Graph Clustering:
Algorithms, Analysis and Query Design

Thesis by
Ramya Korlakai Vinayak

In Partial Fulfillment of the Requirements for the
degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2018
Defended August 21, 2017

Ramya Korlakai Vinayak
ORCID: 0000-0003-0248-9551

To Amma and Appaji,
Jalaja and Vinayak.

ACKNOWLEDGEMENTS

First and foremost, I want to thank my advisor, Prof. Babak Hassibi, for his guidance and kindness. He has been a great teacher and a mentor. I thank him for giving me the opportunity to explore my research interests and help me grow as an independent researcher. I have not only learned mathematical tools from him, but also how to think about various problems and how approach them, and how to keep an eye open for useful and interesting problems. I am deeply thankful to all my thesis committee members, Prof. Adam Wierman, Prof. Pietro Perona, Prof. Venkat Chandrasekaran, and Prof. Yisong Yue. It has been a pleasure to have them on my committee. Their guidance and feedback have immensely helped me in shaping my research ideas.

My lab mates have been great company in this journey at Caltech. I would like to thank Samet, who guided me in my first year in the lab introduced me to the problem of graph clustering. I want to thank all my lab mates – Samet, Matt, Wei, Christos, Weal, Ahn, Bose, Ehsan, Navid, Kishore, Fariborz, Ahmed, Anatoly, Philipp, and Hikmet for all the wonderful conversations whether it was research or politics or entertainment. I will dearly miss the Friday lunch group meetings. I want to thank my undergraduate mentees, Berk, Tijana, and Anna; working with them has helped me grow as a researcher. I also want to thank everyone from the Caltech Vision Group for always welcoming me to their group meetings and for all the wonderful discussions and feedback.

I am immensely thankful for my friends at Caltech. Thank you Vikas, Pinaky, Sujeet, Nisha, Radio (Subrahmanyam), Armeen, Brian, Corina, Surabhi, Priya, Amuthan, Eyrún, Cristina, Prachi, Anupama, Priya, and Amuthan for all the philosophical discussions, fun road trips, cooking, and for being there for me in all the ups and downs, and thanks also to Corina, Surabhi, and Catherine for being such wonderful room mates and bringing color into otherwise gray graduate student life. Being a woman in engineering can sometimes be a very lonely journey and I have been fortunate to have met some wonderful women during my journey throughout undergraduate and graduate studies. I want to thank Eyrún, Azadeh, Rose, Lavanya, Chinmayee, Gauri, and Madeleine, who I met in graduate school, conferences, and internships, and all my wing mates from undergraduate times for being good friends and mentors. I want to thank all my undergraduate friends who kept in touch regardless of thousands of miles of distance. Thank you Naveen, NG, Smruthi, Slinky

(Arun), Prasanna, Vinay, Akila, Adithi, Siva, Subhash, Prakash, and Samaadhi (Arjun) for being there not only to share the good times but also for being supportive through the tough times.

I am deeply grateful to Caltech's International Students Program, the Caltech Y, and the Caltech Center for Diversity for all the events that made me feel welcome at Caltech and gave me the opportunity to experience and explore LA. I am thankful to Shirley, Tanya, and Katie in the EE department and Tess from registrar's office for making my life as a student very smooth.

Last but not the least, I want to thank my family who have been supportive throughout my life. I want to thank my mother, Jalajakshi Hegde, for being a source of strength and always encouraging me to work towards my goals. I also want to thank my late father, Vinayakmurthy Hegde, who was very encouraging of my goals. I am grateful to have a wonderful sister, Rashmi Vinayak, who has always been my source of inspiration. I am also immensely thankful to my brother-in-law Nihar Shah for all the insightful discussions.

ABSTRACT

A wide range of applications in engineering as well as the natural and social sciences have datasets that are unlabeled. Clustering plays a major role in exploring structure in such unlabeled datasets. Owing to the heterogeneity in the applications and the types of datasets available, there are plenty of clustering objectives and algorithms. In this thesis we focus on two such clustering problems: *Graph Clustering* and *Crowdsourced Clustering*.

In the first part, we consider the problem of graph clustering and study convex-optimization-based clustering algorithms. Datasets are often messy – ridden with noise, outliers (items that do not belong to any clusters), and missing data. Therefore, we are interested in algorithms that are robust to such discrepancies. We present and analyze convex-optimization-based clustering algorithms which aim to recover the low-rank matrix that encodes the underlying cluster structure for two clustering objectives: *clustering partially observed graphs* and *clustering similarity matrices with outliers*. Using block models as generative models, we characterize the performance of these convex clustering algorithms. In particular, we provide *explicit bounds*, without any large unknown constants, on the problem parameters that determine the success and failure of these convex approaches.

In the second part, we consider the problem of crowdsourced clustering – the task of clustering items using answers from non-expert crowd workers who can answer similarity comparison queries. Since the workers are not experts, they provide noisy answers. Further, due to budget constraints, we cannot make all possible comparisons between items in the dataset. Thus, it is important to *design queries that can reduce the noise in the responses* and *design algorithms that can work with noisy and partial data*. We demonstrate that random triangle queries (where three items are compared per query) provide less noisy data as well as greater quantity of data, for a fixed query budget, as compared to random edge queries (where two items are compared per query). We extend the analysis of convex clustering algorithms to show that the exact recovery guarantees hold for triangle queries despite involving dependent edges. In addition to random querying strategies, we also present a novel *active querying* algorithm that is guaranteed to find all the clusters regardless of their sizes and without the knowledge of any parameters as long as the workers are better than random guessers. We also provide a tight upper bound on the number of queries made by the proposed active querying algorithm. Apart from providing

theoretical guarantees for the clustering algorithms we also apply our algorithms to real datasets.

PUBLISHED CONTENT AND CONTRIBUTIONS

[VH16a] Ramya Korlakai Vinayak and Babak Hassibi. "Crowdsourced Clustering: Querying Edges vs Triangles". In: *Advances in Neural Information Processing Systems*. 2016, pp. 1316–1324. URL: http://papers.nips.cc/paper/6499-crowdsourced-clustering-querying-edges-vs-triangles.pdf.
Ramya Korlakai Vinayak is the lead author and main contributor to the above paper.

[VH16b] Ramya Korlakai Vinayak and Babak Hassibi. "Similarity clustering in the presence of outliers: Exact recovery via convex program". In: *IEEE International Symposium on Information Theory (ISIT)*. 2016, pp. 91–95. URL: http://ieeexplore.ieee.org/abstract/document/7541267/.
Ramya Korlakai Vinayak is the lead author and main contributor to the above paper.

[VOH14a] Ramya Korlakai Vinayak, Samet Oymak, and Babak Hassibi. "Graph clustering with missing data: Convex algorithms and analysis". In: *Advances in Neural Information Processing Systems*. 2014, pp. 2996–3004. URL: http://papers.nips.cc/paper/5309-graph-clustering-with-missing-data-convex-algorithms-and-analysis.pdf.
Ramya Korlakai Vinayak is the lead author and main contributor to the above paper.

[VOH14b] Ramya Korlakai Vinayak, Samet Oymak, and Babak Hassibi. "Sharp performance bounds for graph clustering via convex optimization". In: *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*. 2014, pp. 8297–8301. URL: http://ieeexplore.ieee.org/abstract/document/6855219/.
Ramya Korlakai Vinayak is the lead author and main contributor to the above paper.

[VZH17] Ramya Korlakai Vinayak, Tijana Zrnic, and Babak Hassibi. "Tensor-based Crowdsourced Clustering via Triangle Queries". In: *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*. 2017. URL: http://ieeexplore.ieee.org/abstract/document/7952571/.
Ramya Korlakai Vinayak is the lead author and main contributor to the above paper.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Number *Page*

LIST OF TABLES

Chapter 1

INTRODUCTION

Unsupervised learning plays a crucial role in finding patterns and extracting useful information from unlabeled data. Clustering is the most widely used unsupervised learning tool. Many applications have data that is unlabeled, for example, biological images, data from observational astronomy, images, videos and texts on the internet, and so on. Clustering broadly refers to grouping together data items that are similar.

Clustering is in general an ill-posed problem and is meaningful when defined more concretely for a given objective. There are two main aspects to clustering. The first aspect concerns with the representation of data, the definitions of similarity and clusters, and the evaluation metrics. As the applications and the types of data available differ widely, so do the representations of data and definitions of similarity and clusters. The second aspect deals with algorithms for clustering a dataset, given an objective. This thesis looks at the second aspect in the context of *graph clustering* and *crowdsourced clustering*.

Clustering or cluster analysis originated in the 1930's with the work of Driver and Kroeber [DK32] in anthropology to understand the similarities between different cultural tribes in the Americas and was further used in psychology [Zub38; Try39] for personality analysis. Today clustering is an essential tool for quantitative analysis in many applications in engineering, the natural and social sciences. Given its applications in various areas, different point of views and many algorithms have been proposed. Hierarchical clustering, center based clustering, density based clustering, data partitioning, mixture models, etc., are some of the examples of different view points for clustering. There are many clustering algorithms, like k-means, k-median, linkage based agglomorative algorithms, partition based algorithms, spectral clustering, expectation-maximization, and variational inference based maximum likelihood algorithms, message passing algorithms, Monte Carlo methods, and many more. Given the breadth of research in clustering, an in-depth survey of the subject is out of the scope of this thesis. We refer the readers to the surveys and books dedicated to clustering [JMF99; VB05; AR13; XT15].

Crowdsourcing – as the name suggests – refers to using a crowd of potentially

non-expert humans to solve problems that are difficult to solve by machines. Using crowdsourcing to collect labeled data became popular around mid-2000's with games with a purpose [Von06] and recaptcha [Von+08]. Today crowdsourcing is used for a wide variety of tasks including collecting data [Ray+10], surveys [Beh+11], and tasks like translation, summarization, transliteration [Sno+08; SBS12; Kha+14], and many more. There are several platforms such as Amazon Mechanical Turk and Crowdflower available that bring the requesters (those who want the tasks done) and workers (those who are willing to do the tasks for monetary compensation or voluntarily) together. Many times enthusiastic general public voluntarily contribute to scientific discoveries via platforms like Zooinverse, Planet Hunters, etc. Applications of crowdsourcing for academic endeavors range from creating labeled datasets for training and testing supervised machine learning algorithms [Ray+10; Sno+08; Von+08; SF08; Wel+10; Yi+12] to making scientific discoveries [SPD14; Lin+13]. In this thesis, we focus on using crowdsourcing for clustering.

In the rest of this chapter we discuss the motivation and relevant literature, and summarize the contributions made in this thesis for problems in graph clustering and crowdsourced clustering.

1.1 Graph Clustering

In many applications the data has an inherent graph structure associated with it. For example, in a social network (Figure 1.1), individuals are the nodes in the graph and the edges represent the relationship between the individuals. In a protein-protein interaction network (Figure 1.2), proteins are the nodes in the graph and the edges represent the biochemical or electrostatic interaction between the proteins. It is often of interest to find a group of nodes in a graph that share high edge density inside the group compared to the density of the edges shared with the other groups in the graph. Such a group of nodes is called a *cluster*. In the example of a social network, one might be interested in finding a group of people who share the same taste in movies or sports. In a protein-protein network, understanding the grouping of different proteins can provide insights into various processes and reactions inside the cells. Thus, given an unweighted graph, finding nodes that are well-connected with each other is a very useful problem with wide applications in social networks [Mis+07; For10], data mining [DR01], biology and bioinformatics [LJG01; XOX02; KPJ04; AS06], sensor networks [YF04], astronomy [BBS85], and many more.

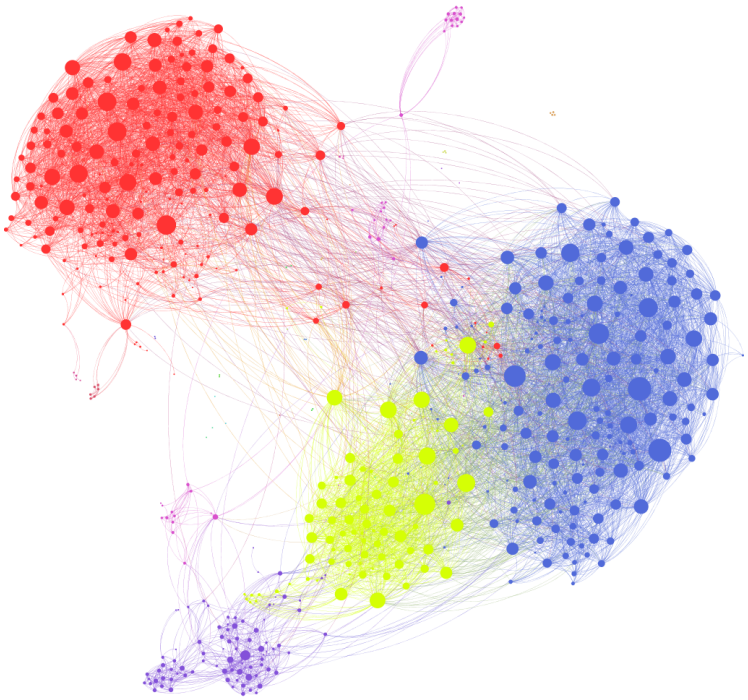


Figure 1.1: Visualization of a social network around an individual [urla].

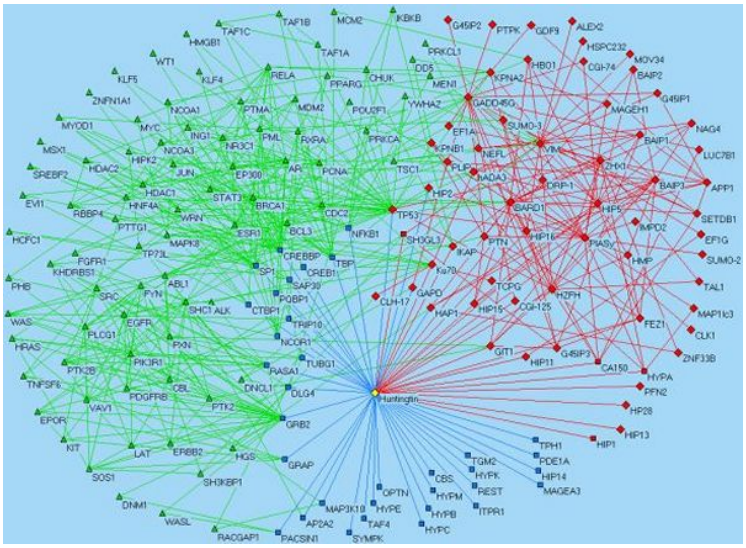


Figure 1.2: Visualization of a protein interaction network in Huntington's disease [urlb].

Different versions of this problem have been studied as graph clustering [Sch07; FTT04; CGW05], correlation clustering [EF03; BBC04; GG06], graph partitioning on planted partition model [CK01; McS01], community detection [FH16], etc. Many graph clustering objectives are NP-hard in the worst case. However, real world datasets are more structured and it is frequently observed that simple clustering algorithms perform quite well in practice. A natural question that arises is:

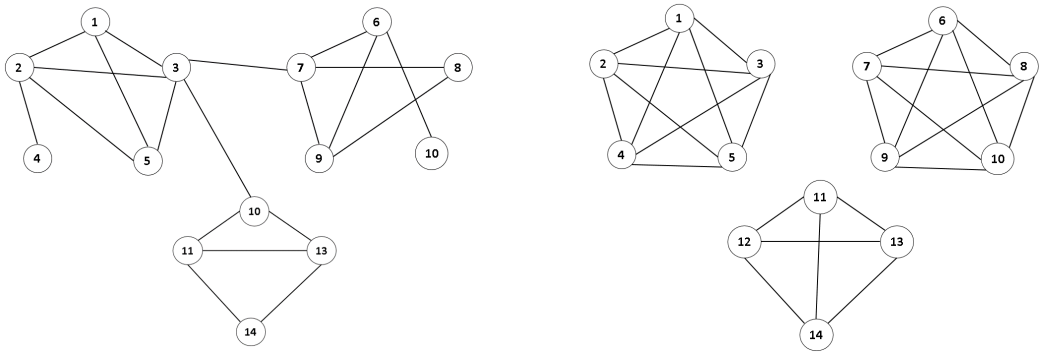
what are the structural properties of datasets that enable computationally efficient clustering?

The pertinent questions are: (1) How much should the difference between the edge density (or similarity) between the nodes inside a cluster and the edge density (or similarity) between the nodes from different clusters be? (2) When can we be confident that this difference is due to underlying structure in the data and not out of randomness or due to noise?

In order to understand these fundamental bottlenecks, we study simple generative *block models* which assume different distribution of edges (or similarity) inside the clusters and between the clusters. For example, consider the case of an unweighted graph with disjoint clusters. A popular block model in this setting is the *Stochastic Block Model* (SBM) or the planted partition model [HLL83; CK01] which assumes that given the assignment of nodes in a graph to (disjoint) clusters, each pair of nodes i and j is connected independently with probability p if they are in the same cluster and with probability q otherwise. Using the Stochastic Block Model, our analysis of simple convex programs for graph clustering (based on low-rank plus sparse decomposition of the adjacency matrix) shows that clusters that are small and sparse are the bottlenecks.

A common and well justified criticism of such block models is that they are not realistic. While the block models are simplistic, they capture the essence of the problem without getting lost in too many details. Studying these simple models can provide valuable insights into the fundamental bottlenecks for clustering. As statistician George Box famously said [B+87]:

"Essentially, all models are wrong, but some are useful."



(a) Toy example. (b) Ideal clusters.

Figure 1.3: A toy example for a graph clustering with clusters (a) and the corresponding ideal cluster structure (b).

Convex Programs for Clustering

Real world datasets are messy. They are noisy, have missing data and contain outliers – nodes that do not belong to any clusters. We need clustering algorithms to be robust to such discrepancies. We also want the algorithms to be tractable and computationally efficient. Further, we want these algorithms to be amenable to analysis that help us understand their strengths and weaknesses and provide guarantees as to when they can successfully recover the clusters.

Suppose a given graph has clusters. For example, consider the graph in Figure 1.3a, which has the following adjacency matrix:

$$\begin{bmatrix}
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1
\end{bmatrix}.$$

The above adjacency matrix can be looked at as the adjacency matrix of an ideal

cluster structure that is a union of disjoint cliques (see Figure 1.3b):

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{bmatrix},$$

(note that the rank of this matrix is 3 which is equal to the number of clusters in this example) plus noise that captures the missing edges inside the cliques and the extra edges that go between the cliques. Note that the adjacency matrix corresponding to a union of disjoint cliques (Figure 1.3b) has rank equal to the number of clusters (we assume that the diagonal of the adjacency matrix is all 1's). In many applications the number of clusters is much smaller than the number of nodes in the graph, and hence the ideal clustered graph has a low-rank adjacency matrix. Our aim is to recover the low-rank matrix corresponding to the union of disjoint cliques since it is equivalent to finding the clusters.

Several graph clustering objectives can be posed as low-rank matrix recovery and completion problems. Developments in convex optimization techniques to recover low-rank matrices [CR09; Can+11; CPW10; Cha+11] via nuclear norm [1] minimization has recently led to the development of several convex algorithms to recover clusters in a graph [XCS10; AV14; AV11; Che+14; OH11; CSX12; Ame13]. This *convex approach is robust to noise, outliers, missing data and unbalanced clusters*. Moreover, these convex approaches are often semi-definite programs (SDPs) that have polynomial computational complexity.

We consider convex algorithms for two clustering objectives. The first objective we consider is the problem of finding clusters in an unweighted graph when only partial observations are available. The second is the problem of clustering a similarity matrix or, equivalently, a weighted graph in the presence of outliers. Chapter 2 and Chapter 3 of this thesis are devoted to these two objectives.

[1] Nuclear norm of a matrix is the sum of the singular values of the matrix.

Graph Clustering with Missing Data

Clearly, a graph on n nodes has $\binom{n}{2}$ possible edges. In many practical scenarios, it is too expensive to measure or infer the existence or non-existence of each of these edges. Thus the problem of finding clusters in a graph that is *partially observed* or has *missing data* is of interest.

Our goal is to understand the fundamental trade-offs of this problem. In particular, we are interested in exploring the following questions:

1. How much noise can be tolerated? In particular, how much should the separation between the density of edges inside and between the clusters be?
2. How many observations are required to guarantee the exact recovery of clusters?
3. How large should the clusters be relative to the size of the graph for them to recovered exactly? How much does the balanced or unbalancedness of the relative cluster sizes matter?
4. How does the presence of outliers affect the ability to recover the clusters?

We consider two convex programs based on low-rank plus sparse decomposition of the observed adjacency matrix. We analyze their performances using the popular *Stochastic Block Model* where the edges in the graph are independent and (in the simplest case) exist with probability p inside the clusters and q between the clusters. To incorporate missing data, we further assume that each edge is independently observed with probability r. We show that the clusters that are both small and sparse are the bottlenecks. In simple terms, we show the following sufficient condition for exact recovery of clusters:

$$n_{\min} \geq 2\sigma \frac{\sqrt{n}}{\sqrt{r}\,(p-q)},$$

where $n_{\min}$ is the size of the smallest cluster and σ captures the noise due to the missing edges inside the cliques and the extra edges that go between the cliques. When $\sqrt{r}(p-q) = \Theta(1)$, our result shows that the sufficient condition for exact recovery requires the minimum cluster size to be at least $\Omega(\sqrt{n})$ which improves the previously known bound of $\Omega(\sqrt{n(\log n)^4})$. Unlike previous results on convex algorithms for graph clustering [AV11; OH11; CSX12; Ame13; Che+14; AV14], our

results do not involve unknown large constants. Furthermore, our analysis also provides precise conditions under which the convex approach fails in exact recovery, thus advancing the understanding of the *phase transition* from failure to success for the convex approach. Apart from providing theoretical guarantees for the problem considered and corroborating them with simulations, we also apply the convex program to a real data set obtained by crowdsourcing an image labeling task. Chapter 2 presents details of the problem of graph clustering with missing data and is based on the following papers [VOH14b; VOH14a].

Similarity Clustering in the Presence of Outliers

Several datasets have heterogeneous attributes, e.g. in census data, each individual person has attributes such as age and income which are numerical and race, religion, address etc., that are categorical. Depending on the application domain, it is often possible to construct a similarity map that assigns a numerical value to how similar two data items are, using which we can construct a *similarity matrix*. Such similarity matrices are generaly *noisy* because of the inherent noise in the data itself and the imperfections in the similarity maps. The data might also contain *outliers*, that is, data points that do not belong to any cluster.

We consider a convex program that aims to recover a (low-rank) matrix that reflects the true cluster structure from the noisy similarity matrix with outliers and no other side information. We analyze the convex program for a generative block model and provide *precise thresholds* (not orderwise) as a function of the problem parameters for successfully recovering the clusters. In simple terms we show the following sufficient condition for exact recovery of clusters:

$$n_{\min} \geq \frac{2\sqrt{n}\,\sigma + 1}{\mu_{in} - \mu_{out}},$$

where μ_{in} and μ_{out} are the average similarities inside and between the clusters respectively and σ is the standard deviation of the noise in the similarity matrix. Although our analysis uses the problem parameters, the program itself does not need to know them.

In the case of unweighted graphs, l_1 penalty on the noise is a natural loss function and as an added bonus the l_1 penalty is also robust to outliers. However, for similarity matrices (or weighted graphs) the l_1 penalty does not fit the noise characteristics. Instead a squared error loss function is more appropriate. Since squared error loss

is not inherently robust to outliers, we need to understand how the presence of outliers affect the recovery of clusters. We show that, in the presence of a large number of outliers ($\geq \Omega(\sqrt{n})$), the convex program forces the outliers to form their own cluster without disturbing the structure of the rest of the clusters.

Our analysis provides a complete picture of the fundamental bottlenecks of the simple convex program for similarity clustering. In contrast to the previously known results in this area [Ame14], our results do not involve large unknown constants and provides insights into the behavior of the solution in the presence of large number of outliers as well as the effect of noise variance. We corroborate our theoretical guarantees with extensive numerical simulations as well as evaluate the performance of the convex program on real datasets. Details on this problem are presented in Chapter 3 and are based on the following paper [VH16b].

1.2 Crowdsourced Clustering

There is an abundant amount of data, for example billions of images, texts, that can be readily scraped from the internet. However, most of these datasets are unlabeled and it is unclear what structures might exist in them. Crowdsourcing can be a very useful resource to explore structure in data [Wel+10]. Consider the task of collecting labels for unlabeled images of dogs of different breeds. To label each image of a dog, a worker should either be trained on dog breeds or one needs to hire workers who are experts in identifying dog breeds. Since hiring experts or training non-experts is expensive, we propose an alternate approach of clustering the images using comparison tasks. While the crowd workers may not be able to label an unlabeled image of a dog directly, they can compare pairs of images of dogs and judge whether they are from the same breed or not. Though different workers might use different criteria, e.g., color of the fur, size of the dog, shape of the ears, etc., we can reasonably resolve the clusters (and label each) from the adjacency matrix filled using responses from the crowd workers [VH15]. Since the crowd workers are often non-experts, the answers obtained will invariably be noisy. Furthermore, due to budget constraints, we cannot make all $\binom{n}{2}$ pairwise comparisons. Therefore, both the problem of *designing queries to obtain quality data from non-expert workers* and the problem of *designing robust algorithms to reliably cluster noisy and partially observed data* are of importance. Chapters 4, 5, and 6 of this thesis are dedicated to exploring these problems.

Figure 1.4: Example of an edge query: "Do these two birds belong to the same species?"

Triangle vs. Edge Querying

In any crowdsourcing system, we expect noisy answers from the crowd workers. Noise in the responses from crowd workers is not solely due to their lack of expertize or ability. Design of queries has a significant impact on the noise levels in the responses by the crowd workers. In Chapter 4 we consider the problem of crowdsourced clustering using similarity queries and compare two types of queries: (1) random edge queries, where a pair of images is revealed for comparison (Figure 1.4), and (2) random triangle queries, where a triplet is revealed (Figure 1.5). Using intuitive generative models, we show that the average number of errors in the entries of the adjacency matrix decreases when it is filled via triangle queries

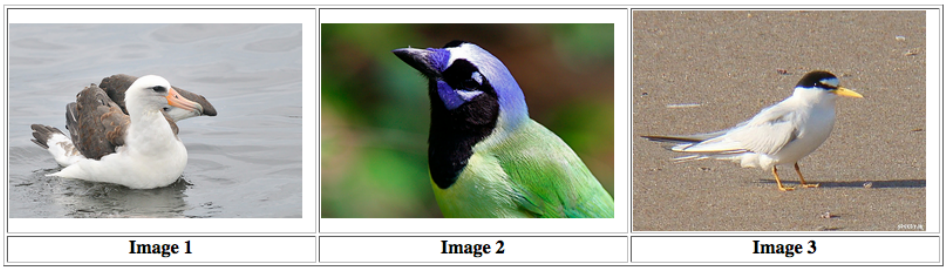


Figure 1.5: Example of a triangle query: "Which of these birds belong to the same species?"

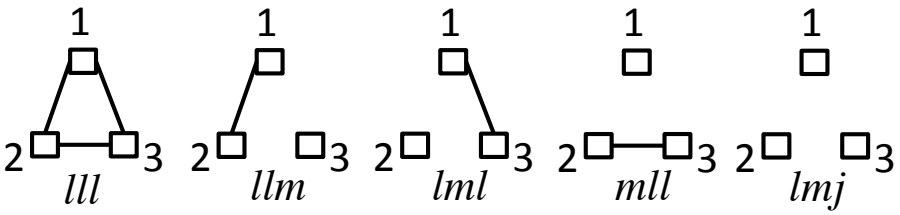


Figure 1.6: Configurations for a triangle query.

compared to when it is filled using using edge queries. Empirical evidence, based on extensive simulations, as well as experiments on Amazon Mechanical Turk using two real data sets, strongly suggests that, for a fixed query budget (the cost of a query is set proportional to the average response time of the workers), triangle queries significantly outperform edge queries. In particular, *under a fixed budget, triangle queries double the number of observed entries of the adjacency matrix while decreasing the average number of errors* by 5%.

For reliably clustering the noisy and partially observed data, we leverage the convex algorithms and analysis from our study of graph clustering. Our analysis of the graph clustering problem (recall the discussion in Section 1.1 and for details see Chapter 2) provides us the insight that convex clustering algorithms perform well when there is a good separation between the edge density inside and between clusters and when we make enough observations. Since triangle queries reduce the noise and provide more data, we expect graph clustering algorithms to work better on the adjacency matrices filled using triangle queries. However, the analysis in Chapter 2 assumes that the entries of the adjacency matrix are independent, which does not hold when it is filled via triangle queries. We extend the analysis to provide theoretical guarantee for the exact recovery of clusters via random triangle queries. Chapter 4 has more details and is based on the following paper [VH16a]).

Tensor Embedding for Triangle Queries

In Chapter 4, we demonstrate that using triangle queries to fill the adjacency matrix gives better clustering results than using edge queries. However, it does not exploit the fact that the response to a triangle query is associated with a triplet and a natural question that arises at this point is whether exploiting this fact adds any benefit. We explore this question in Chapter 5.

An edge query has two possible answers, and any choice of two symbols to represent them will give a true adjacency matrix whose rank is equal to the number of

clusters. In contrast, there are 5 possible answers to a triangle query (Figure 1.6) and in general we need 5 symbols to represent them. We can then embed the responses to the triangle queries into a 3-rd order tensor. Surprisingly, any arbitrary choice of symbols will generally not lead to a true tensor with rank equal to the number of clusters. We propose a general encoding scheme and provide sufficient conditions on it to give a true tensor with unique low-rank CP-decomposition with rank equal to the number of clusters. We also provide extensive numerical simulations that show that using tensor decomposition methods can improve over clustering obtained via the adjacency matrix. Details are available in Chapter 5 (which is based the following paper [VZH17]).

Active Querying for Crowdsourced Clustering

Querying for similarity between random pairs or triples of items followed by performing clustering on the adjacency matrix filled using these queries provides reasonable clusters. However, the bottlenecks in terms of how many observations are needed and what is the size of the smallest cluster that can be recovered are dictated by the graph clustering step. A natural question that arises is whether we can escape these bottlenecks as we are using crowdsourcing. We explore this question in Chapter 6 and consider an active querying setting for crowdsourced clustering. We propose a novel algorithm which uses confidence intervals that shrink as more queries are made and is based on the law of the iterated logarithm [Jam+14]. The algorithm we propose can reliably find the clusters without the knowledge of any parameters and without the small cluster barrier posed by graph clustering. In particular, under mild assumptions, we show that the number of queries made by the proposed algorithm is upper bounded by,

$$\mathcal{O}\left(\frac{nK}{\Delta^2}\log\left(n\log\frac{1}{\Delta}\right)\right),$$

where $\Delta := \min\{p-\frac{1}{2}, \frac{1}{2}-q\}$. In contrast with the state-of-the-art algorithms for this setting [MS16], *our algorithm is parameter free, computationally efficient and can recover clusters regardless of their cluster sizes*. In addition to the theoretical guarantees, we also provide extensive numerical simulations as well as experiments on real datasets, using both synthetic and real crowd workers. Chapter 6 provides the details of active querying for crowdsourced clustering.

Chapter 2

GRAPH CLUSTERING WITH MISSING DATA

In this chapter, we consider the problem of finding clusters in an unweighted graph when the graph is partially observed. We analyze two programs that are based on the convex optimization approach for low-rank matrix recovery using nuclear norm minimization with l_1-norm penalty on the noise. For the commonly used Stochastic Block Model, we obtain *explicit bounds* (not orderwise) on the parameters of the problem (size and sparsity of clusters, the amount of observed data) and the regularization parameter that characterize the success and failure of the convex programs. We corroborate our theoretical findings through extensive simulations. We also run our algorithm on a real data set obtained from crowd sourcing an image clustering task on the Amazon Mechanical Turk, and observe significant performance improvement over traditional clustering methods such as k-means and spectral clustering. This chapter is based on our papers [VOH14b; VOH14a].

2.1 Introduction

Clustering [JMF99] broadly refers to the problem of identifying data points that are similar to each other. It has applications in various problems in machine learning, data mining [EKX95; XJK99], social networks [Mis+07; DR01; For10], bioinformatics [XOX02; YL05], etc. In many applications, data has a graphical structure. For example, in social network, individuals are the nodes in the graph and the edges represent friendship between the nodes. Another example is a protein-protein interaction network, where the nodes in the graph are proteins and an edge between two proteins represents biochemical or electrostatic interaction between them. Given an *unweighted graph*, we define a *cluster* as a set of nodes that are densely connected to each other when compared to the rest of the nodes. Finding clusters in a graph can be useful in exploring and understanding the underlying structure in the data. For example, identifying clusters in a protein-protein interaction network can help us understand the functions of these proteins and gain insights into the processes they influence. Thus, the problem of finding clusters in a graph, that is, *graph clustering* [Sch07], is of importance. Clearly, observing an entire graph on n nodes requires $\binom{n}{2}$ measurements. In many practical scenarios this is too expensive and we can only expect to have *partial observations*. That is, for some node pairs we

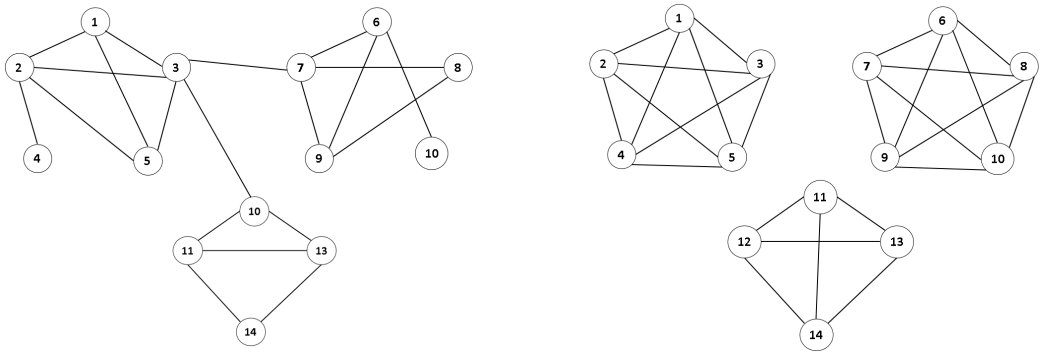
know whether there exists an edge between them or not, whereas for the rest of the node pairs we do not have this knowledge. Given a graph with partial observations, we still want to identify the underlying clusters. This leads us to the problem of clustering graphs with *missing data*.

In this chapter, we consider the problem of identifying clusters when the input is a partially observed adjacency matrix of an unweighted graph and look at two convex algorithms in this regard. We use the popular *Stochastic Block Model* (SBM) [HLL83], also called as *Planted Partition Model* [CK01], to analyze the performance of these convex algorithms. SBM is a random graph model where the edge probability depends on whether the pair of nodes being considered belong to the same cluster or not. More specifically, the edge probability is higher when both nodes belong to the same cluster. Further, we assume that each entry of the adjacency matrix of the graph is observed independently with probability r. We will define the model in detail in Section 2.2.

2.1.1 Clustering by Low-Rank Matrix Recovery and Completion

The idea of using convex optimization for clustering has been proposed in [XCS10; Che+14; AV14; AV11; OH11; CSX12; Che+14]. While each of these works differ in certain ways, and we will comment on their relation to the current paper in Section 2.1.3, the common approach they use for clustering is inspired by recent works on low-rank matrix recovery and completion via regularized nuclear norm (trace norm) minimization [CR06; CR09; Can+11; CPW10; Cha+11].

In the case of unweighted graphs, an ideal clustered graph is a union of disjoint



(a) Toy example. (b) Ideal clusters.

Figure 2.1: A toy example for a graph clustering with clusters (a) and the corresponding ideal cluster structure (b).

cliques (Figure 2.1b). Given the adjacency matrix of an unweighted graph with clusters – denser connectivity inside the clusters compared to outside (Figure 2.1a), we can interpret it as an ideal clustered graph with some missing edges inside the clusters and some erroneous edges in between the clusters. Recovering the low-rank matrix corresponding to the disjoint cliques is equivalent to finding the clusters.

Ideally we want to solve the following optimization problem:

$$\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \;\; \text{rank}\,(\mathbf{L}) + \lambda\|\mathbf{S}\|_0 \tag{2.1.1}$$

subject to

$$\mathbf{L}_{i,j} \in \{0,1\}, \tag{2.1.2}$$

$$\mathbf{L}^{obs} + \mathbf{S}^{obs} = \mathbf{A}^{obs}, \tag{2.1.3}$$

where $\text{rank}(\mathbf{Y})$ is the rank of the matrix $\mathbf{Y}$, $\|\mathbf{Y}\|_0$ is the l_0-norm of $\mathbf{Y}$ (number of non-zero entries in the matrix $\mathbf{Y}$), $\mathbf{L}^{obs}$ and $\mathbf{S}^{obs}$ are the low-rank and sparse parse matrices respectively over the observed entries of $\mathbf{A}$, and $\lambda \geq 0$ is the regularization parameter that balances the weight between $\text{rank}(\mathbf{L})$ and $\|\mathbf{S}\|_0$. The above optimization problem (Program 2.1.1) is NP-hard. Both the terms in the loss function, $\text{rank}(.)$ and l_0-norm, are non-convex functions. Furthermore, the constraint on the entries of $\mathbf{L}$ (Equation 2.1.2) is a very difficult set to optimize over. We note that the objective is a low-rank matrix recovery and completion problem with sparse corruption. This problem can be relaxed to a tractable *convex program*. We consider the following convex program which aims to recover and complete the low-rank matrix ($\mathbf{L}$) from the partially observed adjacency matrix ($\mathbf{A}^{obs}$):

Simple Convex Program:

$$\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \;\; \|\mathbf{L}\|_\star + \lambda\|\mathbf{S}\|_1 \tag{2.1.4}$$

subject to

$$1 \geq \mathbf{L}_{i,j} \geq 0 \;\; \text{for all } i,j \in \{1,2,\ldots n\} \tag{2.1.5}$$

$$\mathbf{L}^{obs} + \mathbf{S}^{obs} = \mathbf{A}^{obs}, \tag{2.1.6}$$

where $\lambda \geq 0$ is the regularization parameter, $\|.\|_\star$ is the nuclear norm (sum of the singular values of the matrix), and $\|.\|_1$ is the l_1-norm (sum of absolute values of the entries of the matrix). $\mathbf{S}$ is the sparse error matrix that accounts for the missing edges inside the clusters and the erroneous edges outside the clusters on the observed entries. $\mathbf{L}^{obs}$ and $\mathbf{S}^{obs}$ denote entries of $\mathbf{L}$ and $\mathbf{S}$ that correspond to the observed part of the adjacency matrix.

Program 2.1.4 is very simple and intuitive. Further, it does not require any information other than the observed part of the adjacency matrix. In [Che+14], the authors analyze Program 2.1.4 without the constraint (2.1.5). While dropping (2.1.5) makes the convex program less effective, it does allow [Che+14] to make use of low-rank matrix completion results for its analysis. In [OH11] and [VOH14b], the authors analyze Program 2.1.4 when the entire adjacency matrix is observed. In [CSX12], the authors study a slightly more general program, where the regularization parameter is different for the extra edges and the missing edges. However, the adjacency matrix is completely observed.

It is not difficult to see that, when the edge probability inside the clusters is $p < \frac{1}{2}$, (as $n \to \infty$) Program 2.1.4 will return $\mathbf{L}^0 = 0$ as the optimal solution (since if the cluster is not dense enough it is more costly to complete the missing edges). As a result our analysis of Program 2.1.4, and the main result of Theorem 1 (Section 2.3.1), assumes $p > 1/2$. Clearly, there are many instances of graphs we would like to cluster where $p < 1/2$. If the total size of the cluster region (i.e, the total number of edges in the cluster, denoted by $|\mathcal{R}|$) is known, then the following convex program can be used, and can be shown to work for $p < 12$ (see Theorem 2 in Section 2.3.2).

Improved Convex Program:

$$\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \ \ \|\mathbf{L}\|_\star + \lambda \|\mathbf{S}\|_1 \tag{2.1.7}$$

subject to

$$1 \geq \mathbf{L}_{i,j} \geq \mathbf{S}_{i,j} \geq 0 \ \text{ for all } i,j \in \{1,2,\dots n\} \tag{2.1.8}$$

$$\mathbf{L}_{i,j} = \mathbf{S}_{i,j} \text{ whenever } \mathbf{A}^{obs}_{i,j} = 0 \tag{2.1.9}$$

$$\text{sum}(\mathbf{L}) \geq |\mathcal{R}|. \tag{2.1.10}$$

As before, $\mathbf{L}$ is the low-rank matrix corresponding to the ideal cluster structure and $\lambda \geq 0$ is the regularization parameter. However, $\mathbf{S}$ is now the sparse error matrix that accounts only for the missing edges inside the clusters on the observed part of adjacency matrix. [OH11] and [Ame13] study programs similar to Program 2.1.7 for the case of a completely observed adjacency matrix. In [Ame13], the constraint 2.1.10 is a strict equality. In [AV11] the authors analyze a program close to Program 2.1.7 but without the l_1 penalty.

If $\mathcal{R}$ is not known, it is possible to solve Problem 2.1.7 for several values of $\mathcal{R}$ until the desired performance is obtained. Our empirical results reported in Section 2.4,

suggest that the solution is not very sensitive to the choice of $\mathcal{R}$.

2.1.2 Our Contributions

Our contributions are multifold:

1. We analyze the Simple Convex Program 2.1.4 for the SBM with partial observations. We provide *explicit bounds* on the regularization parameter as a function of the parameters of the SBM, that characterizes the success and failure conditions of Program 2.1.4 (see results in Section 2.3.1). We show that clusters that are either too small or too sparse constitute the bottleneck. Our analysis is helpful in understanding the **phase transition** from failure to success for the simple approach.

2. We also analyze the Improved Convex Program 2.1.7. We explicitly characterize the conditions on the parameters of the SBM and the regularization parameter for successfully recovering clusters using this approach (see results in Section 2.3.2).

3. Apart from providing theoretical guarantees and corroborating them with simulation results (Section 2.4), we also apply Programs 2.1.4 and 2.1.7 on a real data set (Section 2.4.3) obtained by crowdsourcing an image labeling task on Amazon Mechanical Turk.

2.1.3 Related Work

In [Che+14], the authors consider the problem of identifying clusters from partially observed unweighted graphs. For the SBM with partial observations, they analyze Program 2.1.4 without constraint (2.1.5), and show that under certain conditions, the minimum cluster size must be at least $\mathcal{O}(\sqrt{n(\log(n))^4/r})$ for successful recovery of the clusters. Unlike our analysis, the exact requirement on the cluster size is not known (since the constant of proportionality is not known). Also they do not provide conditions under which the approach fails to identify the clusters. Finding the explicit bounds on the constant of proportionality is critical to understanding the phase transition from failure to successfully identifying clusters.

In [AV14; AV11; OH11; CSX12; Ame13], analyze convex programs similar to the Programs 2.1.4 and 2.1.7 for the SBM and show that the minimum cluster size should be at least $\mathcal{O}(\sqrt{n})$ for successfully recovering the clusters. However, the exact requirement on the cluster size is not known. Also, they do not provide explicit

conditions for failure, and except for [OH11] they do not address the case when the data is missing.

In contrast, we consider the problem of clustering with missing data. We explicitly characterize the constants by providing bounds on the model parameters that decide if Programs 2.1.4 and 2.1.7 can successfully identify clusters. Furthermore, for Program 2.1.4, we also explicitly characterize the conditions under which the program fails.

In [OH11], the authors extend their results to partial observations by scaling the edge probabilities by r (observation probability), which will *not* work for $r < 1/2$ or $1/2 < p < 1/2r$ in Program 2.1.4. We first analyzed Program 2.1.4 for the SBM and provided conditions for success and failure of the program when the entire adjacency matrix is observed [VOH14b]. In this chapter, we start with the partially observed case [VOH14a] and recover the results for the fully observed case as a special case. The dependence on the number of observed entries emerges non-trivially in our analysis.

2.2 Generative Model for Partially Observed Graphs

In this section we describe the model we consider for the partially observed unweighted graphs with clusters which builds on the popular Stochastic Block Model (SBM) [HLL83] or the planted partition model [CK01]. Definition of SBM is given below:

Definition 2.2.1 (Stochastic Block Model)**.** *Let $\mathbf{A} = \mathbf{A}^T$ be the adjacency matrix of a graph on n nodes with K disjoint clusters of size n_i each, $i = 1, 2, \cdots, K$. Let $1 \geq p_i \geq 0,\ i = 1, \cdots, K$ and $1 \geq q \geq 0$. For $1 \leq l < m \leq n$, edge between the nodes l and m exists independently and,*

1. *if l and m are in the same cluster i, then*

$$\mathbf{A}_{l,m} = \begin{cases} 1 \text{ with probability } p_i, \\ 0 \text{ with probability } 1 - p_i. \end{cases}$$

2. *if l and m are not in the same cluster, then*

$$\mathbf{A}_{l,m} = \begin{cases} 1 \text{ with probability } q, \\ 0 \text{ with probability } 1 - q. \end{cases}$$

If $p_i > q$ for each i, then the average density of edges is higher inside the clusters compared to outside.

To capture partial observations, on top of the SBM, we assume that each edge is independently observed with probability r. We say the random variable Y has a $\Phi(r, \delta)$ distribution, for $0 \leq \delta, r \leq 1$, written as $Y \sim \Phi(r, \delta)$, if

$$Y = \begin{cases} 1, \text{ w.p. } r\delta \\ 0, \text{ w.p. } r(1-\delta) \\ *, \text{ w.p. } (1-r), \end{cases}$$

where $*$ denotes unknown. A formal definitio of the partial observation model considered in this chapter is as follows:

Definition 2.2.2 (Partial Observation Model)**.** *Let* $\mathbf{A}$ *be the adjacency matrix of a random graph generated according to the Stochastic Block Model of Definition 2.2.1. Let* $0 < r \leq 1$*. Each entry of the adjacency matrix* $\mathbf{A}$ *is observed independently with probability* r*. Let* $\mathbf{A}^{obs}$ *denote the observed adjacency matrix. Then for* $l > m$*:* $(\mathbf{A}^{obs})_{l,m} \sim \Phi(r, p_i)$ *if both the nodes* l *and* m *belong to the same cluster* i*. Otherwise,* $(\mathbf{A}^{obs})_{l,m} \sim \Phi(r, q)$*.*

Note that the model described can handle *outliers* (nodes that do not belong any clusters). So, $\sum_{i=1}^{K} n_i \leq n$.

2.3 Exact Recovery Guarantees

In this section we present the exact recovery guarantees for the convex programs (Program 2.1.4 and Program 2.1.7). Before we state the results formally, we need some notations. We deonte the set $\{1, 2, \cdots, n\}$ by $[n]$. Let the union of regions induced by the clusters be denoted by $\mathcal{R}$ and its complement $\mathcal{R}^c = [n] \times [n] - \mathcal{R}$. Note that $|\mathcal{R}| = \sum_{i=1}^{K} n_i^2$ and $|\mathcal{R}^c| = n^2 - \sum_{i=1}^{K} n_i^2$. Let the minimum edge density inside the clusters be $p_{min} := \min_{1 \leq i \leq K} p_i$, the minimum cluster size be $n_{\min} := \min_{1 \leq i \leq K} n_i$, and the maximum cluster size be $n_{\max} := \max_{1 \leq i \leq K} n_i$.

We note that, in this chapter, when we say a convex program *succeeds*, we mean that it *exactly* recovers the adjacency matrix corresponding to the underlying cluster structure, that is, the matrix $\mathbf{L}^0$, such that,

$$\mathbf{L}^0_{lm} = \begin{cases} 1, \text{ if } l \text{ and } m \text{ are in the same cluster, and} \\ 0, \text{ otherwise.} \end{cases}$$

2.3.1 Simple Convex Program

Recall the Simple Convex Program (Program 2.1.4):

$$\begin{aligned} &\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \;\; \|\mathbf{L}\|_\star + \lambda \|\mathbf{S}\|_1 \\ &\text{subject to} \\ &\quad 1 \geq \mathbf{L}_{i,j} \geq 0 \;\; \text{for all } i,j \in [n] \\ &\quad \mathbf{L}^{obs} + \mathbf{S}^{obs} = \mathbf{A}^{obs}. \end{aligned}$$

In a nutshell, our analysis shows that

the clusters that are both small and sparse are the bottlenecks.

The quantity that determines whether Program 2.1.4 can recover a cluster exactly is its *effective density* which captures both its the size and sparsity, defined as

$$\mathbf{D}_i := n_i \, r \, (2p_i - 1).$$

Let the smallest effective density be, $\mathbf{D}_{\min} = \min_{1 \leq i \leq K} \mathbf{D}_i$. There are two thresholds on the minimum effective density that determine the conditions for succeess and failure of Program 2.1.4. The threshold for success is defined as follows,

$$\begin{aligned} \mathbf{\Lambda}_{\text{succ}}^{-1} := \; & 2r\sqrt{n}\sqrt{\frac{1}{r} - 1 + 4q(1-q)} \\ & + \max_{1 \leq i \leq K} 2r\sqrt{n_i}\sqrt{2(\frac{1}{r} - 1) + 4\left(q(1-q) + p_i(1-p_i)\right)}. \end{aligned}$$

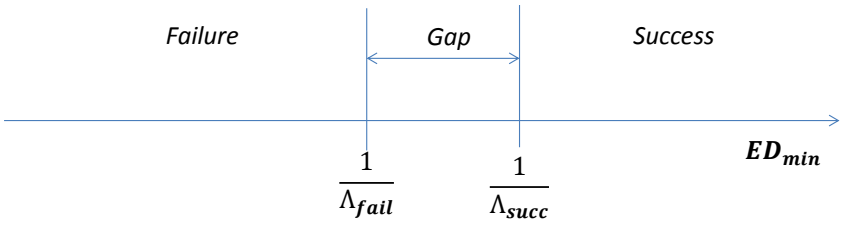
Threshold for failure is defined as follows:

$$\mathbf{\Lambda}_{\text{fail}}^{-1} := \sqrt{rq\left(1 - \sum_{i=1}^{K} \left(\frac{n_i}{n}\right)^2\right) n}.$$

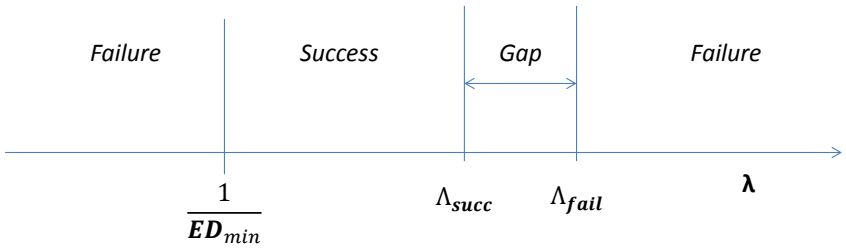
We note that the thresholds, $\mathbf{\Lambda}_{\text{succ}}$ and $\mathbf{\Lambda}_{\text{fail}}$ depend only the parameters of the model and simple algebra shows that $\mathbf{\Lambda}_{\text{fail}}^{-1} < \mathbf{\Lambda}_{\text{succ}}^{-1}$.

For random graphs generated according to the Partial Observation Model of Definition (2.2.2) with K disjoint clusters of sizes $\{n_i\}_{i=1}^K$, and probabilities $\{p_i\}_{i=1}^K$ and q, such that $p_{min} > \frac{1}{2} > q > 0$, the following theorem provides exact recovery guarantees for Program 2.1.4.

Theorem 1 (Simple Program)**.** *Whenever* $\mathbf{D}_{\min} > \mathbf{\Lambda}_{succ}^{-1}$*, for any* $\lambda \in \left(\mathbf{D}_{\min}^{-1}, \; \mathbf{\Lambda}_{succ}\right)$*, Program 2.1.4 succeeds with high probability. For any* $\lambda \geq 0$*, if* $\mathbf{D}_{\min} < \mathbf{\Lambda}_{fail}^{-1}$*, then Program 2.1.4 fails with high probability.*



(a) Feasibility of Program 2.1.4 in terms of the minimum effective density ($\mathbf{D}_{\min}$).



(b) Feasibility of Program 2.1.4 in terms of the regularization parameter (λ).

Figure 2.2: Characterization of the feasibility of Program (2.1.4) in terms of the minimum effective density and the value of the regularization parameter. The feasibility is determined by the values of these parameters in comparison with two constants $\mathbf{\Lambda}_{\text{succ}}$ and $\mathbf{\Lambda}_{\text{fail}}$, derived in Theorem 1 and Theorem 2. The thresholds guaranteeing the success or failure of Program 2.1.4 derived in this paper are fairly close to each other.

The high probability in the above theorem holds with at least $1-c_1 n^2 \exp\left(-c_2 n_{\min}\right)$, where c_1, c_2 are positive constants. In terms of the regularization paramter $\lambda \geq 0$, the success and failure of Program 2.1.4 can be stated as follows:

1. If $\lambda > \mathbf{\Lambda}_{\text{fail}}$, then Program 2.1.4 fails to correctly recover the clusters with probability at least $1 - c_1' \exp(-c_2' |\mathcal{R}^c|)$.

2. If $\mathbf{D}_{\min} > \mathbf{\Lambda}_{\text{succ}}^{-1}$, then,

   - for $\lambda \in \left(\mathbf{D}_{\min}^{-1}, \mathbf{\Lambda}_{\text{succ}}\right)$, then Program 2.1.4 succeeds in correctly recovering the clusters with probability at least $1 - c_1' n^2 \exp(-c_2' n_{\min})$.
   - If $\lambda < \mathbf{D}_{\min}^{-1}$, then Program 2.1.4 fails to correctly recover the clusters with probability at least $1 - c_1' \exp(-c_2' n_{\min})$.

Figure 2.2 depicts these conditions. A detailed proof of the results presented above is in the appendix A.

Theorem 1 characterizes the success and failure of Program 2.1.4 as a function of the regularization parameter λ. In particular, if $\lambda > \mathbf{\Lambda}_{\text{fail}}$, Program 2.1.4 fails with

high probability. If $\lambda < \mathbf{\Lambda}_{\text{succ}}$, Program 2.1.4 succeeds with high probability *if and only if* $\mathbf{D}_{\min} > \frac{1}{\lambda}$. However, Theorem 1 has nothing to say about $\mathbf{\Lambda}_{\text{succ}} < \lambda < \mathbf{\Lambda}_{\text{fail}}$ (Figure 2.2b).

The sufficient condition on $\mathbf{D}_{\min}$ from Theorem 1 is:

$$\min_{1\le i\le K} n_i\, r\,(2p_i-1) > 2r\sqrt{n}\sqrt{\frac{1}{r}-1+4q(1-q)} + \max_{1\le i\le K} 2r\sqrt{n_i}\sqrt{2\left(\frac{1}{r}-1\right)+4\left(q(1-q)+p_i(1-p_i)\right)}, \tag{2.3.1}$$

which can be re-written by eliminating the observation probability r from the LHS as follows,

$$\min_{1\le i\le K} n_i\,(2p_i-1) > 2\sqrt{n}\sqrt{\frac{1}{r}-1+4q(1-q)} + \max_{1\le i\le K} 2\sqrt{n_i}\sqrt{2\left(\frac{1}{r}-1\right)+4\left(q(1-q)+p_i(1-p_i)\right)}.$$

Now using the above equation as a reference, we discuss the effect of various parameters of the SBM on the sufficient condition for success of Program 2.1.4:

1. **Effect of observation probability (r):** Clearly, the smaller the r, the larger is the RHS. This in turn means that the cluster size and density inside the clusters (LHS) need to be larger as well. In other words, if the clusters are small and sparse, then we require more observations to recover them.

2. **Effect of edge density between the clusters (q):** As q tends closer to $\frac{1}{2}$, the variance terms, $q(1-q)$, on the RHS increases.

3. **Effect of edge density inside the clusters (p_i):** As p_i tends closer to $\frac{1}{2}$, the effective density (LHS) tends to 0 and the variance term, $p_i(1-p_i)$, in the RHS increases.

4. **Effect of cluster size**: When p, q, r are constants, then the condition in Equation 2.3.1 will be

   $$n_{\min} \ge \Omega(\sqrt{n}).$$

   This matches the small cluster size bottleneck that has been observed in the analysis of several other computationally efficient clustering algorithms. It is conjectured that this might be a necessary condition (referred to as *hidden clique conjecture*).

5. **Effect of relative cluster sizes**: Note that the relative size of clusters is not a part of the assumptions or the conditions for exact recovery. We do not need any assumptions on the relative cluster sizes for our analysis. Hence, the results hold good regardless of whether the cluster sizes are balanced or not. For example, a graph could have clusters of size $\Theta(n)$ and of size $\Theta(\sqrt{n})$ and as long as the sufficient condtion for success is satisfied, they can all be recovered.

6. **Effect of outliers**: l_1 penalty on the noise in Program 2.1.4 not only serves as a loss funciton but also is robust to outliers. Regardless of the number of outliers present, whether there are very few of them or $\Theta(n)$ of them, as long as the sufficient condition on the effective density of the clusters (which does not depend on the size of the outliers) is satisfied, the clusters can be exactly recovered.

The condition in Equation 2.3.1 can also be re-written as,

$$\min_{1\leq i\leq K} n_i \sqrt{r}\,(2p_i-1) > 2\sqrt{n}\sqrt{1-r+4rq(1-q)} + \max_{1\leq i\leq K} 2\sqrt{n_i}\sqrt{2(1-r)+4r\left(q(1-q)+p_i(1-p_i)\right)}.$$

Fully Observed Case: When we set $r=1$, we obtain the results for the case when the entire adjacency matrix is observed [VOH14b]. Equation 2.3.1 can be written as:

$$\min_{1\leq i\leq K} n_i\,(2p_i-1) > 4\sqrt{n}\sqrt{q(1-q)} + \max_{1\leq i\leq K} 4\sqrt{n_i}\sqrt{q(1-q)+p_i(1-p_i)}.$$

We note that for Program 2.1.4, one cannot simply work with the fully observed case [VOH14b] and then replace p with rp and q with rq (which is equivalent to setting all the unobserved entries to 0). This is because it will *not* work for $r<\frac{1}{2}$ or $\frac{1}{2}<p<\frac{1}{2r}$. It is important to have the constraint in Equation 2.1.6 to be over only the observed entries and construct a dual certificate for it which when $r=1$ reduces to the dual certificate for the case of the fully observed case.

Sharpness of the Performance Bounds: From the definitions, we see that there is a gap between $\mathbf{\Lambda}_{\text{fail}}$ and $\mathbf{\Lambda}_{\text{succ}}$. When $r=1$, the gap is

$$\frac{\mathbf{\Lambda}_{\text{fail}}}{\mathbf{\Lambda}_{\text{succ}}} = \frac{4\sqrt{q(1-q)n} + \max_{1\leq i\leq K} 2\sqrt{n_i}\sqrt{4\left(q(1-q)+p_i(1-p_i)\right)}}{\sqrt{q\left(n-\sum_{i=1}^{K}\left(\frac{n_i}{n}\right)^2 n\right)}}$$

times. In the small cluster regime where $\max_{1\le i\le K} n_i = o(n)$ and $\sum_{i=1}^{K} n_i^2 = o(n^2)$, the ratio $\frac{\mathbf{\Lambda}_{\text{fail}}}{\mathbf{\Lambda}_{\text{succ}}} = 4\sqrt{1-q} + o(1)$, which is at most 4 times in the worst case.

2.3.2 Improved Convex Program

Recall the Improved Convex Program (Program 2.1.7):

$$\begin{aligned}
&\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \ \ \|\mathbf{L}\|_\star + \lambda\|\mathbf{S}\|_1 \\
&\text{subject to} \\
&\quad 1 \ge \mathbf{L}_{i,j} \ge \mathbf{S}_{i,j} \ge 0 \ \text{ for all } i,j \in \{1,2,\dots n\} \\
&\quad \mathbf{L}_{i,j} = \mathbf{S}_{i,j} \text{ whenever } \mathbf{A}^{obs}_{i,j} = 0 \\
&\quad \text{sum}(\mathbf{L}) \ge |\mathcal{R}|.
\end{aligned}$$

Our analysis shows that *the clusters that are both small and sparse are the bottlenecks*. The quantity that determines whether Program 2.1.7 can recover a cluster exactly is its *effective density* which captures both its the size and sparsity, defined as

$$\tilde{\mathbf{D}}_i := n_i\, r\, (p_i - q).$$

Notice that in contrast with Program 2.1.4 where we required $p_i > \frac{1}{2} > q$, for Program 2.1.7, we only need $p_i > q$. Let the smallest effective density be, $\tilde{\mathbf{D}}_{\min} = \min_{1\le i\le K} \tilde{\mathbf{D}}_i$. The threshold for success is defined as follows:

$$\begin{aligned}
\tilde{\mathbf{\Lambda}}^{-1}_{\text{succ}} := {} & 2r\sqrt{n}\sqrt{\left(\frac{1}{r} - 1 + q\right)(1-q)} \\
& + 2r \max_{1\le i\le K} \sqrt{n_i}\sqrt{(1-p_i)\left(\frac{1}{r} - 1 + p_i\right) + (1-q)(\frac{1}{r} - 1 + q)}.
\end{aligned}$$

We note that the threshold, $\tilde{\mathbf{\Lambda}}_{\text{succ}}$ depends only on the parameters of the model.

For random graphs generated according to the Partial Observation Model of Definition (2.2.2) with K disjoint clusters of sizes $\{n_i\}_{i=1}^K$, and probabilities $\{p_i\}_{i=1}^K$ and q, such that $p_{min} > q > 0$, the following theorem provides exact recovery guarantees for Program 2.1.7.

Theorem 2 (Improved Program)**.** *When* $0 \le \tilde{\mathbf{D}}^{-1}_{\min} < \lambda < \tilde{\mathbf{\Lambda}}_{succ}$*, Program 2.1.7 succeeds in recovering the clusters with high probability.*

We note that unlike in the case of Program 2.1.4, we do not have a condition for failure for Program 2.1.7. Theorem 2 gives a sufficient condition for the success

of Program 2.1.7 as a function of λ. In particular, for any $\lambda > 0$, we succeed if $\tilde{\mathbf{D}}_{\min}^{-1} < \lambda < \tilde{\mathbf{\Lambda}}_{\text{succ}}$. However, it does not comment on what happens when this condition does not hold . When we say *high probability* in Theorem 2, we mean it holds with probability at least $1 - c_1' n^2 \exp(-c_2' n_{\min})$, where c_1', c_2' are positive constants. Detailed proof for Theorem 2 is provided in the appendix A.

The sufficient condition on the minimum effective density $\tilde{\mathbf{D}}_{\min}$ from Theorem 2 is

$$\min_{1\le i\le K} n_i\, r\,(p_i - q) > 2r\sqrt{n}\sqrt{\left(\frac{1}{r} - 1 + q\right)(1-q)} + 2r \max_{1\le i\le K}\sqrt{n_i}\sqrt{(1-p_i)\left(\frac{1}{r} - 1 + p_i\right) + (1-q)\left(\frac{1}{r} - 1 + q\right)}. \tag{2.3.2}$$

Note that in comparison to the sufficient condition for success of Program 2.1.4 (Equation 2.3.1) where the term $2p_i - 1$ appears in the LHS (so $p_i > \frac{1}{2} > q$ was needed), the sufficient condition for success of Program 2.1.7 (Equation 2.3.2 has the term $p_i - q$ in the LHS (so we only need $p_i > q$). Apart from this major difference, the effect of observation probability r, edge density inside the clusters p_i and between the clusters q, cluster sizes and outliers are qualitatively similar to that in the case of Program 2.1.4.

Fully Observed Case: When we set $r = 1$, we obtain the results for the case when the entire adjacency matrix is observed. Equation 2.3.2 can be written as:

$$\min_{1\le i\le K} n_i\,(p_i - q) > 2\sqrt{n}\sqrt{q(1-q)} + 2\max_{1\le i\le K}\sqrt{n_i}\sqrt{p_i(1-p_i) + q(1-q)}.$$

(p, q) as a function of n: We now briefly discuss the regime in which cluster sizes are large (i.e. $\mathcal{O}(n)$) and we are interested in the parameters (p, q) as a function of n that allows proposed approaches to be successful. Critical to Program 2.1.7 is the constraint (2.1.9): $\mathbf{L}_{i,j} = \mathbf{S}_{i,j}$ when $\mathbf{A}^{obs}_{i,j} = 0$ (which is the only constraint involving the adjacency $\mathbf{A}^{obs}$). With missing data, $\mathbf{A}^{obs}_{i,j} = 0$ with probability $r(1-p)$ inside the clusters and $r(1-q)$ outside the clusters. Defining $\hat{p} = rp + 1 - r$ and $\hat{q} = rq + 1 - r$, the number of constraints in (2.1.9) becomes statistically equivalent to those of a *fully observed* graph where p and q are replaced by $\hat{p}$ and $\hat{q}$. Consequently, for a fixed $r > 0$, we require $p \ge p - q \gtrsim \mathcal{O}(\frac{1}{\sqrt{n}})$ for success. However, setting the unobserved entries to 0 yields $\mathbf{A}_{i,j} = 0$ with probability $1 - rp$ inside the clusters and $1 - rq$ outside the clusters. This is equivalent to a fully observed graph where p and q are replaced by rp and rq. In this case, we can allow

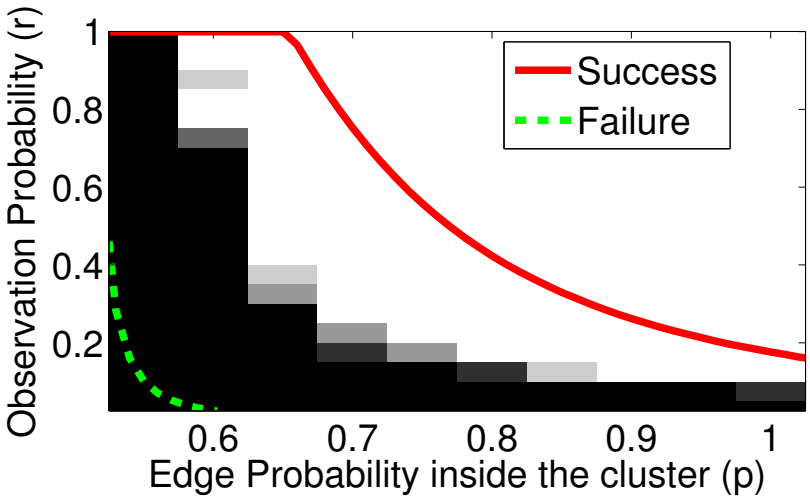


Figure 2.3: Region of success (white region) and failure (black region) of Program 2.1.4 with $\lambda = 1.01\mathbf{D}_{\min}^{-1}$. The solid red curve is the threshold for success ($\lambda < \Lambda_{\text{succ}}$) and the dashed green line which is the threshold for failure ($\lambda > \Lambda_{\text{fail}}$) as predicted by Theorem 1.

$p \approx \mathcal{O}(\frac{polylog(n)}{n})$ for success which is order-wise better, and matches closely to the results in McSherry [McS01]. Intuitively, clustering a fully observed graph with parameters $\hat{p} = rp + 1 - r$ and $\hat{q} = rq + 1 - r$ is much more difficult than one with rp and rq, since the links are *more noisy* in the former case. Therefore, *while it is beneficial to leave the unobserved entries blank in Program 2.1.4, for Program 2.1.7 it is in fact beneficial to set the unobserved entries to* 0. So, the modified program would have the following sufficient condition for success of Program 2.1.7 by setting the unobserved entries to 0 as follows:

$$\min_{1\le i\le K} rn_i\,(p_i - q) > 2r\sqrt{n}\sqrt{rq(1-rq)} + 2r\max_{1\le i\le K}\sqrt{n_i}\sqrt{rp_i(1-rp_i) + rq(1-rq)}.$$

2.3.3 Planted Clique Problem

A fundamental problem related to the hardness of clustering is the *planted clique* problem. Consider an Erdös-Réyni random graph [1] with edge probability $\frac{1}{2}$. A subset of nodes in this graph is picked and all of them are connected to each other to obtain a clique. Given such a graph (the identity of the nodes that form the clique are unknown), the question of interest is whether we can find the clique efficiently (in polynomial time). The planted clique problem is well-studied in the theoretical computer science community and to the best of our knowledge the tightest result

[1] An Erdös-Rényi random graph on n nodes with parameter p is a random graph where each edge exists independently with probability p.

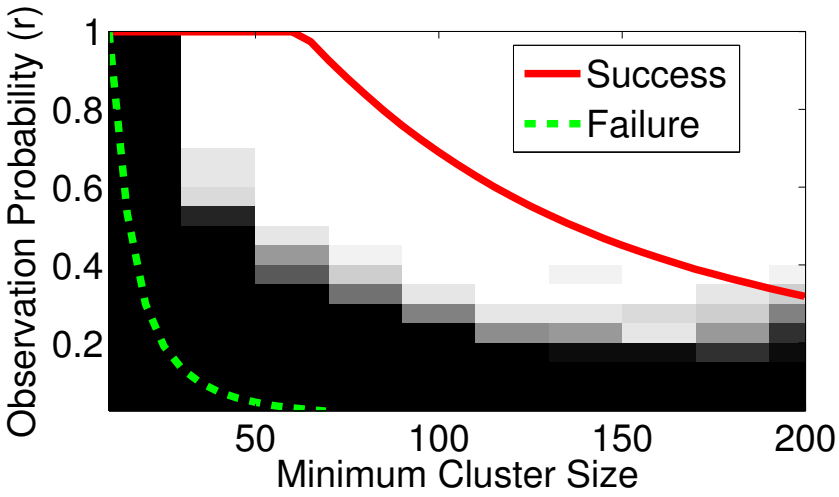


Figure 2.4: Region of success (white region) and failure (black region) of Program 2.1.4 with $\lambda = 1.01\mathbf{D}_{\min}^{-1}$. The solid red curve is the threshold for success ($\lambda < \Lambda_{\text{succ}}$) and the dashed green line which is the threshold for failure ($\lambda > \Lambda_{\text{fail}}$) as predicted by Theorem 1.

(without any unknown constants) is provided in [DM15] where the sufficient condition on the size of the clique (number of nodes in the clique) for recovering it successfully is

$$n_{\min} > \sqrt{\frac{n}{e}} \approx 0.61\sqrt{n},$$

using asymototic message passing analysis tailored for the planted clique problem. In contrast, the planted clique problem is a special case for the SBM, where the number of clusters $K = 1$, the edge density inside the clusters is $p = 1$ and the edge density between he clusters $q = \frac{1}{2}$. Thus, by plugging in these parameters in our exact recovery conditions we obtain the following sufficient condtion on the size of the planted clique for its exact recovery,

$$n_{\min} > 2\sqrt{n}.$$

2.4 Experimental Results

We implement Program 2.1.4 and 2.1.7 using the inexact augmented Lagrange method of multipliers [LCM10; LLS11]. Note that this method solves the Program 2.1.4 and 2.1.7 approximately. Further, the numerical imprecisions will prevent the entries of the output of the algorithms from being strictly equal to 0 or 1. We use the mean of all the entries of the output as a hard threshold to round

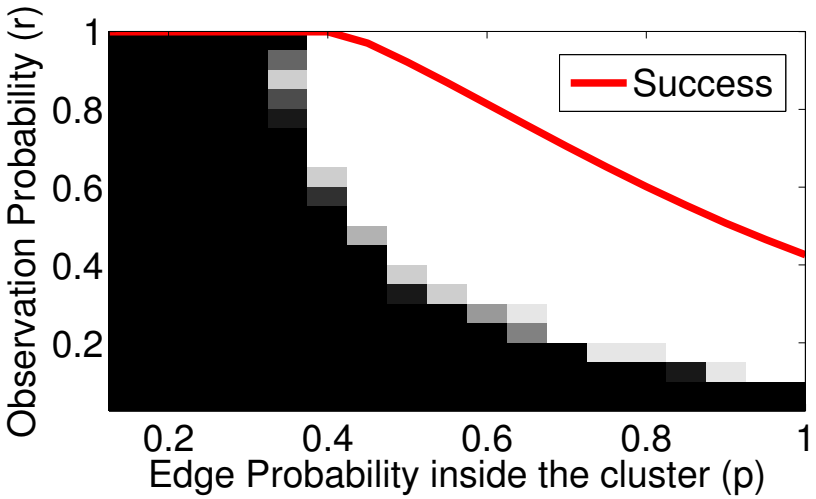


Figure 2.5: Region of success (white region) and failure (black region) of Program 2.1.7 with $\lambda = 0.49\tilde{\mathbf{\Lambda}}_{\text{succ}}$. The solid red curve is the threshold for success ($\tilde{\mathbf{D}}_{\min} > lambda^{-1}$) as predicted by Theorem 2.

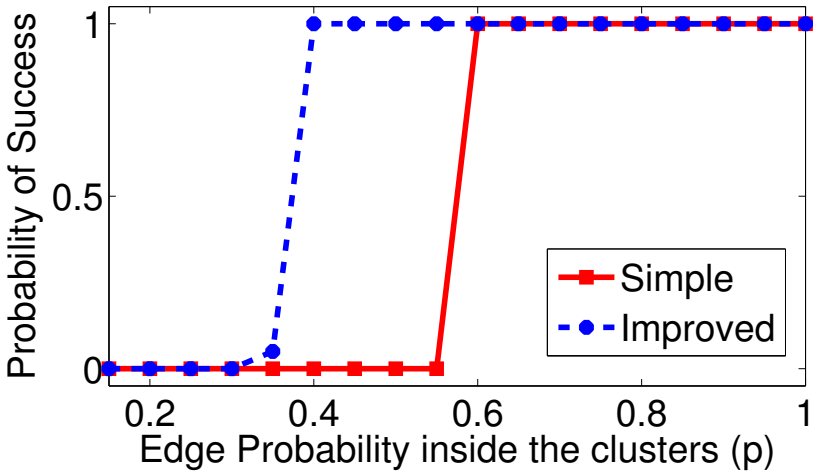


Figure 2.6: Comparison range of edge probability p for Simple Program 2.1.4 and Improved Program 2.1.7.

each entry. That is, if an entry is less than the threshold, it is rounded to 0 and to 1 otherwise. We compare the output of the algorithm after rounding to the optimal solution ($\mathbf{L}^0$), and declare success if the number of wrong entries is less than 0.1%.

Set Up: We consider at an unweighted graph on $n = 600$ nodes with 3 disjoint clusters. For simplicity the clusters are of equal size $n_1 = n_2 = n_3$, and the edge probability inside the clusters are same $p_1 = p_2 = p_3 = p$. The edge probability outside the clusters is fixed, $q = 0.1$. We generate the adjacency matrix randomly according to the Stochastic Block Model 2.2.1 and Partial Observation Model 2.2.2.

All the results are an average over 20 experiments.

2.4.1 Simulations for Simple Convex Program

Dependence between r and p: In the first set of experiments we keep $n_1 = n_2 = n_3 = 200$, and vary p from 0.55 to 1 and r from 0.05 to 1 in steps of 0.05.

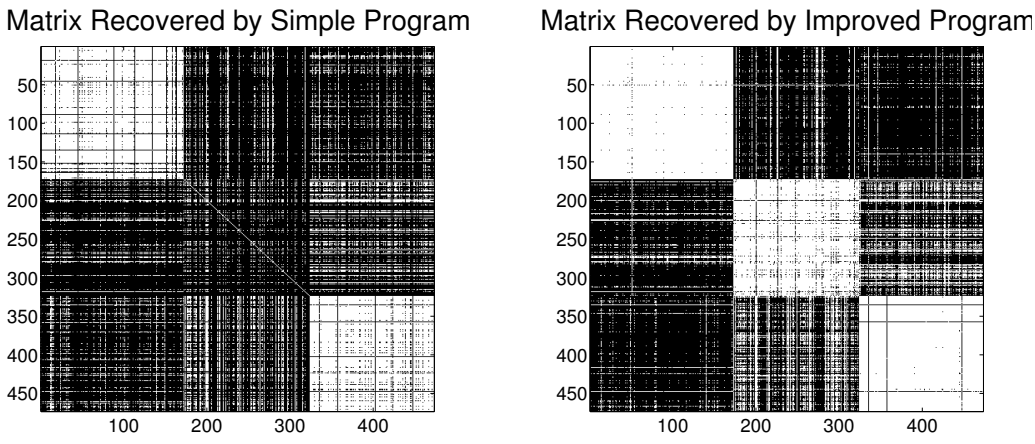
Dependence between $n_{\min}$ and r: In the second set of experiments we keep the edge probability inside the clusters fixed, $p = 0.85$. The cluster size is varied from $n_{\min} = 20$ to $n_{\min} = 200$ in steps of 20 and r is varied from 0.05 to 1 in steps of 0.05.

In both the experiments, we set the regularization parameter $\lambda = 1.01\mathbf{D}_{\min}^{-1}$, ensuring that $\mathbf{D}_{\min} > 1/\lambda$, enabling us to focus on observing the transition around $\mathbf{\Lambda}_{\text{succ}}$ and $\mathbf{\Lambda}_{\text{fail}}$. The outcome of the experiments are shown in the Figures 2.3 and 2.4. The experimental region of success is shown in white and the region of failure is shown in black. The theoretical region of success is above the solid red curve ($\lambda < \mathbf{\Lambda}_{\text{succ}}$) and the region of failure is below the dashed green curve ($\lambda > \mathbf{\Lambda}_{\text{fail}}$). As we can see, the transition indeed occurs between the two thresholds $\mathbf{\Lambda}_{\text{succ}}$ and $\mathbf{\Lambda}_{\text{fail}}$.

2.4.2 Simulations for Improved Convex Program

We keep the cluster size, $n_1 = n_2 = n_3 = 200$ and vary p from 0.15 to 1 and r from 0.05 to 1 in steps of 0.05. We set the regularization parameter, $\lambda = 0.49\tilde{\mathbf{\Lambda}}_{\text{succ}}$, ensuring that $\lambda < \tilde{\mathbf{\Lambda}}_{\text{succ}}$, enabling us to focus on observing the condition of success around $\tilde{\mathbf{D}}_{\min}$. The outcome of this experiment is shown in the Figure 2.5. The experimental region of success is shown in white and the region of failure is shown in black. The theoretical region of success is above solid red curve.

Comparison with the Simple Convex Program: In this experiment, we are interested in observing the range of p for which the Programs 2.1.4 and 2.1.7 work. Keeping the cluster size $n_1 = n_2 = n_3 = 200$ and $r = 1$, we vary the edge probability inside the clusters from $p = 0.15$ to $p = 1$ in steps of 0.05. For each instance of the adjacency matrix, we run both Program 2.1.4 and 2.1.7. We plot the probability of success of both the algorithms in Figure 2.6. As we can observe, Program 2.1.4 starts succeeding only after $p > 1/2$, whereas for Program 2.1.7 it starts at $p \approx 0.35$.



(a) (b)

Figure 2.7: Result of using (a) Program 2.1.4 (Simple) and (b) Program 2.1.7 (Improved) on the real data set.

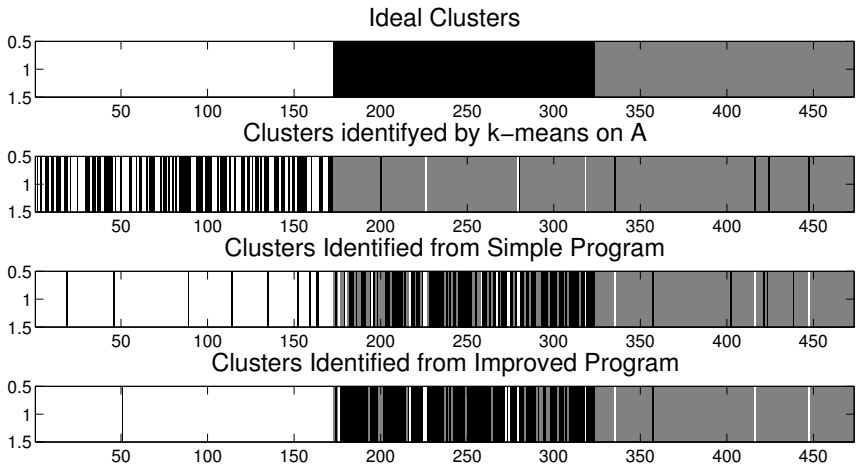


Figure 2.8: Comparing the clustering output after running Program 2.1.4 and Program 2.1.7 with the output of applying k-means clustering directly on A (with unknown entries set to 0).

2.4.3 Labeling Images: Amazon MTurk Experiment

Creating a training dataset by labeling images is a tedious task. It would be useful to crowdsource this task instead. Consider a specific example of a set of images of dogs of different breeds. We want to cluster them such that the images of dogs of the same breed are in the same cluster. One could show a set of images to each worker, and ask him/her to identify the breed of dog in each of those images. But such a task would require the workers to be experts in identifying the dog breeds.

A relatively reasonable task is to ask the workers to compare pairs of images, and for each pair, answer whether they think the dogs in the images are of the same breed or not. If we have n images, then there are $\binom{n}{2}$ distinct pairs of images, and it will pretty quickly become unreasonable to compare all possible pairs. This is an example where we could obtain a subset of the data and try to cluster the images based on the partial observations.

Image Data Set: We used images of 3 different breeds of dogs : Norfolk Terrier (172 images), Toy Poodle (151 images), and Bouvier des Flandres (150 images) from the Standford Dogs Dataset [Kho+11]. We uploaded all the 473 images of dogs on an image hosting server (we used imgur.com).

MTurk Task: We used Amazon Mechanical Turk [BKG11] as the platform for crowdsourcing. For each worker, we showed 30 pairs of images chosen randomly from the $\binom{n}{2}$ possible pairs. The task assigned to the worker was to compare each pair of images, and answer whether they think the dogs belong to the same breed or not. If the worker's response is a "yes", then there we fill the entry of the adjacency matrix corresponding to the pair as 1, and 0 if the answer is a "no".

Collected Data: We recorded around 608 responses. We were able to fill $16,750$ out of $111,628$ entries in $\mathbf{A}$. That is, we observed 15% of the total number of entries. Compared with true answers (which we know a priori), the answers given by the workers had around 23.53% errors (3941 out of 16750). The empirical parameters for the partially observed graph thus obtained are shown Table 2.1.

We ran Program 2.1.4 and Program 2.1.7 with regularization parameter, $\lambda = 1/\sqrt{n}$. Further, for Program 2.1.7, we set the size of the cluster region, $\mathcal{R}$ to 0.125 times $\binom{n}{2}$. Figure 2.7 shows the recovered matrices. Entries with value 1 are depicted by white and 0 is depicted by black. In Figure 2.8 we compare the clusters output by running the k-means algorithm directly on the adjacency matrix $\mathbf{A}$ (with unknown entries set to 0) to that obtained by running k-means algorithm on the matrices recovered after running Program 2.1.4 (Simple Program) and Program 2.1.7 (Improved Program) respectively. The overall error with k-means was 40.8% whereas the error significantly reduced to 15.86% and 7.19% respectively when we used the matrices recoverd from Programs 2.1.4 and 2.1.7 respectively (see Table 2.2). Further, note that for running the k-means algorithm we need to know the exact number of clusters. A common heuristic is to identify the top K eigenvalues that are much larger than the rest. In Figure 2.9 we plot the sorted eigenvalues for the adjacency matrix $\mathbf{A}$ and the recovered matrices. We can see that the top 3 eigen val-

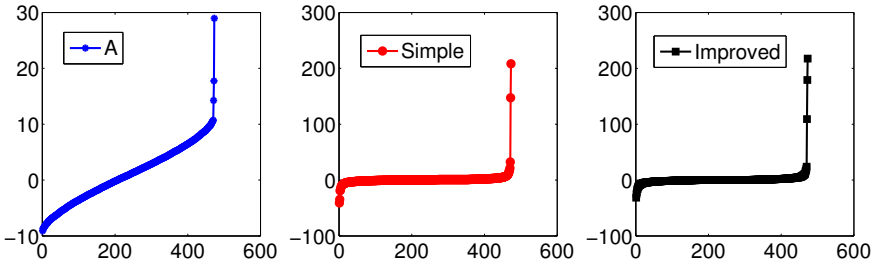


Figure 2.9: Plot of sorted eigen values for (1) Adjacency matrix with unknown entries filled by 0, (2) Recovered adjacency matrix from Program 2.1.4, (3) Recovered adjacency matrix from Program 2.1.7

Table 2.1: Empirical Parameters from the real data.

Params	Value	Params	Value
n	473	r	0.1500
K	3	q	0.1929
n_1	172	p_1	0.7587
n_2	151	p_2	0.6444
n_3	150	p_3	0.7687

Table 2.2: Number of miss-classified images

Clusters$\rightarrow$	1	2	3	Total
K-means	39	150	4	193
Simple	9	57	8	74
Improved	1	29	4	34

ues are very easily distinguished from the rest for the matrix recovered after running Program 2.1.7.

A sample of the data is shown in Figure 2.10. We observe that factors such as color, grooming, posture, face visibility, etc. can result in confusion while comparing image pairs. Also, note that the ability of the workers to distinguish the dog breeds is neither guaranteed nor uniform. Thus, the edge probabilities inside and outside clusters are not uniform. Nonetheless, Programs 2.1.4 and Program 2.1.7, especially Program 2.1.7, are quite successful in clustering the data with only 15% observations.

2.5 Outline of the Proofs

In this section we present an outline of the proofs for the theorems stated in Section 5.3. Detailed proofs are available in Appendix A.

Figure 2.10: Sample images of three breeds of dogs that were used in the MTurk experiment.

2.5.1 Proof Outline for Theorem 1

We prove Theorem 1 by proving the following two lemmas:

Lemma 2.5.1. *If* $\lambda > \mathbf{\Lambda}_{fail}$ *or* $\mathbf{D}_{\min} < \frac{1}{\lambda}$, *then* $(\mathbf{L}^0, \mathbf{S}^0)$ *is not an optimal solution to the Program 2.1.4 with high probability.*

Lemma 2.5.2. *If* $\lambda < \mathbf{\Lambda}_{succ}$ *and* $\mathbf{D}_{\min} > \frac{1}{\lambda}$, *then* $(\mathbf{L}^0, \mathbf{S}^0)$ *is the unique optimal solution to Program 2.1.4 with high probability.*

Outline of Proof of Lemma 2.5.1: To prove the Lemma 2.5.1, we look at the Lagrange of the Program 2.1.4,

$$\mathscr{L}(\mathbf{L}, \mathbf{S}; \mathbf{M}, \mathbf{N}) = \|\mathbf{L}\|_\star + \lambda\|\mathbf{S}\|_1 + \text{trace}(\mathbf{M}(\mathbf{L} - \mathbb{1}\mathbb{1}^T)) - \text{trace}(\mathbf{NL}).$$

where $\mathbf{M}$ and $\mathbf{N}$, entry-wise non-negative, are dual variables corresponding to the inequality constraints (2.1.5).

For $\mathbf{L}^0$ to be an optimal solution to (2.1.4), it has to satisfy the KKT conditions. Therefore, the subgradient of the Lagrangian at $\mathbf{L}^0$ has to be 0, i.e.,

$$\partial\|\mathbf{L}^0\|_\star + \lambda\, \partial\|\mathbf{A}_{obs} - \mathbf{L}_{obs}{}^0\|_1 + \mathbf{M}^0 - \mathbf{N}^0 = 0,$$

where $\mathbf{M}^0$ and $\mathbf{N}^0$ are optimal dual variables, and $\partial\|\mathbf{L}^0\|_\star$ and $\partial\|\mathbf{S}^0\|_1$ are subgradients of nuclear norm and ℓ_1-norm respectively at the points $(\mathbf{L}^0, \mathbf{S}^0)$. Note that in the standard notation, $\partial h(\mathbf{x})$ denotes the set of all subgradients, i.e., the subdifferential of the function $h(.)$ at $\mathbf{x}$. We have slightly abused the notation by denoting a subgradient of the function $h(.)$ at the point $\mathbf{x}$ by $\partial h(\mathbf{x})$. In other words, $\partial h(\mathbf{x})$ has been used to denote any element in the subgradient set of $h(.)$ at $\mathbf{x}$.

We can write $\mathbf{L}^0$ as $\mathbf{L}^0 = \mathbf{U}\Lambda\mathbf{U}^T$, where $\Lambda = \text{diag}\{n_1, n_2, \ldots, n_K\}$, and $\mathbf{U} = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_K] \in \mathbb{R}^{n\times K}$, where

$$\mathbf{u}_{l,i} = \begin{cases} \frac{1}{\sqrt{n_l}} & \text{if } i \text{ is in cluster } l \\ 0 & \text{otherwise.} \end{cases}$$

The subgradient of the nuclear norm at $\mathbf{L}^0$ can be written as:

$$\partial\|\mathbf{L}^0\|_\star = \mathbf{U}\mathbf{U}^T + \mathbf{W}, \tag{2.5.1}$$

where $\mathbf{W} \in \mathcal{M}_U := \{\mathbf{X} : \mathbf{X}\mathbf{U} = \mathbf{U}^T\mathbf{X} = 0, \|\mathbf{X}\| \leq 1\}$. The subgradient of the l_1-norm at $\mathbf{S}^0$ can be written as,

$$\partial\|\mathbf{S}^0\|_1 = \text{sign}(\mathbf{S}^0) + \mathbf{Q}, \tag{2.5.2}$$

where $\mathbf{Q}_{i,j} = 0$ if $\mathbf{S}^0_{i,j} \neq 0$ and $\|\mathbf{Q}\|_\infty \leq 1$.

Further, by complementary slackness, we have the following equalities,

$$\text{trace}(\mathbf{M}^0(\mathbf{L}^0 - \mathbb{1}\mathbb{1}^T)) = 0, \text{ and } \text{trace}(\mathbf{N}^0\mathbf{L}^0) = 0.$$

Using Bernstein's inequality, the above complementary slackness conditions, the form of the solution $\mathbf{L}^0$ and $\mathbf{S}^0$ as well as their subgradient sets (Equation 2.5.1 and Equation 2.5.2), we can show that when $\lambda > \mathbf{\Lambda}_{fail}$ or $\mathbf{D}_{\min} < \frac{1}{\lambda}$, the KKT conditions cannot be satisfied for $\mathbf{L}^0$ and $\mathbf{S}^0$. This implies that $\mathbf{L}^0$ and $\mathbf{S}^0$ cannot be an optimal solution to Program 2.1.4 when $\lambda > \mathbf{\Lambda}_{fail}$ or $\mathbf{D}_{\min} < \frac{1}{\lambda}$. The details of the proof can be found in the appendix (Appendix A.1.1).

Outline of Proof of Lemma 2.5.2: To prove Lemma 2.5.2, we need to show that when $\lambda < \mathbf{\Lambda}_{succ}$ and $\mathbf{D}_{\min} > \frac{1}{\lambda}$, $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to the Program 2.1.4. So, under the assumptions in Lemma 2.5.2, we need to prove the following,

$$(\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda\,\|\mathbf{S}^0 + \mathbf{E}^S\|_1) - (\|\mathbf{L}^0\|_\star + \lambda\,\|\mathbf{S}^0\|_1) > 0,$$

for all feasible perturbations $(\mathbf{E}^L, \mathbf{E}^S)$. The LHS of the above equation can be lower bound using the subgradients for the nuclear norm and l_1-norm at $\mathbf{L}^0$ and $\mathbf{S}^0$ respectively. So, we get the following inequality:

$$(\|\mathbf{L}^0+\mathbf{E}^L\|_\star+\lambda\,\|\mathbf{S}^0+\mathbf{E}^S\|_1)-\big(\|\mathbf{L}^0\|_\star + \lambda\,\|\mathbf{S}^0\|_1\big) \geq \langle\partial\|\mathbf{L}^0\|_\star, \mathbf{E}^L\rangle+\lambda\langle\partial\|\mathbf{S}^0\|_1, \mathbf{E}^S\rangle.$$

Thus, we can show that $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to the Program 2.1.4, if we show the following:

$$\langle\partial\|\mathbf{L}^0\|_\star, \mathbf{E}^L\rangle + \lambda\langle\partial\|\mathbf{S}^0\|_1, \mathbf{E}^S\rangle > 0,$$

for all feasible perturbations $(\mathbf{E}^L, \mathbf{E}^S)$.

Using the forms of the subgradients (Equation 2.5.1 and Equation 2.5.2) and noting that since $\mathbf{L} + \mathbf{S} = \mathbf{A}$, $\mathbf{E}^L = -\mathbf{E}^S$, we obtain the following:

$$\begin{aligned}
\langle \partial \|\mathbf{L}^0\|_\star, \mathbf{E}^L \rangle + \lambda \langle \partial \|\mathbf{S}^0\|_1, \mathbf{E}^S \rangle &= \langle \mathbf{U}\mathbf{U}^T + \mathbf{W}, \mathbf{E}^L \rangle + \lambda \langle \text{sign}(\mathbf{S}^0) + \mathbf{Q}, \mathbf{E}^S \rangle \\
&= \langle \mathbf{U}\mathbf{U}^T + \mathbf{W}, \mathbf{E}^L \rangle + \lambda \langle \text{sign}(\mathbf{S}^0) + \mathbf{Q}, -\mathbf{E}^L \rangle \\
&= \langle \mathbf{W}, \mathbf{E}^L \rangle + \underbrace{\langle \mathbf{U}\mathbf{U}^T - \lambda \left(\text{sign}(\mathbf{S}^0) + \mathbf{Q} \right), \mathbf{E}^L \rangle}_{:= g(\mathbf{E}^L)} \\
&= \langle \mathbf{W}, \mathbf{E}^L \rangle + g\left(\mathbf{E}^L\right).
\end{aligned}$$

Define $f(\mathbf{E}^L, \mathbf{W}) := \langle \mathbf{W}, \mathbf{E}^L \rangle + g\left(\mathbf{E}^L\right)$. Now our goal is to show that, for all feasible perturbations $\mathbf{E}^L$,

$$f(\mathbf{E}^L, \mathbf{W}) = \langle \mathbf{W}, \mathbf{E}^L \rangle + g\left(\mathbf{E}^L\right) > 0, \tag{2.5.3}$$

where $g\left(\mathbf{E}^L\right) = \langle \mathbf{U}\mathbf{U}^T - \lambda \left(\text{sign}(\mathbf{S}^0) + \mathbf{Q} \right), \mathbf{E}^L \rangle$. In order to show the above inequality we need to choose good subgradients, which in turn boils down to carefully constructing $\mathbf{W}$ and $\mathbf{Q}$.

To construct $\mathbf{Q}$ and $\mathbf{W}$, we need some notation. Let $\mathcal{C}_i$, for $i \in [K]$ denote the set of nodes that belong to cluster i and $\mathcal{C}_{K+1}$ denote the set of outliers. Let $\mathcal{R}_{i,j} = \mathcal{C}_i \times \mathcal{C}_j$ for $1 \leq i, j \leq K+1$. One can see that $\{\mathcal{R}_{i,j}\}$ divides $[n] \times [n]$ into $(K+1)^2$ disjoint regions similar to a grid which is illustrated in the Figure 2.11. Thus, $\mathcal{R}_{i,i}$ is the region induced by i'th cluster for any $1 \leq i \leq K$. Recall that $\mathcal{R}$ deontes the

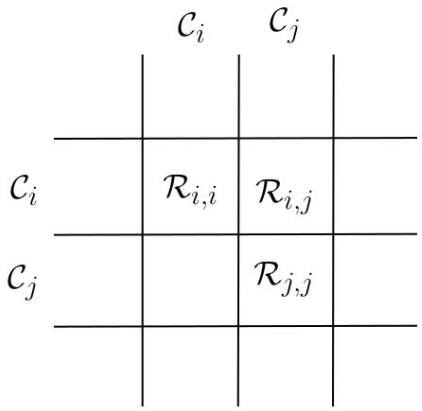


Figure 2.11: Illustration of $\{\mathcal{R}_{i,j}\}$ dividing $[n] \times [n]$ into disjoint regions similar to a grid

union of regions induced by the clusters and its complement is deonoted by $\mathcal{R}^c$. So, $\mathcal{R} = \cup_{i \in [K]} \mathcal{R}_{i,i}$ and $R^c = [n] \times [n] - \mathcal{R}$. Define $\text{sum}(\mathbf{X}) := \sum_{ij} X_{ij}$. Let $\mathcal{A}_1$ and

$\mathcal{A}_0$ are subsets of $[n] \times [n]$ and denotes the set of coordinates of $\mathbf{A}^{obs}$ that are 1 and 0 respectively. For a matrix $\mathbf{X} \in \mathbb{R}^{c \times d}$, let $\mathbf{X}_\beta$ for any $\beta \subset [c] \times [d]$ be the matrix whose entries match that of $\mathbf{X}$ in the positions $(i, j) \in \beta$ and 0 everywhere else.

Since $\mathbf{S} = \mathbf{A} - \mathbf{L}$, we have $\text{sign}(\mathbf{S}^0) = \mathbb{1}^{n\times n}_{\mathcal{A}\cap\mathcal{R}^c} - \mathbb{1}^{n\times n}_{\mathcal{A}^c\cap\mathcal{R}}$. We choose

$$\mathbf{Q} = \mathbb{1}^{n\times n}_{\mathcal{A}\cap\mathcal{R}} - \mathbb{1}^{n\times n}_{\mathcal{A}^c\cap\mathcal{R}^c}.$$

With the choice of $\mathbf{Q}$ as made above, $g\left(\mathbf{E}^L\right)$ has the following form:

$$g\left(\mathbf{E}^L\right) = \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}\left(\mathbf{E}^L_{\mathcal{R}_{i,i}}\right) + \lambda\left(\text{sum}\left(\mathbf{E}^L_{\mathcal{A}_0}\right) - \text{sum}\left(\mathbf{E}^L_{\mathcal{A}_1}\right)\right).$$

We construct $\mathbf{W} \in \mathcal{M}_U$ by projecting the following matrix on to $\mathcal{M}_U$,

$$\mathbf{W}_0 = \sum_{i=1}^{K} c_i \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}} + c\mathbb{1}^{n\times n}_{\mathcal{R}^c} + \lambda\left(\mathbb{1}^{n\times n}_{\mathcal{A}_1} - \mathbb{1}^{n\times n}_{\mathcal{A}_0}\right),$$

where $\mathbb{1}^{n\times n}$ is an all 1's matrix, $c_i = -\lambda(2p_i - 1), i = 1, 2, \ldots K$ and $c = -\lambda(2q - 1)$.

To show that under the assumptions in Lemma 2.5.2, the inequality in Equation 2.5.3 holds, we make use of the following lemma to break the next step into two cases,

Lemma 2.5.3. *Given any $\mathbf{E}^L$, assume there exists $\mathbf{W} \in \mathcal{M}_\mathbf{U}$ with $\|\mathbf{W}\| < 1$ such that $f(\mathbf{E}^L, \mathbf{W}) \geq 0$. Then at least one of the followings holds:*

1. *There exists $\mathbf{W}^* \in \mathcal{M}_\mathbf{U}$ with $\|\mathbf{W}^*\| \leq 1$ and $f(\mathbf{E}^L, \mathbf{W}^*) > 0$.*
2. *For all $\mathbf{W} \in \mathcal{M}_\mathbf{U}$, $\left\langle \mathbf{E}^L, \mathbf{W} \right\rangle = 0$.*

See appendix (Appendix A.1.2) for the proof of the above lemma.

For all $\mathbf{W} \in \mathcal{M}_\mathbf{U}$, $\langle \mathbf{E}, \mathbf{W} \rangle = 0$ is equivalent to $\mathbf{E} \in \mathcal{M}^\perp_\mathbf{U}$ which is the orthogonal complement of $\mathcal{M}_\mathbf{U}$ in $\mathbb{R}^{n\times n}$. $\mathcal{M}^\perp_\mathbf{U}$ has the following characterization:

$$\mathcal{M}^\perp_\mathbf{U} = \{\mathbf{X} \in \mathbb{R}^{n\times n} : \mathbf{X} = \mathbf{U}\mathbf{M}^T + \mathbf{N}\mathbf{U}^T \text{ for some } \mathbf{M}, \mathbf{N} \in \mathbb{R}^{n\times K}\}.$$

Using results from [Vu05] we can show that an upper bound on the spectral norm of $\mathbf{W}_0$, $\|\mathbf{W}_0\|$ is $\lambda\mathbf{\Lambda}^{-1}_{succ}$. Setting $\lambda < \mathbf{\Lambda}_{succ}$, we get $\|\mathbf{W}_0\| < 1$. Since $\mathbf{W}$ is obtained by the projection of $\mathbf{W}_0$ onto $\mathcal{M}_U$, we have $\|\mathbf{W}\| < 1$. We then show that when $\lambda < \mathbf{\Lambda}_{succ}$ and $\mathbf{D}_{\min} > \frac{1}{\lambda}$, the following statements hold, which proves Lemma 2.5.2.:

1. Construct $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$, such that $f(\mathbf{E}^L, \mathbf{W}) \geq 0$ for all feasible perturbations $\mathbf{E}^L$.

2. For all non-zero feasible $\mathbf{E}^L \in \mathcal{M}_{\mathbf{U}}^{\perp}$, show that $g(\mathbf{E}^L) > 0$.

For the details of the proof see Appendix A.1.2.

2.5.2 Proof Outline for Theorem 2

For Program 2.1.7, the optimal solution is as follows:

$$\mathbf{L}^0 = \mathbb{1}_{\mathcal{R}}^{n \times n}, \ \mathbf{S}^0 = \mathbf{S}_{obs}^0 = \mathbb{1}_{\mathcal{R} \cap \mathcal{A}_0}^{n \times n}.$$

We prove Theorem 2 by proving the following lemma:

Lemma 2.5.4. *If $\lambda < \tilde{\Lambda}_{succ}$ and $\tilde{\mathbf{D}}_{\min} > \frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to Program 2.1.7 with high probability.*

The proof of the above lemma follows similar steps as that of Lemma 2.5.2. The main difference is in the construction of the subgradients. For details see Appendix A.2.

2.6 Summary

In this chapter we considered the problem of graph clustering with missing data or partial observations. We analyzed the performance of two convex-optimization-based-algorithms and for a generative model for partially observed graphs (based on the Stochastic Block Model), we provided explicit bounds (not orderwise) on the problem parameters that guarantees the exact recovery of the underlying cluster structure. We also corroborated our theoretical results using synthetic datasets. Further, we also ran the convex algorithms on a real dataset for the problem of clustering using crowdsourcing (which will be explored in more detail later in this thesis in Chapters 4, 5, 6) and demonstrated that the convex algorithms can improve over the state-of-the-art clustering algorithms. The graphs considered in this chapter are unweighted. It is natural to wonder about convex algorithms for clustering weighted graphs or equivalently, similarity matrices, and the next chapter (Chapter 3) is devoted to it.

Chapter 3

SIMILARITY CLUSTERING IN THE PRESENCE OF OUTLIERS

In this chapter, we study the problem of clustering a set of data points based on their similarity matrix, each entry of which represents the similarity between the corresponding pair of points. We propose a convex-optimization-based algorithm for clustering using the similarity matrix, which has provable recovery guarantees. It needs no prior knowledge of the number of clusters and it behaves in a robust way in the presence of outliers and noise. Using a generative stochastic model for the similarity matrix (which can be thought of as a generalization of the Stochastic Block Model) we obtain *precise bounds* (not orderwise) on the sizes of the clusters, the number of outliers, the noise variance, separation between the mean similarities inside and outside the clusters and the values of the regularization parameter that guarantee the exact recovery of the clusters with high probability. The theoretical findings are corroborated with extensive evidence from simulations. We also evaluate the performance of our algorithm on real datasets and show improvement over k-means and spectral clustering. This chapter is based on our paper [VH16b].

3.1 Introduction

Big data sets are collected by companies, governments, and research institutions with the aim of extracting useful and relevant information. Clustering [JMF99] is a widely used pattern recognition tool that broadly refers to the problem of grouping together data points that are similar to each other. In certain instances, the data points can be embedded in Euclidean space; in others, they can be categorical data, which do not readily lend themselves to such an embedding, or a combination of both. For example, in the case of census data, each individual person has different attributes such as age and income which are numerical and race, religion, address etc., that are categorical [TSK05]. Simple encoding schemes, such as using a $D-$dimensional vector for a categorical field of size D, not only artificially inflate the dimension of the data, but might also give poor results when used with a numerical algorithm when compared to learning an embedding based on similarity or kernel methods [Zha+15; Cou05]. Depending on the data and application domain, it is often possible to construct a similarity map between pairs of data points that assigns a numerical value to how similar (or dissimilar) two data points are.

This in turn leads to a *similarity matrix* (also referred to as an affinity matrix in the literature).

If we have noiseless data and an ideal similarity map, then all pairs of points in the same cluster would be mapped to the same similarity value, say 1, and 0 otherwise. However, in reality, the data will be noisy and it is difficult to design a perfectly ideal similarity map. We assume a simple but reasonable probabilistic generative model for the similarity matrix where the average similarity between two data points is higher if they are in the same cluster and lower otherwise. This model can be seen as a natural extension of the popular Stochastic Block Model [HLL83], which is a random unweighted graph model where the probability of the existence of an edge between nodes in the graph that are in the same cluster is higher than those that are not.

The data, apart from being *noisy*, might also contain *outliers*, that is, data points that do not belong to any clusters. Thus, given a noisy similarity matrix, denoted by $\mathbf{A}$, with outliers, and no other side information, we want to reliably find the clusters. In this regard, we seek to identify a matrix $\mathbf{X}$ whose rank is equal to the number of clusters and in the regions corresponding to the same cluster has entries that are non-zero and equal, and zero elsewhere – thus reflecting the cluster regions (defined formally in Section 3.3). In most practical scenarios, the number of clusters is much lower than the total number of data points, which makes the matrix $\mathbf{X}$ low-rank.

Convex programs for clustering have drawn attention recently as they are robust to noise and lend themselves to analysis. A general convex approach of using low-rank plus sparse matrix decomposition via trace-norm minimization with a regularized l_1-norm penalty (robust PCA) for finding clusters in *unweighted graphs* has been well-studied [XCS10; Che+14; AV14; AV11; OH11; CSX12; Ame13; VOH14b; VOH14a]. While the sparse noise model and hence the l_1 penalty works very well for unweighted graphs, it does not fit the similarity model.

Inspired by the robust PCA-based clustering algorithms for unweighted graphs, we propose the following convex program to find the low-rank matrix $\mathbf{X}$ for similarity

clustering:

$$\begin{aligned} \underset{\mathbf{X}}{\text{minimize}} \ & \frac{1}{2}\|\mathbf{A}-\mathbf{X}\|_F^2 + \lambda \,\text{trace}(\mathbf{X}) \\ \text{subject to} \ & \\ & \mathbf{X} \succcurlyeq 0,\ \mathbf{X}_{i,j} \geq 0 \ \text{ for all } i,j \in [n] \\ & \sum_j \mathbf{X}_{i,j} \leq 1, \ \text{ for all } i \in [n], \end{aligned} \tag{3.1.1}$$

where $\|.\|_F$ is the Frobenius norm (square root of the sum of the squares of the entries of the matrix), $[n]$ denotes the set $\{1, 2, \ldots, n\}$ and $\lambda > 0$ is a regularization parameter (we will later comment on how to set this). Also, by $\mathbf{X} \succcurlyeq 0$, we mean that $\mathbf{X}$ is *symmetric* and has non-negative eigenvalues. The constraints $\sum_j \mathbf{X}_{i,j} \leq 1$ along with $\mathbf{X} \geq 0$ helps in forcing the entries outside the cluster to zero and those corresponding to the same cluster to be equal.

Program 3.1.1 is very simple and intuitive. Furthermore, it does not require any information other than the similarity matrix itself. The goal of this work is to understand the *fundamental limits* of the simple Program 3.1.1, that is, the conditions under which it successfully recovers clusters. In particular, we aim to address the following questions (Section 3.3.4):

1. How noisy can the similarity matrix be? At a given noise level, how separated should the average similarity inside and outside the clusters be?
2. How small can the clusters be? How much can their relative sizes vary?
3. How many outliers can be tolerated? How is the performance affected as the number of outliers becomes large, say larger than the size of the clusters?

Related Works

In this section we will discuss some related works.

Convex Penalties: [LOL11; Hoc+11; Che+15; CAB14] have introduced regularized convex relaxations for *hierarchical clustering* and show that as the regularizer is varied in a certain range there is a coalescing of clusters that gives rise to a hierarchical tree. However, they do not give guarantees on the problem parameters that give rise to a particular clustering at any point in the tree. Also, they do not provide theoretical guarantees on clustering in the presence of outliers.

Spectral Clustering: [McS01] analyzes the spectral partitioning of graphs under the Stochastic Block Model and [R+11] studies the asymptotic correctness of spectral clustering for these models. [NJW02; Bal+11] study the stability of the eigenvectors of the graph Laplacian under noisy perturbations. [Bal+11] provides guarantees on the exact recovery of clusters for spectral clustering under noisy perturbations to the similarity matrix. However, they require the clusters be balanced (i.e., the size of the clusters are constant fractions of each other). Moreover, these results do not hold when there are outliers.

Convex Programs for Graph Clustering: [XCS10; Che+14; AV14; AV11; OH11; CSX12; Ame13; VOH14b; VOH14a] consider clustering unweighted graphs via convex optimization based on a low-rank + sparse decomposition of the *unweighted* adjacency matrix of the graph via nuclear norm minimization with l_1 regularization. In the case of unweighted graphs, if the edge density inside the clusters is bigger than that outside, under mild conditions, convex programs can recover clusters of size $\Omega(\sqrt{n})$, regardless of the size of the outliers. Whereas in the case of similarity clustering (Section 3.3.3), if the number of outliers is larger than the smallest cluster, Program 3.1.1 gives an extra cluster that contains all the outliers.

Submatrix Localization: [HWX15] considers the special case of submatrix localization (bi-clustering) when the number of clusters, $K = 1$ and provides guarantees for exact recovery via message passing algorithm when the size of the cluster is known. Section 3 in [CX16] considers the problem of submatrix localization when the clusters are homogeneous (same mean inside all the clusters) and have same size. They provide orderwise bounds on signal strength required for exact recovery of clusters from a convex program which requires the knowledge of cluster sizes and number of clusters. We can recover the results for the case of symmetric submatrix localization by setting $\mu_i = \mu, n_i = m, \ \forall i \in [K]$. Note that the quantities *signal* and *SNR* defined in [HWX15] and [CX16] are related to the separation between means (discussed in 5 in Section 3.3.4) that arises via cross-cluster density.

Convex Program for Similarity Clustering: The work that is closest to ours in terms of the approach and analysis is [Ame14], which considers the problem of clustering a similarity matrix when the number of clusters are known. [Ame14] analyzes a convex program and provides guarantees for recovering the clusters as long as the number of number of outliers is not too large (less than the size of the smallest cluster). While the results in [Ame14] are interesting, it does not comment on the quality of the solution when the number of outliers is large. Further,

the convex program in [Ame14] requires the knowledge of the number of clusters, which can be problematic in the presence of outliers. In contrast, Program 3.1.1 is oblivious to the exact number of clusters and naturally figures it out as a function of the regularization parameter. This is helpful in understanding the behavior of the program when there are large number of outliers. Our analysis shows that Program 3.1.1 can recover the clusters as long as the regularization parameter is within a *range* rather than a specific number, which is robust to error when it is heuristically set. The model and analysis in [Ame14] does not capture the effect of the noise variance in the similarity matrix on the performance of the program. Though our analysis technique is inspired by the work [Ame14], we extend the analysis to understand the behavior of the solution in the presence of large number of outliers as well as to capture the effect of the noise variance. Also, our analysis gives *precise thresholds* for the successful recovery conditions, whereas the results in [Ame14] are orderwise.

Our Contributions: Our contributions are multifold:

1. We analyze Program 3.1.1 on a generative model (defined further below in Section 3.2), that is a natural extension of the Stochastic Block Model, and obtain *precise thresholds* (not orderwise) as a function of the problem parameters sufficient for the exact recovery of the underlying cluster structure (Section 3.3). Though our analysis uses the problem parameters, the program itself is agnostic to them.

2. We provide insights into the behavior of the solution in the presence of outliers (Sections 3.3.2 and 3.3.3), which is important from a practical standpoint. In the presence of a large number of outliers, Program 3.1.1 exhibits an interesting difference from what occurs in robust PCA-based convex algorithms for unweighted graphs.

3. Our analysis also gives insights into the effect of the noise variance in the similarity matrix on the successful recovery of clusters.

3.2 Generative Model for Similarity Matrices

In this section we define the random generative model for similarity matrices used for analysis in this chapter. Let n be the number of data points composed of K disjoint clusters and a set of outliers (points that do not belong to any clusters).

Definition 3.2.1 (Similarity Block Model)**.** *Let* $\mathbf{A} = \mathbf{A}^T$ *be the similarity matrix with entries* $\mathbf{A}_{l,m} \in [0,1]$*. The entries* $\mathbf{A}_{l,m}$ *with* $l \geq m$ *are random, independent of each other given the cluster assignment, with variance* σ^2 *and the means given by:*

$$\mathrm{E}(\mathbf{A}_{l,m}) = \begin{cases} \mu_i, & \text{if } l, m \text{ are in the same cluster } i. \\ \mu_{out}, & \text{if } l, m \text{ are not in the same cluster.} \end{cases}$$

3.3 Exact Recovery Guarantees In the Presence of Outliers

Let n_i, where $i \in [K]$, denote the number of nodes in cluster i, which we will refer to as the *size* of cluster i. If there are outliers, that is nodes that do not belong to any cluster, we denote the number of outliers by n_{K+1} (or n_{out}). Assume that the similarity matrix is generated from the model in Definition 3.2.1. In this section we present the conditions to guarantee the exact recovery of the underlying cluster structure in the cases when there are (a) no outliers, (b) a small number of outliers, and (c) a large number of outliers. These results together provide the complete picture of the performance of Program 3.1.1 in the presence of outliers.. The results presented hold with probability at least $1 - n^2 \exp\{-\Omega(n_{min})\}$, where $n_{min} = \min_{i \leq K} n_i$ is the size of the smallest cluster.

3.3.1 No Outliers

In the case where there are no outliers, we aim to recover the following matrix via Program 3.1.1:

$$\mathbf{X}^* = \sum_{i=1}^{K} \mathbf{x}_i \mathbf{x}_i^T, \quad \mathbf{x}_i = \frac{1}{\sqrt{n_i}} \mathbf{c}_i, \tag{3.3.1}$$

where $\mathbf{c}_i \in \mathrm{R}^n$ is the indicator vector for cluster i, with ones in the entries corresponding to the data points that belong to cluster i and zeros everywhere else. $\mathbf{x}_i$ is the normalized indicator vector for cluster i. So, the entries of $\mathbf{X}^*$ are

$$\mathbf{X}^*_{l,m} = \begin{cases} \frac{1}{n_i}\,, & \text{if both nodes } l, m \text{ are in the same cluster } i \\ 0\,, & \text{if nodes } l, m \text{ are not in the same cluster.} \end{cases}$$

The following quantities are important for our results:

- *Cluster Density:* For each cluster $i \in [K]$, define $\rho_i := n_i \mu_i > 0$, requiring $\mu_i > 0$.
- *Minimum Cluster Density* is defined as $\rho_{min} := \min_{i \leq K} \rho_i > 0$.

- *Cross Cluster Density:* For each pair of clusters $i \neq j \in [K]$, define the *cross cluster density* as $\gamma_{ij} := 2\ \left(\frac{\mu_i+\mu_j}{2} - \mu_{out}\right)\left(\frac{1}{n_i}+\frac{1}{n_j}\right)^{-1} > 0$, requiring $\frac{\mu_i+\mu_j}{2} > \mu_{out}$. That is, the average of the mean similarity of any two clusters clusters i and j must be at least as big as the mean similarity between them.

- *Minimum Cross Cluster Density:* is defined as $\gamma_{min} := \min_{i\neq j\leq K} \gamma_{ij} > 0$.

- *Noise threshold*, $\Lambda := 2\,\sigma\sqrt{n}$ which depends only on the noise variance and number of data points.

Theorem 3. *[No Outliers] When there are no outliers, if the regularizer λ is within the following range,*

$$\Lambda < \lambda < \min\left\{\rho_{min}, \gamma_{min}\right\} - 1, \tag{3.3.2}$$

then $\mathbf{X}^*$ is the unique optimal solution to Program 3.1.1 with high probability. If $\lambda > \min\left\{\rho_{min}, \gamma_{min}\right\} - 1$, then Program 3.1.1 fails to recover $\mathbf{X}^*$ with high probability.

3.3.2 Small Number of Outliers

In the presence of outliers, the solution to Program 3.1.1 depends on the number of outliers compared to the size of the smallest cluster. When $n_{out} \leq \mathcal{O}(n_{min})$, we refer to it as a small number of outliers. In addition to the cross cluster density γ defined before, define the following:

- *Effective cluster density* in presence of outliers for each cluster i as $\eta_i := (\mu_i - 2\mu_{out})\, n_i > 0$, implying $\mu_i > 2\mu_{out}$, required only in the case of small number of outliers.

- *Minimum effective density* be $\eta_{min} := \min_{i\leq K}\eta_i$.

Theorem 4. *[Small Number of Outliers] If the regularizer λ is within the following range,*

$$\Lambda + \mu_{out} n_{K+1} < \lambda < \min\left\{\eta_{min}, \gamma_{min}\right\} - 1, \tag{3.3.3}$$

then $\mathbf{X}^*$ is the unique optimal solution to Program 3.1.1 with high probability. If $\lambda > \min\left\{\eta_{min}, \gamma_{min}\right\} - 1$ then, Program 3.1.1 fails to recover $\mathbf{X}^*$ with high probability.

3.3.3 Large Number of Outliers

When the number of outliers is large (at least $\Omega(\sqrt{n})$) and is comparable or larger than the size of clusters, we cannot hope to recover $\mathbf{X}^*$, which requires the entries corresponding to the outlier region to be all zeros. Instead Program 3.1.1 groups all the outliers together to give an extra cluster and hence recovers $\tilde{\mathbf{X}} := \sum_{i=1}^{K+1} \mathbf{x}_i \mathbf{x}_i^T$, where $\mathbf{x}_{K+1}$ is the normalized indicator vector for the cluster of outliers. So, the entries of $\tilde{\mathbf{X}}$ are

$$\tilde{\mathbf{X}}_{l,m} = \begin{cases} \frac{1}{n_i}\,, & \text{if nodes } l, m \text{ are in the same cluster } i. \\ 0\,, & \text{if nodes } l, m \text{ are in different clusters.} \\ \frac{1}{n_{K+1}} & \text{if both nodes } l, m \text{ are outliers.} \end{cases}$$

Note that this is not a bad scenario. Rather, it is good that outliers get separated out as a cluster and do not get merged with other clusters. Once the cluster structure is revealed, one can compare the average similarity inside the clusters obtained and the average similarity outside to decide if any of the clusters obtained have average similarity very close to that of outside cluster region, and hence discard it.

In addition to the cluster densities ρ and the cross cluster denisities γ, define the following:

- *Outlier Density:* $\rho_{K+1} := \mu_{out} n_{K+1}$.
- *Cross Cluster-Outlier Density:* For each $i \in [K]$, define cross density of cluster i with outliers as, $\gamma_{i,K+1} := (\mu_i - \mu_{out}) \left(\frac{1}{n_i} + \frac{1}{n_{K+1}}\right)^{-1} > 0$.
- *Minimum cluster density* in the presence of *outliers* as $\rho_{min}^{out} := \min_{i \le K+1} \rho_i$.
- *Minimum cross cluster density* in the presence of *outliers* $\gamma_{min}^{out} := \min_{i \ne j \le K+1} \gamma_{ij}$.

Theorem 5. *[Large Number of Outliers] If the regularizer λ is within the following range,*

$$\Lambda < \lambda < \min\left\{\rho_{min}^{out}, \gamma_{min}^{out}\right\} - 1, \tag{3.3.4}$$

then $\tilde{\mathbf{X}}$ is the unique optimal solution to Program 3.1.1 with high probability. If $\lambda > \min\left\{\rho_{min}^{out}, \gamma_{\min}^{out}\right\} - 1$, then, the Program 3.1.1 fails to recover $\tilde{\mathbf{X}}$ with high probability.

Note that Theorems 4 and 5 are not in contradiction, since if the conditions on mean and cluster sizes are satisfied in both cases, setting $\lambda > 2\sigma\sqrt{n} + \mu_{out} n_{K+1}$ (Equation 3.3.3 in Theorem 4) would violate $\mu_{out} n_{K+1} > \lambda + 1$ (Equation 3.3.4 in Theorem 5).

3.3.4 Discussion:

In this section we provide discussion of the theorems presented in this paper.

1. **Size of the Smallest Cluster:** All three theorems stated in this section imply $\rho_i = \mu_i n_i > \Lambda + 1 = 2\sigma\sqrt{n} + 1 \; \forall i$, and hence we require $n_{min} \geq \Omega(\sqrt{n})$ to guarantee success, which matches the earlier known results. The results cannot guarantee success when $\Lambda+1 < \rho_{min}$, that is, when $n_{min} \leq o(\sqrt{n})$. In this regime it is not known whether the clustering problem can be efficiently solved.

2. **Relative Size of Clusters:** Note that the results do not place any restrictions on the relative size of clusters. So, we can have clusters of varying sizes: for example, some clusters of size $\Theta(n)$ and some of size $\Theta(\sqrt{n})$.

3. **Size of Outliers**: In the presence of outliers, Theorem 4 implies $\rho_i > \eta_i > \Lambda + \mu_{out} n_{K+1} + 1$. So to guarantee the exact recovery of $\mathbf{X}^*$ as the optimal solution to Program 3.1.1 we require $n_{min} \geq \max\{\Omega(\sqrt{n}), \Omega(n_{K+1})\}$. Note that this requirement is automatically satisfied if the number of outliers is $o(\sqrt{n})$ as we need $n_{min} \geq \Omega(\sqrt{n})$ in all cases. If the number of outliers is large in comparison to the size of the smallest cluster, we cannot guarantee the recovery of a solution with all zero entries in the region corresponding to the outliers.

4. **Large Number of Outliers:** Theorem 5 implies $\rho_{out} = \mu_{out} n_{out} \geq \Lambda + 1 = 2\sigma\sqrt{n} + 1$. So, if the number of outliers is at least $\Omega(\sqrt{n})$, i.e, the number of outliers is *large*, then we can guarantee that they form their own cluster under the conditions in Theorem 5.

5. **Separation Between the Means Compared to the Noise Variance:** For simplicity, assume all the clusters are of equal size, $n_i = m$ and $\mu_i = \mu_{in} \; \forall i$.

   a) In the case of no outliers, from $\gamma_{ij} > 2\sigma\sqrt{n} + 1$ (Equation 3.3.2) we get the following sufficient condition:

   $$\frac{\mu_{in} - \mu_{out}}{\sigma} > \frac{1}{m}\left(2\sqrt{n} + \frac{1}{\sigma}\right).$$

   If $m = \Theta(\sqrt{n})$, then as $n \to \infty$, we require $\frac{\mu_{in}-\mu_{out}}{\sigma} > \Omega(1)$, whereas if $m = \Theta(n)$ then $\mu_{in} - \mu_{out} > 0$ is sufficient to guarantee exact recovery.

b) In the case of large number of outliers, from Equation 3.3.4 we get:

$$\frac{\mu_{in} - \mu_{out}}{\sigma} > \left(\frac{1}{m} + \frac{1}{n_{K+1}}\right)\left(2\sqrt{n} + \frac{1}{\sigma}\right).$$

c) In the case of small number of outliers, from Equation 3.3.3 we get:

$$\frac{\mu_{in} - 2\mu_{out}}{\sigma} > \frac{1}{m}\left(2\sqrt{n} + \frac{\mu_{out} n_{K+1} + 1}{\sigma}\right).$$

So the average similarity inside the clusters will have to be higher than twice the average similarity outside to recover $\mathbf{X}^*$ in the presence of small number of outliers.

6. **Regularization Parameter:** If the noise variance is known, then the regularizer can be set to $\lambda = 2\sigma\sqrt{n}$. In case there is no information about σ, then we suggest using the empirical variance of $\mathbf{A}$ or setting $\lambda = 2\sqrt{n}$. The value of λ provides a bound on how much noise can be tolerated, e.g, if we set $\lambda = 2\sqrt{n}$, then $\sigma \leq 1$ can be tolerated.

3.3.5 Brief Proof Outline

In this section we provide a brief outline of the proof. Detailed proof is available in the appendix (Appendix B). Define dual variables for the constraints of Program 3.1.1,

1. $\mathbf{Y} \in \mathbb{R}^{n\times n},\ \mathbf{Y} \succcurlyeq 0$ for constraint $\mathbf{X} \succcurlyeq 0$.
2. $\nu \in \mathbb{R}^n,\ \nu \geq 0$ for constraints $\sum_j \mathbf{X}_{i,j} \leq 1,\ \forall i$.
3. $\mathbf{Z} \in \mathbb{R}^{n\times n},\ \mathbf{Z} \geq 0$ for constraints $\mathbf{X} \geq 0$.

If a feasible $\hat{\mathbf{X}}$ is an optimal solution to Program (3.1.1), then the following conditions have to hold (from KKT conditions and complementary slackness):

$$\mathbf{Z} + \mathbf{Y} = \lambda\mathbf{I} + \hat{\mathbf{X}} + \mathbb{1}\nu^T - \mathbf{A} + \nu\mathbb{1}^T$$

$$\text{trace}(\hat{\mathbf{X}}\mathbf{Y}) = 0,\ \text{trace}(\hat{\mathbf{X}}\mathbf{Z}) = 0,\ \nu^T(\hat{\mathbf{X}}\mathbb{1} - \mathbb{1}) = 0.$$

We first construct dual variables that satisfy the conditions above. The dual variables $\mathbf{Z}, \mathbf{Y}, \nu$ thus obtained are functions of the problem parameters $\{\{\mu_i\}_{i\in[K]}, \mu_{out}, \sigma, \{n_i\}_{i\in[K]}, n_{out}\}$. The condition $\mathbf{Y} \succcurlyeq 0$ will give the lower bound on λ of the form Λ or $\Lambda + \mu_{out}n_{out}$ depending on the case. The conditions $\nu \geq 0$ give the lower bounds on the cluster densities ρ. The conditions $\mathbf{Z} \geq 0$ give the lower bounds on cross-cluster densities γ and the effective cluster densities η.

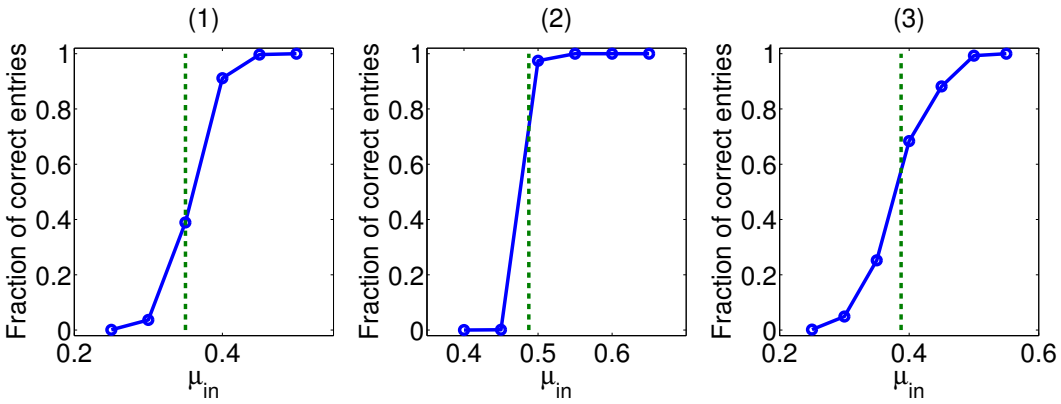


Figure 3.1: Fraction of correct entries in the solution obtained by running Program 3.1.1 with $n = 100$, similarity between the clusters: $\mu_{out} = 0.2$, standard deviation of noise: $\sigma = 0.1$ and varying similarity inside the clusters μ_{in}, for the three cases: (1) no outliers, (2) small number of outliers and (3) large number of outliers. Solid blue line is the curve from the simulations and the dashed green line is the threshold for μ_{in} predicted by theory.

3.4 Simulations

We consider an example of $n = 100$ data points to illustrate the sharp transitions predicted in the main results in Section 3.3 via numerical simulations. The similarity matrix is generated as follows: For $l \geq m$, $\mathbf{A}_{lm}$ is sampled independently from $\mathcal{N}(\mu_i, \sigma^2)$ if both the points l, m belong to the same cluster i, else it is sampled independently from $\mathcal{N}(\mu_{out}, \sigma^2)$. Note that this does not necessarily satisfy $\mathbf{A}_{l,m} \in [0, 1]$. However, we will choose the μ_{in}, μ_{out} and range of σ such that we will not be violating the condition of the average similarity inside the clusters being positive. All the results presented are an average over 5 experiments unless otherwise stated.

3.4.1 Sharpness of Thresholds

We use the CVX package for Matlab [GB14; GB08] to run Program 3.1.1. We set the mean similarity between nodes that are not in the same cluster to $\mu_{out} = 0.2$. The standard deviation σ is set to 0.1. Note that for the clusters sizes of 20 and 80, non-zero entries of the ideal solution are 0.05 and 0.0125 respectively. We declare an entry of the solution matrix $\mathbf{X}_{l,m}$ to be in error if $|\mathbf{X}_{l,m} - \mathbf{X}^{ideal}_{l,m}| > 10^{-3}$. For Theorems 3 and 4, $\mathbf{X}^{ideal}$ is $\mathbf{X}^*$ and for Theorem 5, $\mathbf{X}^{ideal}$ is $\tilde{\mathbf{X}}$.

1. **No Outliers:** We consider five clusters of size 20 each, and no outliers. We set the regularization parameter $\lambda = 1.001\Lambda = 1.001 \times 2\sigma\sqrt{n}$ (lower bound on λ in Equation 3.3.2). We vary the mean similarity inside the clusters μ_{in}

from 0.25 to 0.5 in steps of 0.05. From Theorem 3, we expect Program 3.1.1 to succeed (to obtain solution $\mathbf{X}^*$) with high probability when, $\mu_{in} > \mu_{out} + (\lambda+1)/n_1 = 0.35$ (when $\sigma = 0.1$). Figure 3.1(1) shows the fraction of correct entries of the output of Program 3.1.1. The solid blue line is the curve obtained from the Simulations and the dashed-red line is the threshold for μ_{in} as predicted by our analysis. We observe that the transition occurs around μ_{in} predicted from our results.

2. **Small Number of Outliers:** For this case, we consider one cluster of size $n_1 = 80$ and the rest $n_{out} = 20$ are outliers. We set the regularization parameter to $\lambda = 1.001(\Lambda + \mu_{out} n_{out})$ (lower bound on λ in Equation 3.3.4) and vary μ_{in} from 0.4 to 0.65 in steps of 0.05. From Theorem 4, we expect Program 3.1.1 to succeed (obtain solution $\mathbf{X}^*$) with high probability when, $\mu_{in} > 2\mu_{out} + (\lambda+1)/n_1 = 0.49$ (when $\sigma = 0.1$). Figure 3.1(2) shows the fraction of correct entries of the output of Program 3.1.1. The solid blue line is the curve obtained from the Simulations and the dashed-red line is the threshold for μ_{in} as predicted by our analysis. We observe that the transition occurs around μ_{in} predicted from our results.

3. **Large Number of Outliers:** In this case, we consider one cluster of size $n_1 = 20$ and the rest $n_{out} = 80$ are outliers. We set the regularization parameter to $\lambda = 1.001\Lambda$ (lower bound on λ in Equation 3.3.4) and vary μ_{in} from 0.25 to 0.55 in steps of 0.05. From Theorem 5, we expect Program 3.1.1 to succeed (obtain solution $\tilde{\mathbf{X}}$) with high probability when, $\mu_{in} > \mu_{out} + (\lambda+1)\,(1/n_1 + 1/n_{out}) = 0.39$ (when $\sigma = 0.1$). Figure 3.1(3) shows the fraction of correct entries of the output of Program 3.1.1. The solid blue line is the curve obtained from the Simulations and the dashed-red line is the threshold for μ_{in} as predicted by our analysis. We observe that the transition occurs around μ_{in} predicted from our results.

3.4.2 Impact of the Noise Variance

We generate similarity matrix on dataset of $n = 600$ points with $\mu_{out} = 0.20$. Standard deviation σ is to start with 0.01 and then varied from 0.05 to 0.20 in steps of 0.05. Due to the limitations on the datasize that can be handled by the CVX package on Matlab, we solve Program 3.1.1 using Alternating Direction Method of Multipliers (ADMM) based implementation. We declare an entry of the solution matrix $\mathbf{X}$ to be in error if its absolute difference from the value in ideal solution is

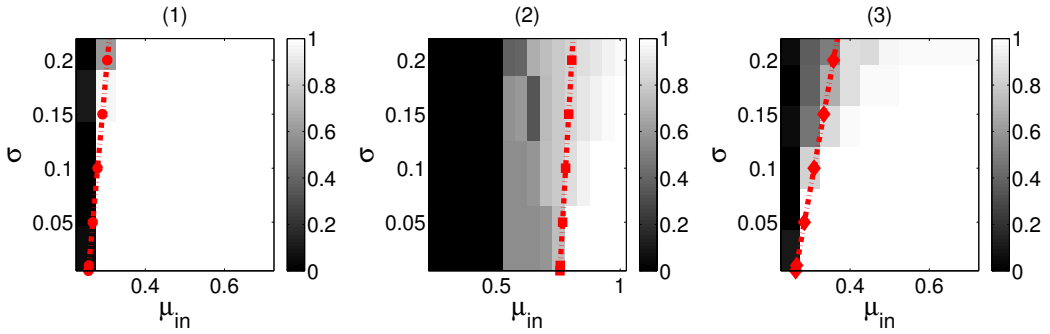


Figure 3.2: Region of success (white) and failure (black) for Program 3.1.1 on synthetic data of $n = 600$, $\mu_{out} = 0.20$, for varying μ_{in} and σ with (1) 3 clusters size 200 each (no outliers), $\lambda = 1.001\Lambda$, (2) 2 clusters of size 200 and rest $n_{out} = 200$ outliers, $\lambda = 1.001(\Lambda + \mu_{out} n_{out})$ (recover only the clusters), (3) 2 clusters of size 200 and rest $n_{out} = 200$ outliers, $\lambda = 1.001\Lambda$ (recover outliers as a cluster). The dotted red lines are the thresholds for μ_{in} predicted by the theory for the corresponding σ.

more that 0.001, that is $|\mathbf{X}_{l,m} - \mathbf{X}^{ideal}_{l,m}| > 10^{-3}$.

No Outliers: In this case we consider three clusters of equal size $n_1 = n_2 = n_3 = 200$ with the same mean similarity inside the clusters μ_{in} varied from 0.25 to 1 in steps of 0.05 and the standard deviation of noise σ starting at 0.01 and then varied from 0.05 to 0.2 in steps of 0.05. We set $\lambda = 1.001\Lambda$ as required by Theorem 3.

With Outliers: Here we consider two clusters of equal size $n_1 = n_2 = 200$ with the same mean similarity inside the clusters. Rest of the $n_{out} = 200$ nodes are outliers. We have the following two cases:

Recover Only the Clusters: We set $\lambda = 1.001\Lambda + \mu_{out} n_{out}$ as required by Theorem 4 and vary μ_{in} from 0.25 to 1 in steps of 0.05.

Outliers Recovered as a Cluster: We set $\lambda = 1.001\Lambda$ as required by Theorem 5 and vary μ_{in} from 0.25 to 1 in steps of 0.05.

Figure 3.2 shows the regions of success (white) and failure (black) in each case. The dotted red lines are the thresholds for μ_{in} predicted by the theory for the corresponding σ. The transition occurs around the predicted threshold. The simulations suggest that our thresholds are sharp.

3.5 Experiments on Real Datasets

We run the inexact implementation of Program 3.1.1 via ADMM on 3 real datasets from UCI Machine Learning Repository. We obtain a rounded output (with entries

Table 3.1: Percentage of mis-classified data points

	Iris	`digit1000`
Program 3.1.1 + K-means	11.33% (K = 3)	16.6% (K = 9)
Spectral Clustering	33.33%(K = 2)	34.0% (K = 7)

round to 0 or 1) by thresholding the entries of the output of Program 3.1.1 by comparing it to the mean of all the entries. Then we run k-means algorithm (we run it 5 times and pick the best solution) on the rounded output to assign clusters. We compare it with Spectral Clustering with normalized Laplacian.

Fisher's Iris Data Set: It has 150 data points with 50 data points in each of the three classes (Iris Setosa, Iris Versicolour, Iris Virginica), and has four attributes (sepal length, sepal width, petal length, and petal width). We use Gaussian Kernel to obtain a similarity matrix, $\mathbf{A}_{l,m} = \exp(-||\mathbf{y}_l - \mathbf{y}_m||^2/2\sigma^2)$ with $\sigma^2 = 10$ (where $\mathbf{y} \in \mathbb{R}^4$ are the data points). We run Program 3.1.1 with $\lambda = 0.5\sqrt{n}$.

NIST Handwritten Digits: We use the similarity matrices of `digit1000` and `digitFive1000` datasets from [VM03][1]. These datasets are generated using Gaussian Kernel with $\sigma = 10$ on a sub-sampled data (100 elements per digit where each digit is represented by 8×8 matrix of integer in the range 0 to 16 to obtain a 1000 data points of 64-dimensions) from the dataset available on UCI Machine Learning Repository. `digit1000` has all the 1000 sample points for 10 digits whereas `digitFive1000` is a subset of with only digits 0, 2, 4, 6, and 7. In both the cases, since the entries of the similarity matrix are very small compared to 1 (diagonal entries), we set the diagonal entries to 0 and run Program 3.1.1 with $\lambda = 2\sqrt{n}\text{var}(\mathbf{A})$.

The rounded output for all the real datasets obtained is shown in Figure 3.3 and the percentage of mis-classified data points is presented in Table 3.1.

3.6 Summary

In this chapter we focused on understanding the performance of convex-optimization-based clustering in the presence of outliers when we only have the similarity matrix given to us (with no additional information). We analyzed a simple and intuitive convex program (3.1.1), and for the stochastic similarity model, we provided guarantees on its performance by deriving precise thresholds (not orderwise) on the cluster sizes, the strength of similarity compared to noise, the number of outliers, and the regularization parameter. We corroborate our results through simulations.

[1] http://www.stat.washington.edu/spectral/datasets.html

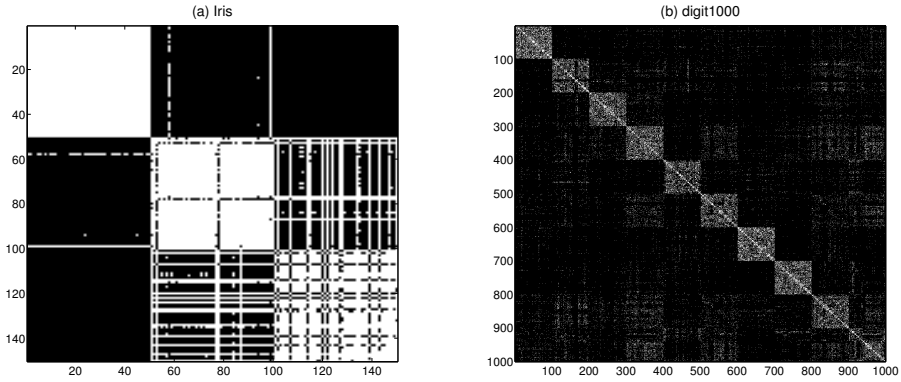


Figure 3.3: Rounded output after running Program 3.1.1 on real datasets. (1) Iris and (2) `digit1000`.

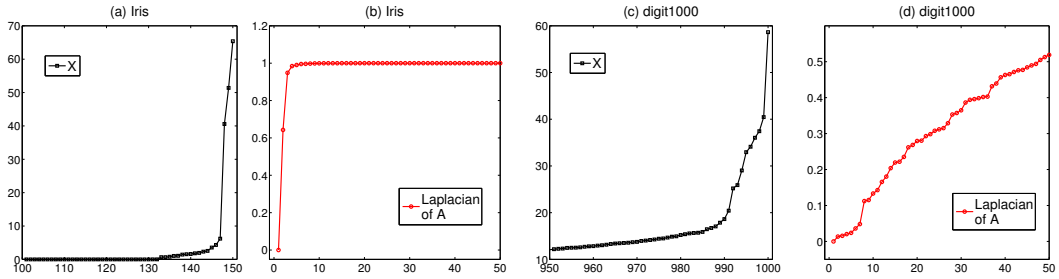


Figure 3.4: Sorted eigenvalues for the rounded output $\mathbf{X}$ and the normalized Laplacian of the similarity matrix $\mathbf{A}$ for Iris and `digit1000` datasets.

One of the drawbacks of convex approach is that it is computationally intensive. A future research direction would be to scale the convex approaches for clustering to work with large datasets.

Chapter 4

CROWDSOURCED CLUSTERING: TRIANGLE VS EDGE QUERY

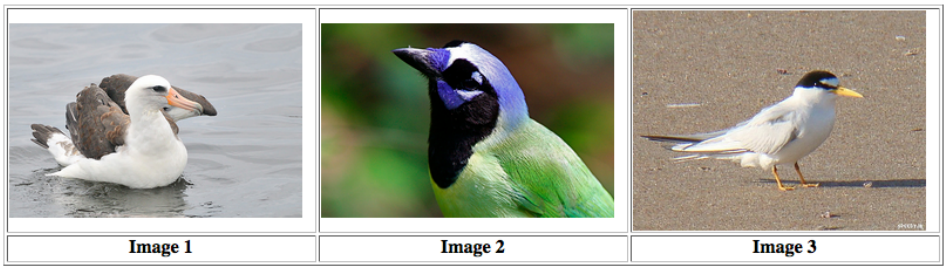
In this chapter, we consider the task of clustering items using answers from non-expert crowd workers. In such cases, the workers are often not able to label the items directly; however, it is reasonable to assume that they can compare items and judge whether they are similar or not. An important question is what queries to make, and we compare two types: random edge queries, where a pair of items is revealed, and random triangles, where a triple is. Since it is far too expensive to query all possible edges and/or triangles, we need to work with partial observations subject to a fixed query budget constraint. When a generative model for the data is available (and we consider a few of these) we determine the cost of a query by its entropy; when such models do not exist we use the average response time per query of the workers as a surrogate for the cost. In addition to theoretical justification, through several simulations and experiments on two real data sets on Amazon Mechanical Turk, we empirically demonstrate that, for a fixed budget, triangle queries uniformly outperform edge queries. Even though, in contrast to edge queries, triangle queries reveal dependent edges, they provide more reliable edges and, for a fixed budget, many more of them. We also provide a sufficient condition on the number of observations, edge densities inside and outside the clusters, and the minimum cluster size required for the exact recovery of the true adjacency matrix via triangle queries using a convex optimization-based clustering algorithm. This chapter is based on our paper [VH16a].

4.1 Introduction

Collecting data from non-expert workers on *crowdsourcing* platforms such as Amazon Mechanical Turk, Zooinverse, Planet Hunters, etc. for various applications has recently become quite popular. Applications range from creating a labeled dataset for training and testing supervised machine learning algorithms [Ray+10; Sno+08; Von+08; SF08; Wel+10; Yi+12] to making scientific discoveries [SPD14; Lin+13]. Since the workers on the crowdsourcing platforms are often non-experts, the answers obtained will invariably be noisy. Therefore the problem of designing queries and inferring quality data from such non-expert crowd workers is of great impor-

Same Species
NOT Same Species

Figure 4.1: Example of an edge query: “Do these two birds belong to the same species?”



ALL are Same Species
ONLY 1 and 2 are Same Species
ONLY 1 and 3 are Same Species
ONLY 2 and 3 are Same Species
NONE

Figure 4.2: Example of a triangle query: “Which of these birds belong to the same species?”

tance.

As an example, consider the task of collecting labels of images, e.g, of birds or dogs of different kinds and breeds. To label the image of a bird, or dog, a worker should either have some expertise regarding the bird species and dog breeds, or should be trained on how to label each of them. Since hiring experts or training non-experts is expensive, we shall focus on collecting labels of images through image comparison followed by clustering. Instead of asking a worker to label an image of a bird, we can show her two images of birds and ask: “Do these two birds belong to the same species?" (Figure 4.1). Answering this comparison question is much easier than the labeling task and does not require expertise or training. Though different workers

might use different criteria for comparison, e.g, color of feathers, shape, size etc., the hope is that, averaged over the crowd workers, we will be able to reasonably resolve the clusters (and label each).

Consider a graph of n images that needs to be clustered, where each pairwise comparison is an 'edge query'. Since the number of edges grows as $\mathcal{O}(n^2)$, it is too expensive to query all edges. Instead, we want to query a subset of the edges, based on our total query budget, and cluster the resulting partially observed graph. Of course, since the workers are non-experts, their answers will be noisy and this should be taken into consideration in designing the queries. For example, it is not clear what the best strategy to choose the subsets of edges to be queried is.

4.1.1 Our Contribution

In this work we compare two ways of partially observing the graph: random edge queries, where a pair of items is revealed for comparison, and random triangle queries, where a triplet is revealed. We give intuitive generative models for the data obtained for both types of queries. Based on these models we determine the *cost* of a query to be its entropy (the information obtained from the response to the query). On real data sets where such a generative model may not be known we use the average response time per query as a surrogate for the cost of the query. To fairly compare the use of edge vs. triangle queries we fix the total budget, defined as the (aforementioned) cost per query times the total number of queries. Empirical evidence, based on extensive simulations, as well as two real data sets (images of birds and dogs, respectively), strongly suggests that, for a fixed query budget, querying for triangles significantly outperforms querying for edges. Even though, in contrast to edge queries that give information on *independent edges*, triangle queries give information on *dependent edges*, i.e., edges that share vertices, we (theoretically and empirically) argue that triangle queries are superior because (1) they allow for far more edges to be revealed, given a fixed query budget, and (2) due to the self-correcting nature of triangle queries, they result in much more reliable edges.

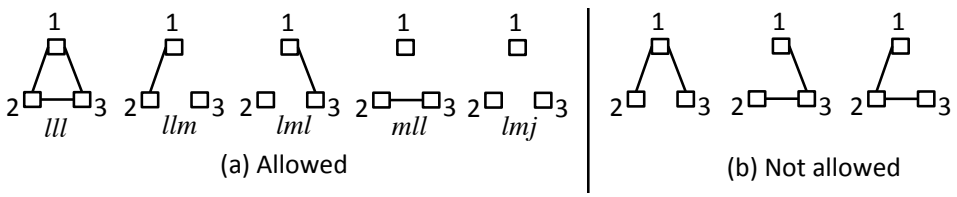


Figure 4.3: Configurations for a triangle query that are (a) observed and (b) not allowed.

Furthermore, for a specific convex optimization-based clustering algorithm, we also provide theoretical guarantee for the exact recovery of the true adjacency matrix via random triangle queries, which gives a sufficient condition on the number of queries, edge densities inside and outside the clusters and the minimum cluster size. In particular, we show that the lower bound of $\Omega(\sqrt{n})$ on the cluster size still holds even though the edges revealed via triangle queries are not independent.

4.1.2 Problem Setup

Consider n items with K disjoint classes/clusters plus outliers (items that do not belong to any clusters). Consider a graph with these n items as nodes. In the true underlying graph $\mathcal{G}^*$, all the items in the same cluster are connected to each other and the items that are not in the same cluster are not connected to each other. We do not have access to $\mathcal{G}^*$. Instead we have a crowdsourced query mechanism that can be used to observe a *noisy* and *partial* snapshot $\mathcal{G}^{\text{obs}}$ of this graph. Our goal is to find the cluster assignments from $\mathcal{G}^{\text{obs}}$. We consider the following two querying methods:

Random Edge Query: We sample E edges uniformly at random from $\binom{n}{2}$ possible edges. Figure 4.1 shows an example of an edge query. For each edge observation, there are two possible configurations: (1) Both items are similar, denoted by ll, and (2) The items are not similar, denoted by lm.

Random Triangle Query: We sample T triangles uniformly at random from $\binom{n}{3}$ possible triangles. Figure 5.1 shows an example of a triangle query. For each triangle observation, there are five possible configurations (Figure 5.2):(1) All items are similar, denoted by lll, (2) Items 1 and 2 are similar, denoted by llm, (3) Items 1 and 3 are similar, denoted by lml, (4) Items 2 and 3 are similar, denoted by mll, and (5) None are similar, denoted by lmj.

4.1.3 Related Works

[KOS11; KOS14; VVV14; Zho+12; LPI12; Zha+14] and references therein focus on the problem of inferring true labels from crowdsoruced multiclass labeling. The common setup in these problems is as follows: a set of items are shown to workers and labels are elicited from them. Since the workers give noisy answers, each item is labeled by multiple workers. Algorithms based on Expectation-Maximization [Zha+14] for maximum likelihood estimation and minimax entropy based optimization [Zho+12] have been studied for inferring the underlying true labels. In our setup we do not ask the workers to label the items. Instead we use

comparison between items to find the clusters of items that are similar to each other.

[Gom+11] considers the problem of inferring the complete clustering on n images from a large set of clustering on smaller subsets via crowdsourcing. Each HIT (Human Intelligent Task) is designed such that all of them share a subset of images to ensure overlapping. Each HIT has M images and all the $\binom{M}{2}$ comparisons are made. Each HIT is then assigned to multiple workers to get reliable answers. These clustering are then combined using an algorithm based on variational Bayesian inference. In our work we consider a different setup, where either pairs or triples of images are compared by the crowd to obtain a partial graph on the images which can be clustered.

In our previous work [VOH14a], we considered a convex approach to graph clustering with partially observed adjacency matrices, and provides an example of clustering images by crowdsourcing pairwise comparisons. However, it does not consider other types of querying such as triangle queries. In this work, we extend the analysis in [VOH14a] and show that a similar performance guarantee holds for clustering via triangle queries.

Another interesting line of work is learning embeddings and kernels through triplet comparison tasks in [Tam+11; WKB14; HVH14; VW12; Wah+14; HU13] and references therein. The 'triplet comparison' task in these works is of type: 'Is a closer to b or to c?', with two possible answers, to judge the relative distances between the items. On the other hand, a triangle query in our work has five possible answers (Figure 5.1) that give a clustering (discrete partitioning) of the three items.

4.2 Generative Models

Probability of observing a particular configuration y is given by:

$$\mathbf{Pr}(y) = \sum_{x \in \mathcal{X}} \mathbf{Pr}(y|x)\mathbf{Pr}(x),$$

where x is the true configuration and $\mathcal{X}$ is the set of true configurations. Let $\mathcal{Y}$ be the set of all observed configurations. Each query has a $|\mathcal{Y}| \times |\mathcal{X}|$ *confusion matrix* $[\mathbf{Pr}(y|x)]$ associated to it. Note that the columns of this confusion matrix sum to 1, i.e $\sum_{y \in \mathcal{Y}} \mathbf{Pr}(y|x) = 1$.

4.2.1 Random Edge Observation Models

For the random edge query case, there are two observation configurations, $\mathcal{Y} = \{ll, lm\}$ where lm denotes 'no edge' and ll denotes 'edge'.

$\mathbf{Pr}(y\|x)$	lll	llm	lmj
lll	$p^3+3p^2(1-p)$	pq^2	q^3
llm	$p(1-p)^2$	$p(1-q)^2+(1-p)q^2+2pq(1-q)$	$q(1-q)^2$
lml	$p(1-p)^2$	$(1-p)q(1-q)$	$q(1-q)^2$
mll	$p(1-p)^2$	$(1-p)q(1-q)$	$q(1-q)^2$
lmj	$(1-p)^3$	$(1-p)(1-q)^2$	$(1-q)^3+3q^2(1-q)$

Table 4.1: Query confusion matrix for the triangle block model for the homogeneous case.

One-coin Edge Model: Assume all the queries are equally hard. Let the ζ be the probability of answering a question wrong. Then $\mathbf{Pr}(ll|ll) = \mathbf{Pr}(lm|lm) = 1-\zeta$, $\mathbf{Pr}(lm|ll) = \mathbf{Pr}(ll|lm) = \zeta$. This model is inspired by the one-coin Dawid-Skene Model [DS79], which is used in inference for item label elicitation tasks. This is a very simple model and does not capture the difficulty of a query depending on which clusters the items in the query belong to. In order to incorporate these differences we consider the popular Stochastic Block model (SBM) [HLL83; CK01] which is one of the most widely used model for graph clustering.

Stochastic Block Model (SBM): Consider a graph on n nodes with K disjoint clusters and outliers. Any two nodes i and j are connected (independent of other edges) with probability p if they belong to the same cluster and with probability q otherwise. That is, $\mathbf{Pr}(ll|ll) = p$, $\mathbf{Pr}(lm|ll) = 1-p$, $\mathbf{Pr}(ll|lm) = q$ and $\mathbf{Pr}(lm|lm) = 1-q$. We assume that the density of the edges inside the clusters is higher than that between the clusters, that is, $p > q$.

4.2.2 Random Triangle Observation Models

For the triangle query model, there are five possible observation configurations (Figure 5.2), $\mathcal{Y} = \{lll, llm, lml, mll, lmj\}$.

One-coin Triangle Model: Let each question be answered correctly with probability $1-\zeta$, and when wrongly answered, all the other configurations are equally confusing. So, $\mathbf{Pr}(lll|lll) = 1-\zeta$ and $\mathbf{Pr}(llm|lll) = \mathbf{Pr}(lml|lll) = \mathbf{Pr}(mll|lll) = \mathbf{Pr}(lmj|lll) = \zeta/4$ and so on. This model, as in the case of the one-coin model for edge query, does not capture the differences in difficulty for different clusters. In order to include the differences in confusion between different clusters, we consider the following observation models for a triangle query. For these 3 items in the triangle query, the edges are first generated from the SBM. This can give rise to 8 configurations, out of which 5 are allowed as an answer to triangle query while the remaining 3 are not allowed (Figure 5.2). The two models differ in how they handle

$\mathbf{Pr}(y\|x)$	lll	llm	lmj
lll	p^3/z_{lll}	pq^2/z_{llm}	q^3/z_{lmj}
llm	$p(1-p)^2/z_{lll}$	$p(1-q)^2/z_{llm}$	$q(1-q)^2/z_{lmj}$
lml	$p(1-p)^2/z_{lll}$	$(1-p)q(1-q)/z_{llm}$	$q(1-q)^2/z_{lmj}$
mll	$p(1-p)^2/z_{lll}$	$(1-p)q(1-q)/z_{llm}$	$q(1-q)^2/z_{lmj}$
lmj	$(1-p)^3/z_{lll}$	$(1-p)(1-q)^2/z_{llm}$	$(1-q)^3/z_{lmj}$

Table 4.2: Query confusion matrix for the conditional block model for the homogeneous case.

the configurations that are not allowed, and are described below:

Triangle Block Model (TBM): In this model we assume that a triangle query helps in correctly resolving the configurations that are not allowed. So, when the triangle generated from the SBM takes one of the 3 non-allowed configurations, it is mapped to the true configuration. This gives a 5×5 query confusion matrix which is given in Table 4.1. Note that the columns for lml and mll can be filled in a similar manner to that of llm.

Conditional Block Model (CBM): In this model when a non-allowed configuration is encountered, it is redrawn again. This is equivalent to conditioning on the allowed configurations. Define the normalizing factors, $z_{lll} := 3p^3 - 3p^2 + 1$, $z_{llm} := 3pq^2 - 2pq - q^2 + 1$, $z_{llm} := 3q^3 - 3q^2 + 1$. The 5×5 query confusion matrix is given in Table 4.2.

Remark: Note that the SBM (and hence the derived models) can be made more general by considering different edge probabilities P_{ii} for cluster i and $P_{ij} = P_{ji}$ between clusters $i \neq j$.

Some intuitive properties of the triangle query models described in this section are:

1. If $p > q$, then the diagonal term will dominate any other term in a row. That is $\mathbf{Pr}(lll|lll) > \mathbf{Pr}(lll|\star \neq lll), \mathbf{Pr}(llm|llm) > \mathbf{Pr}(llm|\star \neq llm)$ and so on.

2. If $p > 1/2 > q$, then the diagonal term will dominate the other terms in the column, i.e, $\mathbf{Pr}(lll|lll) > \mathbf{Pr}(llm|lll) = \mathbf{Pr}(lml|lll) = \mathbf{Pr}(mll|lll) > \mathbf{Pr}(lmj|lll)$ etc.

3. When there is a symmetry between the items, the observation probability should be the same. That is, if the true configuration is llm, then observing lml and mll should be equally likely as item1 and item2 belong to the same

cluster and so on. This property will hold good in the general case as well except for when the true configuration is lmj. In this case, the probability of observing llm, lml, and mll can be different as it depends on the clusters to which items 1, 2 and 3 belong.

4.2.3 Adjacency Matrix: Edge Densities and Edge Errors

The adjacency matrix, $\mathbf{A} = \mathbf{A}^T$ of a graph can be partially filled by querying a subset of edges. Since we query edges randomly, most of the edges are seen only once. Some edges might get queried multiple times, in which case, we randomly pick one of them. Similarly we can also partially fill the adjacency matrix from triangle queries. We fill the unobserved entries of the adjacency matrix with zeros. We can perform clustering on $\mathbf{A}$ to obtain a partition of items. The true underlying graph $\mathcal{G}^*$ has perfect clusters (disjoint cliques). So, the performance of clustering on $\mathbf{A}$ depends on how noisy it is. This in turn depends on the probability of error for each revealed edge in $\mathbf{A}$, i.e, what is the probability that a true edge was registered as no-edge and vice versa. The hope is that triangle queries help workers to resolve the edges better and hence have less errors among the revealed edges than those obtained from edge queries.

If we make E edge queries, then the probability of observing an edge is $r = E/\binom{n}{2}$. If we make T triangle queries, the probability of observing an edge is $r_T = 3T/\binom{n}{2}$. Let rp $(r_T p_T)$ and rq $(r_T q_T)$ be the edge probability inside the clusters and between the clusters, respectively, in $\mathbf{A}$, which is *partially* filled via edge (triangle) queries.

In the case of one-coin model, for edge qurey, $p = 1 - \zeta$ and $q = \zeta$. For triangle query, $p_T = 1 - 3\zeta/4 > 1 - \zeta = p$ and $q_T = \zeta/2 < \zeta = q$. That is, the quality of the edges obtained via triangle queries is better than those obtained via edge queries.

For the TBM, when $p > 1/2 > q$, we can show that $p_T > p$ and $q_T < q$, and hence quality of edges obtained via triangle queries is better. For the CBM, when $p > 1/2 > q$, under reasonable assumptions on r, $r_T q_T < rq$, but depending on the values of r and r_T, $r_T p_T$ can get below rp. If the decrease in edge probability between the clusters is large enough to overcome the fall in edge density inside the clusters, then $p_{err}^{\Delta} < p_{err}^{edge}$.

In summary, when $\mathbf{A}$ is filled by triangle queries, the edge density between the clusters decreases and the overall number of edge errors decreases (we observe this in real data as well, see Table 6.1). Both of these are desirable for clustering algorithms that try to approximate the minimum cut to find the clusters like spectral

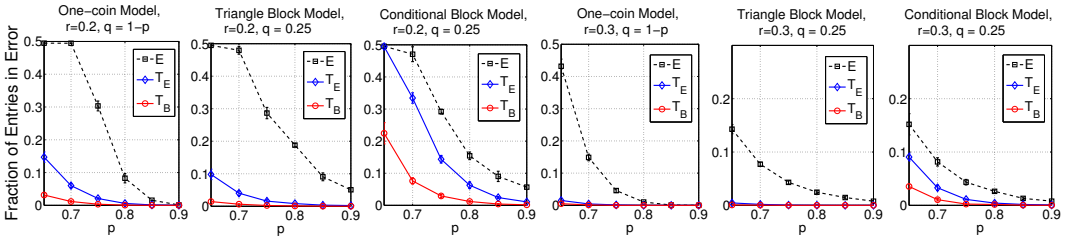


Figure 4.4: Fraction of entries in error in the matrix recovered via Program 4.4.1.

clustering.

4.3 Value of a Query

To make a meaningful comparison between edge queries and triangle queries, we need to fix a budget. Suppose we have a budget to make E edge queries. To find the number of triangle queries that can be made with the same budget, we need to define the value (cost) of a triangle query. Although a triangle query has 3 edges, they are not independent, and hence its relative cost is less than that of making 3 random edge queries. Thus we need a fair way to compare the value of a triangle query to that of an edge query.

Let $\mathbf{s} \in [0,1]^{|\mathcal{Y}|}$, $\sum_{y\in\mathcal{Y}} s_y = 1$ be the probability mass function (pmf) of the observation in a query, with $s_y := \mathbf{Pr}(y) = \sum_{x\in\mathcal{X}} \mathbf{Pr}(y|x)\mathbf{Pr}(x)$. We define the *value* of a query as the information obtained from the observation, which is measured by its *entropy*: $H(\mathbf{s}) = -\sum_{i\in\mathcal{Y}} s_i \log(s_i)$. Ideally, the cost of a query should be proportional to the amount of information it provides. So, if E is the number of edge queries, then the number of triangle queries we can make with the same budget is $T_B = E \times H_E/H_\Delta$.

We should remark that determining the above cost requires knowledge of the generative model of the graph, which may not be available for empirical data sets. In such situations, a very reasonable cost is the relative time it takes for a worker to respond to a triangle query, compared to an edge query. (In this manner, a fixed budget means a fixed amount of time for the queries to be completed.) A good rule of thumb, which is widely supported by empirical data, is the cost of 1.5, ostensibly because in triangle queries workers need to study three images, rather than two, and so it takes them 50% longer to respond. The end result is that, for a fixed budget, triangle queries reveal twice as many edges.

4.4 Guaranteed Recovery of the True Adjacency Matrix

In this section we provide a sufficient condition for the full recovery of the adjacency matrix corresponding to the underlying true $\mathcal{G}^*$ from partially observed noisy $\mathbf{A}$ filled via random triangle queries. We consider the following convex program from [VOH14a] (Chapter 2):

$$\begin{aligned} &\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \ \ \|\mathbf{L}\|_\star + \lambda\|\mathbf{S}\|_1 \\ &\text{s. t. } 1 \geq \mathbf{L}_{i,j} \geq \mathbf{S}_{i,j} \geq 0 \ \text{ for all } i,j \in \{1,2,\dots n\}, \\ &\quad \mathbf{L}_{i,j} = \mathbf{S}_{i,j} \text{ whenever } \mathbf{A}_{i,j} = 0, \ \sum_{i,j=1}^{n} \mathbf{L}_{ij} \geq |\mathcal{R}|, \end{aligned} \tag{4.4.1}$$

where $\|.\|_\star$ is the nuclear norm (sum of the singular values of the matrix), and $\|.\|_1$ is the l_1-norm (sum of absolute values of the entries of the matrix) and $\lambda \geq 0$ is the regularization parameter. $\mathbf{L}$ is the low-rank matrix corresponding to the true cluster structure, $\mathbf{S}$ is the sparse error matrix that accounts only for the missing edges inside the clusters and $|\mathcal{R}|$ is the size of the cluster region.

When $\mathbf{A}$ is filled using a subset of random edge queries, under the SBM with parameters $\{n, n_{min}, K, p, q\}$, [VOH14a] provides the following sufficient condition for the guaranteed recovery of the true $\mathcal{G}^*$:

$$n_{\min}\, r\, (p-q) \geq \frac{1}{\lambda} \geq 2\sqrt{n}\sqrt{rq(1-rq)} + 2\sqrt{n_{\max}}\sqrt{rp(1-rp)+rq(1-rq)}, \tag{4.4.2}$$

where $n_{\min}$ and $n_{\max}$ are the sizes of the smallest and the largest clusters respectively. We extend the analysis in[VOH14a] to the case when $\mathbf{A}$ is filled via a subset of random triangle queries, and obtain the following sufficient condition:

Theorem 6. *If the following condition holds:*

$$\begin{aligned} &n_{\min}\, r_T\, (p_T - q_T) \geq \frac{1}{\lambda} \\ &\geq 3\left(2\sqrt{n}\sqrt{r_T\frac{q_T}{3}(1-r_T\frac{q_T}{3})} + 2\sqrt{n_{\max}}\sqrt{r_T\frac{p_T}{3}(1-r_T\frac{p_T}{3}) + r_T\frac{q_T}{3}(1-r_T\frac{q_T}{3})}\right), \end{aligned}$$

then Program 4.4.1 succeeds in recovering the true $\mathcal{G}^*$ with high probability.

When $\mathbf{A}$ is filled using random edge queries, the entries are independent of each other (since the edges are independent in the SBM). When we use triangle queries to fill $\mathbf{A}$, this no longer holds as the 3 edges filled from a triangle query are not

independent. The key idea of our proof is as follows: The analysis in Chapter 2 [VOH14a] relies on the independence of entries of $\mathbf{A}$ to use Bernstein-type concentration results for the sum of independent random variables and the bound on the spectral norm of random matrix with independent entries. We make the following observation: Split $\mathbf{A}$ filled via random triangle queries into three parts, $\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2 + \mathbf{A}_3$. For each triangle query, allocate one edge to each part randomly. If an edge gets queried as a part of multiple triangle queries, keep one of them randomly. Each $\mathbf{A}_i$ now contains independent entries. The edge density in $\mathbf{A}_i$ is $r_T p_T/3$ and $r_T q_T/3$ inside the clusters and outside respectively. This allows us to use the results on concentration of sum of independent random variables and the $\mathcal{O}(\sqrt{n})$ bound on the spectral norm of random matrices, with a penalty due to triangle inequality for spectral norm.

It can be seen that, when the number of revealed edges is the same ($r_T = r$) and the probability of correctly identifying edges is the same ($p_T = p$ and $1 - q_T = 1 - q$), then the reovery condition of Theorem 6 is worse than that of (4.4.2). (This is expected, since triangle queries yield dependent edges.) However, it is overcompensated by the fact that triangle queries result in more reliable edges ($p_T - q_T > p - q$) and also reveal more edges ($r_T > r$, since the relative cost is less than 3).

To illustrate this, consider a graph on $n = 600$ nodes with $K = 3$ clusters of equal size $m = 200$. We generate the adjacency matrices from different models in Section 4.2 for varying p from 0.65 to 0.9. For the one-coin models, $1 - \zeta = p$. For the rest of the models $q = 0.25$. We run the improved convex program (4.4.1) by setting $\lambda = 1/\sqrt{n}$. Figure 4.4 shows the fraction of the entries in the recovered matrix that are wrong compared to the true adjacency matrix for $r = 0.2$ and 0.3 (averaged over 5 runs; $T_E = \lceil E/3 \rceil$ and $T_B = EH_E/H_\Delta$). We note that the error drops significantly when $\mathbf{A}$ is filled via triangle queries than via edge queries.

4.5 Performance of Spectral Clustering: Simulated Experiments

We generate adjacency matrices from the edge query and the triangle query models (Section 4.2) and run the spectral clustering algorithm [NJW02] on them. We compare the output clustering with the ground truth via *variation of information* (VI) [Mei07] which is defined for two clusterings (partitions) of a dataset and has information theoretical justification. Smaller values of VI indicate a closer match and a VI of 0 means that the clusterings are identical. We compare the performance of the spectral clustering algorithms on the partial adjacency matrices obtained from

querying: (1) $E = \lceil r\binom{n}{2} \rceil$ random edges, (2) $T_B = E \times H_E/H_\Delta$ random triangles, which has the same budget as querying E edges and (3) $T_E = \lceil E/3 \rceil < T_B$ random triangles, which has same number of edges as in the adjacency matrix obtained by querying E edges.

Varying Edge Density Inside the Clusters: Consider a graph on $n = 450$ nodes with $K = 3$ clusters of equal size $m = 150$. We vary edge density inside the cluster p from 0.55 to 0.9. For the one-coin models, $1 - \zeta = p$, and $q = 0.25$ for the rest. Figure 4.5 shows the performance of spectral clustering for $r = 0.15$ and $r = 0.3$ (averaged over 5 runs).

Varying Cluster Sizes: Let $N = 1200$. Consider a graph with K clusters of equal

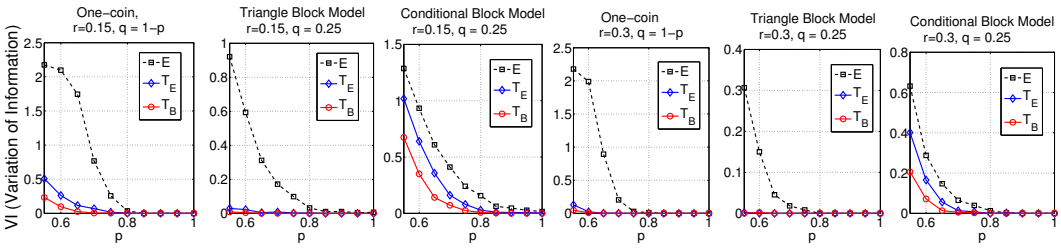


Figure 4.5: VI for Spectral Clustering output for varying edge density inside the clusters.

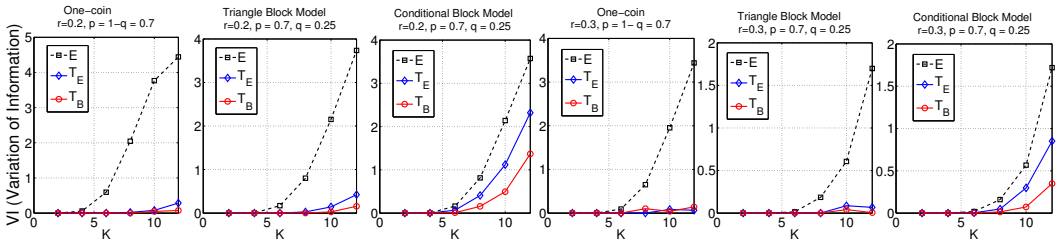


Figure 4.6: VI for Spectral Clustering output for varying number of clusters (K).

sizes $m = \lfloor N/K \rfloor$ and $n = K\,m$. We vary K from 2 to 12 which varies the cluster sizes from 600 (large clusters) to 100 (small clusters, note that $\sqrt{1200} \approx 35$). We set $p = 0.7$. For the one-coin models $1-\zeta = p$ and $q = 0.25$ for the rest. Figure 4.6 shows the performance of spectral clustering for $r = 0.2$ and 0.3. The performance is significantly better with triangle queries compared to that with edge queries.

4.6 Experiments on Real Data

We use Amazon Mechanical Turk as crowdsourcing platform. For edge queries, each HIT (Human Intelligence Task) has 30 queries of random pairs, a sample is shown in Figure 4.1. For triangle queries, each HIT has 20 queries, with each query having 3 random images, a sample is shown in Figure 5.1. Each HIT is answered

E: Edge, T: Δ	# Workers	# Unique Edges	**% of Edges Seen**	**% of Edge Errors**
Dogs3, Edge Query	300	$E' = 8630$	7.73%	25.2%
Dogs3, Δ Query	150	$3T'_E = 8644$	7.74%	19.66%
Dogs3, Δ Query	320	$3T' = 17,626$	15.79%	20%
Birds5, Edge Query	300	$E' = 8319$	14.27%	14.82%
Birds5, Δ Query	155	$3T'_E = 8600$	14.74%	10.96%
Birds5, Δ Query	285	$3T' = 14,773$	25.34%	11.4%

Table 4.3: Summary of the data colleced in the real experiments.

Query (E: Edge, T: Δ)	k-means	Spectral Clustering	Convex Program
$E' = 8630$	0.8374 ± 0.0121 (K=2)	0.6972 ± 0 (K = 3)	0.5176 ± 0 (K=3)
$3T'_E = 8644$	0.6675 ± 0.0246 (K=3)	0.5690 ± 0 (K=3)	0.4605 ± 0 (K = 3)
$3T' = 17626$	$\mathbf{0.3268} \pm 0$ (K=3)	$\mathbf{0.3470} \pm 0$ (K=3)	$\mathbf{0.2279} \pm 0$ (K = 3)

Table 4.4: VI for clustering output by k-means and spectral clustering for the Dogs3 dataset.

Query	k-means	Spectral Clustering	Convex Program
$E' = 8319$	1.4504 ± 0.0338 (K = 2)	1.2936 ± 0.0040 (K = 4)	1.0392 ± 0 (K = 4)
$3T'_E = 8600$	1.1793 ± 0.0254 (K = 3)	$1.1299 \pm 0 (K = 4)$	$\mathbf{0.9105} \pm 0$ (K=4)
$3T' = 14,773$	$\mathbf{0.7989} \pm 0$ (K = 4)	$\mathbf{0.8713} \pm 0$ (K = 4)	0.9135 ± 0 (K = 4)

Table 4.5: VI for clustering output by k-means and spectral clustering for the Birds5 dataset.

by a unique worker. Note that we do not provide any examples of different classes or any training to do the task. We fill $\mathbf{A}$ as described in Section 4.2.3 and run the k-means, the Spectral Clustering and Program 2.1.7 followed by Spectral Clusteirng on it. Since we do not know the model parameters and hence have no access to the entropy information, we can use the the average time taken as the "cost" or value of the query. For E edge comparisons, the equivalent number of triangle comparisons would be $T = E \times t_E/t_\Delta$, where t_E and t_Δ are average time taken to answer an edge query and a triangle query respectively. We consider two datasets:

1. **Dogs3** dataset has images of the following 3 breeds of dogs from the Stanford Dogs Dataset [Kho+11]: Norfolk Terrier (172), Toy Poodle (150), and Bouvier des Flanders (151), giving a total of 473 dogs images. On an average a worker took $t_E = 8.4s$ to answer an edge query and $t_\Delta = 11.7s$ to answer a triangle query.

2. **Birds5** dataset has 5 bird species from CUB-200-2011 dataset [Wah+11]: Laysan Albatross (60), Least Tern (60), Artic Tern (58), Cardinal (57), and Green Jay (57). We also add 50 random species as outliers, giving us a total if 342 bird images. On an average, workers took $t_E = 8.3s$ to answer one edge query and $t_\Delta = 12.1s$ to answer a triangle query.

Details of the data obtained from edge query and triangle query experiments is summarized in Table 6.1. Note that the error in the revealed edges drop significantly for triangle queries.

For the Dogs3 dataset, the empirical edge densities inside and between the clusters for $\mathbf{A}$ obtained from the edge queries ($\hat{P}_E$) and the triangle queries ($\hat{P}_T$) is:

$$\hat{P}_E = \begin{bmatrix} 0.7577 & 0.1866 & 0.2043 \\ 0.1866 & 0.6117 & 0.2487 \\ 0.2043 & 0.2487 & 0.7391 \end{bmatrix}, \; \hat{P}_T = \begin{bmatrix} 0.7139 & 0.1138 & 0.1253 \\ 0.1138 & 0.6231 & 0.1760 \\ 0.1253 & 0.1760 & 0.7576 \end{bmatrix}.$$

For the Birds5 dataset, the emprical edge densities within and between various clusters in $\mathbf{A}$ filled via edge queries ($\hat{P}_E$) and triangle queries ($\hat{P}_T$) are:

$$\hat{P}_E = \begin{bmatrix} 0.801 & 0.304 & 0.208 & 0.016 & 0.032 & 0.100 \\ 0.304 & 0.778 & 0.656 & 0.042 & 0.131 & 0.123 \\ 0.208 & 0.656 & 0.912 & 0.062 & 0.094 & 0.096 \\ 0.016 & 0.042 & 0.062 & 0.855 & 0.154 & 0.110 \\ 0.032 & 0.131 & 0.094 & 0.154 & 0.958 & 0.158 \\ 0.100 & 0.123 & 0.096 & 0.110 & 0.158 & 0.224 \end{bmatrix}, \; \hat{P}_T = \begin{bmatrix} 0.786 & 0.207 & 0.151 & 0.011 & 0.021 & 0.058 \\ 0.207 & 0.797 & 0.625 & 0.023 & 0.047 & 0.1 \\ 0.151 & 0.625 & 0.865 & 0.024 & 0.06 & 0.071 \\ 0.011 & 0.023 & 0.024 & 0.874 & 0.059 & 0.076 \\ 0.021 & 0.047 & 0.06 & 0.059 & 0.943 & 0.08 \\ 0.058 & 0.1 & 0.071 & 0.076 & 0.08 & 0.182 \end{bmatrix}.$$

As we see the triangle queries give rise to an adjacency matrix with significantly less confusion across the clusters (compare the off-diagonal entries in $\hat{P}_E$ and $\hat{P}_T$).

Tables 4.4 and 4.5 show the performance of clustering algorithms (in terms of variation of information) for the two datasets. The number of clusters found is given in brackets. We note that for both the datasets, the performance is significantly better with triangle queries than with edge queries. Furthermore, even with less triangle queries ($3T'_E \approx E$) than that is allowed by the budget, the clustering obtained is better compared to edge queries.

4.7 Summary

In this chapter, we compared two ways of querying for crowdsourced clustering using non-experts: random edge comparisons and random triangle comparisons. We provided simple and intuitive models for both. Compared to edge queries that reveal independent entries of the adjacency matrix, triangle queries reveal dependent

ones (edges in a triangle share a vertex). However, due to their error-correcting capabilities, triangle queries result in more reliable edges and, furthermore, because the cost of a triangle query is less than that of 3 edge queries, for a fixed budget, triangle queries reveal many more edges. Simulations based on our models, as well as empirical evidence strongly support these facts. In particular, experiments on two real datasets suggest that clustering items from random triangle queries significantly outperforms random edge queries when the total query budget is fixed. We also provided theoretical guarantee for the exact recovery of the true adjacency matrix using random triangle queries. In the next chapter (Chapter 5), we explore how to exploit the structure of triangle queries via tensor representations for improved clustering performance.

Chapter 5

CROWDSOURCED CLUSTERING: TENSOR EMBEDDING FOR TRIANGLE QUERIES

In this chapter, we consider the problem of crowdsourced clustering of a set of items based on queries of the similarity of triple of objects. We introduced this approach, called triangle queries, in Chapter 4 and showed that, for a fixed query budget, it outperforms clustering based on edge queries (i.e, comparing pairs of objects). In Chpater 4, the clustering algorithm for triangle and edge queries was identical and each triangle query response was treated as 3 separate edge query responses. In this chapter, we directly exploit the triangle structure of the responses by embedding them into a 3-way tensor. Since there are 5 possible responses to each triangle query, it is a priori not clear how best to embed them into the tensor. We give sufficient conditions on non-trivial embedding such that the resulting tensor has a rank equal to the underlying number of clusters (akin to what happens with the rank of the adjacency matrix). We then propose an alternating least squares tensor decomposition algorithm to cluster a noisy and partially observed tensor and show, through extensive numerical simulations, that it significantly outperforms methods that make use only of the adjacency matrix. This chapter is based on our paper [VZH17].

5.1 Introduction

Crowdsourcing – the process of collecting data from workers on platforms such as Amazon Mechanical Turk for various applications has recently become quite popular [Ray+10; Sno+08]. The workers on these platforms are often *non-experts* and hence the answers obtained will be *noisy*. Therefore both problems of designing *queries* and designing *algorithms* for inferring quality data from such non-expert workers are of importance. In Chapter 4 [VH16a] we considered the problem of query design for crowdsourced clustering and showed that for a fixed query budget, we can obtain better quality answers (and hence better clustering) by making random triangle queries, where three items are compared per query (Figure 5.1) as compared to making random edge queries where a pair of items are compared. However, in Chapter 4, the information obtained from the triangle queries was embedded into an adjacency matrix which was input to graph clustering algorithms.

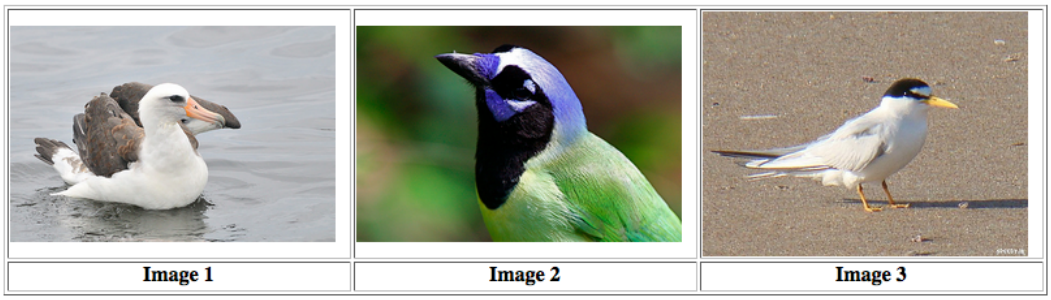


Figure 5.1: Example of a triangle query.

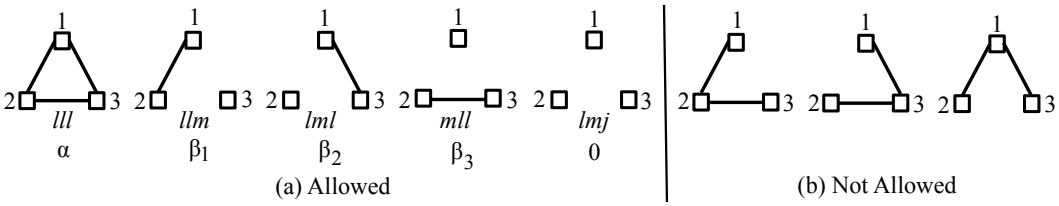


Figure 5.2: Configurations for a triangle query that are (a) observed and (b) not allowed.

Such an embedding treats each triangle query as 3 separate edges and ignores the triangle structure itself. A more natural embedding is to consider a tensor where the query result for each triple of items $\{i, j, k\}$ is embedded into the ijk-th entry of a 3-way tensor. In this chapter we study tensor embedding for triangle queries.

Entry A_{ij} of an adjacency matrix of a graph holds information about the *pair* of nodes $\{i, j\}$, which has two possible configurations, *edge*, encoded by $A_{ij} = 1$ and *no edge*, encoded by $A_{ij} = 0$. The true adjacency matrix, $\mathbf{A}^*$, obtained by this simple encoding has a low-rank structure that reflects the underlying clusters and the rank is equal to the number of clusters. A triangle query has 5 possible answers (Figure 5.2(a)): (1) All items are similar, denoted by lll, (2) Items 1 & 2 are similar, denoted by llm, (3) Items 1 & 3 are similar, denoted by lml, (4) Items 2 & 3 are similar, denoted by mll, and (5) None are similar, denoted by lmj. So, we need an encoding scheme with 5 alphabets to embed the information obtained from a triangle query. Moreover, we also would like the true tensor, $\mathbf{T}^*$, obtained by this embedding, to have a low-rank structure that reflects the underlying clusters.

Our Contributions: We propose a general encoding scheme for filling a tensor from triangle queries (Section 5.3.1) and provide sufficient conditions on this encoding scheme to give a true tensor with unique (up to scaling and permutations) CP-decomposition of rank equal to the number of clusters (Section 5.3.2, 5.3.3). We also provide extensive numerical simulations (Section 5.4) that show that using tensor decomposition methods can improve over clustering obtained via the adjacency matrix.

5.2 Tensors: A Quick Recap

A *tensor* is a multidimensional array. [KB09] provides a very good survery on tensors. In this chapter we focus on 3-way tensors and in this section we recall a few properties of tensors that are relevant for our results.

A rank-1 matrix is an outer product of two vectors $\mathbf{x} \otimes \mathbf{y} = \mathbf{x}\mathbf{y}^\top$, with $(\mathbf{x} \otimes \mathbf{y})_{ij} = x_i y_j$. Similarly, a rank-1 tensor is an outer product of 3 vectors $\mathbf{x} \otimes \mathbf{y} \otimes \mathbf{z}$ with $(\mathbf{x} \otimes \mathbf{y} \otimes \mathbf{z})_{ijk} = x_i y_j z_k$. A rank-$K$ tensor, $\mathbf{T}$, can be written as a sum of K rank-1 tensors (CP-decomposition):

$$\mathbf{T} = \sum_{l=1}^{K} \mathbf{u}_l \otimes \mathbf{v}_l \otimes \mathbf{w}_l = \mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W},$$

where $\mathbf{u}_l, \mathbf{v}_l, \mathbf{w}_l \in \mathbb{C}^n$, $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_K]$, $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_K]$ and $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_K]$. Recall that the *Kruskal rank* of a matrix $\mathbf{A}$, denoted by krank$(\mathbf{A})$, is the maximal number K, such that any set of K columns of $\mathbf{A}$ is linearly independent. The CP-decomposition of a tensor is unique up to scaling and permutations of the factors under mild conditions:

Theorem 7 (Kruskal). *[Kru77; SBG00] The CP-decomposition of a $n \times n \times n$ tensor, $\mathbf{T} = \mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W}$ (with $\mathbf{U}, \mathbf{V}, \mathbf{W}$ being $n \times K$ matrices), is unique up to scaling and permutations if krank$(\mathbf{U})$ + krank$(\mathbf{V})$ + krank$(\mathbf{W}) \geq 2K + 2$.*

5.3 Tensor Embedding for Triangle Queries

In this section we present a scheme to encode the answers to the triangle queries in a tensor and provide sufficient conditions that guarantee a unique (up to scaling and permutations) rank-K CP-decomposition of the true tensor.

5.3.1 Encoding Scheme for Embedding Triangle Queries

Recall that a triangle query has 5 possible configurations (Figure 5.2(a)). We propose the following encoding scheme to embed the response to the query $\{i, j, k\}$:

1. If $\{i, j, k\}$ are in the same cluster, $T_{ijk} = \alpha \neq 0$.
2. If i, j are in the same cluster but k is not, $T_{ijk} = \beta_1 \neq 0$.
3. If i, k are in the same cluster but j is not, $T_{ijk} = \beta_2 \neq 0$.
4. If j, k are in the same cluster but i is not, $T_{ijk} = \beta_3 \neq 0$.
5. If $\{i, j, k\}$ are all in different clusters, $T_{ijk} = 0$.

And $T_{iii} = \alpha,\ \forall i$. Note that $\mathbf{T}$ is *not* a symmetric tensor in general. However it does have the following symmetries:

1. If $\{i, j, k\}$ is of the configuration lll, then $T_{ijk} = T_{jik} = T_{ikj} = T_{jki} = T_{kij} = T_{kji} = \alpha$. Also, $T_{iij} = T_{iji} = T_{jii} = \alpha$, and similarly for all the permutations of $\{jji, iik, kki, jjk, kkj\}$.
2. If $\{i, j, k\}$ is of the configuration llm, then $T_{ijk} = T_{jik} = \beta_1$, $T_{ikj} = T_{jki} = \beta_2$, and $T_{kij} = T_{kji} = \beta_3$. Further, $T_{iij} = T_{jii} = T_{jij} = T_{jji} = T_{jij} = T_{ijj} = \alpha$. Also, $T_{iik} = T_{jjk} = \beta_1$, $T_{iki} = T_{jkj} = \beta_2$, $T_{kii} = T_{kjj} = \beta_3$,

similarly for other configurations.

5.3.2 Low-Rank Tensor Structure

Consider a graph on n items with K disjoint clusters. Let $\mathbf{c}_i \in \mathbb{R}^n$ denote the indicator vector of cluster i. That is, $\mathbf{c}_{i_j} = 1$ if node $j \in$ cluster i and 0 otherwise. Let $\mathbf{C} := [\mathbf{c}_1, \cdots, \mathbf{c}_K] \in \mathbb{R}^{n \times K}$. Note that each row of $\mathbf{C}$ has a single 1, since each item belongs to only one cluster. Let $\mathbf{T}^*$ be the full tensor filled using the scheme in Section 5.3.1 via triangle queries when there is no noise:

$$\begin{aligned}\mathbf{T}^* = \alpha \sum_{l=1}^{K} \mathbf{c}_l \otimes \mathbf{c}_l \otimes \mathbf{c}_l + \beta_1 \sum_{l=1}^{K} \sum_{\substack{m=1 \\ m \neq l}}^{K} \mathbf{c}_l \otimes \mathbf{c}_l \otimes \mathbf{c}_m \\ + \beta_2 \sum_{l=1}^{K} \sum_{\substack{m=1 \\ m \neq l}}^{K} \mathbf{c}_l \otimes \mathbf{c}_m \otimes \mathbf{c}_l + \beta_3 \sum_{l=1}^{K} \sum_{\substack{m=1 \\ m \neq l}}^{K} \mathbf{c}_m \otimes \mathbf{c}_l \otimes \mathbf{c}_l.\end{aligned} \tag{5.3.1}$$

The true adjacency matrix, $\mathbf{A}^* = \sum_{l=1}^{K} \mathbf{c}_l \mathbf{c}_l^\top = \mathbf{C}\mathbf{C}^\top$, has a low-rank structure, with $\text{rank}(\mathbf{A}^*) = K$ being the number of clusters. Our goal is to understand if the

true tensor $\mathbf{T}^*$ (5.3.1) has such a low-rank structure. In particular, we want to write $\mathbf{T}^*$ as:

$$\mathbf{T}^* = \sum_{l=1}^{K} \mathbf{u}_l \otimes \mathbf{v}_l \otimes \mathbf{w}_l = \mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W},$$

where $\mathbf{u}_l, \mathbf{v}_l, \mathbf{w}_l \in \mathbb{C}^n$, $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_K]$, $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_K]$ and $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_K]$.

The following theorem provides the conditions on the encoding scheme (Section 5.3.1) that is sufficient for $\mathbf{T}^*$ to have a unique rank-K CP-decomposition.

Theorem 8. *For the encoding scheme in Section 5.3.1, $\mathbf{T}^*$ is a rank K tensor for $K \geq 2$ with* unique *(up to scaling and permutations) CP-decomposition if the following hold:*

1. $\beta_1\beta_2 + \beta_2\beta_3 + \beta_3\beta_1 \neq 0$.
2. $\alpha = \beta_1 + \beta_2 + \beta_3 - K\frac{\beta_1\beta_2\beta_3}{\beta_1\beta_2+\beta_2\beta_3+\beta_3\beta_1}$.
3. $\beta_1 \neq -\beta_2,\ \beta_2 \neq -\beta_3,\ \beta_3 \neq -\beta_1$.

5.3.3 Proof of Theorem 8

We observe that $\mathbf{T}^*$ (5.3.1) can be re-written as $\mathbf{T}^* = \sum_{l,m,n=1}^{K} \mathbf{B}_{lmn}\mathbf{c}_l \otimes \mathbf{c}_m \otimes \mathbf{c}_n$, where $\mathbf{B}$ is a $K \times K \times K$ tensor (Tucker Decomposition [KB09]). Suppose we can write $\mathbf{B}$ as a sum of K rank-1 tensors:

$$\mathbf{B} = \sum_{l=1}^{K} \mathbf{f}_l \otimes \mathbf{g}_l \otimes \mathbf{h}_l = \mathbf{F} \otimes \mathbf{G} \otimes \mathbf{H}, \tag{5.3.2}$$

where $\mathbf{f}_l, \mathbf{g}_l, \mathbf{h}_l \in \mathbb{C}^K$, $\mathbf{F} := [\mathbf{f}_1, \cdots, \mathbf{f}_K]$, $\mathbf{G} := [\mathbf{g}_1, \cdots, \mathbf{g}_K]$, and $\mathbf{H} := [\mathbf{h}_1, \cdots, \mathbf{h}_K]$. Then, $\mathbf{T}^*$ has the following rank-K CP decomposition: $\mathbf{T}^* = (\mathbf{CF}) \otimes (\mathbf{CG}) \otimes (\mathbf{CH})$. So, proving the following lemma is sufficient to prove Theorem 8.

Lemma 5.3.1. *For the encoding scheme in Section 5.3.1, $\mathbf{B}$ is a rank K tensor for $K \geq 2$ with unique (up to scaling and permutations) CP-decomposition if the conditions in Theorem 8 are satisfied.*

Proof. We prove Lemma 5.3.1 by first constructing $\mathbf{F}, \mathbf{G}, \mathbf{H} \in \mathbb{C}^{K\times K}$ that satisfy (5.3.2) and then showing that it is unique.

Construction: Comparing with (5.3.1), we note that the $l-$th panel of $\mathbf{B}$, where the third index is kept fixed to l (which gives a matrix), has the following structure:

$$\mathbf{B}_{(:,:,l)} = \begin{bmatrix} \beta_1 & 0 & \cdots & \beta_3 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \beta_1 & \beta_3 & 0 & \cdots & 0 \\ \beta_2 & \cdots & \beta_2 & \alpha & \beta_2 & \cdots & \beta_2 \\ 0 & \cdots & 0 & \beta_3 & \beta_1 & \cdots & 0 \\ 0 & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \beta_3 & 0 & \cdots & \beta_1 \end{bmatrix}$$
$$= \beta_1 \mathbf{I} + \begin{bmatrix} \mathbf{1} & \mathbf{e}_l \end{bmatrix} \begin{bmatrix} 0 & \beta_3 \\ \beta_2 & K\delta \end{bmatrix} \begin{bmatrix} \mathbf{1}^\top \\ \mathbf{e}_l^\top \end{bmatrix}, \tag{5.3.3}$$

where $\delta := \frac{\alpha - \beta_1 - \beta_2 - \beta_3}{K}$, $\mathbf{1}$ is a vector of all 1's and $\mathbf{e}_l$ is the standard vector with $\mathbf{e}_l(l) = 1$ and all other entries 0. We note that $\mathbf{B}$ is a *circulant* tensor in the following sense:

$$\mathbf{B}_{(:,:,l+1)} = Z\mathbf{B}_{(:,:,l)}Z^\top, \text{ where } Z := \begin{bmatrix} 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}.$$

This is analogous to a circulant matrix in which the columns get circularly shifted. Note that from (5.3.2), we can write,$\mathbf{B}_{(:,:,l)} = \mathbf{F}\mathbf{D}_{\mathbf{h}_l}\mathbf{G}^\top$, where $\mathbf{D}_{\mathbf{h}_l} = \text{diag}(h_{l1}, \cdots, h_{lK})$. The circulant structure of $\mathbf{B}$ suggests a circulant structure for the factors. Since circulant matrices are diagonalized by Fourier transforms, we may write: $\mathbf{F} := \mathscr{F}\mathbf{\Lambda}\mathscr{F}^\dagger$ and $\mathbf{G}^\top := \mathscr{F}\mathbf{\Xi}\mathscr{F}^\dagger$, where $\mathscr{F}$ is $K \times K$ Fourier matrix normalized with $1/\sqrt{K}$, $\mathbf{\Lambda}$ and $\mathbf{\Xi}$ are diagonal matrices. So, $\mathscr{F}^\dagger\mathbf{B}_{(:,:,l)}\mathscr{F} = \mathbf{\Lambda}\mathscr{F}^\dagger\mathbf{D}_{\mathbf{h}_l}\mathscr{F}\mathbf{\Xi} = \mathbf{\Lambda}\mathbf{A}^{(l)}\mathbf{\Xi}$, where $\mathbf{A}^{(l)} := \mathscr{F}^\dagger\mathbf{D}_{\mathbf{h}_l}\mathscr{F}$ is a circulant matrix. Using (5.3.3), we can verify that $\mathscr{F}^\dagger\mathbf{B}_{(:,:,l)}\mathscr{F}$ has the following structure:

$$\begin{bmatrix} \beta_1 + \beta_3 + \beta_2 + \delta & (\beta_3 + \delta)\eta^{(l-1)} & \cdots & (\beta_3 + \delta)\eta^{(K-1)(l-1)} \\ (\beta_2 + \delta)\eta^{-(l-1)} & \beta_1 + \delta & \cdots & \delta\eta^{(K-2)(l-1)} \\ (\beta_2 + \delta)\eta^{-2(l-1)} & \delta\eta^{-(l-1)} & \cdots & \delta\eta^{(K-3)(l-1)} \\ \vdots & \vdots & \ddots & \vdots \\ (\beta_2 + \delta)\eta^{-(K-1)(l-1)} & \delta\eta^{-(K-1)(l-1)} & \cdots & \beta_1 + \delta \end{bmatrix},$$

where $\eta = e^{j2\pi/K}$, the $K-$th root of unity. Note that the submatrix obtained by removing first row and first column of the above matrix is toeplitz. So, the corresponding submatrix of $\mathbf{\Lambda}\mathbf{A}^{(l)}\mathbf{\Xi}$, where $\mathbf{A}^{(l)}$ is circulant, should also be toeplitz.

This gives us conditions (we omit the details for reasons of space) using which we can show that $\mathbf{\Lambda} = \text{diag}(\lambda, 1, \cdots, 1)$ and $\mathbf{\Xi} = \text{diag}(\mu, 1, \cdots, 1)$. By comparing the entries of $\mathscr{F}^\dagger \mathbf{B}_{(:,:,l)} \mathscr{F}$ to those of $\mathbf{\Lambda} \mathbf{A}^{(l)} \mathbf{\Xi}$, the following should hold:

$$\beta_1 + \beta_2 + \beta_3 + \delta = \lambda\mu(\beta_1 + \delta),\ \beta_2 + \delta = \mu\delta,\ \beta_3 + \delta = \lambda\delta.$$

Note that, if $\delta = 0$, then $\beta_2 = \beta_3 = 0$, which is not allowed in the scheme considered. So, assuming $\delta \neq 0$, we get:

$$\lambda = \frac{\beta_3 + \delta}{\delta},\ \mu = \frac{\beta_2 + \delta}{\delta},\ \beta_1 + \beta_3 + \beta_2 + \delta = \lambda\mu(\beta_1 + \delta).$$

Using the expressions for λ and μ, we can solve for δ and hence α in terms of $\beta_i,\ i = 1, 2, 3$, and K:

$$\begin{aligned} \delta &= \frac{-\beta_1\beta_3\beta_2}{\beta_1\beta_3 + \beta_3\beta_2 + \beta_1\beta_2},\ \beta_1\beta_2 + \beta_2\beta_3 + \beta_3\beta_1 \neq 0 \\ \alpha &= \beta_1 + \beta_3 + \beta_2 - K\frac{\beta_1\beta_3\beta_2}{\beta_1\beta_3 + \beta_3\beta_2 + \beta_1\beta_2}. \end{aligned} \tag{5.3.4}$$

Recall that $\mathbf{A}^{(l)} = \mathscr{F}^\dagger \mathbf{D}_{\mathbf{h}_l} \mathscr{F}$. So, the diagonal of $\mathscr{F}\mathbf{A}^{(l)}\mathscr{F}^\dagger$ gives the $l-$th row of $\mathbf{H}$. More elaborate calculations, omitted here for the reasons of space, show that $\mathbf{h}_l = \beta_1\mathbf{1} + K\delta\mathbf{e}_l$. Thus, $\mathbf{H}$ is a circulant matrix (as expected) with eigenvalues: $K\delta[(\beta_1 + \delta)/\delta, 1, \cdots, 1]$ (Fourier transform of the first row).

Uniqueness: If $\mathbf{F}, \mathbf{G}, \mathbf{H}$ are full rank, then their kruskal rank is K, and hence from Theorem 7, the CP-decomposition of $\mathbf{B}$ (5.3.2) is unique. If $\beta_3 + \delta \neq 0$ (i.e, $\lambda \neq 0$) and $\beta_2 + \delta \neq 0$ (i.e, $\mu \neq 0$), then $\mathbf{F}$ and $\mathbf{G}$ are full rank. Further, if $\beta_1 + \delta \neq 0$, and $\delta \neq 0$, then $\mathbf{H}$ is also full rank. Using the expression for δ in (5.3.4), these sufficient conditions translate to $\beta_1 \neq -\beta_2, \beta_1 \neq -\beta_3, \beta_2 \neq -\beta_3$. ■

5.3.4 Discussion

1. The encoding scheme is a function of $\{\beta_1, \beta_2, \beta_3, K\}$ as they fix the value of α.

2. $\mathbf{T}^*$ with the encoding considered is not orthogonal in general, i.e, the factors $\mathbf{F}, \mathbf{G}, \mathbf{H}$ of $\mathbf{B}$ and hence $\mathbf{U}, \mathbf{V}, \mathbf{W}$ of $\mathbf{T}^*$ might not be orthogonal.

3. $\mathbf{T}^*$ with the encoding considered is not symmetric in general, unless $\beta_1 = \beta_2 = \beta_3$.

4. It is interesting to note that we can recover the adjacency matrix $\mathbf{A}$ from $\mathbf{T}$, even when we encode all single edge results to the same, i.e, $\beta_1 = \beta_2 = \beta_3 = \beta \implies \alpha = (3 - K/3)\,\beta$, as long as $\alpha \neq \beta$ (when $K = 6$).

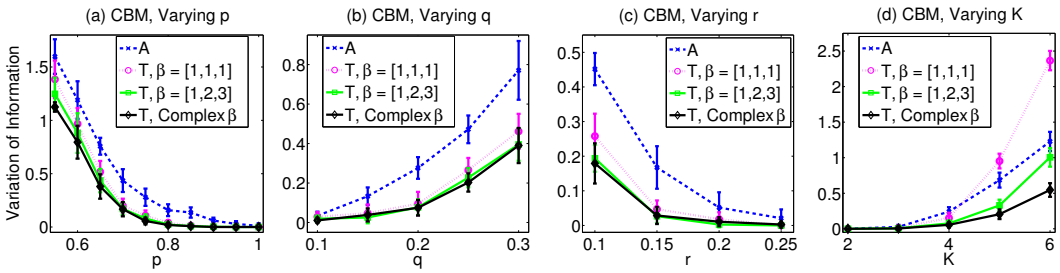


Figure 5.3: Comparison of VI (averaged over 10 experiments) for clustering using the tensor (filled using different encoding schemes) compared to that obtained using adjacency matrix, for varying different parameters for the Conditional Block Model (Section 5.4.3).

5. In general, let $\beta_i = \beta e^{-j\phi_i}$ and $\gamma := e^{j\phi_1} + e^{j\phi_2} + e^{j\phi_3}$. Then from (5.3.4), $\alpha = \beta(|\gamma|^2 - K)/\bar{\gamma}$.

5.4 Numerical Experiments

In this section we numerically compare the performance of clustering on the adjacency matrix $\mathbf{A}$ to that obtained using the tensor $\mathbf{T}$, both filled by the same set of triangle queries. Let r determine the sparsity (to fix a budget on the number of triangle queries). We generate answers to $\lceil \frac{r}{3}\binom{n}{2} \rceil$ random triangle queries using two different models (described in Section 5.4.2). $\mathbf{T}$ is filled using the encoding scheme and the symmetries described in Section 5.3.1 for three different encodings: $\beta = [1, 1, 1]$, $\beta = [1, 2, 3]$ and $\beta = [1, \frac{-1+i}{\sqrt{2}}, \frac{-1-i}{\sqrt{2}}]$. Note that both $\mathbf{A}$ and $\mathbf{T}$ have a lot of missing entries as only a small subset of triples are observed. We use spectral clustering [NJW02] on $\mathbf{A}$ (unobserved entries set to 0).

5.4.1 Clustering via Tensor Decomposition

Let Ω be the set of triangle queries and $\mathbf{\Omega}$ be an $n \times n \times n$ tensor with $\Omega_{ijk} = 1$ if T_{ijk} is observed. We consider the following simple CP-decomposition objective:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \sum_{i,j,k=1}^{n} \{\Omega_{ijk}(T_{ijk} - \sum_{l=1}^{K} u_{il}v_{jl}w_{kl})\}^2, \tag{5.4.1}$$

which is a non-convex optimization problem. Note that we fix the number of clusters. We solve (5.4.1) iteratively using alternating least squares (ALS). In each time step t, ALS updates $\mathbf{U}^t, \mathbf{V}^t, \mathbf{W}^t$, one variable at a time assuming the other two to be fixed. So, assuming $\mathbf{U} = \mathbf{U}^{t-1}, \mathbf{V} = \mathbf{V}^{t-1}$ to be fixed, we can rewrite the objective (5.4.1) as: $\min_{\mathbf{W}} \sum_{i,j,k=1}^{n} \{\Omega_{ij,k}(T_{ij,k} - \sum_{l=1}^{K} M_{ij,l}w_{kl})\}^2$, where $M_{ij,l} := u_{il}^{t-1}v_{jl}^{t-1}$. Note that $\mathbf{M} = \mathbf{U}^{t-1} \odot \mathbf{V}^{t-1} \in \mathbb{C}^{n^2 \times K}$, where $\odot$ denotes

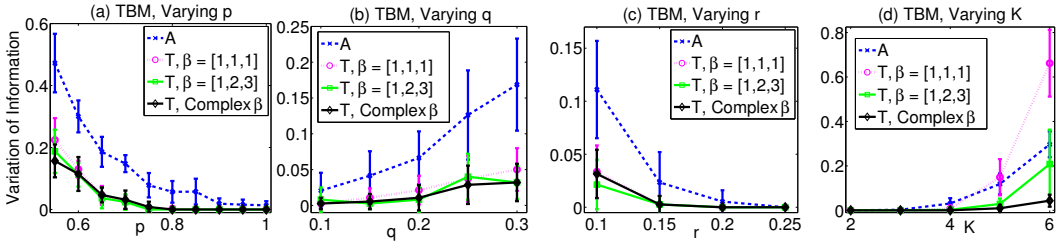


Figure 5.4: Comparison of VI (averaged over 10 experiments) for clustering using the tensor (filled using different encoding schemes) compared to that obtained using adjacency matrix, for varying different parameters for the Triangle Block Model (Section 5.4.3)

Khatri-Rao product. Let $\mathbf{\Omega}^{(w)}, \mathbf{T}^{(w)} \in \mathbb{C}^{n^2 \times n}$ be matricized $\mathbf{\Omega}$ and $\mathbf{T}$ respectively, whose rows are indexed by ij and columns by k of the tensors. Define, $\mathbf{D}_k^{(w)} := \text{diag}(\mathbf{\Omega}^{(w)}(:,k))$, where $(:,k)$ stands for k-th column. Each row of $\mathbf{W}$, equivalently, each column of $\mathbf{W}^\top$ can be solved for by the following least squares problem: $||\mathbf{D}_k^{(w)}\mathbf{T}^{(w)}(:,k) - \mathbf{D}_k^{(w)}\mathbf{M}(\mathbf{W}(k,:))^\top||^2$. We use the clustering $\mathbf{C}^0$ output by the spectral clustering on $\mathbf{A}$ to initialize: $[\mathbf{U}^0, \mathbf{V}^0, \mathbf{W}^0] = [\mathbf{C}^0\mathbf{F}, \mathbf{C}^0\mathbf{G}, \mathbf{C}^0\mathbf{H}]$. We implemented the ALS algorithm on MATLAB using the sparse tensor functions from tensor toolbox [B+15; BK07]. We then run k-means on $[\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{W}}]$ obtained by ALS to get clusters from $\mathbf{T}$.

5.4.2 Models for Triangle Queries

We consider two models for generating answers to the triangle queries: *Triangle Block Model* (**TBM**) and *Conditional Block Model* (**CBM**), both of which are derived from the popular Stochastic Block Model (SBM) [HLL83].

SBM is a random graph model for a graph with disjoint clusters. Given the cluster assignements to the nodes, the edges of the graph are independently generated. Edge probability inside the clusters is p and between the clusters is q. In the context of crowdsourcing, if a worker compares items i and j that belong to same cluster, then she will correctly say they are similar with probability p. If they are not in the same cluster, then the probability that she will make an error and say they are similar is q.

For both the TBM and the CBM, given a triple $\{i, j, k\}$, the 3 edges $\{ij, jk, ki\}$ are generated using the SBM. If the configuration thus obtained is one of the 3 configurations that are not allowed (Figure 5.2(b)), then: (1) TBM assumes that the crowd worker can resolve this to the correct configuration; (2) CBM will regenerate the 3 edges until one of the allowed configurations is obtained. More detailed

descriptions of these models are available in Chapter 4.

5.4.3 Simulation Results

Consider a graph on $n = 450$ nodes with $K = 3$ clusters of equal size. We vary the following parameters:

1. **Varying p:** Let $q = 0.25, r = 0.1$. We vary the edge density inside the clusters p from 0.55 to 1 in steps of 0.05.

2. **Varying q:** Let $p = 0.7, r = 0.1$. We vary the edge density between the clusters q from 0.1 to 0.25 in steps of 0.05.

3. **Varying r:** Let $q = 0.25, p = 0.7$. We vary the sparsity parameter r from 0.1 to 0.25 in steps of 0.05.

4. **Varying K:** Consider a graph on $n = 480$ nodes with clusters of equal sizes and $K = [2, 3, 4, 5, 6]$ and hence the cluster sizes get varied. Let $q = 0.25, p = 0.7, r = 0.2$.

Figures 5.3 and 5.4 show the results for the CBM and TBM respectively. We compare the output clustering with the ground truth via *variation of information* (VI) [Mei07] which is defined for two clusterings (partitions) of a dataset and has information theoretical justification. Smaller values of VI indicate a closer match and VI $= 0$ means that the clusterings are identical. We compare clustering on $\mathbf{A}$ (dashed blue line) to that obtained by clustering $\mathbf{T}$ when $\beta = [1, 1, 1]$ (dotted pink line), $\beta = [1, 2, 3]$ (solid green line) and complex $\beta = [1, \frac{-1+i}{\sqrt{2}}, \frac{-1-i}{\sqrt{2}}]$ (solid black line). Note that $\beta = [1, 1, 1]$ performs worse as K increases (Figures 5.3(d), 5.4(d); note that when $K = 6, \alpha = \beta$). We also note that clustering on $\mathbf{T}$ encoded with different β outperforms that obtained by $\mathbf{A}$. In particular, the complex βs uniformly outperform others.

5.5 Summary

In this chapter we considered the problem of crowdsourced custering via triangle queries. We proposed an encoding scheme to embed the answers to triples in a tensor and provided sufficient conditions for it to give a true tensor of rank equal to the number of clusters. We also showed, through extensive simulations, that using tensor decomposition for clustering significantly improves the clustering obtained

via the adjacency matrix. A useful future direction would be to improve the tensor clustering algorithms that exploit the sparse structure of the noise.

Chapter 6

CROWDSOURCED CLUSTERING: ACTIVE QUERYING

In this chapter, we consider active querying setting for the problem of clustering n items into K disjoint clusters using noisy answers obtained from crowdsourced workers to pairwise queries of the type: "Are items i and j from the same cluster?". We propose a novel active querying algorithm for crowdsourced clustering which is simple and computationally efficient. We prove that our proposed algorithm succeeds in recovering the clusters when the crowd workers provide answers with error probability less than $1/2$. We provide both upper and lower bounds on the number of queries made by our algorithm. While the bounds depend on the error probability, the algorithm does not require this knowledge. In addition to theoretical guarantees, we also provide extensive numerical simulations as well as experiments on real datasets, using both synthetic and real crowd workers, to provide insights in to the behavior of the number of queries made by the algorithm. Based on both the theoretical results and the empirical observations, we conclude that, while the queries made by active clustering algorithms are orderwise better than random querying strategies, the advantage applies only when the datasets are very large or when the cluster sizes are very small. We believe that our observations could inform the design of practical crowdsourced clustering systems.

6.1 Introduction

Crowdsourcing – as the name suggests refers to using a crowd of potentially non-expert humans to solve problems that are difficult to solve by machines. Crowdsourcing has become one of the most popular ways of collecting datasets for supervised learning tasks [SF08; Ray+10]. There is an abundant amount of data, for example billions of images and texts that can be readily scraped from the internet. However, most of these datasets are unlabeled and it is unclear what structures might exist in them. Crowdsourcing can be a very useful resource to explore structure in data [Wel+10].

We consider the problem of crowdsourced clustering – the problem of finding clusters in a dataset with unlabeled items by querying pairs of items for similarity: "Are items i and j from same cluster?" A simple querying strategy is to randomly query pairs of items and then apply a graph clustering algorithm (Chapter 2) on the

data, viewing the unlabeled items as vertices and the answers to pairwise queries as edges. A natural question that arises is what is an active querying strategy? In this chapter we address this question and propose a novel algorithm (Algorithm 2) for active querying for crowdsourced clustering. When the crowd workers are better than random guessers, that is, the error probability is less than $\frac{1}{2}$, the problem of assigning an item to a cluster can be recast as a problem of inferring if the true parameter of a Bernoulli random variable is above or below $\frac{1}{2}$. Our algorithm is inspired by the finite law of the iterated logarithm (LIL) for multi-armed bandits [Jam+14]. We use confidence intervals that monotonically decrease with repeated queries and an assignment is decided once either the lower confidence interval is either above $\frac{1}{2}$ or the upper confidence is below $\frac{1}{2}$.

Our Contributions: We consider the problem of crowdsourced clustering and propose a novel active querying algorithm (Algorithm 2). The proposed algorithm does not require any knowledge of problem parameters. It is computationally efficient, simple to implement and can recover clusters regardless of their sizes. We also provide analysis of the proposed algorithm and show that it is guaranteed to succeed in recovering all the clusters with high probability (with error probability decaying as $1/\text{poly}(n)$). Our analysis also provides upper and lower bounds on the total number of queries made by the algorithm. In addition to theoretical guarantees, we also provide extensive numerical simulations as well as experiments on real datasets, using both synthetic and real crowd workers, to provide insights in to the behavior of the number of queries made by the algorithm. Based on both the theoretical results and the empirical observations, we conclude that, while the queries made by active clustering algorithms are orderwise better than random querying strategies, the advantage applies only when the datasets are very large or when the cluster sizes are very small.

Related Literature

In this section we briefly describe some related works.

The papers [Gom+11; VOH14a; VH16a] consider the problem of crowdsourced clustering using pairwise similarity queries. Prior work employs passive querying with either a deterministic pattern of queries fixed a priori [Gom+11] or randomly chosen queries [VOH14a; VH16a] (Chapters 2, 4).

Another related line of work is entity resolution in databases where the goal is to find data records that represent the same real world entities. There is a rich line of

work in this area (see [Wan+12; VBD14; VG15] and the references there in) which use heuristics-based crowdsourcing algorithms to resolve the entities. Most of these works assume that there is a machine generated similarity matrix between different data records and use this information to decide which data records to query. Recent work [MS17] provides analysis for some of the popular heuristics used when side information is present.

Recent work [MS16] on crowdsourced clustering considers a setting similar to ours where the probability of error made by crowd workers is assumed to be less than $\frac{1}{2}$. A key difference in their setting is that they forbid repeated querying. They provide upper and lower bounds on the number of queries in this setting. However, the algorithm proposed in [MS16] that can achieve a near optimal query complexity is computationally hard. Hence [MS16] propose an alternative computationally efficient algorithm is provided, however, its query complexity is not optimal, in particular, it grows quadratically in the number of clusters K. Furthermore, the algorithm requires knowledge of the error probability. In contrast, we allow repeated querying of the same pair of items and our algorithm is simple and computationally efficient and achieves near optimal (up to logarithmic factors) query complexity. The goal of repeated querying in our setting is not to drive the empirical error to 0 (which might not always be possible) but instead to guarantee that either the lower confidence bound on the true parameter is above or the upper confidence is below $\frac{1}{2}$.

6.2 Problem Setup

In this section we describe the problem setup, notations and model. Consider n items that belong to K disjoint clusters. Consider a set of crowd workers who can provide noisy answers to pairwise queries of the type: "Are items i and j from the same cluster?". Let Query(i, j) denote such a pairwise query. Let $X_{ij}(s)$ denote the answer provided by crowd worker s to Query(i, j). In particular, $X_{ij}(s) = 1$ if the answer to Query(i, j) by worker s is "yes" and $X_{ij}(s) = 0$ if the answer is "no". Suppose the workers were perfect, then with $\Theta(nK)$ [MS16] queries we can assign all the items to the correct clusters. However, the workers on the crowdsourcing platforms are not experts and hence can make errors.

Stochastic Block Model: If two items i and j are from the same cluster then the answer to Query(i, j) is 1 with probability p and 0 with probability $1 - p$. If the two items i and j are not in the same cluster then the answer to Query(i, j) is 1

with probability q and 0 with probability $1-q$. So, when i and j are from the same cluster, for all workers s,

$$X_{ij}(s) = \begin{cases} 1 & \text{with probability } p \\ 0 & \text{with probability } 1-p \end{cases},$$

and when i and j are not from the same cluster, for all workers s,

$$X_{ij}(s) = \begin{cases} 1 & \text{with probability } q \\ 0 & \text{with probability } 1-q \end{cases}.$$

We note that this is same as Stochastic Block Model (SBM) [HLL83; CK01] used in analyzing graph clustering (Chapter 2).

Assumptions: We assume that the answers given by different workers are independent. We also assume that, while the workers make errors they are better than random guessers. More formally,

Assumption 1*:* $X_{ij}(s)$ and $X_{ij}(s')$ are independent when $s \neq s'$.

Assumption 2*:* $p > \frac{1}{2} > q$.

For any pairs of items i and j, and any positive integer m, we use $\overline{X}_{ij}(m)$ to denote the average of m independent answers to the Query(i, j), that is,

$$\overline{X}_{ij}(m) := \frac{1}{m}\sum_{s=1}^{m} X_{ij}(s). \tag{6.2.1}$$

For any item j, we use the notation cluster(j) to denote the cluster to which item j belongs.

Analysis of the Stochastic Block Model (SBM) helps us understand the fundamental bottlenecks and also obtain cleaner expressions in the results which are more intuitive. However, in real settings SBM can be too limiting. The following is a general model for our problem setting:

Generalized Model: We define *confusion matrix* $P \in [0,1]^{n\times n}$ associated with the n items being clustered where P_{ij} is the probability the answer to Query(i,j) is 1. So, for a pair of items i and j, for all workers s,

$$X_{ij}(s) = \begin{cases} 1 & \text{with probability } P_{ij} \\ 0 & \text{with probability } 1-P_{ij} \end{cases}.$$

Since we assume that the workers are better than random guessers, we have the following,

$$P_{ij} \begin{cases} > \frac{1}{2} & \text{if the pair of items } i \text{ and } j \text{ are from the same cluster,} \\ < \frac{1}{2} & \text{otherwise.} \end{cases}$$

While we present the results and analysis for SBM, we also present the extensions of our results and analysis for the general case.

6.3 Active Query Algorithms and Performance Guarantees

We present the main results of our work in this section. We begin by describing a simple active querying algorithm for the case when we know p and q. Later we show how to use the ideas from this simple algorithm to extend to the setting where we do not know p and q.

6.3.1 When we know p and q

Suppose we knew p and q, then the problem of deciding whether an item i belongs to cluster(j) or not boils down to testing whether $X_{ij} \sim \text{Bern}(p)$ or $X_{ij} \sim \text{Bern}(q)$. The following is an algorithm for such a test:

Algorithm 1 (Active querying with the knowledge of p and q)**.** *Let* $\Delta := \frac{1}{2}(p-q)$. *Set* $M := \left\lceil \frac{3 \log n}{2\Delta^2} \right\rceil$. *Let* V *be the set of items to be clustered.*

1. *Initialization: Start with a clustering* $\mathbf{C} = \{\mathbf{C}_1\}$ *with just one cluster with one element, say* i*, randomly chosen from* V*. So,* $\mathbf{C}_1 = \{i\}$*. Set* $V \leftarrow V - \{i\}$.

2. *Until* $V \neq \phi$*, pick a random* $v \in V$ *to for querying. Starting with* $k = 1$*, for* $\mathbf{C}_k \in \mathbf{C}$*, pick an item* $u \in \mathbf{C}_k$ *uniformly at random and ask Query(*v*,* u*) to* M *crowdworkers. One of the following scenarios holds:*

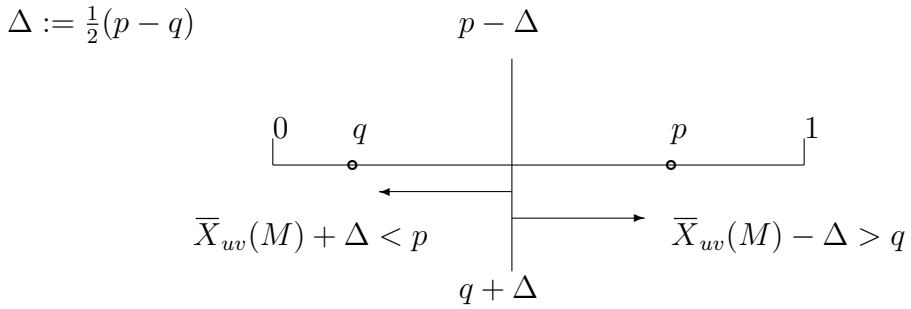
   a) *Lower confidence bound on the empirical average of the answers is above* q*: If* $\overline{X}_{uv}(M) - \Delta > q$*, then assign* v *to* $\mathbf{C}_k$*:* $\mathbf{C}_k \leftarrow \mathbf{C}_k \cup \{v\}$ *and* $V \leftarrow V - \{v\}$.

   b) *Upper confidence bound on the empirical average of the answer is below* p*: If* $\overline{X}_{uv}(M) + \Delta < p$*, then* $v \notin \mathbf{C}_k$*. Move to next cluster:* $k \to k+1$.

   c) *If* $\overline{X}_{uv}(M) - \Delta < q$ *and* $\overline{X}_{uv}(M) + \Delta > p$*, then declare failure to classify* v *and pick the next item.*

If v does not get assigned to any of the existing clusters (and the algorithm has not failed to classify v), then start a new cluster with v: $\mathbf{C} = \mathbf{C} \cup \{v\}$ and update $V \leftarrow V - \{v\}$.

Following is an illustration of the test used in the algorithm described above:



Pseudocode for Algorithm 1 is provided in the appendix (Appendix C.3).

Upper and lower bounds on the number of queries made by Algorithm 1 is given by the following two propositions.

Proposition 1. *Algorithm 1 successfully recovers all the clusters with at most $\mathcal{O}\left(\frac{nK \log n}{(p-q)^2}\right)$ queries with probability at least $1 - \frac{1}{n}$.*

Proposition 2. *$\Omega\left(\frac{nK}{(p-q)^2}\right)$ queries are necessary to guarantee success of Algorithm 1 with probability at least $\frac{3}{4}$.*

Thus, the two propositions in tandem show that the upper bound on the number of queries made by Algorithm 1 is not improvable except up to log factors. In fact, the $\log n$ term is also necessary if we want the error probability to decay as $\frac{1}{\text{poly}(n)}$. Detailed proofs for these propositions are available in Appendix C.1.

6.3.2 Without the knowledge of p and q

Algorithm 1 relied on the fact that for $M = \left\lceil \frac{3 \log n}{2(p-q)^2} \right\rceil$ queries, with high probability, the confidence intervals are such that either the lower confidence bound holds, in which case we assign the item being queried for to the cluster being compared to, or the upper confidence bound holds, in which case we decide otherwise. However, note that setting the value of M as well as the tests for upper and lower confidence bounds required the knowledge of p and q. In most practical scenarios we do not have this knowledge. Therefore, we propose the following algorithm, which uses time varying confidence bounds $\psi(t)$ which are monotonically decreasing in time t.

At each time t, we make a new query for an item i with cluster(j) and we update the confidence interval around the true parameter to $\left[\overline{X}_{ij}(t) - \psi(t), \overline{X}_{ij}(t) + \psi(t)\right]$. We stop repeating Query(i, j) when either the lower confidence is above $\frac{1}{2}$ or the upper confidence is below $\frac{1}{2}$.

Algorithm 2 (Active querying without the knowledge of p and q)**.** *Let V be the set of items to be clustered. Let ζ and δ are input parameters (chosen depending on the probability of error tolerated).*

1. *Initialization: Start with a clustering $\mathbf{C} = \{\mathbf{C}_1\}$ with just one cluster with one element, say i, randomly chosen from V. So, $\mathbf{C}_1 = \{i\}$. Set $V \leftarrow V - \{i\}$.*

2. *Until $V \neq \phi$, pick a random $v \in V$ to for querying. Starting with $k = 1$, for $\mathbf{C}_k \in \mathbf{C}$, pick an item $u \in \mathbf{C}_k$ uniformly at random. Start by setting the cumulative empirical average $\overline{X}_{vu}(0) = 0$. For each time step t, ask Query(v, u) to a distinct crowdworker. Let $X_{vu}(t)$ denote the answer obtained. Update the cumulative empirical average: $\overline{X}_{vu}(t) = \frac{t-1}{t}\overline{X}_{vu}(t-1) + \frac{1}{t}X_{vu}(t)$ and the new confidence interval to:*

$$\psi(t) = (1 + \sqrt{\zeta})\sqrt{\frac{1+\zeta}{2t}\log\left(\frac{\log\left((1+\zeta)t\right)}{\delta}\right)}\,.$$

   Repeat Query(v, u) and updating $\overline{X}_{vu}(t)$ and $\psi(t)$ until one of the following scenarios holds:

   a) *Lower confidence bound on the empirical average of the answers is above $\frac{1}{2}$: If $\overline{X}_{uv}(t) - \psi(t) > \frac{1}{2}$, then assign v to $\mathbf{C}_k$: $\mathbf{C}_k \leftarrow \mathbf{C}_k \cup \{v\}$ and $V \leftarrow V - \{v\}$.*

   b) *Upper confidence bound on the empirical average of the answers is below $\frac{1}{2}$: If $\overline{X}_{uv}(t) + \psi(t) < \frac{1}{2}$, then $v \notin \mathbf{C}_k$. Move to next cluster: $k \to k+1$.*

   If v does not get assigned to any of the existing clusters, then start a new cluster with v: $\mathbf{C} = \mathbf{C} \cup \{v\}$ and update $V \leftarrow V - \{v\}$.

Pseudocode for Algorithm 2 is provided in the appendix (Appendix C.3).

Using the finite law of the iterated logarithm (LIL) bound from [Jam+14] (Theorem C.2.1) we can guarantee the following under the assumptions on our model. For this theorem $\Delta := \frac{1}{2}\min\left\{p - \frac{1}{2}, \frac{1}{2} - q\right\}$.

Theorem 9. *Algorithm 2 succeeds in recovering all the clusters exactly with at most* $\mathcal{O}\left(\frac{nK}{\Delta^2}\log\left(n\log\frac{1}{\Delta}\right)\right)$ *queries with high probability.*

More generally, we can state the following.

Corollary 1. *For any* $\zeta \in (0,1)$, $c \geq 3$, $\frac{\delta}{n^c} \in (0, \log(1+\zeta)/e)$, *with probability at least* $1 - 2\frac{2+\zeta}{\zeta}n^2\left(\frac{\delta}{n^c\log(1+\zeta)}\right)^{1+\zeta}$, *Algorithm 2 succeeds in recovering all the clusters exactly and the total number of queries made is upper bounded by* $\mathcal{O}\left(\frac{nK}{\Delta^2}\log\left(\frac{n^c}{b_3\delta}\log\frac{b_2}{\Delta}\right)\right)$, *where* $b_2 = (1+\zeta)^2$, $b_3 = \frac{1}{(2(1+\sqrt{\zeta}))^3}$.

Note that c, δ and ζ can be chosen such that the error probability decays as $1/\text{poly}(n)$. We further note that the choice of δ and ζ also affects the size of confidence interval $\psi(t)$ and hence the number of queries made by Algorithm 2. Theorem 9 is obtained by choosing $c = 4$ and $\zeta = 0.1151$. Detailed proof is available in the Appendix C.2.

6.3.3 Discussion

In this section we comment on Algorithm 2.

Comparison Point

In Algorithm 2, $\frac{1}{2}$ is used as the comparison point for the upper and lower confidence bound tests. This is because of the assumption that the workers are better than random guessers. Suppose, we know $p \geq \eta \geq q$ for some $\eta \in (0,1)$, then we could use η as a point for comparison instead of $\frac{1}{2}$ and the algorithms and proofs can be modified accordingly. Note that the definition of Δ would get modified to $\Delta := \frac{1}{2}\min\{p-\eta, \eta-q\}$. For example, suppose for a dataset we have $p = 0.51$ and $q = 0.01$. Then in this case, the workers are extremely good at distinguishing items of different clusters as different, but are not as good in identifying items from the same cluster as same cluster elements. So, a more reasonable η to use would be 0.25.

Generalization to Confusion Matrix

We define *confusion matrix* $P \in [0,1]^{n\times n}$ associated with the n items being clustered where P_{ij} is the probability the answer to Query(i,j) is 1. Due to the assumption that the workers are better than random guessers, $P_{ij} > \frac{1}{2}$ when i and j are from the same cluster and $P_{ij} < \frac{1}{2}$ when i and j are not not from the same cluster. For every pair of items i and j, define $\Delta_{ij} := |P_{ij} - \frac{1}{2}|$. We can modify the proof of Theorem 9 to obtain the following:

Corollary 2. *An upper bound on the total number of queries made by Algorithm 2 in the general case is,*

$$\sum_{i,j:\{i,j\}\in\Omega} \frac{b_1}{\Delta_{ij}^2} \log\left(\frac{n^c}{b_3\delta} \log \frac{b_2}{\Delta_{ij}}\right),$$

where Ω is the set of queries made and $|\Omega| \leq nK$.

Modifications in Querying

In Algorithm 2, in order to decided if an item i belongs to a cluster, we randomly chose an item j from the cluster being considered and Query(i, j) repeatedly. Instead, in order to decide if item i belongs to cluster(j), we can pick a random element from cluster(j) each time we query. For the Stochastic Block Model, there is no statistical change and hence the guarantee provided by Theorem 9 still holds. For the general confusion matrix case described above (Section 6.3.3), careful book keeping is needed, as instead of $\mathbb{E}(\overline{X}_{ij}(t)) = P_{ij}$, we have $\mathbb{E}(\overline{X}_{ij}(t)) = \frac{1}{t}\sum_{s=1}^{t} P_{ij_s}$, where j_s is the element picked at time step s from cluster(j).

Comparison of Algorithm 2 with Random Querying

In this section we compare active querying algorithm for crowdsourced clustering (Algorithm 2) to using random queries followed by clustering. The state of the art algorithm for crowdsourced clustering using random queries is a two step process. A random subset of the $\binom{n}{2}$ pairs, say $\lceil r\binom{n}{2}\rceil$ with $r \in (0, 1]$ are queried first and then a graph clustering algorithm is run on it. We focus on computationally efficient (polynomial time) algorithms for comparison.

- **Algorithm 2 succeeds regardless of cluster sizes:** Computationally efficient graph clustering algorithms, for example, spectral clustering [McS01; R+11], convex clustering algorithms [Che+14; VOH14a; Jal+15], have a bottleneck in terms of the size of the smallest cluster that can be recovered. In particular, the smallest cluster has to be sufficiently large, at least $\Omega(\sqrt{n})$, to be recovered exactly. This bottleneck of $\Omega(\sqrt{n})$ on the minimum cluster size is conjectured to also be necessary for any polynomial time graph clustering algorithm (related to the *hidden clique conjecture*). Therefore using any known computationally efficient graph clustering algorithms after querying random

pairs of items can only recover clusters of size up to $\Omega(\sqrt{n})$. On the contrary, the sufficient condition for exact recovery of the clusters (Theorem 9) using the *active querying* algorithm (Algorithm 2), which is computationally efficient, holds *regardless of the cluster sizes*.

- **Algorithm 2 is free of model parameters:** For random querying, the knowledge of $p-q$ and $n_{\min}$ is required to a priori set the number of random queries to be made ($r < 1$ to make $\lceil r\binom{n}{2} \rceil$ queries), if we want to guarantee the exact recovery of clusters, unless we are making all $\binom{n}{2}$ queries. On the other hand, the active querying algorithm (Algorithm 2) does not require the knowledge of p, q, K or the cluster sizes ahead of time to guarantee exact recovery. The only assumption that it makes is that the workers are better than random guessers ($p > \frac{1}{2} > q$).

- **Algorithm 2 vs. random querying: Who wins?** The sufficient number of queries required to guarantee exact recovery of clusters for active query algorithm 2 is $\mathcal{O}\left(\frac{nK}{\Delta^2}\log\left(n\log\frac{1}{\Delta}\right)\right)$. Let us compare this the state of the art sufficient conditions for exact recovery of clusters for SBM (see [Che+14; VOH14a; Jal+15] & references there in). When the smallest cluster is $\Theta(\sqrt{n})$, we can guarantee exact recovery using at most $\mathcal{O}\left(n^2/(p-q)^2\right)$ random queries. When the smallest cluster is large, i.e, when the cluster sizes are $\Theta(n)$, we can obtain correct clustering by using at most $\mathcal{O}\left(n(\log n)^2/(p-q)^2\right)$ random queries. While orderwise, in both the cases, the upper bound for active is better, the advantage does not kick in until the dataset is sufficiently large. This is illustrated in Figure 6.1 and Figure 6.2. Both the figures are generated for parameters $p = 0.8$ and $q = 0.2$. In Figure 6.1, there are 4 clusters of equal size, so that it represents the scenario where the clusters are of size $\Theta(n)$ and in Figure 6.2, there are $\sqrt{n}/10$ clusters of equal sizes so that it represents the scenario with clusters of size $\Theta(\sqrt{n})$. Note that the known computationally efficient graph clustering algorithms have a lower bound of $\Omega(\sqrt{n})$ on the cluster sizes (see the discussion 4 in Chapter 2).

In summary, while active querying has advantages in terms of being free of model parameters and that its success does not depend on a cluster size bottleneck, the number of queries needed might be large for small datasets. Even though active querying algorithm is competitive or better than random queries orderwise, the advantages kick in only when a large number of items are being clustered or when the

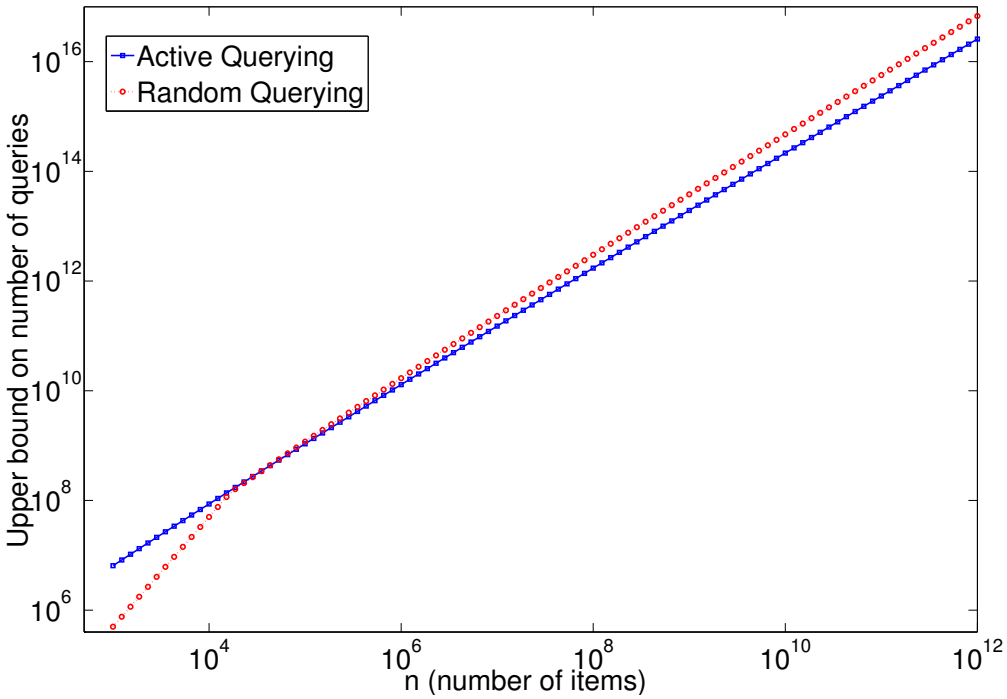


Figure 6.1: Comparison of the upper bounds on the total number of queries required for exact clustering by active and random querying for large clusters (size $\Theta(n)$)

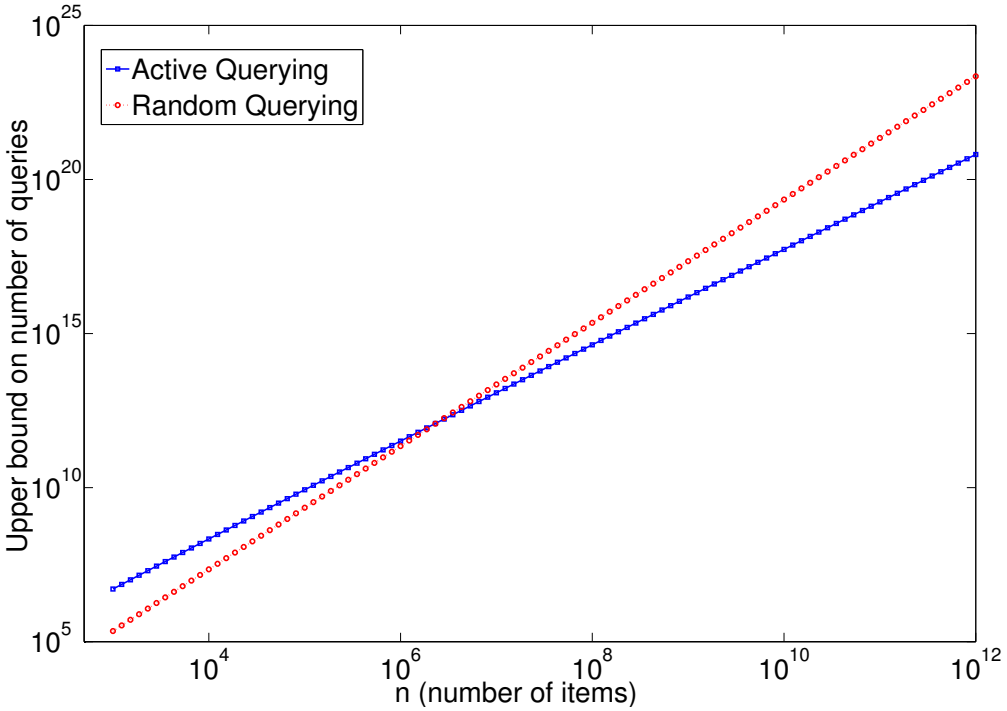


Figure 6.2: Comparison of the upper bounds on the total number of queries required for exact clustering by active and random querying for small clusters (size $\Theta(\sqrt{n})$)

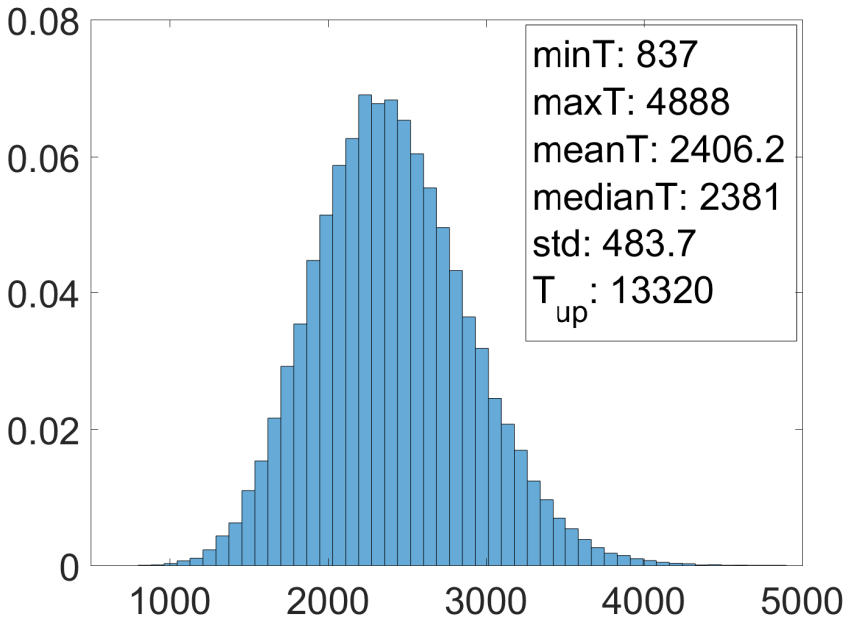


Figure 6.3: Empirical pdf of T_{ij}
Empirical pdf of the number of times each query is repeated by Algorithm 2

cluster sizes are too small. As such, one might be better off using random queries for small datasets whereas active clustering is more advantageous at large scales.

6.4 Simulations

In this section we investigate the empirical performance Algorithm 2 with varying parameters of the model. The results presented throughout this section are averaged over 10 trials and we set $c = 7, \zeta = 0.001$, $\delta = 1/n^c$, unless specified otherwise. Note that the value of ζ, c, and n determine the error probability. Recall that the favorable event $\mathcal{E}_\psi$ occurs with probability at least

$$1 - n^2 \times 2^{\frac{2+\zeta}{\zeta}} \left(\frac{1}{n^c \log(1+\zeta)} \right)^{1+\zeta} .$$

Depending on the error probability that we are willing to tolerate, we get to choose the parameters ζ and c. Furthermore, recall that the ζ δ and c also influence the confidence interval window.

$$\psi(t) = (1+\sqrt{\zeta}) \sqrt{\frac{1+\zeta}{2t} \log\left(n^c \log((1+\zeta)t)\right)} .$$

The size of confidence interval affects the number of queries made.

Let T be the number of times Query(i, j), is repeated in order to decide whether to assign an item i to a cluster(j) or not. Figure 6.3 shows the empirical pdf for T for

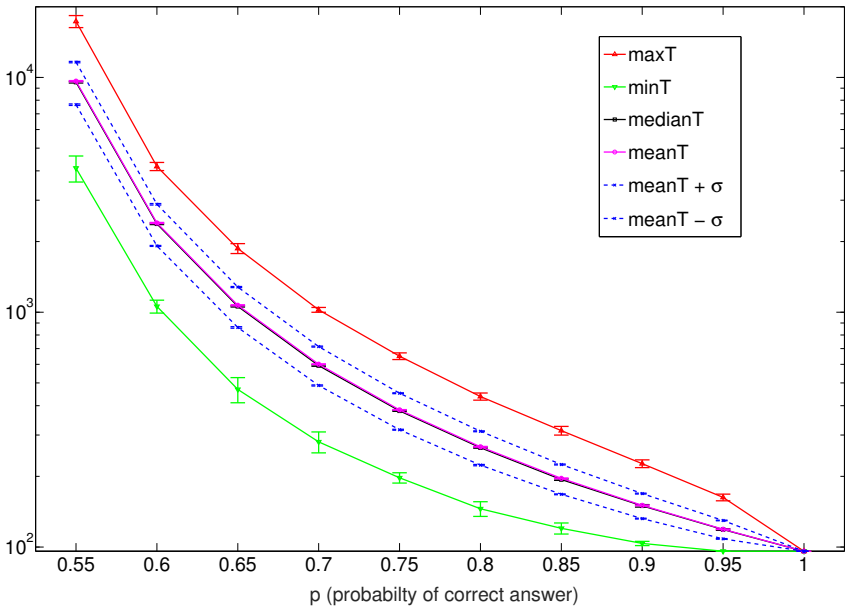


Figure 6.4: Minimum, maximum, median, mean (along with standard deviation) of the number of repeated queries as a function of varying error probability (p). Averaged over 10 trials).

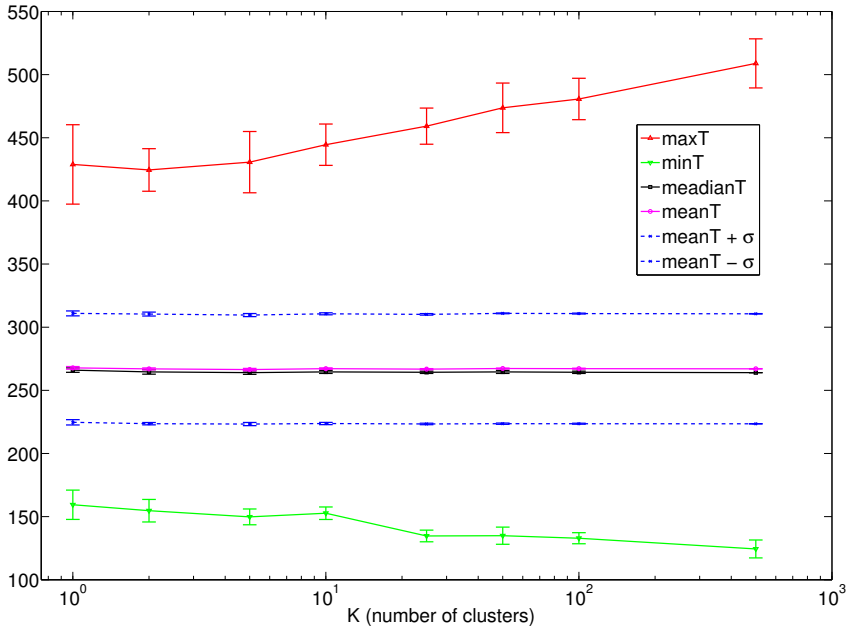


Figure 6.5: Minimum, maximum, median, mean (along with standard deviation) of the number of repeated queries as a function of varying number of clusters (K). Averaged over 10 trials.

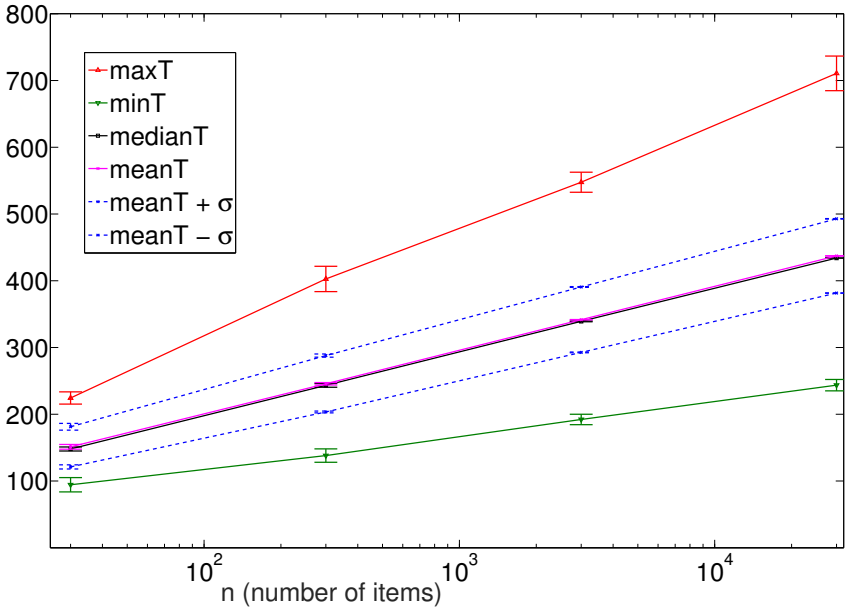


Figure 6.6: Minimum, maximum, median, mean (along with standard deviation) of the number of repeated queries as a function of varying number of items (n). Averaged over 10 trials.

an example with $n = 500$, $K = 500$, $p = 0.6 = 1 - q$. We note that the maximum T is below the upper bound (T_{up}) as expected. Furthermore, we note that most of the distribution is concentrated around the mean. In particular, for the example in Figure 6.3, 94.5% of the queries are within 2 standard deviations of the mean. So, the total number of queries is much less than the conservative bound of nKT_{up}.

Recall that from Theorem 9, $T_{ij} \leq \frac{b_1}{\Delta_{ij}^2} \log\left(\frac{n^c}{b_3\delta} \log \frac{b_2}{\Delta_{ij}}\right)$. We now investigate the behavior of the maximum, minimum, mean, and median of T as a function of the probability of correct answer p (equivalently $\Delta := p - \frac{1}{2}$), the number of clusters K, and the total number of items n.

1. Varying p: We set $n = 500, K = 5$ with equal cluster sizes and vary p from 0.55 to 1 in steps of 0.05. We set $q = 1 - p$, so Δ varies from 0.05 to 0.5. Figure 6.4 shows the behavior of T as a function of p. As we expect, T sharply decreases with increasing p (equivalently increasing Δ).

2. Varying K: We set $n = 500, p = 1 - q = 0.8$ and vary the number of clusters K with values in the set $\{1, 2, 5, 10, 25, 50, 100, 500\}$ and all the K clusters have equal sizes. Note that when $K = 1$ all items are in the same cluster and when $K = 500$ all items are distinct. The number of repeated queries required to either assign or not assign an item to a particular cluster

is independent of the total number of clusters (note that T_{up} is independent of K). Figure 6.5 shows the behavior of T as a function of K. We observe that the median (and mean) T remains the same as K varies. Furthermore, the total number of queries made increases linearly with K. So, the larger K gets, the higher the chance of hitting extreme events. So, we observe a slight increase in maximum T and a slight decrease in minimum T as K increases.

3. Varying n: We st $K = 3, p = 1 - q = 0.8$ and vary the number of items n with values in the set $\{30, 300, 3000, 30000\}$. Figure 6.6 shows the behavior of T as a function of n and we observe T grows logarithmically with n as expected (note that the x-axis is in log scale).

6.5 Experiments Using Real Data

In this section we present experimental results using real datasets and both synthetic and real crowdworkers.

6.5.1 Experiment with real crowdworkers

We use **Dogs3** [Kho+11; VH16a] dataset for this experiment. Dogs3 dataset has 473 dogs of 3 different breeds: Norfolk Terrier (172 images), Toy Poodle (151 images), and Bouvier des Flanders (150 images). We illustrate the behavior of cumulative averages of the responses to repeated querying on a real crowdsourcing platform (we used Amazon Mechanical Turk).

$$\overline{X}_{ij}(m) := \frac{1}{m} \sum_{s=1}^{m} X_{ij}(s) \ .$$

We chose 30 random images from Dogs3 dataset to be compared and another 30 images from the dataset to compare with. For each image i in the first set, we perform 50 repeated queries with the corresponding image j in the second set by querying 50 unqiue workers. So, same Query(i, j) is queried 50 times. We illustrate the behavior of the cumulative averages for 6 sample pairs in the first row of Figure 6.7. In order to illustrate the behavior with modified querying (Section 6.3.3), we also perform 50 queries with image i in the first set and choosing a random image from cluster(j) for corresponding j in the second set. So, for each item i being queried, instead of repeating Query(i, j), a random image from cluster$j$ is chosen each time. The cumulative averages for the same 6 samples is shown in the second row of Figure 6.7. While there is not much difference between two strategies in general, picking a random item each time might be more robust as it can get out of the rare

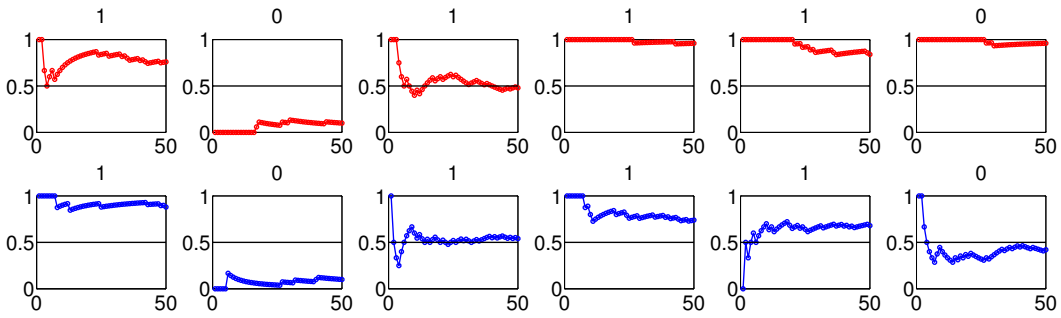


Figure 6.7: Cumulative averages of the answers given by crowdworkers for repeated queries for 6 sample items. The first row corresponds to the experiment where Query(i, j) is repeated with same j and the second row corresponds to the experiment where j is chosen randomly from cluster(j) each time. Note that the true answer is at the top of each plot.

situations when repeating the same query might get one stuck with a bad candidate from the cluster(j).

6.5.2 Experiments with simulated crowdworkers

In this section we use two datasets, **Scenes4** and **AllSports**, which have parameters for worker errors from a set of experiments using real crowdworkers. We use these parameters to simulate crowdworkers.

1. **Scenes4** is a subset of the Scenes dataset [Gom+11; FP05] and has 539 images of scenes from 4 categories: Indoors (308 images), suburbs (77 images), forest (77 images), and highways (77 images). This dataset has one very large and three moderately sized clusters. Using the data from [Gom+11], we compute a $n \times K$ confidence matrix, where the $ik-$th entry corresponds to image i and cluster k and is computed by taking the ratio of the number of "yes" answers out the queries made with items in cluster k. When a Query(v, u) is made, we synthetically generate human responses using the entry in the confusion matrix entry for item v and the true cluster corresponding to item u as the probability of "yes". We run Algorithm 2 with $c = 4, \zeta = 0.001$, which ensures the overall probability of error $\delta' \leq 0.055$. Table 6.1 shows the statistics of the number of repeated queries as well as overall queries and the percentage of node pairs in error in the clustering obtained.

2. **AllSports** is a dataset of 267 images of 86 athletes from 10 different sports [VG15]. The task is to identify images of the same athlete. So, there are 86 different athletes and cluster sizes vary from 1 to 5 (extremely small clusters). This

Dataset	Node pair error%	**median T**	min T	max T	Total Queries
Scenes4	$0.57\% \pm 0.02\%$	77.8 ± 3.50	57 ± 0	$2.29e5 \pm 1.07e5$	$6.09e5 \pm 1.64e5$
AllSports	$0.07\% \pm 0.03\%$	64 ± 0	63 ± 0	2000 ± 0	$8.22e5 \pm 1.74e4$

Table 6.1: Various statistics for the number of repeated queries, the total number of queries made and the percentage of node pairs in error after running Algorithm 2 on the real datasets.

dataset has a $n \times n$ confusion matrix and for a small subset of the pairs 10 repeated queries obtained by crowdworkers on Amazon Mechanical Turk are available. When Query(v, u) is made, if repeated queries are available for the pair $\{v, u\}$, then we uniformly randomly pick an answer out of the 10 repeated queries. Otherwise, we generate the human response using the confusion matrix. We run Algorithm 2 with $c = 4, \zeta = 0.001$ which ensures the overall probability of error $\delta' \leq 0.055$. Table 6.1 shows the statistics of the number of repeated queries as well as overall queries and the percentage of node pairs in error in the clustering obtained. Algorithm 2 is able to find the 86 clusters with very few errors.

Note that for both the Scenes4 and AllSports datasets, most of the queries made by Algorithm 2 are repeated very few times (~ 78 for the Scenes4 and ~ 64 for AllSports). However, there are a few *difficult* items in both datasets that take significantly more number of repetitions as the probability of error associated with then is close to 0.5. This suggests that it might be more practical to set an upper limit on the number of repeated queries above which the items can be added to a *difficult* pool. After the rest of the items are clustered, we can either use some more queries to see how close the items in the difficult pool might be to each of the existing clusters (and assign a low confidence with the association) or for the most difficult queries ask an expert.

We also queried all the pairs and ran graph clustering algorithms (we used k-means, spectral clustering, and convex clustering). For the Scenes4 dataset it recovered all 4 clusters whereas it failed for the AllSports dataset. Note that the Scenes4 dataset has reasonably large clusters whereas AllSports has extremely small clusters. The contrast between these two datasets demonstrates that for small datasets with reasonably large cluster sizes, it might be more beneficial to do random queries, whereas when the cluster sizes are too small active queries can still recover the clusters (see the discussion in Section 6.3.3).

6.6 Summary

In this chapter, we considered the problem of crowdsourced clustering and proposed a novel active querying algorithm. The proposed algorithm is simple, computationally efficient, and does not need the knowledge of any parameters. We showed that our proposed algorithm succeeds in recovering the clusters when the crowd workers are better than random guessers. We provided a tight upper bound (up to log factors) on the number of queries made by our algorithm. While the bounds depend on the error probability, the algorithm does not require this knowledge. A potential future direction would be to characterize the error rate in clustering given a fixed budget.

Chapter 7

CONCLUSIONS AND FUTURE WORK

Clustering is one of the most widely used tools for exploratory data analysis. In this thesis, we focused on two problems related to clustering:

- The first problem is that of *graph clustering* where we analyzed convex algorithms for clustering graphs with the goal of understanding the fundamental structural bottlenecks and trade-offs of clustering. We provided explicit conditions without large unknown constants and polylog factors, on the properties of graphs that determine the efficacy of the convex clusteirng algorithms (Chapters 2, 3).

- The second problem is that of *crowdsourced clustering* where the key question addressed is: How do we design queries to obtain better quality data from a crowd of non-expert workers? We compared two type of random queries and demonstrated the superiority of comparing three items at a time over two at a time. Further, we proposed and analyzed a novel active querying algorithm that works without the knowledge of any problem parameters and provided an upper bound on the number of queries that is sufficient to guarantee exact recovery of the clusters (Chapters 4, 5, 6).

Apart from deriving theoretical guarantees that characterize various trade-offs in the above problems, we also applied our algorithms on real datasets to demonstrate the veracity of our assumptions.

7.1 Future Directions

We conclude this thesis with discussion on some of the challenges and potential future directions.

Non-exact Recovery: In this thesis we focused on *exact recovery* of the clusters. In particular, the sufficient conditions for successfully recovering the clusters using convex programs derived in Chapters 2 and 3 are for exact recovery of the underlying low-rank matrices which encode the underlying cluster structure. Perfectly recovering these low-rank matrices would give the exact clustering. However, if we

do not recover these exactly, it does not necessarily mean that the clustering fails. Furthermore, in many practical scenarios we might be able to tolerate a few errors in clustering. Thus it would be of interest to explore what happens when the convex programs fail to recover the low-rank matrix exactly. In particular, when the exact recovery guarantees for the convex programs fail, it is worthwhile to examine the following questions: how far away will the optimal solution to the convex program be from the true underlying cluster structure? Would the optimal solution completely lose all information about the cluster structure suddenly or is it gradual? Can we characterize this trade-off as a function of problem parameters and provide guarantees on how much of the cluster structure can still be recovered?

Overlapping Clusters: The clustering scenarios considered in this thesis have disjoint clusters. In some applications, it might be more ideal to consider overlapping clusters, where nodes in a graph (or items in a dataset) could belong to more than one cluster. One way to model overlapping clusters is to consider a *soft membership* or *probabilistic membership* model where each node in the graph has a vector of length K (where K is the number of clusters) associated with it and each entry j of this membership vector reflects how much the node belongs to cluster j. One such generative model is that of *mixed membership model* which uses Dirichlet process to generate the membership vectors. Tensor decomposition based clustering algorithm is proposed in [Ana+13] and upper bound on the estimation errors are provided. However the algorithm in [Ana+13] needs the knowledge of the parameters of the generative model to set the thresholds for clustering. Furthermore, it can only recover balanced clusters and is suboptimal when the clusters are unbalanced or when there are outliers. For example, for the planted clique problem where a clique is planted in a random graph, it can only recover the clique if the size is at least $\Omega(n^{2/3})$, whereas convex algorithms discussed in Chapter 2 can recover the cliques of size $\Omega(\sqrt{n})$. For the clustering problems with disjoint clusters, convex programs leverage the fact that the true underlying cluster structure has an adjacency matrix whose rank is equal to the number of clusters. Note that when we allow overlapping clusters in a graph, the true adjacency matrix does not correspond to a union of disjoint clique, and hence its rank is not equal to the number of clusters anymore. It is of interest to see what kind of convex objectives and constraints can encourage recovering cluster structures that are overlapping.

Scalable Algorithms for Semidefinite Programs: In Chapters 2 and 3, we considered convex programs for graph clustering and similarity clustering. The semidef-

inite programs (SDPs) we used for clustering (Programs 2.1.4, 2.1.7, 3.1.1) are robust to noise, missing data and outliers which are of practical importance. While convex algorithms are computationally efficient, in the sense that, they are solvable in polynomial time, their complexity can still be prohibitive in the face of large datasets. Most implementations of convex programs for low-rank matrix completion and recovery take $\mathcal{O}(n^3)$ per iteration. For datasets that exceed a few thousand of nodes, such complexity prohibits the use of SDPs. Therefore, scalable solutions for the SDPs for low-rank matrix recovery and completion can be tremendously useful for practical applications. There are several works recently [PW15; Yur+17] that scale SDPs using sketching, randomized algorithms, sparsification. There are several challenges that still need to be addressed, like the ability to handle non-smooth objective functions like l_1 norm and nuclear norm which are heavily used in the SDPs for clustering.

Data-driven Guarantees for Clustering Algorithms and Evaluation Metrics: In this thesis, we used *block models* to understand the fundamental bottlenecks and performance guarantees for the convex optimization based algorithms for clustering. Most real world datasets are not created via generative models and hence it is important to build on our understanding of the bottlenecks and the performance guarantees for the convex clustering algorithms to the real world data. A pertinent question is to understand how fragile the algorithms and their analysis are to model mismatches. In particular, it is of interest to quantify model mismatches and how they affect the performance guarantees of the convex algorithms. Furthermore, it is of practical interest to have statistical quantities that we can compute using the data and be able to provide certain level of confidence in the solution output by clustering algorithms. While there are some works [Ben15; WM16] that look at this objective of stability and data-driven approaches, we are still far from the goal. Another relevant problem for clustering real world datasets is that of evaluation metrics. Unlike supervised learning, in unsupervised learning methods like clustering there is no labeled test data available that can be used for cross-validation or comparison of various algorithms. It is of practical importance to develop metrics for clustering that can be used for systematic evaluation and comparison of performance of various clustering algorithms.

Optimal Similarity Comparison for Crowdsourced Clustering: In Chapter 4, we showed that comparing three items at a time is more efficient, that is, for a fixed budget it provides more data and less noisy data. A natural extension to this work

is to inquire what is the optimal number of items to compare at a time? Humans are have limitations in terms of on average how many objects or concepts they can comprehend. Studies on how many concepts can human short term memory comprehend [Cow00; Mil56] suggests that comparing more than five to seven items it might become counter productive. There are several works in crowdsourcing literature that have used panels with large number of objects to be compared. However, there still needs to be a systematic study that can compare the trade-offs in terms of quality and quantity of the data obtained in this way.

Limited Budget for Crowdsourcing: In most crowdsourcing systems, we pay the crowd workers to answer the queries, and hence working within a budget constraint is unavoidable. In this thesis, our analysis for crowdsourced clustering provides the number of queries that are sufficient to guarantee the exact recovery of the clusters. Another viewpoint that would be of practical use is to explore the following question: *given a budget, what is the best achievable error rate of clustering using crowdsourcing?*

Machine Learning Systems with Humans-in-the-loop: In application domains such as astronomy and biology (in particular, biological imaging), an enormous amount of data is collected and often at a rate faster than it is being processed. In astronomy, a single sky survey can produce tens of terabytes of data [LSS] and typically there are several such surveys that happen across a wide range of wavelengths. Such a massive scale of data poses problems for storage as well as processing. Mitigating these problems require building real-time systems that decide in an online fashion whether to keep or throw away newly arrived data. For preprocessing the data, it is useful to design a human-in-the-loop system that uses machine learning and signal processing tools (denoising, sampling, clustering, classification, etc.) as well as takes input from both domain experts and non-expert crowd workers when required and store the data in an efficient manner. Such a system can significantly reduce the time required by astronomers and biologists to further process the data. A challenge in these problems will be to balance the trade-offs between computational cost, real-time constraints, budget constraints, and delays associated with human input and accuracy as well as guarantee robustness to noise.

BIBLIOGRAPHY

[Ame13] Brendan PW Ames. "Robust convex relaxation for the planted clique and densest k-subgraph problems". In: *arXiv preprint arXiv:1305.4891* (2013).

[Ame14] Brendan PW Ames. "Guaranteed clustering and biclustering via semidefinite programming". In: *Mathematical Programming* 147.1-2 (2014), pp. 429–465.

[Ana+13] Animashree Anandkumar et al. "A tensor spectral approach to learning mixed membership community models". In: *Conference on Learning Theory*. 2013, pp. 867–881.

[AR13] Charu C Aggarwal and Chandan K Reddy. *Data clustering: algorithms and applications*. CRC press, 2013.

[AS06] Tero Aittokallio and Benno Schwikowski. "Graph-based methods for analysing networks in cell biology". In: *Briefings in bioinformatics* 7.3 (2006), pp. 243–255.

[AV11] Brendan PW Ames and Stephen A Vavasis. "Nuclear norm minimization for the planted clique and biclique problems". In: *Mathematical programming* 129.1 (2011), pp. 69–89.

[AV14] Brendan PW Ames and Stephen A Vavasis. "Convex optimization for the planted k-disjoint-clique problem". In: *Mathematical Programming* 143.1-2 (2014), pp. 299–337.

[B+15] B. W. Bader, T. G. Kolda, et al. *MATLAB Tensor Toolbox Version 2.6*. Available online. Feb. 2015. URL: http://www.sandia.gov/~tgkolda/TensorToolbox/.

[B+87] George EP Box, Norman Richard Draper, et al. *Empirical model-building and response surfaces*. Vol. 424. Wiley New York, 1987.

[Bal+11] Sivaraman Balakrishnan et al. "Noise thresholds for spectral clustering". In: *Advances in Neural Information Processing Systems*. 2011, pp. 954–962.

[BBC04] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. "Correlation clustering". In: *Machine Learning* 56.1-3 (2004), pp. 89–113.

[BBS85] John D Barrow, Suketu P Bhavsar, and DH Sonoda. "Minimal spanning trees, filaments and galaxy clustering". In: *Monthly Notices of the Royal Astronomical Society* 216.1 (1985), pp. 17–35.

[Beh+11] Tara S Behrend et al. "The viability of crowdsourcing for survey research". In: *Behavior research methods* 43.3 (2011), p. 800.

[Ben15] Shai Ben-David. “Clustering is easy when.... What?” In: *arXiv preprint arXiv:1510.05336* (2015).

[BK07] B. W. Bader and T. G. Kolda. “Efficient MATLAB computations with sparse and factored tensors”. In: *SIAM Journal on Scientific Computing* 30.1 (Dec. 2007), pp. 205–231.

[BKG11] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. “Amazon’s Mechanical Turk: A new source of inexpensive, yet high-quality, data?” In: *Perspectives on psychological science* 6.1 (2011), pp. 3–5.

[CAB14] Eric C Chi, Genevera I Allen, and Richard G Baraniuk. “Convex biclustering”. In: *arXiv preprint arXiv:1408.0856* (2014).

[Can+11] Emmanuel J Candès et al. “Robust principal component analysis?” In: *Journal of the ACM (JACM)* 58.3 (2011), p. 11.

[CGW05] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. “Clustering with qualitative information”. In: *Journal of Computer and System Sciences* 71.3 (2005), pp. 360–383.

[Cha+11] Venkat Chandrasekaran et al. “Rank-sparsity incoherence for matrix decomposition”. In: *SIAM Journal on Optimization* 21.2 (2011), pp. 572–596.

[Che+14] Yudong Chen et al. “Clustering partially observed graphs via convex optimization.” In: *Journal of Machine Learning Research* 15.1 (2014), pp. 2213–2238.

[Che+15] Gary K Chen et al. “Convex clustering: an attractive alternative to hierarchical clustering”. In: *PLoS computational biology* 11.5 (2015), e1004228.

[CK01] Anne Condon and Richard M Karp. “Algorithms for graph partitioning on the planted partition model”. In: *Random Structures and Algorithms* 18.2 (2001), pp. 116–140.

[Cou05] Julia Couto. “Kernel k-means for categorical data”. In: *IDA*. Springer. 2005, pp. 46–56.

[Cow00] Nelson Cowan. “The magical number 4 in short-term memory: A reconsideration of mental storage capacity”. In: *BEHAVIORAL AND BRAIN SCIENCES* 24 (2000), pp. 87–185.

[CPW10] Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. “Latent variable graphical model selection via convex optimization”. In: *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE. 2010, pp. 1610–1613.

[CR06] Emmanuel J Candes and Justin Romberg. “Quantitative robust uncertainty principles and optimally sparse decompositions”. In: *Foundations of Computational Mathematics* 6.2 (2006), pp. 227–254.

[CR09] Emmanuel J Candès and Benjamin Recht. “Exact matrix completion via convex optimization”. In: *Foundations of Computational mathematics* 9.6 (2009), p. 717.

[CSX12] Yudong Chen, Sujay Sanghavi, and Huan Xu. “Clustering sparse graphs”. In: *Advances in neural information processing systems*. 2012, pp. 2204–2212.

[CX16] Yudong Chen and Jiaming Xu. “Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 882–938.

[DK32] H. E Driver and A. I Kroeber. “Quantitative expression of cultural relationships”. In: 31 (1932), pp. 211–256.

[DM15] Yash Deshpande and Andrea Montanari. “Finding hidden cliques of size\sqrt {N/e} in nearly linear time”. In: *Foundations of Computational Mathematics* 15.4 (2015), pp. 1069–1128.

[DR01] Pedro Domingos and Matt Richardson. “Mining the network value of customers”. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2001, pp. 57–66.

[DS79] Alexander Philip Dawid and Allan M Skene. “Maximum likelihood estimation of observer error-rates using the EM algorithm”. In: *Applied statistics* (1979), pp. 20–28.

[EF03] Dotan Emanuel and Amos Fiat. “Correlation clustering–minimizing disagreements on arbitrary weighted graphs”. In: *European Symposium on Algorithms*. Springer. 2003, pp. 208–220.

[EKX95] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. “A database interface for clustering in large spatial databases”. In: *KDD*. 1995.

[Fel43] William Feller. “Generalization of a probability limit theorem of Cramér”. In: *Transactions of the American Mathematical Society*. 1943, pp. 361–372.

[FH16] Santo Fortunato and Darko Hric. “Community detection in networks: A user guide”. In: *Physics Reports* 659 (2016), pp. 1–44.

[For10] Santo Fortunato. “Community detection in graphs”. In: *Physics reports* 486.3 (2010), pp. 75–174.

[FP05] Li Fei-Fei and Pietro Perona. “A bayesian hierarchical model for learning natural scene categories”. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE. 2005, pp. 524–531.

[FTT04] Gary William Flake, Robert E Tarjan, and Kostas Tsioutsiouliklis. "Graph clustering and minimum cut trees". In: *Internet Mathematics* 1.4 (2004), pp. 385–408.

[GB08] Michael Grant and Stephen Boyd. "Graph implementations for nonsmooth convex programs". In: *Recent Advances in Learning and Control*. Ed. by V. Blondel, S. Boyd, and H. Kimura. Lecture Notes in Control and Information Sciences. http://stanford.edu/~boyd/graph_dcp.html. Springer-Verlag Limited, 2008, pp. 95–110.

[GB14] Michael Grant and Stephen Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.1*. http://cvxr.com/cvx. Mar. 2014.

[GG06] Ioannis Giotis and Venkatesan Guruswami. "Correlation clustering with a fixed number of clusters". In: *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics. 2006, pp. 1167–1176.

[Gom+11] Ryan G Gomes et al. "Crowdclustering". In: *Advances in neural information processing systems*. 2011, pp. 558–566.

[HLL83] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. "Stochastic blockmodels: First steps". In: *Social networks* 5.2 (1983), pp. 109–137.

[Hoc+11] Toby Dylan Hocking et al. "Clusterpath an algorithm for clustering using convex fusion penalties". In: *28th international conference on machine learning*. 2011, p. 1.

[HU13] Hannes Heikinheimo and Antti Ukkonen. "The crowd-median algorithm". In: *First AAAI Conference on Human Computation and Crowdsourcing*. 2013.

[HVH14] Eric Heim, Hamed Valizadegan, and Milos Hauskrecht. "Relative comparison kernel learning with auxiliary kernels". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2014, pp. 563–578.

[HWX15] Bruce Hajek, Yihong Wu, and Jiaming Xu. "Submatrix localization via message passing". In: *arXiv preprint arXiv:1510.09219* (2015).

[Jal+15] Amin Jalali et al. "Relative Density and Exact Recovery in Heterogeneous Stochastic Block Models". In: *arXiv preprint arXiv:1512.04937* (2015).

[Jam+14] Kevin Jamieson et al. "lil'ucb: An optimal exploration algorithm for multi-armed bandits". In: *Conference on Learning Theory*. 2014, pp. 423–439.

[JMF99] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. “Data clustering: a review”. In: *ACM computing surveys (CSUR)* 31.3 (1999), pp. 264–323.

[KB09] Tamara G Kolda and Brett W Bader. “Tensor decompositions and applications”. In: *SIAM review* 51.3 (2009), pp. 455–500.

[Kha+14] Mitesh M Khapra et al. “When Transliteration Met Crowdsourcing: An Empirical Study of Transliteration via Crowdsourcing using Efficient, Non-redundant and Fair Quality Control.” In: *LREC*. 2014, pp. 196–202.

[Kho+11] Aditya Khosla et al. “Novel dataset for fine-grained image categorization: Stanford dogs”. In: *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*. Vol. 2. 2011, p. 1.

[KOS11] David R Karger, Sewoong Oh, and Devavrat Shah. “Iterative learning for reliable crowdsourcing systems”. In: *Advances in neural information processing systems*. 2011, pp. 1953–1961.

[KOS14] David R Karger, Sewoong Oh, and Devavrat Shah. “Budget-optimal task allocation for reliable crowdsourcing systems”. In: *Operations Research* 62.1 (2014), pp. 1–24.

[KPJ04] Andrew D King, N Prvzulj, and Igor Jurisica. “Protein complex prediction via cost-based clustering”. In: *Bioinformatics* 20.17 (2004), pp. 3013–3020.

[Kru77] Joseph B Kruskal. “Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics”. In: *Linear algebra and its applications* 18.2 (1977), pp. 95–138.

[LCM10] Zhouchen Lin, Minming Chen, and Yi Ma. “The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices”. In: *arXiv preprint arXiv:1009.5055* (2010).

[Lin+13] Chris J Lintott et al. “Planet Hunters: New Kepler planet candidates from analysis of quarter 2”. In: *The Astronomical Journal* 145.6 (2013), p. 151.

[LJG01] Weizhong Li, Lukasz Jaroszewski, and Adam Godzik. “Clustering of highly homologous sequences to reduce the size of large protein databases”. In: *Bioinformatics* 17.3 (2001), pp. 282–283.

[LLS11] Zhouchen Lin, Risheng Liu, and Zhixun Su. “Linearized alternating direction method with adaptive penalty for low-rank representation”. In: *Advances in neural information processing systems*. 2011, pp. 612–620.

[LOL11] Fredrik Lindsten, Henrik Ohlsson, and Lennart Ljung. “Clustering using sum-of-norms regularization: With application to particle filter output computation”. In: *Statistical Signal Processing Workshop (SSP), 2011 IEEE*. IEEE. 2011, pp. 201–204.

[LPI12] Qiang Liu, Jian Peng, and Alexander T Ihler. “Variational inference for crowdsourcing”. In: *Advances in neural information processing systems*. 2012, pp. 692–700.

[LSS] LSST. *https://www.lsst.org/*.

[McS01] Frank McSherry. “Spectral partitioning of random graphs”. In: *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. IEEE. 2001, pp. 529–537.

[Mei07] Marina Meilă. “Comparing clusterings—an information based distance”. In: *Journal of multivariate analysis* 98.5 (2007), pp. 873–895.

[Mil56] George A Miller. “The magical number seven, plus or minus two: some limits on our capacity for processing information.” In: *Psychological review* 63.2 (1956), p. 81.

[Mis+07] Nina Mishra et al. “Clustering Social Networks”. In: *Proceedings of the 5th International Conference on Algorithms and Models for the Web-graph*. WAW’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 56–67.

[MS16] Arya Mazumdar and Barna Saha. “Clustering Via Crowdsourcing”. In: *arXiv preprint arXiv:1604.01839* (2016).

[MS17] Arya Mazumdar and Barna Saha. “A Theoretical Analysis of First Heuristics of Crowdsourced Entity Resolution.” In: *AAAI*. 2017, pp. 970–976.

[NJW02] Andrew Y Ng, Michael I Jordan, and Yair Weiss. “On spectral clustering: Analysis and an algorithm”. In: *Advances in neural information processing systems*. 2002, pp. 849–856.

[OH11] Samet Oymak and Babak Hassibi. “Finding dense clusters via" low rank+ sparse" decomposition”. In: *arXiv preprint arXiv:1104.5186* (2011).

[PW15] Mert Pilanci and Martin J Wainwright. “Randomized sketches of convex programs with sharp guarantees”. In: *IEEE Transactions on Information Theory* 61.9 (2015), pp. 5096–5115.

[R+11] Karl Rohe, Sourav Chatterjee, Bin Yu, et al. “Spectral clustering and the high-dimensional stochastic blockmodel”. In: *The Annals of Statistics* 39.4 (2011), pp. 1878–1915.

[Ray+10] Vikas C Raykar et al. “Learning from crowds”. In: *Journal of Machine Learning Research* 11.Apr (2010), pp. 1297–1322.

[SBG00] Nicholas D Sidiropoulos, Rasmus Bro, and Georgios B Giannakis. "Parallel factor analysis in sensor array processing". In: *IEEE transactions on Signal Processing* 48.8 (2000), pp. 2377–2388.

[SBS12] Marta Sabou, Kalina Bontcheva, and Arno Scharl. "Crowdsourcing research opportunities: lessons from natural language processing". In: *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*. ACM. 2012, p. 17.

[Sch07] Satu Elisa Schaeffer. "Graph clustering". In: *Computer science review* 1.1 (2007), pp. 27–64.

[SF08] Alexander Sorokin and David Forsyth. "Utility data annotation with amazon mechanical turk". In: *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. IEEE. 2008, pp. 1–8.

[Sno+08] Rion Snow et al. "Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks". In: *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics. 2008, pp. 254–263.

[SPD14] Robert Simpson, Kevin R Page, and David De Roure. "Zooniverse: observing the world's largest citizen science platform". In: *Proceedings of the 23rd international conference on world wide web*. ACM. 2014, pp. 1049–1054.

[Tam+11] Omer Tamuz et al. "Adaptively learning the crowd kernel". In: *arXiv preprint arXiv:1105.1033* (2011).

[Try39] J. W. Tryon. *Cluster analysis: Correlation profile and orthometric (factor) analysis for the isolation of unities in mind and personality*. Edwards brother Inc., Ann Arbor, Michigan, 1939.

[TSK05] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining. 1st*. 2005.

[urla] url. URL: https://griffsgraphs.wordpress.com/2012/07/02/a-facebook-network/.

[urlb] url. URL: https://www.mdc-berlin.de/10221541/en/research/research_teams/proteomics_and_molecular_mechanisms_of_neurodegenerative_diseases/research/research1.

[VB05] Ulrike Von Luxburg and Shai Ben-David. "Towards a statistical theory of clustering". In: *Pascal workshop on statistics and optimization of clustering*. 2005, pp. 20–26.

[VBD14] Norases Vesdapunt, Kedar Bellare, and Nilesh Dalvi. "Crowdsourcing algorithms for entity resolution". In: *Proceedings of the VLDB Endowment* 7.12 (2014), pp. 1071–1082.

[VG15] Vasilis Verroios and Hector Garcia-Molina. “Entity resolution with crowd errors”. In: *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE. 2015, pp. 219–230.

[VH15] Ramya Korlakai Vinayak and Babak Hassibi. “Clustering by Comparison: Stochastic Block Model for Inference in Crowdsourcing”. In: *Workshop on Machine Learning and Crowdsourcing, ICML*. 2015.

[VH16a] Ramya Korlakai Vinayak and Babak Hassibi. “Crowdsourced Clustering: Querying Edges vs Triangles”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 1316–1324.

[VH16b] Ramya Korlakai Vinayak and Babak Hassibi. “Similarity clustering in the presence of outliers: Exact recovery via convex program”. In: *IEEE International Symposium on Information Theory (ISIT), 2016.* IEEE. 2016, pp. 91–95.

[VM03] Deepak Verma and Marina Meila. “A comparison of spectral clustering algorithms”. In: *University of Washington Tech Rep UWCSE030501* 1 (2003), pp. 1–18.

[VOH14a] Ramya Korlakai Vinayak, Samet Oymak, and Babak Hassibi. “Graph clustering with missing data: Convex algorithms and analysis”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 2996–3004.

[VOH14b] Ramya Korlakai Vinayak, Samet Oymak, and Babak Hassibi. “Sharp performance bounds for graph clustering via convex optimization”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014*. IEEE. 2014, pp. 8297–8301.

[Von+08] Luis Von Ahn et al. “recaptcha: Human-based character recognition via web security measures”. In: *Science* 321.5895 (2008), pp. 1465–1468.

[Von06] Luis Von Ahn. “Games with a purpose”. In: *Computer* 39.6 (2006), pp. 92–94.

[Vu05] Van H Vu. “Spectral norm of random matrices”. In: *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*. ACM. 2005, pp. 423–430.

[VVV14] Aditya Vempaty, Lav R Varshney, and Pramod K Varshney. “Reliable crowdsourcing for multi-class labeling using coding theory”. In: *IEEE Journal of Selected Topics in Signal Processing* 8.4 (2014), pp. 667–679.

[VW12] Laurens Van Der Maaten and Kilian Weinberger. “Stochastic triplet embedding”. In: *Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on*. IEEE. 2012, pp. 1–6.

[VZH17] Ramya Korlakai Vinayak, Tijana Zrnic, and Babak Hassibi. “Tensor-based Crowdsourced Clustering via Triangle Queries”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017*. IEEE. 2017.

[Wah+11] Catherine Wah et al. “The caltech-ucsd birds-200-2011 dataset”. In: *Technical Report, California Institute of Technology* (2011).

[Wah+14] Catherine Wah et al. “Similarity comparisons for interactive fine-grained categorization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 859–866.

[Wan+12] Jiannan Wang et al. “Crowder: Crowdsourcing entity resolution”. In: *Proceedings of the VLDB Endowment* 5.11 (2012), pp. 1483–1494.

[Wel+10] Peter Welinder et al. “The multidimensional wisdom of crowds”. In: *Advances in neural information processing systems*. 2010, pp. 2424–2432.

[WKB14] Michael J Wilber, Iljung S Kwak, and Serge J Belongie. “Cost-effective hits for relative similarity comparisons”. In: *Second AAAI Conference on Human Computation and Crowdsourcing*. 2014.

[WM16] Yali Wan and Marina Meila. “Graph Clustering: Block-models and model free results”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 2478–2486.

[XCS10] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. “Robust PCA via outlier pursuit”. In: *Advances in Neural Information Processing Systems*. 2010, pp. 2496–2504.

[XJK99] Xiaowei Xu, Jochen Jäger, and Hans-Peter Kriegel. “A fast parallel clustering algorithm for large spatial databases”. In: *High Performance Data Mining*. Springer, 1999, pp. 263–290.

[XOX02] Ying Xu, Victor Olman, and Dong Xu. “Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees”. In: *Bioinformatics* 18.4 (2002), pp. 536–545.

[XT15] Dongkuan Xu and Yingjie Tian. “A comprehensive survey of clustering algorithms”. In: *Annals of Data Science* 2.2 (2015), pp. 165–193.

[YF04] Ossama Younis and Sonia Fahmy. “HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks”. In: *IEEE Transactions on mobile computing* 3.4 (2004), pp. 366–379.

[Yi+12] Jinfeng Yi et al. “Semi-crowdsourced clustering: Generalizing crowd labeling by robust distance metric learning”. In: *Advances in neural information processing systems*. 2012, pp. 1772–1780.

[YL05] Qiaofeng Yang and Stefano Lonardi. “A parallel algorithm for clustering protein-protein interaction networks”. In: *Computational Systems Bioinformatics Conference, 2005. Workshops and Poster Abstracts. IEEE*. IEEE. 2005, pp. 174–177.

[Yur+17] Alp Yurtsever et al. “Sketchy Decisions: Convex Low-Rank Matrix Optimization with Optimal Storage”. In: *20th International Conference on Artificial Intelligence and Statistics (AISTATS2017)*. EPFL-CONF-225653. 2017.

[Zha+14] Yuchen Zhang et al. “Spectral methods meet EM: A provably optimal algorithm for crowdsourcing”. In: *Advances in neural information processing systems*. 2014, pp. 1260–1268.

[Zha+15] Kai Zhang et al. “From categorical to numerical: Multiple transitive distance learning and embedding”. In: *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM. 2015, pp. 46–54.

[Zho+12] Denny Zhou et al. “Learning from the wisdom of crowds by minimax entropy”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 2195–2203.

[Zub38] J.A Zubin. “A technique for measuring likemindedness”. In: 33 (1938), pp. 508–516.

Appendix A

PROOFS FOR RESULTS IN CHAPTER 2

In this appendix we provide the detailed proofs for the results presened in Chapter 2 (Theorem 1 and Theorem 2).

A.1 Proof of Results for Simple Convex Program (Theorem 1)

Recall the Simple Program 2.1.4 from Chapter 2:

$$\begin{aligned} &\underset{\mathbf{L},\mathbf{S}}{\text{minimize}} \ \ \|\mathbf{L}\|_\star + \lambda\|\mathbf{S}\|_1 \\ &\text{subject to} \\ &\quad 1 \geq \mathbf{L}_{i,j} \geq 0 \ \text{ for all } i,j \in \{1,2,\dots n\} \\ &\quad \mathbf{L}_{obs} + \mathbf{S}_{obs} = \mathbf{A}_{obs} \end{aligned}$$

Let $1 \geq p_{min} > \frac{1}{2} > q > 0$ and $0 \leq r \leq 1$. $\mathcal{G}$ be a random graph generated according to the stochastic block model 2.2.1 with cluster sizes $\{n_i\}_{i=1}^K$. Let the observation model be as defined in (Defn 2.2.2). Our goal is to show that the optimal solution of Problem 2.1.4 is the pair $(\mathbf{L}^0, \mathbf{S}^0)$ under reasonable conditions, where

$$\mathbf{L}^0 = \mathbb{1}_{\mathcal{R}}^{n\times n}, \ \mathbf{S}^0 = \mathbf{S}_{obs}^0 = \mathbf{A}_{obs} - \mathbf{L}^0. \tag{A.1.1}$$

Theorem 1 is based on the following lemmas:

Lemma A.1.1. *If $\lambda > \Lambda_{fail}$ or $\mathbf{D}_{\min} < \frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is not an optimal solution to the Program 2.1.4 with high probability.*

Lemma A.1.2. *If $\lambda < \Lambda_{succ}$ and $\mathbf{D}_{\min} > \frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to Program 2.1.4 with high probability.*

Before we proceed, we need some additional notations. Let $\mathcal{R}_{i,j} = \mathcal{C}_i \times \mathcal{C}_j$ for $1 \leq i,j \leq K+1$. One can see that $\{\mathcal{R}_{i,j}\}$ divides $[n] \times [n]$ into $(K+1)^2$ disjoint regions similar to a grid which is illustrated in the Figure A.1. Thus, $\mathcal{R}_{i,i}$ is the region induced by i'th cluster for any $1 \leq i \leq K$.

Let Γ^{out} be the set of entries of adjacency matrix that are *not* observed. Let $\mathcal{A}_1 \subseteq [n] \times [n]$ be the set of nonzero coordinates of $\mathbf{A}_{obs}$, and $\mathcal{A}_0 \subseteq [n] \times [n]$ be the set of coordinates of $\mathbf{A}_{obs}$ that are zero. Then the sets,

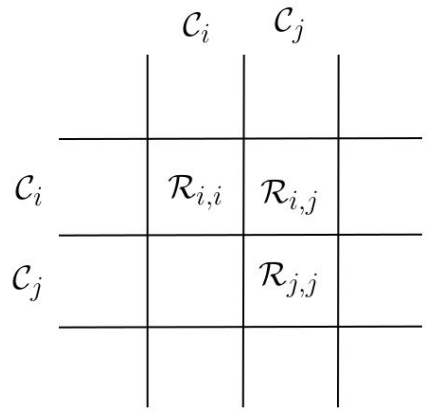


Figure A.1: Illustration of $\{\mathcal{R}_{i,j}\}$ dividing $[n] \times [n]$ into disjoint regions similar to a grid

1. $\mathcal{A}_1 \cap \mathcal{R}$ corresponds to the edges inside the clusters that are observed.
2. $\mathcal{A}_1 \cap \mathcal{R}^c$ corresponds to the set of edges outside the clusters that are observed.
3. $\mathcal{A}_0 \cap \mathcal{R}$ corresponds to the missing edges inside the clusters, that are observed (that is, we know that the edge does not exist).

Let c and d be positive integers. Consider a matrix, $\mathbf{X} \in \mathbb{R}^{c \times d}$. Let β be a subset of $[c] \times [d]$. Then, let $\mathbf{X}_\beta$ denote the matrix induced by the entries of $\mathbf{X}$ on β i.e.,

$$(\mathbf{X}_\beta)_{i,j} = \begin{cases} \mathbf{X}_{i,j} & \text{if } (i,j) \in \beta \\ 0 & \text{otherwise .} \end{cases}$$

In other words, $\mathbf{X}_\beta$ is a matrix whose entries match those of $\mathbf{X}$ in the positions $(i,j) \in \beta$ and zero otherwise. For example, $\mathbb{1}_{\mathcal{A}^{obs}}^{n \times n} = \mathbf{A}^{obs}$. Given a matrix $\mathbf{X}$, $\text{sum}(\mathbf{X})$ will denote the sum of all entries of $\mathbf{X}$. Finally, we introduce the following parameter which will be useful for the subsequent analysis. This parameter can be seen as a measure of distinctness of the "worst" cluster from the "background noise". Here, by background noise we mean the edges over $\mathcal{R}^c$. Given $q, \{p_i\}_{i=1}^K$, let,

$$\begin{aligned} \mathbf{D}_{\mathcal{A}} &= \frac{1}{2} \min \left\{ r(1-2q), \left\{ r(2p_i - 1) - \frac{1}{\lambda n_i} \right\}_{i=1}^{K} \right\} \\ &= \frac{1}{2} \min \left\{ r(1-2q), \frac{\mathbf{D}_i - \lambda^{-1}}{n_i} \right\}. \end{aligned} \tag{A.1.2}$$

For our proofs, we will make use of the following Big O notation. $f(n) = \Omega(n)$ will mean there exists a positive constant c such that for sufficiently large n, $f(n) \geq cn$.

$f(n) = O(n)$ will mean there exists a positive constant c such that for sufficiently large n, $f(n) \leq cn$.

A.1.1 Proof of Lemma A.1.1

Lagrange for the problem (2.1.4) can be written as follows:

$$\mathscr{L}(\mathbf{L}, \mathbf{S}; \mathbf{M}, \mathbf{N}) \quad = \quad \|\mathbf{L}\|_\star + \lambda \|\mathbf{S}_{obs}\|_1 + \text{trace}(\mathbf{M}(\mathbf{L} - \mathbb{1}\mathbb{1}^T)) - \text{trace}(\mathbf{NL}), \tag{A.1.3}$$

where $\mathbf{M}$ and $\mathbf{N}$ are dual variables corresponding to the inequality constraints (2.1.5).

For $\mathbf{L}^0$ to be an optimal solution to (2.1.4), it has to satisfy the KKT conditions. Therefore, the subgradient of (A.1.3) at $\mathbf{L}^0$ has to be 0, i.e.,

$$\partial \|\mathbf{L}^0\|_\star + \lambda \, \partial \|\mathbf{A}_{obs} - \mathbf{L}_{obs}{}^0\|_1 + \mathbf{M}^0 - \mathbf{N}^0 = 0, \tag{A.1.4}$$

where $\mathbf{M}^0$ and $\mathbf{N}^0$ are optimal dual variables, and $\partial \|\mathbf{L}^0\|_\star$ and $\partial \|\mathbf{S}^0\|_1$ are subgradients of nuclear norm and ℓ_1-norm respectively at the points $(\mathbf{L}^0, \mathbf{S}^0)$. Note that in the standard notation, $\partial h(\mathbf{x})$ denotes the set of all subgradients, i.e., the subdifferential of the function $h(.)$ at $\mathbf{x}$. We have slightly abused the notation by denoting a subgradient of the function $h(.)$ at the point $\mathbf{x}$ by $\partial h(\mathbf{x})$. In other words, $\partial h(\mathbf{x})$ has been used to denote any element in the subgradient set of $h(.)$ at $\mathbf{x}$.

Also, by complementary slackness,

$$\text{trace}(\mathbf{M}^0(\mathbf{L}^0 - \mathbb{1}\mathbb{1}^T)) = 0, \tag{A.1.5}$$

and

$$\text{trace}(\mathbf{N}^0\mathbf{L}^0) = 0. \tag{A.1.6}$$

From (A.1.1) and (A.1.5), (A.1.6), we have $(\mathbf{M}^0)_{\mathcal{R}} \geq 0$, $(\mathbf{M}^0)_{\mathcal{R}^c} = 0$, $(\mathbf{N}^0)_{\mathcal{R}} = 0$ and $(\mathbf{N}^0)_{\mathcal{R}^c} \geq 0$. Therefore $(\mathbf{M}^0 - \mathbf{N}^0)_{\mathcal{R}} \geq 0$ and $(\mathbf{M}^0 - \mathbf{N}^0)_{\mathcal{R}^c} \leq 0$.

Let $\mathbf{L}^0 = \mathbf{U}\Lambda\mathbf{U}^T$, where $\Lambda = \text{diag}\{n_1, n_2, \ldots, n_K\}\mathbf{U} = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_K] \in \mathbb{R}^{n \times K}$,

$$\mathbf{u}_{l,i} = \begin{cases} \frac{1}{\sqrt{n_l}} & \text{if } i \in \mathcal{C}_l \\ 0 & \text{else.} \end{cases} \tag{A.1.7}$$

Then the subgradient $\partial \|\mathbf{L}^0\|_\star$ is of the form $\mathbf{U}\mathbf{U}^T + \mathbf{W}$ such that $\mathbf{W} \in \{\mathbf{X} : \mathbf{XU} = \mathbf{U}^T\mathbf{X} = 0, \|\mathbf{X}\| \leq 1\}$. The subgradient $\partial \|\mathbf{S}^0\|_1$ is of the form $\text{sign}(\mathbf{S}^0) + \mathbf{Q}$, where $\mathbf{Q}_{i,j} = 0$ if $\mathbf{S}_{i,j} \neq 0$ and $\|\mathbf{Q}\|_\infty \leq 1$.

From (A.1.4), we have

$$\mathbf{U}\mathbf{U}^T + \mathbf{W} - \lambda\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right) + (\mathbf{M}^0 - \mathbf{N}^0) = 0. \tag{A.1.8}$$

Consider the sum of the entires corresponding to the cluster i ($\mathcal{R}_{i,i}$), i.e.,

$$\begin{aligned}&\underbrace{\text{sum}\left(\mathbf{L}^0\right)_{\mathcal{R}_{i,i}}}_{n_i} - \text{sum}\left(\lambda\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right)_{\mathcal{R}_{i,i}}\right)\\ &+ \underbrace{\text{sum}\left(\mathbf{M}^0 - \mathbf{N}^0\right)_{\mathcal{R}_{i,i}}}_{\geq 0} = 0\end{aligned} \tag{A.1.9}$$

Since each entry of the adjacency matrix is observed with probability r, and the probability of missing edge inside cluster i is $1 - p_i$, we note that $(\mathbf{S}^0_{\mathcal{R}_{i,i}})_{l,m} \neq 0$ with probability $r(1-p_i)$. Recall that $\mathbf{Q}_{l,m} = 0$ if $\mathbf{S}^0_{l,m} \neq 0$.

Then by Bernstein's inequality and using $\|\mathbf{Q}\|_\infty \leq 1$, with probability $1 - \exp\left(-\Omega(n_i^2)\right)$ we have $\text{sum}\left(\text{sign}(\mathbf{S}^0)\right) = -n_i^2 r(1-p_i)$ and $\text{sum}\left(\mathbf{Q}\right) \leq n_i^2(1 - r(1-p_i))$.

Thus,

$$\begin{aligned}-\text{sum}\left(\lambda\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right)_{\mathcal{R}_{i,i}}\right) &\geq \lambda n_i^2 r(1-p_i) - \lambda n_i^2\left(1 - r(1-p_i)\right)\\ &= \lambda n_i^2\left[2r(1-p_i) - 1\right],\end{aligned}$$

and hence LHS of equation (A.1.9) can be lower bounded as ,

$$n_i - \text{sum}\left(\lambda\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right)_{\mathcal{R}_{i,i}}\right) + \underbrace{\text{sum}\left(\mathbf{M}^0 - \mathbf{N}^0\right)_{\mathcal{R}_{i,i}}}_{\geq 0} \geq n_i + \lambda n_i^2\left[2r(1-p_i) - 1\right].$$

We see that $n_i\left(1 - 2r(1-p_i)\right) < \frac{1}{\lambda}$ would imply $n_i + \lambda n_i^2\left[2r(1-p_i) - 1\right] > 0$, in which case, the equation (A.1.4) does not hold. Hence $\mathbf{L}^0$ cannot be an optimal solution to the Program 2.1.4. (Note that, $p_i > \frac{1}{2}$ and hence $2p_i - 1 > 0$.)

Notice that $\left(\mathbf{U}\mathbf{U}^T\right)_{\mathcal{R}^c} = 0$ and the entries of $-\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right)$ and $\mathbf{M}^0 - \mathbf{N}^0$ over $\mathcal{R}^c \cap \mathcal{A}_1$ are negative. Hence from the equation (A.1.8),

$$\|\mathbf{W}\|_F^2 \geq \|\left(\mathbf{U}\mathbf{U}^T + \mathbf{W}\right)_{(\mathcal{R}^c \cap \mathcal{A}_1)}\|_F^2 \geq \|\lambda\left(\text{sign}(\mathbf{S}^0) + \mathbf{Q}\right)_{(\mathcal{R}^c \cap \mathcal{A}_1)}\|_F^2 \tag{A.1.10}$$

Recall that $\mathbf{S}^0_{(\mathcal{R}_c \cap \mathcal{A}_1)} \neq 0$ and hence $\mathbf{Q}_{(\mathcal{R}_c \cap \mathcal{A}_1)} = 0$. Further, recall that by the stochastic block model, each entry of $\mathbf{A}$ over $\mathcal{R}^c$ is non-zero with probability q and by observation model (Defn 2.2.2), each entry of $\mathbf{A}$ is observed with probability r. Hence with probability at least $1 - \exp\left(-\Omega(|\mathcal{R}_c|)\right)$, $|\mathcal{R}_c \cap \mathcal{A}_1| = rq(n^2 - \sum_{i=1}^K n_i^2)$. Thus from equation (A.1.10) we have,

$$\|\mathbf{W}\|_F^2 \geq \lambda^2 rq(n^2 - \sum_{i=1}^K n_i^2). \tag{A.1.11}$$

Recall that $\|\mathbf{W}\| \leq 1$ should hold true for $(\mathbf{L}^0, \mathbf{S}^0)$ to be an optimal solution to Program 2.1.4. $\|\mathbf{W}\| = |\sigma_{\max}(\mathbf{W})| \geq \frac{\|\mathbf{W}\|_F}{\sqrt{n}}$, which on combining with equation (A.1.11) gives us,

$$\|\mathbf{W}\| \geq \lambda \sqrt{\frac{rq\left(n^2 - \sum_{i=1}^{K} n_i^2\right)}{n}}.$$

So, if $\lambda\sqrt{rq\left(n^2 - \sum_{i=1}^{K} n_i^2\right)/n} > 1$ then, $(\mathbf{L}^0, \mathbf{S}^0)$ cannot be an optimal solution to Program 2.1.4. This gives us the result in Lemma A.1.1.

A.1.2 Proof of Lemma A.1.2

To prove Lemma A.1.2, we need to show that when $\lambda < \mathbf{\Lambda}_{succ}$ and $\mathbf{D}_{\min} > \frac{1}{\lambda}$, $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to the Program 2.1.4. So, we need to prove that for all feasible perturbations $(\mathbf{E}^L, \mathbf{E}^S)$,

$$(\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda \|\mathbf{S}^0 + \mathbf{E}^S\|_1) - (\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1) > 0. \tag{A.1.12}$$

We note that $\mathbf{S}$ can be split as $\mathbf{S} = \mathbf{S}_{obs} + \mathbf{S}_{rest}$, where $\mathbf{S}_{rest}$ denotes the entries of $\mathbf{S}$ other than those corresponding to the observed entries of $\mathbf{A}$. Furthermore, we claim that at the optimal, $\mathbf{S}_{rest} = 0$, since if otherwise, the objective can be strictly decreased by setting $\mathbf{S}_{rest} = 0$. Hence, $\mathbf{S} = \mathbf{S}_{obs}$.

We can lower bound the LHS of the equation (A.1.12) using the subgradients as follows:

$$(\|\mathbf{L}^0+\mathbf{E}^L\|_\star+\lambda \|\mathbf{S}^0+\mathbf{E}^S\|_1)-\left(\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1\right) \geq \langle \partial\|\mathbf{L}^0\|_\star, \mathbf{E}^L\rangle+\lambda\langle \partial\|\mathbf{S}^0\|_1, \mathbf{E}^S\rangle, \tag{A.1.13}$$

where $\partial\|\mathbf{L}^0\|_\star$ and $\partial\|\mathbf{S}^0\|_1$ are subgradients of nuclear norm and ℓ_1-norm respectively at the points $(\mathbf{L}^0, \mathbf{S}^0)$. Note that in the standard notation, $\partial\|\mathbf{x}\|_*$ denotes the set of all subgradients, i.e., the subdifferential. We have slightly abused the notation by denoting a subgradient of a norm $\|\cdot\|_*$ at the point $\mathbf{x}$ by $\partial\|\mathbf{x}\|_*$.

To make use of (A.1.13), it is very important to choose good subgradients. In the following section we will focus on construction of such subgradients.

Subgradient construction

Recall that, $\mathbf{L}^0 = \mathbf{U}\Lambda\mathbf{U}^T$, where $\Lambda = \text{diag}\{n_1, n_2, \ldots, n_K\}$ and $\mathbf{U} = [\mathbf{u}_1 \ \ldots \ \mathbf{u}_K] \in \mathbb{R}^{n\times K}$, with $\mathbf{u}_l$ as defined before. Then the subgradient $\partial\|\mathbf{L}^0\|_\star$ is of the form

$\mathbf{U}\mathbf{U}^T + \mathbf{W}$ such that $\mathbf{W} \in \mathcal{M}_U := \{\mathbf{X} : \mathbf{X}\mathbf{U} = \mathbf{U}^T\mathbf{X} = 0, \|\mathbf{X}\| \leq 1\}$. $\|.\|$ is spectral norm (maximum singular value). The subgradient $\partial\|\mathbf{S}^0\|_1$ is of the form $\text{sign}(\mathbf{S}^0) + \mathbf{Q}$ where $\mathbf{Q}_{i,j} = 0$ if $\mathbf{S}^0_{i,j} \neq 0$ and $\|\mathbf{Q}\|_\infty \leq 1$.

$$\begin{aligned}
&\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda \|\mathbf{S}^0 + \mathbf{E}^S\|_1 - (\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1) \\
&\geq \langle \partial\|\mathbf{L}^0\|_\star, \mathbf{E}^L\rangle + \lambda\langle \partial\|\mathbf{S}^0\|_1, \mathbf{E}^S\rangle \\
&= \langle \mathbf{U}\mathbf{U}^T + \mathbf{W}, \mathbf{E}^L\rangle + \lambda\langle \text{sign}(\mathbf{S}^0) + \mathbf{Q}, \mathbf{E}^S\rangle
\end{aligned}$$

Note that, due to the condition $\mathbf{L}_{obs} + \mathbf{S}_{obs} = \mathbf{A}_{obs}$, we have $\mathbf{E}^S = {\mathbf{E}_{obs}}^L$. Further, note that $\text{sign}(\mathbf{S}^0) = \mathbb{1}^{n\times n}_{\mathcal{A}_1\cap\mathcal{R}^c} - \mathbb{1}^{n\times n}_{\mathcal{A}_0\cap\mathcal{R}}$. Choosing $\mathbf{Q} = \mathbb{1}^{n\times n}_{\mathcal{A}_1\cap\mathcal{R}} - \mathbb{1}^{n\times n}_{\mathcal{A}_0\cap\mathcal{R}^c}$, we get,

$$\begin{aligned}
&\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda \|\mathbf{S}^0 + \mathbf{E}^S\|_1 - (\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1) \\
&\geq \left\langle \mathbf{W}, \mathbf{E}^L\right\rangle + \underbrace{\sum_{i=1}^{K} \frac{1}{n_i}\text{sum}(\mathbf{E}_{R_{i,i}}) + \lambda\left(\text{sum}(\mathbf{E}^L_{\mathcal{A}_0}) - \text{sum}(\mathbf{E}^L_{\mathcal{A}_1})\right)}_{:=g(\mathbf{E}^L)}.
\end{aligned}$$

From this point onward, for simplicity we will ignore the superscript L on $\mathbf{E}^L$ and just use $\mathbf{E}$.

Define,

$$g(\mathbf{E}) := \sum_{i=1}^{K} \frac{1}{n_i}\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + \lambda\left(\text{sum}(\mathbf{E}_{\mathcal{A}_0}) - \text{sum}(\mathbf{E}_{\mathcal{A}_1})\right). \tag{A.1.14}$$

Also, define $f(\mathbf{E}, \mathbf{W}) := g(\mathbf{E}) + \langle \mathbf{W}, \mathbf{E}\rangle$. Our aim is to show that for all feasible perturbations $\mathbf{E}$, there exists $\mathbf{W}$ such that,

$$f(\mathbf{E}, \mathbf{W}) = g(\mathbf{E}) + \langle \mathbf{W}, \mathbf{E}\rangle > 0. \tag{A.1.15}$$

Note that $g(\mathbf{E})$ does not depend on $\mathbf{W}$.

Lemma A.1.3. *Given* $\mathbf{E}$*, assume there exists* $\mathbf{W} \in \mathcal{M}_\mathbf{U}$ *with* $\|\mathbf{W}\| < 1$ *such that* $f(\mathbf{E}, \mathbf{W}) \geq 0$*. Then at least one of the followings holds:*

- *There exists* $\mathbf{W}^* \in \mathcal{M}_\mathbf{U}$ *with* $\|\mathbf{W}^*\| \leq 1$ *and* $f(\mathbf{E}, \mathbf{W}^*) > 0$.
- *For all* $\mathbf{W} \in \mathcal{M}_\mathbf{U}$, $\langle \mathbf{E}, \mathbf{W}\rangle = 0$.

Proof. Let $c = 1 - \|\mathbf{W}\|$. Assume $\langle \mathbf{E}, \mathbf{W}' \rangle \neq 0$ for some $\mathbf{W}' \in \mathcal{M}_{\mathbf{U}}$. If $\langle \mathbf{E}, \mathbf{W}' \rangle > 0$, choose $\mathbf{W}^* = \mathbf{W} + c\mathbf{W}'$. Otherwise, choose $\mathbf{W}^* = \mathbf{W} - c\mathbf{W}'$. Since $\|\mathbf{W}'\| \leq 1$, we have, $\|\mathbf{W}^*\| \leq 1$ and $\mathbf{W}^* \in \mathcal{M}_{\mathbf{U}}$. Consequently,

$$f(\mathbf{E}, \mathbf{W}^*) = f(\mathbf{E}, \mathbf{W}) + \langle \mathbf{E}, c\mathbf{W}' \rangle > f(\mathbf{E}, \mathbf{W}) \geq 0. \tag{A.1.16}$$

■

Notice that, for all $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$, $\langle \mathbf{E}, \mathbf{W} \rangle = 0$ is equivalent to $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$ which is the orthogonal complement of $\mathcal{M}_{\mathbf{U}}$ in $\mathbb{R}^{n \times n}$. $\mathcal{M}_{\mathbf{U}}^{\perp}$ has the following characterization:

$$\mathcal{M}_{\mathbf{U}}^{\perp} = \{\mathbf{X} \in \mathbb{R}^{n \times n} : \mathbf{X} = \mathbf{U}\mathbf{M}^T + \mathbf{N}\mathbf{U}^T \text{ for some } \mathbf{M}, \mathbf{N} \in \mathbb{R}^{n \times K}\}. \tag{A.1.17}$$

Now we have broken down our aim into two steps.

1. Construct $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$, such that $f(\mathbf{E}, \mathbf{W}) \geq 0$ for all feasible perturbations $\mathbf{E}$.
2. For all non-zero feasible $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$, show that $g(\mathbf{E}) > 0$.

As a first step, in Section A.1.3, we will argue that, under certain conditions, there exists a $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$ such that with high probability, $f(\mathbf{E}, \mathbf{W}) \geq 0$ for all feasible $\mathbf{E}$. This $\mathbf{W}$ is called the dual certificate. Secondly, in Section A.1.4, we will show that, under certain conditions, for all $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$ with high probability, $g(\mathbf{E}) > 0$. Finally, combining these two arguments, and using Lemma A.1.3 we will conclude that $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal with high probability.

A.1.3 Showing existence of the dual certificate

Recall that

$$f(\mathbf{E}, \mathbf{W}) = \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + \langle \mathbf{E}, \mathbf{W} \rangle + \lambda \left(\text{sum}\left(\mathbf{E}_{\mathcal{A}_0}\right) - \text{sum}\left(\mathbf{E}_{\mathcal{A}_1}\right) \right).$$

$\mathbf{W}$ will be constructed from the candidate $\mathbf{W}_0$, which is given as follows.

Candidate $\mathbf{W}_0$

Based on Program 2.1.4, we propose the following:

$$\mathbf{W}_0 = \sum_{i=1}^{K} c_i \mathbb{1}_{\mathcal{R}_{i,i}}^{n \times n} + c\mathbb{1}_{\mathcal{R}^c}^{n \times n} + \lambda \left(\mathbb{1}_{\mathcal{A}_1}^{n \times n} - \mathbb{1}_{\mathcal{A}_0}^{n \times n} \right), \tag{A.1.18}$$

where $\{c_i\}_{i=1}^K, c$ are real numbers to be determined.

$$f(\mathbf{E}, \mathbf{W}^0) = \sum_{i=1}^{K} (\frac{1}{n_i} + c_i) \text{ sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + c \text{ sum}(\mathbf{E}_{\mathcal{R}^c}).$$

Note that $\mathbf{W}_0$ is a random matrix where randomness is due to $\mathbf{A}_{obs}$. In order to ensure a small spectral norm, we will set its expectation to 0, i.e., we will choose $c, \{c_i\}'s$ to ensure that $\mathbb{E}[\mathbf{W}_0] = 0$.

Following from the partially observed Stochastic Block Model (Defn 2.2.1 and 2.2.2), the expectation of an entry of $\mathbf{W}_0$ on $\mathcal{R}_{i,i}$ (region corresponding to cluster i) and $\mathcal{R}^c$ (region outside the clusters) is $c_i + \lambda r(2p_i - 1)$ and $c + \lambda r(2q - 1)$ respectively. Therefore, we set,

$$c_i = -\lambda r(2p_i - 1) \quad \text{and} \quad c = -\lambda r(2q - 1),$$

With these choices, the candidate, $\mathbf{W}_0$ and $f(\mathbf{E}, \mathbf{W}_0)$ take the following forms:

$$\begin{aligned}\mathbf{W}_0 = \lambda \sum_{i=1}^{K} &(1 + r(1 - 2p_i)) \ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_1} + \lambda(1 - r(1 - 2p_i)) \ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_0} \\ &+\lambda r(1 - 2p_i)\mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\Gamma^{out}} + \lambda(1 + r(1 - 2q)) \ \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_1} \\ &+\lambda(1 - r(1 - 2q)) \ \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_0} + \lambda r(1 - 2q)\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\Gamma^{out}} \end{aligned} \tag{A.1.19}$$

$$f(\mathbf{E}, \mathbf{W}_0) = \lambda \left[r(1 - 2q) \text{ sum}(\mathbf{E}_{\mathcal{R}^c})\right] - \lambda \left[\sum_{i=1}^{K} \left(r(2p_i - 1) - \frac{1}{\lambda n_i}\right) \text{ sum}(\mathbf{E}_{\mathcal{R}_{i,i}})\right]$$

From $\mathbf{L}^0$ and the constraint $1 \geq \mathbf{L}_{i,j} \geq 0$, it follows that,

$$\begin{aligned} &\mathbf{E}_{\mathcal{R}^c} \text{ is (entrywise) nonnegative.} \\ &\mathbf{E}_{\mathcal{R}} \text{ is (entrywise) nonpositive.} \end{aligned} \tag{A.1.20}$$

Thus, $\text{sum}(\mathbf{E}_{\mathcal{R}^c}) \leq 0$ and $\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) \geq 0$. When $\lambda(2p_i - 1) - \frac{1}{n_i} \geq 0$ and $\lambda(2q - 1) \leq 0$; we will have $f(\mathbf{E}, \mathbf{W}_0) \geq 0$ for all feasible $\mathbf{E}$. This indeed holds due to the assumptions of Theorem 1 (see (A.1.2)), as we assumed $r(2p_i - 1) > \frac{1}{\lambda n_i}$ for $i = 1, 2 \cdots, K$ and $1 > 2q$.

We will now proceed to find a tight bound on the spectral norm of $\mathbf{W}^0$. We will say that random variable X has a $\Delta(\zeta, \delta)$ distribution for $0 \leq \zeta, \delta \leq 1$, written as $X \sim \Delta(\zeta, \delta)$ if,

$$X = \begin{cases} 1 + \zeta(1-2\delta) \text{ w.p. } \zeta\delta \\ 1 - \zeta(1-2\delta) \text{ w.p. } \zeta(1-\delta) \\ r(1-2\delta) \text{ w.p. } 1-\zeta \end{cases}$$

Variance of the above distribution is

$$\text{Var}(X) = 1 - \zeta + 4\,\zeta\,\delta\,(1-\delta). \tag{A.1.21}$$

Theorem 10. *Assume* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *obeys the Stochastic Block Model* (2.2.1) *and let* $\mathbf{M} \in \mathbb{R}^{n\times n}$. *Let entries of* $\mathbf{M}$ *be as follows.*

$$\mathbf{M}_{i,j} \sim \begin{cases} \Delta(r, p_k) & \textit{if} \quad (i,j) \in \mathcal{R}_{k,k} \\ \Delta(r, q) & \textit{if} \quad (i,j) \in \mathcal{R}^c \end{cases}$$

Then, for a constant ϵ' *(to be determined) each of the following holds with probability* $1 - \exp(-\Omega(n))$.

- $\|\mathbf{M}\| \leq 2\sqrt{nr}\sqrt{1-r+4rq(1-q)}$
  $+ \max\limits_{1\leq i\leq K} 2\sqrt{n_i r}\sqrt{2(1-r)+4r(q(1-q)+p_i(1-p_i))} + \epsilon'\sqrt{n}.$
- *Assume* $\max\limits_{1\leq i\leq K} n_i = o(n)$. *Then, for sufficiently large* n,

$$\|\mathbf{M}\| \leq (2\sqrt{r(1-r+4rq(1-q))} + \epsilon')\sqrt{n}.$$

Proof. For the first statement, let $\mathbf{M}_1$ be a random matrix with independent entries distributed as:

$$\mathbf{M}_1(i,j) \sim \Delta(r, q).$$

From standard results on random matrix theory [Vu05], it follows that,

$$\|\mathbf{M}_1\| \leq (2\sqrt{r(1-r+4rq(1-q))} + \epsilon')\sqrt{n}$$

with the desired probability.

Also let $\mathbf{M}_2 = \mathbf{M} - \mathbf{M}_1$. We note that $\mathbf{M}_2$ is a block diagonal random matrix. Observe that $\mathbf{M}_2$ over $\mathcal{R}_{i,i}$, $\mathbf{M}_{2,\mathcal{R}_{i,i}}$ is the sum of two independent random variables

$\mathbf{M}_{\mathcal{R}_{i,i}} \sim \Delta(r, p_i)$ and $-\mathbf{M}_{1,\mathcal{R}_{i,i}} \sim \Delta(r, q)$. So, the variance is $2r(1-r)+4r^2(q(1-q)+p_i(1-p_i))$. This similarly gives,

$$\|\mathbf{M}_{2,\mathcal{R}_{i,i}}\| \leq 2\sqrt{2r(1-r)+4r^2(q(1-q)+p_i(1-p_i))}\sqrt{n_i}+\epsilon'\sqrt{n}$$

Now, observing, $\|\mathbf{M}_2\| = \sup_{1\leq i\leq K} \|\mathbf{M}_{2,\mathcal{R}_{i,i}}\|$ and using a union bound over $i \leq K$ we have,

$$\|\mathbf{M}_2\| \leq \max_{1\leq i\leq K} 2\sqrt{2r(1-r)+4r^2(q(1-q)+p_i(1-p_i))}\sqrt{n_i}+\epsilon'\sqrt{n}.$$

Finally, we use the triangle inequality $\|\mathbf{M}\| \leq \|\mathbf{M}_1\| + \|\mathbf{M}_2\|$ to conclude. ■

The following lemma gives a bound on $\|\mathbf{W}_0\|$.

Lemma A.1.4. *Recall that, $\mathbf{W}_0$ is a random matrix; where randomness is on the partially observed stochastic block model $\mathbf{A}_{obs}$ and it is given by,*

$$\begin{aligned}\mathbf{W}_0 \;=\; & \lambda\sum_{i=1}^{K}(1+r(1-2p_i))\ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_1} + \lambda(1-r(1-2p_i))\ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_0} \\ & +\lambda r(1-2p_i)\mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\Gamma^{out}} + \lambda(1+r(1-2q))\ \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_1} \\ & +\lambda(1-r(1-2q))\ \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_0} + \lambda r(1-2q)\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\Gamma^{out}}.\end{aligned}$$

Then, for any $\epsilon' > 0$, with probability $1-\exp(-\Omega(n))$, we have

$$\begin{aligned}\|\frac{1}{\lambda}\mathbf{W}_0\| \leq\ & 2\sqrt{nr}\sqrt{1-r+4rq(1-q)} \\ & + \max_{1\leq i\leq K} 2\sqrt{n_i r}\sqrt{2(1-r)+4r(q(1-q)+p_i(1-p_i))}+\epsilon'\sqrt{n}.\end{aligned}$$

Further, if $\max_{1\leq i\leq K} n_i = o(n)$. Then, for sufficiently large n, with the same probability,

$$\|\mathbf{W}_0\| \leq 2\lambda\sqrt{nr}\sqrt{1-r+4rq(1-q)}+\epsilon'\lambda\sqrt{n}.$$

Proof. $\frac{1}{\lambda}\mathbf{W}_0$ is a random matrix whose entries are i.i.d. and distributed as $\Delta(r, p_i)$ on $\mathcal{R}_{i,i}$ and $\Delta(r, q)$ on $\mathcal{R}^c$. Consequently, using Theorem 10 we obtain the result. ■

Lemma A.1.4 verifies that asymptotically with high probability we can make $\|\mathbf{W}_0\| < 1$ as long as λ is sufficiently small. However, $\mathbf{W}_0$ itself is not sufficient for construction of the desired $\mathbf{W}$, since we do not have any guarantee that $\mathbf{W}_0 \in \mathcal{M}_{\mathbf{U}}$. In order to achieve this, we will *correct* $\mathbf{W}_0$ by projecting it onto $\mathcal{M}_{\mathbf{U}}$. The following lemma suggests that $\mathbf{W}_0$ does not change much by such a correction.

Correcting the candidate $\mathbf{W}_0$

Lemma A.1.5. *$\mathbf{W}_0$ is as described previously in (A.1.19). Let $\mathbf{W}^H$ be the projection of $\mathbf{W}_0$ on $\mathcal{M}_{\mathbf{U}}$. Then*

- $\|\mathbf{W}^H\| \leq \|\mathbf{W}_0\|$
- *For any $\epsilon'' > 0$ (constant to be determined), with probability $1 - 6n^2 \exp(-2\epsilon''^2 n_{min})$ we have*

$$\|\mathbf{W}_0 - \mathbf{W}^H\|_\infty \leq 3\lambda\epsilon''.$$

Proof. Choose arbitrary vectors $\{\mathbf{u}_i\}_{i=K+1}^n$ to make $\{\mathbf{u}_i\}_{i=1}^n$ an orthonormal basis in $\mathbb{R}^n$. Call $\mathbf{U}_2 = [\mathbf{u}_{K+1} \ \dots \ \mathbf{u}_n]$ and $\mathbf{P} = \mathbf{U}\mathbf{U}^T$, $\mathbf{P}_2 = \mathbf{U}_2\mathbf{U}_2^T$. Now notice that for any matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$, $\mathbf{P}_2\mathbf{X}\mathbf{P}_2$ is in $\mathcal{M}_{\mathbf{U}}$ since $\mathbf{U}^T\mathbf{U}_2 = 0$. Let $\mathbf{I}$ denote the identity matrix. Then,

$$\begin{aligned}\mathbf{X} - \mathbf{P}_2\mathbf{X}\mathbf{P}_2 &= \mathbf{X} - (\mathbf{I} - \mathbf{P})\mathbf{X}(\mathbf{I} - \mathbf{P}) \\ &= \mathbf{P}\mathbf{X} + \mathbf{X}\mathbf{P} - \mathbf{P}\mathbf{X}\mathbf{P} \in \mathcal{M}_{\mathbf{U}}^{\perp}. \end{aligned} \tag{A.1.22}$$

Therefore, $\mathbf{P}_2\mathbf{X}\mathbf{P}_2$ is the orthogonal projection on $\mathcal{M}_{\mathbf{U}}$. Clearly,

$$\|\mathbf{W}^H\| = \|\mathbf{P}_2\mathbf{W}_0\mathbf{P}_2\| \leq \|\mathbf{P}_2\|^2\|\mathbf{W}_0\| \leq \|\mathbf{W}_0\|.$$

For analysis of $\|\mathbf{W}_0 - \mathbf{W}^H\|_\infty$ we can consider terms on the right hand side of (A.1.22) separately as we have:

$$\|\mathbf{W}_0 - \mathbf{W}^H\|_\infty \leq \|\mathbf{P}\mathbf{W}_0\|_\infty + \|\mathbf{W}_0\mathbf{P}\|_\infty + \|\mathbf{P}\mathbf{W}_0\mathbf{P}\|_\infty.$$

Clearly $\mathbf{P} = \sum_{i=1}^K \frac{1}{n_i}\mathbb{1}_{\mathbb{R}_{i,i}}^{n\times n}$. Then, each entry of $\frac{1}{\lambda}\mathbf{P}\mathbf{W}_0$ is either a summation of n_i i.i.d. $\Delta(r, p_i)$ or $\Delta(r, q)$ mean zero random variables scaled by n_i^{-1} for some $i \leq K$ or 0. Hence any $c, d \in [n]$ and $\epsilon'' > 0$

$$\mathbb{P}\left(|(\mathbf{P}\mathbf{W}_0)_{c,d}| \geq \lambda\epsilon''\right) \leq 2\exp(-2\epsilon''^2 n_{min}).$$

Same (or better) bounds holds for entries of $\mathbf{W}_0\mathbf{P}$ and $\mathbf{P}\mathbf{W}_0\mathbf{P}$. Then a union bound over all entries of the three matrices will give with probability $1-6n^2\exp(-2\epsilon''^2 n_{min})$, we have $\|\mathbf{W}_0 - \mathbf{W}^H\|_\infty \leq 3\lambda\epsilon''$. ∎

Recall that, $\gamma_{\text{succ}} := \max_{1\le i\le K} 2r\sqrt{n_i}\sqrt{2(\frac{1}{r}-1)+4\left(q(1-q)+p_i(1-p_i)\right)}$, and $\mathbf{\Lambda}_{\text{succ}}^{-1} := 2r\sqrt{n}\sqrt{\frac{1}{r}-1+4q(1-q)}+\gamma_{\text{succ}}$.

We can summarize our discussion so far in the following lemma,

Lemma A.1.6. *$\mathbf{W}_0$ is as described previously in (A.1.19). Choose $\mathbf{W}$ to be projection of $\mathbf{W}_0$ on $\mathcal{M}_{\mathbf{U}}$. Also suppose $\lambda \le (1-\delta)\mathbf{\Lambda}_{succ}$. Then, with probability $1-6n^2\exp(-\Omega(n_{min}))-4\exp(-\Omega(n))$ we have,*

- $\|\mathbf{W}\|<1$
- *For all feasible* $\mathbf{E}$, $f(\mathbf{E},\mathbf{W})\ge 0$.

Proof. To begin with, observe that $\mathbf{\Lambda}_{\text{succ}}^{-1}$ is $\Omega(\sqrt{n})$. Since $\lambda \le \Lambda_{\text{succ}}$, $\lambda\sqrt{n} = \mathcal{O}(1)$. Consequently, using $\lambda\Lambda_{\text{succ}}^{-1}<1$ and applying Lemma A.1.4, and choosing a sufficiently small $\epsilon'>0$, we conclude with,

$$\|\mathbf{W}\| \le \|\mathbf{W}_0\| < 1$$

with probability $1-\exp(-\Omega(n))$ where the constant in the exponent depends on the constant $\epsilon'>0$.

Next, from Lemma A.1.5 with probability $1-6n^2\exp(-\frac{2}{9}\epsilon''^2 n_{min})$ we have $\|\mathbf{W}_0-\mathbf{W}\|_\infty \le \lambda\epsilon''$. Then based on (A.2.10) for all $\mathbf{E}$, we have that,

$$\begin{aligned}
f(\mathbf{E},\mathbf{W}) &= f(\mathbf{E},\mathbf{W}_0)-\langle \mathbf{W}_0-\mathbf{W},\mathbf{E}\rangle \\
&\ge f(\mathbf{E},\mathbf{W}_0)-\lambda\epsilon''\left(\text{sum}(\mathbf{E}_{\mathcal{R}})-\text{sum}(\mathbf{E}_{\mathcal{R}^c})\right) \\
&= \lambda\left[(r(1-2q)-\epsilon'')\text{sum}(\mathbf{E}_{\mathcal{R}^c})\right] \\
&\quad -\lambda\sum_{i=1}^{K}\left[\left(r(2p_i-1)-\frac{1}{\lambda n_i}-\epsilon''\right)\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}})\right] \\
&\ge 0,
\end{aligned}$$

where we chose ϵ'' to be a sufficiently small constant. In particular, we set $\epsilon'' < \mathbf{D}_{\mathcal{A}}$, i.e., set $\epsilon'' < r(1-2q)$ and $\epsilon'' < r(2p_i-1)-\frac{1}{\lambda n_i}$ for all $1\le i\le K$.

Thus, by using a union bound $\mathbf{W}$ satisfies both of the desired conditions. ∎

Summary so far: Combining the last lemma with Lemma A.1.3, with high probability, either there exists a dual vector $\mathbf{W}^*$ which ensures $f(\mathbf{E},\mathbf{W}^*)>0$ or $\mathbf{E}\in\mathcal{M}_{\mathbf{U}}^{\perp}$. If former, we are done. Thus, we need to focus on the latter case and show that for all perturbations $\mathbf{E}\in\mathcal{M}_{\mathbf{U}}^{\perp}$, the objective will strictly increase at $(\mathbf{L}^0,\mathbf{S}^0)$ with high probability.

A.1.4 Solving for $\mathbf{E}^L \in \mathcal{M}_{\mathbf{U}}^{\perp}$ case

Recall that,

$$g\left(\mathbf{E}\right) = \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{E}_{R_{i,i}}) + \lambda \left(\text{sum}(\mathbf{E}_{\mathcal{A}_0}) - \text{sum}(\mathbf{E}_{\mathcal{A}_1})\right)$$

. Let us define,

$$g_1(\mathbf{X}) := \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{X}_{\mathcal{R}_{i,i}}),$$

$$g_2(\mathbf{X}) := \text{sum}(\mathbf{X}_{\mathcal{A}_0}) - \text{sum}(\mathbf{X}_{\mathcal{A}_1}),$$

so that, $g\left(\mathbf{X}\right) = g_1(\mathbf{X}) + \lambda g_2(\mathbf{X})$. Also let $\mathbf{V} = [\mathbf{v}_1 \ \ldots \ \mathbf{v}_K]$ where $\mathbf{v}_i = \sqrt{n_i}\mathbf{u}_i$. Thus, $\mathbf{V}$ is basically obtained by, normalizing columns of $\mathbf{U}$ to make its nonzero entries 1. Assume $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$. Then, by definition of $\mathcal{M}_{\mathbf{U}}^{\perp}$, we can write,

$$\mathbf{E} = \mathbf{V}\mathbf{M}^T + \mathbf{N}\mathbf{V}^T.$$

Let $\mathbf{m}_i, \mathbf{n}_i$ denote i'th columns of $\mathbf{M}, \mathbf{N}$ respectively. From $\mathbf{L}^0$ and the objective function it follows that

$$\mathbf{E}_{\mathcal{R}^c} \text{ is (entrywise) nonnegative}$$

$$\mathbf{E}_{\mathcal{R}} \text{ is (entrywise) nonpositive.}$$

Now, we list some simple observations regarding structure of $\mathbf{E}$. We can write

$$\mathbf{E} = \sum_{i=1}^{K} (\mathbf{v}_i \mathbf{m}_i^T + \mathbf{n}_i \mathbf{v}_i^T) = \sum_{i=1}^{K+1} \sum_{j=1}^{K+1} \mathbf{E}_{\mathcal{R}_{i,j}}. \tag{A.1.23}$$

Notice that only two components : $\mathbf{v}_i \mathbf{m}_i^T$ and $\mathbf{n}_j \mathbf{v}_j^T$, contribute to the term $\mathbf{E}_{\mathcal{R}_{i,j}}$.

Let $\mathbf{E}^{i,j} \in \mathbb{R}^{n_i \times n_j}$, which is $\mathbf{E}$ induced by entries on $\mathcal{R}_{i,j}$. Basically, $\mathbf{E}^{i,j}$ is same as $\mathbf{E}_{\mathcal{R}_{i,j}}$ when we get rid of trivial zero rows and zero columns. Then

$$\mathbf{E}^{i,j} = \mathbb{1}^{n_i} (\mathbf{m}_i^{\mathcal{C}_j})^T + \mathbf{n}_j^{\mathcal{C}_i} \mathbb{1}^{n_j T}, \tag{A.1.24}$$

where $\mathbf{m}_i^{\mathcal{C}_j}$ is the vector corresponding to the entries of $\mathcal{C}_j$ in $\mathbf{m}_i$. Similarly, $\mathbf{n}_j^{\mathcal{C}_i}$ is the vector corresponding to the entries of $\mathcal{C}_i$ in $\mathbf{n}_j$.

Clearly, given $\{\mathbf{E}^{i,j}\}_{1 \le i,j \le n}$, $\mathbf{E}$ is uniquely determined. Now, assume we fix $\text{sum}(\mathbf{E}^{i,j})$ for all i, j and we would like to find the *worst* $\mathbf{E}$ subject to these constraints. Vari-

ables in such an optimization are $\mathbf{m}_i, \mathbf{n}_i$. Basically we are interested in,

$$\min g(\mathbf{E}) \tag{A.1.25}$$

subject to

$$\text{sum}(\mathbf{E}^{i,j}) = c_{i,j} \text{ for all } i, j$$

$$\mathbf{E}^{i,j} \begin{cases} \text{nonnegative if } i \neq j \\ \text{nonpositive if } i = j \end{cases}, \tag{A.1.26}$$

where $\{c_{i,j}\}$ are constants. Constraint (A.1.26) follows from (A.2.10).
Remark: For the special case of $i = j = K + 1$, notice that $\mathbf{E}^{i,j} = 0$.

In (A.1.25), $g_1(\mathbf{E})$ is fixed and is equal to $\sum_{i=1}^{K} \frac{1}{n_i} c_{i,i}$. Consequently, we just need to do the optimization with the objective $g_2(\mathbf{E}) = \text{sum}(\mathbf{E}_{\mathcal{A}_0}) - \text{sum}(\mathbf{E}_{\mathcal{A}_1})$.

Let $\beta_{i,j} \subseteq [n_i] \times [n_j]$ be a set of coordinates defined as follows. For any $(c, d) \in [n_i] \times [n_j]$

$$(c, d) \in \beta_{i,j} \text{ iff } (a_{i,c}, a_{j,d}) \in \mathcal{A}$$

For $(i_1, j_1) \neq (i_2, j_2)$, $(\mathbf{m}_{i_1}^{c_{j_1}}, \mathbf{n}_{j_1}^{c_{i_1}})$ and $(\mathbf{m}_{i_2}^{c_{j_2}}, \mathbf{n}_{j_2}^{c_{i_2}})$ are independent variables. Consequently, due to (A.1.24), we can partition problem (A.1.25) into the following smaller disjoint problems.

$$\min_{\mathbf{m}_i^j, \mathbf{n}_j^i} \text{ sum}(\mathbf{E}_{\beta_{i,j}^c}^{i,j}) - \text{sum}(\mathbf{E}_{\beta_{i,j}}^{i,j}) \tag{A.1.27}$$

subject to

$$\text{sum}(\mathbf{E}^{i,j}) = c_{i,j}$$

$$\mathbf{E}^{i,j} \text{ is } \begin{cases} \text{nonnegative if } i \neq j \\ \text{nonpositive if } i = j \end{cases}$$

Then, we can solve these problems locally (for each i, j) to finally obtain,

$$g_2(\mathbf{E}^{L,*}) = \sum_{i,j} \text{sum}(\mathbf{E}_{\beta_{i,j}^c}^{i,j,*}) - \sum_{i,j} \text{sum}(\mathbf{E}_{\beta_{i,j}}^{i,j,*})$$

to find the overall result of problem (A.1.25), where $*$ denotes the optimal solutions in problems (A.1.25) and (A.1.27). The following lemma will be useful for analysis of these local optimizations.

Lemma A.1.7. *Let* $\mathbf{a} \in \mathbb{R}^c$, $\mathbf{b} \in \mathbb{R}^d$ *and* $X = \mathbb{1}^c \mathbf{b}^T + \mathbf{a} \mathbb{1}^{d^T}$ *be variables and* $C_0 \geq 0$ *be a constant. Also let* $\beta \subseteq [c] \times [d]$. *Consider the following optimization*

problem

$$\min_{\mathbf{a},\mathbf{b}} \ \textit{sum}(\mathbf{X}_\beta)$$
$$\textit{subject to}$$
$$\mathbf{X}_{i,j} \geq 0 \ \textit{for all } i,j$$
$$\textit{sum}(\mathbf{X}) = C_0$$

For this problem there exists a (entrywise) nonnegative minimizer $(\mathbf{a}^0, \mathbf{b}^0)$.

Proof. Let x_i denotes i'th entry of vector $\mathbf{x}$. Assume $(\mathbf{a}^*, \mathbf{b}^*)$ is a minimizer. Without loss of generality assume $b_1^* = \min_{i,j}\{\mathbf{a}_i^*, \mathbf{b}_j^*\}$. If $b_1^* \geq 0$ we are done. Otherwise, since $\mathbf{X}_{i,j} \geq 0$ we have $a_i^* \geq -b_1^*$ for all $i \leq c$. Then set $\mathbf{a}^0 = \mathbf{a}^* + \mathbb{1}^c b_1^*$ and $\mathbf{b}^0 = \mathbf{b}^* - \mathbb{1}^d b_1^*$. Clearly, $(\mathbf{a}^0, \mathbf{b}^0)$ is nonnegative. On the other hand, we have:

$$\mathbf{X}^* = \mathbb{1}^c \mathbf{b}^{*T} + \mathbf{a}^* \mathbb{1}^{dT} = \mathbb{1}^c \mathbf{b}^{0T} + \mathbf{a}^0 \mathbb{1}^{dT} = \mathbf{X}^0,$$

which implies,

$$\text{sum}(\mathbf{X}_\beta^*) = \text{sum}(\mathbf{X}_\beta^0) = \text{optimal value}$$

■

The follwing lemma is a direct consequence of Lemma A.1.7 and follows from the structure of $\mathbf{E}^{i,j}$ given in (A.1.24) and (A.2.10):

Lemma A.1.8. *In the local optimizations* (A.1.27)*, without loss of generality, we can assume that* $(\mathbf{m}_i^{\mathcal{C}_j}, \mathbf{n}_j^{\mathcal{C}_i})$ *is entrywise nonnegative whenever* $i \neq j$ *and entrywise nonpositive whenever* $i = j$.

The following lemma will help us characterize the relationship between $\text{sum}(\mathbf{E}^{i,j})$ and $\text{sum}(\mathbf{E}^{i,j}_{\beta^c_{i,j}})$.

Lemma A.1.9. *Let* β *be a random set generated by choosing elements of* $[c] \times [d]$ *independently with probability* $0 \leq \eta \leq 1$. *Then for any* $\epsilon' > 0$ *with probability* $1 - d\exp(-2\epsilon'^2 c)$ *for all nonzero and entrywise nonnegative* $\mathbf{a} \in \mathbb{R}^d$ *we'll have:*

$$\textit{sum}(\mathbf{X}_\beta) > (\eta - \epsilon')\textit{sum}(\mathbf{X}), \tag{A.1.28}$$

where $\mathbf{X} = \mathbb{1}^c \mathbf{a}^T$. *Similarly, with the same probability, for all such* $\mathbf{a}$*, we'll have* $\textit{sum}(\mathbf{X}_\beta) < (\eta + \epsilon')\textit{sum}(\mathbf{X})$.

Proof. We'll only prove the first statement (A.1.28) as the proofs are identical. For each $i \leq d$, a_i occurs exactly c times in $\mathbf{X}$ as i'th column of X is $\mathbb{1}^c a_i$. By using a Chernoff bound, we can estimate the number of coordinates of i'th column which are element of β (call this number C_i) as we can view this number as a sum of c i.i.d. Bernoulli(η) random variables. Then

$$\mathbb{P}(C_i \leq c(\eta - \epsilon')) \leq \exp(-2\epsilon'^2 c).$$

Now, we can use a union bound over all columns to make sure for all i, $C_i > c(\eta - \epsilon')$

$$\mathbb{P}(C_i > c(r - \epsilon') \text{ for all } i \leq d) \geq 1 - d\exp(-2\epsilon'^2 c)$$

On the other hand if each $C_i > c(\eta - \epsilon')$ then for any nonnegative $\mathbf{a} \neq 0$,

$$\text{sum}(\mathbf{X}_\beta) = \sum_{(i,j)\in\beta} \mathbf{X}_{i,j} = \sum_{i=}^{d} C_i a_i > c(\eta - \epsilon') \sum_{i=1}^{d} a_i = (\eta - \epsilon')\text{sum}(\mathbf{X}).$$

■

Using Lemma A.1.9, we can calculate a lower bound for $g(\mathbf{E})$ with high probability as long as the cluster sizes are sufficiently large. Due to (A.1.23) and the linearity of $g(\mathbf{E})$, we can focus on contributions due to specific clusters i.e. $\mathbf{v}_i\mathbf{m}_i^T + \mathbf{n}_i\mathbf{v}_i^T$ for the i'th cluster. We additionally know the simple structure of $\mathbf{m}_i, \mathbf{n}_i$ from Lemma A.1.8. In particular, subvectors $\mathbf{m}_i^{\mathcal{C}_i}$ and $\mathbf{n}_i^{\mathcal{C}_i}$ of $\mathbf{m}_i, \mathbf{n}_i$ can be assumed to be nonpositive and rest of the entries are nonnegative.

Lemma A.1.10. *Assume,* $1 \leq l \leq K$, $\mathbf{D}_{\mathcal{A}} > 0$. *Then, with probability* $1 - n\exp(-2\mathbf{D}_{\mathcal{A}}^2(n_l - 1))$, *we have* $g(\mathbf{v}_l\mathbf{m}_l^T) \geq 0$ *for all* $\mathbf{m}_l$. *Also, if* $\mathbf{m}_l \neq 0$ *then inequality is strict.*

Proof. Recall that $\mathbf{m}_l$ satisfies $\mathbf{m}_l^{\mathcal{C}_i}$ is nonpositive/nonnegative when $i = l / i \neq l$ for all i. Define $\mathbf{X}^i := \mathbb{1}^{n_l}\mathbf{m}_l^{\mathcal{C}_i T}$. We can write

$$g(\mathbf{v}_l\mathbf{m}_l^T) = \frac{1}{n_l}\text{sum}(\mathbf{X}^l) + \sum_{i=1}^{K} \lambda h(\mathbf{X}^i, \beta_{l,i}^c),$$

where $h(\mathbf{X}^i, \beta_{l,i}^c) = \text{sum}(\mathbf{X}^i_{\beta_{l,i}^c}) - \text{sum}(\mathbf{X}^i_{\beta_{l,i}})$. Now assume $i \neq l$. Using Lemma A.1.9 and the fact that $\beta_{l,i}$ is a randomly generated subset (with parameter q), with probability $1 - n_i\exp(-2\epsilon'^2 n_l)$, for all $\mathbf{X}^i$, we have,

$$\begin{aligned} h(\mathbf{X}^i, \beta_{l,i}^c) &\geq (r(1-q) - \epsilon')\text{sum}(\mathbf{X}^i) - (rq + \epsilon')\text{sum}(\mathbf{X}^i) \\ &= (r(1-2q) - 2\epsilon')\text{sum}(\mathbf{X}^i), \end{aligned}$$

where inequality is strict if $X^i \neq 0$. Similarly, when $i = l$ with probability at least $1 - n_l \exp(-2\epsilon'^2(n_l - 1))$, we have,

$$\frac{1}{\lambda n_l}\text{sum}(\mathbf{X}^l) + h(\mathbf{X}^l, \beta^c_{l,l}) \geq \left(r(1-p_l) + \epsilon' + \frac{1}{\lambda n_l}\right)\text{sum}(\mathbf{X}^l) - (rp_l - \epsilon')\,\text{sum}(\mathbf{X}^l)$$
$$= -\left(r(2p_l - 1) - \frac{1}{\lambda n_l} - 2\epsilon'\right)\text{sum}(\mathbf{X}^l).$$

Choosing $\epsilon' = \frac{\mathbf{D}_{\mathcal{A}}}{2}$ and using the facts that $r(1-2q) - 2\mathbf{D}_{\mathcal{A}} \geq 0$, $r(2p_l - 1) - \frac{1}{\lambda n_l} - 2\mathbf{D}_{\mathcal{A}} \geq 0$ and using a union bound, with probability $1 - n\exp(-2\mathbf{D}^2_{\mathcal{A}}(n_l - 1))$, we have $g(\mathbf{v}_l\mathbf{m}_l^T) \geq 0$ and the inequality is strict when $\mathbf{m}_l \neq 0$ as at least one of the $\mathbf{X}^i$'s will be nonzero. ■

The following lemma immediately follows from Lemma A.1.10 and summarizes the main result of the section.

Lemma A.1.11. *Let $\mathbf{D}_{\mathcal{A}}$ be as defined in (A.1.2) and assume $\mathbf{D}_{\mathcal{A}} > 0$. Then with probability $1 - 2nK\exp(-2\mathbf{D}^2_{\mathcal{A}}(n_{min} - 1))$ we have $g(\mathbf{E}^L) > 0$ for all nonzero feasible $\mathbf{E}^L \in \mathcal{M}^{\perp}_U$.*

A.1.5 The Final Step

Lemma A.1.12. *Let $p_{min} > \frac{1}{2} > q$ and $\mathcal{G}$ be a random graph generated according to Model 2.2.1 and 2.2.2 with cluster sizes $\{n_i\}_{i=1}^K$. If $\lambda \leq (1-\epsilon)\mathbf{\Lambda}_{succ}$ and $\mathbf{D}_{\min} = \min_{1\leq i\leq n} r\,(2p_i - 1)\,n_i \geq (1+\epsilon)\frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to Program 2.1.4 with probability $1 - \exp(-\Omega(n)) - 6n^2\exp(-\Omega(n_{min}))$.*

Proof. Based on Lemma A.1.6 and Lemma A.1.11,
with probability $1 - cn^2\exp(-C\left(\min\{r(1-2q), r(2p_{min}-1)\}\right)^2 n_{min})$,

- There exists $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$ such that for all feasible $\mathbf{E}^L$, $f(\mathbf{E}^L, \mathbf{W}) \geq 0$.
- For all nonzero $\mathbf{E}^L \in \mathcal{M}^{\perp}_{\mathbf{U}}$ we have $g(\mathbf{E}^L) > 0$.

Consequently based on Lemma A.1.3, $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal of Problem 2.1.4. ■

A.2 Proof of Results for Improved Convex Program (Theorem 2)

This section will show that, the optimal solution of Problem 2.1.7 is the pair $(\mathbf{L}^0, \mathbf{S}^0)$ under reasonable conditions, where,

$$\mathbf{L}^0 = \mathbb{1}_{\mathcal{R}}^{n\times n},\ \mathbf{S}^0 = \mathbf{S}_{obs}^0 = \mathbb{1}_{\mathcal{R}\cap\mathcal{A}_0}^{n\times n}. \tag{A.2.1}$$

Also denote the true optimal pair by $(\mathbf{L}^*, \mathbf{S}^*)$. Let $1 \geq p_{min} > q > 0$ and $0 \leq r \leq 1$. $\mathcal{G}$ be a random graph generated according to the stochastic block model 2.2.1 with cluster sizes $\{n_i\}_{i=1}^K$. Let the observation model be as defined in (2.2.2). Theorem 2 is based on the following lemma:

Lemma A.2.1. *If $\lambda < \tilde{\mathbf{\Lambda}}_{succ}$ and $\tilde{\mathbf{D}}_{\min} > \frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to Program 2.1.7 with high probability.*

Given $q, \{p_i\}_{i=1}^K$, define the following parameter which will be useful for the subsequent analysis. This parameter can be seen as a measure of distinctness of the "worst" cluster from the "background noise". Here, by background noise we mean the edges over $\mathcal{R}^c$.

$$\begin{aligned}\tilde{\mathbf{D}}_{\mathcal{A}} &= \frac{1}{2}\min\left\{r(1-q), \left\{r(p_i - q) - \frac{1}{\lambda n_i}\right\}_{i=1}^K\right\} \\ &= \frac{1}{2}\min\left\{r(1-q), \frac{\tilde{\mathbf{D}}_i - \lambda^{-1}}{n_i}\right\}\end{aligned} \tag{A.2.2}$$

A.2.1 Perturbation Analysis

Our aim is to show that $(\mathbf{L}^0, \mathbf{S}^0)$ defined in (A.2.1) is unique optimal solution to Problem 2.1.7.

Lemma A.2.2. *Let $(\mathbf{E}^L, \mathbf{E}^S)$ be a feasible perturbation. Then, the objective will increase by at least,*

$$f(\mathbf{E}^L, \mathbf{W}) = \sum_{i=1}^K \frac{1}{n_i} sum(\mathbf{E}^L_{\mathcal{R}_{i,i}}) + \langle \mathbf{E}^L, \mathbf{W}\rangle + \lambda sum(\mathbf{E}^L_{\mathcal{A}_0}) \tag{A.2.3}$$

for any $\mathbf{W} \in \mathcal{M}$, $\|\mathbf{W}\| \leq 1$.

Proof. From constraint (2.1.9), we have $\mathbf{L}_{i,j} = \mathbf{S}_{i,j}$ whenever $\mathbf{A}_{i,j}^{obs} = 0$. So, $\mathbf{L}^*_{\mathcal{A}_0} = \mathbf{S}^*_{\mathcal{A}_0}$. Further, recall that $\mathbf{S}$ can be split as $\mathbf{S} = \mathbf{S}_{obs} + \mathbf{S}_{rest}$, where $\mathbf{S}_{rest}$ denotes the entries of $\mathbf{S}$ other than those corresponding to the observed entries of

$\mathbf{A}$. Furthermore, we claim that at the optimal, $\mathbf{S}_{rest} = 0$, since if otherwise, the objective can be strictly decreased by setting $\mathbf{S}_{rest} = 0$. So, without loss of generality,

$$\mathbf{S}^* = \mathbf{L}^*_{\mathcal{A}_0}. \tag{A.2.4}$$

Recall that,

$$\begin{aligned}
&\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda \|\mathbf{S}^0 + \mathbf{E}^S\|_1 - (\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1) \\
&\geq \langle \partial\|\mathbf{L}^0\|_\star, \mathbf{E}^L\rangle + \lambda\langle \partial\|\mathbf{S}^0\|_1, \mathbf{E}^S\rangle \\
&= \langle \mathbf{U}\mathbf{U}^T + \mathbf{W}, \mathbf{E}^L\rangle + \lambda\langle \text{sign}(\mathbf{S}^0) + \mathbf{Q}, \mathbf{E}^S\rangle
\end{aligned}$$

Using $\text{sign}(\mathbf{S}^0) = \mathbb{1}^{n\times n}_{\mathcal{A}_0\cap\mathcal{R}}$, and choosing $\mathbf{Q} = \mathbb{1}^{n\times n}_{\mathcal{A}_0-(\mathcal{A}_0\cap\mathcal{R})}$, we get,

$$\begin{aligned}
&\|\mathbf{L}^0 + \mathbf{E}^L\|_\star + \lambda \|\mathbf{S}^0 + \mathbf{E}^S\|_1 - (\|\mathbf{L}^0\|_\star + \lambda \|\mathbf{S}^0\|_1) \\
&\geq \left\langle \mathbf{W}, \mathbf{E}^L\right\rangle + \underbrace{\sum_{i=1}^{K} \frac{1}{n_i}\text{sum}(\mathbf{E}^L_{R_{i,i}}) + \lambda\left(\text{sum}(\mathbf{E}^L_{\mathcal{A}_0})\right)}_{:=g(\mathbf{E}^L)}
\end{aligned} \tag{A.2.5}$$

for any $\mathbf{W} \in \mathcal{M}$. ■

From this point onward, for simplicity we will ignore the superscript L on $\mathbf{E}^L$ and just use $\mathbf{E}$.

Define,

$$g(\mathbf{E}) := \sum_{i=1}^{K} \frac{1}{n_i}\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + \lambda\text{sum}(\mathbf{E}_{\mathcal{A}_0})). \tag{A.2.6}$$

Also, define $f\left(\mathbf{E}, \mathbf{W}\right) := g\left(\mathbf{E}\right) + \langle \mathbf{W}, \mathbf{E}\rangle$. Our aim is to show that for all feasible perturbations $\mathbf{E}$, there exists $\mathbf{W}$ such that,

$$f\left(\mathbf{E}, \mathbf{W}\right) = g(\mathbf{E}) + \langle \mathbf{W}, \mathbf{E}\rangle > 0. \tag{A.2.7}$$

Note that $g(\mathbf{E})$ does not depend on $\mathbf{W}$.

We can directly use Lemma A.1.3. So, as in the previous section, we have broken down our aim into two steps.

1. Construct $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$, such that $f(\mathbf{E}, \mathbf{W}) \geq 0$ for all feasible perturbations $\mathbf{E}$.

2. For all non-zero feasible $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$, show that $g(\mathbf{E}) > 0$.

As a first step, in Section A.2.2, we will argue that, under certain conditions, there exists a $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$ such that with high probability, $f(\mathbf{E}, \mathbf{W}) \geq 0$ for all feasible $\mathbf{E}$. Recall that such a $\mathbf{W}$ is called the dual certificate. Secondly, in Section A.2.3, we will show that, under certain conditions, for all $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$ with high probability, $g(\mathbf{E}) > 0$. Finally, combining these two arguments, and using Lemma A.1.3 we will conclude that $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal with high probability.

A.2.2 Showing existence of the dual certificate

Recall that

$$f(\mathbf{E}, \mathbf{W}) = \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + \langle \mathbf{E}, \mathbf{W} \rangle + \lambda \text{sum}\left(\mathbf{E}_{\mathcal{A}_0}\right).$$

$\mathbf{W}$ will be constructed from the candidate $\mathbf{W}_0$, which is given as follows.

Candidate $\mathbf{W}_0$

Based on Program 2.1.7, we propose the following:

$$\mathbf{W}_0 = \sum_{i=1}^{K} c_i \mathbb{1}_{\mathcal{R}_{i,i}}^{n \times n} + c\mathbb{1}^{n \times n} - \lambda \mathbb{1}_{\mathcal{A}_0}^{n \times n}, \tag{A.2.8}$$

where $\{c_i\}_{i=1}^{K}, c$ are real numbers to be determined.

$$f(\mathbf{E}, \mathbf{W}^0) = \sum_{i=1}^{K} (\frac{1}{n_i} + c_i) \text{ sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) + c \text{ sum}(\mathbf{E}).$$

Note that $\mathbf{W}_0$ is a random matrix where randomness is due to $\mathbf{A}_{obs}$. In order to ensure a small spectral norm, we will set its expectation to 0, i.e., we will choose $c, \{c_i\}'s$ to ensure that $\mathbb{E}[\mathbf{W}_0] = 0$.

Following from the partially observed Stochastic Block Model (Definition 2.2.1 and Definition 2.2.2), the expectation of an entry of $\mathbf{W}_0$ on $\mathcal{R}_{i,i}$ (region corresponding to cluster i) and $\mathcal{R}^c$ (region outside the clusters) is $c_i + \lambda r(p_i - q)$ and $c + \lambda r(q - 1)$ respectively. Therefore, we set,

$$c_i = -\lambda r(p_i - q) \quad \text{and} \quad c = \lambda r(1 - q).$$

With these, choices, the candidate $\mathbf{W}_0$ and $f(\mathbf{E}, \mathbf{W}_0)$ take the following forms:

$$\begin{aligned}\mathbf{W}_0 &= \lambda\left[\sum_{i=1}^{K} -(1-r(1-p_i))\ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_0} + r(1-p)\ \left(\mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_1} + \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\Gamma^{out}}\right)\right] \\ &\quad +\lambda\left[-(1-r(1-q))\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_0} + r(1-q)\ \left(\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_1} + \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\Gamma^{out}}\right)\right] \quad \text{(A.2.9)}\end{aligned}$$

$$f(\mathbf{E}, \mathbf{W}_0) = \lambda\left[r(1-q)\ \text{sum}(\mathbf{E})\right] - \lambda\left[\sum_{i=1}^{K}\left(r(p_i-q) - \frac{1}{\lambda n_i}\right)\ \text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}})\right].$$

From $\mathbf{L}^0$ and the constraint $1 \geq \mathbf{L}_{i,j} \geq 0$, it follows that,

$$\begin{aligned}&\mathbf{E}_{\mathcal{R}^c} \text{ is (entrywise) nonnegative.} \quad \text{(A.2.10)}\\ &\mathbf{E}_{\mathcal{R}} \text{ is (entrywise) nonpositive.}\end{aligned}$$

Thus, $\text{sum}(\mathbf{E}_{\mathcal{R}^c}) \leq 0$ and $\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}}) \geq 0$. When $\lambda r(p_i - q) - \frac{1}{n_i} \geq 0$ and $\lambda(1-q) \geq 0$; we will have $f(\mathbf{E}, \mathbf{W}_0) \geq 0$ for all feasible $\mathbf{E}$. This indeed holds due to the assumptions of Theorem 2 (see (A.2.2)), as we assumed $r(p_i - q) > \frac{1}{\lambda n_i}$ for $i = 1, 2\cdots, K$ and $1 > q$.

Using the same technique as in Theorem 10, we can bound the spectral norm of $\mathbf{W}^0$ as follows.

Lemma A.2.3. *Recall that, $\mathbf{W}_0$ is a random matrix; where randomness is on the partially observed stochastic block model $\mathbf{A}_{obs}$ and it is given by,*

$$\begin{aligned}\mathbf{W}_0 = \lambda\left[\sum_{i=1}^{K} -(1-r(1-p_i))\ \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_0} + r(1-p)\ \left(\mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\mathcal{A}_1} + \mathbb{1}^{n\times n}_{\mathcal{R}_{i,i}\cap\Gamma^{out}}\right)\right] \\ +\lambda\left[-(1-r(1-q))\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_0} + r(1-q)\ \left(\mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\mathcal{A}_1} + \mathbb{1}^{n\times n}_{\mathcal{R}^c\cap\Gamma^{out}}\right)\right]\end{aligned}$$

Then, for any $\epsilon' > 0$, with probability $1 - \exp(-\Omega(n))$, we have

$$\begin{aligned}\|\frac{1}{\lambda}\mathbf{W}_0\| &\leq 2\sqrt{nr}\sqrt{(1-q)(1-r+rq)} \\ &\quad + \max_{1\leq i\leq K} 2\sqrt{n_i r}\sqrt{(1-p_i)(1-r+rp_i) + (1-q)(1-r+rq)} + \epsilon'\sqrt{n}\end{aligned}$$

Further, if $\max_{1\leq i\leq K} n_i = o(n)$. Then, for sufficiently large n, with the same probability,

$$\|\mathbf{W}_0\| \leq 2\lambda\sqrt{nr}\sqrt{(1-q)(1-r+rq)} + \epsilon'\lambda\sqrt{n}.$$

Lemma A.2.3 verifies that asymptotically with high probability we can make $\|\mathbf{W}_0\| < 1$ as long as λ is sufficiently small. However, $\mathbf{W}_0$ itself is not sufficient for construction of the desired $\mathbf{W}$, since we do not have any guarantee that $\mathbf{W}_0 \in \mathcal{M}_{\mathbf{U}}$. In order to achieve this, we will *correct* $\mathbf{W}_0$ by projecting it onto $\mathcal{M}_{\mathbf{U}}$. Lemma A.1.5 can be used to here.

Recall that, $\tilde{\gamma}_{\text{succ}} := 2 \max_{1 \le i \le K} r\sqrt{n_i}\sqrt{(1-p_i)(\frac{1}{r}-1+p_i)+(1-q)(\frac{1}{r}-1+q)}$ and $\tilde{\mathbf{\Lambda}}^{-1}_{\text{succ}} := 2r\sqrt{n}\sqrt{(\frac{1}{r}-1+q)(1-q)} + \tilde{\gamma}_{\text{succ}}$.

We can summarize our discussion so far in the following lemma:

Lemma A.2.4. *$\mathbf{W}_0$ is as described previously in (A.2.9). Choose $\mathbf{W}$ to be projection of $\mathbf{W}_0$ on $\mathcal{M}_{\mathbf{U}}$. Also suppose $\lambda \le (1-\delta)\tilde{\mathbf{\Lambda}}_{succ}$. Then, with probability $1 - 6n^2\exp(-\Omega(n_{min})) - 4\exp(-\Omega(n))$ we have,*

- $\|\mathbf{W}\| < 1$
- *For all feasible* $\mathbf{E}$, $f(\mathbf{E}, \mathbf{W}) \ge 0$.

Proof. To begin with, observe that $\tilde{\mathbf{\Lambda}}^{-1}_{\text{succ}}$ is $\Omega(\sqrt{n})$. Since $\lambda \le \tilde{\mathbf{\Lambda}}_{\text{succ}}$, $\lambda\sqrt{n} = \mathcal{O}(1)$. Consequently, using $\lambda\tilde{\mathbf{\Lambda}}^{-1}_{\text{succ}} < 1$ and applying Lemma A.2.3, and choosing a sufficiently small $\epsilon' > 0$, we conclude with,

$$\|\mathbf{W}\| \le \|\mathbf{W}_0\| < 1$$

with probability $1 - \exp(-\Omega(n))$ where the constant in the exponent depends on the constant $\epsilon' > 0$.

Next, from Lemma A.1.5 with probability $1 - 6n^2\exp(-\frac{2}{9}\epsilon''^2 n_{min})$ we have $\|\mathbf{W}_0 - \mathbf{W}\|_\infty \le \lambda\epsilon''$. Then based on (A.2.10) for all $\mathbf{E}$, we have that,

$$\begin{aligned}
f(\mathbf{E}, \mathbf{W}) &= f(\mathbf{E}, \mathbf{W}_0) - \langle \mathbf{W}_0 - \mathbf{W}, \mathbf{E} \rangle \\
&\ge f(\mathbf{E}, \mathbf{W}_0) - \lambda\epsilon'' \left(\text{sum}(\mathbf{E}_{\mathcal{R}}) - \text{sum}(\mathbf{E}_{\mathcal{R}^c})\right) \\
&= \lambda\left[(r(1-q) - \epsilon'')\text{sum}(\mathbf{E}_{\mathcal{R}^c})\right] \\
&\quad -\lambda\sum_{i=1}^{K}\left[\left(r(p_i - q) - \frac{1}{\lambda n_i} - \epsilon''\right)\text{sum}(\mathbf{E}_{\mathcal{R}_{i,i}})\right] \\
&\ge 0,
\end{aligned}$$

where we chose ϵ'' to be a sufficiently small constant. In particular, we set $\epsilon'' < \tilde{\mathbf{D}}_{\mathcal{A}}$, i.e., set $\epsilon'' < r(1-q)$ and $\epsilon'' < r(p_i - q) - \frac{1}{\lambda n_i}$ for all $1 \le i \le K$.

Thus, by using a union bound $\mathbf{W}$ satisfies both of the desired conditions. ∎

Summary so far: Combining the last lemma with Lemma A.1.3, with high probability, either there exists a dual vector $\mathbf{W}^*$ which ensures $f(\mathbf{E}, \mathbf{W}^*) > 0$ or $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$. If the former, we are done. Therefore, we need to focus on the latter case and show that for all perturbations $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$, the objective will strictly increase at $(\mathbf{L}^0, \mathbf{S}^0)$ with high probability.

A.2.3 Solving for $\mathbf{E}^L \in \mathcal{M}_{\mathbf{U}}^{\perp}$ case

Recall that,

$$g(\mathbf{E}) = \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{E}_{R_{i,i}}) + \lambda \text{sum}(\mathbf{E}_{\mathcal{A}_0}).$$

Let us define,

$$g_1(\mathbf{X}) := \sum_{i=1}^{K} \frac{1}{n_i} \text{sum}(\mathbf{X}_{\mathcal{R}_{i,i}}),$$

$$g_2(\mathbf{X}) := \text{sum}(\mathbf{X}_{\mathcal{A}_0}),$$

so that, $g(\mathbf{X}) = g_1(\mathbf{X}) + \lambda g_2(\mathbf{X})$. Also let $\mathbf{V} = [\mathbf{v}_1 \ \dots \ \mathbf{v}_K]$ where $\mathbf{v}_i = \sqrt{n_i}\mathbf{u}_i$. Thus, $\mathbf{V}$ is basically obtained by, normalizing columns of $\mathbf{U}$ to make its nonzero entries 1. Assume $\mathbf{E} \in \mathcal{M}_{\mathbf{U}}^{\perp}$. Then, by definition of $\mathcal{M}_{\mathbf{U}}^{\perp}$, we can write,

$$\mathbf{E} = \mathbf{V}\mathbf{M}^T + \mathbf{N}\mathbf{V}^T.$$

Let $\mathbf{m}_i, \mathbf{n}_i$ denote i'th columns of $\mathbf{M}, \mathbf{N}$ respectively.

Again as in previous section A.1.4, we consider optimization problem A.1.25. Since $g_1(\mathbf{E})$ is fixed, we just need to optimize over $g_2(\mathbf{E})$. This optimization can be reduced to local optimizations A.1.27. Since $\mathbf{L}^0 = \mathbb{1}_{\mathcal{R}}^{n \times n}$ and the condition (2.1.9),

$$\mathbf{E}_{\mathcal{R}^c} \text{ is (entrywise) nonnegative}$$

$$\mathbf{E}_{\mathcal{R}} \text{ is (entrywise) nonpositive.}$$

We can make use of Lemma A.1.8 and assume $\mathbf{m}_l^{\mathcal{C}_i}$ is nonpositive/nonnegative when $i = l/i \neq l$ for all i. Thus, using Lemma A.1.28 we lower bound $g(\mathbf{v}_l \mathbf{m}_l^T)$ as described in the following section.

Lower bounding $g(\mathbf{E})$

Lemma A.2.5. *Assume,* $1 \leq l \leq K$, $\tilde{\mathbf{D}}_{\mathcal{A}} > 0$. *Then, with probability* $1 - n\exp(-2\tilde{\mathbf{D}}_{\mathcal{A}}^2(n_l - 1))$, *we have* $g(\mathbf{v}_l \mathbf{m}_l^T) \geq \lambda(1 - q - \tilde{\mathbf{D}}_{\mathcal{A}}) sum(\mathbf{v}_l \mathbf{m}_l^T)$ *for all* $\mathbf{m}_l$. *Also, if* $\mathbf{m}_l \neq 0$ *then inequality is strict.*

Proof. Recall that $\mathbf{m}_l$ satisfies $\mathbf{m}_l^{\mathcal{C}_i}$ is nonpositive/nonnegative when $i = l/i \neq l$ for all i. Define $\mathbf{X}^i := \mathbb{1}^{n_l}\mathbf{m}_l^{\mathcal{C}_i T}$. We can write

$$g(\mathbf{v}_l\mathbf{m}_l^T) = \frac{1}{n_l}\text{sum}(\mathbf{X}^l) + \sum_{i=1}^{K} \lambda \text{sum}(\mathbf{X}^i_{\beta^c_{l,i}}).$$

Now assume $i \neq l$. Using Lemma A.1.9 and the fact that $\beta_{l,i}$ is a randomly generated subset (with parameter q), with probability $1 - n_i \exp(-2\epsilon'^2 n_l)$, for all $\mathbf{X}^i$, we have,

$$\text{sum}(\mathbf{X}^i_{\beta^c_{l,i}}) \geq (r(1-q) - \epsilon')\text{sum}(\mathbf{X}^i), \tag{A.2.11}$$

where inequality is strict if $X^i \neq 0$. Similarly, when $i = l$ with probability at least $1 - n_l \exp(-2\epsilon'^2(n_l - 1))$, we have,

$$\frac{1}{\lambda n_l}\text{sum}(\mathbf{X}^l) + \text{sum}(\mathbf{X}^l_{\beta^c_{l,l}}) \geq \left(\frac{1}{\lambda n_l} + r(1-p_l) + \epsilon'\right)\text{sum}(\mathbf{X}^l).$$

Together,

$$\begin{aligned} g(\mathbf{v}_l\mathbf{m}_l^T) &\geq \lambda \sum_{i\neq l}(r(1-q) - \epsilon')\text{sum}(\mathbf{X}^i) + \left(\frac{1}{\lambda n_l} + r(1-p_l) + \epsilon'\right)\text{sum}(\mathbf{X}^l) \\ &\geq \lambda(r(1-q) - \epsilon')\sum_{i=1}^{K}\text{sum}(\mathbf{X}^i) = \lambda(r(1-q) - \epsilon')\text{sum}(\mathbf{v}_l\mathbf{m}_l^T). \end{aligned} \tag{A.2.12}$$

Choosing $\epsilon' = \frac{\tilde{\mathbf{D}}_{\mathcal{A}}}{2}$ and using the facts that $r(1-q) - 2\tilde{\mathbf{D}}_{\mathcal{A}} \geq 0$, $r(p_l - q) - \frac{1}{\lambda n_l} - 2\tilde{\mathbf{D}}_{\mathcal{A}} \geq 0$ and using a union bound, with probability $1 - n\exp(-2\tilde{\mathbf{D}}^2_{\mathcal{A}}(n_l - 1))$, we have $g(\mathbf{v}_l\mathbf{m}_l^T) \geq 0$ and the inequality is strict when $\mathbf{m}_l \neq 0$ as at least one of the $\mathbf{X}^i$'s will be nonzero. ■

The following lemma immediately follows from Lemma A.2.5 and summarizes the main result of the section.

Lemma A.2.6. *Let $\tilde{\mathbf{D}}_{\mathcal{A}}$ be as defined in (A.1.2) and assume $\tilde{\mathbf{D}}_{\mathcal{A}} > 0$. Then with probability $1 - 2nK\exp(-2\tilde{\mathbf{D}}^2_{\mathcal{A}}(n_{min} - 1))$ we have $g(\mathbf{E}^L) > 0$ for all nonzero feasible $\mathbf{E}^L \in \mathcal{M}_U^{\perp}$.*

A.2.4 The Final Step

Lemma A.2.7. *Let $p_{min} > q$ and $\mathcal{G}$ be a random graph generated according to Model 2.2.1 and 2.2.2 with cluster sizes $\{n_i\}_{i=1}^K$. If $\lambda \leq (1-\epsilon)\tilde{\mathbf{\Lambda}}_{succ}$ and $\mathbf{D}_{\min} = \min_{1\leq i\leq n} r\,(p_i - q)\,n_i \geq (1+\epsilon)\frac{1}{\lambda}$, then $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal solution to Program 2.1.4 with probability $1 - \exp(-\Omega(n)) - 6n^2\exp(-\Omega(n_{min}))$.*

Proof. Based on Lemma A.2.4 and Lemma A.2.6, with probability at least $1 - cn^2\exp(-C(r(p_{min}-q))^2 n_{min})$,

- There exists $\mathbf{W} \in \mathcal{M}_{\mathbf{U}}$ with $\|\mathbf{W}\| < 1$ such that for all feasible $\mathbf{E}^L$, $f(\mathbf{E}^L, \mathbf{W}) \geq 0$.
- For all nonzero $\mathbf{E}^L \in \mathcal{M}_{\mathbf{U}}^{\perp}$ we have $g(\mathbf{E}^L) > 0$.

Consequently based on Lemma A.1.3, $(\mathbf{L}^0, \mathbf{S}^0)$ is the unique optimal of Problem 2.1.7. ■

Appendix B

PROOFS FOR RESULTS IN CHAPTER 3

In this appendix we provide the detailed proofs for the results presented in Chapter 3. Recall the convex program:

$$\underset{\mathbf{X}}{\text{minimize}} \ \frac{1}{2}\|\mathbf{A}-\mathbf{X}\|_F^2 + \lambda \,\text{trace}(\mathbf{X}) \tag{B.0.1}$$

subject to

$$\mathbf{X} \succcurlyeq 0 \tag{B.0.2}$$

$$\sum_j \mathbf{X}_{i,j} \leq 1, \ \text{ for all } i \tag{B.0.3}$$

$$\mathbf{X}_{i,j} \geq 0 \ \text{ for all } i,j \in \{1,2,\dots n\}, \tag{B.0.4}$$

where $\|.\|_F$ is the Frobenius norm (square root of the sum of the squares of the entries of the matrix), and $\lambda > 0$ is a regularization parameter. Also, by $\mathbf{X} \succcurlyeq 0$, we mean that $\mathbf{X}$ is symmetric and has non-negative eigenvalues.

B.1 Proof Sketches

First we note that the objective function in Program 3.1.1 is strongly convex in $\mathbf{X}$, and hence has an unique optimal solution. So, it is enough to produce an optimal solution.

Define dual variables for the constraints,

1. $\mathbf{Y} \in \mathbb{R}^{n\times n},\ \mathbf{Y} \succcurlyeq 0$ for constraint (B.0.2).
2. $\nu \in \mathbb{R}^n,\ 2\nu \geq 0$ for constraints (B.0.3).
3. $\mathbf{Z} \in \mathbb{R}^{n\times n},\ \mathbf{Z} \geq 0$ for constraints (B.0.4), where $\geq$ is entry-wise.

Lagrange can be written as,

$$\begin{aligned}\mathcal{L}(\mathbf{X};\mathbf{Y},\mathbf{Z},\nu) = \max_{\mathbf{Z}\geq 0,\mathbf{Y}\succcurlyeq 0,\nu\geq 0} &- 2\nu^T\,\mathbb{1}\min_{\mathbf{X}} \ \frac{1}{2}||\mathbf{A}-\mathbf{X}||_F^2 \\ &+\text{trace}(\lambda\mathbf{I}-\mathbf{Z}-\mathbf{Y}+\mathbb{1}\nu^T+\nu\mathbb{1}^T+\mathbf{X}),\end{aligned} \tag{B.1.1}$$

where $\mathbb{1} \in \mathbb{R}^n$ is a vector of all 1's.

If a feasible $\hat{\mathbf{X}}$ is an optimal solution to Program (3.1.1), then the following conditions have to hold (from KKT conditions and complementary slackness):

$$\mathbf{Z} + \mathbf{Y} = \lambda \mathbf{I} + \hat{\mathbf{X}} + \mathbb{1}\nu^T - \mathbf{A} + \nu \mathbb{1}^T \tag{B.1.2}$$

$$\text{trace}(\hat{\mathbf{X}}\mathbf{Y}) = 0, \tag{B.1.3}$$

$$\text{trace}(\hat{\mathbf{X}}\mathbf{Z}) = 0, \tag{B.1.4}$$

$$\nu^T \left(\hat{\mathbf{X}}\mathbb{1} - \mathbb{1} \right) = 0. \tag{B.1.5}$$

Since $\hat{\mathbf{X}}, \mathbf{Y} \succcurlyeq 0$, from(B.1.3), we get

$$\hat{\mathbf{X}}\mathbf{Y} = \mathbf{Y}\hat{\mathbf{X}} = 0. \tag{B.1.6}$$

We first construct dual variables that satisfy the conditions (B.1.2) (B.1.3) (B.1.4) and (B.1.5). The dual variables $\mathbf{Z}, \mathbf{Y}, \nu$ thus obtained are functions of the problem parameters $\{\{\mu_i\}_{i\in[K]}, \mu_{out}, \sigma, \{n_i\}_{out}, \mathbf{n}_{K+1}\}$. The condition $\mathbf{Y} \succcurlyeq 0$ will give the lower bound on λ of the form Λ or $\Lambda + \mu_{out} n_{out}$ depending on the case. The conditions $\nu \geq 0$ gives the lower bounds on the cluster densities ρ. The condition $\mathbf{Z} \geq 0$ gives the lower bounds on cross-cluster densities γ and the effective cluster densities η.

Notation: Let $[m] := \{1, 2, \cdots, m\}$. Let n be the number of data points denoted by $[n]$ with K disjoint clusters. Let $\mathcal{C}_i \subset [n]$ be the set of data points in cluster i and the size (number of data points in the cluster) be n_i. μ_i is the mean similarity between nodes inside the same cluster i. μ_{out} is the mean similarity between nodes that are not in the same cluster, and σ^2 is the noise variance.

For a vector $\mathbf{v} \in \mathbb{R}^n$, $\mathbf{v}_{[i]}$ denotes a vector zero-entries everywhere except for the indices corresponding to $\mathcal{C}_i$ where it is equal to the entries of $\mathbf{v}$ corresponding to $\mathbf{C}_i$. Similarly for a matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$, $\mathbf{B}_{[i][j]}$ denotes a matrix where the entries in the block corresponding to $\mathbf{C}_i \times \mathbf{C}_j$ are the same as the $\mathbf{C}_i \times \mathbf{C}_j$ block in matrix $\mathbf{B}$, and the rest of the entries (entries except the block corresponding to $\mathbf{C}_i \times \mathbf{C}_j$) are zeros.

Let us first look at the case without outliers. The solution to the case with outliers builds on the case without outliers.

B.2 No Outliers

For the case with K disjoint clusters and no outliers, the desired solution is

$$\mathbf{X}^*_{l,m} = \begin{cases} \frac{1}{n_i}\,, & \text{if both nodes } l, m \text{ are in the same cluster } i \\ 0\,, & \text{if nodes } l, m \text{ are not in the same cluster,} \end{cases} \tag{B.2.1}$$

which has non-zero entries only in the region corresponding to clusters.

Our solution does not depend on the order in which the data points are arranged, but the description of the proof will be easy to visualize when they are arranged in the order. Thus, to aid visual understanding, assume that all the data points are arranged in the order of which cluster they belong to. In this case, $\mathbf{X}^*$ has block diagonal structure with $K \times K$ blocks. Diagonal blocks, $\mathbf{X}^*_{[i][i]} = \frac{1}{n_i}\mathbb{1}_{[i]}\mathbb{1}^T_{[i]}$ and off diagonal blocks, $\mathbf{X}^*_{[i][j]} = 0$ for $i \neq j$, $i, j \in [K]$.

Let $\mathbf{A} = \mathbf{M} + \sigma\ \mathbf{\Phi}$, where $\mathbf{\Phi}$ has i.i.d entries with zero mean and variance 1 and $\mathbf{M}$ is the matrix of mean similarities. Note that $\mathbf{M}_{[i][i]} = \mu_i \mathbb{1}_{[i]}\mathbb{1}^T_{[i]}$ and $\mathbf{M}_{[i][j]} = \mu_{out}\mathbb{1}_{[i]}\mathbb{1}^T_{[j]}$ for $i \neq j \in [K]$.

The condition (B.1.6) implies, for $i, j \in [K]$,

$$\mathbf{Y}_{[i][i]}\mathbb{1}_{[i]} = 0, \tag{B.2.2}$$

and

$$\mathbf{Y}_{[i][j]}\mathbb{1}_{[j]} = 0. \tag{B.2.3}$$

Also, since $\text{trace}(\mathbf{X}^*\mathbf{Z}) = 0$ and $\mathbf{X}^*_{[i][i]} > 0$, we have

$$\mathbf{Z}_{[i][i]} = 0. \tag{B.2.4}$$

B.2.1 Expression for ν

From (B.1.2), for $i \in [K]$,

$$\begin{aligned} \mathbf{Z}_{[i][i]} + \mathbf{Y}_{[i][i]} = \lambda\mathbf{I}_{[i][i]} + \mathbf{X}^*_{[i][i]} - \mathbf{M}_{[i][i]} - \sigma\ \mathbf{\Phi}_{[i][i]} \\ + \mathbb{1}_{[i]}\nu^T_{[i]} + \nu_{[i]}\mathbb{1}_{[i]}. \end{aligned} \tag{B.2.5}$$

From (B.2.4) and (B.2.2),

$$\begin{aligned} 0 = \lambda\mathbb{1}_{[i]} + \frac{1}{n_i}\mathbb{1}_{[i]}n_i - \mu_i\mathbb{1}_{[i]}n_i - \sigma\ \mathbf{\Phi}_{[i][i]}\mathbb{1}_{[i]} \\ + \mathbb{1}_{[i]}\left(\nu^T_{[i]}\mathbb{1}_{[1]}\right) + \nu_{[i]}n_i. \end{aligned} \tag{B.2.6}$$

From which we can solve for $\nu_{[i]}$ to obtain,

$$\begin{aligned}\nu_{[i]} = &\left(\frac{\mu_i n_i - (\lambda+1)}{2n_i}\right)\mathbb{1}_{[i]} \\ &+\left(-\frac{\sigma}{2n_i^2}\left(\mathbb{1}_{[i]}^T\mathbf{\Phi}_{[i][i]}\mathbb{1}_{[i]}\right)\mathbf{I}_{[i][i]} + \frac{\sigma}{n_i}\mathbf{\Phi}_{[i][i]}\right)\mathbb{1}_{[i]}.\end{aligned} \tag{B.2.7}$$

Theorem 11 (Hoeffding)**.** *Let $W_1, \cdots, W_m$ be independent random variables with zero mean and $|W_i| \leq R_i$ almost surely for all i. Define*

$$S := \sum_{i=1}^{m} W_i \qquad \tau^2 := \sum_{i=1}^{m} R_i.$$

Then, Var$(S) \leq \tau^2$ and $\mathbb{P}\{|S| > \delta\} \leq 2\exp^{-\delta^2/\tau^2}$.

Lemma B.2.1. *If $\frac{-(\lambda+1)+n_i\mu_i}{2n_i} > 0,\ \forall i \in [K]$, then $\nu \geq 0$ with probability at least $1 - n\exp\frac{-\delta^2 n_{\min}}{2\sigma^2}$, for $\delta = \min_{i\in[K]}\frac{-(\lambda+1)+n_i\mu_i}{2n_i}$.*

The condition $\frac{-(\lambda+1)+n_i\mu_i}{2n_i} > 0$ implies $\rho_i > \lambda + 1,\ i \in [K]$ from the definition of ρ_i and hence the condition $\rho_{\min} > \lambda + 1$.

B.2.2 Expression for Z

Now consider the off-diagonal blocks of (B.1.2), for $i \neq j \in [K]$,

$$\begin{aligned}\mathbf{Z}_{[i][j]} + \mathbf{Y}_{[i][j]} = &\underbrace{\lambda\mathbf{I}_{[i][j]}}_{0} + \underbrace{\mathbf{X}^*_{[i][j]}}_{0} - \mathbf{M}_{[i][j]} - \sigma\,\mathbf{\Phi}_{[i][j]} \\ &+\mathbb{1}_{[i]}\nu_{[j]}^T + \nu_{[i]}\mathbb{1}_{[j]}^T.\end{aligned}$$

From (B.2.3), we have

$$\begin{aligned}\mathbf{Z}_{[i][j]}\mathbb{1}_{[j]} + \underbrace{\mathbf{Y}_{[i][j]}\mathbb{1}_{[j]}}_{0} = &-\mu_{out}\mathbb{1}_{[i]}\mathbb{1}_{[j]}^T\mathbb{1}_{[j]} - \sigma\,\mathbf{\Phi}_{[i][j]}\mathbb{1}_{[j]} \\ &+\mathbb{1}_{[i]}\nu_{[j]}^T\mathbb{1}_{[j]} + \nu_{[i]}\mathbb{1}_{[j]}^T\mathbb{1}_{[j]},\end{aligned}$$

which simplifies to

$$\mathbf{Z}_{[i][j]}\mathbb{1}_{[j]} = -\mu_{out}n_j\mathbb{1}_{[i]} - \sigma\,\mathbf{\Phi}_{[i][j]}\mathbb{1}_{[j]} + \mathbb{1}_{[i]}\nu_{[j]}^T\mathbb{1}_{[j]} + n_j\nu_{[i]}. \tag{B.2.8}$$

Define,

$$\mathbf{N}_{[i][j]} = -\mathbf{M}_{[i][j]} - \sigma\,\mathbf{\Phi}_{[i][j]} + \mathbb{1}_{[i]}\nu_{[j]}^T + \nu_{[i]}\mathbb{1}_{[j]}^T. \tag{B.2.9}$$

The expected value of $\mathbf{N}_{[i][j]}$ is,

$$\hat{\mathbf{N}}_{[i][j]} = \left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right)\mathbb{1}_{[i]}\mathbb{1}_{[j]}^T. \tag{B.2.10}$$

We now proceed to construct $\mathbf{Z}_{[i][j]}$ based on $\hat{\mathbf{N}}$ as follows:

$$\mathbf{Z}_{[i][j]} = \hat{\mathbf{N}}_{[i][j]} + \mathbf{w}_{[i]}^j\mathbb{1}_{[j]}^T + \mathbb{1}_{[i]}\left(\mathbf{u}_{[j]}^i\right)^T, \tag{B.2.11}$$

where $\mathbf{w}_{[i]}^j$ is variable vector with non-zero entries only for indices in $\mathbf{C}_i$ (the superscript j is to identify its association to the block matrix $\mathbf{Z}_{[i][j]}$) and $\mathbf{u}_{[j]}^i$ is a variable vector with non-zero entries only for indices in $\mathbf{C}_j$ (the superscript i is to identify its association to the block matrix $\mathbf{Z}_{[i][j]}$). The variables in $\mathbf{w}_{[i]}^j$ and $\mathbf{u}_{[j]}^i$ can be found as a solution to the following system of equations obtained from $\mathbf{Z}_{[i][j]}\mathbb{1}_{[j]} = \mathbf{N}_{[i][j]}\mathbb{1}_{[j]}$ and $\mathbb{1}_{[i]}^T\mathbf{Z}_{[i][j]} = \mathbb{1}_{[i]}^T\mathbf{N}_{[i][j]}$:

$$\mathbf{w}_{[i]}^j n_j + \mathbb{1}_{[i]}\left(\mathbf{u}_{[j]}^i\right)^T\mathbb{1}_{[j]} = \underbrace{\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)\mathbb{1}_{[j]}}_{:=\mathbf{t}_1}, \tag{B.2.12}$$

$$\mathbb{1}_{[j]}\left(\mathbf{w}_{[i]}^j\right)^T\mathbb{1}_{[i]} + n_i\mathbf{u}_{[j]}^i = \underbrace{\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)^T, \mathbb{1}_{[i]}}_{:=\mathbf{t}_2} \tag{B.2.13}$$

which can be written as,

$$\begin{bmatrix} n_j\mathbf{I}_{[i][i]} & \mathbb{1}_{[i]}\mathbb{1}_{[j]}^T \\ \mathbb{1}_{[j]}\mathbb{1}_{[i]}^T & n_i\mathbf{I}_{[j][j]} \end{bmatrix}\begin{bmatrix} \mathbf{w}_{[i]}^j \\ \mathbf{u}_{[j]}^i \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \end{bmatrix}. \tag{B.2.14}$$

Since the system of equations (B.2.14) is singular with null space spanned by $(\mathbb{1}_{[i]}; -\mathbb{1}_{[j]})$, we proceed by solving the following perturbed system for $\theta > 0$,

$$\begin{bmatrix} n_j\mathbf{I}_{[i][i]} + \theta\mathbb{1}_{[i]}\mathbb{1}_{[i]}^T & (1-\theta)\mathbb{1}_{[i]}\mathbb{1}_{[j]}^T \\ (1-\theta)\mathbb{1}_{[j]}\mathbb{1}_{[i]}^T & n_i\mathbf{I}_{[j][j]} + \theta\mathbb{1}_{[j]}\mathbb{1}_{[j]}^T \end{bmatrix}\begin{bmatrix} \mathbf{w}_{[i]}^j \\ \mathbf{u}_{[j]}^i \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \end{bmatrix}. \tag{B.2.15}$$

Left multiplying the system (B.2.15) by $(\mathbb{1}_{[i]}; -\mathbb{1}_{[j]})^T$ and observing that $(\mathbb{1}_{[i]}; -\mathbb{1}_{[j]})^T(\mathbf{t}_1; \mathbf{t}_2) = 0$, we obtain,

$$\underbrace{\theta(n_i+n_j)}_{>0}\left(\mathbb{1}_{[i]}^T\mathbf{w}_{[i]}^j - \mathbb{1}_{[j]}^T\mathbf{u}_{[j]}^T\right) = 0$$

and hence,

$$\mathbb{1}_{[i]}^T\mathbf{w}_{[i]}^j = \mathbb{1}_{[j]}^T\mathbf{u}_{[j]}. \tag{B.2.16}$$

Using (B.2.16) in the equations (B.2.12) and (B.2.13), we get,

$$\left(n_j \mathbf{I}_{[i][i]} + \mathbb{1}_{[i]}\mathbb{1}_{[i]}^T\right) \mathbf{w}_{[i]}^j = \left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right) \mathbb{1}_{[j]}$$

and

$$\left(\mathbb{1}_{[j]}\mathbb{1}_{[j]}^T + n_i \mathbf{I}_{[j][j]}\right) \mathbf{u}_{[j]}^i = \left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)^T \mathbb{1}_{[i]},$$

which can be solved to get,

$$\mathbf{w}_{[i]}^j = \frac{1}{n_j}\left(\mathbf{I}_{[i][i]} - \frac{\mathbb{1}_{[i]}\mathbb{1}_{[i]}^T}{n_i + n_j}\right)\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)\mathbb{1}_{[j]} \tag{B.2.17}$$

and

$$\mathbf{u}_{[j]}^i = \frac{1}{n_i}\left(\mathbf{I}_{[j][j]} - \frac{\mathbb{1}_{[j]}\mathbb{1}_{[j]}^T}{n_i + n_j}\right)\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)^T \mathbb{1}_{[i]}. \tag{B.2.18}$$

Substituting in (B.2.11), we get the following expression for $\mathbf{Z}_{[i][j]}$,

$$\begin{aligned}
&\mathbf{Z}_{[i][j]} = \\
&\left(-\mu_{out} + \frac{-(\lambda+1) + n_i\mu_i}{2n_i} + \frac{-(\lambda+1) + n_j\mu_j}{2n_j}\right)\mathbb{1}_{[i]}\mathbb{1}_{[j]}^T \\
&+ \frac{1}{n_j}\left(\mathbf{I}_{[i][i]} - \frac{\mathbb{1}_{[i]}\mathbb{1}_{[i]}^T}{n_i + n_j}\right)\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)\mathbb{1}_{[j]}\mathbb{1}_{[j]}^T \\
&+ \frac{1}{n_i}\mathbb{1}_{[i]}\mathbb{1}_{[i]}^T\left(\mathbf{N}_{[i][j]} - \hat{\mathbf{N}}_{[i][j]}\right)\left(\mathbf{I}_{[j][j]} - \frac{\mathbb{1}_{[j]}\mathbb{1}_{[j]}^T}{n_i + n_j}\right).
\end{aligned} \tag{B.2.19}$$

B.2.3 Expression for $\mathbf{Y}$

From (B.1.2), and the expressions for ν and $\mathbf{Z}$ (equations (B.2.7) and (B.2.19)), we have a construction for $\mathbf{Y}$ since,

$$\mathbf{Y} = -\mathbf{Z} + \lambda\mathbf{I} + \mathbf{X}^* - \mathbf{M} - \sigma\mathbf{\Phi} + \mathbb{1}\nu^T + \nu\mathbb{1}^T.$$

Now that we have expressions for candidate dual variables $\mathbf{Z}, \mathbf{Y}, \nu$, we proceed to show that under reasonable conditions on the problem parameters $\{\{\mu_i\}_{i\in[K]}, \mu_{out}, \sigma, \{n_i\}_{i\in[K]}\}$ and the regularization parameter λ, the conditions, $\mathbf{Y} \succcurlyeq 0,\ \nu \geq 0, \mathbf{Z} \geq 0$ hold with high probability.

B.2.4 Positive semidefiniteness of $\mathbf{Y}$

The expression for $\mathbf{Y}$ is as follows,

$$\mathbf{Y} = \lambda\mathbf{I} - \sigma\mathbf{\Phi}\underbrace{-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T}_{\text{all have } \mathbb{1}_{[i]} \text{or} \mathbb{1}_{[i]}^T \text{ by construction}}.$$

For any vector $\mathbf{x} \in \mathbb{R}^n$, consider the decomposition,

$$\mathbf{x} = \sum_{i=1}^{K} x_i \mathbb{1}_{[i]} + \mathbf{x}_\perp, \tag{B.2.20}$$

where $\mathbf{x}_\perp$ is sum of components perpendicular to $\mathbb{1}_{[i]}$, $i \in [K]$. From KKT, $\mathbf{Y}\mathbb{1}_{[i]} = 0$, $\mathbb{1}_{[i]}^T \mathbf{Y} = 0$. Also, from construction of $\mathbf{Z}$, ν and forms of $\mathbf{X}^*$ and $\mathbf{M}$,

$$\mathbf{x}_\perp^T \left(-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T\right) = \mathbf{0}^T$$

$$\left(-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T\right)\mathbf{x}_\perp = \mathbf{0}.$$

So, $\mathbf{x}^T\mathbf{Y}\mathbf{x} = \mathbf{x}_\perp^T(\lambda\mathbf{I} - \sigma\mathbf{\Phi})\mathbf{x}_\perp \geq (\lambda - \sigma||\mathbf{\Phi}||)||\mathbf{x}_\perp||_2^2$. Since $\mathbf{\Phi}$ is a random matrix with bounded i.i.d. entries with zero mean and unit variance, using the standard results in random matrix theory [Vu05], with high probability $(1 - \exp(-\Omega(n)))$, $||\mathbf{\Phi}|| = 2\sqrt{n}$. Thus, if $\lambda > \sigma 2\sqrt{n}$, then $\mathbf{Y}$ is positive semidefinite with high probability (Lemma B.2.2).

Lemma B.2.2. *If* $\lambda > \sigma 2\sqrt{n} := \Lambda$, *then* $\mathbf{Y} \succcurlyeq 0$ *with at least* $1 - \exp(-\Omega(n))$.

Summary

Lemma B.2.3. *If* $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right) > 0$, *for all* $i \neq j \in [K]$, *then* $\mathbf{Z} \geq 0$ *with probability at least* $1 - n^2 \exp^{-(\delta')^2\Omega(n_{\min})/\sigma^2}$, *for* $\delta' = \min_{i,j}\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right)$.

The condition, for all $i \neq j \in [K]$, $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right) > 0$ implies $\gamma_{ij} > \lambda + 1$ for all $i \neq j \in [K]$ by the definition of γ_{ij}, and hence the condition $\gamma_{\min} > \lambda + 1$.

From Lemma B.2.2, B.2.1, and B.2.3 we can conclude Theorem 3.

B.3 Large Number of Outliers

For the case with K disjoint clusters with large number of outliers, the desired solution is

$$\tilde{\mathbf{X}}_{l,m} = \begin{cases} \frac{1}{n_i}\,, & \text{if both nodes } l, m \text{ are in the same cluster } i. \\ 0\,, & \text{if nodes } l, m \text{ are in different clusters.} \\ \frac{1}{n_{K+1}} & \text{if both nodes } l, m \text{ are outliers.} \end{cases}, \tag{B.3.1}$$

The outliers together get considered as a new cluster $K+1$. We have,

$$\hat{\mathbf{X}}_{[K+1][K+1]} = n_{K+1}\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T,\ \mathbf{M}_{[K+1][K+1]} = \mu_{out}\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T,\ \mu_{K+1} = \mu_{out}.$$

So, $\mathbf{Z}_{[K+1][K+1]} = 0$ and expressions for $\nu_{[K+1]}$, $\mathbf{Z}_{[i][K+1]}$ obtained just the same way as for other clusters as in equations (B.2.7) and (B.2.19). The conditions $\rho_{K+1} > \lambda + 1$ and $\gamma_{i,K} > \lambda + 1$ for all $i \in [K]$ are required for $\nu_{K+1} > 0$ and $\mathbf{Z}_{[i][K+1]} > 0$ respectively.

Lemma B.3.1. *If* $\lambda > \sigma 2\sqrt{n} := \Lambda$, *then* $\mathbf{Y} \succcurlyeq 0$ *with at least* $1 - \exp(-\Omega(n))$.

Lemma B.3.2. *If* $\frac{-(\lambda+1)+n_i\mu_i}{2n_i} > 0,\ \forall i \in [K+1]$, *then* $\nu \geq 0$ *with probability at least* $1 - n\exp^{\frac{-\delta^2 n_{\min}}{2\sigma^2}}$, *for* $\delta = \min_{i\in[K+1]} \frac{-(\lambda+1)+n_i\mu_i}{2n_i}$.

The condition $\frac{-(\lambda+1)+n_i\mu_i}{2n_i} > 0$ implies $\rho_i > \lambda + 1,\ i \in [K+1]$ from the definition of ρ_i and hence the condition $\rho_{\min} > \lambda + 1$.

Lemma B.3.3. *If* $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right) > 0$, *for all* $i \neq j \in [K+1]$, *then* $\mathbf{Z} \geq 0$ *with probability at least* $1 - n^2\exp^{-(\delta')^2\Omega(n_{\min})/\sigma^2}$, *for* $\delta' = \min_{i,j}\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right)$.

The condition, for all $i \neq j \in [K+1]$, $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right) > 0$ implies $\gamma_{ij} > \lambda + 1$ for all $i \neq j \in [K+1]$ by the definition of γ_{ij}, and hence the condition $\gamma_{\min} > \lambda + 1$.

From Lemma B.3.1, B.3.2 and B.3.3 we can conclude Theorem 5.

B.4 Small Number of Outliers

In the case where the number of outliers is small, the desired solution is (B.2.1). So, $\mathbf{X}^*$ has non-zero entries only in the diagonal blocks for $i \in [K]$ and $\mathbf{X}^*_{[K+1][K+1]} = \mathbf{0}$. So, from (B.1.5),

$$\nu^T(\mathbf{X}^*\mathbb{1} - \mathbb{1}) = \mathbf{0}.$$

and $\nu \geq 0$, we get, $\nu_{[K+1]} = 0$. The expressions for $\nu_{[i]}, i \in [K]$ remain the same as in the case of no outliers (Equation (B.2.7)). The expressions for $\mathbf{Z}_{[i][j]}$ for $i \neq j \in [K]$ remain the same as in (B.2.19). From steps similar to the case of no outliers, we can find the expression for $\mathbf{Z}_{[i][K+1]},\ i \in [K]$ as,

$$\mathbf{Z}_{[i][K+1]} = \hat{\mathbf{N}}_{[i][K+1]} + \mathbf{w}_{[i]}^{K+1}\mathbb{1}_{[K+1]}^T + \mathbb{1}_{[i]}\left(\mathbf{u}_{[K+1]}^i\right)^T, \tag{B.4.1}$$

where,

$$\hat{\mathbf{N}}_{[i][K+1]} = \left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i}\right) \mathbb{1}_{[i]}\mathbb{1}_{[K+1]}^T, \tag{B.4.2}$$

since $\nu_{[K+1]} = 0$ and $\mathbf{w}$ and $\mathbf{u}$ are defined as in (B.2.17) and (B.2.18). For $\mathbf{Z}_{[i][K+1]} > 0$ to hold with high probability we will require, $-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} > 0$, which gives the condition $\eta_i > \lambda + 1$ from the definition of η_i.

B.4.1 Positive semidefiniteness of $\mathbf{Y}$

The expression for $\mathbf{Y}$ is as follows:

$$\mathbf{Y} = \lambda\mathbf{I} - \sigma\mathbf{\Phi}\underbrace{-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T}_{\text{all have } \mathbb{1}_{[i]} \text{or} \mathbb{1}_{[i]}^T \text{ by construction}}$$

Further, $\mathbf{X}^*_{[K+1][K+1]}$, $\nu_{[K+1]}$ and $\mathbf{Z}_{[K+1][K+1]}$ are all zeros.

For any vector $\mathbf{x} \in \mathbb{R}^n$, consider the decomposition,

$$\mathbf{x} = \sum_{i=1}^{K} x_i \mathbb{1}_{[i]} + \mathbf{x}_\perp + \mathbf{x}_{out}, \tag{B.4.3}$$

where $\mathbf{x}_\perp$ is sum of components perpendicular to $\mathbb{1}_{[i]}$, $i \in [K]$ and has all zero entries in the entries corresponding to $\mathbf{C}_{K+1}$, $\mathbf{x}_{out}$ has non-zero entries only in the entries corresponding to C_{K+1}.

From KKT, $\mathbf{Y}\mathbb{1}_{[i]} = 0$, $\mathbb{1}_{[i]}^T\mathbf{Y} = 0$ for $i \in [K]$. Also, from construction of $\mathbf{Z}$, ν and forms of $\mathbf{X}^*$ and $\mathbf{M}$,

$$\mathbf{x}_\perp^T\left(-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T\right) = \mathbf{0}^T$$

$$\left(-\mathbf{Z} + \mathbf{X}^* - \mathbf{M} + \mathbb{1}\nu^T + \nu\mathbb{1}^T\right)\mathbf{x}_\perp = \mathbf{0}$$

$$\begin{aligned}
&\mathbf{x}^T\mathbf{Y}\mathbf{x} = \mathbf{x}_\perp^T\mathbf{Y}\mathbf{x}_\perp + \mathbf{x}_{out}^T\mathbf{Y}\mathbf{x}_{out} + 2\mathbf{x}_{out}^T\mathbf{Y}\mathbf{x}_\perp \\
&= \mathbf{x}_\perp^T\left(\lambda\mathbf{I} - \sigma\mathbf{\Phi}\right)\mathbf{x}_\perp + \mathbf{x}_{out}^T\left(\lambda\mathbf{I} - \sigma\mathbf{\Phi} - \mu_{out}\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T\right)\mathbf{x}_{out} \\
&+2\mathbf{x}_{out}^T\left(-\sigma\mathbf{\Phi} - \underbrace{\mu_{out}\mathbb{1}_{[K+1]}\mathbb{1}_{[1:K]}^T + \mathbb{1}_{[K+1]}^T\nu_{[1:K]} - Z_{[i][K+1]}}_{\text{all have } \mathbb{1}_{[i]} or \mathbb{1}_{[i]}^T}\right)\mathbf{x}_\perp \\
&= \mathbf{x}_\perp^T\left(\lambda\mathbf{I} - \sigma\mathbf{\Phi}\right)\mathbf{x}_\perp + \mathbf{x}_{out}^T\left(\lambda\mathbf{I} - \sigma\mathbf{\Phi} - \mu_{out}\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T\right)\mathbf{x}_{out} \\
&\qquad +2\mathbf{x}_{out}\left(-\sigma\mathbf{\Phi}_{[K+1][1:K]}\right)\mathbf{x}_\perp,
\end{aligned} \tag{B.4.4}$$

where $\mathbf{v}_{[1:K]}$ denotes a vector with non-zero values in entries corresponding to $\cup_i \mathbf{C}_i,\ i \in [K]$ and zeros in the entries corresponding to $\mathbf{C}_{K+1}$ and similarly for a matrix, $\mathbf{B}_{[i][1:K]}$ has non-zero block corresponding to $\mathbf{C}_i \times \cup_j \mathbf{C}_j,\ j \in [K]$ and zero everywhere else.

Defining $\mathbf{q} := (\mathbf{x}_\perp; \mathbf{x}_{out})$,

$$\begin{aligned}\mathbf{x}^T\mathbf{Y}\mathbf{x} &= \mathbf{q}^T\lambda\mathbf{I}\mathbf{q} - \mathbf{q}^T\sigma\mathbf{\Phi}\mathbf{q} - \mathbf{q}^T\mathbf{M}_{[K+1][K+1]}\mathbf{q} \\ &\geq \left(\lambda - \sigma||\mathbf{\Phi}|| - \mu_{out}||\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T||\right)||\mathbf{q}||_2^2,\end{aligned} \tag{B.4.5}$$

since $||\sigma\mathbf{\Phi} + \mu_{out}\mathbb{1}_{[K+1][K+1]}|| \leq \sigma||\mathbf{\Phi}|| + \mu_{out}||\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T||$. $||\mathbb{1}_{[K+1]}\mathbb{1}_{[K+1]}^T|| = n_{K+1}$. Since $\mathbf{\Phi}$ is a random matrix with bounded i.i.d. entries with zero mean and unit variance, using the standard results in random matrix theory [Vu05], with high probability, $||\mathbf{\Phi}|| = 2\sqrt{n}$. Hence, if $\lambda > \sigma 2\sqrt{n} + \mu_{out}n_{K+1} := \Lambda + \mu_{out}n_{out}$, then $\mathbf{Y}$ is positive semidefinite with high probability.

Lemma B.4.1. *(Small Outliers) If* $\lambda > \sigma 2\sqrt{n} + \mu_{out}n_{out} := \Lambda + \mu_{out}n_{out}$, *then* $\mathbf{Y} \succcurlyeq 0$ *with at least* $1 - \exp(-\Omega(n))$.

Lemma B.4.2. *If* $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i} + \frac{-(\lambda+1)+n_j\mu_j}{2n_j}\right) > 0$ *and* $\left(-\mu_{out} + \frac{-(\lambda+1)+n_i\mu_i}{2n_i}\right) > 0$ *for all* $i \neq j \in [K]$, *then* $\mathbf{Z} \geq 0$ *with probability at least* $1 - n^2\exp^{-(\delta^{``})^2\Omega(n_{\min})/\sigma^2}$, *for* $\delta'' = \min\{\eta_{\min}, \gamma_{\min}\}$.

From Lemma B.4.1, B.2.1, and B.4.2 we can conclude Theorem 4.

Appendix C

PROOFS FOR RESULTS IN CHAPTER 6

In this appendix we provide proofs for the results presented in Chapter 6.

C.1 Proof for Propositions 1 and 2

Following is the proof for Proposition 1.

Proof. Define the following event:

$$\mathcal{E}(\Delta, m) = \left\{ |\overline{X}_{ij}(m) - \mathbb{E}(\overline{X}_{ij}(m))| \leq \Delta, \forall\, i, j \right\}.$$

Suppose the event $\mathcal{E}(\Delta, m)$ occurs. Then the following holds:

$$-\Delta \leq \overline{X}_{ij}(m) - \mathbb{E}(\overline{X}_{ij}(m)) \leq \Delta \Rightarrow \mathbb{E}(\overline{X}_{ij}(m)) - 2\Delta \leq \overline{X}_{ij}(m) - \Delta \leq \mathbb{E}(\overline{X}_{ij}(m)).$$

If $X_{ij} \sim \text{Bern}(p)$, then $\mathbb{E}(\overline{X}_{ij}(m)) = p$. Therefore we have,

$$p - 2\Delta \leq \overline{X}_{ij}(m) - \Delta.$$

Recall that $\Delta = \frac{1}{2}(p-q)$, and hence we have $p - 2\Delta = q$. So, if $X_{ij} \sim \text{Bern}(p)$, then $\overline{X}_{ij}(m) - \Delta > q$ is true when the event $\mathcal{E}(\Delta, m)$ occurs. Using similar arguments, we can show that if $X_{ij} \sim \text{Bern}(q)$, then $\overline{X}_{ij}(m) + \Delta < p$ is true when the event $\mathcal{E}(\Delta, m)$ occurs.

Now we need to show that under the assumptions of the proposition, the event $\mathcal{E}(\Delta, m)$ holds with high probability. We use Hoeffding's inequality for this:

Lemma C.1.1 (Hoeffding's Inequality for Bernoulli Random Variables)**.** *Let* $\{X_i\}_{i=1}^m$ *be i.i.d random variables with* $X_i \sim$ *Bern*(p)*. Then the* $\mathbf{Pr}\left(\left|\overline{X}_i(m) - p\right| \geq \epsilon\right) \leq 2e^{-2m\epsilon^2}$.

From Hoeffding's inequality, for $M = \left\lceil \frac{7 \log n}{2\Delta^2} \right\rceil$, for any given pair of items i and j, $|\overline{X}_{ij}(M) - \mathbb{E}(\overline{X}_{ij}(M))| \leq \Delta$ with probability at least $1 - \frac{1}{n^7}$. Using union bound, we can argue that the event $\mathcal{E}(\Delta, M)$ holds with probability at least $1 - \frac{1}{n^5}$. Therefore, for each pair of items i and j, with M independent queries, we can guarantee with high probability that either $\overline{X}_{ij}(M) - \Delta > q$ or $\overline{X}_{ij}(M) + \Delta < p$.

■

Proof for Proposition 2 is as follows.

Furthermore, $\Omega\left(\frac{1}{\Delta^2}\right)$ repetitions per query and hence $\Omega\left(\frac{nK}{\Delta^2}\right)$ queries are necessary to guarantee success with probability $1-\delta'$.

For the lower bounds, we will use the following result on the tail probability of sum of independent random variables due to Feller [Fel43]

Lemma C.1.2 (Feller, Lower Bound on Tail Probability of Sum of Independent Random Variables)**.** *There exists positive universal constants c_1 and c_2 such that for any set of independent random variables $X_1, \cdots, X_m$ satisfying $\mathbb{E}[X_i] = 0$ and $|X_i| \leq M$, for every $i \in \{1, \cdots, m\}$, if $\sum_{i=1}^{m} \mathbb{E}[(X_i)^2] \geq c_1$, then for every $\epsilon \in [0, \frac{\sum_{i=1}^{m} \mathbb{E}[(X_i)^2]}{M\sqrt{c_1}}]$,*

$$\mathbf{Pr}\left(\sum_{i=1}^{m} X_i > \epsilon\right) \geq c_2 \exp\left(\frac{-\epsilon^2}{12\sum_{i=1}^{m} \mathbb{E}[(X_i)^2]}\right). \tag{C.1.1}$$

We start with the following result on the tail probability of sum of independent Bernoulli random variables due to Feller [Fel43]

Lemma C.1.3 (Lower Bound on Tail Probability of Sum of Independent Bernoulli Random Variables [Fel43])**.** *There exists positive universal constant a such that for any set of iid Bernoulli random variables $X_1, \cdots, X_m$, and $Var(X_i) > 0$, then for every $\epsilon \in [0, \sqrt{\frac{Var(X_i)}{m}}]$,*

$$\mathbf{Pr}\left(\left|\frac{1}{m}\sum_{i=1}^{m}(X_i - \mathbb{E}(X_i))\right| > \epsilon\right) \geq a \exp\left(\frac{-m\epsilon^2}{12Var(X_i)}\right). \tag{C.1.2}$$

So, if $m < \frac{\text{Var}(X_i)}{\Delta^2}$,

$$\mathbf{Pr}\left(\left|\frac{1}{m}\sum_{i=1}^{m}(X_i - \mathbb{E}(X_i))\right| > \Delta\right) \geq ae^{\frac{-1}{12}}.$$

We want to show that, $\Omega\left(\frac{1}{\Delta^2}\right)$ repetitions per query and hence $\Omega\left(\frac{nK}{\Delta^2}\right)$ queries are necessary to guarantee success with probability $1-\delta'$.

Suppose we are making Query(i, j) for $i \notin$ cluster(j). So, $X_{ij} \sim$ Bern(q). If we make $M < \frac{q(1-q)}{\Delta^2}$ queries, then there exists universal constant $a > 0$ such that,

$$\mathbf{Pr}(\overline{X}_{ij}(M) - q > \Delta) \geq ae^{\frac{-1}{12}}.$$

So, the event $\{\overline{X}_{ij}(M) - \Delta > q\}$ occurs with constant probability. By definition of Δ,

$$\overline{X}_{ij}(M) - \Delta > q \Rightarrow \overline{X}_{ij}(M) > q + \Delta = 1/2 - 2\Delta + \Delta \Rightarrow \overline{X}_{ij}(M) + \Delta > 1/2.$$

So, when $\{\overline{X}_{ij}(M) - \Delta > q\}$ occurs, $\overline{X}_{ij}(M) + \Delta > 1/2$. Now, if $\overline{X}_{ij}(M) - \Delta > 1/2$, then item i will get wrongly assigned to cluster(j). Else, if $\overline{X}_{ij}(M) - \Delta \leq 1/2$, then Algorithm 1 will terminate due to failure.

Similar argument holds if $i \in$ cluster(j), i.e. $X_{ij} \sim$ Bern(p). Since at least $\Omega(n)$ unique queries are made, if $M < \frac{\min p(1-p), q(1-q)}{\Delta^2}$, then either constant number of items get misclassified or the algorithm terminates due to failure. Thus, each Query(i, j) is made $\Omega(\frac{1}{\Delta^2})$ times and since at least $\Omega(nK)$ unique queries are made, $\Omega(\frac{nK}{\Delta^2})$ queries are needed to guarantee the success of Algorithm 1 with probability at least $1 - 1/poly(n)$.

C.2 Proof of Corollary 1 and Theorem 9

Define the event:

$$\mathcal{E}_\psi := \{\left|\overline{X}_{ij}(t) - \mathbb{E}(\overline{X}_{ij}(t))\right| \leq \psi(t), \forall t \geq 1, \forall 1 \leq i < j \leq n\}\ .$$

We will use the following finite LIL bound (see Lemma 1 in [Jam+14]) to show that event $\mathcal{E}_\psi$ holds with high probability:

Lemma C.2.1 (Finite LIL Bound [Jam+14] for Bernoulli Random Variables)**.** *Let $X(1), X(2), \cdots$, be i.i.d. Bernoulli random variables. For any $\zeta \in (0, 1)$ and $\delta \in (0, \log(1+\zeta)/e)$,*

$$\mathbf{Pr}\left(\left|\overline{X}(t) - \mathbb{E}[\overline{X}(t)]\right| \geq \psi(t), \textit{ for any } t \geq 1,\right) \leq \delta', \tag{C.2.1}$$

where $\psi(t) := (1+\sqrt{\zeta})\sqrt{\frac{(1+\zeta)}{2t}\log\left(\frac{\log(1+\zeta)t}{\delta}\right)}$ and $\delta' := 2^{\frac{2+\zeta}{\zeta}}\left(\frac{\delta}{\log(1+\zeta)}\right)^{(1+\zeta)}$.

When $\mathcal{E}_\psi$ holds, assuming all the previous assignments of items to clusters have been correct, consider the assignment of an yet to be assigned item i. We will first bound the number of repetitions T_{ij} of Query(i,j) required to either assign i to cluster(j) or decide $i \notin$ cluster(j). If $i \in$ cluster(j), then $X_{ij} \sim$ Bern(p). When $\mathcal{E}_\psi$ holds, $\forall t \geq 1, -\psi(t) \leq \overline{X}_{ij}(t) - p \leq \psi(t) \Rightarrow -2\psi(t) + p \leq \overline{X}_{ij}(t) - \psi(t) \leq p$. If for some T_{up}, $-2\psi(T_{up}) + p > 1/2$, then we have $\overline{X}_{ij}(T_p) - \psi(T_p) > 1/2$. So our goal is find an upper bound T_{up} such that $2\psi(T_{up}) < p - 1/2$. Let $\Delta := \min\{p -$

$1/2, 1/2-q\}$. Let $T_{up} := \frac{b_1}{\Delta^2}\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)$. Define, $\alpha(t) := \frac{1}{t}\log\left(\frac{\log(1+\zeta)t}{\delta}\right)$. So, $\psi(t) = (1+\sqrt{\zeta})\sqrt{\frac{1+\zeta}{2}\alpha(t)}$.

$$
\begin{aligned}
\alpha(T_{up}) &= \frac{\Delta^2}{b_1}\frac{1}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{1}{\delta}\log\left\{(1+\zeta)\frac{b_1}{\Delta^2}\log\left(\frac{1}{b_3\delta}\log\frac{b_2}{\Delta}\right)\right\}\right\} \\
&\overset{(a)}{\leq} \frac{\Delta^2}{b_1}\frac{1}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{1}{\delta}\log\left\{\frac{(1+\zeta)b_1}{b_2^2b_3\delta}\left(\frac{b_2}{\Delta}\right)^3\right\}\right\} \\
&= \frac{\Delta^2}{b_1}\frac{1}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{3}{\delta}\log\left\{\left(\frac{(1+\zeta)b_1}{b_2^2b_3\delta}\right)^{1/3}\frac{b_2}{\Delta}\right\}\right\} \\
&\overset{(b)}{\leq} \frac{\Delta^2}{b_1}\frac{1}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{3}{\delta}\left(\frac{(1+\zeta)b_1}{b_2^2b_3\delta}\right)^{1/3}\log\left(\frac{b_2}{\Delta}\right)\right\} \\
&= \frac{\Delta^2}{b_1}\frac{1}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{3\left(\frac{(1+\zeta)b_1b_3^2}{b_2^2}\right)^{1/3}\frac{1}{b_3\delta^{4/3}}\log\left(\frac{b_2}{\Delta}\right)\right\} \\
&\overset{(c)}{\leq} \frac{\Delta^2}{b_1}\frac{3\left(\frac{(1+\zeta)b_1b_3^2}{b_2^2}\right)^{1/3}}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{1}{b_3\delta^{4/3}}\log\left(\frac{b_2}{\Delta}\right)\right\} \\
&\overset{(d)}{\leq} \frac{\Delta^2}{b_1}\frac{3\left(\frac{(1+\zeta)b_1b_3^2}{b_2^2}\right)^{1/3}\frac{4}{3}}{\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)}\log\left\{\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right\} \\
&= \frac{\Delta^2}{b_1}4\left(\frac{(1+\zeta)b_1b_3^2}{b_2^2}\right)^{1/3}\log\left\{\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right\},
\end{aligned}
$$

where (a) follows from $\log x \leq x, \forall x > 0$, $(b),(c)$ follows from $\log(ax) \leq a\log x$, when $x > 2, a > 2$ applied twice and (d) follows from $\log(x^a y) \leq a\log(xy)$, when $y > 1, a > 1$.

Substituting in $\psi(T)$, we obtain the following inequality:

$$
\psi^2(T_{up}) = (1+\sqrt{\zeta})^2\frac{1+\zeta}{2}\alpha(T_{up})^2 \leq \frac{\Delta^2}{4}\left[(1+\sqrt{\zeta})^2(1+\zeta)\frac{8}{b_1}\left(\frac{(1+\zeta)b_1b_3^2}{b_2^2}\right)^{1/3}\right]. \tag{C.2.2}
$$

For (b) and (c) to go through we need the following inequalities to hold:

$$\frac{b_2}{\Delta} > 2 \tag{C.2.3}$$

$$\left(\frac{(1+\zeta)b_1}{b_2^2 b_3 \delta}\right)^{1/3} > 2 \tag{C.2.4}$$

$$\frac{1}{b_3 \delta^{4/3}} \log\left(\frac{b_2}{\Delta}\right) > 2 \tag{C.2.5}$$

$$\left(\frac{(1+\zeta)b_1 b_3^2}{b_2^2 \delta}\right)^{1/3} > 2 \tag{C.2.6}$$

and for (d) to go through the following inequality has to hold:

$$\frac{1}{b_3} \log\left(\frac{b_2}{\Delta}\right) > 1. \tag{C.2.7}$$

Further, to guarantee $\psi(T_{up}) < \Delta/2$, the following inequality has to hold:

$$\left[(1+\sqrt{\zeta})^2(1+\zeta)\frac{8}{b_1}\left(\frac{(1+\zeta)b_1 b_3^2}{b_2^2}\right)^{1/3}\right] < 1. \tag{C.2.8}$$

Choose: $\delta = 1/n^c$ and

$$b_1 = 3,\ b_2 = (1+\zeta)^2,\ b_3 = \frac{1}{(2(1+\sqrt{\zeta}))^3}. \tag{C.2.9}$$

By substitution into inequality C.2.8, the LHS is $\frac{2}{3^{2/3}} < 1$.

By definition, $\Delta < 1/2$ and together with $\zeta > 0 \Rightarrow b_2 = (1+\zeta)^2 > 1$, inequality C.2.3 is satisfied.

For $c \geq 3$, choosing $\delta = 1/n^c$, with the choice of b_1, b_2, b_3 as in (C.2.9), the inequalities C.2.4, C.2.5, C.2.6 will hold for $n > 7$.

LHS of the inequality C.2.7 is at least $2^3 \log 2 > 1$.

Similar arguments hold good when $i \notin$ cluster(j), i.e, when $X_{ij} \sim$ Bern(q), when $\mathcal{E}_\psi$ holds, $2\psi(T_{up}) + q < 1/2$ and hence $\overline{X}(T_{up}) + \psi(T_q) < 1/2$. Thus we have the following upper bound on T_{ij},

$$T_{ij} \leq \frac{b_1}{\Delta^2} \log\left(\frac{1}{b_3 \delta} \log\left(\frac{b_2}{\Delta}\right)\right), \tag{C.2.10}$$

where $\delta = 1/n^c$, $b_1 = 3,\ b_2 = (1+\zeta)^2,\ b_3 = \frac{1}{(2(1+\sqrt{\zeta}))^3}$.

As the maximum number of unique queries that will be made is nK, the total number of queries made is $\mathcal{O}\left(nK \frac{b_1}{\Delta^2} \log\left(\frac{1}{b_3\delta} \log\left(\frac{b_2}{\Delta}\right)\right)\right)$. Also, with c such that

$\delta = 1/n^c \in (0, \log(1+\zeta)/e)$ and $\delta' := n^2 2^{\frac{2+\zeta}{\zeta}} \left(\frac{\delta}{\log(1+\zeta)}\right)^{1+\zeta}$, from Lemma C.2.1, event $\mathcal{E}_\psi$ occurs with probability at least $1-\delta'$. Further, choosing c sufficiently large will ensure that the error probability δ' decays as $1/\text{poly}(n)$.

Setting $\delta = 1/n^c$ and choosing b_1, b_2, b_3 as in Theorem 9, is valid for inequalities (b), (c), and $\psi^2(T_{up}) \leq \Delta^2/4$ to hold[1]. Similar arguments hold good when $i \notin$ cluster(j), i.e, when $X_{ij} \sim$ Bern(q), when $\mathcal{E}_\psi$ holds, $2\psi(T_{up}) + q < 1/2$ and hence $\overline{X}(T_{up}) + \psi(T_{up}) < 1/2$. Thus we have the following upper bound on T_{ij},

$$T_{ij} \leq \frac{b_1}{\Delta^2} \log\left(\frac{1}{b_3\delta} \log\left(\frac{b_2}{\Delta}\right)\right), \tag{C.2.11}$$

where $\delta = 1/n^c$, $b_1 = 3$, $b_2 = (1+\zeta)^2$, $b_3 = \frac{1}{(2(1+\sqrt{\zeta}))^3}$. As the maximum number of unique queries that will be made is nK, the total number of queries made is $\mathcal{O}\left(nK\frac{b_1}{\Delta^2}\log\left(\frac{1}{b_3\delta}\log\left(\frac{b_2}{\Delta}\right)\right)\right)$. Also, with c such that $\delta = 1/n^c \in (0, \log(1+\zeta)/e)$ and $\delta' := n^2 2^{\frac{2+\zeta}{\zeta}} \left(\frac{\delta}{\log(1+\zeta)}\right)^{1+\zeta}$, from Lemma C.2.1, event $\mathcal{E}_\psi$ occurs with probability at least $1-\delta'$. Further, choosing c sufficiently large will ensure that the error probability δ' decays as $1/\text{poly}(n)$. Substituting $c = 4$ and $\zeta = 0.1151$ gives the simplified result in Theorem 9.

C.3 Pseudocode

Pseudocodes for the algorithms in Chapter 6 are provided below.

[1] See supplementary material for more details.

Algorithm 1 Active querying for crowdsourced clustering, with known p and q

Input: V: Set of items, $\Delta = \frac{1}{2}(p-q)$, $M = \lceil \frac{7 \log n}{2\Delta^2} \rceil$.
Output: $\mathbf{C}$: Clusters.
Initialize: Pick an item i uniformly at random from V and set $\mathbf{C} = \{\{i\}\}$ and $V \leftarrow V - \{i\}$.
while $V \neq \phi$ **do**
    Pick $v \in V$ uniformly at random.
    for each $\mathbf{C}_k \in \mathbf{C}$ **do**
        Pick $u \in \mathbf{C}_k$ uniformly at random. Set $\overline{X}_k(0) = 0$.
        for $t = 1$ to M **do**
            $X_{vu}(t) \leftarrow$ Query (v, u), $\overline{X}_k(0) = 0$.
        end for
        if $\overline{X}_{vu}(M) - \Delta > q$ **then**
            Then $v \in \mathbf{C}_k$, $\mathbf{C}_k \leftarrow \mathbf{C}_k \cup \{v\}$. Set flag = 1. Break.
        else if $\overline{X}_{vu}(M) + \Delta < p$ **then**
            Then $v \notin \mathbf{C}_k$. Break.
        end if
        if flag = 1 **then**
            Break.
        end if
    end for
    if $v \notin \mathbf{C}_k \; \forall \mathbf{C}_k \in \mathbf{C}$ **then**
        $\mathbf{C} \leftarrow \mathbf{C} \cup \{v\}$
    end if
    $V \leftarrow V - \{v\}$
end while
return $\mathbf{C}$

Algorithm 2 Active querying for crowdsourced clustering

Input: V: Set of items, c, η, δ.
Output: $\mathbf{C}$: Clusters.
Initialize: Pick an item i uniformly at random from V and set $\mathbf{C} = \{\{i\}\}$ and $V \leftarrow V - \{i\}$.
while $V \neq \phi$ **do**
    Pick $v \in V$ uniformly at random. Set flag = 0.
    for each $\mathbf{C}_k \in \mathbf{C}$ **do**
        Pick $u \in \mathbf{C}_k$ uniformly at random. Set $\bar{X}_k(0) = 0$, $t = 1$.
        while true **do**
            $X_t \leftarrow$ Query (v, u).
            $\bar{X}_k(t) = \frac{t-1}{t}\bar{X}_k(t-1) + \frac{1}{t}X_t$, $\psi(t) := (1+\sqrt{\eta})\sqrt{\frac{1+\eta}{2t}\log\left(\frac{\log((1+\eta)t)}{\delta}\right)}$.
            if $\bar{X}_k(t) - \psi(t) > \frac{1}{2}$ **then**
                Then $v \in \mathbf{C}_k$, $\mathbf{C}_k \leftarrow \mathbf{C}_k \cup \{v\}$. Set flag = 1. Break.
            else if $\bar{X}_k(t) + \psi(t) < \frac{1}{2}$ **then**
                Then $v \notin \mathbf{C}_k$. Break.
            end if
            $t \leftarrow t + 1$.
        end while
        if flag = 1 **then**
            Break.
        end if
    end for
    if $\mathbf{v} \notin \mathbf{C}_k$, $\forall \mathbf{C}_k \in \mathbf{C}$ **then**
        $\mathbf{C} \leftarrow \mathbf{C} \cup \{v\}$
    end if
    $V \leftarrow V - \{v\}$
end while
return $\mathbf{C}$