlying simple probability distribution. Starting from the fact that \( \mathbb{E}A \) is trivial to cluster, the result that \( \|A - \mathbb{E}A\|_2 \) is small is used to claim that vertices in \( A \) are well-behaved, though the non-negligible value of \( \|A - \mathbb{E}A\|_2 \) implies that some might be misclassified. To rectify this, one uses some sort of “clean-up” scheme. It has been conjectured [33] that these schemes are unnecessary, the failure to remove them stems from an incomplete understanding of the nature of spectral deviation in random graph-like structures. In the next chapter, we solve this conjecture. This chapter is involved with even a stronger version of the conjecture. In McSherry’s conjecture one is allowed to use cross-projections, i.e. use spectral information of one part of the data to cluster the other part (and the algorithm in the next chapter in fact does that). This allows one to use independence between samples. Here we are concerned with the possibility that a spectral analysis followed by a simple greedy algorithm will complete the clustering, without an appeal to independence of samples.

We make a step towards resolving these questions through entrywise deviation in the eigenvectors of random graphs with clusters. We will show that for a simple clustering problem, the second eigenvector obeys the cluster boundaries, thus no clean-up phase is necessary. We will provide two analyses for the model, resulting in different bounds, each providing better bounds than the other in certain situations (neither is as good as standard separation bounds). Though our model requires conditions stronger than the ones used in standard results for spectral clustering,
the results are non-trivial in the sense that mere eigenvalue bounds are not enough to prove them.

4.2 Model and Main Result

A(G) will be used to denote the adjacency matrix of graph G, and A when G is clear from the context. We will use \( \lambda_i(M) \) and \( v_i(M) \) to denote the \( i^{th} \) largest (in absolute value) eigenvalue and its corresponding eigenvector of \( M \). Also, \( c,c_1,\ldots \) etc are constants throughout. Norms will be subscripted to indicate which norm is being used, no subscript will mean the \( \| \cdot \|_2 \) norm. For any vector \( x \in \mathbb{R}^N \), \( x_{\max} = \max_{i \in [N]} x_i \) and \( x_{\min} = \min_{i \in [N]} x_i \). For sets \( R,S \in V \), \( e(R,S) = \sum_{i \in R, j \in S} A_{ij} \), i.e. the number of edges between \( R \) and \( S \). For any set of vertices \( B \), \( N(B) \) denotes the set of its neighbors. For a vector \( x \), let \( x(i) \) be its \( i^{th} \) entry. We will use the phrase "with high probability" to mean with probability \( 1 - o(1) \).

The following is the main result. Define \( \Delta = \sqrt{\log n / d} \).

**Theorem 27** Let \( G \in G_{n,p} \) be a random graph and \( A \) be its adjacency matrix. Assume \( p \geq \log^6 n/n \). Let \( d = np \) and \( v = \gamma v_1(A) \), where \( \gamma \) is a normalization factor such that \( v_{\max} = 1 \). Then, with high probability

\[
v_{\min} \geq 1 - c_2 \frac{\log n}{\log d} \Delta
\]  

\((4.1)\)

for some constant \( c_2 \).

Since \( v_1(\mathbb{E}A) = \{\frac{1}{\sqrt{n}} \ldots \frac{1}{\sqrt{n}}\} \), the get the following corollary:

**Corollary 28** Let \( G \in G_{n,p} \) be a random graph and \( A \) be its adjacency matrix.
Assume $p \geq \log^6 n/n$. Then, with high probability

$$\|v_1(A) - v_1(\mathbb{E}A)\|_\infty \leq c \frac{\log n}{\log d \sqrt{n}} \sqrt[3]{\log n \sqrt{np}}$$

for some constant $c$.

### 4.3 Bound

We use the following properties of a random graph $G \in G_{n,p}$ that hold with high probability

1. Each vertex has degree in the range $d \pm \sqrt{d \log n}$

2. Quasi-randomness: for all subsets $R, T \in V$

$$|e(R, T) - \frac{|R||T|d}{n}| \leq 2 \sqrt{d|R||T|}$$

See the recent survey by Krivelevich and Sudakov [32] for a reference.

The intuition behind our proof of Theorem 27 can be demonstrated by the following simple observation. Let $v$ be normalized such that $v(i) = 1$ for some $i \in V$. Now, the first eigenvalue of $A$ is close to $d$, while $i$ has a degree of $d \pm \sqrt{d \log n}$. As $A_i \cdot v \approx dv(i) = d$, we need $\sum_{j \in N(i)} v_j = d$ where $N(i)$ is $i$'s neighborhood set. This means on average, $N(i)$ will have weights in the range $1 \pm \sqrt{\log n/d}$. Our technical lemmas that follow show how this intuition can be shown to be true for not only vertices but sets, and how an absolute (not only average) result can be achieved.
Assume, \( v \) is defined as in Theorem 27, and \( v(1) = 1 \), without loss of generality.

We define a sequence of sets \( \{S_t\} \) for \( t = 1 \ldots \) in the following way:

\[
S_1 = \{1\} \tag{4.2}
\]
\[
S_{t+1} = \{i : i \in N(S_t) \text{ and } v(i) \geq 1 - c(t+1)\Delta\} \tag{4.3}
\]

Now, we define \( n_t \) and \( F_t \)

- \( n_t = |S(t)| \)
- \( F_t = \sum_{i \in S(t)} v(i) \)

Note that \( n_1 = 1 \) and \( F_1 = 1 \).

First a simple result we need about the largest eigenvalue

**Lemma 29** Let \( \lambda \) be the first eigenvalue of \( A \). Then whp, \( \lambda \geq d - \sqrt{p \log n} \) and \( \lambda \leq d + \sqrt{d \log n} \)

**Proof** \( \mathbb{E}e(V,V) = nd \). By Chernoff bounds,

\[
\mathbb{P}(e(V,V) \leq nd - \sqrt{nd \log n}) \leq O \left( \frac{1}{n} \right)
\]

Now, \( \lambda \geq \frac{1}{n} 1^T A 1 \geq \frac{1}{n} (nd - \sqrt{nd \log n}) = d - \sqrt{\frac{d}{n} \log n} = d - \sqrt{p \log n} \). The upper bound is similar.

**Lemma 30** Let \( t' \) be the last index such that \( n_{t'} \leq \frac{60n}{d} \). For all \( t \leq t' \)

\[
n_{t+1} \geq \frac{d n_t}{9 \log^2 n}
\]

**Proof** Let \( N = N(S(t)) \). Also \( e(S(t)) = e(S(t), N) \leq n_t (d + \sqrt{d \log n}). \)

59
Since the edges to $S(t)$ must provide the multiplicative factor of $\lambda$,

$$\lambda F_t = \sum_{i \in N} v(i) e(i, S(t))$$

Now,

$$\lambda F_t = \sum_{i \in N} v(i) e(i, S(t))$$

$$= \sum_{N - S(t+1)} v(i) e(i, S(t)) + \sum_{S(t+1)} v(i) e(i, S(t))$$

$$\leq (1 - c(t + 1)\Delta) e(N - S(t + 1), S(t))$$

$$+ e(S(t + 1), S(t))$$

$$= (1 - c(t + 1)\Delta) e(S(t))$$

$$+ c(t + 1) e(S(t), S(t + 1)) \Delta$$

$$\leq (1 - c(t + 1)\Delta) e(S(t))$$

$$+ c(t + 1) \left( \frac{nt_{t+1}d}{n} + 2\sqrt{nt_{t+1}d} \right) \Delta$$

The last line uses the quasi-randomness property. Since $\lambda \geq d - \sqrt{p \log n}$ [Lemma 29] and $F_t \geq nt(t(1 - ct\Delta)$ [by definition]

$$(1 - c(t + 1)\Delta)e(S(t)) +$$

$$c(t + 1) \left( \frac{nt_{t+1}d}{n} + 2\sqrt{nt_{t+1}d} \right) \Delta$$

$$\geq (d - \sqrt{p \log n}) F_t$$

$$\Rightarrow (1 - c(t + 1)\Delta) nt(d + \sqrt{d \log n}) + c(t + 1) \left( \frac{nt_{t+1}d}{n} +$$

$$2\sqrt{nt_{t+1}d} \right) \Delta \geq nt(1 - ct\Delta)(d - \sqrt{p \log n})$$
\[ \Rightarrow c(t + 1) \left( \frac{n_t n_{t+1} d}{n} + 2\sqrt{n_t n_{t+1} d} \right) \Delta \]
\[ \geq n_t d(1 - ct\Delta - \Delta - 1 + c(t + 1)\Delta) \]
\[ - \Delta + c(t + 1)\Delta^2 \geq n_t d(c - 2)\Delta \]
\[ \Rightarrow \frac{n_t n_{t+1} d}{n} + 2\sqrt{n_t n_{t+1} d} \geq \left( 1 - \frac{2}{c} \right) \frac{n_t d}{t + 1} \]

As \( n_t \leq \frac{60n}{d^2} \), by assumption,
\[ 60n_{t+1} + 2\sqrt{n_t n_{t+1} d} \geq \left( 1 - \frac{2}{c} \right) \frac{n_t d}{t + 1} \]

Assuming \( c \geq 10 \), either (or both) of the following is true:
\[ 60n_{t+1} \geq \frac{1}{3} \frac{n_t d}{t + 1} \] (4.4)
\[ 2\sqrt{n_t n_{t+1} d} \geq \frac{1}{3} \frac{n_t d}{t + 1} \] (4.5)

Eqn 4.5 implies
\[ n_{t+1} \geq \frac{1}{180} \frac{n_t d}{(t + 1)^2} \] (4.6)

As Eqn 4.6 is weaker than Eqn 4.4, if suffices to deal with it. Consider \( t + 1 \leq \log n \).

Then, \( n_{t+1} \geq \frac{1}{180} \frac{n_t d}{\log^2 n} \geq n_t d^{2/3} \). With this growth rate, for \( n_t \) to grow to \( \frac{60n}{d^2} \), we need \( t = \log_{d^{2/3}} n = \frac{4}{3} \frac{\log n}{\log d} \). This is consistent with our initial assumption for \( t \), hence the lemma is proven. \( \square \)

The following lemma deals with the case of large sets.

**Lemma 31** Let \( U \subset V \), where \( u = |U| \geq \frac{60n}{d^2} \). Also, assume that \( F(U) = \sum_{i \in U} v(i) \geq u(1 - \alpha \Delta) \) for some \( \alpha \geq 1 \). Let \( W(U) = \{ i : i \in N(U) \text{ and } v(i) \geq (1 - (12\alpha + 24)\Delta) \} \).
Then we claim, \( w = |W(U)| > \frac{6n}{10} \).
Proof
Assuming that the claim of the lemma is false, the number of nodes, 
\[ w \leq \frac{6n}{10}. \]

We know that \( \lambda F_t = \sum_{i \in S_{t+1}} v(i) e(i, S_t) \). Using the assumption on \( F_t \), we need
\[
\sum_{i \in S_{t+1}} v(i) e(i, S_t) \geq \lambda u (1 - \alpha \Delta) \geq du (1 - (\alpha + 1) \Delta) \tag{4.7}
\]

However, using the fact \( u \geq \frac{60n}{d} \)
\[
eq (W(U), U) \leq \frac{6ndu}{10n} + 2\sqrt{\frac{6nd}{10}} \leq \frac{4ud}{5}
\]

As \( v(i) \leq 1 \) for all \( i \) and \( v(i) \leq 1 - (12\alpha + 24) \Delta \) for edges with no end in \( W(U) \), this yields,
\[
\sum_{i \in S_{t+1}} v(i) e(i, S(t)) \leq e(W(U), U) \times 1 + (e(U, N(U)) - W(U)) \times (1 - (12\alpha + 24) \Delta) \\
\leq \frac{4ud}{5} + \left( u(d + \sqrt{d \log n}) - \frac{4ud}{5} \right) (1 - (12\alpha + 24) \Delta) \\
\leq ud + u\sqrt{d \log n} - \frac{ud}{6} (12\alpha + 24) \Delta \\
\leq ud(1 - (2\alpha + 2) \Delta)
\]

This contradicts Equation 4.7. \( \square \)

The following two lemmas follow along the same lines as Lemmas 30 and 31, respectively. For these lemmas, we assume \( v(1) = v_{\text{min}} = b > 0 \) and define \( S_t \) and \( n_t \) analogously:

- \( S_1 = \{1\} \)
\[ S_{t+1} = \{ i : i \in N(S_t) \land v(i) \leq b(1 + c(t + 1)\Delta) \} \]

And,

\[ n_t = |S(t)| \]

\[ F_t = \sum_{i \in S(t)} v(i) \]

Note that \( n_1 = 1 \) and \( F_1 = b \).

**Lemma 32** Let \( t' \) be the last index such that \( n_{t'} \leq \frac{n}{d} \). For all \( t \leq t' + 1 \)

\[ n_{t+1} \geq \frac{dn_t}{9 \log^2 n} \]

In addition, \( n_{t'+1} \geq \frac{n}{d} \).

**Proof** Let \( N = N(S(t)) \). Also \( e(S(t)) = e(S(t), N) \geq n_t(d - \sqrt{d \log n}) \).

Since the edges to \( S(t) \) must provide the multiplicative factor of \( \lambda \),

\[ \lambda F_t = \sum_{i \in N} v(i) e(i, S(t)) \]
Now,

\[
\lambda F_t = \sum_{i \in N} v(i)e(i, S(t))
\]

\[
= \sum_{N - S(t+1)} v(i)e(i, S(t)) + \sum_{S(t+1)} v(i)e(i, S(t))
\]

\[
\geq b(1 + c(t + 1)\Delta) e(N - S(t + 1), S(t)) + e(S(t + 1), S(t))b
\]

\[
= b((1 + c(t + 1)\Delta) e(S(t))
\]

\[
- c(t + 1)e(S(t), S(t + 1))\Delta)
\]

\[
\geq b((1 + c(t + 1)\Delta) e(S(t))
\]

\[
- c(t + 1)\left(\frac{nt_{t+1}d}{n} + 2\sqrt{nt_{t+1}d}\Delta\right)
\]

The last line uses the quasi-randomness property. Since \(\lambda \leq d(1 + \Delta)\) [Lemma 29] and \(F_t \geq bn_t(1 + ct\Delta)\) [by definition]

\[
b((1 + c(t + 1)\Delta) e(S(t))
\]

\[
- c(t + 1)(\frac{nt_{t+1}d}{n} + 2\sqrt{nt_{t+1}d})\Delta)
\]

\[
\leq d(1 + \Delta)F_t
\]

\[
\Rightarrow (1 + c(t + 1)\Delta) n_t(d + \sqrt{d\log n})
\]

\[
- c(t + 1)(\frac{nt_{t+1}d}{n} + 2\sqrt{nt_{t+1}d})\Delta \leq dn_t(1 + ct\Delta)(1 + \Delta)
\]
\[\Rightarrow - c(t + 1)\left(\frac{n_t n_{t+1} d}{n} + 2\sqrt{n_t n_{t+1} d}\right) \Delta \leq n_t d(1 + \Delta)\]

\[(1 + ct \Delta - 1 - c(t + 1) \Delta) \leq -dn_t c\Delta\]

\[\Rightarrow \frac{n_t n_{t+1} d}{n} + 2\sqrt{n_t n_{t+1} d} \geq \frac{n_t d}{t + 1}\]

As \(n_t \leq \frac{60n}{d}\), by assumption,

\[60n_{t+1} + 2\sqrt{n_t n_{t+1} d} \geq \frac{n_t d}{t + 1}\]

Hence either (or both) of the following is true:

\[60n_{t+1} \geq \frac{1}{3} \frac{n_t d}{t + 1} \quad (4.8)\]

\[2\sqrt{n_t n_{t+1} d} \geq \frac{1}{180} \frac{n_t d}{t + 1} \quad (4.9)\]

Eqn 4.5 implies

\[n_{t+1} \geq \frac{1}{180} \frac{n_t d}{(t + 1)^2} \quad (4.10)\]

As Eqn 4.6 is weaker than Eqn 4.4, if suffices to deal with it. Now consider \(t + 1 \leq \log n\).

Then, \(n_{t+1} \geq \frac{1}{180} \frac{n_t d}{\log^2 n} \geq n_t d^{2/3}\). With this growth rate, for \(n_t\) to grow to \(\frac{60n}{d}\), we need \(t = \log_{d^{2/3}} n = \frac{4 \log n}{3 \log d}\). This is smaller than our initial assumption for \(t\), hence the lemma is proven.

\[\Box\]

**Lemma 33** Let \(U \subset V\), where \(u = |U| \geq \frac{60n}{d}\). Also, assume that \(F(U) = \sum_{i \in U} v(i) \leq ub(1 + \alpha \Delta)\) for some \(\alpha \geq 1\). Let \(W(U) = \{i : i \in N(U) \wedge v(i) \leq b(1 + (12\alpha + 24) \Delta)\}\).

Then we claim, \(w = |W(U)| > \frac{6n}{10}\).

**Proof** Assuming that the claim of the lemma is false, the number of nodes, \(w \leq \frac{6n}{10}\).
We know that $\lambda F_t = \sum_{i \in S_{t+1}} v(i)e(i, S_t)$. Using the assumption on $F_t$, we need

$$\sum_{i \in S_{t+1}} v(i)e(i, S_t) \leq \lambda ub(1 + \alpha \Delta) \leq dub(1 + (2\alpha + 1)\Delta) \quad (4.11)$$

However, using the fact $u \geq \frac{60n}{d}$

$$e(W(U), U) \leq \frac{6ndu}{10n} + 2\sqrt{\frac{u6nd}{10}} \leq \frac{4ud}{5}$$

As $v(i) \geq b$ for all $i$ and $v(i) \geq b(1 + (10\alpha + 20)\Delta)$ for edges with no end in $W(U)$, this yields,

$$\sum_{i \in S_{t+1}} v(i)e(i, S(t))$$

$$\geq e(W(U), U) \times b + (e(U, N(U)) - W(U))$$

$$\times b(1 + (12\alpha + 24)\Delta)$$

$$\geq b\left(\frac{4ud}{5} + \left(u(d - \sqrt{d\log n}) - \frac{4ud}{5}\right)\right)$$

$$(1 + (12\alpha + 24)\Delta)$$

$$\geq b\left(ud - u\sqrt{d\log n} + \frac{ud}{6}(12\alpha + 24)\Delta\right)$$

$$\geq b(ud(1 + (2\alpha + 2)\Delta))$$

This contradicts Equation 4.11. \qed

**Proof of Theorem 27** Let us consider $S_t$ (as defined in Eqns 4.2 and 4.3) for the first $t$ such that $n_t \geq \frac{60n}{d}$. From Lemma 30, $F_t \geq n_t(1 - \frac{\log n}{\log n}\Delta)$. As $n_t \geq \frac{60n}{d}$, we can invoke Lemma 31 with $U = S_t$. This gives us a set $W$ with $|W| \geq \frac{n}{2}$, such that for every $i \in W, v(i) \geq 1 - \beta\Delta$, where $\beta = c_1\frac{\log n}{\log d}$ for some constant $c_1$.

A similar argument can be put forward using Lemmas 32 and 33. So, for another
set $Y$, where $|Y| > \frac{n}{2}$, $v(i) \leq b(1+\beta \Delta)$ for each $i \in Y$. Using the pigeonhole principle to observe that $X$ and $Y$ must intersect, we can conclude that

$$b(1+\beta \Delta) \geq 1 - \beta \Delta$$

$$\Rightarrow b \geq 1 - 3\beta \Delta$$

This completes the proof of Theorem 27.

4.4 Lower Bound

Though our result is stated for random graphs, it is essentially a result about quasi-random graphs with degree bounds. For these more general objects we will show that our bound is tight up to a constant factor. We note that this does not imply a lower bound for random graphs. We leave that as an open question.

Quasi-random graphs can be defined as follows [32]: A $(d, \alpha)$ quasi-random graph $G = (V, E)$ on $n$ nodes satisfies the property that for each subset $A, B \in V$

$$|e(R, T) - \frac{|R||T|d}{n}| \leq \alpha \sqrt{|R||T|}$$

For random graphs $\alpha = 2\sqrt{d}$.

We prove the following:

**Theorem 34** For any large enough $n$, and any $\sqrt{n} \geq d \geq \log^6 n$ there exists $(d, 27\sqrt{d})$-quasi-random graphs on $n$ nodes so that each vertex has a degree in the range $d(1 \pm 2\Delta)$ and

$$v_{\text{max}} - v_{\text{min}} \geq \frac{\log n}{\log d} \Delta$$  \hspace{1cm} (4.12)

where $v$ is the first eigenvector of the adjacency matrix of the graph.
Figure 4.1: a) A sorted plot of $v_1(A) - v_1(\mathbb{E}A)$ where $A$ was computer generated according to $G_{1000,0.1}$. b) A sorted plot of the second eigenvector of the planted clique problem, where a clique of size 50 is embedded in a 500 node graph. The graph is generated by selecting every edge with $p = \frac{1}{2}$. The largest 50 entries correspond to the clique.

Figure 4.2: Construction of a quasi-random graph depicting tightness of the bound.

**Proof**  We construct the quasi-random graph as follows:

**Construction** Starting from a single node as root, construct a $d(1 + \epsilon)$ degree complete tree $T_1$ of depth $t$, where we will define $\epsilon$ and $t$ later. Construct another complete tree $T_2$ of same depth, but with degree $d(1 - \epsilon)$. Define $L(T)$ to be the set of leaves of a tree $T$. Now let $M$ be a set of $m$ new nodes. Set $P = L(T_1) \cup L(T_2) \cup M$ and construct a $d$-regular expander on $P$ (by, for example, generating a random $d$-regular graph on them). Now $G$ is the graph with vertex set $M \cup T_1 \cup T_2$ and the edges in the two trees and the random regular part. We define $n = |V| = m + 2t$. 
Set \( t = \frac{\log n}{10 \log d} \) and \( \epsilon = \sqrt{\frac{\log n}{d}} \). We claim:

1. \( G \) is \((d, 27\sqrt{d})\)-quasi-random, has vertex degrees in the range \( d(1 \pm 2\epsilon) \).

2. Let \( v = \gamma v_1(A) \) such that \( v_{\text{max}} = 1 \). Then, \( v_{\text{min}} \leq 1 - \frac{1}{3} \frac{t \epsilon}{1+\epsilon} \).

Let us prove the second part of the claim first. Assume that \( \lambda_1 \leq d \) and that the root of \( T_1 \) is vertex 1. Let, \( u = \gamma_1 v_1(A) \) such that \( u(1) = 1 \). In Lemma 35, we show that at level \( r \) of \( T_1 \), there is a vertex \( j \) for which

\[
u(j) \leq 1 - \frac{1}{2} r \frac{\epsilon}{1+\epsilon}\]

Hence, there is a vertex \( j \) such that \( u(j) \leq 1 - \frac{1}{3} r \epsilon \). To get \( v \), we must divide \( u \) with a number larger than 1, hence \( v_{\text{min}} \leq u(j) \). This proves the claim.

Now, if \( \lambda_1 \geq d \), we can use a similar argument on \( T_2 \) and prove that if \( v_x = 1 \) (where \( x \) is the root of \( T_2 \)) then there exists a vertex \( j \) at level \( t \) of \( T_2 \) such that \( v_j \geq 1 + \frac{1}{2} t \frac{\epsilon}{1+\epsilon} \). This proves the claim, from which the tightness of our bounds follow once we plug in values of \( \epsilon \) and \( t \).

We now come to the first part. The claim about degrees is clear from the construction. Proving the quasi-randomness property is a matter of checking the property for each possible pair of vertex sets. We analyze the cases below:

**Case 1:** Let \( R, S \in P \) and define \( p = |P| \). Now as the subgraph on \( P \) is quasi-random

\[
|e(R, S) - \frac{d|R||S|}{p}| \leq 2\sqrt{d|R||S|}
\]

\[
\Rightarrow |e(R, S) - \frac{d|R||S|}{n}| \leq 2\sqrt{d|R||S|} + \frac{d|R||S||(n-p)}{np}
\]

\[
\leq 3\sqrt{d|R||S|}
\]
Case 2: Now let $X, Y \in T_1$, define $x = |X|, y = |Y|$ and assume without loss of generality that $x \leq y$.

First, we need to prove $e(X, Y) \geq \frac{d_{xy}}{n} - 2\sqrt{d_{xy}n}$. As, $x, y \leq o\left(\frac{n}{d}\right)$, $\frac{d_{xy}}{n} < 2\sqrt{d_{xy}n}$.

So, the bound is trivially true.

Next, we claim $e(X, Y) \leq \frac{d_{xy}}{n} + 2\sqrt{d_{xy}}$. We analyze two cases:

• if $x < \frac{y}{d}$. Now,

\[ e(X, Y) \leq x(d(1 + \epsilon) + 1) \leq 2\sqrt{d_x}\sqrt{d_x} \leq 2\sqrt{d_x}\sqrt{y} \leq 2\sqrt{d_{xy}} \]

We use the assumption on $x$ in the last inequality.

• if $x \geq \frac{y}{d}$. As $T_1$ is a tree, $e(X, Y) \leq x + y \leq 2y$ Now,

\[ e(X, Y) \leq 2y = 2\sqrt{y}\sqrt{y} \leq 2\sqrt{y}\sqrt{d_x} \leq 2\sqrt{d_{xy}} \]

Case 3: Let $X \in P$ and $Y \in T_1$. First, we need to prove $e(X, Y) \geq \frac{d_{xy}}{n} - 2\sqrt{d_{xy}n}$.

As, $y \leq o\left(\frac{n}{d}\right)$, $\frac{d_{xy}}{n} < 2\sqrt{d_{xy}n}$. So, the bound is trivially true.

For the other case, note that the only edges from $T_1$ to $P$ will involve $L(T_1)$, hence the same arguments as the previous case will suffice to prove the claim.

Similar bounds will work for sets involving $T_2$. 70
Now for any two sets $S, R$,

$$e(S, T) - \frac{d_{sr}}{n} = \sum_{i,j \in \{1,2,3\}} e(S_i, R_j) - \frac{d}{n} \left( \sum_{i,j \in \{1,2,3\}} s_i r_j \right)$$

$$\leq \sum_{i,j \in \{1,2,3\}} |e(S_i, R_j) - \frac{d}{n} s_i r_j|$$

$$\leq \sum_{i,j \in \{1,2,3\}} |e(S_i, R_j) - s_i r_j|$$

$$\leq \sum_{i,j \in \{1,2,3\}} 3\sqrt{d_{sr} r_j}$$

$$\leq 27\sqrt{d_{sr}}$$

Hence the bound.

**Lemma 35** Consider a graph constructed as in Theorem 34. Assume that $\lambda_1 \leq d$ and that the root of $T_1$ is vertex 1. Let, $u = \gamma_1 v_1(A)$ such that $u(1) = 1$. Then, at level $r \leq t$ of $T_1$, there is a vertex $j$ for which

$$u(j) \leq 1 - \frac{1}{2} r \frac{\epsilon}{1 + \epsilon}$$

**Proof** We prove the bound inductively. The claim is trivially true at level 0. Assume the hypothesis is true at level $r < t$, and that $u(j) \leq 1 - \frac{1}{2} r \frac{\epsilon}{1 + \epsilon}$. Now,

$$\sum_{i \in N(j)} u(i) \leq du_j$$

$$\Rightarrow \sum_{i \in N(j) - \{j\}} u(i) \leq (d - 1)u_j$$
Since \(|N(j) - \{p\}| = d(1 + \epsilon)\), by a basic averaging argument, for some \(k \in N(i)\),

\[
u(k) \leq \frac{(d - 1)u(j)}{d(1 + \epsilon)} \leq (1 - \frac{1}{2} \frac{r\epsilon}{1 + \epsilon})(1 - \frac{1}{d})(1 - \epsilon + \epsilon^2) \leq (1 - \frac{1}{2} \frac{r\epsilon}{1 + \epsilon})(1 - \epsilon) \leq 1 - \frac{1}{2} \frac{r\epsilon}{1 + \epsilon} - \epsilon + \frac{r\epsilon^2}{1 + \epsilon} \leq 1 - \frac{1}{2}(r + 1)\frac{\epsilon}{1 + \epsilon}
\]

Provided \(\epsilon \geq r\epsilon^2\) which is true for our construction.

### 4.5 Application to Clustering

In this section we present two analyses of a spectral algorithm for clustering using our approach. We will show that a very simple algorithm (Algorithm 8) manages to bi-parition a randomly generated graph matrix with two clusters provided that the cluster means are separated. The separation conditions assumed are much stronger than required for a standard spectral clustering result. Nevertheless, they are both interesting in the sense that eigenvalue bounds alone are insufficient to prove the results we claim.

We analyze the following natural algorithm:

#### 4.5.1 Model

The input to the algorithm is a graph \(G(V, E)\), which has two clusters \(T_1\) and \(T_2\) such that \(T_1 \cup T_2 = V\). Assume \(|T_a| = n\) for \(a = 1, 2\). The adjacency matrix \(A\) is generated thus: If \(r \in T_a\) and \(s \in T_b\) then \(A_{rs} = A_{sr} = 1\) with probability \(p_{ab}\) and 0
**Algorithm 8** Threshold$(A, m)$

1: \( A \) is the adjacency matrix, \( m \) is the number of vertices

2: Find \( v = v_2(A) \), the second eigenvector of \( A \)

3: Find the largest “gap” in \( v \), i.e.
   
   \[
   \text{The largest } \delta \text{ such that } \quad \forall i, v(i) \leq \gamma - \delta \text{ or } v(i) \geq \gamma + \delta, \text{ for some } \gamma
   \]

4: Let \( L = \{ i : v(i) \leq \gamma - \delta \} \)

5: Return \( L \) and \( [n] - L \)

otherwise.

If \( x \in T_a \), \( d_1 = E d(x, T_a) \), \( d_2 = E d(x, T_b \neq a) \) and \( d = \max d_a = d_1 \) then we assume that the following *separation condition* holds,

\[
    d_1 - d_2 = g \tag{4.13}
\]

where we will specify \( g \) later.

In both cases, we prove some version of the following:

**Theorem 36** *Algorithm 8 succeeds in correctly bi-partitioning the graph.*

### 4.5.2 Related work

Clustering problems on probabilistic models have a long history (see references in [33], for example). Algorithms based on the spectrum of graphs have been considered for both discrete and continuous models by a number of papers [1, 14, 28, 33]. A major part in all these papers involves dealing with the “error” \( \|A - EA\| \). One of the most common techniques employed is to partition the data into two parts \( A \) and \( B \), and project one part on a spectral subspace of the other. This method succeeds on many continuous models of data, for example gaussian and log-concave distributions \[28, 43\]. The situation is more complex on discrete models, where a combinatorial clean-up phase is seemingly needed. In [33], for example, it was shown that a spectral
projection based algorithm followed by a combinatorial cross-training succeeded in clustering the data. The model presented in that paper is quite general, and works under essentially tight separation conditions. For the simplified model presented here, standard algorithms ([13, 33]) successfully clusters the data as long as

\[ g \geq c \sqrt{d \log n} \]

where \( c \) is some constant. Later works proposed different forms combinatorial clean-up techniques [15]. Here we show that for our (simpler) model neither is necessary. Apart from simplicity of the algorithm involved, we believe the question of how the spectral error is distributed is important in extending spectral methods to more complex models.

### 4.5.3 The first analysis

We prove:

**Theorem 37** Assume \( d_1 = n^\frac{1}{l} \) for some \( l \geq 1 \). Algorithm 8 works if

\[ g \geq c \sqrt{\min (n \frac{2l+1}{(l+1)n} \log \frac{l-1}{l+1} d, n \frac{2l+1}{(l+1)n} \log \frac{l-1}{l+1} d)} \]

(4.14)

for some constant \( c \)

What is noticeable in the bound is that it gets worse as \( l \) increases, i.e. the degree \( d \) decreases. For small \( l \), the bound is in fact close to those of [33], though not quite equal, and the difference grows as \( l \) grows.

**Proofs**

Let’s first prove the theorem in a simpler case. For this result, we shall assume that
\[ p_{aa} = \frac{1}{\sqrt{n}} \text{ and } p_{ab} \leq \frac{1}{\sqrt{n}} - c_1 \sqrt{\frac{\log n}{n^{5/6}}} \text{ (for } b \neq a). \] This implies the following separation condition:

\[ d_1 - d_2 = g \geq c_1 \sqrt{n^{5/6} \log n} \quad (4.15) \]

To prove Theorem 36, we start with the following lemma which says that \( v_2(A) \) is close to a vector that bi-partitions the data correctly.

**Lemma 38** Let \( v = v_2(A) \). Then \( v = u + w \) such that

\[ u(i) = \text{sign}(a) \frac{1}{\sqrt{2n}} \quad \forall i \in T_a \]

and \( \|w\| \leq \frac{5}{c_1 n^{1/6} \sqrt{\log n}} \), where \( \text{sign}(1) = 1 \) and \( \text{sign}(2) = -1 \).

**Proof** This can be proved following known methods (see [13], for example).

Note at this point that the value of \( \|w\| \) is enough to induce errors (in fact many of them) of the order \( \sqrt{\frac{2}{n}} \) in \( v \), which is all that is necessary to cause Algorithm 8 to fail, and it is at this point that clean-up phases are necessary. We will show that this doesn’t happen. Our idea is to use an analysis of neighborhood sets of a vertex \( s \) to show that \( \|w\| \) cannot be distributed in an arbitrary fashion.

We will need the following proposition, easily proved from the relation between \( l_1 \) and \( l_2 \) norms.

**Proposition 39** Consider any subset \( S \subset [2n] \). Then

\[ \sum_S |w(i)| \leq \sqrt{|S| \|w\|} \]

Here is the main lemma:
Lemma 40 For all \( s \in T_1, v(s) > \frac{4}{5\sqrt{2n}} \) and for all \( s \in T_2, v(s) < -\frac{4}{5\sqrt{2n}} \).

Proof The claims for \( T_1 \) and \( T_2 \) are symmetric, hence we will only prove the first claim. Let \( s \in T_1 \). We use the following notation \( N_a = \{ T_a \cap N(s) \} \) and \( N_{ab} = \{ T_a \cap N(N_b(s)) \} \). Assume \( e_i(S) \) is the number of neighbors node \( i \) has in set \( S \). It is known that \( \lambda_2(A) = d_1 - d_2 \). Then,

\[
v(s) = \frac{1}{d_1 - d_2} \left( \sum_{N_1} v(i) + \sum_{N_2} v(i) \right)
= \frac{1}{(d_1 - d_2)^2} \left( \sum_{N_{11}} v(i)e_i(N_1) + \sum_{N_{21}} v(i)e_i(N_1) + \sum_{N_{12}} v(i)e_i(N_2) + \sum_{N_{22}} v(i)e_i(N_2) \right)
\]

We claim:

Claim 41

\[
\sum_{N_{11}} v(i)e_i(N_1) + \sum_{N_{12}} v(i)e_i(N_2)
+ \sum_{N_{21}} v(i)e_i(N_1) + \sum_{N_{22}} v(i)e_i(N_2) > \frac{4}{5\sqrt{2n}} (d_1 - d_2)^2
\]

Since

\[
\sum_{N_{ab}} v(i)e_i(N_b) = \sum_{N_{ab}} u(i)e_i(N_b) + \sum_{N_{ab}} w(i)e_i(N_b)
= \text{sign}(b) \frac{1}{\sqrt{2n}} e(N_b, N_{ab}) + \sum_{N_{ab}} w(i)e_i(N_b)
\]
We can rewrite the claim as

\[
\frac{1}{\sqrt{2n}} (e(N_1, N_{11}) + e(N_2, N_{12}) - e(N_2, N_{22}) - e(N_1, N_{21})) + \sum_{a b} \sum_{N_{ab}} w(i) e_i(N_b) > \frac{4}{5\sqrt{2n}} (d_1 - d_2)^2
\]

Now, since \( s \in T_1, N_1 \geq d_1 - \sqrt{d_1 \log n} \) and \( e(N_1, N_{11}) \geq (d_1 - \sqrt{d_1 \log n})^2 \). Using similar bounds for \( N_{12}, N_{21} \) and \( N_{22} \)

\[
e(N_1, N_{11}) + e(N_2, N_{12}) - e(N_2, N_{22}) - e(N_1, N_{21}) \\
\geq (d_1 - \sqrt{d_1 \log n})^2 + (d_2 - \sqrt{d_2 \log n})^2 - 2(d_1 + \sqrt{d_1 \log n})(d_2 + \sqrt{d_2 \log n}) \\
\geq d_1^2 + d_2^2 - 2d_1d_2 - 8d_1^{3/2}\sqrt{\log n} - 4d_1 \log n \\
\geq (d_1 - d_2)^2 - 8d_1^{3/2}\sqrt{\log n} - 4d_1 \log n
\]

Then,

\[
e(N_1, N_{11}) + e(N_2, N_{12}) - e(N_2, N_{22}) - e(N_1, N_{21}) \geq \frac{19}{20} (d_1 - d_2)^2 \quad (4.16)
\]
Now we need to bound $\sum_{ab} \sum_{N_a} |w(i)e_i(N_b)|$. The four terms in the sum are of the same order hence we will only bound one of them. We claim,

$$ |\sum_{N_{11}} w(i)e_i(N_1)| \leq \frac{4}{c_1} n^{1/3} \sqrt{\log n} \leq \frac{1}{50} \frac{1}{\sqrt{2n}} (d_1 - d_2)^2 $$

(4.17)

To prove this observe that $e_i(N_1) \leq d_1 (1 + \sqrt{\log n} d_1)$. Then,

$$ \sum_{N_{11}} |w(i)e_i(N_1)| \leq \sum_{t=1}^{\log 2d_1} \sum_{i: e_i \geq 2^{t-1}} 2^t \cdot |w(i)| \leq \log n \max_{t \leq \log d_1} 2^t \sum_{i: e_i(N_1) \geq 2^{t-1}} |w(i)| $$

The problem here is that, conceivably, for a large number of vertices $i$, $w(i)$ is large and so is $e_i(N_1)$, thus amplifying the effect of $w$. What we will show is that the size of a vertex set for which each $e_i(N_1)$ is large is small, thus disallowing this effect.

Let us bound for any $t \geq 1$ the value of

$$ 2^t \sum_{i: e_i(N_1) \geq 2^{t-1}} |w(i)| $$

For any $f$ define

- $M(f) = \{ i : e(i, N_1) \geq f \}$
- $m(f) = |M(f)|$

Then setting $f = 2^t$, by Proposition 39 we can write

$$ 2^t \sum_{i: e_i \geq 2^{t-1}} w(i) = f \sum_{M(f/2)} w(i) \leq \|w\| \sqrt{m(f/2) f} $$
By the definition of $M(f)$, $e(M(f), N_1) \geq f m(f)$. Now by quasirandomness

$$m(f) + 2\sqrt{n} \sqrt{m(f)} \geq f m(f)$$

Then, it is easy to see that $\sqrt{m(f)} f \leq 4\sqrt{n}$, the main point being that $f$ doesn’t appear on the right hand side.

Therefore $\sum_{i: e_i \geq 2^{k-1}} 2^t w_i \leq \|w\|4\sqrt{n} \leq \frac{4}{c_1} \frac{n^{1/3}}{\sqrt{\log n}}$. Eqn 4.17 now follows by simple manipulation and by assuming an appropriate value of $c_1$. Comparing Eqn 4.16 and 4.17, Claim 41 follows.

Lemma 40 proves lower and upper bounds for positive and negative entries of $v$, respectively. The same technique can be used to prove respective upper and lower bounds, which then implies the Theorem.

**Proof sketch for general case**  The proof follows that of Lemma 40. Starting from any $s \in T_1$, we consider successive neighborhood sets for $[l]$ or $\lceil l \rceil$ steps. These two alternatives result in two bounds, the minimum of which we use as the final bound. As the analysis is almost identical, let us consider the case of $\lfloor l \rfloor$ iterations.

Let $N_r$ be the $r^{th}$ neighborhood set.

As in Lemma 40, the analysis will finally boil down to a bound like Eqn 4.17, albeit more complicated:

$$ \left| \sum_{N_{\lfloor l \rfloor}} w(i)p(i) \right| \leq \frac{1}{50} g^{\lfloor l \rfloor} \quad (4.18)$$

Where $p(i)$ is the number of paths of length $\lfloor l \rfloor$ from $i$ to $s$.

Now we need to bound $p(i)$. Consider sequence of sets $F_k \subset N_k$ such that each vertex in $F_k$ has exactly $f_k$ edges to $F_{k-1}$. Then each vertex of $F_{\lfloor l \rfloor}$ will have $\prod f_k$
paths of length \(|l|\) to \(s\). Using quasirandomness, it is possible to bound

\[ |F_{[l]}| \leq \frac{2n}{\prod f_k^2} \]

Hence

\[ |\sum_{F_{[l]}} w(i)p(i)| \leq \|w\| \sqrt{|F_{[l]}| \prod f_k} \leq \sqrt{2n}\|w\| \]

As \(f_k \leq d + \sqrt{d \log n}\), we can discretize each step using the same technique as in Lemma 40. So we will have to consider \(\log|l|^{d-1}d\) sequences.

Now we know that \(\|w\| \leq \frac{2\sqrt{d}}{g} \leq \frac{2n\frac{1}{d}}{g}\), so it suffices for 4.18 that

\[
\frac{1}{50}g^{|l|} \geq \frac{2n\frac{1}{d}}{g}\sqrt{n}\log|l|^{d-1}d
\]

or,

\[
g \geq n^{\frac{2(l+1)}{2l+1}} \log|l|^{d-1}d
\]

But this is exactly our separation condition (Eqn 4.14).

The main challenge in converting this bound to one comparable to [33] is reducing the separation conditions. It seems that the main bottleneck in our technique is an appeal to a bound on the “error vector” \(\|w\|\).

**4.5.4 The second analysis**

The following analysis provides slightly different bounds, and works better than the first analysis as \(p\) gets smaller. Let \(\delta = c\frac{\log n}{\log d} \sqrt{\frac{\log n}{d}}\) and \(f(r) = \sum_{t=1}^r 2(1 + \frac{d}{g})^r\). The we prove
Theorem 42 Algorithm 8 succeeds if the separation condition holds:

\[ d_1 - d_2 = g \]

where \( g \) is such that for some \( l \geq 1 \)

\[
\begin{align*}
    f(l) &\leq \frac{1}{4\delta} \\
    \left( \frac{2\sqrt{d}}{g} \right)^{2l} &\leq \frac{1}{4\sqrt{2n}}
\end{align*}
\]

Let’s first see how to prove the theorem. Then we’ll deal with the issue of what these specific conditions imply. For any vector \( x \), we will use the notation \((x)_a\) for the vector with entries of \( x \) corresponding to \( T_a \). Then, \( x = [(x)_1^t \quad (x)_2^t]^t \). We will simplify notation by using \( x = y \pm \epsilon \) to mean \( y - \epsilon \leq x \leq y + \epsilon \). We will also use \( \text{vec}[\text{expression}] \) to mean a vector for which every entry is defined by the expression. For example \( \text{vec}[y \pm \epsilon] \) will denote a vector \( x \) for which \( y - \epsilon \leq x(i) \leq y + \epsilon \), for all \( i \).

Now, let’s rewrite \( A \) as

\[
A = \begin{pmatrix} S_1 & B \\ B^t & S_2 \end{pmatrix}
\]

Where \( S_a \) corresponds to \( T_a \).

Let \( q = v_1(S_1) \), \( r = v_1(S_2) \), \( x = v_1(B) \) and \( y = v_1(B^t) \). Also assume,
\[(v)_1 = s + s_\perp \quad (4.20)\]
\[(v)_1 = x + x_\perp \quad (4.21)\]
\[(v)_2 = r + r_\perp \quad (4.22)\]
\[(v)_2 = y + y_\perp \quad (4.23)\]

where \(s_\perp \cdot s = 0\). Note that, by Theorem 27.

\[s(i) = \frac{1}{\sqrt{2^n}} (1 \pm \delta) \quad (4.24)\]

Also, \(s_\perp \leq \frac{2\sqrt{d}}{q}\) by Lemma 38. Similar statements are true for \(x, r, y\) and for \(x_\perp, r_\perp\) and \(y_\perp\).

The following two are observations we will need in our proof.

**Lemma 43** Let \(x \perp s\). Then \(\|S_1 x\| \leq \sqrt{d}\|x\|\). Same is true of \(B\) and \(S_2\).

**Proof** By eigenvalue bounds from [24, 45].

**Lemma 44** Let \(x, y, z\) be vectors such that

\[x \cdot y = 0\]

and

\[\|x - z\|_\infty = \epsilon\]

Then \(|y \cdot z| \leq \sqrt{n}\epsilon\|y\|\)
Proof

\[ |y \cdot z| \]
\begin{align*}
= |y \cdot (z - x + x)| &= |y \cdot x + y \cdot (z - x)| \\
= |y \cdot (z - x)| &\leq \|z - x\|_{\infty} \sum_i |y(i)| \\
= \epsilon \sqrt{n} \|y\|
\end{align*}

Lemma 45 There is a sequence of vectors \( \{s^{(t)}, r^{(t)}, e^{(t)}, f^{(t)}\} \) for \( t \geq 1 \) such that

\begin{align*}
(v)_1 &= s^{(t)} + s^{(t)}_{\perp} = y^{(t)} + y^{(t)}_{\perp} \\
(v)_2 &= r^{(t)} + r^{(t)}_{\perp} = x^{(t)} + x^{(t)}_{\perp}
\end{align*}

where

\begin{align*}
    s^{(t)}(i) &= \frac{1}{\sqrt{2n}} (1 \pm f(t) \delta) & (4.25) \\
    y^{(t)}(i) &= \frac{1}{\sqrt{2n}} (1 \pm f(t) \delta) & (4.26) \\
    r^{(t)}(i) &= -\frac{1}{\sqrt{2n}} (1 \pm f(t) \delta) & (4.27) \\
    x^{(t)}(i) &= -\frac{1}{\sqrt{2n}} (1 \pm f(t) \delta) & (4.28)
\end{align*}
\[ \| s(t) \| \leq \left( \frac{2 \sqrt{d}}{g} \right)^t \]  
\[ \| r(t) \| \leq \left( \frac{2 \sqrt{d}}{g} \right)^t \]  
\[ \| x(t) \| \leq \left( \frac{2 \sqrt{d}}{g} \right)^t \]  
\[ \| y(t) \| \leq \left( \frac{2 \sqrt{d}}{g} \right)^t \]  
\[ s(t) \cdot s = 0 \]  
\[ r(t) \cdot r = 0 \]  
\[ x(t) \cdot x = 0 \]  
\[ y(t) \cdot y = 0 \]

**Proof** The base case follows from Equations (4.20), (4.21), (4.22), (4.23), and (4.24).

Now for the induction step, assume this is true for \( t = r - 1 \).

Then

\[
(Av)_1 = S_1(v)_1 + B(v)_2 \\
= S_1(s^{(r-1)} + s^{(r-1)}_\perp) + B(x^{(r-1)} + x^{(r-1)}_\perp) \\
= \frac{1}{\sqrt{2n}} (d_1 - d_2) \left[ (1 \pm \delta)(1 \pm f(r-1)\delta) \right]_{\text{vec}} \\
+ \frac{1}{\sqrt{2n}} \left[ \pm d_2(\delta + f(r-1)\delta) \right]_{\text{vec}} + S_1 s^{(t)}_\perp + Bx^{(t)}_\perp \\
= \frac{1}{\sqrt{2n}} g \left[ (1 \pm \delta \pm f(r-1)\delta) \right]_{\text{vec}} \\
+ \frac{1}{\sqrt{2n}} \frac{d_2}{g} \left[ \delta + f(r-1)\delta \right]_{\text{vec}} + S_1 s^{(t)} + Bx^{(t)}_\perp
\]
Therefore

\[
(v)_1 = \frac{(Av)_1}{g} = \frac{1}{\sqrt{2n}} \left[ (1 \pm \delta \pm f(r-1)\delta) \pm \frac{d_2}{g} \left( \delta + f(r-1)\delta \right) \right]_{\text{vec}} + \frac{1}{g} S_1 s^{(t)}_{\perp} + \frac{1}{g} B x^{(t)}_{\perp}
\]

\[
= \frac{1}{\sqrt{2n}} \left[ (1 \pm \delta \left( f(r) - 1 - \frac{d_2}{g} \right) \right]_{\text{vec}} + a + b
\]

Where \( a = \frac{1}{g} S_1 s^{(t)}_{\perp} \) and \( b = \frac{1}{g} B x^{(t)}_{\perp} \).

Now, by Lemma 43,

\[
\|a\| \leq \frac{2\sqrt{d}}{g} \|s^{(t)}_{\perp}\| \leq \left( \frac{2\sqrt{d}}{g} \right)^r
\]

\[
\|b\| \leq \frac{2\sqrt{d}}{g} \|x^{(t)}_{\perp}\| \leq \left( \frac{2\sqrt{d}}{g} \right)^r
\]

This is almost what we want, as \( a \cdot s = 0 \). However, \( b \) might not be orthogonal to \( s \).

However, it is “almost” orthogonal. By Lemma 44,

\[
b \cdot s \leq \sqrt{2n} \frac{1}{\sqrt{2n}} \|b\| \leq \left( \frac{2\sqrt{d}}{g} \right)^r \delta
\]

\[
\Rightarrow (b \cdot s)s = \pm \frac{1}{\sqrt{2n}} \left( \frac{2\sqrt{d}}{g} \right)^r \delta
\]

85
Factoring out the component of $b$ in the direction of $s$ we then get

$$\frac{1}{\sqrt{2n}} \left[ 1 \pm \delta(f(r) - 1 - \frac{d_2}{g}) \right]_{\text{vec}} + a + b$$

$$= \frac{1}{\sqrt{2n}} \left[ 1 \pm \delta(f(r) - 1 - \frac{d_2}{g}) \right]_{\text{vec}}
+ a + (b \cdot s)s + (b - (b \cdot s)s)
= \frac{1}{\sqrt{2n}} \left[ 1 \pm \delta(f(r) - 1 - \frac{d_2}{g}) \right]_{\text{vec}}
+ \frac{1}{\sqrt{2n}} \left[ \pm (\sqrt{2d/g})^r \right]_{\text{vec}} + (a + b - (b \cdot s)s)
= \frac{1}{\sqrt{2n}} \left[ (1 \pm \delta f(r)) \right]_{\text{vec}} + (a + b - (b \cdot s)s)$$

Setting $(a + b - (b \cdot s)b) = s_{1}^{(r)}$, we have found $(v)_1 = s^{(r)} + s_{1}^{(r)}$ with required conditions.

The other bounds can be proven in a similar way.

**Proof of the Theorem** Consider the $l$ for which the Separation condition 4.19 holds. Then $\|s_{\perp}^{l}\|_{\infty} \leq \|s_{\perp}^{l}\| \leq \frac{1}{16\sqrt{n}}$ while $\frac{1}{\sqrt{n}}(1 - \frac{1}{4}) \leq s^{l}(i) \leq \frac{1}{\sqrt{n}}(1 + \frac{1}{4})$. Hence $(v)_1 = \frac{1}{\sqrt{n}} \left[ 1 \pm \frac{1}{3} \right]_{\text{vec}}$ similarly, $(v)_2 = -\frac{1}{\sqrt{n}} \left[ 1 \pm \frac{1}{3} \right]_{\text{vec}}$. This implies the required gap and proves Theorem 42.

**Implications** If $d = n^{\alpha}$ for a constant $\alpha \leq 1$, then we require $g \geq n^{\beta}$ for some other constant $\beta < \alpha$. The bounds are comparable, but somewhat weaker than that of Eqn 4.14. The bounds become comparable for $d_1 \approx n^{\frac{1}{\log n}}$, and Eqn 4.19 beats Eqn 4.14 below that level. Specifically, for degrees polylogarithmic in $n$, the analysis improves on the bounds from our first analysis. For example $g \geq d_1 - \sqrt{d_1 \log n}$ suffices for all degrees, which is better than Eqn 4.14 for $d_1 \approx \text{polylog}(n)$. These bounds, as mentioned before, are much weaker than bounds from [33], but the worst-case norm of $\|w\|$ even for these conditions is large enough to create errors in Algorithm 8.
Chapter 5

A Simple Algorithm for Discrete Distributions

5.1 Introduction

In this chapter, we propose a projection-based, rotationally invariant algorithm for clustering mixture of distributions, including discrete distributions. This resolves a conjecture of McSherry. It also gives first clustering algorithms for a class of discrete distributions previously not dealt with, and gives a natural algorithm that works for wide class of discrete and continuous distributions.

We are concerned here with probabilistic mixture models. We start with $k$ “simple” probability distributions $D_1, D_2 \ldots D_k$ where each is a distribution on $\mathbb{R}^n$. With each $D_r$ we can associate a center $\mu_r$ of the distribution, defined by $\mu_r = E_{D_r}(v)$. A mixture of these probability distributions is a density of the form $w_1D_1 + w_2D_2 + \ldots + w_kD_k$, where the $w_i$ are fixed non-negative reals summing to 1. A clustering question in this case would then be: What is the required condition on the probability distributions so that given a collection of vectors generated from the mixture, we
can group the vectors into \(k\) clusters, where each cluster consists precisely of vectors picked according to one of the component distributions of the mixture?

The “planted partition model” we have been studying so far is a special case of the mixture model, except for the added requirement of symmetry. Symmetry only adds very mild dependence to the model, and can be handled easily. Consequently, we will restrict ourselves to mixture models. We present and analyze a very natural algorithm that works for quite general mixture models. The algorithm is completely “geometric”, in the sense that it is not coordinate specific in any way, is based on ordinary projections to natural subspaces, and is robust under rotation. Our algorithm is the first such algorithm shown to work for discrete distributions. The existence of such an algorithm was first conjectured by McSherry in [33].

Why is this interesting? To elucidate some of these reasons, we need to introduce the common techniques used in the literature and their limitations.

The method very commonly used to cluster mixture models (discrete and continuous) is “spectral clustering” — which refers to a large class of algorithms where the eigenvectors or singular vectors of \(A\) are used in some fashion. Many works have come out on spectral clustering for continuous distributions, which have focused on high-dimensional gaussians and their generalizations [1, 14, 28, 43]. On the other hand, the use of eigenvectors for discrete, graph-based models was pioneered by Boppana [10]. The essential aspects of many techniques used now first appeared in [4, 3]. These results were generalized by McSherry [33] and a large body of work followed ([14, 15] etc).

Despite their variety, these algorithms have a common first step. Starting with well-known spectral norm bounds on \(\|A - \mathbb{E}(A)\|\) (e.g. [45], there are many others), a by-now standard analysis can be used to shown that the projection on best \(k\)-dimensional approximation would find a good approximate partitioning of the data.
The major effort, often, goes into “cleaning up” this approximate partitioning.

At this point the methods for discrete and and continuous methods diverge. For continuous models, spectral clustering is often followed by projecting new samples on the spectral subspace obtained. Here is why, broadly speaking, this works: the spectral norm bounds imply that the spectral subspace is a span of the real centers of the data, plus some error. This error can be thought of a (or a small number of) vector(s) of small $l_2$ norm. Now, if we consider a new, random sample, this sample will be almost orthogonal to the error vector(s) with high probability. Hence, it is as if we were projecting to the span of centers, which is a good subspace to project to, and the clustering is complete.

Unfortunately, this is simply not true for discrete distributions. This has resulted in rather ad-hoc methods for cleaning up mixture of discrete distributions. A most pertinent example would be “combinatorial projections”, proposed by McSherry [33], and all other algorithms for discrete distributions share the same central features we are concerned with. Here, one uses the span of characteristic vectors of the approximate clusters as the subspace to project down to. Though successful in that particular context, the idea of combinatorial projections is unsatisfactory for a number of inter-related reasons. First, such algorithms aren’t robust under many natural transformations of the data for which we expect the clustering to remain the same. The most important case is perhaps rotation. It is natural to expect that if all vectors of the data are rotated by the same rotation matrix, the clustering would remain the same. Projective algorithms would continue to work in these cases, while combinatorial methods might not. To take the case of “combinatorial projection”, the behavior of the algorithm becomes unpredictable if the vectors are rotated. A related issue is this: the overriding idea used to cluster discrete distributions is that the feature space (in addition to the object space) has clearly delineated clusters.
One way to state this condition is that $\mathbb{E}(A)$ can be divided into a small number of sub matrices, each of which contains the same value for each entry. This is implicit in graph-based models, as the objects and features are the same things — vertices. These results will extend to the case of rectangular matrices as long as some generalization of that condition holds, but for general cases where the centers don’t necessarily have such strong structure it is not clear how to extend these ideas. Our algorithm solves this problem.

We will focus on discrete distributions (i.e. Bernoulli distributions) where each entry of a sample vector is an independent 0/1 variable. Discrete distributions are the “hard” distributions in the present context. That our algorithm will work for many usual continuous distributions, such as spherical or sphere-like high-dimensional gaussians will not be very hard to see.

This chapter is organized as follows. In Section 5.2 we present the basic model and assumptions, our results and review of more relevant literature. Next, in Section 5.3 the algorithm is presented. Section 5.4 consists of the technical results that prove our bounds.

5.2 Model

There are $k$-centers $\mu_r$, such that $0 \leq \mu_r(i) \leq \sigma^2 \leq 1$ for all $r \in [k], i \in [n]$, and some $\sigma^2 \geq \frac{\log^3 n}{n}$. Each center defines a probability distribution on $\mathbb{R}^n$: a $n$-dimensional sample $v$ from this distribution is generated by setting $v(i) = 1$ with probability $\mu_r(i)$ and 0 otherwise, independently for all $i \in [n]$.

With each distribution we associate weight $w_r \geq 0$, such that $\sum_{r\in[k]} w_r = 1$. The data matrix is generated in the following way. For each distribution, a set $T_r$ of $w_r m$ samples are chosen from it, adding up to $\sum_{r\in[k]} w_r m = m$ total samples. The $m$
samples are arranged as columns of a $n \times m$ matrix $A$, which is presented as the data. We will use $A$ to mean both the matrix, and the set of vectors that are rows of $A$, where the particular usage will be clear from the context. $E(A)$ is defined by rule: $A_i \in T_r$ then $(E(A))_i = \mu_r$ ($M_i$ is the $i^{th}$ column of $M$). The algorithmic problem is: Given $A$, can we find the clusters $T_1, T_2 \ldots T_k$?

For successful clustering, we need to assume the following:

**Separation condition:** We assume, for all $r, s \in [k], r \neq s$

$$\|\mu_r - \mu_s\|^2 \geq 8100ck\sigma^2 \frac{1}{w_{\min}} \left(1 + \frac{n}{m}\right) + \log m$$  \hspace{1cm} (5.1)

for some constant $c$.

The following can be proved using large deviation inequalities (for example, Theorem 52). For a sample $v \in T_r$, with probability $1 - \frac{1}{m^r}$,

$$|(v - \mu_r) \cdot (\mu_s - \mu_r)| \leq \frac{1}{10} \|\mu_s - \mu_r\|^2$$  \hspace{1cm} (5.2)

This is a quantitative version of the property that the distance of a sample to its own center is smaller than its distance to the center of another cluster, but this representation has some technical advantages. In fact, one can reasonably consider this as an assumption of the model, since without such a property, it’s not clear how one could cluster the data even in principle.

**Related Work**

Let us discuss another example to further illustrate the dichotomy between discrete and continuous distributions in the literature. Two related papers, [15] and [1] employ a very natural linkage algorithm to perform the clean-up phase, and share
Figure 5.1: If $v \in T_r$, the projection of $v - \mu_r$ on $\mu_r - \mu_s$ will be closer to $\mu_r$ than it is to $\mu_s$.

much of their algorithm. However, these papers respectively address discrete and continuous distributions, and the difference influences the algorithm used in telling ways. In [1], the authors generalize high-dimensional Gaussians by adopting the notion of $f$-concentrated distributions. Consequently, a simple projection algorithm suffices. In [15], on the other hand, the clean-up procedure uses the “combinatorial projection” method pioneered in [33].

5.3 The Algorithm

The algorithm, reduced to its essentials, is simple and natural. First, we randomly divide the rows into two equal parts — $A_1$ and $A_2$. We will use information of one part to cluster the other part, as this allows to utilize independence of these parts. For $A_1$, we will find its best rank $k$ approximation $A_1^{(k)}$ (by computing the Singular Value Decomposition of the matrix). We will find approximately correct centers by a procedure described in the next few paragraphs. Once we have the centers, we measure the distance of each sample in $A_2$ (the other part) to these centers, and group the sample with the closest center.

To find the approximate centers we will have to solve the solution to the $l_2^2$-
clustering problem (also known as the “k-means problem”).

$l_2^2$ clustering problem: Given a set of vectors $S = \{v_1, \ldots, v_l\}$ in $\mathbb{R}^d$ and a positive integer $k$, the problem is to find $k$ points $f_1, \ldots, f_k \in \mathbb{R}^d$ (called “centers”) so as to minimize the sum of squared distances from each vector $v_i$ to its closest center. This naturally defines a partitioning of the $n$ points into $k$ clusters. We compute average of the vectors in each cluster, and these will be the approximate centers.

Quite efficient constant factor deterministic approximation algorithms (even PTAS’s) are available for this problem (See [19, 27, 30] and references therein). We will work with the factor 9 approximation algorithm of Kanungo et. al. [30].

Algorithm 9 Cluster($A, k$)

1: Randomly divide the rows $A$ into two equal parts $A_1$ and $A_2$
2: $(\theta_1, \ldots, \theta_k) = \text{Centers}(A_1, k)$
3: $(\nu_1, \ldots, \nu_k) = \text{Centers}(A_2, k)$
4: $(P_1^1, \ldots, P_1^k) = \text{Project}(A_1, \nu_1, \ldots, \nu_k)$
5: $(P_2^1, \ldots, P_2^k) = \text{Project}(A_2, \theta_1, \ldots, \theta_k)$
6: Assume $P_r^1$ and $P_r^2$ correspond the same clusters for all $r$ (this can be done by comparing centers and permuting the index set accordingly)
7: return $(P_1^1 \cup P_2^1, \ldots, P_1^k \cup P_2^k)$

Algorithm 10 Centers($A, k$)

1: Find $A^{(k)}$, the best rank-$k$ approximation of the matrix $A$.
2: Solve the $l_2^2$ clustering problem with the following parameters
   
   $\bullet$ The input vectors are rows of $A^{(k)}$
   
   $\bullet$ $k$ is the number of centers we seek

3: Let the clusters computed by the $l_2^2$ algorithm be $P_1 \ldots P_k$
4: For all $r$, compute $\mu^*_r = \frac{1}{|P_r|} \sum_{v \in P_r} v$
5: return $(\mu^*_1, \ldots, \mu^*_k)$
Algorithm 11 \(\text{Project}(A, \mu_1^*, \ldots, \mu_k^*)\)

1: For each \(v \in A\) 
2: For each \(r \in [k]\) 
3: \quad If \(|(v - \mu^*_r) \cdot (\mu^*_s - \mu^*_r)| \leq |(v - \mu^*_s) \cdot (\mu^*_r - \mu^*_s)|\) for all \(r \neq s\) 
4: \quad Put \(v\) in \(P_r\) (breaking ties arbitrarily) 
5: return \((P_1, \ldots, P_k)\)

5.4 Analysis

The following theorem of Vu [45] bounds the spectral norm of \(\|A - \mathbb{E}(A)\|\):

**Theorem 46** If \(A\) is generated as described in Section 5.2, then with probability 
\(1 - o(1)\)
\[
\|A - \mathbb{E}(A)\|^2 \leq c\sigma^2(m + n) \tag{5.3}
\]

for some (small) constant \(c\).

The following was proved by McSherry [33]:

**Lemma 47** With probability \(1 - o(1)\), the best rank-\(k\) approximation matrix \(A^{(k)}\) satisfies
\[
\|A^{(k)} - \mathbb{E}(A)\|_F^2 \leq 8ck\sigma^2(m + n)
\]

**Proof** As both \(A^{(k)}\) and \(\mathbb{E}(A)\) are of rank \(k\), \(A^{(k)} - \mathbb{E}(A)\) has rank at most \(2k\). Then
\[
\|A^{(k)} - \mathbb{E}(A)\|_F^2 \\
\leq 2k\|A^{(k)} - \mathbb{E}(A)\|^2 \\
= 2k\|A^{(k)} - A + A - \mathbb{E}(A)\|^2 \\
\leq 4k(\|A - A^{(k)}\|^2 + \|A - \mathbb{E}(A)\|^2) \\
\leq 8k(\|A - \mathbb{E}(A)\|^2) \leq 8ck\sigma^2(m + n)
\]
Lemma 48 Given an instance of the clustering problem, consider any vector $y$ such that, for some $r$

$$\|y - \mu_r\|^2 \leq \frac{1}{9}\|\mu_r - \mu_s\|^2$$

for all $s \neq r$. Let, $B_s = \{x \in T_s : \|A^{(k)}(x) - y\|^2 \leq \frac{1}{4}\|\mu_r - \mu_s\|^2\}$. Also let

$$\sum_{B_s} \|A^{(k)}(x) - \mu_s\|^2 = E_s$$

Then for all $s \neq r$,

$$|B_s| \leq \frac{40E_s}{\|\mu_r - \mu_s\|^2}$$

And specifically,

$$|B_s| \leq \frac{320ck\sigma^2(m + n)}{\|\mu_r - \mu_s\|^2}$$

Proof From the conditions of the Lemma,

$$\|y - \mu_s\| = \|y - \mu_r + \mu_r - \mu_s\|$$

$$\geq \|\mu_r - \mu_s\| - \|y - \mu_r\|$$

$$\geq 0.66\|\mu_r - \mu_s\|$$

Now, for each $x \in B_s$

$$\|A^{(k)}(x) - \mu_s\|$$

$$\geq (\|y - \mu_s\| - \|A^{(k)}(x) - y\|)$$

$$\geq 0.16 \times \|\mu_r - \mu_s\|$$
By assumption,
\[ \sum_{B_s} \| A^{(k)}(x) - \mu_s \|^2 = E_s \]
\[ \Rightarrow 0.0256 \times \| \mu_r - \mu_s \|^2 |B_s| \leq E_s \]
\[ \Rightarrow |B_s| \leq \frac{40E_s}{\| \mu_r - \mu_s \|^2} \]

Now as
\[ E_s = \sum_{B_s} \| A^{(k)}(x) - \mu_s \|^2 \leq \| A - \mathbb{E}(A) \|_F^2 \leq 8ck\sigma^2(m + n) \]
\[ |B_s| \leq \frac{320ck\sigma^2(m + n)}{\| \mu_r - \mu_s \|^2} \]

This proves the Lemma. \( \square \)

The important aspect of the previous Lemma is that the size of \( |B_s| \) goes down as \( \| \mu_r - \mu_s \|^2 \) increases. This property will be used later.

We characterize the clustering produced in \textbf{Centers}(A, k) in the following Lemma.

**Lemma 49** Consider a clustering \( P_1, P_2 \ldots P_k \) produced by the algorithm \textbf{Centers}(A, k).
We claim:

- Each \( P_r \) can be identified with a unique and different \( T_s \) such that
\[ |P_r \cap T_s| \geq \frac{4}{5}m_s \] (5.4)

Without loss of generality, we shall assume \( s = r \).
We define
\[
\begin{align*}
\bar{P}_r &= T_r \cap P_r \\
Q'_r &= P_r - \bar{P}_r \\
Q'_{rs} &= P_s \cap Q'_r \\
E_{rs} &= \sum_{v \in Q'_{rs}} \| A^{(k)}(v) - \mu_r \|^2
\end{align*}
\]

Then for all \( r, s \in [k] \) such that \( r \neq s \)

\[
q'_{rs} = |Q'_{rs}| \leq \frac{40E_{rs}}{\|\mu_r - \mu_s\|^2} \leq \frac{320ck\sigma^2(m + n)}{\|\mu_r - \mu_s\|^2}
\]

(5.5)

**Proof** First we claim that there is a solution to the \( l_2^2 \) clustering problem so that the cost of the solution \( \leq 8ck\sigma^2(m + n) \). Set the centers to be \( f_r = \mu_r \) for all \( r \) and let the cost of this solution be \( C \).

Now,

\[
C \leq \sum_r \sum_{A^{(k)}(i) \in [T_r]} \| A^{(k)}(i) - \mu_r \|^2
\]

\[
= \| A^{(k)} - \mathbb{E}(A) \|^2_{F}
\]

\[
\leq 8ck\sigma^2(m + n)
\]

By Lemma 47.

**Remark** The problem calls for centers \( f_r \) to be in the span of vectors in \( A^{(k)} \), and \( \mu_r \) might not be. But this simply strengthens the result.

Accordingly, \( \text{Cluster}(G, k) \) produces a solution to the \( l_2^2 \) clustering problem with cost no more than \( 72ck\sigma^2(m + n) \) (since we compute a factor 9 approximation to the
solution). We claim, for each \( P_r \) the center \( f_r \) will be such that, for all \( s \neq r \)

\[
\|\mu_r - f_r\|^2 \leq \frac{1}{9}\|\mu_r - \mu_s\|^2
\]  

(5.6)

If this is not true for some \( r \), then the cost of the solution is at least

\[
\sum_{A^{(k)}(i) \in T_r} \|A^{(k)}(i) - f_r\|^2 \\
\geq \sum_{A^{(k)}(i) \in T_r} \|A^{(k)}(i) - \mu_r + f_r - \mu_r\|^2 \\
\geq \frac{1}{4} \sum_{A^{(k)}(i) \in T_r} \|f_r - \mu_r\|^2 - 3 \sum_{A^{(k)}(i) \in T_r} \|A^{(k)}(i) - \mu_r\|^2 \\
\geq \frac{1}{36w_{\min}} \left( 8100ck\sigma^2(1 + \frac{n}{m}) \log m \right) m_r - 216ck\sigma^2(m + n) \\
\geq 9ck\sigma^2(m + n)
\]

which is a contradiction. We used the bound \( \|u + v\|^2 \geq \frac{1}{4}\|u\|^2 - 3\|v\|^2 \) for the second inequality, and the assumption that Eqn (5.6) is false for the third inequality.

Assuming (5.6), Lemma 48 implies (5.5). Equation (5.4) follows from essentially the same argument. \( \square \)

We now focus on the centers \( \mu^*_r \) produced by \textbf{Centers}(\( A, k \)). First we would like to show that they are close to the real centers.

**Lemma 50** Let \( \mu^*_1 \ldots \mu^*_k \) be the centers returned by the procedure \textbf{Centers}(\( A, k \)). Then for all \( r \in \{1, \ldots, k\} \), and for all \( s \neq r \)

\[
\|\mu^*_r - \mu_r\|^2 \leq 81ck\sigma^2\frac{1}{w_{\min}} \left( 1 + \frac{n}{m} \right) \leq \frac{1}{20}\|\mu_r - \mu_s\|^2
\]  

(5.7)
Proof. We know,

\[ \mu^*_r = \frac{1}{p_r} \sum_{v \in P_r} v \]

\[ \Rightarrow p_r \mu^*_r = \sum_{v \in P_r} v = \sum_{P_r} v + \sum_{s} Q'_{rs} v \]

\[ \Rightarrow p_r (\mu^*_r - \mu_r) = \sum_{P_r} (v - \mu_r) + \sum_{s} Q'_{rs} (v - \mu_r) \]

Then,

\[ \|p_r(\mu^*_r - \mu_r)\| \leq \| \sum_{P_r} (v - \mu_r) \| + \sum_{s} \| \sum_{Q'_{rs}} (v - \mu_r) \| \] (5.8)

Let, the samples in \( P_r \) be \( v^1 \ldots v^\bar{p}_r \). Let’s define the matrix \( S \) with these samples as columns:

\[ S = (v^1 \ v^2 \ldots v^\bar{p}_r) \]

And, \( U \) be the matrix of same dimensions with all columns equal to \( \mu_r \)

\[ U = (\mu_r \ \mu_r \ldots \mu_r) \]

Also let \( 1 = \{1, \ldots 1\}^T \), with \( \bar{p}_r \) entries. Now, by Theorem 46,

\[ \|S - U\| \leq \|A - \mathbb{E}(A)\| \leq \sigma \sqrt{c(m+n)} \]

\[ \Rightarrow \| (S - U) 1 \| \leq \| 1 \| \sigma \sqrt{c(m+n)} \leq \sigma \sqrt{c(m+n)} \bar{p}_r \]
As,

\[ \sum_{P_r} (v - \mu_r) = (S - U)1 \]

\[ \| \sum_{P_r} (v - \mu_r) \| \leq c\sigma \sqrt{(m+n)\bar{p}_r} \tag{5.9} \]

Now, for any \( s \)

\[ \| \sum_{Q'_rs} (v - \mu_r) \| \leq \| \sum_{Q'_rs} (v - \mu_s) \| + \| q'_{rs}\mu_s - q'_{rs}\mu_r \| \]

But we know by Lemma 48

\[ q'_{rs} \leq \frac{40E_{rs}}{\| \mu_r - \mu_s \|^2} \]

Then,

\[ \| q'_{rs}\mu_s - q'_{rs}\mu_r \| = ((q'_{rs})^2\| \mu_s - \mu_r \|^2)^{\frac{1}{2}} \leq (40q'_{rs}E_{rs})^{\frac{1}{2}} \]

\[ \Rightarrow \| q'_{rs}\mu_s - q'_{rs}\mu_r \| \leq \sqrt{q'_{rs}E_{rs}} \tag{5.10} \]

On the other hand, through an argument similar to the bound for \( \sum_{P_r} (v - \mu_r) \)

\[ \| \sum_{Q'_rs} (v - \mu_s) \| \leq \sigma \sqrt{cq'_{rs}(m+n)} \tag{5.11} \]
Combining equations (5.9)—(5.11),

\[ \| p_r(\mu^*_r - \mu_r) \| \leq \sigma \sqrt{c(m + n)p_r} + \sum_s \sigma \sqrt{c q'_{rs}(m + n)} + \sum_s \sqrt{40q'_{rs}E_{rs}} \]

\[ \leq \sigma \sqrt{ck(m + n)p_r} + \sum_s \sqrt{40q'_{rs}E_{rs}} \]

\[ \leq 2\sigma \sqrt{ck(m + n)p_r} + \sqrt{\sum_s 40q'_{rs}} \sqrt{\sum_s E_{rs}} \]

Using the Cauchy-Schwartz inequality a few times. But we know that \( \sum_s E_{rs} \leq 8ck\sigma^2(m + n) \) and \( \sum q'_{rs} \leq \frac{1}{5}p_r \) (Eqn (5.4) in Lemma 49). Hence,

\[ \| p_r(\mu^*_r - \mu_r) \| \leq \sigma \sqrt{ck(m + n)p_r} + 8\sigma \sqrt{ck\sigma^2p_r(m + n)} \]

\[ \leq 9\sigma \sqrt{ck(m + n)p_r} \]

\[ \Rightarrow \| \mu^*_r - \mu_r \| \leq 9\sigma \sqrt{\frac{ck(m + n)}{p_r}} \]

\[ \leq 9\sigma \sqrt{ck \cdot \frac{1}{w_{\min}} \left(1 + \frac{n}{m}\right)} \]

Finally we would like to show that Project\((A, \mu_1 \ldots \mu_k)\) returns an accurate partitioning of the data. To this end, we claim that \((v - \mu^*_r) \cdot (\mu^*_r - \mu^*_t)\) behaves essentially like \((v - \mu_r) \cdot (\mu_r - \mu_t)\).

**Lemma 51** For each sample \( u \), if \( u \in T_r \), then for all \( s \neq r \)

\[ |(u - \mu^*_r) \cdot (\mu^*_r - \mu^*_t)| \leq \frac{2}{5} \| \mu_r - \mu_t \|^2 \]  

(5.12)

with high probability.
Proof Assume $\mu_r^* = \mu_r + \delta_r; \forall r$. Then,

\[
(u - \mu_r^*) \cdot (\mu_t^* - \mu_r^*)
= (u - \mu_r - \delta_r) \cdot (\mu_t - \mu_r - \delta_r + \delta_t)
= (u - \mu_r) \cdot (\mu_r - \mu_t) - \delta_r \cdot (\mu_t - \mu_r) - \delta_r \cdot (\delta_t - \delta_r) + (u - \mu_r) \cdot (\delta_t - \delta_r)
\]

Let’s consider each term in the last sum separately.

By equation (5.2),

\[
(u - \mu_r) \cdot (\mu_r - \mu_t) \leq \frac{1}{10} \|\mu_r - \mu_t\|^2
\]

(5.13)

We have already shown (Lemma 50) that $\|\delta_r\| \leq \frac{1}{\sqrt{20}} \|\mu_r - \mu_t\|$. Then

\[
|\delta_r \cdot (\mu_t - \mu_r)| \leq \|\delta_r\| \|\mu_r - \mu_t\| \leq \frac{1}{\sqrt{20}} \|\mu_r - \mu_t\|^2
\]

(5.14)

And

\[
|\delta_r \cdot (\delta_t - \delta_r)| \leq \frac{1}{20} \|\mu_r - \mu_t\|^2
\]

(5.15)

The remaining term is $(u - \mu_r) \cdot (\delta_r + \delta_t)$. We prove in Lemma 53 (below) that

\[
|(u - \mu_r) \cdot (\delta_r + \delta_t)| \leq \frac{1}{100} \|\mu_r - \mu_s\|^2
\]

(5.16)

Combining equations (5.13)—(5.16), we get the proof. \square

For the proof of Lemma 53, we will need Bernstein’s inequality (see [37] for a reference):

**Theorem 52** Let $\{X_i\}_{i=1}^n$ be a collection of independent, almost surely absolutely
bounded random variables, that is \( \Pr \{ |X_i| \leq M \} = 1 \ \forall i \). Then, for any \( \varepsilon \geq 0 \)

\[
\Pr \left\{ \left| \sum_{i=1}^{n} (X_i - \mathbb{E}[X_i]) \right| \geq \varepsilon \right\} \leq \exp \left( -\frac{\varepsilon^2}{2(\theta^2 + \frac{M^3}{3}\varepsilon)} \right) \tag{5.17}
\]

where \( \theta = \sum \mathbb{E}X^2_i \)

In the following, it is crucial to assume that the sample \( u \) is independent of \( \delta_r \) and \( \delta_t \). We can assume this because we use centers from \( A_1 \) on samples from \( A_2 \), and vice-versa, which are independent.

**Lemma 53** If \( u \in D_r \) is a sample independent of \( \delta_r \) and \( \delta_t \), for all \( t \neq r \)

\[
|(u - \mu_r) \cdot (\delta_r + \delta_t)| \leq 15ck\sigma^2 \frac{1}{w_{\text{min}}} \left( \left( 1 + \frac{n}{m} \right) + \log m \right) \leq \frac{1}{100} ||\mu_r - \mu_s||^2 \tag{5.18}
\]

with high probability.

**Proof** It suffices to prove a bound on \( (u - \mu_r) \cdot \delta_r \). The case for \( \delta_t \) is similar. As

\[
(u - \mu_r) \cdot \delta_r = \sum_{i \in [n]} (u(i) - \mu_r(i))\delta_r(i) = \sum_{i \in [n]} x(i)
\]

where \( x(i) = (u(i) - \mu_r(i))\delta_r(i) \). This is a sum of independent random variables.

Note that by Lemma 50

\[
||\delta_r||^2 \leq 81ck\sigma^2 \frac{1}{w_{\text{min}}} \left( 1 + \frac{n}{m} \right)
\]

103
Now,

\[ \mathbb{E}(x(i)) = \mathbb{E}(u(i) - \mu_r(i))\delta_r(i) = 0 \]

\[ \mathbb{E}(x(i)^2) \leq 2\delta_r(i)^2\sigma^2 \]

\[ \Rightarrow \sum_i \mathbb{E}(x(i)^2) \leq 2\sigma^2 \|\delta_r\|^2 \leq 162ck\sigma^4 \frac{1}{w_{\text{min}}} \left(1 + \frac{n}{m}\right) \]

Also note that \(|x(i)| \leq |\delta_i| \leq \frac{2\sigma^2}{w_{\text{min}}}\). This is simply because the number of 1’s in a column can be at most \(1.1m\sigma^2\). Hence \(|\delta_i| \leq \frac{1}{pr}1.1m\sigma^2 \leq \frac{2\sigma^2}{w_{\text{min}}}\).

We are ready to apply Bernstein’s inequality, using which we get,

\[
\Pr\left\{ \left| \sum_{i \in [n]} x(i) \right| \geq 15ck\sigma^2 \frac{1}{w_{\text{min}}} \left( (1 + \frac{n}{m}) + \log m \right) \right\} \\
\leq \exp\left( -\frac{-225c^2k^2\sigma^4 \frac{1}{w_{\text{min}}} \left( (1 + \frac{n}{m}) + \log m \right)^2}{162ck\sigma^4 \frac{1}{w_{\text{min}}} \left( (1 + \frac{n}{m}) + \log m \right)} \right) \\
\leq \frac{1}{m^4}
\]

The correctness of the algorithm follows:

**Theorem 54** Cluster\((A,k)\) successfully clusters the matrix \(A\).

**Proof** If suffices to show that Project\((A,\mu_1^* \ldots \mu_k^*)\) works. Merging the clusters from two calls of Project is easy just by comparing the respective centers.

Now, Let \(u \in T_r\). For all \(t \neq r\),
\[
(u - \mu_r^*) \cdot (\mu_t^* - \mu_t^*) + (u - \mu_t^*) \cdot (\mu_r^* - \mu_t^*) \\
= (\mu_r^* - \mu_t^*) \cdot (u - \mu_t^* - u + \mu_r^*) \\
= (\mu_r^* - \mu_t^*) \cdot (-\mu_t^* + \mu_r^*) \\
= \|\mu_r^* - \mu_t^*\|^2
\]

By Lemma 50

\[
\|\mu_r^* - \mu_t^*\|^2 \geq 0.95\|\mu_r - \mu_t\|^2 \\
\Rightarrow (v - \mu_r^*) \cdot (\mu_t^* - \mu_r^*) + (v - \mu_t^*) \cdot (\mu_r^* - \mu_t^*) \geq 0.95\|\mu_r - \mu_t\|^2 \\
\Rightarrow |(v - \mu_r^*) \cdot (\mu_t^* - \mu_r^*)| + |(v - \mu_t^*) \cdot (\mu_r^* - \mu_t^*)| \geq 0.95\|\mu_r - \mu_t\|^2
\]

As \(|(v - \mu_r^*) \cdot (\mu_t^* - \mu_r^*)| \leq 0.4\|\mu_r - \mu_t\|^2\) by Lemma 51

\[
|(v - \mu_r^*) \cdot (\mu_t^* - \mu_r^*)| \geq 0.55\|\mu_r - \mu_t\|^2 > |(v - \mu_r^*) \cdot (\mu_t^* - \mu_r^*)|
\]

This proves our claim. \(\square\)

As mentioned in the introduction, our analysis extends beyond the case of Bernoulli distributions. Bernstein’s inequality works for subgaussian distributions [37], and similar bounds are available for vectors with limited independence as well (e.g. [39]). Given these bounds, all we need to complete the proof for these cases is a bound on the spectral norm of \(\|A - \mathbb{E}(A)\|\) for such distributions, which is also available (e.g. [14, 38]).
Chapter 6

Conclusion

Here we will provide a technical summary of the results proved in the previous four chapters.

Analysis of the second singular vector method

In Chapter 2 an algorithm that uses the second singular vector to partition a graph is presented. We prove its soundness for the following model:

$A$ is a matrix of probabilities where the entry $A_{uv} = A_{vu}$ is the probability of an edge being present between the vertices $u$ and $v$. The vertices are partitioned into $k$ clusters $T_1, T_2, \ldots, T_k$. The probability $A_{uv}$ depends only on the two clusters that the vertices $u$ and $v$ belong to. In other words, there exist probabilities $P_{rs}(= P_{sr})$ for all $r, s \in [k]$, such that, if $u \in T_r$ and $v \in T_s$, then $A_{uv} = P_{rs}$.

The size of the $r^{th}$ cluster $T_r$ is $n_r$. Define $w_r = \frac{n_r}{n}$ and $n_{\min} = \min_r \{n_r\}$, $w_{\min} = \min_r \{w_r\}$.

Given the probability matrix $A$, the random graph $\hat{A}$ is then generated by independently setting each $\hat{A}_{uv}(= \hat{A}_{vu})$ to 1 with probability $A_{uv}$ and 0 otherwise. In what follows, we use the terms “column” and “vertex” interchangeably, noting that
vertex $x$ corresponds to column $A_x$.

Now, the expectation of the random variable $\hat{A}_{uv}$ is equal to $A_{uv}$. The variance of $\hat{A}_{uv}$ is thus $A_{uv}(1 - A_{uv})$. The maximum variance of any entry of $\hat{A}$ is denoted $\sigma^2$, and the maximum variance for all columns belonging to a cluster $T_r$ is denoted by $\sigma^2_r$. The number of vertices is $n$.

We assume that each of the variances $\sigma_r$ satisfies $\sigma^2_r \geq \log^6 n/n$. We also need an assumption:

**Separation Condition:** There exists a large enough constant $c$ such that for vertices $u \in T_r$ and $v \in T_s$, the columns $A_u$ and $A_v$ of the probability matrix $A$ corresponding to different clusters $T_r$ and $T_s$ satisfy

$$\|A_u - A_v\|_2^2 \geq 1024c^3k^7(\sigma_r + \sigma_s)^2 \log(n) \frac{\log(n)}{w^6_{\text{min}}}$$

(6.1)

We prove the following:

**Theorem 55** Given $\hat{A}$ that is generated as above, there is a polynomial time algorithm that can find the clusters $T_1, T_2 \ldots T_k$ with probability at least $1 - \frac{1}{n^r}$.

**Quasi-random models for clustering**

In Chapter 3 we propose a model of clustering that allows for constant degree graphs using the notion of quasi-randomness, and then design an algorithm to compute the clustering. $G(V, E)$ is a graph with $|V| = n$ vertices. Let $A$ be the adjacency matrix of the graph. There are $k$ clusters $T_r; r = 1 \ldots k$ such that $T_r \cap T_s = \emptyset$ and $\bigcup_r T_r = V$.

Let, $n_r = |T_r|$, $w_r = \frac{n_r}{n}$, $n_{\text{min}} = \min_r n_r$, and $w_{\text{min}} = \min w_r$. We assume that for a parameter $\alpha$, for all pairs $T_r, T_s$ (not necessarily different), there is $d_{rs}$ and $d_{rs}$ such that
1. $d_{rs}n_r = d_{sr}n_s$

2. $|e(x, T_s) - d_{rs}| \leq c_1 \alpha \log d_{rs}$ for all $x \in T_r$

3. $|e(x, T_r) - d_{sr}| \leq c_1 \alpha \log d_{sr}$ for all $x \in T_s$

4. Let $X = A(T_r, T_s)$. Then,

$$|e_X(S, R) - sr \frac{d_{rs}}{n_s}| \leq \alpha \sqrt{sr}$$

for each $S \subset T_r$ and $R \subset T_s$, where $s = |S|, r = |R|$.

Let $A'$ be the $n \times n$ matrix where $A'(i, j) = \frac{d_{rs}}{n_s}$ if $i \in T_r$ and $j \in T_s$.

We assume the following separation condition among centers of clusters. Define the center of a cluster $T_r$, denoted $\mu_r$ to be $\mu_r(x) = \frac{d_{rs}}{n_s}$ for all $x \in T_s$. Now, we will assume, for all $r \neq s$

$$g_{rs} \equiv ||\mu_r - \mu_s||^2 \geq 2c_1k^2 \frac{\alpha^2}{w_{\text{min}}n} \log^2 d$$

(6.2)

for some constant $c_1$.

We prove the following:

**Theorem 56** There is a polynomial time algorithm such that, given a graph $G(V, E)$ and $k$ that satisfies the conditions of the model described and the separation condition, the algorithm finds the clusters $T_1 \ldots T_k$

The algorithm we use is a spectral algorithm, followed by an incremental algorithm that improves the clustering until it finds the accurate clustering.
Entrywise bounds for eigenvectors

We prove in Chapter 4 a nearly optimal bound for entry wise differences of the first eigenvector of the adjacency matrix of a random graph.

$A(G)$ will be used to denote the adjacency matrix of graph $G$, and $A$ when $G$ is clear from the context. For any vector $x \in \mathbb{R}^N$, $x_{\max} = \max_{i \in [N]} x_i$ and $x_{\min} = \min_{i \in [N]} x_i$. For a vector $x$, let $x(i)$ be its $i^{th}$ entry. We will use the phrase “with high probability” to mean with probability $1 - o(1)$.

The following is the main result. Define $\Delta = \sqrt{\frac{\log n}{d}}$.

**Theorem 57** Let $G \in G_{n,p}$ be a random graph and $A$ be its adjacency matrix. Assume $p \geq \log^{6} n/n$. Let $d = np$ and $v = \gamma v_1(A)$, where $\gamma$ is a normalization factor such that $v_{\max} = 1$. Then, with high probability

$$v_{\min} \geq 1 - c_2 \frac{\log n}{\log d} \Delta$$

for some constant $c_2$.

Since $v_1(\mathbb{E}A) = \{\frac{1}{\sqrt{n}} \ldots \frac{1}{\sqrt{n}}\}$, the get the following corollary:

**Corollary 58** Let $G \in G_{n,p}$ be a random graph and $A$ be its adjacency matrix. Assume $p \geq \log^{6} n/n$. Then, with high probability

$$\|v_1(A) - v_1(\mathbb{E}A)\|_{\infty} \leq c \frac{\log n}{\log d} \frac{1}{\sqrt{n}} \sqrt{\frac{\log n}{np}}$$

for some constant $c$.

We then use similar techniques to prove some results for spectral clustering.
Projective algorithms for discrete distributions

In Chapter 5 we provide a simple, rotationally invariant algorithm for mixtures of discrete distributions. There are \(k\)-centers \(\mu_r\), such that \(0 \leq \mu_r(i) \leq \sigma^2 \leq 1\) for all \(r \in [k], i \in [n]\), and some \(\sigma^2 \geq \frac{\log^6 n}{n}\). Each center defines a probability distribution on \(\mathbb{R}^n\): a \(n\)-dimensional sample \(v\) from this distribution is generated by setting \(v(i) = 1\) with probability \(\mu_r(i)\) and 0 otherwise, independently for all \(i \in [n]\).

With each distribution we associate weight \(w_r \geq 0\), such that \(\sum_{r \in [k]} w_r = 1\). The data matrix is generated in the following way. For each distribution, a set \(T_r\) of \(w_r m\) samples are chosen from it, adding up to \(\sum_{r \in [k]} w_r m = m\) total samples. The \(m\) samples are arranged as columns of a \(n \times m\) matrix \(A\), which is presented as the data. We will use \(A\) to mean both the matrix, and the set of vectors that are rows of \(A\), where the particular usage will be clear from the context. \(\mathbb{E}(A)\) is defined by rule: \(A_i \in T_r\) then \((\mathbb{E}(A))_i = \mu_r\) (\(M_i\) is the \(i^{th}\) column of \(M\)).

For successful clustering, we need to assume the following:

**Separation condition:** We assume, for all \(r, s \in [k], r \neq s\)

\[
\|\mu_r - \mu_s\|^2 \geq 8100ck\sigma^2 \frac{1}{w_{\min}} \left( \left( 1 + \frac{n}{m} \right) + \log m \right)
\] (6.4)

for some constant \(c\).

We prove the following:

**Theorem 59** Given \(A\) that is generated as above, where the separation condition between centers is satisfied, there is a polynomial time algorithm that can find the clusters \(T_1, T_2 \ldots T_k\) with probability at least \(1 - \frac{1}{n^2}\).
Bibliography


[47] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for
document datasets. In International Conference on Information and Knowledge