THESIS

MAXIMUM CLIQUE AND EDGE BI-CLIQUE PROBLEMS VIA MATRIX DECOMPOSITION AND TRUNCATED NUCLEAR NORM

WITS
UNIVERSITY

Submitted by

Salma D.A.A. Omer

School of Computer Science and Applied Mathematics

In fulfillment of the requirements

For the Degree of Doctor of Philosophy

University of the Witwatersrand,

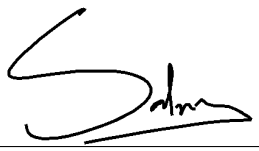
Johannesburg, South Africa

August 2022

Supervisor: Professor Montaz Ali

DECLARATION

As the undersigned here, I declare that this thesis and its contents are my own original work, except where citations are made. It is being submitted for the degree of Doctor of Philosophy at the University of the Witwatersrand, Johannesburg. This work has never been submitted before to any other university for an academic degree or examination.



Salma Dafa Allah Ahmed Omer

August 31, 2022.

ABSTRACT

MAXIMUM CLIQUE AND EDGE BI-CLIQUE PROBLEMS VIA MATRIX DECOMPOSITION AND TRUNCATED NUCLEAR NORM

In network science, cliques and bi-cliques are important structures well suited to several applications, including community detection, clustering of gene expression data, social network analysis, code theory, and bioinformatics.

In this work, we consider the problem of identifying the planted clique, the planted edge bi-clique in graphs, and the maximum clique and maximum edge bi-clique in random graphs. We have proposed mathematical models for these problems. The model for each problem resembles the matrix decomposition of the adjacency matrix of a given graph. The objective function of one of the proposed mathematical models includes the nuclear norm and a weighted ℓ_1-norm of the low-rank and sparse matrix of the decomposition, respectively. The weighted ℓ_1-norm has an advantage over the known ℓ_1-norm in reducing the error. The use of dynamically changing the weights for the ℓ_1-norm has been motivated. We have used proximal operators within the iterates of the ADMM (Alternating Direction Method of Multipliers) algorithm to solve the optimization problem. We have also proposed another mathematical model which is based on the truncated nuclear norm and the weighted ℓ_1-norm for finding the planted clique, the planted edge bi-clique in graphs, and the maximum clique and maximum edge bi-clique in random graphs. Furthermore, the optimization problems are solved by a two-step ADMM algorithm. The algorithms have been implemented for both the nuclear norm-based matrix decomposition model as well as the truncated nuclear norm-based matrix decomposition model. Convergence of the proposed ADMM algorithms for both models has been provided. One of the main features of our algorithms is that both algorithms require no input from the user other than the adjacency matrix of the input graph. The theoretical guarantee of the planted clique, the maximum clique, the planted edge bi-clique, and the maximum

edge bi-clique, in the form of the low-rank matrix, has also been established using approximate dual certificates constructed via golfing scheme for the nuclear norm based matrix decomposition model. We have suggested conditions that guarantee the recovery and uniqueness of the solution, as well as a tight bound on the dual matrix that validates optimality conditions.

We have tested our ADMM algorithm for numerous graphs, including graphs in which a clique or a bi-clique is planted and the remaining edges are added with probability p, random graphs where no cliques or bi-cliques are being planted, and real-world graphs. Numerical results for planted cliques and bi-cliques are presented showing clear advantages of our model when compared with two recent mathematical models. Results are also presented for randomly generated graphs with minimal errors. These errors are found using a formula we have proposed based on the sizes of the clique and the bi-clique. Moreover, we have applied our algorithm to real-world graphs for which cliques have been recovered successfully.

We have also tested our algorithm on truncated nuclear norm problem with different graphs, including graphs with a planted clique or bi-clique, and random graphs where no cliques or bi-cliques are planted. Numerical results for both graphs show clear advantages of our model when compared with the 'regular' truncated nuclear norm model.

ACKNOWLEDGEMENTS

Firstly, I am deeply grateful to Almighty Allah for protecting my life, making it possible to complete this journey, and for His many blessings. Thank you almighty Allah.

In addition, my sincere appreciation and thanks go out to Professor Montaz Ali, my PhD advisor, for the constant encouragement, support, and exceptional patience he provided during my PhD studies. It is an honor and a privilege to have had the opportunity of learning from his remarkable mathematical skills, generosity, and kindness. I will always be indebted to him for introducing me to a wide range of useful and high-impact researches that would impact my academic career and allow me to become a mature and self-dependent researcher. Indeed, without his generous assistance and support, I could not have completed this thesis.

During the last four years at the University of the Witwatersrand, I have made some good friends. My gratitude goes to all of them. I am extremely grateful to the staff and faculty members of the Department of Computational and Applied Mathematics.

I am beholden to my parents, my uncle Abd Alwahab, and my aunt Dr. Nagwa. Through their endless love, encouragement, and prayers, I was able to achieve my career goal in South Africa. In my family, my brothers, my sister in law, and my niece were a source of joy and happiness. Thanks to them for caring for and being close to my parents. Wishing my parents, brothers, my sister in law, and my niece a lifetime of happiness.

I would like to thank the Organization for Women in Science for the Developing World (OWSD) and the Swedish International Development Cooperation Agency (Sida) for their generous financial support during my PhD studies at Witwatersrand University.

Dedicated to my parents.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The Maximum Clique (MC) and Maximum Edge Bi-clique (MEB) problems are known for their flexibility in modeling a wide range of combinatorial and real-world problems. The objective of this Chapter is to present the MC and MEB problems and briefly explain the motivations for solving them. We also discuss this thesis' contributions. This Chapter concludes by summarizing the rest of this thesis.

1.1 Problem Statement and Motivation

During the past few years, optimization has expanded in all directions at a staggering rate, and it has now become an essential research tool to address a range of practical problems successfully, including problems from finance, economics, industrial operations, and telecommunications.

In this thesis, we have studied combinatorial optimization problems with a finite but often large set of possible solutions. These problems, however, are known to be very difficult to solve. More specifically, real-world problems that fall into the category of NP-hard problems are inefficiently solved.

A clique induces the complete subgraph on a simple graph. Identifying the clique with the greatest number of nodes in a graph is referred to as the MC problem, a widely known and widely studied combinatorial optimization problem, with applications in social network analysis [147, 164], community detection [10, 146, 188], bioinformatics [121, 168], coding theory [61], chemoinformatics [79], computational chemistry, analysis of protein-protein interaction graphs and others. A social network, for example, consists of nodes representing people, while undirected edges denote relationships among individuals (friendship, communication, working together, and so on). Accordingly, cliques refer to groups of individuals connected through a particular relation. Consequently, a MC matches the largest group of mutually connected individuals. Furthermore, the MC problem is closely related to several other significant combinatorial problems, including

community detection in complex networks, graph clustering [48], graph coloring [191], and vertex cover [29].

Bi-clique is a fundamental structure of bipartite graphs. This structure is often used in a number of bipartite applications to capture cohesive sub-graphs. The MEB problem consists of determining the bi-clique with the greatest number of edges (greatest number of nodes for the Maximum Vertex Bi-clique (MVB)) in a bipartite graph. Peeters [148] has demonstrated the NP-completeness of the MEB problem. However, Johnson [95] has proven that with a bipartite input graph the MVB problem can be solved in polynomial time. On the other hand, the problem is NP-complete when the input graph is not bipartite [194].

This thesis mainly focuses on the Planted Clique Problem (PCP), the problem of identifying the MC in random graphs which we refer to as MCP (the Maximum Clique Problem) in random graphs, and the problem of recovering the MC in real-world graphs. In addition, we have considered the Planted Edge Bi-clique Problem (PEBP), and the Maximum Edge Bi-clique Problem (MEBP) in random graphs. We have developed two matrix decomposition models for all these problems. The first model is based on the nuclear norm and the weighted ℓ_1-norm, and the second one is based on the truncated nuclear norm and the weighted ℓ_1-norm. Once decomposed the adjacency matrix into the low-rank and sparse component, for all these problems, the low-rank component corresponds to rank-one matrix. For a given graph with a planted (hidden) clique (respectively bi-clique) we wish to recover the planted clique (respectively bi-clique). For random graphs, the goal is to obtain the largest clique (respectively MEB).

The MC problem is known to be NP-hard, and it belongs to Karp's 21 NP-complete problems [100]. Finding $\omega(G)$, the number of the vertices in MC, is also known as NP-hard problem [100]. In light of the theoretical and practical importance of the MC problem, extensive research and several approaches have been committed to solving the MC problem. In general, the most exact efficient techniques are determined from the branch-and-bound (B&B) scheme. These techniques guarantee that the solution found is optimal. The MC problem being computationally complex, exact techniques often require an adequate amount of computing time and thus they are only ap-

plicable to very small problems. For large size problems, the optimal solutions cannot be reached in a certain complexity. To provide good sub-optimal solutions with less complexity, several meta-heuristic and heuristic algorithms have been devised. This manuscript provides two approaches that can be applied to large-scale MC problem and MEB problem efficiently.

1.2 Contributions

The main contributions of this thesis involve two general mathematical models, namely, the NN-based (Nuclear Norm) matrix decomposition model and the TNN-based (Truncated Nuclear Norm) matrix decomposition model, for the PCP, the PEBP, and the MCP and the MEBP in random graphs. Because of the mathematical modelling approach we have taken, a single model represents all four problems (PCP, PEBP, MCP, and MEBP). The contribution also includes two general ADMM based algorithms that can recover low-rank matrices in the context of above problems, specifically, rank-one matrices. Detailed descriptions of efficient implementations of our proposed algorithms are provided.

The problems we considered are related to graph theory, but we have considered mathematical models associated with matrix decomposition approach, hence we have taken theoretical approach from both fields. We have studied the convergence of the ADMM algorithms. For the uniqueness and recovery, we have used golfing scheme associated with the principle component analysis. From both the theoretical and practical perspectives, the fundamental contributions of this thesis contain seven points, which can be summarized as follows:

- We consider the model of exact low-rank and sparse matrix decomposition (the 'regular' model) as a model for solving the PCP, the PEBP, and the MCP and the MEBP in random graphs.

- We have suggested a mathematical model, which is based on the NN and the weighted ℓ_1-norm, for PCP (respectively PEBP) and the MCP (respectively MEBP) in random graphs. The proposed model resembles the matrix decomposition of the adjacency matrix of the input graph. Our proposed model differs from the known matrix decomposition model in that our

objective function includes a new surrogate function (non-convex) that better approximates the ℓ_0-norm than the one commonly used, the ℓ_1-norm.

- Based on the assumptions of specific standard identifiability conditions for the sparse and low-rank components of decomposition, we have established conditions that guarantee the recovery and the uniqueness of the solution. In addition, we have obtained a tight bound of the dual matrix that certifies the optimality conditions of our proposed model.

- We have evaluated the 'regular' matrix decomposition and our proposed model and compared them with recent algorithms.

- We have proposed a second mathematical model for the PCP, the PEBP, and the MCP and the MEBP in random graphs which is based on TNN and the weighted ℓ_1-norm.

- We have developed an ADMM based algorithm for the NN and the weighted ℓ_1-norm model, and demonstrated that the proposed algorithm approaches to the optimal solution. We have also developed a two-step ADMM based algorithm for the TNN and the weighted ℓ_1-norm model, and showed that the proposed algorithm converges to the optimal solution.

- Moreover, we have suggested a new expression for error calculations for the MCP and the MEBP in random graphs for both models.

1.3 Outline of the Thesis

The outline of the remaining parts of this thesis is as follows.

- In Chapter 2, we start with some definitions of graph theory. We review a few terminologies and basic concepts related to matrix analysis. We then discuss some definitions and fundamentals of convex analysis.

- Chapter 3 covers the Rank Minimization Problem (RMP) and its reformulations, and applications. We also review the Nuclear Norm Minimization Problem (NNMP), which is the

convex relaxation of the RMP. We present some properties of the NN, the optimality conditions, and numerical methods of the NNMP.

- Chapter 4 gives a brief overview and some applications of the MC problem. We have also discussed the existing algorithms, approaches, and mathematical models for the MC problem. We end this Chapter with a brief discussion of the MEB problem and its reformulations.

- In Chapter 5, we have presented some of the contributions of this thesis, including a new (non-convex) surrogate function for the ℓ_0-norm, and introduced our new models for the MC and the MEB problems using the matrix decomposition. The associated ADMM based algorithms are also presented here. We have also provided proof for the convergence for the algorithms to the optimal value of optimization models.

- In Chapter 6, we have provided our theoretical guarantee and recovery for the NN-based matrix decomposition model for the PCP and the MCP in random graphs. We have established our main theorem based on some assumptions regarding the incoherence conditions of the low-rank matrix and other assumptions on the random signs and the support of the sparse matrix. We have proven the related theorems by establishing some optimality conditions for proving the uniqueness of the solution and by the constructing the dual certificates of the low-rank and the sparse matrices. We have used the golfing scheme and the least square method for constructing the dual certificates. At the end of this Chapter, we have validated both certificates.

- In Chapter 7, we have provided comprehensive numerical experiments to evaluate the impact and performance of our suggested algorithms. We have demonstrated that our models can solve the PCP, the PEBP, and the MCP and the MEBP in random graphs. In addition, we have compared the results of the 'regular' matrix decomposition model and some recent algorithms to our proposed model. The results have shown that our model performs better than the regular model and recent algorithms. Moreover, we have compared the unweighed

and weighted versions of TNN-based model, where the later performs better than the first one.

- The last Chapter, of the thesis is devoted to making conclusions and proposing research directions for the future.

Some parts of Chapter 5, 6, and 7 were used for preparation of research articles [132], [133], and [134].

Chapter 2

Preliminaries and Notations

2.1 Introduction

We start with some definitions on graph theory. This is followed by an introduction to some basic concepts related to matrix analysis. After that we discuss tangent spaces and orthogonal projections of low-rank and sparse matrices. Then we proceed to some definitions and fundamentals of convex functions and convex analysis. To conclude, we introduce the Alternating Direction Method of Multipliers (ADMM), one of the most effective convex optimization algorithms.

2.2 Graph Terminology

In this research, the main focus is on graph theory, which is a key tool in combinatorial (discrete) optimization, topology, operation research, and a host of other areas. Scientists have been using graph theory as an effective mechanism to identify and interpret problems in their respective fields.

In mathematics, graphs are used to formalize the network, which is a collection of related objects. Graphs are defined mathematically as ordered pairs, i.e., $G = (V, E)$ where V (or $V(G)$) is the set of nodes (also called vertices) and E (or $E(G)$) is the set of links (also called edges). If V and E are finite sets, then the graph G is finite. Two vertices are considered to be adjacent if they are joined by an edge. Vertex degree, $d(v)$, represents the number of vertices adjacent to a vertex v. $\Delta(G)$ defines the maximum degree of a vertex of G, while the lowest degree of a vertex of G is denoted by $\delta(G)$. The size of the graph is indicated by $|V(G)|$, or simply $|V| = N_1$, and it describes the number of nodes in a graph G; $|E(G)|$, or simply $|E| = N_2$, indicates the number of edges in G. $\bar{G} = (V, \bar{E})$ where $\bar{E} = \{(u, v) \big| \ u, v \in V,\ u \neq v,\ \textit{and}\ (u, v) \notin E\}$ is known as the complement of graph G.

Sub-graphs are constructed by eliminating some of the vertices and edges from the main graph. That is, $F = (V', E')$ is a sub-graph of G if $V'(F) \subseteq V(G)$, $E'(F) \subseteq E(G)$, where $V'(F)$ and $E'(F)$ are the sets of vertices and edges of F, respectively. If a vertex is connected to itself by an edge, then it is called a loop. When a graph includes no loop, it is described as a simple graph.

A graph can be represented by two types of matrices, namely, incidence matrices and adjacent matrices. The incidence matrix B is a 0-1 matrix of size $N_1 \times N_2$ whose kl-th entry is 1 if and only if the k-th vertex and the l-th edge are connected and 0 otherwise. The adjacency matrix A is a 0-1 matrix of size $N_1 \times N_1$ whose kl-th entry is 1 is and only if the $(k, l) \in E$, and 0 otherwise. Throughout this thesis we will work with the adjacency matrix for undirected graphs.

Figure 2.1 represents a graph $G = (V, E)$, having $N_1 = 6$ vertices and $N_2 = 10$ edges. The adjacency matrix of the graph G in Figure 2.1 is shown in Table 2.1.

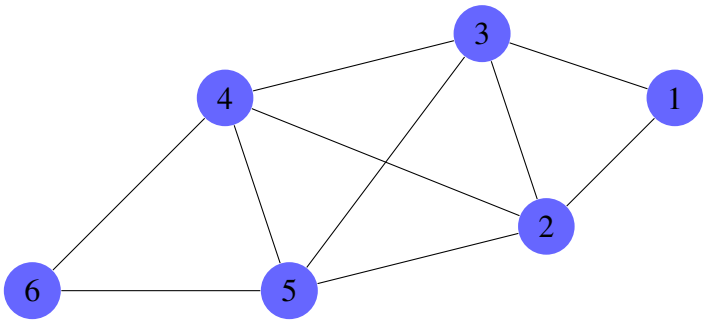


Figure 2.1: Graph $G = (V, E)$ with $|V| = 6$, $|E| = 10$

	1	2	3	4	5	6
1	0	1	1	0	0	0
2	1	0	1	1	1	0
3	1	1	0	1	1	0
4	0	1	1	0	1	1
5	0	1	1	1	0	1
6	0	0	0	1	1	0

Table 2.1: Adjacency matrix of graph G

According to the size of E, graphs can be categorised into null graphs, which has empty set of edges, and a complete graph, which has the greatest number of edges, i.e., $|E| = \frac{N_1(N_1-1)}{2}$, which means that each pair of vertices are mutually joined.

The vetrex set of a complete subgraph is called a clique. In a graph G, a Maximum Clique (MC) is a complete sub-graph of G with the greatest number of vertices. $\omega(G)$ is the clique number of the MC and denotes the number of vertices in the MC. A maximal clique of G is a complete sub-graph of G which is not included in any other complete sub-graph of G. For instance, Figure 2.1

has three cliques, $C_1 = \{4, 5, 6\}$, $C_2 = \{2, 3, 4, 5\}$, and $C_3 = \{1, 2, 3\}$, where C_2 is the MC, since it has 4 vertices.

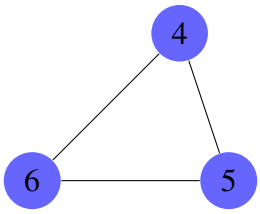


Figure 2.2: Clique C_1 of G

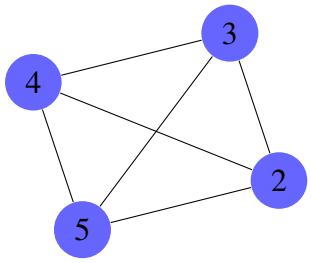


Figure 2.3: Clique C_2 of G

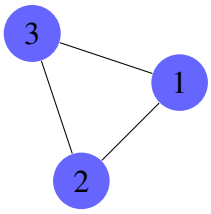


Figure 2.4: Clique C_3 of G

Opposite to a clique is a co-clique, known as an independent set (stable set), in which each two vertices are not adjacent.

The proper coloring of a graph involves assigning colors to the vertices so that there is no repeating color between adjacent vertices. A k-coloring of a graph is a proper coloring using a total of k colors, i.e., a k-coloring of a graph $G = (V, E)$ is a mapping ϕ of V into the set of colors $\{1, 2, ..., k\}$, for which any adjacent nodes $u, v \in V$, $\phi(u) \neq \phi(v)$. A graph G is declared to be k-colorable if G has k-coloring. For a non-empty graph G, $\chi(G)$ is the least integer k such that G is k-colorable, it is known as the chromatic number.

Proposition 2.2.1 ([143]). *The clique number $\omega(G)$, for any graph G, is a lower bound on its chromatic number $\chi(G)$, that is, $\omega(G) \leq \chi(G)$.*

2.3 Basics in Vector and Matrix Analysis

We have presented some of the common vector, matrix properties, and notations that will be used throughout this thesis. Let $\mathbb{R}^n$ be the n-dimensional Euclidean space. Then the inner product of vector $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^n$ is denoted by $\langle u, v \rangle = u^\top v = \sum_{k=1}^{n} u_k v_k$, where $\top$ is the transpose operation. When a function $\|.\| : \mathbb{R}^n \longrightarrow \mathbb{R}_+$ satisfies the following criteria, it is called a vector norm:

1. Non-negativity: $\|u\| \geq 0$, $\forall u \in \mathbb{R}^n$, and $\|u\| = 0$, provided $u = 0$,

2. Homogeneity: $\|\beta u\| = |\beta|\|u\|$, $\forall u \in \mathbb{R}^n$, and any scalar $\beta \in \mathbb{R}$,

3. Sub-additivity (Triangular inequality): $\|u + v\| \leq \|u\| + \|v\|$, $\forall u, v \in \mathbb{R}^n$.

The Euclidean norm of $u \in \mathbb{R}^n$ is expressed by $\|u\|_2 = (u^\top u)^{\frac{1}{2}}$. The Euclidean norm belongs to a class of p-norm (or ℓ_p-norm), is one of the most widely used vector norms. The ℓ_p-norm is defined by

$$\|u\|_p = \left(\sum_{k=1}^{n} |u|^p\right)^{\frac{1}{p}}, \textit{ for all } p \geq 1;$$

in the case of $p = 1$, we have what is termed a ℓ_1-norm, $\|u\|_1 = \sum_{k=1}^{n} |u_k|$; $\|u\|_\infty$ indicates the ℓ_∞-norm of $u \in \mathbb{R}^n$, which is described as the most significant component in magnitude of u, i.e., $\|u\|_\infty = \max_i |u_i|$.

The Holder inequality is a well known inequality concerning the ℓ_p-norms:

$$|u^\top v| \leq \|u\|_{p_1}\|v\|_{p_2}, \quad \frac{1}{p_1} + \frac{1}{p_2} = 1, \quad \textit{for any } u, v \in \mathbb{R}^n.$$

Cauchy-Schwartz inequality is a particular case of the Holder inequality,

$$|u^\top v| \leq \|u\|_2\|v\|_2.$$

It has been mentioned in [78] that all the p-norms are equivalent. For instance, if $v \in \mathbb{R}^n$, then

$$\|v\|_2 \leq \|v\|_1 \leq \sqrt{n}\|v\|_2,$$
$$\|v\|_\infty \leq \|v\|_2 \leq \sqrt{n}\|v\|_\infty,$$
$$\|v\|_\infty \leq \|v\|_1 \leq n\|v\|_\infty.$$

Assume that Euclidean space of the set of $N_1 \times N_2$ matrices is expressed by $\mathbb{R}^{N_1 \times N_2}$. Let $A, B \in \mathbb{R}^{N_1 \times N_2}$ then the inner product in $\mathbb{R}^{N_1 \times N_2}$ is cast as $\langle A, B\rangle = \mathrm{Tr}(A^\top B) = \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} A_{ij}B_{ij}$,

where $\mathrm{Tr}(A)$ is the trace of A (it corresponds to the sum of elements on the main diagonal), and it satisfies:

$$\langle A^\top, B\rangle = \langle A, B^\top\rangle = \langle B, A^\top\rangle = \langle B^\top, A\rangle, \;\; \forall A, B \in \mathbb{R}^{N_1\times N_2}.$$

The matrix norm definition is analogous to the definition of a vector norm. Thus, $\|.\| : \mathbb{R}^{N_1\times N_2} \longrightarrow \mathbb{R}_+$ is a matrix norm if for any $A, B \in \mathbb{R}^{N_1\times N_2}$ the following properties hold:

1. $\|A\| \geq 0$, and $\|A\| = 0$, provided $A = 0$,
2. $\|\beta A\| = |\beta|\|A\|$, any scalar $\beta \in \mathbb{R}$,
3. $\|A + B\| \leq \|A\| + \|B\|$.

Some norms also satisfy the sub-multiplicative property $\|AB\| \leq \|A\|\|B\|$.

The third inequality is known as the triangular inequality, while the reverse triangle inequality is described by $\|A - B\| \geq \Big|\|A\| - \|B\|\Big|$.

The norm concerned with the inner product is the Frobenius norm, which belongs to the most widely used matrix norms and is defined by

$$\sqrt{\langle A, A\rangle} = \|A\|_F = \left(\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}|a_{ij}|^2\right)^{\frac{1}{2}} = \left(\sum_{l=1}^{\min\{N_1,N_2\}}\sigma_l^2(A)\right)^{\frac{1}{2}}, \tag{2.1}$$

where $\sigma_l(A)$ is the l-th singular value of A. The cardinality of entries of A that are not zero is known as the ℓ_0-norm, denoted $\|A\|_0$. However, it is not a norm, because it does not satisfies all the above properties of the definition of norm. The ℓ_1-norm for $A \in \mathbb{R}^{N_1\times N_2}$ is denoted by

$$\|A\|_1 = \sup_{u\neq 0}\frac{\|Au\|_1}{\|u\|_1} = \max_{1\leq j\leq N_2}\sum_{i=1}^{N_1}|a_{ij}|, \tag{2.2}$$

while the infinity norm is defined by

$$\|A\|_\infty = \sup_{u \neq 0} \frac{\|Au\|_\infty}{\|u\|_\infty} = \max_{1 \leq i \leq N_1} \sum_{j=1}^{N_2} |a_{ij}|. \tag{2.3}$$

However, throughout this thesis, we utilize $\|A\|_\infty$ to characterize the largest element of A in magnitude. The ℓ_2-norm, also known as the spectral norm for $A \in \mathbb{R}^{N_1 \times N_2}$ corresponds to the greatest singular value of A, that is

$$\|A\| = \max_{k \in \{1,2,\ldots,\min\{N_1,N_2\}\}} \sigma_k(A) = \left(\lambda_{\max}(A^\top A)\right)^{\frac{1}{2}}, \tag{2.4}$$

where $\lambda_{\max}$ is the greatest eigenvalue of $A^\top A$, whereas the Nuclear Norm (NN) is denoted by

$$\|A\|_* = \sum_{k=1}^{\min\{N_1,N_2\}} \sigma_k(A). \tag{2.5}$$

The following relationships hold for the matrix norms

$$\begin{aligned}
&\|A\| \leq \|A\|_F \leq \sqrt{\min\{N_1, N_2\}}\|A\|, \\
&\frac{1}{\sqrt{N_2}}\|A\|_\infty \leq \|A\| \leq \sqrt{N_1}\|A\|_\infty, \\
&\frac{1}{\sqrt{N_1}}\|A\|_1 \leq \|A\| \leq \sqrt{N_2}\|A\|_1, \\
&\|A\|^2 \leq \|A\|_1\|A\|_\infty,
\end{aligned}$$

where the above norms are defined in (2.1)-(2.5).

The Frobeious norm, NN, and the spectral norm are related by the following inequality [154],

$$\|A\| \leq \|A\|_F \leq \|A\|_* \leq \sqrt{\min\{N_1, N_2\}}\|A\|_F \leq \min\{N_1, N_2\}\|A\|.$$

There exists a dual norm $\|.\|_d$ for any norm, $\|.\|$, [154] defined by

$$\|A\|_d = \sup_B \{\mathrm{Tr}(A^\top B) \mid \|B\| \leq 1\}.$$

In particular, the dual norm for the ℓ_1-norm is the infinity norm, the dual norm for the NN and the Frobenious norm are the spectral norm and the Frobenious norm, respectively.

Now, for $A = (a_{ij}) \in \mathbb{R}^{N_1 \times N_2}$, let us define its support by $supp(A) = \{(i,j) \mid a_{ij} \neq 0\}$, that is, the support of A is expressed as the cardinality of entries in A which are not zero. For $P, Q \in \mathbb{R}^{N_1 \times N_2}$, $P \circ Q$ describes the Hadamard product of P and Q, with elements given by $(P \circ Q)_{ij} = (P)_{ij}(Q)_{ij}$, for $1 \leq i \leq N_1, 1 \leq j \leq N_2$.

The signum function (also called the sign function) of any scalar a is defined by:

$$\text{sign}(a) = \begin{cases} 1, & \textit{if } a > 0 \\ 0, & \textit{if } a = 0 \\ -1, & \textit{if } a < 0. \end{cases}$$

2.4 Rank and Singular Value Decomposition

The largest number of linearly independent columns (or rows) in a matrix is specified as the rank of a matrix, and it is represented by r. The rank of a matrix is zero when all its elements are zero. In general, the rank of a matrix cannot be more than the dimension of its columns or rows space, that is $r \leq \min\{N_1, N_2\}$. A matrix has a full rank when $r = \min\{N_1, N_2\}$.

Theorem 1 (Rank plus Nullity is Columns [14]). *For any $Q \in \mathbb{R}^{N_1 \times N_2}$, we have,*

$$N_2 = \text{rank}(Q) + \mathcal{N}(Q),$$

$\mathcal{N}(Q)$ is the dimension of the nullity of Q, defined by

$$nullity(Q) = \{u \in \mathbb{R}^{N_2} \mid Qu = 0\}.$$

The SVD (Singular Value Decomposition) is a methodology for decomposing a matrix into three components (or matrices). It is widely used for solving large systems [53, 161]. The SVD theorem is stated below.

Theorem 2 (SVD [78]). *If H is a real $N_1 \times N_2$ matrix, then there exist orthogonal matrices, $\bar{U} \in \mathbb{R}^{N_1 \times N_1}$, $\bar{V} \in \mathbb{R}^{N_2 \times N_2}$, and a diagonal matrix $\bar{\Sigma} \in \mathbb{R}^{N_1 \times N_2}$, such that*

$$H = \bar{U}\bar{\Sigma}\bar{V}^\top,$$

where $\bar{U} = \{u_1|u_2|...|u_{N_1}\}$, with $\bar{U}^\top\bar{U} = \bar{U}\bar{U}^\top = I_{N_1}$, $\bar{V} = \{v_1|v_2|...|v_{N_2}\}$, with $\bar{V}^\top\bar{V} = \bar{V}\bar{V}^\top = I_{N_2}$, I_{N_1} and I_{N_2} are identity matrices of sizes N_1 and N_2, respectively. $\bar{U}$ (respectively V) denotes the left (respectively the right) singular vectors. $\bar{\Sigma}$ is described by

$$\Sigma_{ij} = \begin{cases} \sigma_i, & \text{if } i = j \leq \min\{N_1, N_2\}, \\ 0, & \text{Otherwise}, \end{cases}$$

where $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_{\min\{N_1,N_2\}}$, σ_i represents the i-th singular value of H.

If $H \in \mathbb{R}^{N \times N}$ is a real symmetric matrix, then

$$H = U\Sigma U^\top,$$

where $UU^\top = I = U^\top U$. Furthermore, if H has non-negative eigenvalues, then the eigenvalues and singular values are the same. The above decomposition is known as the full SVD. The reduced form of SVD (also known as the compact form) is defined as $H = U\Sigma V^\top$, $U \in \mathbb{R}^{N_1 \times r}$, $\Sigma \in \mathbb{R}^{r \times r}$, $V \in \mathbb{R}^{N_2 \times r}$.

Corollary 1 ([78]). *If $H \in \mathbb{R}^{N_1 \times N_2}$ with $\mathrm{rank}(H) = r$, then*

$$H = \sum_{i=1}^{r} \sigma_i u_i v_i^\top.$$

2.5 Tangent Spaces and Orthogonal Projections

We present two spaces of matrices along with the orthogonal projections onto them. Assume that $H \in \mathbb{R}^{N_1 \times N_2}$ has SVD given by $H = U\Sigma V^\top$, $U = [u_1, u_2, ..., u_r] \in \mathbb{R}^{N_1 \times r}$, $V =$

$[v_1, v_2, ..., v_r] \in \mathbb{R}^{N_2 \times r}$, and $\Sigma = \text{Diag}(\sigma_1, \sigma_2, ..., \sigma_r) \in \mathbb{R}^{r \times r}$ where σ_l is the l-th singular value of H, r is the rank of H. Let $\mathcal{B} = \{B \in \mathbb{R}^{N_1 \times N_2} \mid \text{rank}(B) \leq r\}$ be the set of matrices constrained by rank. Thus, the tangent space to the set $\mathcal{B}$ at H is the linear space of matrices that have identical row and column spaces to those of H. Let $\mathcal{R}$ indicate the linear space of matrices

$$\mathcal{R}(H) := \left\{UF^\top + GV^\top, F \in \mathbb{R}^{N_2 \times r},\ G \in \mathbb{R}^{N_1 \times r}\right\}. \tag{2.6}$$

If $H \in \mathbb{R}^{N \times N}$, is a symmetric real matrix, then,

$$\mathcal{R}(H) := \left\{UF^\top + GU^\top, F,\ G \in \mathbb{R}^{N \times r}\right\}. \tag{2.7}$$

The orthogonal projection $\mathcal{P}_\mathcal{R}$ onto $\mathcal{R}$, is as follows:

$$\mathcal{P}_\mathcal{R}(Y) = UU^\top Y + YVV^\top - UU^\top YVV^\top, \quad \forall Y, \tag{2.8}$$

and $\mathcal{P}_\mathcal{R}^\perp(Y) = \left(\mathcal{I} - UU^\top\right) Y \left(\mathcal{I} - VV^\top\right)$ is the orthogonal complement projection onto $\mathcal{R}$, where $\mathcal{I}$ is the identity operator. For any symmetric matrix $Q \in \mathbb{R}^{N \times N}$, the orthogonal projection of Q onto $\mathcal{R}$, is given by:

$$\mathcal{P}_\mathcal{R}(Q) = UU^\top Q + QUU^\top - UU^\top QUU^\top,$$

and $\mathcal{P}_\mathcal{R}^\perp(Q) = \left(\mathcal{I} - UU^\top\right) Q \left(\mathcal{I} - UU^\top\right)$ is the orthogonal complement projection onto $\mathcal{R}$. For any matrix F, $\|\mathcal{P}_\mathcal{R}^\perp F\| \leq \|F\|$ holds, where $\|.\|$ denotes the spectral norm. Note that $\mathcal{P}_\mathcal{R}$ and $\mathcal{P}_\mathcal{R}^\perp$ are both self-adjoint, that is, $\mathcal{P}_\mathcal{R} = \mathcal{P}_\mathcal{R}^\top$, $\mathcal{P}_\mathcal{R}^\perp = \mathcal{P}_\mathcal{R}^{\perp\top}$ [33].

Consider a linear space of sparse matrices defined by

$$\Omega := \left\{C \in \mathbb{R}^{N_1 \times N_2} \mid |supp(C)| = m, \textit{with given locations}\right\}. \tag{2.9}$$

Define $\mathcal{P}_\Omega$ to be the orthogonal projection onto Ω, that is,

$$\mathcal{P}_{\Omega}(Y) = \begin{cases} Y, & \textit{if } Y \in \Omega \\ 0, & \textit{otherwise}, \end{cases}$$

then $\mathcal{P}_{\Omega}^{\perp}$ defined by $\mathcal{P}_{\Omega}^{\perp}(Y) = Y - \mathcal{P}_{\Omega}(Y)$ represents the orthogonal complement projection onto Ω.

2.6 Convex Sets and Functions

The set that includes all the points of any straight-line segment linking any two points of the set is known as a convex set. That is, a subset $\mathcal{C}$ of $\mathbb{R}^n$ is convex if for any arbitrary two points $(x_1 \neq x_2 \in \mathcal{C})$, the line segment linking them is also in $\mathcal{C}$, i.e.,

$$(1-\theta)x_1 + \theta x_2 \in \mathcal{C}, \;\; 0 \leq \theta \leq 1.$$

For instance, half-spaces are convex sets. For any $\zeta \in \mathbb{R}$ and any non-zero $u \in \mathbb{R}^n$, the set

$$H^+ = \{x \;\big|\; \langle x, u\rangle \geq \zeta\}, \quad H^- = \{x \;\big|\; \langle x, u\rangle \leq \zeta\},$$

are called closed half-spaces, while open half-spaces are defined equivalently by replacing $\geq, \leq$ by $>, <$.

Theorem 3 (Convex analysis[158])**.** *Any intersection of convex sets is convex.*

Assume that $\{u_1, u_2, ..., u_m\}$ is a finite subset of $\mathcal{C}$. The convex hull of $\{u_1, u_2, ..., u_m\}$, denoted $Con(\{u_1, u_2, ..., u_m\})$, consists of all the combinations of the form

$$Con(\{u_1, u_2, ..., u_m\}) = \left\{ \sum_{k=1}^{m} \theta_k u_k \Big| \;\; \theta_k \geq 0, \;\; \sum_{k=1}^{m} \theta_k = 1 \right\}.$$

A subset $\mathcal{K}$ of $\mathbb{R}^n$ is closed under non-negative scalar multiplication is known as a cone, that is, for any $u \in \mathcal{K}$, $u\beta \in \mathcal{K}$, $\beta > 0$. The cone that is also a convex set is called a convex cone.

We say that a function $f : \mathcal{C} \longrightarrow \mathbb{R}$ is a convex function if and only if

$$f\left((1-\theta)u_1 + \theta u_2\right) \leq (1-\theta)f(u_1) + \theta f(u_2), \;\; \forall u_1, u_2 \in \mathcal{C}, \;\; 0 \leq \theta \leq 1.$$

Let the epigraph of f be indicated by $epif = \{(u, \theta) \mid \theta \geq f(u), \; \forall u \in \mathbb{R}^n, \; \theta \in \mathbb{R}\}$. Then we say that f is a convex function if its epigraph is a convex set. A function $f : \mathcal{C} \longrightarrow \mathbb{R}^n$ is a concave function if $-f$ is convex, that is, f is concave if and only if

$$f\left((1-\theta)u_1 + \theta u_2\right) \geq (1-\theta)f(u_1) + \theta f(u_2), \;\; \forall u_1, u_2 \in \mathcal{C}, \;\; 0 \leq \theta \leq 1.$$

The domain of any function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is expressed by $dom(f) = \{u \in \mathbb{R}^n \;\; \big| \;\; f(u) < +\infty\}$. A convex function f is proper if and only if $dom(f)$ is a non-empty convex set; with $f(u) > -\infty, \forall u \in dom(f)$ and $f(u) < +\infty$ for at least one u. A convex function is improper when it is not proper.

The largest convex function $g(u)$ that majorizes $f(u)$ is referred to as the convex envelope of a function f, denoted $Conev(f)$, i.e.,

$$Conev(f) = \sup\left\{g(u) \,\big|g(u) \;\; \textit{is convex}, \; g(u) \leq f(u), \; \forall u \in \mathbb{R}^n\right\}.$$

Suppose that $f : \mathcal{C} \longrightarrow \mathbb{R}$ is a differentiable function. We say the function f is convex if and only if

$$f(u_2) - f(u_1) \geq \nabla f(u_1)^\top (u_2 - u_1), \tag{2.10}$$

is applicable to all $u_1, \; u_2 \in \mathcal{C}$. Note that if $\nabla f(u_1) = 0$, then, $\forall u_2 \in \mathcal{C}$, $f(u_2) \geq f(u_1)$, which means that u_1 is a global minimizer of f.

We say that $\nabla f(u_1)$ is a sub-gradient of f at u_1 if (2.10) holds for all $u_2 \in \mathcal{C}$. Furthermore, the set containing all the sub-gradients of f at u_1, indicated by $\partial f(u_1)$, is called the sub-differential of f at u_1. For differentiable convex functions, the sub-gradient and the gradient coincide, that is,

$$\partial f(u_1) = \{\nabla f(u_1) \mid f(u_2) \geq f(u_1) + \nabla f(u_1)^\top (u_2 - u_1), \ \ \forall u_2 \in \mathcal{C}\}.$$

However, a sub-gradient can still exist even when f is not differentiable function at u. For more about convexity, we refer to [22, 158].

For any sparse matrix $S \in \mathbb{R}^{N_1 \times N_2}$ supported on Ω in (2.9) and a constant matrix $C \in \mathbb{R}^{N_1 \times N_2}$, the sub-gradient of the ℓ_1-norm at $C \circ S$ takes the form $\text{sign}(C \circ S) + F$, $\mathcal{P}_\Omega(F) = 0$, $\|F\|_\infty \leq 1$. Also the sub-gradient of the NN at $L \in \mathbb{R}^{N_1 \times N_2}$ is of the form $UV^\top + W$, $\mathcal{P}_\mathcal{R} W = 0$, $\|W\| \leq 1$, where $\mathcal{R}$ is defined in (2.6). Moreover, the sub-gradient of the NN at a symmetric matrix L is of the form $UU^\top + W$, $\mathcal{P}_\mathcal{R} W = 0$, $\|W\| \leq 1$.

2.7 Convex Programming

In convex optimization problems, the aim is either to maximize a concave objective function, or minimize an objective function that is convex, with respect to some convex inequality and/or affine equality constraints. A convex optimization problem is expressed as follows:

$$\begin{aligned} &\textit{minimize } f(x) \\ &\textit{subject to } x \in \mathcal{C}, \end{aligned} \tag{2.11}$$

where $f : \mathcal{C} \longrightarrow (-\infty, +\infty]$ is a proper, convex function, defined over a convex set $\mathcal{C}$.

Standard convex optimization problem is written as

$$\textit{minimize } f(x) \tag{2.12a}$$

$$\textit{subject to } g_i(x) \leq 0, \ i = 1, 2, ..., m_1 \tag{2.12b}$$

$$h_i(x) = 0, \ i = 1, 2, ..., m_2, \tag{2.12c}$$

where $x \in \mathcal{C}$ is the optimization variable, $f : \mathcal{C} \longrightarrow \mathbb{R}$ is the objective function that should be minimized over all $x \in \mathcal{C}$, with $\mathcal{C} = \cap_{i=0}^{m_1} \mathcal{C}_{g_i} \cap_{i=0}^{m_2} \mathcal{C}_{h_i}$, $\mathcal{C}_{g_i}$, $\mathcal{C}_{h_i}$ are the domains for the inequality and

equality constraint functions, respectively. The inequalities in (2.12b) and equalities (2.12c) are called the inequality and equality constraints, respectively. Both $f(x)$ and g_i are convex functions, and $h_i(x)$ are affine for all $i = 1, 2, ..., m_1$, and $i = 1, 2, ..., m_2$ respectively. A point $x \in \mathcal{C}$ where both the constraints in (2.12b) and (2.12c) are satisfied known as a feasible point. The feasible region of problem (2.12), denoted $\mathcal{C}$, is the set containing all the feasible points, while the optimal solution, denoted o^*, of problem (2.12) is the minimum of the feasible set $\mathcal{C}$. That is

$$o^* = \inf\{f(x) \,|x \in \mathcal{C}\},$$

where $\inf$ of a set is the smallest element in the set. If $f(x^*) = o^*$, then the feasible point x^* is optimal. When h_i are not affine and/or g_i are not convex then problem (2.12) is a general non-linear optimization problem, which is difficult to solve globally in general.

Because of the format of the objective function and the constraint functions, the standard convex optimization problem can be effectively solved. Each optimization problem has a Lagrangian function corresponding to it. The Lagrangian function, is a function $\mathcal{L} : \mathcal{C} \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \longrightarrow \mathbb{R}$ of the form

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m_1} \lambda_i g_i(x) + \sum_{i=1}^{m_2} \mu_i h_i(x),$$

where $\mu = \{\mu_1, \mu_2, ..., \mu_{m_2}\}$, $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_{m_1}\}$ are the Lagrange multipliers associated with the equality and inequality constraints, respectively.

The optimization problem in the form of (2.12) is called the primal problem, which has been written using the Lagrangian function as

$$Primal_P = \inf_{x \in \mathcal{C}} \sup_{\lambda \in \mathbb{R}_+^{m_1}, \ \mu \in \mathbb{R}^{m_2}} \mathcal{L}(x, \lambda, \mu).$$

Each primal problem has its associated dual problem defined by

$$Dual_P = \sup_{\lambda \in \mathbb{R}_+^{m_1},\ \mu \in \mathbb{R}^{m_2}} \inf_{x \in \mathcal{C}} \mathcal{L}(x, \lambda, \mu).$$

The duality gap occurs when the primal optimal value, denoted $Primal_{P^*}$, is strictly larger than the dual optimal value, denoted by $Dual_{P^*}$, that is,

$$Primal_{P^*} > Dual_{P^*}.$$

When

$$Primal_{P^*} \geq Dual_{P^*},$$

holds, we call it the weak duality, and this holds even when the problem is not convex [21]. For the case of strong duality

$$Primal_{P^*} = Dual_{P^*},$$

holds. There are some results that develop the conditions for the strong duality to occur, simple one is Slater's condition [22], which states that, there exists $x^* \in \mathcal{C}$, such that,

$$g_i(x^*) < 0, \quad \forall i = 1, 2, ..., m_1$$
$$h_i(x^*) = 0, \quad \forall i = 1, 2, ..., m_2,$$

x^* is called strictly feasible point as the inequality constraints are satisfied with strict inequality. The following theorem holds.

Theorem 4 (Dual attainment [21])**.** *When Slater's condition holds for problem* (2.12)*, both dual and primal values are equivalent, and* $Dual_{P^*}$ *will be achieved if it is finite.*

Solving a zero duality gap problem is reduced to a collection of equations and inequalities that one can solve either analytically or numerically under some conditions known as Karush–Kuhn–Tucker (KKT) conditions [22, 109].

Let x^* be any primal optimal point of problem (2.12) and (λ^*, μ^*) is the dual optimal points. The KKT conditions for problem (2.12) are

$$0 \in \partial \left(f(x^*) + \sum_{i=1}^{m_1} \lambda_i^* g_i(x^*) + \sum_{i=1}^{m_2} \mu_i^* h_i(x^*) \right), \tag{2.13a}$$

$$\lambda_i^* g_i(x^*) = 0, \ \ i = 1, 2, ..., m_1, \tag{2.13b}$$

$$g_i(x^*) \leq 0, \ \ i = 1, 2, ..., m_1, \tag{2.13c}$$

$$h_i(x^*) = 0, \ \ i = 1, 2, ..., m_2, \tag{2.13d}$$

$$\lambda_i^* \geq 0, \ \ i = 1, 2, ..., m_1. \tag{2.13e}$$

Conditions (2.13a) are known as stationary conditions, (2.13b) is the complementary slackness conditions. The conditions in (2.13c), (2.13d) ensure that x^* is primal feasible, and they are known as the feasibility conditions, and (2.13e) are the dual feasibility conditions. We have the following result.

Theorem 5 ([63]). *In any optimization problem that has a differentiable objective function and constraint functions where strong duality holds, any pair of x^* and (λ^*, μ^*) must satisfy the KKT conditions* (2.13).

In general convex optimization problems, the KKT conditions are both sufficient and necessary conditions [63]. This is declared in the following theorem.

Theorem 6 ([63]). *In a convex optimization problem that has a differentiable objective function and constraint functions, if any (x^*, λ^*, μ^*) satisfies KKT conditions, then x^* is primal optimal, (λ^*, μ^*) is dual optimal, and the strong duality holds.*

2.8 Alternating Direction Method of Multiplier

The Alternate Direction Method of Multiplier (ADMM) is a method dedicated to solving a constraint convex optimization problem by spilling it into small problems that can be handled more easily. The standard ADMM solves problems in the form

$$\textit{minimize}_{x,\,y} \;\; F(x,y) = f_1(x) + f_2(y) \tag{2.14}$$

$$\textit{s.t.}\; Ax + By = c, \tag{2.15}$$

with two sets of variable $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, and a separable objective function $F(x,y)$, $f_1,\ f_2$ are convex functions, $A \in \mathbb{R}^{q\times n}$, $B \in \mathbb{R}^{q\times m}$, and $c \in \mathbb{R}^q$.

The Augmented Lagrangian of problem (2.14)-(2.15) is of the form:

$$\mathcal{L}_\rho(x,y,\mu) = f_1(x) + f_2(y) + \langle \mu, Ax + By - c\rangle + \frac{\rho}{2}\|Ax + By - c\|_F^2, \tag{2.16}$$

where $\rho > 0$ is the ADMM penalty parameter, and μ is the Lagrange multiplier parameter, also known as the dual variable. Thus, using (2.16), for $J = 0, 1, 2, ...$, the ADMM iterations are as follow:

$$x_{J+1} = \text{argmin}_x \;\; \mathcal{L}_\rho(x, y_J, \mu_J) \tag{2.17}$$

$$y_{J+1} = \text{argmin}_y \;\; \mathcal{L}_\rho(x_{J+1}, y, \mu_J) \tag{2.18}$$

$$\mu_{J+1} = \mu_J + \rho(Ax_{J+1} + By_{J+1} - c). \tag{2.19}$$

It is clear that the ADMM algorithm performs alternate minimization of x and y in a sequential fashion.

The ADMM is usually formatted in a different form by scaling the dual variable, which can sometimes be more convenient. Thus, the scaled form of the augmented Lagrangian is as follows:

$$\mathcal{L}_\rho(x, y, \mu) = f_1(x) + f_2(y) + \frac{\rho}{2}\|Ax + By - c + \beta\|_F^2 - \frac{\rho}{2}\|\beta\|_F^2, \tag{2.20}$$

where $\beta = \frac{1}{\rho}\mu$ is the scaled dual variable. Thus, the scaled ADMM consists of the following iterations:

$$\begin{aligned}
x_{J+1} &= \text{argmin}_x\, \rho\left(\frac{1}{\rho}f_1(x) + \frac{1}{2}\|Ax + By_J - c + \beta_J\|_F^2\right) \\
y_{J+1} &= \text{argmin}_y\, \rho\left(\frac{1}{\rho}f_2(y) + \frac{1}{2}\|Ax_{J+1} + By - c + \beta_J\|_F^2\right) \\
\beta_{J+1} &= \beta_J + \rho(Ax_{J+1} + By_{J+1} - c).
\end{aligned}$$

Under some assumptions of $f_1,\ f_2$, the iterates of ADMM satisfy [23]:

- Residual convergence, that is, $Ax_J - By_J - c \longrightarrow 0$, as $J \longrightarrow 0$.
- Objective convergence, that is, $f_1(x_J) + f_2(y_J) \longrightarrow f_1^* + f_2^*$, as $J \longrightarrow \infty$ where $f_1^* + f_2^*$ is the optimal primal objective value.
- Dual convergence, that is $\beta_J \longrightarrow \beta^*$, as $J \longrightarrow \infty$, where β^* is a dual optimal solution.

For problem (2.14)-(2.15), the sufficient and necessary conditions for optimality are the primal feasibility, i.e.,

$$Ax_* + By_* - c = 0,$$

and the dual feasibility,

$$\begin{aligned}
0 &\in \partial f_1(x_*) + A^\top \beta_* \\
0 &\in \partial f_2(y_*) + B^\top \beta_*,
\end{aligned}$$

where ∂ denotes the sub-differential operator as discussed in Section 2.6.

Proximal Operators

We now begin with the singular value thresholding and proximal operator which have been used in the iterates of ADMM.

Let X be a matrix of size $N_1 \times N_2$. Assume that the full SVD of X is defined by $X = \bar{U}\bar{\Sigma}\bar{V}^\top$, where $\bar{U} \in \mathbb{R}^{N_1 \times N_1}$, $\bar{V} \in \mathbb{R}^{N_2 \times N_2}$ and $\bar{\Sigma} \in \mathbb{R}^{N_1 \times N_2}$.

For $\tau > 0$, we define the SVT (Singular Value Thresholding) operator [27] as

$$SVT_\tau(X) = \bar{U}\mathcal{D}_\tau(\bar{\Sigma})\bar{V}^\top, \tag{2.21}$$

$\mathcal{D}_\tau(\bar{\Sigma}) = \mathrm{Diag}(\max\{(\sigma_i - \tau), 0\})$, taking note of the fact that $\sigma_i = |\lambda_i|$ (eigenvalues) and $\bar{U} = \bar{V}$ for real symmetric X.

Let STT_τ be the soft thresholding or shrinkage operator with parameter τ. Then for $\tau > 0$, STT_τ can be defined on each element of X by:

$$STT_\tau(X, O) = \mathrm{sign}(X) \circ \max(|X| - \tau O, 0), \tag{2.22}$$

where sign is the (element-wise) sign function introduced in Section 2.3; O is a matrix of all ones.

The proximal operator [145], is also known as proximal mapping, $Prox_f : \mathbb{R}^{N_1 \times N_2} \longrightarrow \mathbb{R}^{N_1 \times N_2}$ of a closed proper convex function, f, for every $X \in \mathbb{R}^{N_1 \times N_2}$ is defined by

$$\begin{aligned} Prox_{\tau f}(X) &= \mathrm{argmin}_{Y \in \mathbb{R}^{N_1 \times N_2}} \ \tau f(Y) + \frac{1}{2}\|Y - X\|_F^2 \\ &= \begin{cases} SVT_\tau(X) & if\ f(Y) = \|Y\|_*, \\ ST_\tau(X) & if\ f(Y) = \|Y\|_1, \end{cases} \end{aligned} \tag{2.23}$$

where, for a square matrix X, $ST_\tau(X) = STT_\tau(X, I)$ when $O = I$, I is the identity matrix.

It is easy to see that

$$\begin{aligned}\operatorname{argmin}_{Y\in\mathbb{R}^{N_1\times N_2}}\ \ & \tau\|C\circ Y\|_1+\frac{1}{2}\|Y-X\|_F^2\\ &= \operatorname{sign}(X)\circ\max(|X|-\tau C,0)=STT_\tau(X,C).\end{aligned}\tag{2.24}$$

The soft thresholding can also be cast, element-wise, as

$$ST_\tau(X)=\begin{cases}0, & \textit{if } |X|\le\tau\\ X-\tau\operatorname{sign}(X), & \textit{otherwise}.\end{cases}$$

Chapter 3

Rank Minimization Problem and Reformulation

3.1 Introduction

This Chapter aims to introduce the Rank Minimization Problem (RMP) and its formulations, which are central to the clique problem discussed in this thesis. We begin with the general RMP. Then we proceed with another known formulation of the RMP known as the affine rank minimization problem. We present some properties of the Nuclear Norm (NN), the optimality conditions of the Nuclear Norm Minimization Problem (NNMP); the NN is referred to as a convex relaxation of the RMP. We also present the conditions of the success of NNMP. Then we discuss some applications of the RMP. We conclude this Chapter by introducing some numerical methods applied to the NNMP.

3.2 The General Rank Minimization Problem

The rank-constrained optimization problems have extensively been studied in different research areas [43, 52, 135, 140, 170, 200]. This type of problem refers to optimizing a convex objective function with respect to the rank constraint, and some other convex constraints.

Over the past few years, it has become more popular to solve problems that concern minimizing a matrix's rank over a convex set. This is what is known as RMP. A wide variety of applications of RMP are found in system theory [65], noise removal in images [90], minimum-order system approximation [64, 66], and euclidean distance matrix problems [64].

Following is a general description of the RMP [66]:

$$\begin{aligned} &\textit{minimize} \quad \operatorname{rank}(Z) \\ &\textit{subject to } Z \in \mathcal{C}, \end{aligned} \tag{3.1}$$

where $Z \in \mathbb{R}^{N_1 \times N_2}$ is the optimization variable and $\mathcal{C}$ is a convex set. The problem in (3.1) is generally a difficult combinatorial optimization problem which is hard to solve because of the non-convex nature of the rank function. Various heuristics techniques have been proposed [65, 67, 98], such as alternating projections, interior-point-based methods, the log-det heuristic, and the trace heuristic to handle this problem.

For a positive semi-definite $Z \in \mathbb{R}^{N \times N}$, $Z \succeq 0$, the log-det heuristic replaces the rank of Z in (3.1) with a smooth surrogate function [67], $\log\, det(Z + \delta I)$; $\delta > 0$ is a small regularization parameter, thus it solves

$$\begin{aligned} &\textit{minimize} \quad \log\, det(Z + \delta I_N) \\ &\textit{subject to } Z \in \mathcal{C}, \end{aligned} \tag{3.2}$$

where I_N is the identity matrix. The log-det heuristic has also been generalised for any matrix $Z \in \mathbb{R}^{N_1 \times N_2}$. Thus, the minimization of the log-det function can be achieved using a local minimization method [67].

Alternatively, the trace heuristic solves the following problem [66]

$$\begin{aligned} &\textit{minimize} \quad \|Z\|_* \\ &\textit{subject to } Z \in \mathcal{C}, \end{aligned} \tag{3.3}$$

instead of (3.1), for $Z \in \mathbb{R}^{N_1 \times N_2}$

$$\|Z\|_* = \sum_{l=1}^{\min(N_1, N_2)} \sigma_l(Z),$$

where $\sigma_l(Z) = \sqrt{\lambda_l(Z^\top Z)}$ indicates the singular values of Z; λ_l is the l-th eigenvalue of $Z^\top Z$. An accurate description of the NN is the ℓ_1-norm of the singular values vector.

3.3 Affine Rank Minimization Problem

When the general RMP is defined on the affine space, the problem is then called the Affine Rank Minimization (ARM) problem. In other words, for a given linear map $\mathcal{A} : \mathbb{R}^{N_1 \times N_2} \longrightarrow \mathbb{R}^{N_3}$ and a vector $b \in \mathbb{R}^{N_3}$, the ARM problem is given by:

$$\begin{aligned} &\textit{minimize} \quad \operatorname{rank}(Z) \\ &\textit{subject to } \mathcal{A}(Z) = b, \end{aligned} \tag{3.4}$$

where $Z \in \mathbb{R}^{N_1 \times N_2}$. This problem seeks to reduce the rank of Z according to particular affine constraints. This is a well-known NP-hard problem and several heuristics have been proposed for ARM problem. We have focused on the relaxation of the problem in (3.4) by considering the NNMP proposed in [64, 66]. That is the relaxed problem:

$$\begin{aligned} &\textit{minimize} \quad \|Z\|_* \\ &\textit{subject to } \mathcal{A}(Z) = b. \end{aligned} \tag{3.5}$$

The above NNMP is a convex optimization problem that can be modelled as a Semi-Definite Program (SDP) and solved with SDP solvers [66, 154].

3.3.1 Characteristics of the NN

For $Z \in \mathbb{R}^{N_1 \times N_2}$, the NN of Z, $\|Z\|_*$, is the dual norm of the spectral norm of Z, $\|Z\|$ [154]. This is given by:

$$\|Z\|_* = \sup_Y \{\operatorname{Tr}(Y^\top Z) \big| \ \|Y\| \leq 1\},$$

where Tr is the trace function that has been introduced in Section 2.3, Chapter 2. It is known that the NN is the nearest function that approximates the rank function, as the NN is referred to as the convex envelope of the rank function [64, 66].

Theorem 7 ([66] Theorem 1). *The convex envelope of the function* $\Psi(Z) = \text{rank}(Z)$*, on the set* $\mathcal{C} = \{Z \in \mathbb{R}^{N_1 \times N_2} \big| \|Z\| \leq 1\}$*, is* $\Psi_{env}(Z) = \|Z\|_*$.

Note that, for $K > 0$, $\frac{1}{K}\|Z\|_*$ is the convex envelope of the rank of Z on the set $\{Z \big| \ \|Z\| \leq K\}$.

Corollary 2 ([5]). *The convex envelope of the function* $\Psi(Z) = \text{rank}(Z)$*, on the set* $\mathcal{C} = \{Z \in \mathbb{R}^{N_1 \times N_2} \big| \|Z\| \leq K\}$*, is* $\Psi_{env}(Z) = \frac{1}{K}\|Z\|_*$*, for all* $K > 0$.

Corollary 3 ([5, 154]). *Assume that* Z_0*,* Z_* *are the solutions of problem* (3.4) *and* (3.5)*, respectively, and let* $\|Z_0\| = K$*. Then,*

$$\frac{\|Z_*\|_*}{K} \leq \text{rank}(Z_0) \leq \text{rank}(Z_*).$$

3.3.2 Optimality Conditions for NNM Problem

Before we characterize the optimality conditions of the NNMP in (3.5), we need to define the sub-differential of NN.

Lemma 3.3.3. *[112, 154, 187] Let* $Z \in \mathbb{R}^{N_1 \times N_2}$ *be of rank* r *matrix and admit a compact SVD* $Z = U\Sigma V^\top$ *where* $U \in \mathbb{R}^{N_1 \times r}$*,* $V \in \mathbb{R}^{N_2 \times r}$*, and* $\Sigma \in \mathbb{R}^{r \times r}$ *is diagonal. Consequently, the sub-differential of NN at* Z *is equal to*

$$\partial\|Z\|_* = \{UV^\top + W \big| W^\top U = 0,\ WV = 0, \|W\| \leq 1\}.$$

The following theorem gives the optimality conditions for the NNMP in (3.5).

Theorem 8 ([5], Theorem 3.2.3, [154]). *For* (3.5)*, a matrix* $Z \in \mathbb{R}^{N_1 \times N_2}$ *is optimal if there exists* $z \in \mathbb{R}^{N_3}$ *provided that*

$$\mathcal{A}(Z) = b \tag{3.6}$$

$$\mathcal{A}^\top(z) \in \partial\|Z\|_*. \tag{3.7}$$

The condition in (3.6) ensures the feasibility of the affine constraint in problem (3.5), while (3.7) ensures there is no feasible improvement direction.

3.3.4 The NN Heuristic: Conditions for Success

A widely used heuristic algorithm typically uses the NN in place of the rank function. This section describes the necessary and sufficient conditions for determining whether this heuristic determines the lowest rank solution for a linear constraint set. These conditions are given concerning the Restricted Isometry Property (RIP), which is a basic property of a matrix that allows sparse matrices recovery and the null space property.

The Restricted Isometry Properties

Let Z_0 be a matrix of rank r, $\mathcal{A}(Z_0) = b$. Define

$$Z_* := \operatorname{argmin}_Z \|Z\|_*, \quad \textit{subject to} \quad \mathcal{A}(Z) = b.$$

In other words, Z_* is the element with the smallest NN in the affine space described by $\mathcal{A}$ and b. This section describes specific cases in which $Z_* = Z_0$ can be assured a priori. When the linear map $\mathcal{A}$ is confined to the set of matrices of rank r, the key conditions are defined by δ, a sequence of parameters that measures its behaviour. The following definition provides a natural extension of RIP to matrices from vectors [38].

Definition 3.3.1. *[154] Let $\mathcal{A} : \mathbb{R}^{N_1 \times N_2} \to \mathbb{R}^q$ be a linear map. With no loss of generality, let $N_1 \leq N_2$. The r-restricted isometry constant is the least number $\delta_r(\mathcal{A})$ for which every integer r with $1 \leq r \leq N_1$,*

$$(1 - \delta_r(\mathcal{A}))\|Z\|_F \leq \|\mathcal{A}(Z)\| \leq (1 + \delta_r(\mathcal{A}))\|Z\|_F, \tag{3.8}$$

applies for all matrices Z with a top rank of r.

Candes and Tao [38] state that RIPs for sparse vectors requires (3.8), where Frobenious norm and rank replace Euclidean norm and cardinality, respectively.

Following are two theorems that describe the capability of restricted isometry constants. The two theorems are almost generalized from the sparse case to the low-rank one. A generalization of Lemma 1.2 in [38] is given by the first theorem, which provides a characterization of the case when Z_0 denotes the lowest rank unique solution of $\mathcal{A}(Z) = b$.

Theorem 9 ([154], Theorem 3.2)**.** *For some integer $r \geq 1$, assume that $\delta_{2r} < 1$. Hence, Z_0 is the only matrix whose rank does not exceed r and is satisfied with $\mathcal{A}(Z) = b$.*

A simple condition ensures that $Z_* = Z_0$ is declared in the following theorem.

Theorem 10 ([154], Theorem 3.3)**.** *Let $r \geq 1$ be such that $\delta_{5r} < 1/10$. Then $Z_* = Z_0$.*

The Nearly Isometric Random Matrices.

This section illustrates how linear maps will follow RIP (3.8) when sampled from a family of probability distributions that follow particular inequalities with great deviation, when q, N_1, and N_2 approach infinity at appropriate rates. This family of random linear transformations can be described as follows.

Definition 3.3.2 ([154], Definition 4.1)**.** *Assume that the random operator $\mathcal{A}$ maps to values from linear maps from $\mathbb{R}^{N_1 \times N_2}$ to $\mathbb{R}^q$. For all $Z \in \mathbb{R}^{N_1 \times N_2}$, we declare that $\mathcal{A}$ is nearly isometric if*

$$\mathbb{E}\left[\|\mathcal{A}(Z)\|^2\right] = \|Z\|_F^2$$

and, for $0 < \varepsilon < 1$, one have

$$Pr\left(\left|\|\mathcal{A}(Z)\|^2 - \|Z\|_F^2\right| \geq \varepsilon \|Z\|_F^2\right) \leq 2\exp\left(-\frac{q}{2}\left(\frac{\varepsilon^2}{2} - \frac{\varepsilon^3}{3}\right)\right),$$

and, for all $t > 0$, we have

$$Pr\left(\|\mathcal{A}\| \geq 1 + \sqrt{\frac{N_1 N_2}{q}} + t\right) \leq \exp(-\gamma q t^2)$$

for some constant $\gamma > 0$.

Most of the nearly isometric random maps can be represented by random matrices. Given a linear map $\mathcal{A} : \mathbb{R}^{N_1 \times N_2} \to \mathbb{R}^q$, the matrix representation of $\mathcal{A}$ is expressed as [154]:

$$\mathcal{A}(Z) = A\, Vec(Z),$$

where A is a $q \times N_1 N_2$ matrix and $Vec(Z)$ is composed of the columns of Z stacked orderly one after another. This matrix representation will now discuss various examples of nearly isometric random matrices. One of the most widely known ensembles contains independent Gaussian entries [49]

$$A_{ij} \sim \mathcal{N}\left(0, \frac{1}{q}\right).$$

Additionally, two ensembles of matrices are presented in [1]. One has entries drowned by i.i.d. symmetric Bernoulli distribution,

$$A_{ij} = \begin{cases} \sqrt{\frac{1}{q}} & \textit{w.p.}\ 1/2, \\ -\sqrt{\frac{1}{q}} & \textit{w.p.}\ 1/2, \end{cases}$$

and two-thirds of the entries in the other one have zeros,

$$A_{ij} = \begin{cases} \sqrt{\frac{3}{q}} & \textit{w.p.}\ 1/6, \\ 0 & \textit{w.p.}\ 2/3, \\ -\sqrt{\frac{3}{q}} & \textit{w.p.}\ 1/6. \end{cases}$$

We have the following theorem.

Theorem 11 ([154], Theorem 4.2)**.** *Set* $0 < \delta < 1$*. Assuming* $\mathcal{A}$ *to be a nearly isometric random variable, so that, for every* $1 \leq r \leq N_1$*, there exist positive constants* $a_0,\ a_1$ *that only depend on* δ *such that,* $\delta_r(\mathcal{A}) \leq \delta$*, with probability at least* $1-\exp(-a_1 q)$*, when* $q \geq a_0 r(N_1+N_2)\log(N_1 N_2)$.

Exact recovery is possible with weaker assumptions regarding the r-restricted isometry constants compared to the assumptions of Theorems 9 and 10 [125, 126, 139].

Conditions of Null Space

For the ARM problem, Recht et al. [155, 156] established sufficient and necessary conditions for the NNMP based on the null space corresponding to the linear operator $\mathcal{A}$ defining the set of conditions. These conditions are mainly concerned when the optimal solution of (3.4) matches with that of (3.5). Whenever $q < N_1 N_2$, the null space of $\mathcal{A}$, that is the set of X such that $\mathcal{A}(X) = 0$, is not empty. Note that Z is an optimal solution for (3.5) if and only if for every X in the null space of $\mathcal{A}$

$$\|Z + X\|_* \geq \|Z\|_*.$$

Further, they demonstrate that when null space is sampled from a uniform distribution on subspace, the characterization of null space remains viable with a high probability, until a certain threshold of equality constraints is attained.

Dvijotham and Fazel [58] provide conditions of the exact recovery of the low-rank matrix under a property of the null space of $\mathcal{A}$ describing the set of the constraints called the spherical section property.

Definition 3.3.3. *[58] We say that* $Null(\mathcal{A})$ *satisfies the* Δ *spherical section property if* $\frac{\|X\|_*}{\|X\|_F} \geq \sqrt{\Delta}$ *for all* $X \in Null(\mathcal{A})$*,* $X \neq 0$.

3.4 Some Applications of Rank Minimization Problems

RMPs have recently gained much attention in both engineering and mathematical fields. Here are some applications of the matrix RMP.

3.4.1 The Matrix Completion Problem

The matrix completion problems study whether it is possible to "complete" a matrix that has some of its entries known. These problems are concerned with describing conditions under which we can fill unknown entries. The fulfilment of these conditions warrants that the low-rank matrices can be retrieved. The problem has many applications in numerous fields, such as the Collaborative Filtering with application in Netflix recommendations system [34], Genomic Data Integration [31], and Network Coding [86, 106].

In such applications, the matrix to be retrieved has a low rank or almost low rank structure. Assume that m entries of an $N_1 \times N_2$ matrix M are perceived. The aim is to retrieve the matrix M (that is, to complete the matrix M), generally structured as a low-rank matrix.

Let Ω be the set of locations of the observed entries of M, that is, $(i, j) \in \Omega$ if M_{ij} is observed. Then, the matrix completion problems can be cast as RMPs as follows:

$$\begin{aligned} &\textit{minimize} \quad \operatorname{rank}(Z) \\ &\textit{subject to} \quad Z_{ij} = M_{ij}, \ (i,j) \in \Omega, \end{aligned} \tag{3.9}$$

where Z is the optimization variable. As mentioned before, this problem is NP-hard, so the NN heuristic is proposed as a relaxation of this problem, which solves,

$$\begin{aligned} &\textit{minimize} \quad \|Z\|_* \\ &\textit{subject to} \quad Z_{ij} = M_{ij}, \ (i,j) \in \Omega. \end{aligned} \tag{3.10}$$

In many cases, the recovery of the low-rank matrix M is not guaranteed. For instance, if $M = e_i e_j^\top$; $e_i's$ are the canonical vectors, $i, j \in \{1, 2, ..., N\}$, $N = \max\{N_1, N_2\}$, then the recovery is not possible. This matrix is both low-rank and sparse. Clearly, some conditions on the singular vectors of M have to be satisfied in order to guarantee the recovery. In particular, theoretical guarantees have been provided for the NNMP (3.10) in [34, 39, 103]. For the exact

recovery of the low-rank M, theories have been developed based on the properties of the low-rank matrix M and the number of the observed entries [35, 36, 39, 45, 80]

Candes and Recht [35] showed that the NN heuristic in (3.10) recovers the low-rank matrix M of rank r, when there are m number randomly enough sampled entries. That is, there exist numerical constants η and τ such that if

$$m \geq \eta N^{\frac{5}{4}} r \log N,$$

the minimizer of problem (3.10) with probability at least $1 - \tau N^{-3} \log N$ is unique and equal to M. Furthermore, if $r \leq N^{\frac{1}{5}}$, then with probability at least $1 - \tau N^{-3} \log N$, the recovery is exact, under the condition that

$$m \geq \eta N^{\frac{6}{5}} r \log N.$$

To reduce the number of observations m necessary to recover the low-rank matrix M, one needs the singular vectors of M to be spread out. Hence, both the right and the left singular vectors must be uncorrelated with the canonical basis. This is known as the incoherence; we have the following definition.

Definition 3.4.1 ([127], Definition 2)**.** *Assume U denotes a subspace of $\mathbb{R}^N$ with dimension r, and the orthogonal projection onto it is P_U. Thus, the coherence condition of U, (regarding the canonical basis (e_i)) is as follows:*

$$\mu(U) \equiv \frac{N}{r} \max_{1 \leq i \leq N} \|P_U e_i\|_2^2, \quad \mu(U) \in \left[1, \frac{N}{r}\right],$$

where $P_U = UU^\top$.

Consider the singular value decomposition of $M \in \mathbb{R}^{N_1 \times N_2}$ of rank r as

$$M = U\Sigma V^\top = \sum_{i=1}^{r} \sigma_i u_i v_i^\top,$$

where $\sigma_1, \sigma_2, ..., \sigma_r$, are the singular values, the matrices $U = [u_1, u_2, ..., u_r]$, $V = [v_1, v_2, ..., v_r]$ are the left and the right singular vectors, respectively.

Therefore, incoherence conditions with parameter μ state:

$$\max_i \|U^\top e_i\|^2 \leq \frac{\mu r}{N_1}, \quad \max_i \|V^\top e_i\|^2 \leq \frac{\mu r}{N_2}, \tag{3.11}$$

are the so-called standard incoherence conditions [33]. The condition

$$\|UV^\top\|_\infty \leq \sqrt{\frac{\mu r}{N_1 N_2}}, \tag{3.12}$$

is the so-called the strong incoherence condition (also known as the joint incoherence condition). We are concerned with the case of $\mu(U)$ being small to ensure that the column and row space of the low-rank matrix are spread out and not concentrated in a few columns (or rows). Thus, a small number of observed entries of M, m, can be used to retrieve M. In fact, the following theorem holds.

Theorem 12 ([153], Theorem 1.1)**.** *Assume that M is an $N_1 \times N_2$ matrix of rank r with a singular value decomposition $U\Sigma V^\top$. For simplicity, assume that $N_1 \leq N_2$, Σ is $r \times r$, U is $N_1 \times r$ and V is $N_2 \times r$. Consider that*

$\boldsymbol{A}_0$ The column and row spaces have coherences bounded above by $\mu_0 > 0$.

$\boldsymbol{A}_1$ In the matrix $UV^\top$, the highest entry for some $\mu_1 > 0$ is bounded by $\left|\mu_1 \sqrt{\frac{r}{N_1 N_2}}\right|$.

Assume that we observe m entries of M, and the locations of the entries are randomly sampled. Consequently, if

$$m \geq 32 \max\{\mu_1^2, \mu_0\} r(N_1 + N_2)\beta \log^2(2N_2) \tag{3.13}$$

for some $\beta > 1$*, the minimizer to problem* (3.10) *with probability at least* $1 - 6\log(N_2)(N_1 + N_2)^{2-2\beta} - N_2^{2-2\beta^{\frac{1}{2}}}$ *is unique and equal to* M.

3.4.2 The Matrix Decomposition Model

Assume that we have a matrix that results from summing up an unknown low-rank matrix to an unknown sparse matrix. The decomposition process includes segmenting the provided matrix into its low-rank and sparse components. This problem arises in several applications including video surveillance [33, 129], background separation and image processing [41, 99, 137, 152, 171]. However, in practice finding an exact solution is NP-hard.

The exact recovery is obtained by solving a convex relaxation problem which employs the NN and the ℓ_1-norm to act as surrogates for low-rank and sparsity, respectively. For low-rank and sparse recovery, the authors of [45] suggested solving the convex optimization problem given below,

$$\min_{L,S} \|L\|_* + \gamma\|S\|_1 \tag{3.14}$$

$$\textit{s.t. } L + S = M, \tag{3.15}$$

where M is a given $N_1 \times N_2$ matrix to be decomposed; $L \in \mathbb{R}^{N_1 \times N_2}$ is the low-rank component; $S \in \mathbb{R}^{N_1 \times N_2}$ is the sparse component; $\gamma > 0$ is a constant that provides a trade-off between L and S, the decomposition components.

The problem (3.14)-(3.15) is referred to as RPCA (Robust Principal Component Analysis), which has several applications in bioinformatics, image analysis, and computer vision. Wright et al. [118, 189] have demonstrated that under relatively strong conditions, for a sufficiently sparse error matrix S (relative to L's rank), the low-rank matrix L can be precisely obtained from $M = L + S$ by solving the convex program (3.14)-(3.15).

The problem (3.14)-(3.15) can be transformed into SDP [185], for which polynomial-time SDP solvers exist. SDP solvers like SDPT3 and SeDuMi are based on the interior point method.

However, in many cases the matrix to be decomposed has large dimensions. As a matter of fact, extensive research has demonstrated that the interior point method can be numerically ineffective for large-scale and medium-scale problems [195].

Identifiability Issues

The matrix decomposition, as discussed in the introduction, can be a fundamentally ill-posed problem. This section describes two scenarios involving identifiability issues. Examples below illustrate what additional conditions need to exist to guarantee that a unique partitioning into low-rank and sparse matrices is possible:

- Suppose S_* be any sparse matrix, and $L_* = e_i v^\top$ where e_i indicates the i-th standard basis vector and v is an arbitrary vector. The existence of a unique decomposition may not be guaranteed in this situation. Because the low-rank L_* itself is very sparse, it is difficult to distinguish it from the other sparse matrix S_*. Therefore, to avoid the low-rank matrix becoming very sparse, we need conditions that guarantee that L_* is not sparse.

- Suppose L_* be any low-rank matrix, and $S_* = e_i v^\top$, where v is spread, e.g., $v = (1, 1, ..., 1)^\top$. Here, the sparse matrix S_* is of low-rank. Therefore, no method can recover the low-rank L_* exactly. It is, therefore, essential to assure that S_* is not a low-rank matrix.

To define the situation where exact recovery is possible, rank-sparsity incoherence concept has been introduced [33, 45] which links a matrix's sparsity pattern to the spaces of its rows and columns spaces via an uncertainty principle. The rank-sparsity incoherence concept is formed as standard and joint incoherence conditions given in (3.11) and (3.12), respectively. Tangent spaces introduced in Section 2.5 can also be used to define similar conditions. Chandrasekaran et al. [44, 45] proposed two quantities with regards to the tangent spaces of the sparse and the low-rank matrices to ensure that the low-rank is not too sparse and the sparse component can not have low rank. For any matrix A, these quantities are:

$$\zeta(A) \triangleq \max_{Z \in \mathcal{R}(A),\ \|Z\| \leq 1} \|Z\|_\infty, \quad \mu(A) \triangleq \max_{Z \in \Omega(A),\ \|Z\|_\infty \leq 1} \|Z\|,$$

where $\|Z\|_\infty = \max_{i,j} |Z_{ij}|$. Small value of $\zeta(A)$ implies that elements of $\mathcal{R}(A)$ are diffused which means that these elements are not very sparse. Consequently, A cannot be a very sparse matrix. Furthermore, the quantity $\mu(A)$ is small, each element of Ω has a diffused spectrum, which means that these elements have relatively small singular values. Considering a matrix $A \neq 0$, neither $\zeta(A)$ nor $\mu(A)$ can be small at the same time [45].

According to [44, 45], for any matrix for any $A \in \mathbb{R}^{N\times N}$, the smaller $\zeta(A)$, the less coherent the row/column spaces of A, i.e., we have

$$inc(A) \leq \zeta(A) \leq 2inc(A),$$

where $inc(A)$ is the incoherence of the row/column spaces of A, which is clearly expressed by:

$$inc(A) \triangleq \max \left[\max_i \|P_U e_i\|_2, \max_i \|P_V e_i\|_2 \right].$$

In addition, they have indicated that any sparse matrix A, with confined degree has small $\mu(A)$, i.e.,

$$deg_{min}(A) \leq \mu(A) \leq deg_{max}(A),$$

where $deg_{max}(A)$ is the largest number of non-zero entries per row/column of A and $deg_{min}(A)$ is the smallest number of non-zero entries per row/column of A. Their main results is as follows:

Theorem 13 ([44, 45]). *Given $M = L_* + S_*$ with*

$$\zeta(L_*)\mu(S_*) < 1/6,$$

the unique optimum $(\tilde{L}, \tilde{S})$ of (3.14)-(3.15) is (L_*, S_*) for the following range of γ (in the objective (3.14)):

$$\gamma \in \left(\frac{\zeta(L_*)}{1 - 4\zeta(L_*)\mu(S_*)}, \frac{1 - 3\zeta(L_*)\mu(S_*)}{\mu(S_*)} \right).$$

Candes et al. [33] provide some assumptions, in addition to incoherence conditions in (3.11) and (3.12), on the low-rank component and the sparse component. They have the following main result.

Theorem 14 ([33] Theorem 1.1)**.** *Let $\tilde{L}$ be $N \times N$, obeying both (3.11) and (3.12). Fix any $N \times N$ matrix Σ of signs. Consider the case where the support set Ω of $\tilde{S}$ is distributed uniformly for all sets of cardinality m, and that $\operatorname{sign}([\tilde{S}]_{ij}) = \Sigma_{ij}$ for all $(i, j) \in \Omega$. It follows that there exists a numerical constant c such that with probability at least $1 - cN^{-10}$ (depending on the choice of support), problem (3.14)-(3.15) with $\gamma = \frac{1}{\sqrt{N}}$ is exact, that is $L_* = \tilde{L}$ and $S_* = \tilde{S}$, where:*

$$\operatorname{rank}(L_*) \leq \rho_r N \mu^{-1} (\log N)^{-2}, \quad \textit{and} \quad m \leq \rho_s N^2,$$

where ρ_r, ρ_s are numerical constants.

Let (L_*, S_*) be the optimal solution of problem (3.14)-(3.15), that is $L_* + S_* = M$. In addition, assume that we have some foreknowledge of the tangent spaces $\mathcal{R}(L_*)$ and $\Omega(S_*)$. Transversal intersection of the tangent spaces $\mathcal{R}(L_*)$ and $\Omega(S_*)$ is a necessary and sufficient condition for unique recovery.

3.5 Numerical Methods Applied on NNMP

Within this section, we discuss numerical methods that can be applied to solve the NNMP. Presented here are some existing approaches grouped into four categories. In Chapter 5, we review one of these approaches in detail, an approach which is based on convex optimization, namely, the ADMM (Alternating Direction Method of Multipliers).

3.5.1 Iterative Thresholding

The Singular Value Thresholding (SVT) algorithm presented in [28] forms the basis of many popular numerical schemes for minimizing the NN, which arises in the low-rank matrix recovery problem such as the matrix completion problems. Conventionally, SVT is based on finding the SVD and then shrinking the singular values. A first-order iterative algorithm has been proposed, which produces a sequence of matrices $\{Z_J, X_J\}$; it usually operates soft thresholding procedures on singular values of X_J within each step. The MCP in (3.10) can be expressed as

$$\begin{aligned} &\textit{minimize} \ \ \|Z\|_* \\ &\textit{subject to} \ \ \mathcal{P}_\Omega(Z) = \mathcal{P}_\Omega(M), \end{aligned} \tag{3.16}$$

where $\mathcal{P}_\Omega$ is the orthogonal projection onto the space of matrices vanishing outside of Ω. Set $\rho > 0$, and set a sequence of scalar step sizes $\{\tau_J\}_{J\geq 1}$. The algorithm starts with $X = 0 \in \mathbb{R}^{N_1 \times N_2}$, then inductively defines

$$\begin{cases} Z_J = SVT_\rho(X_{J-1}) \\ X_J = X_{J-1} + \tau_J \mathcal{P}_\Omega(M - Z_J) \end{cases}$$

until the stopping criteria are met. SVT_ρ is the shrinking operator defined in (2.21).

3.5.2 Interior Point Method

The NNMP in (3.10) can be expressed as a semi-definite program [66, 116, 185]. The most common procedure for addressing an SDP is the interior point method. The SDP formulation for the problem in (3.10) is casted as follows [66, 116, 185]:

$$\min \frac{1}{2}\left(\mathrm{Tr}(Q)+\mathrm{Tr}(R)\right) \tag{3.17}$$

$$\textit{subject to } Y=\begin{bmatrix} Q & Z \\ Z^{\top} & R \end{bmatrix} \succeq 0, \tag{3.18}$$

$$Z_{ij}=M_{ij},\ (i,j)\in\Omega, \tag{3.19}$$

where $Q \in \mathbb{R}^{N_1\times N_1}$, $R \in \mathbb{R}^{N_2\times N_2}$ are the optimization variables. Here $Y \succeq 0$ denotes that Y is positive semi-definite. It is well know that when M is of small size (size less than 100×100), the SDP solvers, like SDPT3 [177] and SeDuMi [169], give accurate and robust solution [116, 154]. The interior point based methods often require a large amount of computation per iteration as well as high memory requirements as the above SDP in (3.17)-(3.19) has $N_1(N_1+1)/2+N_2(N_2+1)/2$ variables and it is very hard to solve with N_1 and N_2 approaching 100. The following Corollary applies to the structure at low-rank solutions.

Corollary 4 (Corollary [89])**.** *Let Z_* be the optimal for the problem in (3.10) with $\mathrm{rank}(Z_*)=r$. Let $Z_*=U\Sigma V^{\top}$ be the reduced (compact) SVD. Define*

$$Q=U\Sigma U^{\top},\ \ R=V\Sigma V^{\top},$$

and

$$Y=\begin{bmatrix} Q & Z^{\top} \\ Z & R \end{bmatrix}=\begin{bmatrix} U \\ V \end{bmatrix}\Sigma\begin{bmatrix} U \\ V \end{bmatrix}^{\top}.$$

Then we have $Y \succeq 0$ and optimal in (3.17)-(3.19) with $\mathrm{rank}(Y) =: r_Y = r$ and

$$\|Z_*\|_*=\frac{1}{2}\mathrm{Tr}(Y)=\mathrm{Tr}(\Sigma).$$

3.5.3 Proximal Gradient Algorithm

Several algorithms have been proposed for solving NN regularized least squares to determine low-rank data matrices from samples of entries, including the first-order proximal gradient algorithm and its accelerated versions [73, 178, 189]. According to this algorithm, an iteration is generated by combining the latest three points, namely, the previous point, the current point, and the gradient's proximal point. The accelerated proximal gradient algorithm achieves a better convergence rate of $O(\frac{1}{J^2})$ than the proximal gradient algorithm [13, 130], where J is the number of iterations.

Consider a general unconstrained convex optimization problem as follows:

$$\min_{Z} \Big\{ F(Z) = f(Z) + g(Z) \Big\}, \tag{3.20}$$

where $f(Z)$ is a smooth convex function and $g(Z)$ is a non-smooth convex function. The accelerated proximal gradient method constructs a quadratic approximation of the $F(Z)$ at a given point X as

$$Q_{\upsilon}(Z, X) = f(X) + \langle Z - X, \nabla f(X) \rangle + \frac{1}{2\upsilon} \|Z - X\|_F^2 + g(Z), \tag{3.21}$$

where υ is the step size. Thus, the accelerated proximal algorithm solves the problem in (3.21) iteratively by updating Z, X, and υ.

The matrix completion problem in (3.16) can be expressed as an unconstrained convex optimization problem as follows:

$$\textit{minimize} \;\; \|Z\|_* + \frac{\lambda}{2} \|\mathcal{P}_{\Omega}(Z) - \mathcal{P}_{\Omega}(M)\|_F^2,$$

λ is the regularization parameter. Thus, for the problem in (3.20), we have $f(Z) = \|Z\|_*$ and $g(Z) = \frac{\lambda}{2} \|\mathcal{P}_{\Omega}(Z) - \mathcal{P}_{\Omega}(M)\|_F^2$.

The accelerated proximal gradient algorithm has also been applied on the matrix decomposition model for the recovery of the low-rank and the sparse components of the input matrix [73, 189].

3.5.4 Alternating Direction Method of Multipliers

The ADMM algorithm has been presented in Section 2.8 is a first-order algorithm for solving separable convex optimization problems. This algorithm has been applied to solve the matrix decomposition model in (3.14)-(3.15) for recovering the sparse and the low-rank components of the given matrix [114, 195].

There are a number of applications of ADMM in convex programming and image processing, see for example [46, 59, 60, 71, 108, 183].

For the problem in (3.14)-(3.15) compared to the general convex optimization problem in (2.14)-(2.15) we have $f_1(L) = \|L\|_*$ and $f_2(S) = \|S\|_1$. Then, it solves the augmented Lagrangian in (2.16) by solving (2.17)-(2.19) alternately.

The augmented Lagrangian function for problem (3.14)-(3.15) is

$$\mathcal{L}_\rho(L, S, \mu) = \|L\|_* + \gamma\|S\|_1 + \langle \mu, M - L - S\rangle + \frac{\rho}{2}\|M - L - S\|_F^2,$$

where $\mu \in \mathbb{R}^{N_1 \times N_2}$ is the Lagrange multiplier of the linear constraint given in (3.15); $\rho > 0$ is the penalty parameter. The classical alternative direction of multiplier (see, e.g., [72, 77]) solves the following problems to get the new iteration:

$$\begin{cases} L_J \in \operatorname{argmin}_{L \in \mathbb{R}^{N_1 \times N_2}} \{\mathcal{L}_\rho(L, S_{J-1}, \mu_{J-1})\}, \\ S_J \in \operatorname{argmin}_{S \in \mathbb{R}^{N_1 \times N_2}} \{\mathcal{L}_\rho(L_J, S, \mu_{J-1})\}, \\ \mu_J = \mu_{J-1} + \rho(L_J + S_J - M), \end{cases}$$

where $J = 1, 2, 3, \ldots$. This, according to Yuan and Yang [195], is equivalent to

$$0 \in \partial(\|L_J\|_*) - [\mu_{J-1} + \rho(L_J - S_{J-1} - M)]\,, \tag{3.22}$$

$$0 \in \gamma\partial(\|S_J\|_*) - [\mu_{J-1} + \rho(L_J - S_J - M)]\,, \tag{3.23}$$

$$\mu_J = \mu_{J-1} + \rho(L_J + S_J - M). \tag{3.24}$$

The ADMM algorithm, according to [114, 195], with the same stopping condition as in Chapter 5, for low-rank and sparse matrix recovery is given by:

Algorithm 1 The ADMM algorithm for problem (3.14)-(3.15)

Input: Adjacency matrix M, regularization parameter $\mu > 0$ and penalty parameter $\rho > 0$

Initialization: Begin with feasible L_{J-1}, S_{J-1}, $\mu_{J-1} = 0$, and $J = 1$

WHILE Stopping condition not satisfied

- $$\begin{aligned} L_J &= \text{argmin}_L \left(\frac{1}{\rho}\|L\|_* + \frac{1}{2}\|L - S - M + \frac{1}{\rho}\mu\|_F \right) \\ &= SVT_{\frac{1}{\rho}}(M - S_{J-1} + \frac{1}{\rho}\mu_{J-1}), \end{aligned}$$

- $$\begin{aligned} S_J &= \text{argmin}_S \left(\frac{\gamma}{\rho}\|L\|_* + \frac{1}{2}\|L - S - M + \frac{1}{\rho}\mu\|_F \right) \\ &= ST_{\frac{\gamma}{\rho}}(M - L_J + \frac{1}{\rho}\mu_{J-1}), \end{aligned}$$

- Update μ_J via (3.24).

ENDWHILE

Output: The recovered matrix L_J and S_J.

For any $A \in \mathbb{R}^{N_1 \times N_2}$, the singular value thresholding operator $SVT_{\frac{1}{\rho}}(A)$ and the soft thresholding operator $ST_{\frac{\gamma}{\rho}}(A)$ have been introduced in Section 2.8 in (2.21) and (2.22), respectively.

Tao et al. [173] proposed an extended model of (3.14) to deal with problems with incomplete and noisy observations, the proposed model is casted as

$$\min_{L,S\in\mathbb{R}^{N_1\times N_2}} \|L\|_* + \gamma\|S\|_1 \tag{3.25}$$

$$\textit{s.t.}\ \|P_\Omega(M-L-S)\|_F \leq \delta,\ \ \delta > 0, \tag{3.26}$$

where $P_\Omega : \mathbb{R}^{N_1\times N_2} \to \mathbb{R}^{N_1\times N_2}$ is the orthogonal projection of the span of matrices that vanish outside of Ω. The authors in [173] proposed the general augmented Lagrangian method for solving (3.25), they also proposed the alternating splitting augmented Lagrangian method and its variants.

Chapter 4

The Maximum Clique and Bi-clique Problems

4.1 Introduction

Due to the importance and wide area of applications, the Maximum Clique (MC) problem attracted many researchers from various fields. Two research directions have been continuing in solving the MC problem. The first direction is to find mathematical models of MC problem that provide solution quickly. The second is to develop algorithms for the MC problem and related convergence theorems and bounds. In this Chapter, we present some of the applications of the MC problem, followed by the discussion on some existing algorithms and approaches. After that we present some bounds on the clique numbers followed by some mathematical models for the MC problem. Furthermore, we conclude this Chapter with a brief discussion of the Maximum Edge Bi-clique (MEB) problem.

4.2 The MC Problem

4.2.1 Applications of the MC Problem

Maximum clique problems are relevant in the real world because they are used in a broad range of applications in community detection [10, 146, 188], bio-informatics [121, 168], coding theory [61], chemo-informatics [79], social network analysis [147, 164], and others. Most of these problems can be modelled as a MC problem, or may have some sub-problems that require a solution of a MC problem.

In the world of combinatorial optimization and integer programming, there are numerous important clique problems, such as graph clustering [48], graph coloring [191], and vertex cover [29]. Considering the significance of MC problem, numerous theoretical and practical studies have been carried out, especially, ever since the second DIMACS implementation challenge had been devoted [94]. The challenge was focused on three NP-hard combinatorial optimization problems:

the problem of finding cliques in a graph, the problem of coloring the vertices of a graph, and the satisfiability problem [96, 97]. Below we have a brief description of some applications of MC problem.

Community detection [10, 146, 188]

In real-world networks, it is common to determine vertices (members) belonging to more than one group (community) at the same time. Naturally, people form groups, whether in their work-place, family, or friendship circles. In order to accomplish this task, there are several methods developed for identifying the mutually connected communities in a given network. Most of the proposed methods are based on the maximum clique problems.

Recently, research has focused on the analysis of complex real-world networks. By discovering communities, we can learn more about their structure and functionality. Community detection is generally achieved by using clustering algorithms based on social network analysis. As one can search for all cliques for all possible sizes in a given graph. Social networks are classic examples of a graph with communities. The following in Figure 4.1 is a graph with its n-clique communities for $n = 3$ and $n = 4$.

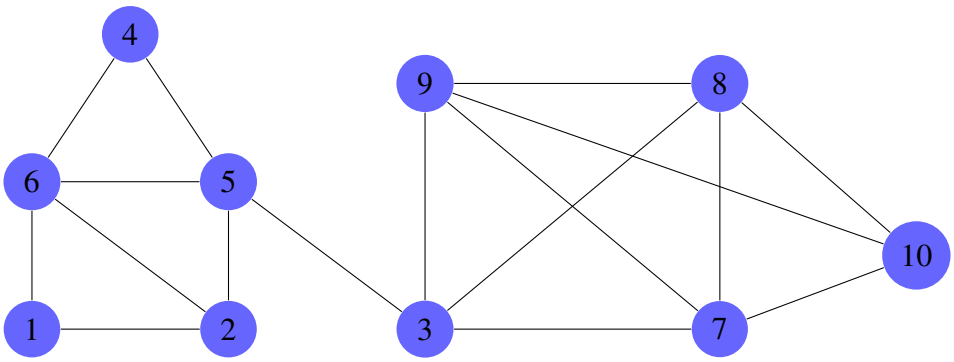


Figure 4.1: 3-clique and 4-clique communities

As shown in the above figure that the graph includes 3-clique communities, however, the right hand-side part of the graph has a 4-clique community.

Social network analysis [147, 164]

Consider a social network that is composed of vertices that represent people, and edges that represent their mutual acquaintances. Then, the MC problem is to identify the largest group of people who know each other. Systematically examining all subsets is the only way to figure out which group of people all know each other, but this can take a long time for social networks with more than a few dozen members.

Social network analysis involves measuring, analysing, and mapping the connections and relationships among individuals or people, groups, organisation, computers, and other entities associated with information and knowledge. An aspect of social network analysis is analysing the interaction between the people and users in the case of online social networks.

An important social network structure is a group of users formed through a mutually connected relationships, which is known as cliques. Therefore, the identification of MCs in social networks is important for analysing certain groups and communities.

The following figure illustrates an example of a terrorists network as a social network, which shows the connection (represented by the edges) between the terrorists (represented by the vertices) who were responsible for the September 11, 2001 attack at the World Trade Centre [175].

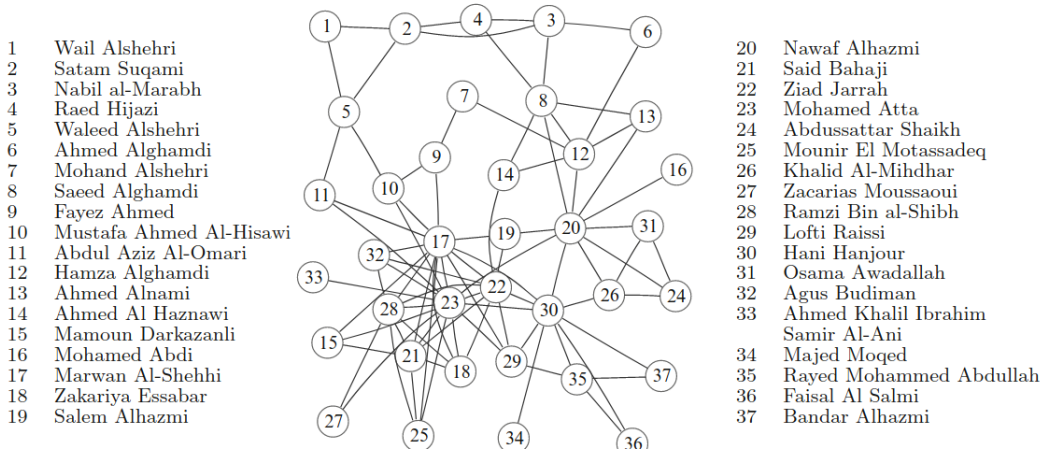


Figure 4.2: Terrorists network illustrates the connection between the terrorists associated with September 11 events

Coding theory [19, 61]

In coding theory, it is a common problem to identify the largest possible binary code for a given binary word (binary vectors of size n) that can correct a particular number of errors, in which the error correction is reliable and simple.

The Hamming distance between two binary vectors, $a = (a_1, a_2, ..., a_n)^\top$ and $b = (b_1, b_2, ..., b_n)^\top$, denoted by $d(a, b)$, is given by $d(a, b) = \big|\{i \mid a_i \neq b_i,\ 1 \leq i \leq n\}\big|$. In general, a binary code that consists of binary vectors of which any two have Hamming distances that is greater than or equal to d can correct $\frac{d-1}{2}$ errors [19]. Thus, in coding theory the goal is find the greatest number of binary vectors of size n, with Hamming distance d, denoted by $Q(n, d)$.

Given a Hamming graph $H(n, d)$ of size n, distance d, and with a set of vertices representing the binary vector of size n. A pair of vertices are adjacent if their Hamming distances is at least d. Thus, the graph, $H(n, d)$, has a MC of size $Q(n, d)$.

Bioinformatics and Longest Common Subsequence [121, 168]

The MC problem algorithms are widely used in applications related computational biology and bioinformatics, and construction of RNA and DNA sequences for biomolecular analysis, where typically the purpose is to find the similarities between molecules. Moreover, the algorithms for the MC problem can be used to compare protein structures, in order to obtain information about protein functions as well as information concerning protein interactions [107]. A crucial task in molecular biology is to compute the similarity between two protein structures, and this has been extensively investigated as an MC problem.

Longest Common Subsequence (LCS) is a classical problem concerned about finding the longest common sequence to a set of different strings. The LCS problem is one of the central bio-informatics problems, often involving stings representing DNA or RNA segments. In general, LCS problems solved by transforming it into instances of the MC problem [17]. The key idea is to construct a conflict graph for each instance. Consequently, an independent set in the conflict graph matches the LCS instance's common subsequence. Further, in conflict graphs, the maximum

independent set corresponds to the longest common LCS instance subsequence. It is known that finding the maximum independent set is equivalent to finding the MC as shown in Section 4.2.2.

Cluster analysis of gene expression data

According to cluster analysis, a given data set is partitioned into groups based on specific characteristics, so that points within a particular group are more similar to each other than those within other groups. It has been proved that clustering is an effective technique in understanding gene functions, gene regulations, cellular processes, and subtypes of cells.

Gene expression data holds vital information that can be used to understand how biological processes take place within an organism and how that relates to its environment. Gene expression data consist of millions of measurements. Due to the complexity of the biological network and the volumes of genes present, it is challenging to interpret and understand it. So one needs to exclude the ambiguity and imprecision of the data. In the light of this, the clustering techniques are important step toward addressing these issues.

Clustering techniques involve finding all sub-groups of genes that have similar expression patterns. This task is achievable through the use of the algorithms for MC problem to enumerate all the cliques (patterns) in the given gene expression data [138, 149, 199].

4.2.2 Algorithms for MC Problems and Existing Approaches

In general, given a graph, clique problems are about determining the MC in the given graph or about clustering the vertices of the given graph into mutually exclusive cliques. MC problems also include listing all maximal cliques or solving the decision problem of identifying whether a graph contains a clique greater than a certain size. Moreover, given a weighted graph, MC problem concerns about identifying the maximum weight clique. Finding or estimating $\omega(G)$, the clique number, and determining the MC in size in G are basic questions in theoretical computer science. Numerous researchers within the combinatorial optimization community have long been focusing on the MC problem.

The MC problem is identical to two well-known graph problems which are the Maximum Independent Set (MIS) problem, and the Minimum Vertex Cover (MVC) problem. Given a graph $G = (V, E)$, the MIS problem involves determining the ultimate cardinality of an independent set in a graph G, while the MVC problem is to determine the lowest cardinality of a vertex cover in a graph G. These three problems are identical because, a sub-set of nodes $C \subset V$ is a clique in G if and only if C is MIS in the complement graph $\bar{G} = (V, \bar{E})$, and $V \backslash C$ is a vertex cover of $\bar{G}$.

The MC problem is known as NP-hard, and it belongs to Karp's 21 NP-complete problems [100]. Finding $\omega(G)$ is also known as NP-hard problem [100]. NP-hard means that it usually cannot be solved exactly in polynomial time. In addition, real-world graphs are usually extremely rich (including billions of edges and vertices), which makes the combinatorial problems more challenging to solve in practice.

Before we present existing mathematical formulations for MC problem, let us introduce briefly existing algorithms for the MC problem. These are generally grouped into two categories, heuristics algorithms and exact algorithms. Exact algorithms seek to detect the optimal solution. Several exact solvers have been proposed for the MC problem in the light of well-known Branch and Bound (B&B) algorithms [42]. The B&B algorithms are based on the idea of breaking a given problem into smaller sub-problems. These sub-problems are further decomposed until no longer decomposable or there is no possibility of obtaining an optimal solution to the resulting sub-problems. The search strategy indicates the order of decomposition or testing of partial problems. Among those strategies are the depth-first search and the breath-first search. The B&B algorithms aim to search through the entire B&B tree to attain a global optimum. Here are the references where the main exact algorithms for finding a MC have been suggested [11, 42, 62, 107, 110, 113, 124, 136, 157, 160, 179, 180, 181]. B&B algorithms have been used in most of the formulations discussed in Section 4.2.4.

Alternatively, heuristic algorithms seek to find superior solutions in an adequate time frame, but without necessarily ensuring the optimality. This class of algorithms start with an empty clique, then iteratively add vertices and remove some vertices that do not form a clique using different

greedy principles. For instance, Harary and Ross [85] propose the first algorithm to list all cliques within an arbitrary graph where no more than three cliques exist. Based on what is known, the most effective heuristic method for the MC problem is the local search [172]. The local search scheme is to search for all the possible cliques, then find the ones with high cardinality, this is known as the legal strategy. There is another strategy for the local search scheme, which is called the n-fixed penalty strategy; this seeks a n-clique and then increase the value of n [172].

In [190] the authors present an effective heuristic approach to MC problem which is an adaptive multi start tabu-search algorithm (denoted AMTS). Numerical results presented indicate that the AMTS algorithm achieves the greatest known clique among 79 benchmark problems along with two generalizations (i.e., the maximum edge weight clique problem and the maximum vertex weight clique problem). Recently, the ATMS algorithm has been applied to solve other two combinatorial optimization problems namely the graph colouring and the graph sum colouring [123]. A novel heuristic algorithm based on the Reactive Local Search (RLS) algorithm [12] has also been designed to solve the MC problem.

Meta-heuristics approaches have also been proposed for the MC problem. For example, the Ant-Colony Optimization (ACO) has been suggested in [165]. The ACO algorithm starts by adding specific vertices to the partial cliques to construct maximal cliques. ACO repeatedly add vertices into partial cliques to generate maximal clique, here ACO is suggested as a heuristic to determine which vertices should be included in the clique at each step. The basic idea behind ACO is to model the MC problem as finding the lowest cost path in a graph, then utilize artificial ants to find the optimal path. More recently, another meta-heuristic approach has been suggested for the MC problem, where a Hybrid Particle Swarm Optimization Algorithm (HPSOD) is suggested. The authors also combine it with the local search heuristics to enhance its performance [174]. Here are the references of main heuristics algorithms for the MC problem [9, 12, 16, 30, 82, 83, 93, 102, 150, 192]. In meta-heuristics greedy approach [101], random vertices are added to subsets of vertices, then these vertices are reduced to a single clique and finally expanded to the largest clique. Suyudi et al. [172], have suggested a greedy procedure for the MC problem, that takes

into account the largest number of adjacent vertices, the solution is then determined by removing vertices with a low probability of belonging to the MC.

4.2.3 Some Bounds on $\omega(G)$

An interesting area of research consists of detecting the clique number, $\omega(G)$, of a graph. Several bounds for $\omega(G)$ have been found in the literature.

For a connected graph G with N nodes and M edges, $\omega(G)$ is bounded by [8]

$$\omega(G) \leq \frac{3 + \sqrt{9 - 8(N - M)}}{2}.$$

Let $\rho(A_G) = \max_{1 \leq i \leq N} |\lambda_i|$ indicate the spectral radius of the adjacency matrix A_G of graph G, where $\lambda_1, \lambda_2, ..., \lambda_N$ represents the eigenvalues of A_G. It was shown in [8] that,

$$\omega(G) \leq \rho(A_G) + 1,$$

when G is a complete graph, the inequality becomes equality.

Theorem 15 ([8]). *Assume that A_G is the adjacency matrix of G and N_{-1} is the number of eigenvalues of A_G which are no larger than -1. Then*

$$\omega(G) \leq N_{-1} + 1.$$

Let N_0 be the number of eigenvalues that have zero values, the authors in [8] demonstrated that

$$\omega(G) \leq N_{-1} + 1 < N - N_0 + 1.$$

In complex space, the MC problem can be formulated geometrically [25] with clique number bounded by

$$\omega(G) \leq \frac{\bar{N}_0 + N}{2},$$

as $\bar{N}_0$ represents the number of eigenvalues that have zero values of $\bar{A}$, $\bar{A}$ represents the adjacency matrix of $\bar{G}$.

The sandwich theorem [105] bounded $\omega(G)$ using the Lovasz number of $\bar{G}$, denoted $\vartheta(\bar{G})$, and $\chi(G)$ is the chromatic number , that is

$$\omega(G) \leq \vartheta(\bar{G}) \leq \chi(G),$$

where $vartheta(\bar{G})$ is determinable in a polynomial time and is defined by [57]:

$$\vartheta(\bar{G}) = \min\{\ \rho(ee^\top + Y) \ \big|\ Y \in \mathcal{Y}(G)\},$$

where $\rho(ee^\top + Y)$ is maximum eigenvalue of $ee^\top + Y$, $e = [1, 1, ..., 1]^\top \in \mathbb{R}^N$, and

$$\mathcal{Y}(G) = \{Y \in \mathbb{R}^{N\times N} \ \big|\ Y^\top = Y,\ Y_{ij} = 0,\ \textit{if}\ Y(i,j) \in E\}.$$

Emmanuel et al. [20] proposed the cutting-plane approach in combinatorial optimization for the MC problem. It begins with solving a Semi-Definite Programming (SDP) relaxation of the MC problem, then they introduce one or more cuts that violates this relaxation to tighten the bound. Let $\mathcal{S}^N$ indicate the set of symmetric matrices, that is, $\mathcal{S}^N = \{Z \in \mathbb{R}^{N\times N} \ \big|\ Z = Z^\top\}$. Thus, the clique number, $\omega(G)$, for $G \in \mathcal{S}^N$, is given by [20, 51]:

$$\omega(G) = \min\{\eta \in \mathbb{R} \ \big|\ \eta(I_N + \bar{A}) - ee^\top \in \mathcal{C}^N\},$$

where $\bar{A}$ represents the adjacency matrix of $\bar{G}$ and $\mathcal{C}^N$ indicates the set of $N \times N$ positive symmetric matrices, $\mathcal{C}^N = \{Z \in \mathcal{S}^N \ \big|\ x^\top Z x \geq 0, \forall x \in \mathbb{R},\ x \geq 0\}$.

The spectral hypergraph theory has also been used to establish new upper and lower bounds on $\omega(G)$ [26].

Theorem 16 (Upper Bound[26]). *Assume $\omega(G)$ denotes the clique number of an undirected graph G and Q, a hypergraph, is a n-clique $(n+1)$-graph of G with spectral radius $\rho(Q)$. Then*

$$\omega(G) \leq \frac{\rho(Q)}{n!} + n.$$

Theorem 17 (Lower Bound[26]). *Assume $\omega(G)$ denotes the clique number of an undirected graph G and Q, a hypergraph, is a n-clique $(n+1)$-graph of G with spectral radius $\rho(Q)$. Then*

$$\omega(G) \geq \psi_n^{-1}\left(\frac{\rho(Q)}{n!\|x_p\|_n^{n+1}}\right), \quad \psi_n(x) = (x-n)\binom{x}{n}^{-\frac{1}{n}},$$

where x_p is the Perron eigenvector of Q.

Based on a Lagrangian relaxation of a linear (integer) programming model of the maximum edge weight clique problem, $\omega(G)$ is bounded by [87]

$$\omega(G) \leq \left\lfloor \frac{1}{2}\left(\sqrt{4b+1}+1\right) \right\rfloor,$$

where b is defined by

$$b = \binom{N}{2} - |E| + \sum_{v \in V'}(2d_v) - N + 1,$$

where d_v denotes the vertex degree and $V' = \{v \in V \,\big|\, d_v \geq \frac{N-1}{2}\}$, where the symbol "$\lfloor\ \rfloor$" denotes the floor function.

4.2.4 Mathematical Formulations

Generally, the MC problem can be casted in the following mathematical form:

$$\textit{maximize} \sum_{i \in V} x_i$$

$$\textit{subject to } x \in \{\textit{all cliques of } G\}$$

$$x_i \in \{0, 1\},\ \forall i \in V,$$

for $x = \{x_1, x_2, ..., x_N\}$. A B&B algorithm based on an unconstrained quadratic zero-one programming (Q01) formulation has been proposed for the MC problem [143]. Given a graph $G = (V, E)$ with $u_1, u_2, ..., u_N$ vertices, then the MC problem is identical to the following integer program [143]:

$$\begin{aligned} &\textit{minimize } f(x) = -\sum_{i=1}^{N} x_i \\ &\textit{subject to } x_i + x_j \leq 1,\ \forall (u_i, u_j) \in \bar{E},\ x \in \{0, 1\}^N, \end{aligned} \tag{4.1}$$

where a solution x^* to this problem is a MC, C, that is:

$$x_i^* = \begin{cases} 1, & \textit{if } u_i \in C \\ 0, & \textit{if } u_i \notin C, \end{cases}$$

the clique C is then with cardinality $|C| = -f(x^*)$. This formulation is known as the edge formulation. For pair $(i, j) \in \bar{E}$, only one of the vertices i or j may belong the clique, i.e., $x_i + x_j \leq 1, \forall (i, j) \in \bar{E}$. The $Q01$ program in (4.1) can be expressed as

$$\begin{aligned} &\textit{minimize } f(x) = -\sum_{i=1}^{N} x_i + 2 \sum_{(u_i, u_j) \notin E,\ i > j} x_i x_j \\ &\textit{s.t. } x \in \{0, 1\}^N, \end{aligned}$$

which is identical to the quadratic form:

$$\max f(x) = x^\top Ax$$
$$\textit{s.t. } x \in \{0,1\}^N,$$

where $A = \bar{A} - I_N$ and I_N is the $N \times N$ identity matrix.

Another formulation for the MC problem is the independent set formulation [19]:

$$\max \sum_{i=1}^{N} x_i$$
$$\textit{s.t.} \sum_{i \in s} x_i \leq 1, \ \forall s \in S$$
$$x_i \in \{0,1\}, \ \ i = 1, 2, ..., N,$$

where the set S contains all maximal independent sets. However, it is not easy to list all possible independent sets of the graph because their number raises exponentially with the graph size.

Motzkin and Straus [128] devised a continuous quadratic formulation for the MC problem, that is:

$$\max \ f(x) = x^\top Ax$$
$$\textit{s.t. } x \in \Delta,$$

where A denotes the adjacency matrix for a given graph $G = (V, E)$ and Δ is the N-dimensional simplex defined by:

$$\Delta := \{x \in \mathbb{R}^N \big| x \geq 0 \textit{ and } e^\top x = 1\},$$

where e is the vector of one's. Assume that x^* is a global maximizer of f on Δ. Motzkin and Straus [128] derived the following relation between $\omega(G)$ and $f(x^*)$:

$$\omega(G) = \frac{1}{1 - f(x^*)} \geq \frac{1}{1 - f(x)}, \forall x \in \Delta.$$

Recently, a general regularized continuous formulation for the MC problem has been proposed [91]:

$$\max f(x) = x^\top A x + \Phi(x)$$
$$\textit{s.t. } x \in \Delta,$$

where $\Phi : X \longrightarrow \mathbb{R}$ is a twice continuously differentiable function defined on some open set $X \subset \Delta$. They consider three different regularization functions with specified choices of some parameters, which satisfy specific conditions.

Beside these formulations, there are many other formulations of the MC problem found in the literature, such as the indefinite quadratic problem suggested in [141, 144]. Unfortunately, no efficient algorithm was suggested as the model is indefinite.

In [142] the authors proposed a new formulation of the MC problem as an indefinite quadratic global optimization problem with linear constraints. In addition, they developed an algorithm to find the MC. The authors tested the proposed model with graph dimensions varying from $n = 10$ to $n = 74$.

A polynomial-time spectral algorithm can, with a high probability, identify a unique Planted Clique (PC) of size $n \geq a\sqrt{N}$, assuming a high enough value of a, in the random graph $G = (N, \frac{1}{2})$ has been presented in [3]. Here is a description of their algorithm. Let $\tilde{A}$ represent the adjacency matrix of $\tilde{G} = (V, \tilde{E})$, where $\tilde{G}$ is chosen from the distribution $G = (N, \frac{1}{2})$, and a PC, C, that has size n added to the G, with $\tilde{A}_{ii} = 0$, and for $i \neq j$, $\tilde{A}_{ij} = 1$ if $(i, j) \in \tilde{E}$, $\tilde{A}_{ij} = -1$ if $(i, j) \notin \tilde{E}$. For $\tilde{G}$ selected from G, it is highly unlikely to have an eigenvalue exceed $2\sqrt{N}$. Moreover, the presence of the clique C in the corresponding ± 1 adjacency matrix A of G results in at least a value $n - 1$ eigenvalue. Thus, if $n \gg 2\sqrt{N}$, the largest eigenvalue of A distinguishes graphs with

PCs from graphs $G = (N, \frac{1}{2})$. Finding the actual PC, C, can then be done using the corresponding eigenvector. They have proposed the following algorithm.

Algorithm 2 [[3] Algorithm A]

Input: A_G, the adjacency matrix of the input graph $G = (V, E)$ from $G = (N, 1/2, n)$ that has a clique C with $n \geq 10\sqrt{N}$

- From A_G, compute the eigenvector v_2 that corresponds to λ_2 (the second maximum eigenvalue,) of A_G,
- Assume that $W \subseteq V$ corresponds to the N highest entries (in absolute value) in v_2. The subset $C \subset V$ consists of all nodes in G with a minimum of $3N/4$ neighbours in W.

Output: C.

Another polynomial-time algorithm (Low Degree Removal algorithm) has also been suggested for determining hidden cliques of size $n > a\sqrt{N}$, given that the constant a is sufficiently large, in the spectral approach [69], a is smaller than that. They have the following theorem.

Theorem 18 ([69] Theorem 1)**.** *Under the assumption that $a > 0$ is sufficiently large, the Low Degree Removal solves the MCP with probability p not less than $2/3$ when $n \geq a\sqrt{N}$, where p is weighted based on the selection of the random input graph G.*

Feige and Krauthgamer [68] have proposed an algorithm based on $\vartheta(\bar{G})$ [117] for finding the hidden clique of size $n \geq \Omega(\sqrt{N})$ in $G = (N, \frac{1}{2})$. Their results are as follows.

Lemma 4.2.5 ([68] Lemma 2)**.** *Assume that $G = (N, 1/2, n) = G_{max}$, in which $n > a\sqrt{(N)}$ when a is large enough. Then with high probability, $\vartheta(\bar{G}) = n$.*

They suggest the following algorithm.

Algorithm 3 [[68] The BasicFind algorithm]

Input: The input graph $G = G_{max}$ that includes a PC, C, of size $n > a\sqrt{N}$ when a is large enough

- $\forall\, v \in V$ find $\vartheta(\overline{G \backslash v})$ and $\vartheta(\bar{G})$.

Output: $P \leftarrow \{v : \vartheta(\bar{G}) - 1/2 > \vartheta(\overline{G \backslash v})\}$ and $\vartheta(\bar{G})$.

They also proposed an improved algorithm.

Algorithm 4 [[68] The ImprovedBasicFind algorithm]

Input: The input graph $G = G_{max}$ that includes a PC, C, of size $n > a\sqrt{N}$ provided a large enough of a

- Compute $\vartheta \leftarrow \vartheta_4(\bar{G})$ where $\vartheta_4(G) = \max_{o,\{u_i\}} \Sigma_{i \in V}(o.u_i)^2$, $o \in \mathbb{R}^N$ is known as a handle, which contains all vectors of unit length, with orthonormal representation of $\bar{G}$ being $u_i \in \mathbb{R}^N :\ i \in V$ with small a cumulative error $\epsilon = 1/3$.
- $P \leftarrow \{i :\ (o.u_i)^2 > 1/2\}$.

Output: ϑ and $P = C$.

The above algorithm returns a hidden clique C of size n identical to an output P with high probability, and $\vartheta(\bar{G}) = |P|$. In addition, their algorithm can be applied to a semi-random hidden clique model, where some edges can be removed by an adversary from the random part of the graph. This is the result of the following theorem.

Theorem 19 ([68] Theorem 1)**.** *Given a semi-random graph G^*, a polynomial-time algorithm exists, for any $n = \Omega(\sqrt{N})$, that determines a clique of size n with high probability along with a tight upper bound of n on the MC size in G^*.*

An indefinite quadratic global optimization model is formulated for the MC problem [142]. That is

$$\max_{x \in S} f_G(x) = \frac{1}{2} x^\top A_G x,$$

where A_G is an $N \times N$ denotes the adjacency matrix of G, $S = \{x \in \mathbb{R}^N \big| e^\top x = 1, x \geq 0\}$, $e = (1, 1, ..., 1) \in \mathbb{R}^N$. The following theorem shows the connection between the MC problem and the quadratic global optimization model.

Theorem 20 ([142]). *The graph $G = (V, E)$ has a MC, C, of size $n = \frac{1}{1-2\alpha}$, provided that If $\alpha = \max f_G(x)$ over S. One can reach this by setting $x_i = \frac{1}{n}$ if $u_i \in C$ and $x_i = 0$ if $u_i \notin C$.*

They have also bounded the MC size by $n \geq \frac{N}{N-(N-1)d}$, where d indicates the density of G defined by $d = \frac{|E|}{\binom{N}{2}}$.

More recently, a new continuous formulation that is based on rank-one non-negative matrix approximation for the MC problem has been proposed [15]. Given $H = A + I_N$, where A denotes the adjacency matrix of G and I_N denotes the identity matrix with dimension $N = |V|$. For $s \geq 0$, define a symmetric matrix $D_s = (d_{ij})_{i,j=1}^N$ by

$$d_{ij} = \begin{cases} 1, & \text{if } h_{ij} = 1, \\ -s, & \text{if } h_{ij} = 0. \end{cases}$$

Then, the MCP is equivalent to solve the following symmetric rank-one matrix approximation:

$$\min_{u \in \mathbb{R}^{+N}} \|D_s - uu^\top\|_F^2,$$

where u^* is the optimal vector that indicates the MC.

4.2.6 The MC Problem as Rank Minimization Problem

More recently, Ames and Vavasis [5] have taken a different approach in modelling the MC problem as a rank minimization where a matrix Z is used as the optimization variable. Rank of Z is then minimized subject to some constraints. The convex equivalent of the model is then solved with known convex solver. This approach is also suitable for determining the PC in a given graph.

In the given graph $G = (V, E)$, the adjacency matrix of a clique C forms a rank-one matrix. That is, for a characteristic vector u of C and $\forall\, i \in V$:

$$u_i = \begin{cases} 1, & \textit{if } i \in V(C), \\ 0, & \textit{otherwise}\ . \end{cases}$$

Accordingly, the matrix $uu^\top$ clearly has rank of 1. Further, if G includes a clique of size n, then $uu^\top$ is optimal for the following Rank Minimization Problem (RMP):

$$\textit{minimize}\ \operatorname{rank}(Z) \tag{4.2}$$

$$\textit{s.t} \quad \sum_{i \in V} \sum_{j \in V} Z_{ij} \geq n^2, \tag{4.3}$$

$$Z_{ij} = 0,\ \textit{if}\ (i,j) \notin E\ \textit{and}\ i \neq j, \tag{4.4}$$

$$Z = Z^\top, \tag{4.5}$$

$$Z \in [0,1]^{V \times V}. \tag{4.6}$$

Since the rank function is discontinuous and non-convex, most recent studies use the convex relaxation technique and solve the nuclear norm minimization instead of solving the RMP. Ames and Vavasis [5] used the following reformulated model for the MC problem:

$$\textit{minimize}\ \|Z\|_* \tag{4.7}$$

$$\textit{s.t} \quad \sum_{i \in V} \sum_{j \in V} Z_{ij} \geq n^2, \tag{4.8}$$

$$Z_{ij} = 0,\ \textit{if}\ (i,j) \notin E\ \textit{and}\ i \neq j. \tag{4.9}$$

Bounds are provided for the nodes in input graph G, the PC size, the number of diversionary edges [5]; these guarantees that NN relaxation (4.7) is exact for formulation (4.2). For the input graph $G = (V, E)$, first a clique C of size n is planted randomly, then the remaining edges are added to the set of edges E either independently at random based on with a predetermined prob-

ability $p \in [0,1)$ or deterministically by an adversary. As a result they have the following two theorems. The first theorem applies when the remaining edges are deterministically added.

Theorem 21 ([5] Theorem 4.1.2)**.** *Let $G = (V, E)$ contain a clique V^* of size n. Assume that no more than g edges of G are not in $G(V^*)$ and for some $\delta \in (0,1)$ each $v \in V \backslash V^*$ is adjacent to not more than δn vertices in V^*. Thus, there exists a scalar $a < 1/2$ that depends on δ only so that if*

$$g \leq an^2$$

then the MC V^* of G is unique and can be found by solving (4.7)-(4.9)*.*

The following theorem relates to the case in which the remaining edges are added with a fixed probability.

Theorem 22 ([5] Theorem 4.1.3)**.** *Assume that a clique V^* of size n is contained in the graph $G = (V, E)$. In addition, assume that each possible edge of G in $(V \times V) \backslash (V^* \times V^*)$ is allocated to E independently at random with a fixed probability $p \in [0,1)$. Therefore there exists a scalar $b > 0$ that depends on the determined probability for which if*

$$n \geq b\sqrt{N},$$

where $N = |V|$, hence, with probability tends toward 1 as $N \to \infty$, V^* is the MC of G and corresponds to the unique optimal solution of (4.7)*.*

In order to recover $V^* = \{1, 2, ..., n\}$, Ames [5] solves the following NN minimization problem using PPAPack, NN minimization software package [115] in Matlab

$$\min\{\|Z\|_* = \langle Z, ee^\top \rangle \geq n, Z_{ij} = 0 \text{ if } (A_G)_{ij} = 0, i \neq j\},$$

where $(A_G)_{ij} = 1, \ \forall (i,j) \in V^* \times V^*$, and for $i < j$ and $(i,j) \in (V \times V) \backslash (V^* \times V^*)$, set $A_{ij} = 0$ with probability p and 1 with probability $1 - p$. Then let $(A_G)_{ji} = (A_G)_{ij}$. Ames [5] also used the

following semi-definite program to recover V^*

$$\min\{\langle Z, ee^\top \rangle \ : \ Z \succeq 0, \ Ze \leq e, \ \mathrm{Tr}(Z) = 1, \ Z_{ij} = 0 \ \textit{if} \ (A_G)_{ij} = 0, i \neq j\}.$$

The above semi-definite program is then solved using the semi-definite programming software package SDPNAL [198] in Matlab.

4.3 The Maximum Edge Bi-clique Problem

Bi-clique is a fundamental structure of bipartite graphs. This structure is widely used in a variety of bipartite applications to capture cohesive sub-graphs. Application areas for the MEB problem include community detection in social network, e-commerce network and identification of anomalies.

A bipartite graph (or bi-graph) consists of vertices that can be separated into two independent and disjoint sets. A bi-clique in such a graph is a complete bipartite sub-graph.
Let $G_b = (V, E) = (V_1 \cup V_2, E \subseteq (V_1 \times V_2))$ be a bipartite graph with two independent and disjoint sets, V_1 and V_2, such that every edge links a vertex in V_1 to one in V_2. It is obvious that a bi-clique (V_1^*, V_2^*) has $|V_1^*| + |V_2^*|$ vertices and $|V_1^*| \times |V_2^*|$ edges. When $|V_1^*| = |V_2^*|$, then the problem is also called balanced bi-clique problem.

A bi-clique is defined as maximum if it is of the largest size, measured either by the number of vertices (vertex-maximum), known as the maximum vertex bi-clique problem, or by the number of edges (edge-maximum), known as the MEB problem. In the maximum balanced bi-clique problem the goal is to find a complete bipartite graph with a maximum number of vertices and it has been proved to be NP-complete [95]. Peeters [148] has proved that MEB problem is NP-complete. On the other hand, Johnson [95] has proved that the maximum vertex bi-clique problem is solvable in a polynomial time when the input graph is bipartite. However, the problem is NP-complete when the input graph is not bipartite [194].

Lyu et al. [119] have proposed a progressive bounding framework to find the bi-clique with the maximum number of edges in large-scale real-world datasets. They have also extended the branch and bound algorithm for the MEB problem.

A rank-one non-negative factorization model has been proposed for MEB [75]. In this formulation a bi-clique is constructed by reducing the number of edges outside the bi-clique instead of increasing the number of edges inside.

The MEB problem is known to have no good approximation algorithm. However, based on some assumptions, Ambühl et al. [4] have obtained hardness results for the MEB problem. [122] presents conditional inapproximability results for the MEB problem based on the small set expansion hypothesis.

Currently, the majority of algorithms that solve the MEB problem enumerate all maximal bi-cliques in the graph [2, 54, 104]. Nussbaum et al. [131] have developed a pruning strategy that avoids enumerating all maximal bi-cliques.

Recently, Sözdinler and Özturan [166] have introduced an algorithm based on integer programming for determining the MEB in bipartite graphs. However, the maximum bi-clique size must be known a priori.

A nuclear norm heuristic has also been proposed for the planted bi-clique problem [5, 7]. It has been shown that when the given bipartite graph contains a planted bi-clique of sufficiently large size, the proposed heuristic is exact.

In a recent paper, Bombina and Ames [18] have focused on identifying the densest subgraph and densest submatrix problems, which can be seen as a generalization of the MCP and the MEB problems.

4.3.1 Mathematical Models for the MEB Problem

Let $M_b \in \{0,1\}^{N \times M}$ be the adjacency matrix of $G_b = (V, E) = (V_1 \cup V_2, E \subseteq (V_1 \times V_2))$ with $V_1 = \{k_1, k_2, ..., k_N\}$ and $V_2 = \{l_1, l_2, ..., l_M\}$, provided that $M_b(i,j) = 1$ if and only if $(k_i, l_j) \in E$. Then, the MEB problem can be expressed as [76]

$$\max_{u,v} \sum_{i,j} u_i v_j \tag{4.10}$$

$$\textit{s.t. } u_i + v_j \leq 1 + (M_b)_{ij},\ \forall i,\ j, \tag{4.11}$$

$$u \in \{0,1\}^N,\ v \in \{0,1\}^M, \tag{4.12}$$

such that if $u_i = 1$ (respectively $v_j = 1$), then node k_i (respectively l_j) belongs to the solution, otherwise $u_i = 0$ (respectively $v_j = 0$). The constraint in (4.11) ensures that if $(M_b)_{ij} = 0$, then either u_i or v_j is equal to zero, i.e., they both cannot be part of a feasible solution if there is no edge between k_i and l_j. The above model can be formulated as a minimization problem where the goal is to minimize the number of edges outside the bi-clique, that is

$$\min_{u,v} \|M_b - uv^\top\|_F^2 \tag{4.13}$$

$$\textit{s.t. } u_i + v_j \leq 1 + (M_b)_{ij},\ \forall i,\ j, \tag{4.14}$$

$$u \in \{0,1\}^N,\ v \in \{0,1\}^M, \tag{4.15}$$

Authors in [76] have also proposed an approximate rank-one matrix factorization problem as a continuous characterization of the MEB problem. For a parameter $d > 0$ the MEB problem can be reformulated as

$$\min_{u,v} \|M_d - uv^\top\|_F^2 \tag{4.16}$$

$$\textit{s.t. } u \geq 0, v \geq 0, \tag{4.17}$$

where $M_d = (1+d)M_b - d * O_{N\times M}$, $O_{N\times M}$ is a matrix of ones.

Assume that the bipartite graph G_b includes a bi-clique (V_1', V_2') of size nm, that is $|V_1'| = n$, $|V_2'| = m$, then $v_1 v_2^\top$ is optimal for the following RMP, where v_1 and v_2 are the characteristic vectors of V_1' and V_2', respectively. The associated mathematical model is given by:

$$\textit{minimize } \operatorname{rank}(Z) \tag{4.18}$$

$$\textit{s.t.} \quad \sum_{i \in V_1} \sum_{j \in V_2} Z_{ij} \geq nm, \tag{4.19}$$

$$Z_{ij} = 0, \textit{ if } (i,j) \in (V_1 \times V_2) \backslash E, \tag{4.20}$$

$$Z \in [0,1]^{V_1 \times V_2}. \tag{4.21}$$

Ames and Vavasis [5] used the following reformulated model for the planted edge bi-clique problem:

$$\textit{minimize } \|Z\|_* \tag{4.22}$$

$$\textit{s.t.} \quad \sum_{i \in V_1} \sum_{j \in V_2} Z_{ij} \geq nm, \tag{4.23}$$

$$Z_{ij} = 0, \textit{ if } (i,j) \in (V_1 \times V_2) \backslash E \textit{ and } i \neq j. \tag{4.24}$$

More recently, Shahinpour et al. [162, 163] have also proposed an integer programming formulation for the MEB problem.

Chapter 5

Model Formulations and ADMM Algorithms

5.1 Introduction

The objective of this Chapter is to introduce our proposed models for the Planted Clique Problem (PCP), the Planted Edge Bi-clique Problem (PEBP), and the Maximum Clique Problem (MCP) and the Maximum Edge Bi-clique Problem (MEBP) in random graphs. These models are the NN-based (Nuclear Norm) matrix decomposition model and the TNN-based (Truncated Nuclear Norm) matrix decomposition model. We begin by introducing the Cardinality Minimization Problem (CMP). Then we suggest a new (non-convex) surrogate function for the ℓ_0-norm. We present our NN-based matrix decomposition model for the PCP, the PEBP, the MCP, and the MEBP, and followed by the TNN-based matrix decomposition model for the four problems. Finally, we present the ADMM based algorithms used to solve the above optimization models. We conclude this Chapter with proof of the convergence of the algorithms for both models.

5.2 Cardinality Minimization Problem

Given a set of linear (and possibly non-linear) constraints, the problem of minimization of cardinality is to obtain the vector with the least cardinality. The general CMP is formulated as follows:

$$\begin{aligned} &\textit{minimize} \;\; Card(z) \\ &\textit{subject to } z \in \mathcal{C}, \end{aligned} \tag{5.1}$$

where the cardinality of $z \in \mathbb{R}^N$, known as $Card(z)$. It represents how many non-zero components of z, $\mathcal{C}$ denotes a convex set of constraints. In other words, given a vector z, CMP is concerned

about maximizing the number of the zero components in z or, alternatively, minimizing the number of the non-zero components in z, with respect to certain constraints.

In fact, this problem is closely related to what is known as "compressive sensing problem" [36, 56]. The goal of CMP is to find the sparsest vector within a feasible set, while in compress sensing problem, one seeks to obtain the signal $z \in \mathbb{R}^{N_2}$ given $b \in \mathbb{R}^{N_1}$, an observed signal, by solving $Az = b$; $A \in \mathbb{R}^{N_1 \times N_2}$, $N_1 < N_2$.

According to the literature, the cardinality function, $Card(z)$, in (5.1) can be approximated by the ℓ_0-norm, as

$$\|z\|_0 = \lim_{p \longrightarrow 0} \|z\|_p = \lim_{p \longrightarrow 0} \left(\sum_{i=1}^{N_2} |z|^p \right)^{\frac{1}{p}} = Card(z),$$

where $\|z\|_0$ is the so-called ℓ_0-norm, but it is not a norm due to the non-satisfaction of the homogeneity condition of a norm, see Section 2.3. Therefore, when the set $\mathcal{C}$ is linear, the sparsest solution of $Az = b$, $A \in \mathbb{R}^{N_1 \times N_2}$, $b \in \mathbb{R}^{N_1}$, can be determined by solving the following problem:

$$\begin{aligned} &\textit{minimize} \ \ \|z\|_0 \\ &\textit{subject to} \ Az = b. \end{aligned} \tag{5.2}$$

One of the most common heuristics to solve problem (5.2) involves replacing $\|z\|_0$ by $\|z\|_1$ which results in the following problem of minimization:

$$\begin{aligned} &\textit{minimize} \ \ \|z\|_1 \\ &\textit{subject to} \ Az = b. \end{aligned} \tag{5.3}$$

The problem (5.3) is called the ℓ_1-norm minimization problem which has several applications, particularly, in the field of signal processing [24, 32, 37, 167].

In many situations, re-weighted ℓ_1-norm performs better than the un-weighted ℓ_1-norm according to several numerical experiments [40, 47, 50, 70, 111].

Candes et al. [40] have proposed a weighted ℓ_1-norm minimization problem as follows:

$$\begin{aligned} &\textit{minimize}_{\ z\in\mathbb{R}^N} \ \|Wz\|_1 \\ &\textit{subject to } Az = b, \end{aligned} \tag{5.4}$$

where W is a diagonal matrix with diagonal entries $\{w_1, w_2, ..., w_N\}$. The authors also suggested an algorithm for (5.4). The iterative re-weighted ℓ_1 algorithm is described as follows: in the J-th step, if the solution z^J is computed, then the weight at the $J+1$-th step is given by

$$w^{J+1} = \frac{1}{|z^J| + \epsilon}, \ \ \epsilon > 0.$$

Zhao and Li [111] have also proposed

$$\Phi_\epsilon(z) = \sum_{i=1}^{N} \log(|z_i| + \epsilon) + \sum_{i=1}^{N} (|z_i| + \epsilon)^p \,, \ p \in (0, 1),$$

as an approximation for the ℓ_0-norm in (5.2).

5.3 Model Formulations

Here, we will introduce the mathematical models for the PCP, the PEBP, the MCP and the MEBP together with the ADMM algorithms.

5.3.1 Mathematical Model for PCP and MCP

We have taken the matrix decomposition approach for the planted clique problem. The matrix decomposition problem separates a given matrix M into its low-rank and sparse component by solving the problem

$$\begin{aligned} &\min_{L,S\in\mathbb{R}^{N\times N}} \ \text{rank}(L) + \lambda\|S\|_0 \\ &\textit{s.t.} \ \ L + S = M, \end{aligned}$$

where $\|S\|_0 = \text{card}(S)$ is the number of non-zero entries in S. Both the rank function and ℓ_0-norm minimization are non-convex. The NN, $\|L\|_*$, is the sum of singular values $\sigma_i(L)$; it is used as the convex relaxation of rank function, and the ℓ_1-norm, $\|S\|_1 = \sum_{i=1}^{N}\sum_{i=1}^{N}|S_{ij}|$, is used as the convex relaxation of $\|S\|_0$.

In the context of the planted clique problem of size n, if we include self-loops and assign 1 to the diagonal elements of the adjacency matrix M, then M can be split into a rank-one matrix L (corresponding to the maximum clique) and a sparse matrix S. Thus, the formulation for MCP is

$$\min_{L,S\in\mathbb{R}^{N\times N}} \|L\|_* + \lambda\|S\|_1 \tag{5.5}$$

$$\textit{s.t.}\ \ M - L - S = 0, \tag{5.6}$$

$$S_{ij} \in [0,1], \tag{5.7}$$

where $M \in \mathbb{R}^{N\times N}$ is the adjacency matrix of graph G; L and S are the optimization variables. $\lambda > 0$ is the regularization parameter. We refer to problem (5.5)-(5.7) as the 'regular' matrix decomposition formulation for PCP and MCP.

The main difficulty of the above model is that the entries of the optimal L and S may not be integers. Indeed, this was observed for the NN minimization model by Ames [5] who rounded each entry of the optimal L to the nearest integers. The rounding of entries causes noisy recovery as indicated by a number of figures presented in [5]. Below, we propose our mathematical model which can overcome the above difficulty. The main concept of our approach is to use the weighted $\|S\|_1$ norm in the objective function (5.5). We have demonstrated a posteriori that the weighted $\|S\|_1$ norm is central to achieving integer value of the entries of L and S. We have taken a systematic approach to generate the weights. We approximate each term of $\|S\|_0$,

$$\|S\|_0 = \#\begin{cases} 1 & \textit{if } S_{ij} \neq 0, \\ 0 & \textit{otherwise}. \end{cases}$$

with the function

$$\psi(S_{ij}) = \frac{S_{ij}}{S_{ij} + \epsilon}, S_{ij} \neq -\epsilon,\ \epsilon > 0. \tag{5.8}$$

The function ψ is concave in $[0, \infty)$ with $\psi'(S_{ij}) > 0$ for all $S_{ij} \geq 0$, $\psi'(0) < \infty$.

Before presenting the mathematical model we make a graphical comparison of $\psi(x)$ in (5.8) with other, approximations for $\|x\|_0$ in the case of single variable. The comparison in Figure 5.1 shows that $\psi(x)$, $\epsilon \to 0$, gives a better approximation of $|x|_0$ than $|x|$ and $\log(|x| + \epsilon)$. This motivates our choice of small values of ϵ.

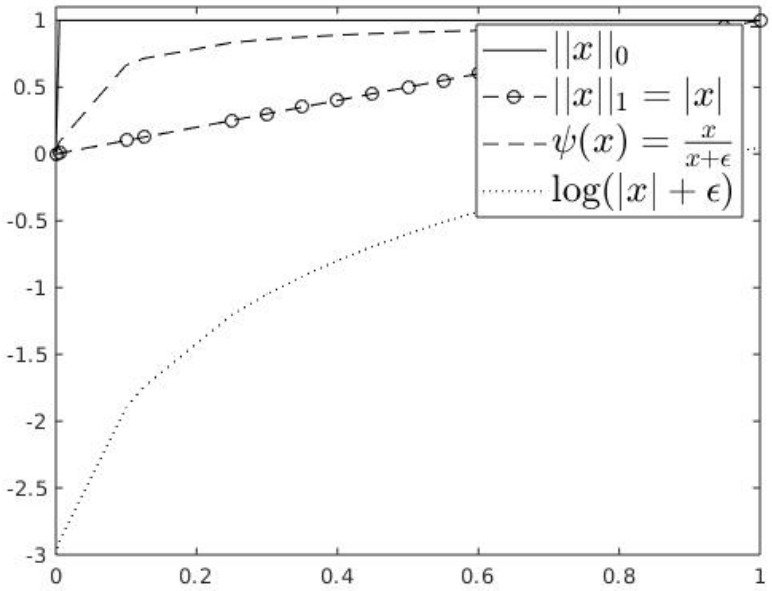


Figure 5.1: Comparison between $\psi(x)$, the penalty log function $\log(|x| + \epsilon)$, ℓ_1-norm and ℓ_0-norm in the case of single variable, $\epsilon = 0.05$.

Hence, our choice of function $\psi(S_{ij})$ further validates the claim made in [40] that the ℓ_1-norm is not a good approximation for ℓ_0-norm. Furthermore, in our mathematical model $S_{ij} = -\epsilon$ can be readily avoided, since $S_{ij} \in [0, 1]$. We write our relaxed objective function as $\|L\|_* + \lambda\Phi(S)$,

$$\Phi(S) = \sum_{i=1}^{N}\sum_{j=1}^{N}\psi(S_{ij}). \tag{5.9}$$

We now construct a convex surrogate of $\Phi(S)$ at a known feasible S_{J-1}, say at $(J-1)$-th iteration of an algorithm. It follows from the concavity of ψ that

$$\psi(S_{ij}) \leq \psi((S_{J-1})_{ij}) + \psi'((S_{J-1})_{ij})(S_{ij} - (S_{J-1})_{ij}).$$

Hence, we have

$$\begin{aligned}\Phi(S) = \sum_{i=1}^{N}\sum_{j=1}^{N}\psi(S_{ij}) &\leq \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\psi((S_{J-1})_{ij}) + \psi'((S_{J-1})_{ij})(S_{ij} - (S_{J-1})_{ij})\right)\\ &= \widetilde{\Phi}(S),\end{aligned}$$

where $\psi'((S_{J-1})_{ij}) = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$. $\Phi(S)$ satisfies

$$\Phi(S) \leq \widetilde{\Phi}(S) \;\; \textit{and} \;\; \Phi(S_{J-1}) = \widetilde{\Phi}(S_{J-1}). \tag{5.10}$$

We now ignore the constant terms in $\widetilde{\Phi}(S)$ and treat the remaining expression as the surrogate for $\Phi(S)$. The concept of the surrogate function has been reported in [84]. Hence, the surrogate becomes

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\psi'((S_{J-1})_{ij})S_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{N}|C_{ij}S_{ij}| = \|C \circ S\|_1,$$

where C is a constant matrix with entries $\frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$. It is clear that the entries of C are strictly positive. The symbol "$\circ$" is known as Hadamard product. Function $\Phi(S)$ defined in (5.9) is a concave function. However, the surrogate function $\|C \circ S\|_1$ at S_{J-1} is convex which we refer to as the weighted ℓ_1-norm, where the weights are computed dynamically.

We now compare compare our surrogate function, the 'regular' convex relaxation, and the relaxation suggested in [40] for the case of single variable in Fig. 5.2 in the interval $[\epsilon, 1]$ for a

sufficiently small ϵ. Our surrogate function for single variable case is given by $\|cx\|_1$, $c = \frac{\epsilon}{(x+\epsilon)^2}$; $\epsilon = 0.005$ is constant.

Figure 5.2 shows the surrogate is a low lying flat like convex in $[\epsilon, 1]$ which allows iterate of an algorithm to land over a range and thereby producing sparse solution via proximal operator.

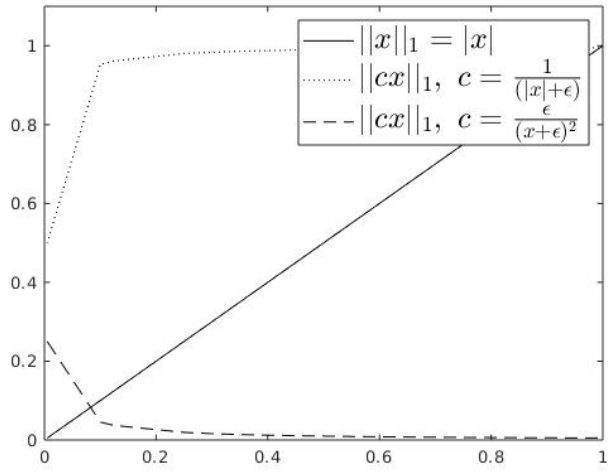


Figure 5.2: Comparison between the convex surrogate function $\|cx\|_1$, the function $\|cx\|_1$, $c = \frac{1}{|x|+\epsilon}$, $\epsilon = 0.005$, suggested in in [40], and ℓ_1-norm in the case of single variable.

We initialize the optimization algorithm with (L_0, S_0), $C_k = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$, $k = 0$, $J = 1$, then update the matrix C after every q number of iterations of the algorithm, i.e., C is constant for each epoch k, J is the iteration index. We then take the corresponding iterates

$$\{(L_0, S_0), (L_q, S_q), (L_{2q}, S_{2q}), (L_{3q}, S_{3q}), ...\} \tag{5.11}$$

and denote it as the subsequence $\{(L_k, S_k)\}$, where (L_k, S_k) is the last solution at epoch k. The relation between k and J is as follows. If $\mod(J, q) = 0$, then we increase k by 1, and update the value of C_k via $C_k = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$, where $\mod(a, b)$ is the modulo operation that finds the reminder after dividing a by b. Thus our proposed mathematical model for the PCP and the MCP is given by:

$$\min_{L,S\in\mathbb{R}^{N\times N}} \|L\|_* + \lambda\|C\circ S\|_1 \tag{5.12}$$

$$\textit{s.t.}\ \ (5.6)-(5.7), \tag{5.13}$$

where the matrix C is updated at each epoch k of the algorithm used to solve it.

5.3.2 The Truncated Nuclear Norm Model

Many real-world applications require recovering the low-rank and sparse components of a matrix. Approaches used to address all problems considered here are usually reformulated as low-rank approximation problems involving the nuclear norm of the matrix. These approaches may not provide an accurate estimation of the rank [88, 196]. Various studies have demonstrated that the TNN in sparse and low-rank matrix decomposition leads to a better approximation of the matrix rank function and provides significant recovery effects [41, 55, 151, 193, 197]. Motivated by these, some researchers have studied the following TNN-based matrix decomposition model:

$$\min_{L,S\in\mathbb{R}^{N\times N}} \|L\|_t + \lambda\|S\|_1 \tag{5.14}$$

$$\textit{s.t.}\ \ M-L-S=0, \tag{5.15}$$

where t is the number of the truncated singular values; $\|L\|_t$ is the TNN of L, which is defined as the sum of the smallest $r-t$ singular values, i.e., $\|L\|_t=\sum_{i=t+1}^{r}\sigma_i(L)$, r is the rank of L, i.e., the number of the nonzero singular values.

Assume that $L\in\mathbb{R}^{N\times N}$ is of rank r and has a full SVD of the form $L=U\Sigma U^\top$, where $U=(u_1,u_2,...,u_N)\in\mathbb{R}^{N\times N}$ and $\Sigma\in\mathbb{R}^{N\times N}$. Let $A=(u_1,u_2,...,u_t)^\top\in\mathbb{R}^{t\times N}$, $AA^\top=I_{t\times t}$. Zhang et al. [88, 196] have shown that the TNN can be cast as:

$$\|L\|_t=\|L\|_*-\max_{AA^\top=I_{t\times t}}\mathrm{Tr}(ALA^\top).$$

We consider the following TNN-based 'regular' matrix decomposition model for PCP and MCP:

$$\min_{L,S\in\mathbb{R}^{N\times N}} \ \|L\|_t + \lambda\|S\|_1 \tag{5.16}$$

$$\textit{s.t.} \ \ M - L - S = 0, \tag{5.17}$$

$$S_{ij} \in [0,1]. \tag{5.18}$$

We refer to this model as the TNN-based regular model.

We have proposed the following TNN-based matrix decomposition model for PCP and MCP:

$$\min_{L,S\in\mathbb{R}^{N\times N}} \ \|L\|_* - \max_{AA^\top = I_{t\times t}} \mathrm{Tr}(ALA^\top) + \lambda\|C\circ S\|_1 \tag{5.19}$$

$$(5.17) - (5.18), \tag{5.20}$$

where C is defined as in the previous section, Section 5.3.1. Here, when optimum of L is obtained, $\mathrm{Tr}(ALA^\top)$ represents the spectral norm $\|L\|$, σ_1, the maximum singular value.

5.3.3 Mathematical Models for the MEB Problem

Given a bipartite graph $G_b = (V_1 \cup V_2, E)$, $|V_1| = N$ and $|V_2| = M$, the adjacency matrix $M_b \in \mathbb{R}^{N\times M}$ of G_b can be divided into a rank-one matrix L (corresponding to the maximum edge bi-clique) and a sparse matrix S. Here, we assume that the bi-clique is of size nm. Hence matrix L of rank one corresponds to the bi-clique with nm entry values one and remaining entries zeros. The entries of M_b corresponding to the edges outside the bi-clique go to S. Thus, we consider, the NN-based matrix decomposition model, the convex formulation in (5.12)-(5.13) for the PEBP, and the MEBP in random graphs, where the matrices L and S are of sizes $N \times M$; and M_b here corresponds to M in (5.12)-(5.13). Moreover, we have considered the same ADMM algorithm, Algorithm 5, for the PEBP and the MEBP since both problems, MCP and MEBP, have the same low-rank structure. Furthermore, we consider, the TNN-based matrix decomposition model along

with its corresponding algorithm, the convex formulation in (5.19)-(5.20) for the PEBP, and the MEBP in random graphs.

5.3.4 The ADMM Algorithm for the NN Model

Here, we describe the ADMM algorithm that we have used to solve (5.12)-(5.13).

The augmented Lagrangian of problem (5.12)-(5.13) is given by

$$\mathcal{L}_\rho(L,S,\mu) = \|L\|_* + \lambda\|C\circ S\|_1 + \langle \mu, M-L-S\rangle + \frac{\rho}{2}\|M-L-S\|_F^2. \tag{5.21}$$

The scaled form of the augmented Lagrangian is as follows:

$$\mathcal{L}_\rho(L,S,\mu) = \|L\|_* + \lambda\|C\circ S\|_1 + \frac{\rho}{2}\|M-L-S+\frac{1}{\rho}\mu\|_F^2 - \frac{1}{2\rho}\|\mu\|_F^2, \tag{5.22}$$

where C is a constant matrix; μ is the Lagrange multiplier. The implementation of constants (5.7) in $\mathcal{L}_\rho(L,S,\mu)$ is not required as the iterates of ADMM do not produce negative S_{ij} due to the following reasons. It is straightforward to use an initial feasible solution for problem (5.12)-(5.13) solved by ADMM; the entries of input matrix M are $\{0,1\}$; the minimization of $\|L\|_*$ or $\mathrm{rank}(L)$ ensures entries of L cannot be too different. The ADMM iterates using the scaled form of augmented Lagrangian (5.22) are defined as:

$$\begin{aligned} L_J &= \mathrm{argmin}_L \ \mathcal{L}_\rho(L, S_{J-1}, \mu_{J-1}) \\ &= \mathrm{argmin}_L \ \rho\left(\frac{1}{\rho}\|L\|_* + \frac{1}{2}\|M-L-S_{J-1}+\frac{1}{\rho}\mu_{J-1}\|_F^2\right), \end{aligned}$$

where the iteration index J and its initialization are given in pages 74-75.

$$\begin{aligned} S_J &= \mathrm{argmin}_S \ \mathcal{L}_\rho(L_J, S, \mu_{J-1}), \ \textit{and} \\ &= \mathrm{argmin}_S \ \rho\left(\frac{\lambda}{\rho}\|C\circ S\|_1 + \frac{1}{2}\|M-L_J-S-\frac{1}{\rho}\mu_{J-1}\|_F^2\right). \end{aligned}$$

The Lagrangian multipliers μ_J is updated as follows,

$$\mu_J = \mu_{J-1} + \rho(M - L_J - S_J). \tag{5.23}$$

Since ρ is a constant, we minimize the following problem using the proximal operator in (2.21) for L_J:

$$\begin{aligned} L_J &= \text{argmin}_L \ \frac{1}{\rho}\|L\|_* + \frac{1}{2}\|M - L - S_{J-1} + \frac{1}{\rho}\mu_{J-1}\|_F^2 \\ &= SVT_{\frac{1}{\rho}}(M - S_{J-1} + \frac{1}{\rho}\mu_{J-1}) \qquad (5.24) \\ &= \bar{U}\mathcal{D}_{\frac{1}{\rho}}(\bar{\Sigma})\bar{U}^\top. \qquad (5.25) \end{aligned}$$

where $\mathcal{D}_{\frac{1}{\rho}}(\bar{\Sigma}) = Diag(\max\{(\sigma_i - \frac{1}{\rho}), 0\})$, assuming that the full SVD of $M - S_{J-1} + \frac{1}{\rho}\mu_{J-1}$ is defined by $M - S_{J-1} + \frac{1}{\rho}\mu_{J-1} = \bar{U}\bar{\Sigma}\bar{V}^\top$, where $\bar{U} \in \mathbb{R}^{N\times N}$, $\bar{V} \in \mathbb{R}^{N\times N}$ and $\bar{\Sigma} \in \mathbb{R}^{N\times N}$.

Similarly, the proximal operator (2.24) is used in finding S_J:

$$\begin{aligned} S_J &= \text{argmin}_S \ \frac{\lambda}{\rho}\|C \circ S\|_1 + \frac{1}{2}\|M - L_J - S - \frac{1}{\rho}\mu_{J-1}\|_F^2 \qquad (5.26) \\ &= STT_{\frac{\lambda}{\rho}}(M - L_J + \frac{1}{\rho}\mu_{J-1}, C) \\ &= \text{sign}\left(\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right) \max\left(\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| - \frac{\lambda}{\rho}(C_{J-1})_{ij}, 0\right). \qquad (5.27) \end{aligned}$$

With the above calculations of the variables $L,\ S$, and μ, the steps of the ADMM algorithm of problem (5.12)-(5.13) are summarized in **Algorithm** 5.

Algorithm 5 The ADMM algorithm for problem (5.12)-(5.13)

Input: Adjacency matrix M, regularization parameter $\lambda > 0$, penalty parameter $\rho > 0$, $\epsilon > 0$, and integer q

Initialization: Start with feasible $L_k = L_{J-1}$, $S_k = S_{J-1}$, $C = C_k (= C_{J-1})$, $\mu_{J-1} = 0$, $J = 1$, and $k = 0$

- Set $(C_{J-1})_{ij} = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$, for $i, j \in \{1, 2, ..., N\}$, since $(C_{J-1})_{ij} \in \partial\psi((S_{J-1})_{ij})$, where $(S_{J-1})_{ij} \in \{0, 1\}$

WHILE Stopping condition not satisfied

- $L_J = SVT_{\frac{1}{\rho}}(M - S_{J-1} + \frac{1}{\rho}\mu_{J-1})$
- $S_J = STT_{\frac{\lambda}{\rho}}(M - L_J + \frac{1}{\rho}\mu_{J-1}, C)$
- Update μ_J via (5.23), set $J = J + 1$
- If $\mod(J, q) = 0$, then update $(C_J)_{ij} = \frac{\epsilon}{((S_J)_{ij}+\epsilon)^2}$, set $L_k = L_J$, $S_k = S_J$, $C_k = C_J$, $\mu_k = \mu_J$, and $k = k + 1$.

ENDWHILE

Output: The recovered matrix L_J represents the planted clique

The stopping condition is given by

$$\|M - L_J - S_J\|_F \leq \delta,\ \delta > 0,\ \textit{a small constant}. \tag{5.28}$$

The 'regular' model (5.5)-(5.7) can be solved by adapting **Algorithm** 5 where the ADMM iterates are as follows:

$$L_J = SVT_{\frac{1}{\rho}}(M - S_{J-1} + \frac{1}{\rho}\mu_{J-1}),\ \textit{and} \tag{5.29}$$

$$S_J = ST_{\frac{\lambda}{\rho}}(M - L_J + \frac{1}{\rho}\mu_{J-1})$$
$$= \text{sign}\left(\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right) \max\left(\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| - \frac{\lambda}{\rho}, 0\right). \quad (5.30)$$

The update of μ_J is the same as in (5.23).

We compare $\lambda\|C \circ S\|_1$ in our model (5.12)-(5.13) with $\lambda\|S\|_1$ in the 'regular' matrix decomposition model (5.5)-(5.7), using the iterates of **Algorithm** 5. A comparison of ADMM iterates (5.27) and (5.30) of S_J shows that (5.27) carries additional information from $(J-1)$-th to J-th iteration via $(C_{J-1})_{ij} = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$. It follows from C_{J-1} that $(S_{J-1})_{ij} \longrightarrow 0 \Rightarrow (C_{J-1})_{ij} \longrightarrow 1/\epsilon$, $\epsilon << 1$.

Before making further comparisons, we look at the shrinkage parameter $\tau = \frac{\lambda}{\rho}$ in (5.27) and (5.30). The theoretical value $\lambda = \frac{1}{\sqrt{N}}$ has been suggested in [33]. Clearly, $\tau = \frac{\lambda}{\rho}$ is a small fraction provided $\rho > 1$ (a value we have implemented). Comparison of the shrinkage operators (5.27) and (5.30) suggests that if $(S_{J-1})_{ij}$ in previous iteration is small or close to zero then $(\frac{\lambda}{\rho})\,(C_{J-1})_{ij}$ in (5.27) is larger than $\frac{\lambda}{\rho}$ in (5.30). This implies that

$$(S_J)_{ij} = \text{sign}\left(\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right) \max\left(\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| - \frac{\lambda}{\rho}, 0\right),$$

has more chance of staying fractional than

$$(S_J)_{ij} = \text{sign}\left(\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right) \max\left(\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| - \frac{\lambda}{\rho}(C_{J-1})_{ij}, 0\right),$$

as $\frac{\lambda}{\rho}(C_{J-1})_{ij} > \frac{\lambda}{\rho}$. At later stages of the algorithm when (majority) entries of S_{J-1} approach towards zero at iteration $J-1$ then this information is fed into iteration J via C_{J-1} with $(C_{J-1})_{ij} > 1$, $(C_{J-1})_{ij} \longrightarrow \frac{1}{\epsilon}$ when $(S_{J-1})_{ij} \longrightarrow 0$.

This increases the likelihood of $\left(\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| - \frac{\lambda}{\rho}(C_{J-1})_{ij}\right)$ in (5.27) being negative, and thus making $(S_J)_{ij} = 0$. The iterate (5.30) of the regular model does not have this feature, and thus $(S_J)_{ij}$ remains a fraction if $\left|\left[M - L_J + \frac{1}{\rho}\mu_{J-1}\right]_{ij}\right| > \frac{\lambda}{\rho}$. On the other hand,

$(S_{J-1})_{ij}$ approaching 1 implies $(C_{J-1})_{ij} < 1$. However, in this case, the integer value of $(S_J)_{ij}$ is not an immediate event but rather a gradual optimization process.

5.3.5 The ADMM Algorithm for the TNN Model

Here, we describe the ADMM algorithm that we have used to solve (5.19)-(5.20).

The augmented Lagrangian of problem (5.19)-(5.20)is defined as follows:

$$\begin{aligned}\mathcal{L}_\rho(L,S,\mu) &= \|L\|_* - \mathrm{Tr}(ALA^\top) + \lambda\|C\circ S\|_1 + \langle \mu, M-L-S\rangle + \frac{\rho}{2}\|M-L-S\|_F^2 \\ &= \|L\|_* - \mathrm{Tr}(ALA^\top) + \lambda\|C\circ S\|_1 + \frac{\rho}{2}\|M-L-S+\frac{1}{\rho}\mu\|_F^2 - \frac{1}{2\rho}\|\mu\|_F. \quad (5.31)\end{aligned}$$

The ADMM iterates are as follows:

$$\begin{aligned}L_J &= \mathrm{argmin}_L \ \mathcal{L}_{\rho_{J-1}}(L, S_{J-1}, \mu_{J-1}) \\ &= \mathrm{argmin}_L \ \ \rho_{J-1}\left(\frac{1}{\rho_{J-1}}\|L\|_* + \frac{1}{2}\|M-L-S_{J-1}+\frac{1}{\rho_{J-1}}(\mu_{J-1}-A^\top A)\|_F^2\right) \\ &= SVT_{\frac{1}{\rho_{J-1}}}(M-S_{J-1}+\frac{1}{\rho_{J-1}}\left(\mu_{J-1}-A^\top A\right)).\end{aligned}$$

$$\begin{aligned}S_J &= \mathrm{argmin}_S \ \mathcal{L}_{\rho_{J-1}}(L_J, S, \mu_{J-1}), \\ &= \mathrm{argmin}_S \ \ \rho_{J-1}\left(\frac{\lambda}{\rho_{J-1}}\|C\circ S\|_1 + \frac{1}{2}\|M-L_J-S-\frac{1}{\rho_{J-1}}\mu_{J-1}\|_F^2\right) \\ &= STT_{\frac{\lambda}{\rho_{J-1}}}(M-L_J+\frac{1}{\rho_{J-1}}\mu_{J-1}, C).\end{aligned}$$

The Lagrangian multipliers μ_J are updated as follows,

$$\mu_J = \mu_{J-1} + \rho_{J-1}(M-L_J-S_J). \quad (5.32)$$

We compute the parameter ρ as follows:

$$\rho_k = \min(\varrho\rho_{k-1}, \rho_{max}), \tag{5.33}$$

where $\varrho > 1$ (e.g. $\varrho = 0.00001$) is a constant, and $\rho_{\max}$ (e.g. $\rho_{\max} = 10^5$) is the maximum value of ρ.

With the above calculations of the variables $L,\ S$, and μ, the steps of the ADMM algorithm for problem (5.19)-(5.20) are summarized in **Algorithm** 6, where we have the parameters ρ, $\rho_{\max}$, ϱ, ε, and q.

Algorithm 6 The ADMM algorithm for the PCP and the MCP

Input: Adjacency matrix M, regularization parameter $\lambda > 0$, $\rho, \rho_{max} > 0$, $\delta > 0$, $\varrho > 0$, $\varepsilon > 0$, and integer q

Step 1: Given M, find $[U, \Sigma, U] = SVD(M)$, where $U = (u_1, u_2, ..., u_N) \in \mathbb{R}^{N\times N}$, and $A = (u_1, u_2, ..., u_t)^\top$.

Step 2: Start with feasible $L_k = L_{J-1}$, $S_k = S_{J-1}$, $C = C_k (= C_{J-1})$, $\mu_{J-1} = 0$, $k = 0$, and $J = 1$

- Set $(C_{J-1})_{ij} = \frac{\epsilon}{((S_{J-1})_{ij}+\epsilon)^2}$, for $i, j \in \{1, 2, ..., N\}$,

WHILE Stopping condition (5.28) not satisfied

- $L_J = SVT_{\frac{1}{\rho_{J-1}}}(M - S_{J-1} + \frac{1}{\rho_{J-1}}(\mu_{J-1} - A^\top A))$
- $S_J = STT_{\frac{\lambda}{\rho_{J-1}}}(M - L_J + \frac{1}{\rho_{J-1}}\mu_{J-1}, C)$
- Update μ_J via (5.32), ρ_J via (5.33), set $J = J + 1$
- If $\mod(J, q) = 0$, then update $(C_J)_{ij} = \frac{\epsilon}{((S_J)_{ij}+\epsilon)^2}$, set $L_k = L_J$, $S_k = S_J$, $C_k = C_J$, $\mu_k = \mu_J$, and $J = J + 1$.

ENDWHILE

Output: The recovered matrix L_J represents the planted clique

---

The 'regular' model (5.16)-(5.18) can be solved by adapting **Algorithms** 6 where the ADMM iterates are as follows:

$$L_J = SVT_{\frac{1}{\rho_{J-1}}}(M - S_{J-1} + \frac{1}{\rho_{J-1}}(\mu_{J-1} - A^\top A)), \text{ and} \tag{5.34}$$

$$S_J = ST_{\frac{\lambda}{\rho_{J-1}}}(M - L_J + \frac{1}{\rho_{J-1}}\mu_{J-1}). \tag{5.35}$$

The update of μ_J is the same as in (5.32).

5.4 Convergence Analysis

Here, we show that the proposed algorithms for both of our models converge to the optimal solution. For simplicity, we consider the square matrices in our proofs, i.e., the PCP and MCP, and for non-square matrices, i.e., the PEBP and MEBP, the proof follows. All the theorems in this section hold for a very small parameter $\epsilon > 0$, as described in Section 5.3.1.

5.4.1 Convergence of the NN Model Algorithm

For the NN-based matrix decomposition model algorithm, **Algorithm** 5, we obtain the following theorems.

Theorem 23. *Any accumulation point (L_*, S_*) of the sequence $\{(L_J, S_J)\}$ generated by **Algorithm** 5 is an optimal solution of* (5.12)-(5.13)*, with index J corresponding to large epoch k.*

Proof. **Algorithm** 5 computes (L_J, S_J) by alternate minimization with respect to one variable while keeping the other one fixed. The problem being convex, for sufficiently large J and μ_{J-1} close to μ_*, and using (5.22) we get,

$$
\begin{aligned}
\mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) &= \min_{L,S\in\mathbb{R}^{N\times N}} \mathcal{L}_\rho(L, S, \mu_{J-1}) \\
&= \min_{L,S\in\mathbb{R}^{N\times N}} \left(\|L\|_* + \lambda\|C_J \circ S\|_1 + \frac{\rho}{2}\|M - L - S + \frac{1}{\rho}\mu_{J-1}\|_F^2 - \frac{1}{2\rho}\|\mu_{J-1}\|_F^2 \right) \\
&\leq \min_{L,S\in\mathbb{R}^{N\times N}} \left(\|L\|_* + \lambda\|C_J \circ S\|_1 + \frac{\rho}{2}\|M - L - S + \frac{1}{\rho}\mu_{J-1}\|_F^2 \right) \\
&\leq \min_{L,S\in\mathbb{R}^{N\times N},\ L+S=M} \left(\|L\|_* + \lambda\|C_J \circ S\|_1 + \frac{\rho}{2}\|M - L - S + \frac{1}{\rho}\mu_{J-1}\|_F^2 \right) \\
&= \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 + \frac{1}{2\rho}\|\mu_{J-1}\|_F^2,
\end{aligned}
$$

since (L_*, S_*) is the optimal solution of (5.12)-(5.13), and C_* is the optimal value for C. It follows that

$$
\mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) \leq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 + \varepsilon_1,\ \varepsilon_1 > 0, \tag{5.36}
$$

where ρ is large, μ_{J-1} is bounded [74, 120] and $\varepsilon_1 = \frac{1}{2\rho}\|\mu_{J-1}\|_F^2$.

Now it follows from the unscaled Lagrangian in (5.21), and (5.23) that

$$
\begin{aligned}
\|L_J\|_* + \lambda\|C_J \circ S_J\|_1 &= \mathcal{L}_\rho\left(L_J, S_J, \mu_{J-1}\right) - \frac{1}{2\rho}\left(\|\mu_J\|_F^2 - \|\mu_{J-1}\|_F^2\right) && (5.37) \\
&\leq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 + \varepsilon_1 + \frac{1}{2\rho}\left(\|\mu_{J-1}\|_F^2 - \|\mu_J\|_F^2\right) && (5.38) \\
&\leq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 + \varepsilon_1 + \frac{1}{2\rho}\|\mu_{J-1} - \mu_J\|_F^2. && (5.39)
\end{aligned}
$$

Thus we have

$$
\|L_J\|_* + \lambda\|C_J \circ S_J\|_1 \leq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 + \hat{\varepsilon_1},\ \hat{\varepsilon_1} > 0, \tag{5.40}
$$

where we have used $\hat{\varepsilon_1} = \varepsilon_1 + \frac{1}{2\rho}\|\mu_{J-1} - \mu_J\|_F^2$ using the boundedness of $\{\mu_J\}$ and large ρ. The inequality in (5.38) follows from (5.36) and the inequality (5.39) follows from the reverse triangular inequality applied to μ_J and μ_{J-1} (reverse triangle inequality holds for any matrix norm [92, 176]).

Using the reverse triangular inequality, $\|A - B\|_* \geq \|A\|_* - \|B\|_*$, of the nuclear norm, and by the optimizer (L_*, S_*) we get

$$\begin{aligned}\|L_J\|_* + \lambda\|C_J \circ S_J\|_1 &\geq \|M - S_J\|_* - \|M - L_J - S_J\|_* + \lambda\|C_J \circ S_J\|_1 \\ &\geq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 - \|M - L_J - S_J\|_* \\ &= \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 - \frac{1}{\rho}\left(\|\mu_J - \mu_{J-1}\|_*\right), \text{ by (5.23)} \end{aligned} \tag{5.41}$$

where we have used $A = M - S_J$, $B = M - L_J - S_J$, and assumed $M - S_J$ being the deviation from L_* (noting that $M - S_* = L_*$) and hence $\|M - S_J\|_* \geq \|L_*\|_*$. This together with the boundedness of $\{\mu_J\}$ imply

$$\|L_J\|_* + \lambda\|C_J \circ S_J\|_1 \geq \|L_*\|_* + \lambda\|C_* \circ S_*\|_1 - \hat{\varepsilon_2},\ \hat{\varepsilon_2} > 0, \tag{5.42}$$

where $\hat{\varepsilon_2} = \frac{1}{\rho}\|\mu_J - \mu_{J-1}\|_*$.

Thus, from equations (5.40) and (5.42) we have,

$$-\hat{\varepsilon} \leq \Big(\|L_J\|_* + \lambda\|C_J \circ S_J\|_1\Big) - \Big(\|L_*\|_* + \lambda\|C_* \circ S_*\|_1\Big) \leq \hat{\varepsilon},$$

where $\hat{\varepsilon} = \max\{\hat{\varepsilon}_1, \hat{\varepsilon}_2\}$. This implies

$$\left| \Big(\|L_J\|_* + \lambda\|C_J \circ S_J\|_1\Big) - \Big(\|L_*\|_* + \lambda\|C_* \circ S_*\|_1\Big) \right| \leq \hat{\varepsilon}.$$

For sufficiently large J, $\hat{\varepsilon} \longrightarrow 0$ and $\|L_J\|_* + \lambda\|C_J \circ S_J\|_1$ converges to $\|L_*\|_* + \lambda\|C_* \circ S_*\|_1$. □

In addition, we have established another convergence result as stated in the following theorem by demonstrating that any limit point in an iteration sequence generated by **Algorithm** 5 is a KKT point.

Theorem 24. *The limit point (L_*, S_*) of the sequence $\{(L_J, S_J)\}$ generated by **Algorithm** 5 is the KKT point for problem* (5.12)-(5.13).

Proof. We begin by showing the boundedness of $\{L_J\}$ and $\{S_J\}$. It follows that

$$\begin{aligned}\mathcal{L}_\rho(L_{J+1}, S_{J+1}, \mu_J) &\leq \mathcal{L}_\rho(L_{J+1}, S_J, \mu_J) \\ &\leq \mathcal{L}_\rho(L_J, S_J, \mu_J) = \mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) + \langle \mu_J - \mu_{J-1}, M - L_J - S_J \rangle \\ &= \mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) + \frac{1}{\rho}\|\mu_J - \mu_{J-1}\|_F^2,\end{aligned} \tag{5.43}$$

where we have used the augmented Lagrangian (5.21); the equality in (5.43) follows by writing $\mathcal{L}_\rho(L_J, S_J, \mu_J) = \|L_J\|_* + \lambda\|C_J \circ S_J\|_1 + \langle \mu_{J-1}, M - L_J - S_J \rangle + \frac{\rho}{2}\|M - L_J - S_J\|_F^2 + \langle \mu_J, M - L_J - S_J \rangle - \langle \mu_{J-1}, M - L_J - S_J \rangle$. Then by recalling the boundedness of $\{\mu_J\}$, $\frac{1}{\rho}\|\mu_J - \mu_{J-1}\|_F^2 < \infty$, we have

$$\mathcal{L}_\rho(L_{J+1}, S_{J+1}, \mu_J) - \mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) < \infty, \forall J. \tag{5.44}$$

Thus, it follows from (5.36) and (5.44) that $\mathcal{L}_\rho(L_J, S_J, \mu_{J-1})$ is bounded. On the other hand, from (5.37) we have

$$\|L_J\|_* + \lambda\|C_J \circ S_J\|_1 = \mathcal{L}_\rho(L_J, S_J, \mu_{J-1}) - \frac{1}{2\rho}(\|\mu_J\|_F^2 - \|\mu_{J-1}\|_F^2),$$

for which $\|\mu_J\|_F^2 - \|\mu_{J-1}\|_F^2$ and $\mathcal{L}_\rho(L_J, S_J, \mu_{J-1})$ are bounded. Hence $\|L_J\|_*$ and $\lambda\|C_J \circ S_J\|_1$ are also bounded. Therefore, both $\{L_J\}$ and $\{S_J\}$ are bounded.

We have shown in Theorem 23 that $\{(L_J, S_J)\}$ converges to (L_*, S_*) for $J \longrightarrow \infty$. The KKT conditions of problem (5.12)-(5.13) are

$$\begin{cases} 0 \in \partial\|L_*\|_* - \mu_*, \\ 0 \in \lambda\partial\|C_* \circ S_*\|_1 - \mu_*, \\ M - L_* - S_* = 0. \end{cases} \tag{5.45}$$

By the boundedness of $\{\mu_J\}$, $\{L_J\}$ and $\{S_J\}$ we have

$$\lim_{J\longrightarrow\infty}(M-L_J-S_J)=\frac{1}{\rho}\lim_{J\longrightarrow\infty}(\mu_J-\mu_{J-1})=0, \tag{5.46}$$

thus, we have $M-L_*-S_*=0$.

The optimizers L_J and S_J of the sub-problems using (5.22) at the J-th iteration of ADMM imply

$$0\in\partial_L\mathcal{L}_\rho(L_J,S_J,\mu_{J-1}),\ \ 0\in\partial_S\mathcal{L}_\rho(L_J,S_J,\mu_{J-1}),$$

that is

$$0\in\partial\|L_J\|_*-\mu_{J-1}-\rho(M-L_J-S_J),\ \ 0\in\lambda\partial\|C_J\circ S_J\|_1-\mu_{J-1}-\rho(M-L_J-S_J).$$

Hence, there exist $Y_J\in\partial\|L_J\|_*$, and $Z_J\in\lambda\partial\|C_J\circ S_J\|_1$ such that

$$Y_J-\mu_{J-1}-\rho(M-L_J-S_J)=0,\ \ Z_J-\mu_{J-1}-\rho(M-L_J-S_J)=0, J\longrightarrow\infty.$$

It follows from (5.46) that $Y_*-\mu_*=0$, and $Z_*-\mu_*=0$, $J\longrightarrow\infty$. This implies that

$$\begin{cases}\mu_*\in\partial\|L_*\|_*,\ \ \mu_*\in\lambda\partial\|C_*\circ S_*\|_1,\\ 0\in\partial\|L_*\|_*-\mu_*,\ \ 0\in\lambda\partial\|C_*\circ S_*\|_1-\mu_*.\end{cases}$$

Now we can see that (L_*,S_*,μ_*) satisfies the KKT conditions (5.45). □

5.4.2 Convergence of the TNN Model Algorithm

For the TNN-based matrix decomposition model algorithm, **Algorithm** 6, we obtain the following theorems.

Theorem 25. *Assume that (L_*,S_*,μ_*) and (L_J,S_J,μ_J) are the optimal solution to problem (5.19)-(5.20). and the iterative solution to **Algorithm** 6, respectively. Then for large J, $J\longrightarrow\infty$, $|P_*-P_J|\longrightarrow 0$ and (L_J,S_J,μ_J) converges to (L_*,S_*,μ_*), where $P_*=\|L_*\|_*-\mathrm{Tr}(AL_*A^\top)+\lambda\|C_*\circ S_*\|_1$ and $P_J=\|L_J\|_*-\mathrm{Tr}(AL_JA^\top)+\lambda\|C_J\circ S_J\|_1$.*

Proof. From the scaled augmented Lagrangian in (5.31), we have

$$\mathcal{L}_{\rho_*}(L_*, S_*, \mu_*) = \|L_*\|_* - \mathrm{Tr}(AL_*A^\top) + \lambda\|C_* \circ S_*\|_1 + \frac{\rho_*}{2}\|M - L_* - S_* + \frac{1}{\rho_*}\mu_*\|_F^2 - \frac{1}{2\rho_*}\|\mu_*\|_F \tag{5.47}$$

$$= \|L_*\|_* - \mathrm{Tr}(AL_*A^\top) + \lambda\|C_* \circ S_*\|_1, \text{ since } M = L_* + S_* \tag{5.48}$$

$$= P_*. \tag{5.49}$$

From the other hand, we have

$$\begin{aligned}\mathcal{L}_{\rho_*}(L_J, S_J, \mu_*) &= \|L_J\|_* - \mathrm{Tr}(AL_JA^\top) + \lambda\|C_J \circ S_J\|_1 \\ &\quad + \frac{\rho_*}{2}\|M - L_J - S_J + \frac{1}{\rho_*}\mu_*\|_F^2 - \frac{1}{2\rho_*}\|\mu_*\|_F \qquad (5.50)\\ &= P_J + \frac{\rho_*}{2}\|M - L_J - S_J + \frac{1}{\rho_*}\mu_*\|_F^2 - \frac{1}{2\rho_*}\|\mu_*\|_F. \qquad (5.51)\end{aligned}$$

Since (L_*, S_*, μ_*) is the optimal solution of problem (5.19)-(5.20)., then

$$\mathcal{L}_{\rho_*}(L_*, S_*, \mu_*) \leq \mathcal{L}_{\rho_*}(L_J, S_J, \mu_*) \tag{5.52}$$

This together with (5.49) and (5.51) implies that

$$\begin{aligned}P_* &\leq P_J + \frac{\rho_*}{2}\|M - L_J - S_J + \frac{1}{\rho_*}\mu_*\|_F^2 - \frac{1}{2\rho_*}\|\mu_*\|_F^2 \\ &= P_J + \frac{\rho_*}{2}\|d_J + \frac{1}{\rho_*}\mu_*\|_F^2 - \frac{1}{2\rho_*}\|\mu_*\|_F^2 \\ &\leq P_J + \frac{\rho_*}{2}\|d_J\|_F^2, \text{ using the triangle inequality on } \frac{\rho_*}{2}\|d_J + \frac{1}{\rho_*}\mu_*\|_F^2,\end{aligned}$$

where $d_J = M - L_J - S_J$.

Thus, we have

$$P_* - P_J \leq \frac{\rho_*}{2}\|d_J\|_F^2. \tag{5.53}$$

Now, we study the sub-differential of the scaled augmented Lagrangian in (5.31) at $(L_J, S_{J-1}, \mu_{J-1})$

$$\begin{aligned}
0 &\in \partial\mathcal{L}_{\rho_{J-1}}(L_J, S_{J-1}, \mu_{J-1}) \\
&= \partial\|L_J\|_* - A^\top A - \rho_{J-1}(M - L_J - S_{J-1} + \frac{1}{\rho_{J-1}}\mu_{J-1}) \\
&= \partial\|L_J\|_* - A^\top A - \rho_{J-1}(S_J - S_{J-1} + d_J + \frac{1}{\rho_{J-1}}\mu_{J-1}),\ \textit{since } M - L_J = S_J + d_J,
\end{aligned}$$

this implies that L_J minimizes

$$\|L\|_* - \mathrm{Tr}(ALA^\top) - \langle \mu_{J-1} + \rho_{J-1}(S_J - S_{J-1} + d_J), L\rangle. \tag{5.54}$$

Then, for the optimal solution L_* we have

$$\begin{aligned}
&\|L_J\|_* - \mathrm{Tr}(AL_JA^\top) - \langle \mu_{J-1} + \rho_{J-1}(S_J - S_{J-1} + d_J), L_J\rangle \\
&\quad\leq \|L_*\|_* - \mathrm{Tr}(AL_*A^\top) - \langle \mu_{J-1} + \rho_{J-1}(S_J - S_{J-1} + d_J), L_*\rangle,
\end{aligned} \tag{5.55}$$

since L_J minimizes (5.54).

Now, we study the sub-differential of the augmented Lagrangian in (5.31) at (L_J, S_J, μ_{J-1}),

$$\begin{aligned}
0 &\in \partial\mathcal{L}_{\rho_{J-1}}(L_J, S_J, \mu_{J-1}) \\
&= \partial\lambda\|C_J \circ S_J\|_1 - \rho_{J-1}(M - L_J - S_J + \frac{1}{\rho_{J-1}}\mu_{J-1}) \\
&= \partial\lambda\|C_J \circ S_J\|_1 - \mu_{J-1} - \rho_{J-1}(M - L_J - S_J) \\
&= \partial\lambda\|C_J \circ S_J\|_1 - \mu_J,\ \textit{using (5.32)}.
\end{aligned}$$

This implies that S_J minimizes

$$\lambda\|C_J \circ S\|_1 - \langle \mu_J, S\rangle. \tag{5.56}$$

Then, for the optimal solution S_* we have

$$\lambda\|C_J \circ S_J\|_1 - \langle \mu_J, S_J\rangle \leq \lambda\|C_* \circ S_*\|_1 - \langle \mu_J, S_*\rangle, \tag{5.57}$$

since S_J minimizes (5.56).

By combining (5.55) and (5.57) and using the definition of P_* and P_J, we get,

$$\begin{aligned}
P_J - P_* &\leq \langle \mu_{J-1} + \rho_{J-1}(S_J - S_{J-1} + d_J), L_J\rangle - \langle \mu_{J-1} + \rho_{J-1}(S_J - S_{J-1} + d_J), L_*\rangle \\
&\qquad + \langle \mu_J, S_J\rangle - \langle \mu_J, S_*\rangle \\
&= \langle \mu_{J-1} + \rho_{J-1}(S_{J-1} - S_{J-1} + d_J), L_J - L_*\rangle + \langle \mu_J, S_J - S_*\rangle \\
&= \langle \mu_{J-1}, L_J - L_*\rangle + \rho_{J-1}\langle S_J - S_{J-1} + d_J, L_J - L_*\rangle + \langle \mu_J, S_J - S_*\rangle \\
&= \langle \mu_{J-1}, L_J - L_*\rangle + \rho_{J-1}\langle S_J - S_{J-1} + d_J, L_J - L_*\rangle + \langle \mu_{J-1} + \rho_{J-1}d_J, S_J - S_*\rangle, \text{ \textit{using} (5.32)}, \\
&= \rho_{J-1}\langle S_J - S_{J-1} + d_J, L_J - L_*\rangle + \langle \mu_{J-1}, L_J + S_J - (L_* + S_*)\rangle + \langle \rho_{J-1}d_J, S_J - S_*\rangle \\
&= \rho_{J-1}\langle S_J - S_{J-1} + d_J, L_J - M + S_*\rangle + \langle \mu_{J-1}, L_J + S_J - M\rangle + \langle \rho_{J-1}d_J, S_J - S_*\rangle \\
&= \rho_{J-1}\langle S_J - S_{J-1} + d_J, S_* - S_J - d_J\rangle + \langle \mu_{J-1}, -d_J\rangle + \langle \rho_{J-1}d_J, S_J - S_*\rangle.
\end{aligned}$$

Then, it follows that

$$P_* - P_J \geq \langle \mu_{J-1}, d_J\rangle + \rho_{J-1}\langle S_J - S_{J-1} + d_J, S_J - S_* + d_J\rangle + \langle \rho_{J-1}d_J, S_J - S_*\rangle. \tag{5.58}$$

Thus, by combining (5.53) and (5.58), it follows that when $J \longrightarrow \infty$, $d_J \longrightarrow 0$, and $S_J - S_{J-1} \longrightarrow 0$, which implies that $|P_* - P_J| \longrightarrow 0$. Thus, (L_J, S_J, μ_J) converges to (L_*, S_*, μ_*).

□

In addition, we have established another convergence result as stated in the following theorem by demonstrating that any limit point in an iteration sequence generated by **Algorithm** 6 is a KKT point.

Theorem 26. *Assume that μ_J is bounded and satisfies $\lim_{J\to\infty} \mu_J - \mu_{J-1} = 0$. Any accumulation point (L_*, S_*, μ_*) of the sequence (L_J, S_J, μ_J) generated along the iterations of **Algorithm** 6 satisfies the KKT conditions.*

Proof. We have demonstrated in Theorem 25 that $\{(L_J, S_J)\}$ converges to (L_*, S_*) for $J \longrightarrow \infty$. The KKT conditions of problem (5.19)-(5.20). are

$$\begin{cases} 0 \in \partial\|L_*\|_* - A^\top A - \mu_*, \\ 0 \in \lambda\partial\|C_* \circ S_*\|_1 - \mu_*, \\ M - L_* - S_* = 0. \end{cases} \tag{5.59}$$

By the boundedness of $\{\mu_J\}$ we have

$$\lim_{J\longrightarrow\infty} (M - L_J - S_J) = \frac{1}{\rho_{J-1}} \lim_{J\longrightarrow\infty} (\mu_J - \mu_{J-1}) = 0, \tag{5.60}$$

thus, we have $M - L_* - S_* = 0$.

The optimizers L_J and S_J of the sub-problems using (5.31) at the Jth iteration of ADMM imply

$$0 \in \partial_L \mathcal{L}_{\rho_{J-1}}(L_J, S_J, \mu_{J-1}), \;\; 0 \in \partial_S \mathcal{L}_{\rho_{J-1}}(L_J, S_J, \mu_{J-1}),$$

that is

$$0 \in \partial\|L_J\|_* - A^\top A - \mu_{J-1} - \rho_{J-1}(S_J - S_{J-1}), \;\; 0 \in \lambda\partial\|C_J \circ S_J\|_1 - \mu_J.$$

Hence, there exist $Y_J \in \partial\|L_J\|_*$, and $Z_J \in \lambda\partial\|C_J \circ S_J\|_1$ such that

$$Y_J - A^\top A - \mu_{J-1} - \rho_{J-1}(S_J - S_{J-1}) = 0, \;\; Z_J - \mu_{J-1} - \rho_{J-1}(M - L_J - S_J) = 0, J \longrightarrow \infty$$

, *as* $S_J - S_{J-1} \longrightarrow 0$. It follows from (5.60) that $Y_* - \mu_* = 0$, and $Z_* - \mu_* = 0$, $J \longrightarrow \infty$. This implies that

$$\begin{cases} \mu_* \in \partial\|L_*\|_* - A^\top A, \;\; \mu_* \in \lambda\partial\|C_* \circ S_*\|_1, \\ 0 \in \partial\|L_*\|_* - A^\top A - \mu_*, \;\; 0 \in \lambda\partial\|C_* \circ S_*\|_1 - \mu_*. \end{cases}$$

Now we can see that (L_*, S_*, μ_*) satisfies the KKT conditions (5.59).

□

Chapter 6

Theoretical Guarantee and Recovery

6.1 Introduction

This Chapter presents our theoretical guarantee and recovery for the Planted Clique Problem (PCP), the Planted Edge Bi-clique Problem (PEBP), and the Maximum Clique Problem (MCP) and the Maximum Edge Bi-clique Problem (MEBP) in random graphs. This has been done through some assumptions regarding the incoherence conditions of the low-rank matrix L_* and assumptions on the random signs and support of S_*, based on this we have established our main theorem. The proof of our main theorem has been done through, firstly, by establishing some optimality conditions for proving the solution's uniqueness; secondly, by constructing dual certificates. This construction has been carried out for low-rank and sparse matrices, using the golfing scheme and the least square method, respectively. Then, to conclude this Chapter, we validate both certificates.

6.2 Technical Assumptions of the Low-Rank and Sparse Matrices and Main Result

We begin with some preliminaries. Let the rank of symmetric L_* be r. Hence, L_* is orthogonally diagonalizable. Then $L_* = U\Sigma U^\top$, $U = [u_1, u_2, ..., u_r]$, where u_l is the l-th singular vector of L_*. $\Sigma = \mathrm{Diag}(\sigma_1, \sigma_2, ..., \sigma_r)$ where σ_l is the l-th singular value of L_*. Assume that S_* contains m non-zero entries, i.e., $|supp(S_*)| = m$. We can easily see that the support sets of S_* and $C_* \circ S_*$ are the same and hence $|supp(S_*)| = |supp(C_* \circ S_*)|$. Assume that L_*, C_*, and S_* are optimal to problem (5.12)-(5.13).

We now discuss the regularization parameter λ and the rank-sparsity incoherence in the context of our problem. The value of λ and the satisfaction of incoherence conditions play central role in the recovery of (L_*, S_*).

The value $\lambda = \frac{1}{\sqrt{N}}$ suggested in [33] follows the inverse square root law. We would like to make λ reliant on the prior information about the problem at hand. In particular, we use the size n of the clique and define λ to be $\lambda = \frac{\frac{N}{m}}{\sqrt{N}} < 1$. Then λ follows inverse square root law provided that $\frac{N}{m} < 1$. The value of m must obey $m < (N^2 - n^2)$ since n is the clique size in L_* and $S_* = M - L_*$. We make a reasonable choice for the size of the sparsity by taking $m = p(N^2 - n^2)$; $p = \frac{1}{2}$ is generally used in the PCP [5]. We restrict our planted clique size such that $c_1 N \leq n \leq c_2 N$, $n = cN$, where $c = [c_1, c_2] \approx [0.1, 0.9]$. It follows that

$$\frac{m}{N} = \frac{p\left(N^2 - n^2\right)}{N} = \frac{p\left(N^2 - c^2 N^2\right)}{N} = p\left(1 - c^2\right) N > 1, \textit{ for large } N.$$

The above inequality holds even for $N = 15$, $c = 0.9$, and $p = 0.1$. Hence $\frac{N}{m} < 1$ holds. Our choice of λ is therefore given by

$$\lambda = \frac{\frac{N}{m}}{\sqrt{N}} = \frac{\alpha}{\sqrt{N}}, \textit{ where } \alpha = \frac{N}{m} < 1. \tag{6.1}$$

We have estimated a range, $[0.0021, 0.0914]$, of values of α numerically by plotting α against N for a number of clique sizes n in $m = p(N^2 - n^2)$, see Chapter 7.

6.2.1 Assumption A: Incoherence Conditions for L_*

We now present the incoherence conditions. By construction, M, L_* and S_* are all symmetric matrices in the case of both PCP and MCP. With this prior information we now present the conditions on L_* and S_* for their guaranteed recovery. We begin with the rank-one matrix L_*. It can be easily to see that $\sigma_1 = \|L_*\|_* = \sqrt{\sum_{i=1}^{N} \sigma_i^2} = \|L_*\|_F = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} (L_*)_{ij}^2} = n$, since $\sigma_i = 0$, $i = 2, 3, ..., N$. Consequently, the elements of $UU^\top$ are from $\{0, \frac{1}{n}\}$ and the elements of U are from $\{0, \frac{1}{\sqrt{n}}\}$, since $L_* = nUU^\top$. The column space of L_* not aligned with the canonical basis vectors is guaranteed with the condition [33]

$$\max_i \|U^\top e_i\|_2^2 \leq \frac{\mu_0 r}{N}, \quad \forall i, \quad 1 \leq \mu_0 \leq \frac{N}{r}.$$

It is easy to see that

$$\frac{1}{n} = \max_i \|U^\top e_i\|_2^2 \leq \frac{\mu_0 r}{N}, \ \ \forall i, \tag{6.2}$$

holds for $1 \leq \mu_0 \leq \frac{N}{r}$.

The joint incoherent condition [33] is defined by

$$\|UV^\top\|_\infty \leq \sqrt{\frac{\mu_1 r}{N^2}}, \ \mu_1 = \mu_0^2 r, \ 1 \leq \mu_0 \leq \frac{N}{r}. \tag{6.3}$$

It is easy to see that the above condition also holds for $1 \leq \mu_0 \leq \frac{N}{r}$ for our problem, since

$$\frac{1}{n} = \|UU^\top\|_\infty \leq \frac{\mu_0 r}{N}. \tag{6.4}$$

For the PEBP and the MEBP, the matrices M_b, L, and S are not symmetric. We can easily see that the only non-zero singular value (the maximum singular value) of L_* is $\sqrt{nm}$. That is $\sigma_1 = \|L_*\| = \|L_*\|_* = \sqrt{\sum_{i=1}^{\min(N,M)} \sigma_i^2} = \|L_*\|_F = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{M}(L_*)_{ij}^2} = \sqrt{nm}$, since $\sigma_i = 0, \ i = 2, 3, ..., \min(N, M)$. It follows that the elements of $UU^\top$ and $VV^\top$ are from $\{0, \frac{1}{n}\}$ and $\{0, \frac{1}{m}\}$, respectively, and the elements of U and V are from $\{0, \frac{1}{\sqrt{n}}\}$ and $\{0, \frac{1}{\sqrt{m}}\}$, respectively, since $L_* = \sqrt{nm}UV^\top$.

For the perfect disentanglement of the low-rank and the sparse components, we need to assume the low-rank component L_* is not sparse. This can be accomplished by ensuring that the columns space of L_* are unaligned with the standard basis vectors. This is guaranteed by the following conditions with a parameter μ:

$$\frac{1}{n} = \max_j \|U^\top e_j\|_2^2 \leq \frac{\mu_0 r}{N}, \quad \frac{1}{m} = \max_j \|V^\top e_j\|_2^2 \leq \frac{\mu_1 r}{M}, \forall j, \tag{6.5}$$

hold for $1 < \mu_0 \leq \frac{N}{r}$ and $1 < \mu_1 \leq \frac{M}{r}$, respectively.

By construction of M_b and L_*, $UV^\top$ has nm non-zero entries, each of the entries is $\frac{1}{\sqrt{nm}}$. It is also easy to see for our problem that the joint incoherent condition holds when $1 < \mu \leq \frac{\sqrt{NM}}{r}$, since

$$\frac{1}{\sqrt{nm}} = \|UV^\top\|_\infty \leq \sqrt{\frac{\mu r}{NM}}, \quad 1 < \mu \leq \frac{\sqrt{NM}}{r}. \tag{6.6}$$

The equations in (6.5) and (6.6) show that both incoherent conditions hold for our matrix L_*.

6.2.2 Assumption B: Random Signs and Support for S_*

For the guaranteed recovery of (L_*, S_*), for the PCP and MCP, the condition on S_* is that its sparsity pattern is not too structured. This can be achieved by considering Bernoulli model with probability ρ. However S_* must be symmetric and its construction is such that $|supp(S_*)| < N^2 - n^2$; the probability ρ can be adjusted for this support. These properties are needed to ensure feasibility of constraint (5.6). Hence, we work with the empirical probability p and the construction of which is as follows. We divide the set of entry locations of S_* into three sets, $A_1 = \{(i,j) \mid i < j\}$, $A_2 = \{(i,j) \mid i > j\}$, and $A_3 = \{(i,j) \mid i = j\}$, $\forall i, j \in \{1, 2, ..., N\}$. We then apply Bernoulli model in the set $A_1 \cup A_3$. The entry values corresponding to A_1 are then copied to set A_2. We then calculate the empirical probability p using entries in $S_* = A_1 \cup A_2 \cup A_3$, treating the entry values as the results of random experiments[1].

To see the structure of sparsity pattern of S_* we calculate the variance of the elements of each row or column. Each component of a row associates a random variable which assumes 1 with probability p and 0 with probability $1 - p$. This implies that the mean and variance of the random variables are p and $p(1-p)$. The expected cardinality of a row or column is Np, and similarly $Np(1-p)$ for the variance. Hence we can see that no pattern is guaranteed since the expected value $Np(1-p)$ is the same for every column. That is

[1]Since non-zero entries of S_* are formed using Bernoulli probability model and $M - S_* = L_*$, the planted clique location can be considered random.

$$\left|Var(S_{i*}) - Var(S_{j*})\right| < \varrho, \tag{6.7}$$

for any $\varrho > 0$, where $Var(S_{i*})$ represents the variance of entries in the column i.

For the PEBP and MEBP, the second condition for guaranteed recovery is that the sparsity pattern of S_* must not be too structured. To achieve this, Bernoulli model with probability p will be considered. We construct S_* such that $|supp(S_*)| < p(NM - nm)$, this due to the fact that nm entries out of NM are fixed to be ones for the bi-clique; in this case, we can adjust the probability p for the support of S_*. Thus, we use probability p to construct S_* such that $(S_{ij})_* = 1$ with probability p and 0 with probability $1 - p$. To determine the structure of the sparsity pattern in S_*, the variances of each of the columns or rows are calculated. Components of each row are associated with a random variable that assumes 1 with probability p and 0 with probability $1 - p$. It therefore follows that the mean and variance of the random variables are p and $p(1 - p)$. For each row and column, the expected cardinality is Np and Mp, respectively. Similarly, the variance for each row and column is $Np(1 - p)$ and $Mp(1 - p)$, respectively. Therefore, we can conclude that no pattern is guaranteed, since the expected value of $Np(1 - p)$ or $Mp(1 - p)$ is the same for every row and column. That is

$$\left|Var(S_{i*}) - Var(S_{j*})\right| < \varepsilon_1, \quad \left|Var(S_{*i}) - Var(S_{*j})\right| < \varepsilon_2, \tag{6.8}$$

for any $\varepsilon_1, \varepsilon_2 > 0$, where $Var(S_{i*})$ and $Var(S_{j*})$ are the variance of the entries of the i-th row and the j-th column, respectively.

Given the above incoherence conditions on L_* and S_*, the recovery is guaranteed by the convex optimization. Our main theorem is based on the nuclear norm based matrix decomposition model. The following theorem holds for the PCP and MCP, and it can easily be generalized for the PEBP and MEBP.

Theorem 27. *Suppose L_* is an $N \times N$ matrix of rank r which obeys incoherence conditions* (6.2) *and* (6.4). *Moreover, entries for all the rows or columns of S_* satisfy* (6.7). *Hence, there exists*

a numerical constant c which ensures that with a probability no less than $1 - cN^{-10}$, the output (L_*, S_*) of the optimization problem

$$\min_{L,S} \|L\|_* + \lambda \|C \circ S\|_1 \tag{6.9}$$

$$\textit{s.t. } M = L + S,$$

$$S_{ij} \in [0, 1], \tag{6.10}$$

$\lambda = \frac{\alpha}{\sqrt{N}}$, $0.0021 < \alpha < 0.0914$, C is a constant matrix as defined in Chapter 5, is exact, provided that

$$\text{rank}(L_*) \leq \tau \sqrt{\frac{N}{\rho_0 \log^2 N}}, \ \tau \geq 1, \quad m < N^2 - n^2,$$

$\rho_0 = \frac{m}{N^2}$, $m < N^2 - n^2$, is a positive constant.

6.3 Proof of the Main Theorem

Based on approximate dual certificates, we prove Theorem 27 via proving Lemmas 6.3.5-6.4.4.

We establish some conditions to prove that (L_*, S_*) is the unique optimal solution to our proposed model. These conditions, expressed in terms of the dual matrix W, are given by Lemma 6.3.6 which is similar to Lemma 2.4 in [33]. However, we have tightened the conditions by using different bounds for our proof. For the PCP and MCP this was possible due to the fact that the conditions $U^\top W = 0$, $UW = 0$ and $\|W\| < 1$ must hold. $U \in \mathbb{R}^{N \times 1}$ has exactly n non-zero entries since $L_* = nUU^\top$; each non-zero element equals to $\frac{1}{\sqrt{n}}$. The satisfaction of $U^\top W = 0$ and $WU = 0$ implies that most elements of W must be zero, and the non-zero elements W_{ij} of W must be very small in magnitude so that $\frac{1}{\sqrt{n}} \times W_{ij} \approx 0$, $i, j \in \{1, 2, ..., N\}$. This results in $\|W\| << 1$. Hence, for a moderate approximation of non-zero W we suggest it satisfies $\|W\| \leq \frac{\alpha}{2}$, where α is defined in (6.1).

For the PEBP and MEBP this was also possible due to the fact that the conditions $U^\top W = 0$, $VW = 0$ and $\|W\| < 1$ must hold. $U \in \mathbb{R}^{N \times 1}$ has exactly n non-zero entries; each non-zero element equals to $\frac{1}{\sqrt{n}}$, and $V \in \mathbb{R}^{M \times 1}$ has exactly m non-zero entries; each non-zero element

equals to $\frac{1}{\sqrt{m}}$. since $L_* = \sqrt{nm}UV^\top$. The satisfaction of $U^\top W = 0$ and $WV = 0$ implies that most elements of W must be zero, and the non-zero elements W_{ij} of W must be very small in magnitude so that $\frac{1}{\sqrt{nm}} \times W_{ij} \approx 0, i \in \{1, 2, ..., N\},\ j \in \{1, 2, ..., M\}$. This results in $\|W\| << 1$. Hence, for a moderate approximation of non-zero W we suggest it satisfies $\|W\| \leq \frac{\alpha}{2}$, where α is defined in (6.1).

The proofs presented here are for the PCP and MCP; however, proofs for the PEBP, and MEBP will follow directly.

6.3.1 Supporting Lemmas and Theorems

Now let us state some lemmas which we will utilize in establishing our main findings.

Lemma 6.3.2 ([35], Lemma 3.2)**.** *For every Q and T, we have*

$$\langle Q, T \rangle \leq \|Q\| \|T\|_*.$$

Moreover, there is a Q which obeys $\|Q\| = 1$ for each T, which achieves the equality.

Definition 6.3.1 ([186], Nets, Definition 5.1)**.** *Assume $\varepsilon > 0$ and (Z, d) is a metric space. In addition, assume that $\mathcal{N}_\varepsilon \subseteq Z$. We say that $\mathcal{N}_\varepsilon$ is an ε-net of Z if each point $z \in Z$ can be approximated to about ε by some $x \in \mathcal{N}_\varepsilon$, i.e., $d(z, x) \leq \varepsilon$.*

Lemma 6.3.3 ([186], A net-based spectral norm computation, Lemma 5.4)**.** *Let M be a symmetric matrix of size $N \times N$. Further, for some $\varepsilon \in [0, 1)$, let $\mathcal{N}_\varepsilon$ be an ε-net of S^{N-1}. Then*

$$\|M\| = \sup_{y \in S^{N-1}} |\langle My, y \rangle| \leq (1 - 2\varepsilon)^{-1} \sup_{y \in \mathcal{N}_\varepsilon} |\langle My, y \rangle|.$$

Theorem 28 ([182], Bernstein Inequality, Theorem 1.6.1)**.** *Assume that $Y_1, ..., Y_N$ are all independent, centred, real random variables, and each is uniformly bounded:*

$$\mathbb{E}\, Y_s = 0 \quad \textit{and} \quad |Y_s| \leq F \quad \textit{for each } s = 1, ..., N.$$

Let $Var(Z)$ denote the variance of Z, where $Z = \sum_{s=1}^{N} Y_s$, and is defined by:

$$Var(Z) = \mathbb{E}\, Z^2 = \sum_{s=1}^{N} \mathbb{E}\, Y_s^2.$$

Then

$$Pr\{|Z| \geq t\} \leq 2 \exp\left(\frac{-t^2/2}{Var(Z) + Ft/3}\right) \quad \textit{for all } t \geq 0.$$

6.3.4 Optimality Conditions

We have established conditions that guarantee the recovery and the uniqueness of the solution. Moreover, we have derived a tight bound of the dual matrix that certifies the optimality conditions of our proposed model. Our sufficient conditions are closely related to those given in the references [6, 7, 18] as they also consider the low rank matrix and its dual. However, our sufficient conditions are stronger in that we do not require a number of additional assumptions such as the number of vertices adjacent to clique vertices (Theorems 2.1 & 2.2 [6], also assumptions 1, 2, and 3 in [7]).

Let us call the linear spaces $\mathcal{R}$ and Ω defined in Chapter 2, and their orthogonal complements $\mathcal{R}^{\perp}$ and $\Omega^{\perp}$, rspectively. Here, $\mathcal{P}_{\mathcal{R}}$ and $\mathcal{P}_{\Omega}$ are the orthogonal projections onto $\mathcal{R}$ and Ω, respectively, and $\mathcal{P}_{\mathcal{R}}^{\perp}$ and $\mathcal{P}_{\Omega}^{\perp}$ are the projections onto the orthogonal complement of $\mathcal{R}$ and Ω, respectively.

Lemma 6.3.5. *Assume that the subspaces $\mathcal{R}$ and Ω have a trivial intersection, $(\mathcal{P}_{\Omega} \cap \mathcal{P}_{\mathcal{R}} = 0)$, and assume that there exists $W \in \mathcal{R}^{\perp}$ and $F \in \Omega^{\perp}$ for which the following conditions hold:*

1. $UU^{\top} + W = \lambda\,[\mathrm{sign}(C \circ S_*) + F]$,
2. $\mathcal{P}_{\mathcal{R}}(X) = UU^{\top} = I$ *and* $\mathcal{P}_{\Omega}(Y) = \mathrm{sign}(C \circ S_*)$, *where* $X \in \partial\|.\|_*$ *and* $Y \in \partial\|.\|_1$ *at* L_* *and* $C \circ S_*$, *respectively.*
3. $\|W\| \leq \alpha$ *and* $\|F\|_{\infty} < 1$, *for* $0.0021 < \alpha < 0.0914$.

Then (L_*, S_*) is a unique solution to problem (6.9).

Proof. Consider any feasible solution $(\hat{L}, \hat{S}) = (L_* + P, S_* - P)$ to (6.9) such that $(\hat{L} - L_*, \hat{S} - S_*) \neq (0, 0)$. It is clear that this feasible solution is a perturbation of the optimal solution (L_*, S_*), and it satisfies the feasibility constraint in (6.9). We show that $G(\hat{L}, \hat{S}) > G(L_*, S_*)$ for non-zero P, where $G(.,.)$ represents the objective function of problem (6.9).

Let $X \in \partial\|.\|_*$ at L_* and $Y \in \partial\|.\|_1$ at $C \circ S_*$, then by the definition of the sub-gradient we have

$$G(\hat{L},\hat{S}) \geq G(L_*,S_*) + \langle X, \hat{L}-L_*\rangle + \lambda\langle Y, \hat{S}-S_*\rangle \tag{6.11}$$

$$= G(L_*,S_*) + \langle X, \hat{L}-L_*\rangle + \lambda\langle Y, \hat{S}-S_*\rangle + \langle UU^\top + W, \hat{L}-L_*\rangle - \langle UU^\top + W, \hat{L}-L_*\rangle + \lambda\langle \operatorname{sign}(C\circ S_*) + F, \hat{S}-S_*\rangle - \lambda\langle \operatorname{sign}(C\circ S_*) + F, \hat{S}-S_*\rangle \tag{6.12}$$

$$= G(L_*,S_*) + \langle X - UU^\top - W, \hat{L}-L_*\rangle + \lambda\langle Y - \operatorname{sign}(C\circ S_*) - F, \hat{S}-S_*\rangle + \langle UU^\top + W, \hat{L}-L_*+\hat{S}-S_*\rangle \tag{6.13, 6.14}$$

$$= G(L_*,S_*) + \langle X - \mathcal{P}_{\mathcal{R}}(X) - W, \hat{L}-L_*\rangle + \lambda\langle Y - \mathcal{P}_{\Omega}(Y) - F, \hat{S}-S_*\rangle \tag{6.15}$$

$$= G(L_*,S_*) + \langle \mathcal{P}^{\perp}_{\mathcal{R}}(X) - W, \hat{L}-L_*\rangle + \lambda\langle \mathcal{P}^{\perp}_{\Omega}(Y) - F, \hat{S}-S_*\rangle \tag{6.16}$$

$$= G(L_*,S_*) + \langle \mathcal{P}^{\perp}_{\mathcal{R}}(X), \hat{L}-L_*\rangle - \langle W, \hat{L}-L_*\rangle + \lambda\langle \mathcal{P}^{\perp}_{\Omega}(Y), \hat{S}-S_*\rangle - \lambda\langle F, \hat{S}-S_*\rangle \tag{6.17}$$

$$= G(L_*,S_*) + \langle X, \mathcal{P}^{\perp}_{\mathcal{R}}(\hat{L}-L_*)\rangle - \langle W, \hat{L}-L_*\rangle + \lambda\langle Y, \mathcal{P}^{\perp}_{\Omega}(\hat{S}-S_*)\rangle - \lambda\langle F, \hat{S}-S_*\rangle \tag{6.18}$$

$$G(\bar{L},\bar{S}) \geq G(L_*,S_*) + \|\mathcal{P}^{\perp}_{\mathcal{R}}(\hat{L}-L_*)\|_* - \|W\|\|\mathcal{P}^{\perp}_{\mathcal{R}}(\hat{L}-L_*)\|_* + \lambda\|\mathcal{P}^{\perp}_{\Omega}(\hat{S}-S_*)\|_1 \tag{6.19}$$

$$- \lambda\|F\|_\infty\|\mathcal{P}^{\perp}_{\Omega}(\hat{S}-S_*)\|_1 \tag{6.20}$$

$$= G(L_*,S_*) + (1-\|W\|)\,\|\mathcal{P}^{\perp}_{\mathcal{R}}(\hat{L}-L_*)\|_* + \lambda\,(1-\|F\|_\infty)\,\|\mathcal{P}^{\perp}_{\Omega}(\hat{S}-S_*)\|_1 \tag{6.21}$$

$$\geq G(L_*,S_*) + (1-\alpha)\,\|\mathcal{P}^{\perp}_{\mathcal{R}}(\hat{L}-L_*)\|_* + \lambda\,(1-\eta)\,\|\mathcal{P}^{\perp}_{\Omega}(\hat{S}-S_*)\|_1 \tag{6.22}$$

$$\geq G(L_*,S_*),\ \textit{since}\ 0<\alpha<\frac{1}{2}\ \textit{and}\ 0<\eta<1, \tag{6.23}$$

where the equality in (6.13) follows by condition 1 of Lemma 6.3.5, the equality in (6.15) follows by fact that both (L_*,S_*) and $(\hat{L},\hat{S})$ satisfies the feasibility constraint, thus, $\langle UU^\top + W, \hat{L} - L_* + \hat{S} - S_*\rangle = 0$, and we make use of condition 2 of Lemma 6.3.5, that is, $\mathcal{P}_{\mathcal{R}}(X) = UU^\top$, and $\mathcal{P}_{\Omega}(Y) = \operatorname{sign}(C\circ S_*)$. The equality in (6.16) follows by $\mathcal{P}_{\mathcal{R}}(X) + \mathcal{P}^{\perp}_{\mathcal{R}}(X) = X$, and $\mathcal{P}_{\Omega}(Y) + \mathcal{P}^{\perp}_{\Omega}(Y) = Y$. The inequality in (6.19) follows by the duality between the spectral norm

and the nuclear norm, $\langle W, \mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*)\rangle \leq \|W\|\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*)\|_*$, and the duality between the infinity norm and the ℓ_1-norm, $\langle F, \mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*)\rangle \leq \|F\|_{\infty}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*)\|_1$. The details can be found in Lemma 3.2 in [35]. We have chosen X and Y, such that $\langle X, \mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*)\rangle = \|\mathcal{P}_{T^{\perp}}(\hat{L} - L_*)\|_*$ and $\langle Y, \mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*)\rangle = \|\mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*)\|_1$. Given $\|W\| \leq \alpha$, and $\|F\|_{\infty} < 1$ one has $G(\hat{L}, \hat{S}) > G(L_*, S_*)$, for $(\hat{L} - L_*, \hat{S} - S_*) \neq (0, 0)$, unless $\left(\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*), P_{\Omega}^{\perp}(\hat{S} - S_*)\right) = (0, 0)$. However, $\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*)\|_* = \|\mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*)\|_1 = 0$ only if $\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L} - L_*) = \mathcal{P}_{\Omega}^{\perp}(\hat{S} - S_*) = 0$ (i.e., $\hat{L} - L_* = \hat{S} - S_* = P \in \Omega \cap \mathcal{R}$) then the injectivity assumption (that $\mathcal{R}$ and Ω have a trivial intersection) forces $(\hat{L} - L_*, \hat{S} - S_*) = (P, P) = (0, 0)$.

Consequently, any minimizer $(\hat{L}, \hat{S})$ with $(\hat{L} - L_*, \hat{S} - S_*) \neq (0, 0)$ must satisfies $G(\hat{L}, \hat{S}) > G(L_*, S_*)$. Thus, (L_*, S_*) is a unique minimizer to problem (6.9). □

According to **Lemma 6.3.5**, for the exact recovery of problem (6.9), it is sufficient to determine an appropriate W, for which:

$$\begin{cases} W \in \mathcal{R}^{\perp}, \\ \|W\| < \alpha, \ 0.0021 < \alpha < 0.0914, \\ \mathcal{P}_{\Omega}(UU^{\top} + W) = \lambda \operatorname{sign}(C \circ S_*), \\ \|\mathcal{P}_{\Omega}^{\perp}(UU^{\top} + W)\|_{\infty} < \lambda. \end{cases}$$

Lemma 6.3.6. *Assume that the subspaces $\mathcal{R}$ and Ω have a trivial intersection with $\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\|_F \leq \frac{1}{2}$ and $\lambda < \alpha$, and assume that there exists $W \in \mathcal{R}^{\perp}$, $F \in \Omega^{\perp}$, and $B \in \Omega$, for which the following conditions hold:*

1. $UU^{\top} + W = \lambda(\operatorname{sign}(C \circ S_*) + F + \mathcal{P}_{\Omega}B)$,
2. $\mathcal{P}_{\mathcal{R}}(X) = UU^{\top}$ *and* $\mathcal{P}_{\Omega}(Y) = \operatorname{sign}(C \circ S_*)$, *where* $X \in \partial\|.\|_*$ *and* $Y \in \partial\|.\|_1$ *at* L_* *and* $C \circ S_*$, *respectively.*
3. $\|W\| \leq \frac{\alpha}{2}$, $\|F\|_{\infty} < \frac{1}{2}$ *and* $\|\mathcal{P}_{\Omega}B\|_F \leq \frac{1}{4}$, *for* $0.0021 < \alpha < 0.0914$.

Then (L_*, S_*) is a unique solution to problem (6.9).

Proof. According to the proof of Lemma 6.3.5, we can easily see

$$G(\hat{L},\hat{S}) \geq G(L_*,S_*) + \left(1-\frac{\alpha}{2}\right)\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*)\|_* + \frac{\lambda}{2}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 - \frac{\lambda}{4}\|\mathcal{P}_{\Omega}(\hat{S}-S_*)\|_F. \tag{6.24}$$

It follows from $\hat{S}-S_* = \mathcal{P}_{\mathcal{R}}(\hat{S}-S_*) + \mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)$ that

$$\begin{aligned}\|\mathcal{P}_{\Omega}(\hat{S}-S_*)\|_F &\leq \|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}(\hat{S}-S_*)\|_F + \|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_F \\ &\leq \frac{1}{2}\|\hat{S}-S_*\|_F + \|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_F \\ &\leq \frac{1}{2}\|\mathcal{P}_{\Omega}(\hat{S}-S_*)\|_F + \frac{1}{2}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_F + \|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_F.\end{aligned}$$

This implies that

$$\frac{1}{2}\|\mathcal{P}_{\Omega}(\hat{S}-S_*)\|_F \leq \frac{1}{2}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_F + \|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_F.$$

Hence, it follows that

$$\begin{aligned}\frac{\lambda}{4}\|\mathcal{P}_{\Omega}(\hat{S}-S_*)\|_F &\leq \frac{\lambda}{4}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_F + \frac{\lambda}{2}\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_F \\ &\leq \frac{\lambda}{4}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 + \frac{\lambda}{2}\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{S}-S_*)\|_* \\ &= \frac{\lambda}{4}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 + \frac{\lambda}{2}\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*)\|_*,\end{aligned}$$

where the last equality follows by the feasibility of $(\hat{L},\hat{S})$ and (L_*,S_*), that is $\hat{L}+\hat{S} = L_*+S_*$. Therefore, from (6.24) we have

$$G(\hat{L},\hat{S}) \geq G(L_*,S_*) + \left(1-\frac{\alpha}{2}\right)\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*)\|_* + \frac{\lambda}{2}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 - \frac{\lambda}{4}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 - \frac{\lambda}{2}\|\mathcal{P}_{R}^{\perp}(\hat{L}-L_*)\|_*$$
$$= G(L_*,S_*) + \left(1-\frac{\alpha+\lambda}{2}\right)\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*)\|_* + \frac{\lambda}{2}\|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1,$$

Given $\|W\| \leq \frac{\alpha}{2}$, and $\|F\|_\infty < \frac{1}{2}$, and we have $\frac{\alpha+\lambda}{2} < 1$ as $0 < \alpha < \frac{1}{2}$ and $\lambda < \alpha$, one has $G(\hat{L},\hat{S}) > G(L_*,S_*)$, for $(\hat{L}-L_*,\hat{S}-S_*) \neq (0,0)$, unless $\left(\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*),\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\right) = (0,0)$. However, $\|\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*)\|_* = \|\mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*)\|_1 = 0$ only if $\mathcal{P}_{\mathcal{R}}^{\perp}(\hat{L}-L_*) = \mathcal{P}_{\Omega}^{\perp}(\hat{S}-S_*) = 0$ (i.e., $\hat{L}-L_* = \hat{S}-S_* = P \in \Omega \cap \mathcal{R}$) then the injectivity assumption (that $\mathcal{R}$ and Ω have a trivial intersection) forces $(\hat{L}-L_*,\hat{S}-S_*) = (P,P) = (0,0)$. Consequently, any minimizer $(\hat{L},\hat{S})$ with $(\hat{L}-L_*,\hat{S}-S_*) \neq (0,0)$ must satisfies $G(\hat{L},\hat{S}) > G(L_*,S_*)$. Thus, (L_*,S_*) is a unique minimizer to problem (6.9). □

According to **Lemma 6.3.6**, for the exact recovery of problem (6.9), it is sufficient to determine an appropriate W, for which:

$$\begin{cases} W \in \mathcal{R}^{\perp}, \\ \|W\| \leq \frac{\alpha}{2},\ 0.0021 < \alpha < 0.0914, \\ \|\mathcal{P}_{\Omega}(UU^{\top}+W-\lambda\,\mathrm{sign}(C\circ S_*))\|_F \leq \frac{\lambda}{4}, \\ \|\mathcal{P}_{\Omega}^{\perp}(UU^{\top}+W)\|_\infty \leq \frac{\lambda}{2}. \end{cases} \tag{6.25}$$

In the following section, we obtain an approximation of the dual certificate in the setting of separating a matrix into its sparse and low-rank components utilizing the golfing scheme and the least-squares technique.

6.4 Construction of Dual Certification by the Golfing Scheme and the Least Squares

A key idea is to construct W so that it satisfies the conditions in (6.25). Assume that entries of S are sampled with probability p according to Bernoulli model. This means that all the matrices

in $\Omega^{\perp} \sim Ber(1-p)$. Assume that all the matrices in $\Omega^{\perp}$ have the same distribution as $\Omega_1 \cup \Omega_2 \cup ... \cup \Omega_K$, where Ω_k's are drown independently with replacement from $Ber(q)$, $K = 20\lceil \log N \rceil$; the parameter q is found empirically. This can be described by Binomial model, $Bin(K, q)$, that is,

$$Pr((i,j) \in \Omega) = Pr(Bin(K,q) = 0) = (1-q)^K.$$

Thus, the two models are equivalent if $p = (1-q)^K$.

The main idea is to disintegrate W into W^S (sparse component) and W^L (low-rank component), that is, $W = W^L + W^S$. Then

$$\begin{aligned} UU^{\top} + W &= UU^{\top} + W^L + W^S \\ &= \mathcal{P}_{\Omega}(UU^{\top} + W^L + W^S) + \mathcal{P}_{\Omega}^{\perp}(UU^{\top} + W^L + W^S) \\ &= \mathcal{P}_{\Omega}(UU^{\top} + W^L) + \lambda \operatorname{sign}(C \circ S_*) + \mathcal{P}_{\Omega}^{\perp}(UU^{\top} + W^L + W^S) \\ &= \lambda \left(\mathcal{P}_{\Omega}\left(\frac{UU^{\top} + W^L}{\lambda} \right) + \operatorname{sign}(C \circ S_*) + \mathcal{P}_{\Omega}^{\perp}\left(\frac{UU^{\top} + W^L + W^S}{\lambda} \right) \right), \end{aligned}$$

where $\mathcal{P}_{\Omega}(W^S) = \lambda \operatorname{sign}(C \circ S_*)$, since L_* and S_* are supported on $\mathcal{R}$ and Ω, respectively. We take

$$\begin{aligned} B &= \left(\frac{UU^{\top} + W^L}{\lambda} \right), \text{ and} \\ F &= \mathcal{P}_{\Omega}^{\perp}\left(\frac{UU^{\top} + W^L + W^S}{\lambda} \right), \end{aligned}$$

and so B and F adhere to the conditions stated in Lemma 6.3.6. W^L and W^S adhering to (6.25) certify that problem (6.9) perfectly recovers, with high probability, the sparse matrix S_* and the low-rank matrix L_*. That is

$$\begin{cases} \|W^L + W^S\| \leq \frac{\alpha}{2}, \ 0.0021 < \alpha < 0.0914, \\ \|\mathcal{P}_{\Omega}(UU^{\top} + W^L)\|_F \leq \frac{\lambda}{4}, \\ \|\mathcal{P}_{\Omega}^{\perp}(UU^{\top} + W^L + W^S)\|_{\infty} \leq \frac{\lambda}{2}. \end{cases} \tag{6.26}$$

We will use the golfing scheme to construct W^L, and the least squares method to construct W^S. The golfing scheme [81] is a tool to construct an approximate dual certificate. In the golfing scheme, an interim solution is improved, iteratively, until the final approximation of the dual certificate is obtained. W^L is constructed as follows:

$$Q_k = Q_{k-1} + q^{-1}\mathcal{P}_{\Omega_k}\mathcal{P}_{\mathcal{R}}(UU^\top - Q_{k-1}),\ Q_0 = 0,\ k = 1, 2, ..., K \tag{6.27}$$
$$W^L = \mathcal{P}_{\mathcal{R}}^{\perp} Q_K$$

According to the least square method [33], W^S is constructed as follows:

$$W^S = \lambda \mathcal{P}_{\mathcal{R}}^{\perp} \left(\mathcal{P}_\Omega - \mathcal{P}_\Omega \mathcal{P}_{\mathcal{R}} \mathcal{P}_\Omega\right)^{-1} \operatorname{sign}(C \circ S_*),$$

using Neumann series [33], W^S can be stated as follows:

$$W^S = \lambda \mathcal{P}_{\mathcal{R}}^{\perp} \sum_{k=0}^{K} \left(\mathcal{P}_\Omega \mathcal{P}_{\mathcal{R}} \mathcal{P}_\Omega\right)^k \operatorname{sign}(C \circ S_*). \tag{6.28}$$

6.4.1 Validity of Certificate of Low-Rank Part

We now declare and verify some sufficient conditions on the approximated dual certificate W^L constructed by the golfing scheme for (L_*, S_*) to be the unique optimal solution for (6.9).

Lemma 6.4.2. *Assume that all the matrices in* $\Omega \sim Ber(p)$, $\|\mathcal{P}_\Omega \mathcal{P}_{\mathcal{R}}\| \leq \frac{1}{2}$, *i.e.,* $\mathcal{P}_\Omega \cap \mathcal{P}_{\mathcal{R}} = 0$, *and* $K = 20\lceil \log N \rceil$. *Then, for* $\lambda = \frac{\alpha}{\sqrt{N}}$, $0.0021 < \alpha < 0.0914$, *the dual matrix* W^L *in* (6.27) *satisfies:*

a. $\|W^L\| < \frac{\alpha}{4}$,

b. $\|\mathcal{P}_\Omega(UU^\top + W^L)\|_F < \frac{\lambda}{4}$,

c. $\|\mathcal{P}_\Omega^{\perp}(UU^\top + W^L)\|_\infty < \frac{\lambda}{4}$.

The proof technique of this Lemma closely follows that of Lemma 2.8 [33], but we have have used different bounds in our proof [33].

Proof. **Proof of a.** Let

$$Y_{k-1} = UU^\top - \mathcal{P}_{\mathcal{R}} Q_{k-1},\ Y_{k-1} \in \mathcal{R}, \tag{6.29}$$

then, $Q_K = \sum_{k=1}^{K} q^{-1} \mathcal{P}_{\Omega_k} \mathcal{P}_{\mathcal{R}} Y_{k-1}$, and $\|W^L\| = \|\mathcal{P}_{\mathcal{R}}^{\perp} Q_K\|$. Note that it has been shown in [33] that, for

$$q \geq c_0 \zeta^{-2} \mu_0 r \log N/N,\ \ 0 < \zeta < 1/2, \tag{6.30}$$

with high probability $\|Y_k\|_\infty \leq \frac{1}{2}\|Y_{k-1}\|_\infty$ and $\|Y_k\|_F \leq \frac{1}{2}\|Y_{k-1}\|_F$ hold, where c_0 is absolute constant.

From the definition of Q_k in (6.27) and using (6.29) we have,

$$\mathcal{P}_{\mathcal{R}}^{\perp} Q_K = \sum_{k=1}^{K} q^{-1} \mathcal{P}_{\mathcal{R}}^{\perp} \mathcal{P}_{\Omega_k} \mathcal{P}_{\mathcal{R}} Y_{k-1}. \tag{6.31}$$

Thus, using the following inequalities

$$\|\mathcal{P}_{\mathcal{R}}^{\perp} \mathcal{P}_{\Omega_k} Y_{k-1}\| \leq \frac{1}{4\sqrt{\text{rank}(Y_{k-1})}} \|Y_{k-1}\|_F,\ \ \|Y_k\|_F \leq \frac{1}{2}\|Y_{k-1}\|_F, \tag{6.32}$$

found in [81], we get:

$$\begin{aligned}
\|W^L\| &= \|\mathcal{P}_{\mathcal{R}}^{\perp} Q_K\| \\
&\leq \sum_{k=1}^{K} q^{-1} \|\mathcal{P}_{\mathcal{R}}^{\perp} \mathcal{P}_{\Omega_k} Y_{k-1}\|,\ \ \textit{by}\ \ (6.31), \\
&\leq q^{-1} \sum_{k=1}^{K} \frac{1}{4\sqrt{a}} \|Y_{k-1}\|_F,\ \textit{by}\ (6.32), \textit{where the rank of}\ Y_{k-1}\ \textit{is}\ a, \\
&\leq \frac{q^{-1}}{4\sqrt{a}} \sum_{k=1}^{K} \left(\frac{1}{2}\right)^{k-1} \|UU^\top\|_F, \textit{using}\ \|Y_{k-1}\|_F \leq \left(\frac{1}{2}\right)^{k-1} \|Y_0\|_F,\ \textit{by}\ (6.32),\ Y_0 = UU^\top,
\end{aligned}$$

thus, it follows that

$$\begin{aligned}
\|W^L\| &\leq \frac{q^{-1}}{4\sqrt{a}} \sum_{k=1}^{K} \left(\frac{1}{2}\right)^{k-1} \|UU^\top\|_F, \textit{using } \|Y_{k-1}\|_F \leq \left(\frac{1}{2}\right)^{k-1} \|Y_0\|_F, \textit{ by (6.32), } Y_0 = UU^\top, \\
&\leq \frac{q^{-1}}{\sqrt{a}} \|UU^\top\|_F, \quad \textit{since } \sum_{k=1}^{K} \left(\frac{1}{2}\right)^{k-1} \leq 4, \textit{ for large } K \\
&\leq \frac{C_0 \zeta^2 N\sqrt{N}}{\mu_0 r \sqrt{a} \log N} \|UU^\top\|_\infty, \textit{ by (6.30), and the fact that } \|.\|_F \leq \sqrt{N}\|.\|_\infty, \; C_0 = \frac{1}{c_0} \\
&= \frac{C_0 \zeta^2 N^{\frac{3}{2}}}{\mu_0 r \sqrt{a} \log N} \frac{1}{n}, \textit{by the joint incoherence condition in (6.4), } \|UU^\top\|_\infty = \frac{1}{n} \\
&< \zeta^2 \alpha, \textit{ by choosing } C_0 < \frac{\alpha n \mu_0 r \sqrt{a} \log N}{N^{\frac{3}{2}}}, \\
&< \frac{\alpha}{4}, \textit{ by choosing } 0 < \zeta < \frac{1}{2}.
\end{aligned}$$

Proof of b. From the definition of Q_k in (6.27), it is clear that Q_K is supported on $\Omega^\perp$, this means that $\mathcal{P}_\Omega Q_K = 0$. Thus, from the definition of W^L we have

$$\mathcal{P}_\Omega \left(UU^\top + W^L\right) = \mathcal{P}_\Omega \left(UU^\top + \mathcal{P}_{\mathcal{R}}^\perp Q_K\right). \tag{6.33}$$

We also know that $\mathcal{P}_\Omega \left(\mathcal{P}_{\mathcal{R}}^\perp Q_K + \mathcal{P}_{\mathcal{R}} Q_K\right) = \mathcal{P}_\Omega(Q_K) = 0$, this means that

$$\mathcal{P}_\Omega(\mathcal{P}_{\mathcal{R}}^\perp Q_K) = -\mathcal{P}_\Omega(\mathcal{P}_{\mathcal{R}} Q_K),$$

and thus, substituting this in (6.33) and using (6.29), we get

$$\mathcal{P}_\Omega \left(UU^\top + W^L\right) = \mathcal{P}_\Omega \left(UU^\top - \mathcal{P}_{\mathcal{R}} Q_K\right) = \mathcal{P}_\Omega(Y_K).$$

Therefore,

$$\|\mathcal{P}_\Omega\left(UU^\top + W^L\right)\|_F = \|\mathcal{P}_\Omega Y_K\|_F \tag{6.34}$$

$$\leq \|Y_K\|_F, \text{ since } \|\mathcal{P}_\Omega Y_K\|_F \leq \|Y_K\|_F,$$

$$\leq \left(\frac{1}{2}\right)^K \|UU^\top\|_F, \ \|Y_K\|_F \leq \left(\frac{1}{2}\right)^K \|Y_0\|_F, \ Y_0 = UU^\top,$$

$$\leq \left(\frac{1}{2}\right)^K \sqrt{N}\|UU^\top\|, \text{ using the fact that } \|.\|_F \leq \sqrt{N}\|.\|,$$

$$= \left(\frac{1}{2}\right)^K \sqrt{N}, \text{ as } \|UU^\top\| = 1,$$

$$= \frac{N}{\alpha\ 2^K}\frac{\alpha}{\sqrt{N}} < \frac{\lambda}{8} < \frac{\lambda}{4}, \text{ where } \lambda = \frac{\alpha}{\sqrt{N}}, \tag{6.35}$$

by choosing K large enough such that $\frac{N}{\alpha\ 2^K} < \frac{1}{4}$, e.g., $K = 20\log N$.

Proof of c. We have

$$UU^\top + W^L = UU^\top + \mathcal{P}_\mathcal{R}^\perp Q_K, \text{ by (6.27)},$$

$$= UU^\top - \mathcal{P}_\mathcal{R} Q_K + Q_K$$

$$= Y_K + Q_K. \tag{6.36}$$

Thus,

$$\|\mathcal{P}_\Omega^\perp\left(UU^\top + W^L\right)\|_\infty = \|\mathcal{P}_\Omega^\perp\left(Y_K + Q_K\right)\|_\infty, \text{ by (6.36)}$$

$$\leq \|Y_K\|_\infty + \|Q_K\|_\infty$$

$$\leq \|Y_K\|_F + \|Q_K\|_\infty, \ \|Y_K\|_\infty \leq \|Y_K\|_F$$

$$\leq \frac{\lambda}{8} + q^{-1}\sum_{k=1}^K \|\mathcal{P}_\Omega\mathcal{P}_\mathcal{R} Y_{k-1}\|_\infty$$

$$\leq \frac{\lambda}{8} + q^{-1}\frac{1}{2}\sum_{k=1}^K \|Y_{k-1}\|_\infty, \text{ as } \|\mathcal{P}_\Omega\mathcal{P}_\mathcal{R}\| \leq \frac{1}{2},$$

$$\leq \frac{\lambda}{8} + q^{-1}\frac{1}{2}\sum_{k=1}^K (\frac{1}{2})^{k-1}\|UU^\top\|_\infty, \ \|Y_{k-1}\|_\infty \leq \left(\frac{1}{2}\right)^{k-1}\|Y_0\|,$$

thus, it follows that

$$
\begin{aligned}
\|\mathcal{P}_{\Omega}^{\perp}\left(UU^{\top}+W^{L}\right)\|_{\infty} &\leq \frac{\lambda}{8}+q^{-1}\frac{1}{2}\sum_{k=1}^{K}(\frac{1}{2})^{k-1}\|UU^{\top}\|_{\infty},\ \|Y_{k-1}\|_{\infty}\leq\left(\frac{1}{2}\right)^{k-1}\|Y_0\| \\
&= \frac{\lambda}{8}+2q^{-1}\|UU^{\top}\|_{\infty},\ \sum_{k=1}^{K}\left(\frac{1}{2}\right)^{k-1}\leq 4,\ \textit{for large } K,\ Y_0=UU^{\top}, \\
&\leq \frac{\lambda}{8}+2\frac{C_0\zeta^2 N}{\mu_0 r\log N}\frac{1}{n},\ \textit{by (6.4)},\ \|UU^{\top}\|_{\infty}=\frac{1}{n},\ q\ \textit{from (6.30)},\ C_0=\frac{1}{c_0}, \\
&= \frac{\lambda}{8}+4\frac{C_0\frac{\zeta^2}{2}N}{n\mu_0 r\log N}\left(\frac{\sqrt{N}}{\alpha}\right)\left(\frac{\alpha}{\sqrt{N}}\right) \\
&< \frac{\lambda}{8}+\frac{\lambda}{8},\ \textit{by choosing } 0<\zeta<\frac{1}{2},\ \textit{thus, } \frac{\zeta^2}{2}\frac{\alpha}{\sqrt{N}}<\frac{\lambda}{8},
\end{aligned}
$$

and by choosing $C_0 < \frac{\alpha n\mu_0 r\log^2 N}{4N^{3/2}}$. In the above derivation, the third inequality follows by the proof of b, i.e., the first inequality in (6.34) and (6.35), along with (6.27) and (6.29). □

6.4.3 Validity of Certificate of Sparse Part

We now declare and verify some sufficient conditions on the approximated dual certificate W^S in Lemma 6.4.4. The following Lemma is somewhat similar to Lemma 2.9 in [33], however, we have used the Bernstein's inequality, that provided a tighter bound than the one used in [33].

Lemma 6.4.4 ([33]). *Assume that S_* is supported on Ω, and $\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\|<\gamma$, γ very small absolute number, with high probability. Then for $\lambda=\frac{\alpha}{\sqrt{N}}$, $0.0021<\alpha<0.0914$, the dual matrix W^S in (6.28) satisfies:*

a. $\|W^S\|<\frac{\alpha}{4}$,

b. $\|\mathcal{P}_{\Omega}^{\perp}(W^S)\|_{\infty}<\frac{\lambda}{4}$.

Proof. We consider the random variable $\delta_{ij}=\operatorname{sign}((C\circ S_*)_{ij})$, such that:

$$
\delta_{ij}=\begin{cases}1, & \textit{with propability } p,\\ 0, & \textit{with propability } 1-p.\end{cases}
$$

Proof of a.

W^S can be separated into two terms, using $k = 0$ and $k \geq 1$ in (6.28):

$$W^S = \lambda \mathcal{P}_{\mathcal{R}}^{\perp} \operatorname{sign}(C \circ S_*) + \lambda \mathcal{P}_{\mathcal{R}}^{\perp} \sum_{k=1}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*). \tag{6.37}$$

The main idea of this prove is to determine an upper bound of $\|W^S\|$. It follows that

$$\|\lambda \mathcal{P}_{\mathcal{R}}^{\perp} \operatorname{sign}(C \circ S_*)\| \leq \lambda \|\operatorname{sign}(C \circ S_*)\| \leq c\lambda\sqrt{Np} = c\alpha\sqrt{p} \leq \frac{\alpha}{8}, \tag{6.38}$$

for small absolute constant c, where we have used the fact that in every dimension N, $\|\operatorname{sign}(C \circ S_*)\| \leq c\sqrt{Np}$ [186], and we have $\lambda = \frac{\alpha}{\sqrt{N}}$.

We now consider the spectral norm of the second term of W^S in (6.37). Define $\mathcal{H} = \sum_{k=1}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k$ as an operator, then we can write $\mathcal{P}_{\mathcal{R}}^{\perp} \sum_{k=1}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*)$ as $\mathcal{P}_{\mathcal{R}}^{\perp}\mathcal{H}\left(\operatorname{sign}(C \circ S_*)\right)$ and thus show that this term is bounded above by small absolute constant with high probability.

Indicate by N the ϵ-net of $\mathcal{S}^{N-1}$ of size at most 6^N, $\epsilon > 0$. According to [186], Lemma 5.4 (A net-based spectral norm computation), we have

$$\begin{aligned}
\|\mathcal{P}_{\mathcal{R}}^{\perp}\mathcal{H}(\operatorname{sign}(C \circ S_*))\| &\leq \|\mathcal{H}(\operatorname{sign}(C \circ S_*))\| \\
&= \sup_{x,y \in \mathcal{S}^{N-1}} \langle \mathcal{H}(yx^{\top}), \operatorname{sign}(C \circ S_*) \rangle \\
&\leq (1 - 2\epsilon)^{-1} \sup_{x,y \in N} \langle \mathcal{H}(yx^{\top}), \operatorname{sign}(C \circ S_*) \rangle \\
&= 4 \sup_{x,y \in N} \langle \mathcal{H}(yx^{\top}), \operatorname{sign}(C \circ S_*) \rangle, \ \textit{using } \epsilon = \frac{3}{8}.
\end{aligned}$$

Define the random variable $Z(x, y) = \langle \mathcal{H}(yx^{\top}), \operatorname{sign}(C \circ S_*) \rangle$, then by Matrix Bernstein's inequality [182], for unit-normed vectors x and y, that is, $\|x\| = \|y\| = 1$, with mean being zero, and variance $Var(Z(x, y)) = \frac{1}{N^2} \sum_N \sum_N Z^2(x, y) = \frac{1}{N^2} \|Z(x, y)\|_F^2$, we have,

$$Pr(\|Z(x,y)\| > a \mid \Omega) \leq 2N \exp\left(\frac{-a^2/2}{\frac{1}{N^2}\|Z(x,y)\|_F^2 + a/3}\right),$$

where Ω is the support of matrix (δ_{ij}). Since x and y are unit-normed vectors, $\|yx^\top\|_F = 1$, $\|\mathcal{H}(yx^\top)\|_F \leq \|\mathcal{H}\|$ thus,

$$\begin{aligned} Pr(\|Z(x,y)\| > a \mid \Omega) &\leq 2N \exp\left(\frac{-a^2/2}{\frac{1}{N^2}\|\mathcal{H}(yx^\top)\|_F^2 + a/3}\right) \\ &\leq 2N \exp\left(\frac{-a^2/2}{\frac{1}{N^2}\|\mathcal{H}\|^2 + a/3}\right). \end{aligned}$$

Therefore, we have

$$\begin{aligned} Pr(\lambda\|\mathcal{H}(C \circ \operatorname{sign}(S_*))\| > a \mid \Omega) &\leq 2N \exp\left(\frac{-(a/4\lambda)^2/2}{\frac{1}{N^2}\|\mathcal{H}\|^2 + (a/4\lambda)/3}\right) \\ &= 2N \exp\left(\frac{-(a/\lambda)^2/32}{\frac{1}{N^2}\|\mathcal{H}\|^2 + a/12\lambda}\right). \end{aligned}$$

Assume that $\|\mathcal{P}_\Omega \mathcal{P}_\mathcal{R}\| \leq \gamma$ with high probability, for a very small absolute constant γ, we have

$$\|\mathcal{H}\| = \Big\| \sum_{k=1}^{K} \left(\mathcal{P}_\Omega \mathcal{P}_\mathcal{R} \mathcal{P}_\Omega\right)^k \Big\| \tag{6.39}$$

$$\leq \sum_{k=1}^{K} \Big\| \left(\mathcal{P}_\Omega \mathcal{P}_\mathcal{R} \mathcal{P}_\Omega\right)^k \Big\| \tag{6.40}$$

$$\leq \sum_{k=1}^{K} \gamma^{2k} = \frac{\gamma^2}{1-\gamma^2}.$$

Thus, unconditionally,

$$Pr(\lambda\|\mathcal{H}(\operatorname{sign}(S_*))\| > a) \leq 2N \exp(D)\, Pr(\|\mathcal{P}_\Omega \mathcal{P}_\mathcal{R}\| \leq \gamma) + Pr\left(\|\mathcal{P}_\Omega \mathcal{P}_\mathcal{R}\| > \gamma\right) < \frac{\alpha}{8},$$

where $\exp(D)$, with $D = \frac{-(a/\lambda)^2/32}{\frac{1}{N^2}\left(\frac{\gamma^2}{1-\gamma^2}\right)^2 + a/12\lambda}$, is very small number, $\lambda = \frac{\alpha}{\sqrt{N}}$, and we put $a = \frac{\alpha}{8}$. This together with the bound of the first term of $\|W^S\|$ in (6.38) completes the proof.

Proof b. We know that $\mathcal{P}_\Omega W^S + \mathcal{P}_\Omega^\perp W^S = W^S$. Recalling W^S, we have

$$
\begin{aligned}
\mathcal{P}_{\Omega}^{\perp} W^S &= W^S - \mathcal{P}_{\Omega} W^S \\
&= \lambda \mathcal{P}_{\mathcal{R}}^{\perp} (\mathcal{I} - \mathcal{P}_{\Omega}) \sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*), \ \textit{using (6.28)} \\
&= \lambda \mathcal{P}_{\Omega}^{\perp} (\mathcal{I} - \mathcal{P}_{\mathcal{R}}) \sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*) \\
&= -\lambda \mathcal{P}_{\Omega}^{\perp} \mathcal{P}_{\mathcal{R}} \sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*), \qquad (6.41)
\end{aligned}
$$

where $\mathcal{I}$ represents the identity operator and the last equality follows since $\operatorname{sign}(C \circ S_*)$ is supported on $\mathcal{P}_{\Omega}$. The idea here is to express $\|\mathcal{P}_{\Omega}^{\perp} W^S\|_{\infty}$ in the form of $\langle H, \operatorname{sign}(C \circ S_*)\rangle$, then derive an upper bound on it, given $\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| \leq \gamma$ (where γ is a very small constant).

For any indices (i, j) of $S \in \Omega^{\perp}$, and noting that $\mathcal{P}_{\Omega}$ and $\mathcal{P}_{\mathcal{R}}$ are self ad-joint, thus

$$
\begin{aligned}
W_{ij}^S &= e_i^{\top} W^S e_j = \langle e_i e_j^{\top}, W^S \rangle \\
&= \lambda \langle e_i e_j^{\top}, -\mathcal{P}_{\Omega}^{\perp} \mathcal{P}_{\mathcal{R}} \sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*) \rangle, \ \textit{using (6.41)}, \\
&= \lambda \langle -\mathcal{P}_{\Omega}^{\perp} \mathcal{P}_{\mathcal{R}} \left(e_i e_j^{\top}\right), \sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \operatorname{sign}(C \circ S_*) \rangle \\
&= \langle -\sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \mathcal{P}_{\Omega}^{\perp} \mathcal{P}_{\mathcal{R}} \left(e_i e_j^{\top}\right), \lambda \operatorname{sign}(C \circ S_*) \rangle. \qquad (6.42)
\end{aligned}
$$

Define $Z(i, j) = -\sum_{k=0}^{K} \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \mathcal{P}_{\Omega}^{\perp} \mathcal{P}_{\mathcal{R}} \left(e_i e_j^{\top}\right)$, thus using the union bound

$$
\begin{aligned}
Pr\left(\|\mathcal{P}_{\Omega}^{\perp}(W^S)\| > a\lambda \mid \Omega\right) &\leq \sum_{i,j} Pr\left(|e_i^{\top} W^S e_j| > a\lambda \mid \Omega\right) \\
&\leq N^2 Pr\left(|e_i^{\top} W^S e_j| > a\lambda \mid \Omega\right). \qquad (6.43)
\end{aligned}
$$

Thus, using the matrix Bernstein's inequality, we have

$$\begin{aligned}
\Pr\left(\|\mathcal{P}_{\Omega}^{\perp}(W^S)\|_{\infty} > a\lambda \mid \Omega\right) &\leq \Pr\left(\sqrt{N}\|\mathcal{P}_{\Omega}^{\perp}(W^S)\| > a\lambda \mid \Omega\right), \ \textit{as } \|.\|_{\infty} \leq \|.\|, \\
&\leq N^{5/2}\Pr\left(|e_i^{\top} W^S e_j| > a\lambda \mid \Omega\right), \ \textit{using (6.43)}, \\
&\leq N^{5/2}\Pr\left(|\langle Z(i,j), \operatorname{sign}(S_*)\rangle| > a \mid \Omega\right), \ \textit{by (6.42)}, \\
&\leq 2N^{5/2}\exp\left(\frac{-a^2/2}{\frac{1}{N^2}\|Z(i,j)\|_F^2 + a/3}\right),
\end{aligned}$$

where the last inequality follows by Bernstein's inequality. Now for any indices (i,j) of $S \in \Omega^{\perp}$, assume that $\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| \leq \gamma$, γ small absolute number, then $\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\| \leq \gamma^2$, thus we have

$$\begin{aligned}
\|Z(i,j)\|_F &= \sum_{k=0}^{K} \| \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \mathcal{P}_{\Omega}^{\perp}\mathcal{P}_{\mathcal{R}}(e_i e_j^{\top})\| \\
&\leq \sum_{k=0}^{K} \| \left(\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\mathcal{P}_{\Omega}\right)^k \| \|\mathcal{P}_{\Omega}^{\perp}\mathcal{P}_{\mathcal{R}}(e_i e_j^{\top})\| \\
&\leq \frac{\gamma^2}{1-\gamma^2}\|\mathcal{P}_{\mathcal{R}}(e_i e_j^{\top})\|_F, \ \textit{since } \|\mathcal{P}_{\Omega}^{\perp}\mathcal{P}_{\mathcal{R}}(e_i e_j^{\top})\| \leq \|\mathcal{P}_{\mathcal{R}}(e_i e_j^{\top})\| \\
&\leq \frac{\gamma^2}{1-\gamma^2}\sqrt{1 - \|\mathcal{P}_{\mathcal{R}}^{\perp}(e_i e_j^{\top})\|_F^2} \\
&\leq \frac{\gamma^2}{1-\gamma^2}\sqrt{1 - \|(I - \tilde{U}\tilde{U}^{\top})e_i\|^2\|(I - \tilde{V}\tilde{V}^{\top})e_j\|^2}, \ e_i e_j^{\top} \textit{ has SVD } \tilde{U}\Sigma\tilde{V}^{\top}. \\
&= \frac{\gamma^2}{1-\gamma^2}, \ \tilde{U}\tilde{U}^{\top} = I, \ \tilde{V}\tilde{V}^{\top} = I,
\end{aligned}$$

where the third inequality follows due to the fact that $\|\mathcal{P}_{\mathcal{R}}(e_i e_j)\|_F^2 + \|\mathcal{P}_{\mathcal{R}}^{\perp}(e_i e_j)\|_F^2 = 1$, the fourth inequality follows by the definition of the orthogonal complement projection onto $\mathcal{R}$. Thus, unconditionally,

$$
\begin{aligned}
Pr\left(\|\mathcal{P}_{\Omega}^{\perp}(W^S)\|_{\infty} > a\lambda\right) &\leq 2N^{5/2}\exp\left(\frac{-a^2/2}{\frac{1}{N^2}\|Z(i,j)\|_F^2 + a/3}\right) Pr(\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| \leq \gamma) \\
&\qquad + Pr(\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| > \gamma) \\
&\leq 2N^{5/2}\exp\left(G\right) Pr(\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| \leq \gamma) + Pr(\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| > \gamma) \\
&< \frac{\lambda}{4},
\end{aligned}
$$

where $\exp\left(G\right)$, with $G = \frac{-a^2/2}{\frac{1}{N^2}\left(\frac{\gamma^2}{1-\gamma^2}\right)^2 + a/3}$, is a very small number, $\Pr\left(\|\mathcal{P}_{\Omega}\mathcal{P}_{\mathcal{R}}\| \leq \gamma\right)$ with high probability and using $a = \frac{\lambda}{4}$, and $\lambda = \frac{\alpha}{\sqrt{N}}$. □

The fact that W^L and W^S adhere to Lemma 6.3.6 and Lemma 6.4.2, respectively, certifies that, with high probability, problem (6.9) correctly recovers L_* and S_*.

Chapter 7

Numerical Results

7.1 Introduction

Throughout this section, we evaluate the performance of our proposed algorithms by applying them to several problems considered in this thesis. These problems include identifying the planted cliques in given graphs, referred to as the Planted Clique Problem (PCP), and the problem of finding Maximum Cliques (MCs) in random graphs (where no cliques are planted), referred to as the Maximum Clique Problem (MCP) in random graphs. We also consider the problem of identifying cliques in real-world graphs. Finally, we consider the Planted Edge Bi-clique Problem (PEBP), and the Maximum Edge Bi-clique Problem (MEBP) in random graphs. All results are computed in Matlab 2021b, using a standard desktop computer with an Intel Core i7, 3.60GHz CPU, and 16 GB RAM.

7.2 The MC Problem via Nuclear Norm

In this section, we evaluate the performance of our NN-based (Nuclear Norm) matrix decomposition model for the MC problem as we consider three different problems: PCP, MCP, and the problem of finding MCs in real world graphs.

7.2.1 Planted Cliques via NN

Let V^* denote the planted clique of size n. Let M represent the adjacency matrix of the graph $G = (V, E)$, $|V| = N$. We set $M_{ij} = 1$ for $(i, j) \in V^* \times V^*$, $M_{ii} = 1$ for all i; we add an edge (i, j) with probability p for all $(i, j) \in (V \times V) \backslash (V^* \times V^*)$ such that $M_{ji} = M_{ij}$. **Algorithm** 5 solves all problems tested with $p \in [\frac{1}{2}, 0.85]$ and achieves very similar accuracies for all p values. However, we report here the results obtained for $p = \frac{1}{2}$, for making a fair comparison with other algorithms in the literature.

We have used $N = 200,\ 500$ and 1000. For a fixed value of N, we have used $n = 10, 20, ..., N-10$. Hence the number of problems considered for $N = 200,\ 500$, and 1000, are $19,\ 49$, and 99 respectively. Each of these problem is generated 15 times and hence the total number of test runs was 2505.

We have initialized **Algorithm** 5 with a randomly generated feasible S of zeros with probability $p = 0.75$ and ones with probability $1 - p = 0.25$. Then we initialize the feasible L as $L = M - S$, $C = (C_{ij}) = \left(\frac{\epsilon}{(S_{ij}+\epsilon)^2}\right)$, and we set $q = 1$ (see section 5.3.1, page 74, for the definition of q). We would like to note here that the infeasible initialization such as $(L, S) = (0, 0)$ equally produces similar final results. The parameters involved are δ, and λ, where $\delta = 0.0001$ is used as the tolerance for stopping the algorithm. We have used a constant λ throughout our numerical testing. Our numerical investigations suggest that **Algorithm** 5 produces almost insensitive results for $\lambda = \frac{\alpha}{\sqrt{N}}$ for any $\alpha \in [0.0021, 0.0914]$. We have estimated the range, $[l, u]$, for α as follows. First we calculate three ranges $[l_i, u_i]$, $i = 1, 2, 3$, corresponding to $c = 0.25, 0.5$ *and* 0.75, respectively in $m = \frac{1}{2}(N^2 - n^2)$, $n = cN$. We plot α $(= N/m)$ against N for each c value and obtained $[l_i, u_i]$ for α. We then take $l = \min l_i$ and $u = \max u_i$, $i = 1, 2, 3$. We have used $\alpha = 0.054$ for all (N, n) pairs for the results presented here. The suitable values of ϵ in (5.8) lie in $[0.05, 0.42]$. For the results presented here we have used $\epsilon = 0.05$. We have used augmented Lagrangian parameter in (5.21) as $\rho = \frac{1}{mean(M)}$, where $mean(M)$ is the mean value of entries of M. The regular version (5.5)-(5.7) has been also implemented with these parameter values.

The final solution of ADMM algorithm for the regular model is denoted as (L^1, S^1) while the final solution of the proposed model (5.12)-(5.13) is denoted as (L^2, S^2).

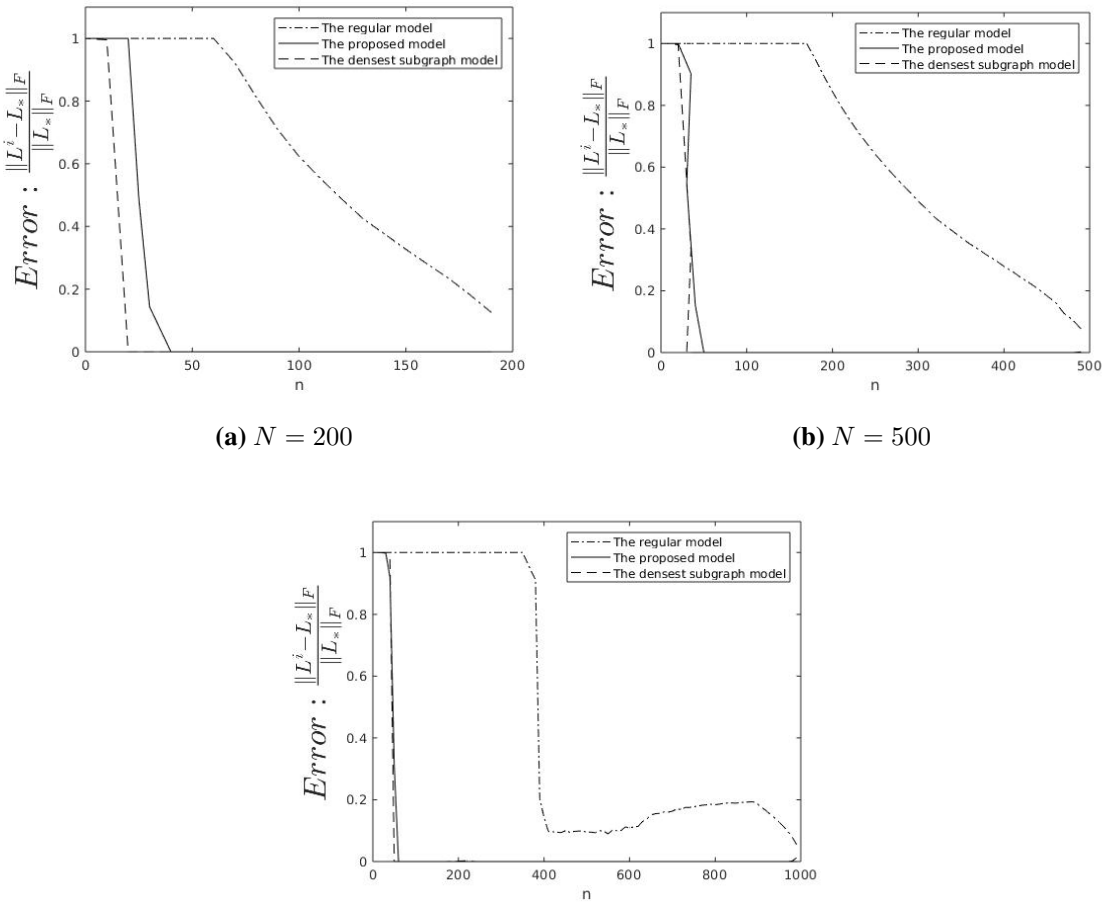
We use the Frobenius norm to calculate the relative error $errL^i$ for each algorithm,

$$errL^i = \frac{\|L^i - L_*\|_F}{\|L_*\|_F},\ i = 1, 2, \tag{7.1}$$

where L_* corresponds to V^*, the planted clique.

We terminate **Algorithm** 5 when (5.28) holds. We have compared **Algorithm** 5 with the the densest subgraph algorithm (DSA) [6, 18], for all the problems considered in this section.

First we compare the average errors in Figure 7.1, where the y-axis indicates the average of relative errors in (7.1); the average is taken over 15 runs on each problem. The value n in the x-axis denotes the size of the planted clique.



(a) $N = 200$

(b) $N = 500$

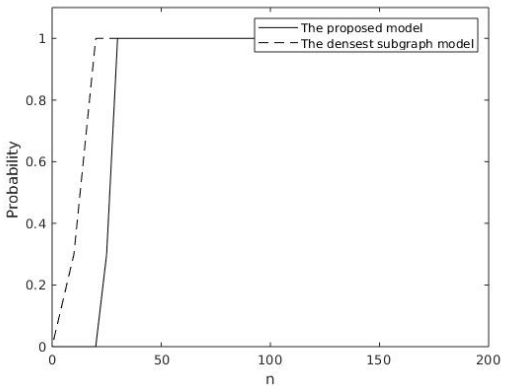
(c) $N = 1000$

Figure 7.1: Average error for all problems.

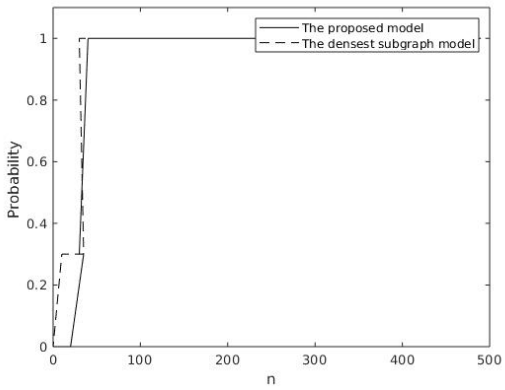
Figure 7.1 shows errors for the ADMM algorithm for the regular model (5.5)-(5.7) are worse than the proposed model (5.12)-(5.13) for all (N, n) pairs. It also shows that the error for the ADMM algorithm for the regular model does improve for higher values of n, i.e., for the problems that are easier to solve. On the other hand, our proposed model (5.12)-(5.13) achieves errors less

than 10^{-8} for all $n \geq 30$, $n \geq 50$, and $n \geq 60$ for $N = 200,\ 500$, and 1000, respectively. However, the errors produced by DSA are about 10^{-5}; it fails to produce less error than 10^{-5}.

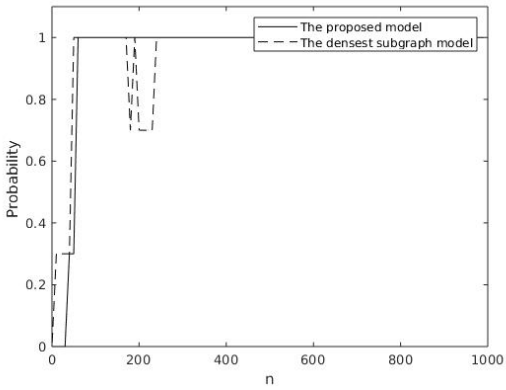
Next, we compare our algorithm for the model in (5.12)-(5.13) with the DSA using the probability of recovery for all problems corresponding to all (N, n) pairs. This comparison has been summarized in Figure 7.2. Here, by recovery we mean that the obtained solution has average error less than 10^{-8} for **Algorithm** 5 and about 10^{-5} for the DSA [6, 18].



(a) $N = 200$



(b) $N = 500$



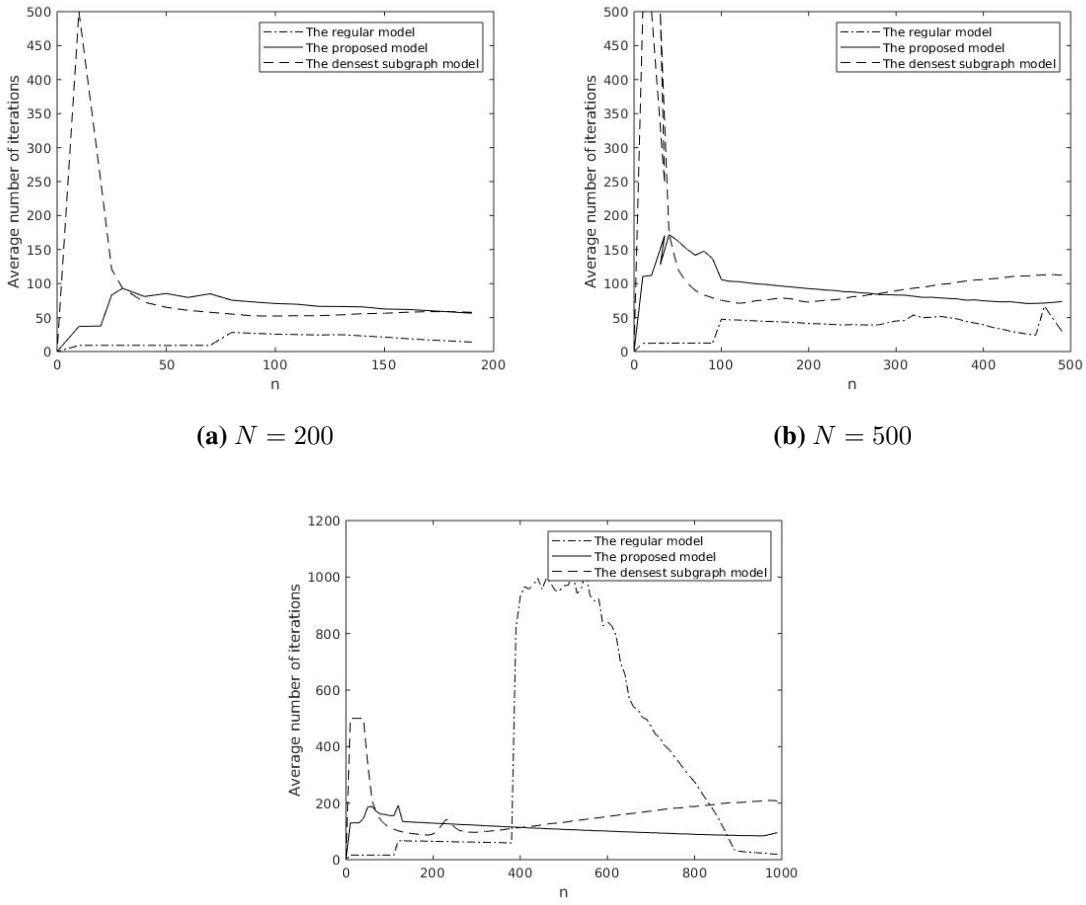
(c) $N = 1000$

Figure 7.2: Average recovery probability for all problems.

Figure 7.2 shows that the probability equals one almost for all (N, n) pairs using **Algorithm** 5. Figure 7.2 also shows that DSA has not provided perfect recovery for all the problems considered.

For example, for $N = 1000$, DSA has recovered some cliques of sizes around $n = 200$ with probability less than 1 for a number of problems.

In Figure 7.3 we present the average number of iterations needed by Algorithm 5 for producing average error of 10^{-8} and DSA for producing average error of 10^{-5}.



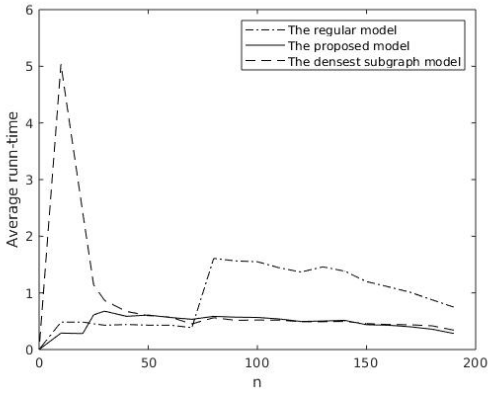
(a) $N = 200$ **(b)** $N = 500$

(c) $N = 1000$

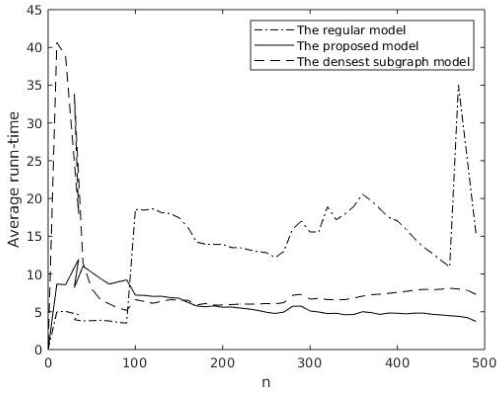
Figure 7.3: Average number of iterations for all problems.

Figure 7.3 demonstrates that our proposed algorithm for model (5.12)-(5.13) requires less number of iterations, on average, than the DSA to converge.

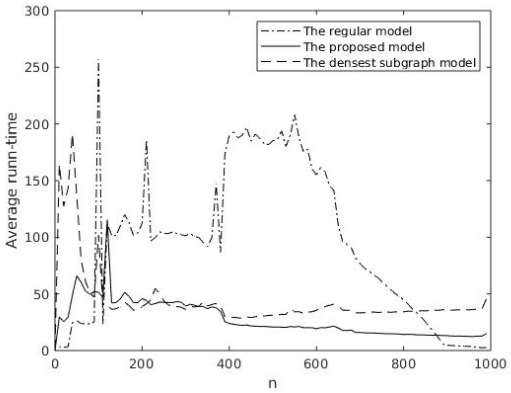
To give an impression of the runtime needed by our algorithm and DSA, we have summarized the average of the total runtime and average runtime per iterations in the following figures.



(a) $N = 200$



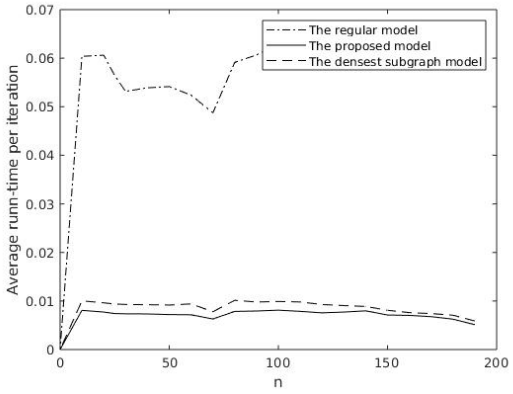
(b) $N = 500$



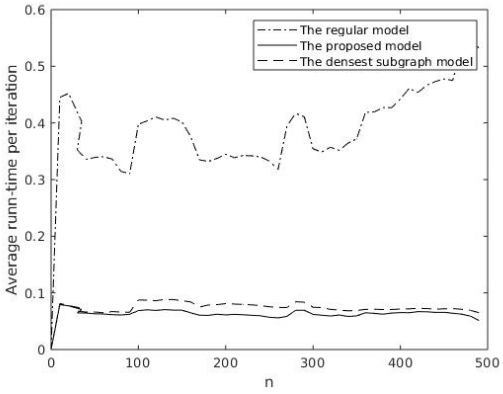
(c) $N = 1000$

Figure 7.4: Average runtime for all problems.

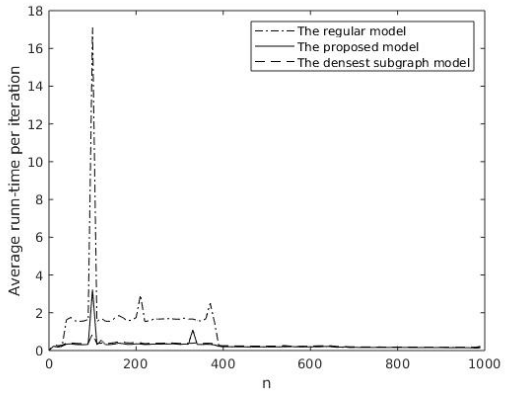
Figure 7.4 shows the average runtime for all (N, n) pairs for the problems considered, the averages are taken over 15 runs on each problem. Figure 7.4 shows that our algorithm performs better than DSA in finding the optimal solution. Figure 7.5 shows the average runtime per iteration for the all the problems considered, where our algorithm performs slightly better.



(a) $N = 200$



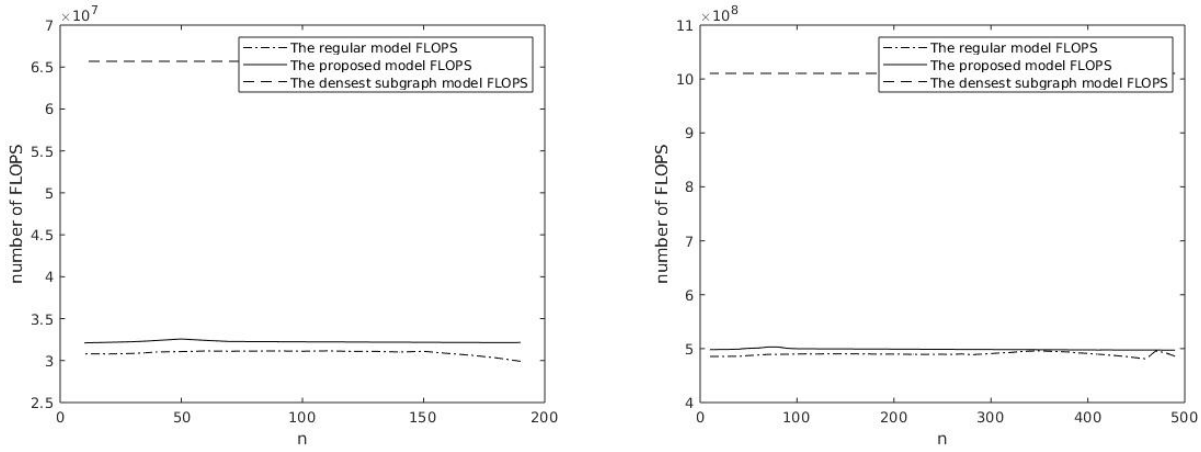
(b) $N = 500$



(c) $N = 1000$

Figure 7.5: Average runtime per iteration for all problems.

To clarify the scaling of our proposed approach, we sketch the number of FLOPS (Floating Point Operations per Second) needed per iteration. Figure 7.6 shows the average number of FLOPS needed per iteration for $N = 200$ and $N = 500$.



(a) $N = 200$ **(b)** $N = 500$

Figure 7.6: Average number of FLOPS per iteration.

Figure 7.6 shows that the number of FLOPS needed per iteration is $\mathcal{O}(N^3)$. It also shows that our algorithm needed less number of FLOPS than the densest subgraph algorithm.

Finally, we also compare our proposed model for the planted clique problem with two further known mathematical models. These are the NNM model (Nuclear Norm Minimization) and the model based on semi-definite programming (SDP) [5]. NNM was solved using PPAPack, a software package in Matlab. NNM failed to obtain optimal solutions with the desired error tolerance better than 10^{-2} where the errors were found using (7.1) [5]. On the other hand, SDP solver failed to provide optimal planted clique of any size when $N \geq 100$. In addition, Ames [5] reported that final solutions of NNM had to be obtained by rounding the entries of solution matrix provided by the software, PPAPack, used. This is not the case for the optimal solutions obtained by our proposed algorithm, as we have claimed earlier in Chapter 5. Furthermore, the results shown in [5] show that all solutions were obtained with an error tolerance of 10^{-2} which is much inferior to our error tolerance of 10^{-8}. Our proposed approach has probability 1 for all tested (N, n) pairs, with $n \in [30, 190]$, $n \in [50, 490]$, and $n \in [60, 990]$ for $N = 200,\ 500$, and 1000, respectively. On the other hand NNM does not achieve probability 1 for all (N, n) pairs with $n \in [110, 140]$,

$n \in [200, 250]$, and $n \in [200, 400]$ for $N = 200,\ 500$, and 1000, respectively. Clearly, our algorithm solves harder problems than NNM and SDP.

7.2.2 The MCP in Random Graphs via NN

We have also performed experiments on random graphs where all the edges are assigned with probability p, with no clique being planted. These results are shown in Table 7.1 where n is the size of the maximum clique obtained by our algorithm. We have used the same stopping condition (5.28) to stop the algorithm, but calculated the errors using the formula

$$Error = \left| \sqrt{\sum_{j=1}^{N} \sum_{i=1}^{N} (L_*)_{ij}} - \left\| L_* \right\| \right|, \tag{7.2}$$

since we have shown earlier in Chapter 6 that $\|L_*\| = n$, where L_* is the adjacency matrix of the low-rank (corresponds to the clique) matrix obtained by our algorithm. Our suggested formula (7.2) measures the recovery of maximum clique based on nodes in the clique, $\sqrt{\sum_{j=1}^{N} \sum_{i=1}^{N} (L_*)_{ij}}$.

The solution (L, S) is initialized with $(0, 0)$. A total of 32 runs have been performed for each N, and the results presented in each row of Table 7.1 are obtained for a single run. We have used $p = 0.8$ and $p = 0.87$ for $N = 200$ and $N = 500$ respectively. We have used higher values for p to ascertain that a reasonable size of maximum clique is formed in each random graph. Results in Table 7.1 show the perfect recovery of maximum cliques based on nodes in the clique except for a small number of cases having some errors. These errors occur because the generated random graph has a bi-clique of size greater than the size of the clique.

$N = 200$					
n	Error	Runtime	n	Error	Runtime
59	0	2.02	46	0	1.97
49	0	1.63	53	0	1.85
74	0	1.99	57	0	2.09
59	0	1.88	43	0	1.75
48	0	1.86	48	0	1.96
55	0	1.86	51	0	1.89
48	0	1.88	63	0	2.12
54	0	1.82	58	0	1.88
62	0	2.00	61	0	2.04
57	0	2.10	50	0	1.84
42	0	1.88	53	0	1.82
47	0	2.35	44	0	1.92
62.49	0.49	1.94	58	0	1.88
62	0	1.76	42	0	1.83
51	0	1.95	68	0	2.10
45	0	1.83	36	0	1.87

$N = 500$					
n	Error	Runtime	n	Error	Runtime
483	0	59.46	486.49	0.49	54.77
485.49	0.49	67.04	475	0	67.05
482	0	65.00	494	0	54.23
485	0	67.42	479	0	62.02
481	0	67.35	483.49	0.49	62.03
487	0	64.71	491	0	56.26
485	0	57.99	481	0	57.16
483.49	0.49	61.92	478	0	57.09
491	0	55.43	489.49	0.49	54.71
489	0	50.26	470	0	61.56
490	0	63.37	491	0	55.39
494	0	54.36	485	0	54.04
495	0	56.35	488	0	53.82
487	0	52.95	488	0	59.48
494	0	54.23	488	0	63.48
481	0	63.20	488	0	64.87

Table 7.1: Maximum clique in random graphs via NN

7.2.3 Cliques in Real-World Graphs via NN

Our experiments include a few real-world graphs from the 10th DIMACs Implementation Challenge, which focus on clustering and partitioning graphs. The results of the real graphs are provided in Table 7.2. We first consider the graph JAZZ, which is a representation of a collaboration network between Jazz musicians [159]. The nodes represent Jazz musicians, whereas the edges indicate that two musicians have collaborated in a band. The JAZZ graph consists of 198 vertices and 2742 edges. In a earlier study [184], a clique of 30 vertices was found in this network. With the value $\rho = 0.25$, we employ **Algorithm** 5 in the adjacency matrix of this graph. After 37 iterations, our algorithm reaches the maximum clique of size 30 within 0.2152 second. We have implemented DSA with $\tau = 0.35$, and stopped DSA with tolerance 10^{-4}. It solves JAZZ in 0.5811 seconds with 94 iterations. We have also applied DSA to all 18 DIMACS benchmark problem and it failed in all problems.

We now compare our algorithm with the algorithm presented in [15] using 18 DIMACS benchmark problems. We implement **Algorithm** 5 with value $\rho = 0.4$. Comparisons are summarized in Table 7.2, where the symbol '-' denotes non-availability of data. Results for the other algorithm under column 4, Table 7.2, were taken from [15].

The number of iterations needed by **Algorithm** 5 is given in the last column. Here $(N, \omega(G))$ represents the number of vertices and the clique number, respectively, while $n(\textit{respectively}, n[15])$ denotes the size of the clique obtained by our algorithm (respectively, by the algorithm in [15]).

Graph	$(N, \omega(G))$	Number of edges	$n(n[15])$	Number of iterations
BROCK200-1	(200,21)	14834	24 (19)	277
BROCK200-4	(200,17)	13089	34 (10)	124
BROCK400-2	(400,29)	59786	46 (24)	284
BROCK400-4	(400,33)	59765	37 (24)	324
C125.9	(125,34)	6963	34 (-)	769
C250.9	(250, 44)	27984	44(-)	1012
C500.9	(500,$\geq$ 57)	112332	216(50)	1442
C-fat500-10	(500,-)	46627	306(-)	3
GEN200-P0.9-44	(200,44)	17910	44 (-)	1000
GEN200-P0.9-55	(200,55)	17910	55 (-)	989
GEN400-P0.9-55	(400,55)	71820	134 (-)	1442
GEN400-P0.9-65	(400,65)	71820	135 (-)	1360
GEN400-P0.9-75	(400,75)	71820	57 (-)	1430
P-HAT300-2	(300,25)	21928	40 (-)	256
P-HAT300-3	(300,36)	33390	219 (-)	225
P-HAT500-2	(500,-)	62946	159 (-)	201
P-HAT700-2	(700,44)	121728	55 (-)	478
P-HAT700-3	(700,62)	183010	209 (-)	654

Table 7.2: Maximum cliques in real-world graphs

Comparison made in Table 7.2 shows that our algorithm performs better than the algorithm in [15] in the tested DIMACS benchmark data sets. Our algorithm recovers the confirmed clique

sizes for 4 problems, while algorithm proposed in [15]) failed to obtained confirmed clique for any problem.

7.3 The MC Problem via Truncated NN

In this section, we evaluate the performance of our TNN-based (Truncated Nuclear Norm) matrix decomposition model for the MC problem as we consider three different problems: PCP, MCP, and the problem of finding MCs in real world graphs.

7.3.1 Planted Cliques via TNN

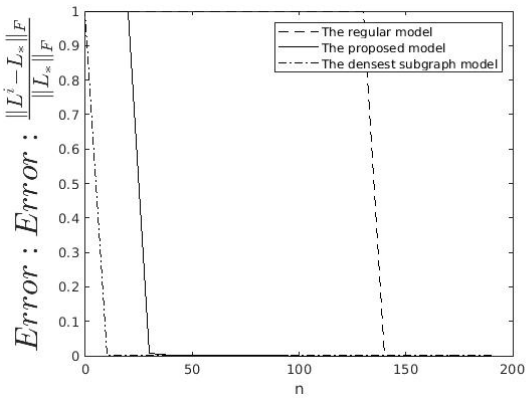
We have implemented the Truncated Nuclear Norm (TNN) algorithm, **Algorithm** 6, and performed experiments on graphs with planted cliques and we have compared the TNN-based model (5.19)-(5.20) with the TNN-based regular model (5.16)-(5.18). We have generated the matrix M in the same way as in Section 7.2.1. The relative errors have been calculated using (7.1).

We have used $N = 200,\ 500$ and 1000. For a fixed value of N, we have used $n = 10, 20, ..., N-10$. Hence the number of problems considered for $N = 200,\ 500$, and 1000, are $19,\ 49$, and 99 respectively. Each of these problem is generated 15 times and hence the total number of test runs was 2505.

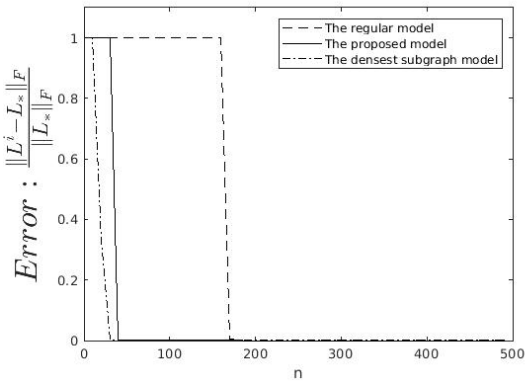
We have initialized **Algorithm** 6 with $M = M_l$, $l = 1$, and **Algorithm** 6 with a randomly generated feasible S of zeros with probability $p = 0.75$ and ones with probability $1 - p = 0.25$. Then we initialize the feasible L as $L = M - S$.

We have used $\delta = 10^{-5}$, δ used in (5.28), and $\rho = \frac{1}{mean(M)}$ in the augmented Lagrangian in (5.31). Our numerical investigations suggest that **Algorithm** 6 produces almost insensitive results for $\lambda = \frac{\alpha}{\sqrt{N}}$ for any $\alpha \in [0.0021, 0.0914]$. Here, we have used $\alpha = 0.0440$. We have used $\rho = 0.07$, $\rho_{\max} = 10000$, $\delta = 10^{-5}$, and $\varrho = 1.0001$. These parameters are related to TNN-based models given in Section 5.3.5. We terminate **Algorithm** 6 when (5.28) is satisfied.

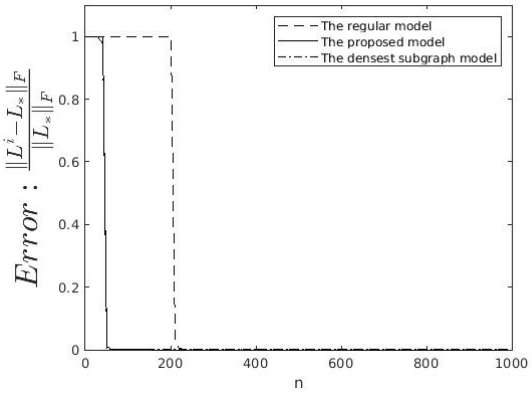
First we compare the average errors in Figure 7.7, where the y-axis denotes the average of relative errors in (7.1); the average is taken over 15 runs on each problem. The value n in the x-axis denotes the size of the planted clique.



(a) $N = 200$



(b) $N = 500$



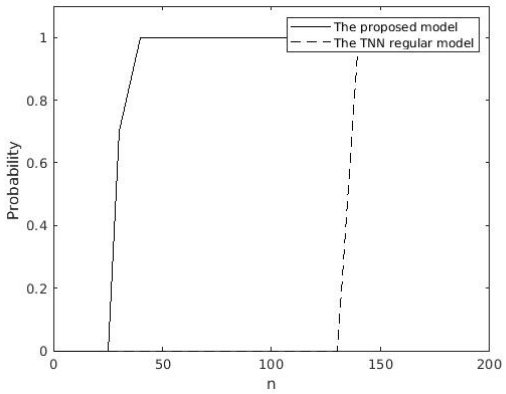
(c) $N = 1000$

Figure 7.7: Average error for all problems.

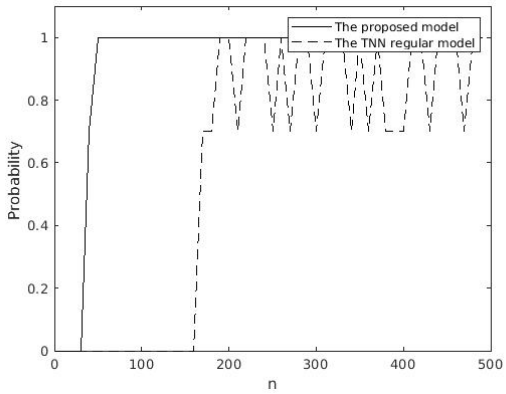
Figure 7.7 shows errors for the regular TNN model (5.16)-(5.18) are worse than the proposed model (5.19)-(5.20) as well as DSA for all (N, n) pairs. It also shows that error for the ADMM algorithm for the regular TNN model does improve for higher values of n, i.e., for the problems that are easier to solve. On the other hand, our proposed model (5.19)-(5.20) achieves errors less than 10^{-8} for all $n \geq 30$, $n \geq 50$, and $n \geq 60$ for $N = 200,\ 500$, and 1000, respectively. The

errors produced by the regular TNN model (5.16)-(5.18) are about 10^{-4} for some (N, n) pairs, it fails to produce less error than 10^{-4}. However, the errors produced by DSA are about 10^{-5}, it fails to produce less error than 10^{-5}.

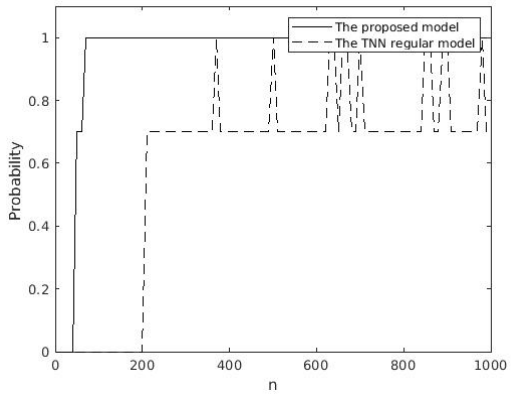
Next, we compare both the regular TNN model (5.16)-(5.18) and the proposed model (5.19)-(5.20), with the DSA using the probability of recovery for all problems corresponding to all (N, n) pairs. This comparison has been summarized in Figure 7.8. Here, by recovery we mean that the obtained solution has average error less than 10^{-8} for the proposed model (5.19)-(5.20), and 10^{-4} for the regular TNN model (5.20)-(5.18) using **Algorithm** 6, and about 10^{-5} using the DSA [6, 18].



(a) $N = 200$



(b) $N = 500$

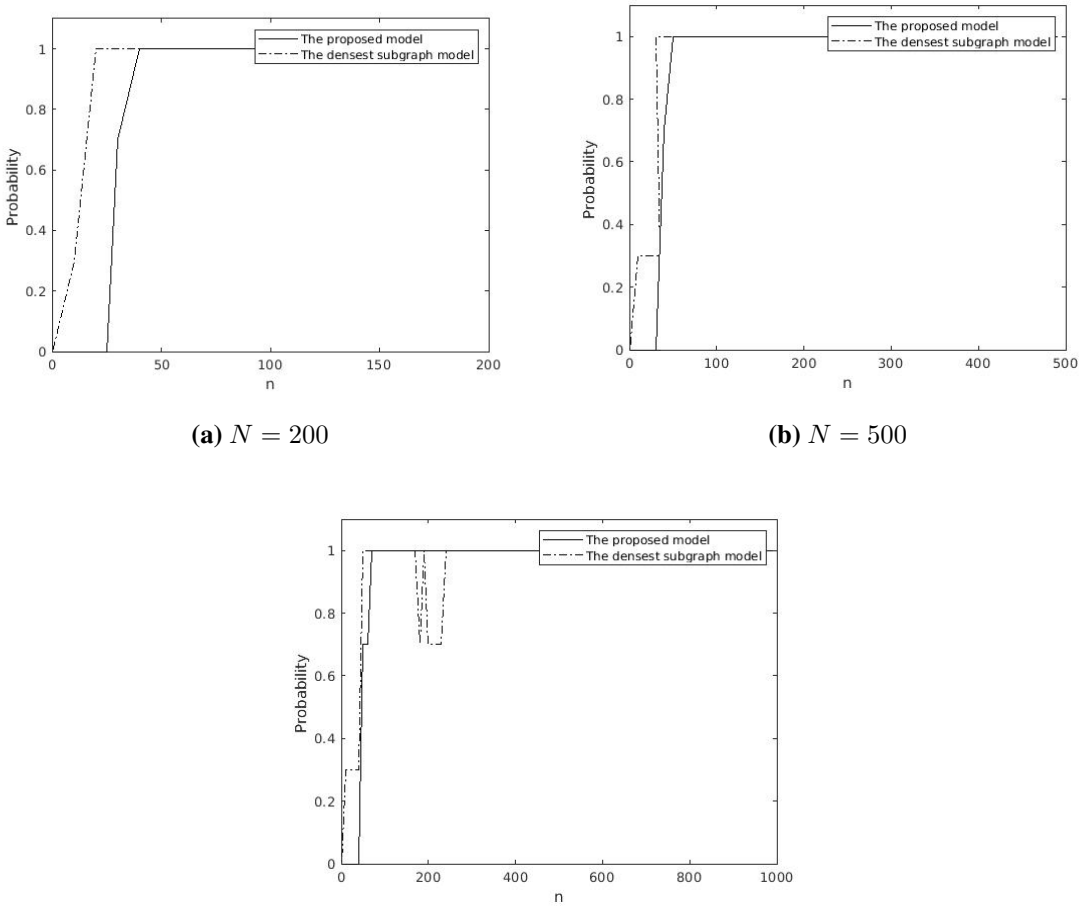


(c) $N = 1000$

Figure 7.8: Average recovery probability for all problems.

Figure 7.8 shows that the probability equals one almost for all (N, n) pairs using the proposed model (5.19)-(5.20). Figure 7.8 also shows that the regular TNN model (5.16)-(5.18) has not provided perfect recovery for all the problems considered.

We have also compared our proposed model with DSA. Figure 7.9, shows that both models recover the planted cliques, however, DSA failed to recover the planted clique for some values when $N = 1000$.



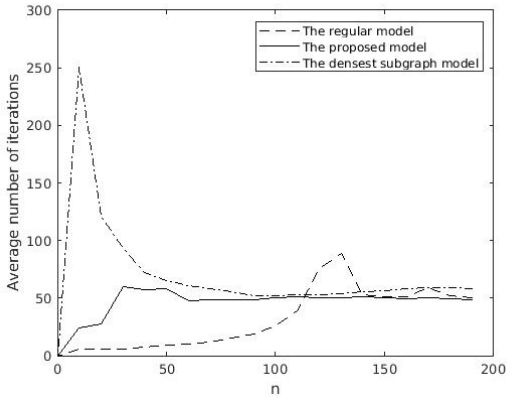
(a) $N = 200$

(b) $N = 500$

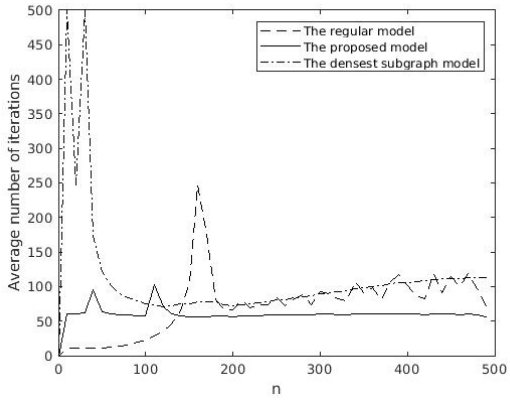
(c) $N = 1000$

Figure 7.9: Average recovery probability for all problems.

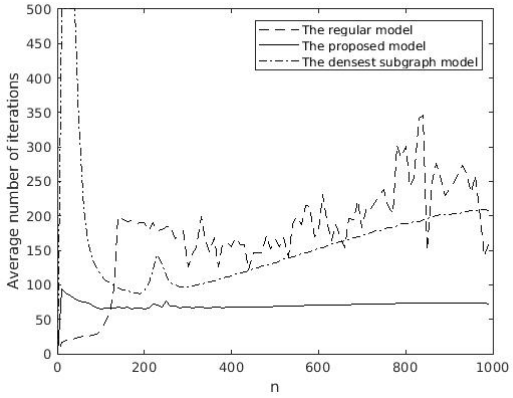
In Figure 7.10 we present the average number of iterations needed, for both models, by **Algorithm** 6, for producing average error of 10^{-8} for the proposed model (5.19)-(5.20) and 10^{-4} using the regular TNN model (5.16)-(5.18), and DSA for producing error of 10^{-5}.



(a) $N = 200$



(b) $N = 500$

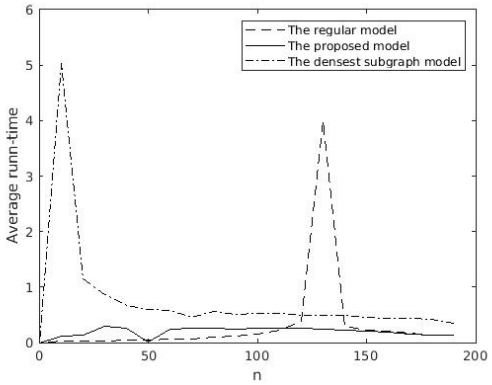


(c) $N = 1000$

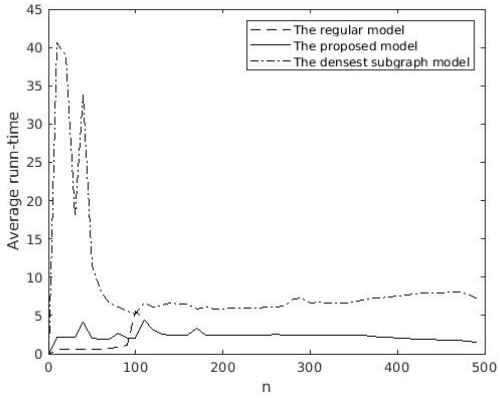
Figure 7.10: Average number of iterations for all problems.

Figure 7.10 demonstrates that both the regular TNN model (5.16)-(5.18) and DSA require more iterations to converge than our proposed model (5.19)-(5.20).

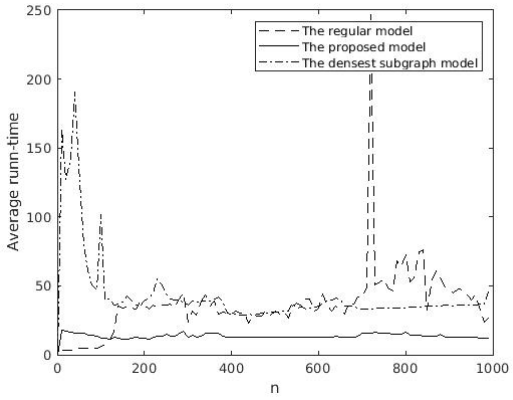
To give an impression of the runtime needed by our algorithm for the regular TNN model (5.16)-(5.18), the proposed model (5.19)-(5.20), and DSA we have summarized the average of the total runtime and average runtime per iterations in the following figures.



(a) $N = 200$



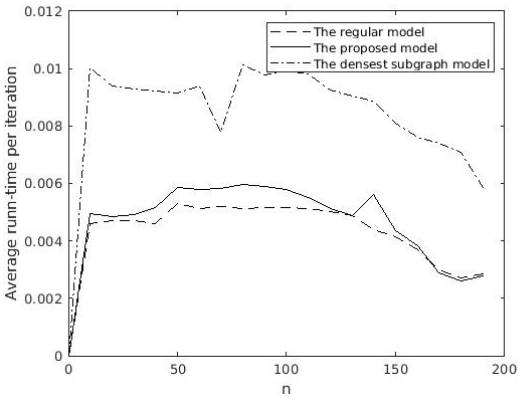
(b) $N = 500$



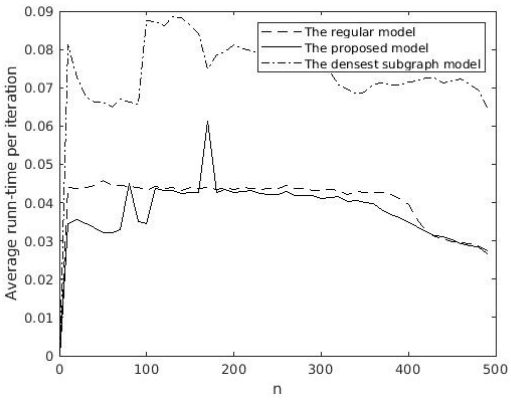
(c) $N = 1000$

Figure 7.11: Average runtime for all problems.

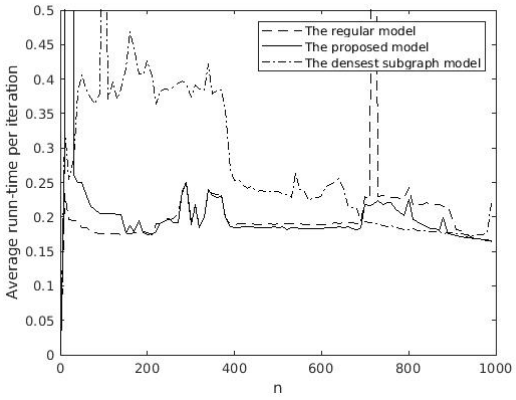
Figure 7.11 shows the average runtime for all (N, n) pairs for the problems considered, the averages are taken over 15 runs on each problem. Figure 7.11 shows that our algorithm for the proposed model (5.19)-(5.20) performs better than the regular TNN model (5.16)-(5.18) and DSA in finding the optimal solution. Figure 7.12 shows the average runtime per iteration for the all the problems considered, where our algorithm for the proposed model (5.19)-(5.20) performs slightly better.



(a) $N = 200$



(b) $N = 500$



(c) $N = 1000$

Figure 7.12: Average runtime per iteration for all problems.

7.3.2 The MCP in Random Graphs via TNN

We have also performed experiments on random graphs where all the edges are assigned with probability p, with no clique being planted, using the TNN approach. These results are shown in Table 7.3 where n is the size of the maximum clique obtained by our algorithm. We have used the same stopping condition (5.28) to stop the algorithm, but calculated the errors using the formula in (7.2).

The solution (L, S) is initialized with $(0, 0)$. A total of 32 runs have been performed for each N, and the results presented in each row of Table 7.3 are obtained for a single run. We have used

$p = 0.8$ and $p = 0.85$ for $N = 200$ and $N = 500$ respectively. We have used higher values for p to ascertain that a reasonable size of maximum clique is formed in each random graph. Results in Table 7.3 show the perfect recovery of maximum cliques based on nodes in the clique except for a small number of cases having some errors. These errors occur because the generated random graph has a bi-clique of size greater than the size of the clique.

$N = 200$					
n	Error	Runtime	n	Error	Runtime
114	0	1.26	125	0	1.12
114.99	0.99	3.88	113	0	1.06
159	0	0.99	162	0	2.94
113	0	1.06	116	0	1.09
130	0	1.08	125	0	1.25
116.99	0.99	3.82	124	0	1.14
142	0	1.01	126	0	1.13
105	0	1.00	120	0	1.20
119	0	1.14	132	0	1.07
123	0	1.09	115	0	1.09
112	0	1.09	110	0	1.02
106	0	1.14	125	0	1.06
128	0	1.04	107.99	0.99	3.63
126	0	1.15	159	0	0.94
116	0	1.05	124	0	1.16
121	0	1.13	119	0	1.07

$N = 500$					
n	Error	Runtime	n	Error	Runtime
426	0	13.20	444	0	12.70
452	0	12.14	452	0	12.34
488	0	13.27	433	0	11.98
437	0	12.65	490	0	13.74
440	0	12.96	431	0	11.96
425	0	11.85	439	0	12.04
437	0	12.46	448	0	12.36
452	0	12.64	440	0	12.01
493	0	14.16	411	0	14.19
434	0	12.28	433	0	14.33
452	0	12.43	490	0	12.92
400	0	14.95	439	0	12.24
447	0	12.35	435	0	12.27
434	0	12.25	398	0	13.65
424	0	13.92	444	0	13.98
435	0	11.82	418	0	13.42

Table 7.3: Maximum clique in random graphs via TNN

7.3.3 Cliques in Real-World Graphs via TNN

Here, we have implemented our proposed model (5.19)-(5.20). We also consider the graphs JAZZ, and 18 DIMACS benchmark data sets, which have been considered in Section 7.2.3, and we have determined the results in Table 7.4.

We now compare our algorithm with the algorithm presented in [15] using 18 DIMACS benchmark problems. We implement **Algorithm** 6 with value $\rho = 0.4$. Comparisons are summarized in Table 7.2, where the symbol '-' denotes non-availability of data. Results for the other algorithm under column 4, Table 7.4, were taken from [15].

The number of iterations needed by the alghorithm in [134] and **Algorithm** 6 are given under the columns k_1 and k_2, respectively. The algorithm in [134] replaces the objective function (5.19) with $\|L\|_* + \lambda\|C \circ S\|_1$. Here $(N, \omega(G))$ represents the number of vertices and the clique number, respectively, while n*[134]*$(respectively, n$*[15]*$)$ denotes the size of the clique obtained by the algorithm in [134](respectively, by the algorithm in [15]). The clique sizes obtained by our algorithm is under the last column Algorithm 6.

Graph	$(N, \omega(G))$	# edges	n*[134]*$(n$ *[15]*$)$	k_1	k_2	Algorithm 6
BROCK200-1	(200,21)	14834	24 (19)	277	293	48
BROCK200-4	(200,17)	13089	34 (10)	124	41	211
BROCK400-2	(400,29)	59786	46 (24)	284	1509	83
BROCK400-4	(400,33)	59765	37 (24)	324	1233	68
C125.9	(125,34)	6963	34 (-)	769	11800	43
C250.9	(250, 44)	27984	44(-)	1012	7928	52
C500.9	(500,$\geq$ 57)	112332	216(50)	1442	2910	460
C-fat500-10	(500,-)	46627	306(-)	3	4	306
GEN200-P0.9-44	(200,44)	17910	44 (-)	1000	1089	171
GEN200-P0.9-55	(200,55)	17910	55 (-)	989	1134	173
GEN400-P0.9-55	(400,55)	71820	134 (-)	1442	2590	382
GEN400-P0.9-65	(400,65)	71820	135 (-)	1360	2489	390
GEN400-P0.9-75	(400,75)	71820	57 (-)	1430	2629	289
P-HAT300-2	(300,25)	21928	40 (-)	256	590	36
P-HAT300-3	(300,36)	33390	219 (-)	225	411	148
P-HAT500-2	(500,-)	62946	159 (-)	201	201	159
P-HAT700-2	(700,44)	121728	55 (-)	478	225	195
P-HAT700-3	(700,62)	183010	209 (-)	654	244	201

Table 7.4: Maximum cliques in real-world graphs

Comparison made in Table 7.4 shows that our algorithm performs better than the algorithm in [15] in the tested DIMACS benchmark data sets. Our algorithm in [134] recovers the confirmed clique sizes for four problems, while algorithm proposed in [15]) failed to obtained confirmed clique for any problem. However, our proposed algorithm presented in this paper, **Algorithm** 6, successfully produced maximum clique with larger sizes than those reported in DIMACs website. We have reported these findings to DIMACs website.

7.4 The Maximum Edge Bi-clique Problem via NN

In this section, we evaluate the performance of our NN-based matrix decomposition model for the Maximum Edge Bi-clique (MEB) problem as we consider two different problems: PEBP and MEBP.

7.4.1 Planted Bi-cliques via NN

Let $V_1^* \times V_2^*$ denote the planted bi-clique, $|V_1^*| = n$, $|V_2^*| = m$. Let M_b represent the adjacency matrix of the bipartite graph $G_b = (V_1, V_2, E)$, $|V_1| = N$, and $|V_2| = M$. We set $(M_b)_{ij} = 1$ for $(i, j) \in V_1^* \times V_2^*$. The remaining edges $(i, j) \in (V_1 \times V_2) \backslash (V_1^* \times V_2^*)$ are included with probability p. The proposed algorithm has been tested and achieved very similar accuracy for $p \in [0.5, 0.85]$ for all problems considered. However, here we present results based on a p value of 0.5.

We have implemented **Algorithm** 5 with $M = M_b$. **Algorithm** 5 has been initialized with a randomly generated feasible S of ones with probability $p = 0.25$ and zeros with probability $1 - p = 0.75$. The feasible L is then initialized to be $L = M_b - S$. Throughout our numerical testing, we have used a constant regularization parameter λ. According to our numerical findings, **Algorithm** 5 provides almost insensitive results for $\lambda = \frac{\alpha}{\sqrt{\max(N,M)}}$ for any $\alpha \in [0.0027, 0.15]$. We have estimated the range, $[l, u]$, for α as follows. First we calculate three ranges $[l_i, u_i]$, $i = 1, 2, 3$, corresponding to $c = 0.25, 0.5$ *and* 0.75, respectively in $s = \frac{1}{2}(NM - nm)$, $n = cN$, $m = cM$. We plot α $(= \max(N, M)/s)$ against $\max(N, M)$ for each c value and obtained $[l_i, u_i]$ for α. We then take $l = \min l_i$ and $u = \max u_i$, $i = 1, 2, 3$. We have used $\alpha = 0.08$ for all (N, n) pairs for the results presented here. The values of ϵ in (5.8) lies within the range $[0.05, 0.42]$. We have used $\epsilon = 0.05$ for the results presented here. We have used $\rho = \frac{1}{mean(M_b)}$, where $mean(M_b)$ is the average value of entries in M_b.

As before, for the regular model, the final solution of ADMM algorithm is denoted as (L^1, S^1), whereas for the proposed model (5.12)-(5.13), the final solution is denoted as (L^2, S^2).

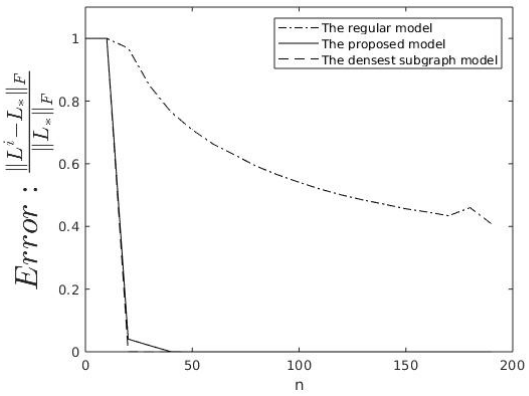
For each algorithm, the relative error $errL^i$ is calculated using the Frobenius norm formula (7.1), where L_* corresponds to $V_1^* \times V_2^*$, the planted bi-clique.

We terminate **Algorithm** 5 when (5.28) holds, where $M = M_b$ and (L_J, S_J) is the solution at iterate J.

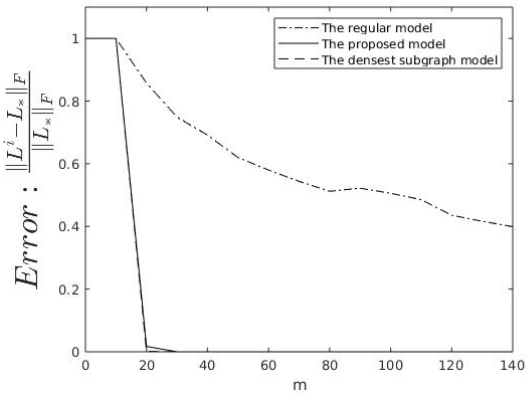
We have used bipartite graphs of sizes 200×150, 500×350, and 1000×800. For each value of n and m the procedures below of generating the bipartite graph M_b is repeated 15 times. In each case we have used the procedure in each M_b. For a bi-clique of size nm, we fixed n at one value and set $m = 10, 20, ..., M - 10$, therefore, the number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 210, 510, and 1185, respectively. Similarly, we also fix the value of m at one value and let $n = 10, 20, ..., N - 10$, therefore, the number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 285, 735, and 1485, respectively. Hence the total number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 495, 1245, and 2535, respectively, and hence the total number of test runs was 4410.

We first start with a bipartite graph of size 200×150 where the input matrix M_b is constructed as above. For the bi-clique of size nm, we fix the value of n to be 100 and vary the value of m; we have used $m = 10, 20, ..., M - 10$. We also fix the value of m to be 70 and vary the value of n; we have used $n = 10, 20, ..., N - 10$.

We have compared Algorithm 5 with the densest subgraph algorithm (DSA) [18] for all the problems considered in this section. Here we note that, unlike our algorithm, DSA requires size nm of the planted bi-clique as input. Figure 7.13 shows average errors obtained, where the y-axis indicates the average of relative errors; this average is determined over 15 runs on each problem. The value n or m in the x-axis denotes the varied part of the size of the planted bi-clique while the other part remained fixed.



(a) $N = 200, M = 150$, and fixed $m = 70$

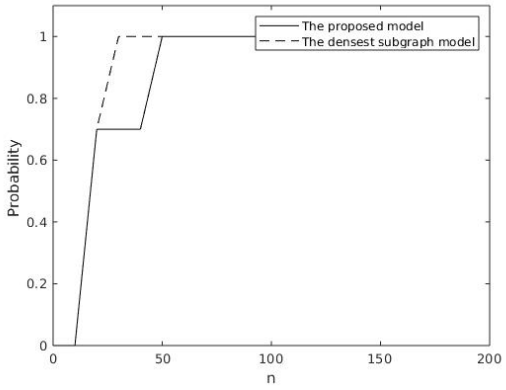


(b) $N = 200, M = 150$, and fixed $n = 100$

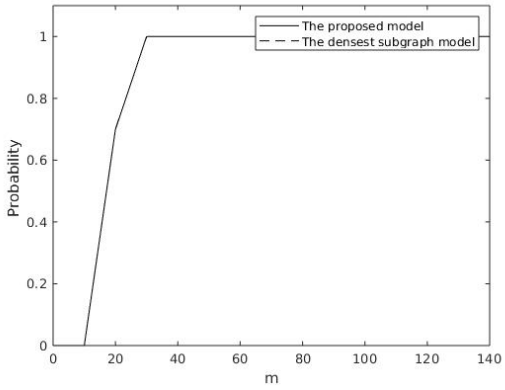
Figure 7.13: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{200\times 150}$.

Figure 7.13 shows that the ADMM algorithm for the regular model (5.5)-(5.7) has worse errors than that of the proposed model for all pairs (N, M) and (n, m). In addition, the ADMM algorithm for the regular model shows that error does improve for higher values of n and m, i.e., for the easier problems. On the other hand, our proposed model (5.12)-(5.13) achieves errors less than 10^{-8} for all $n, m \geq 30$ for $N = 200$. However, the errors produced by DSA are about 10^{-5}.

We now present the probability of recovery for each problem where recovery we mean that the obtained solution has average error less than 10^{-8} for **Algorithm** 5 and about 10^{-5} for DSA [18]; DSA fails to produce errors less than 10^{-5}. This is demonstrated in Figure 7.14 for all N, M, n and m. Figure 7.14 shows both algorithms are comparable, noting the fact that the ADMM for (5.12)-(5.13) produces small errors than DSA.



(a) $N = 200, M = 150$, and fixed $m = 70$

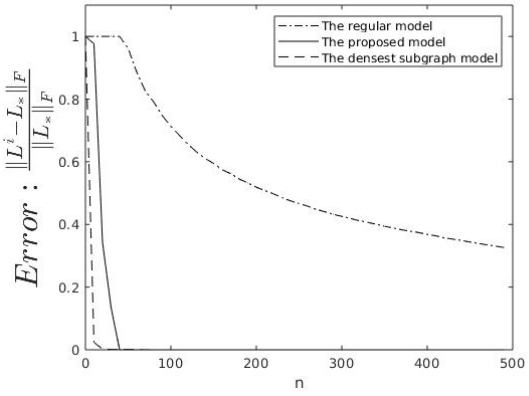


(b) $N = 200, M = 150$, and fixed $n = 100$

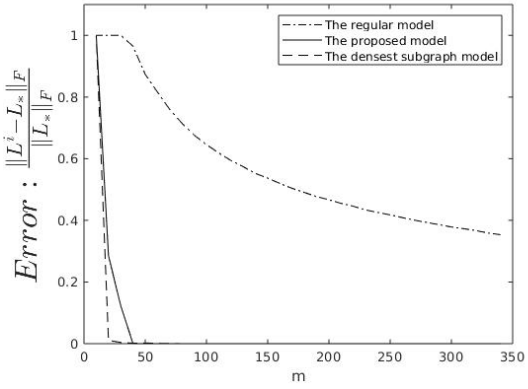
Figure 7.14: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{200\times150}$.

The average recovery probabilities for bi-cliques of sizes nm are presented in Figure 7.14a (respectively Figure 7.14b) correspond to the results in Figure 7.13a (respectively 7.13b).

We now perform numerical studies using larger values for $N, M, n,$ and m. The relative error values we shown in Figure 7.15 and Figure 7.16 for problems corresponding to $M_b \in \mathbb{R}^{500\times350}$ and $M_b \in \mathbb{R}^{1000\times800}$, respectively.

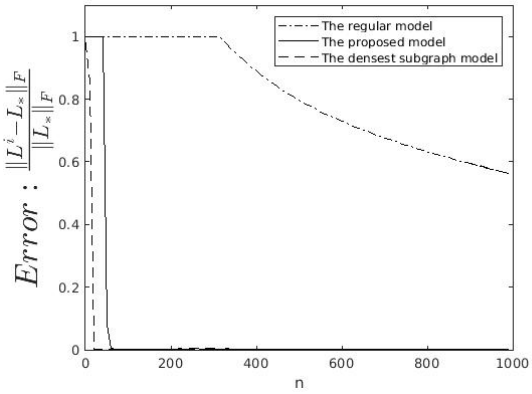


(a) $N = 500, M = 350$, and fixed $m = 200$

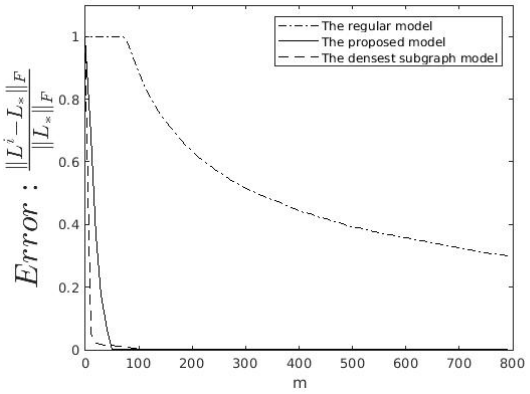


(b) $N = 500, M = 350$, and fixed $n = 250$

Figure 7.15: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{500\times350}$.



(a) $N = 1000, M = 800$, and fixed $m = 150$

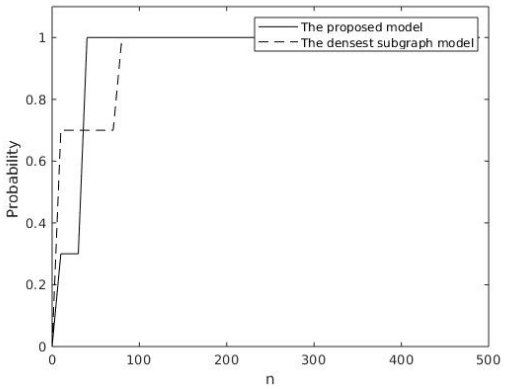


(b) $N = 1000, M = 850$, and fixed $n = 600$

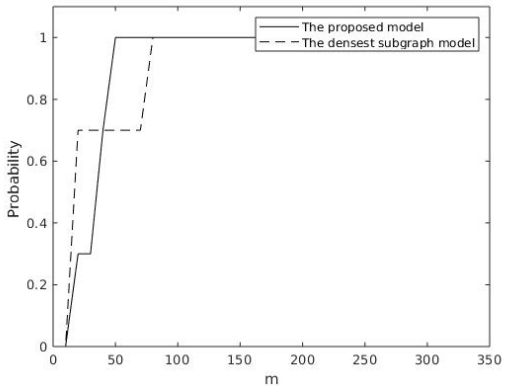
Figure 7.16: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{1000\times800}$.

Figures 7.15 and 7.16 show that while the ADMM for (5.12)-(5.13) produces small errors than DSA; both algorithms outperform the regular model.

The average recovery probabilities for bi-cliques of size nm using $M_b \in \mathbb{R}^{500\times350}$ and $M_b \in \mathbb{R}^{1000\times800}$ are demonstrated in Figure 7.17 and 7.18, respectively. Figure 7.15a (respectively Figure 7.15b) corresponds to the results in Figure 7.17a (respectively 7.17b). Similarly, Figure 7.16a (respectively Figure 7.16b) corresponds to the results in Figure 7.18a (respectively 7.18b).

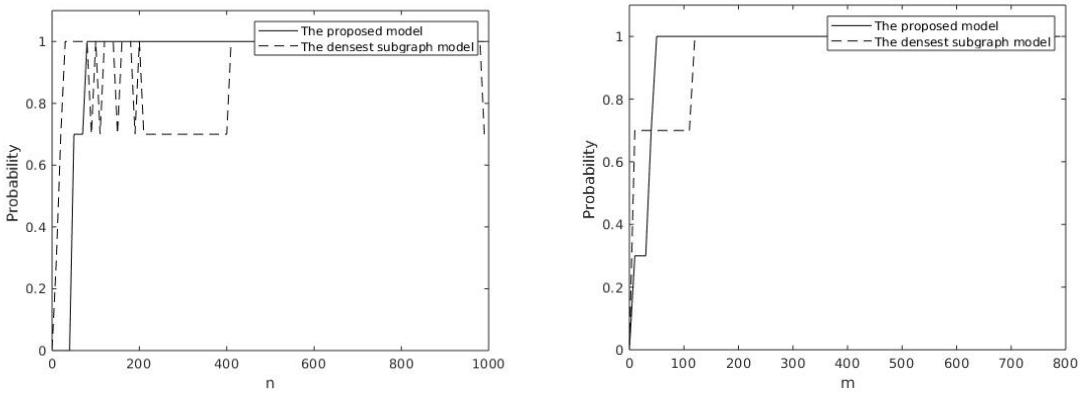


(a) $N = 500, M = 350$, and fixed $m = 200$



(b) $N = 500, M = 350$, and fixed $n = 250$

Figure 7.17: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{500\times350}$.

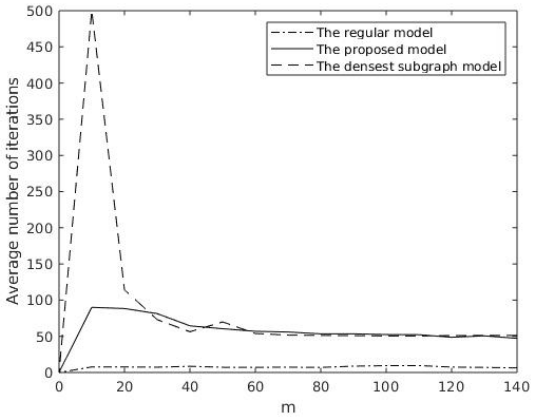


(a) $N = 1000, M = 800$, and fixed $m = 150$ **(b)** $N = 1000, M = 800$, and fixed $n = 600$

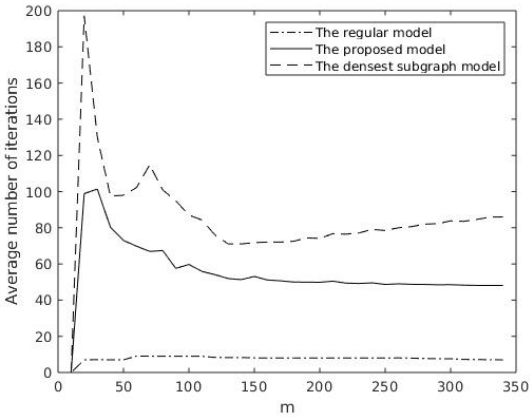
Figure 7.18: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{1000\times 800}$.

Results summarized in Figures 7.17 and 7.18 show the perfect recovery of planted bi-cliques with our algorithm for (5.12)-(5.13) with probability equals to one for most of the considered cases. On the other hand, DSA fails to obtain perfect recovery for a number of problems.

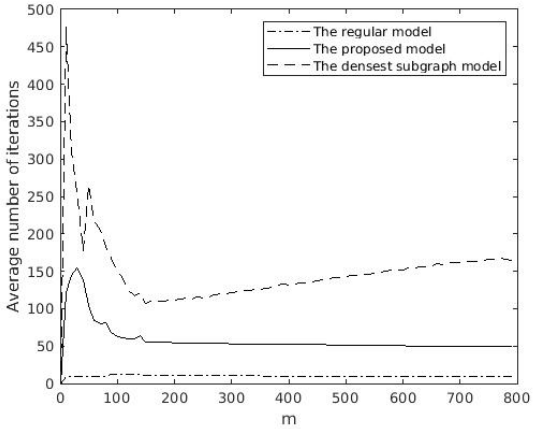
In Figure 7.19 and 7.20 we present the average number of iterations needed by Algorithm 5 for producing average error of 10^{-8} and DSA for producing average error of 10^{-5}.



(a) $N = 200,\ M = 150$, and fixed $n = 100$

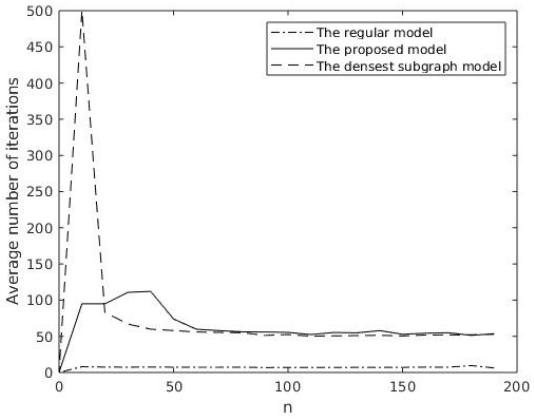


(b) $N = 500,\ M = 350$, and fixed $n = 250$

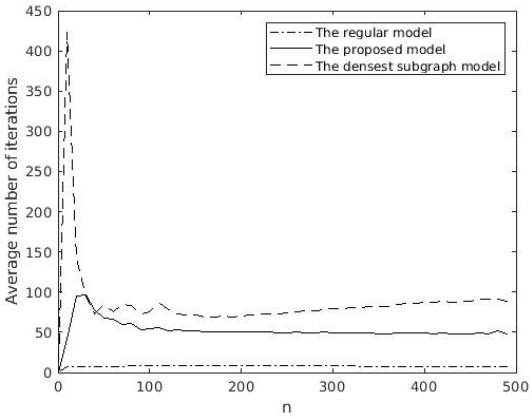


(c) $N = 1000,\ M = 800$, and fixed $n = 600$

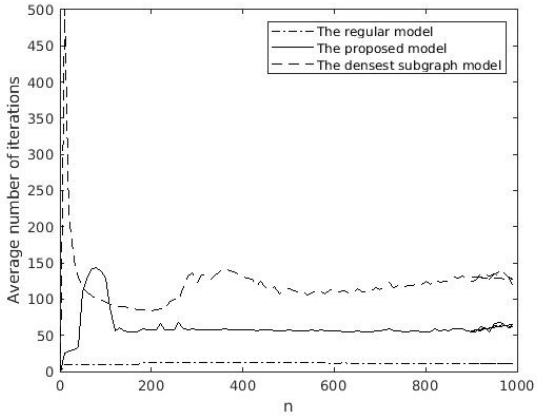
Figure 7.19: The average number of iterations for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$

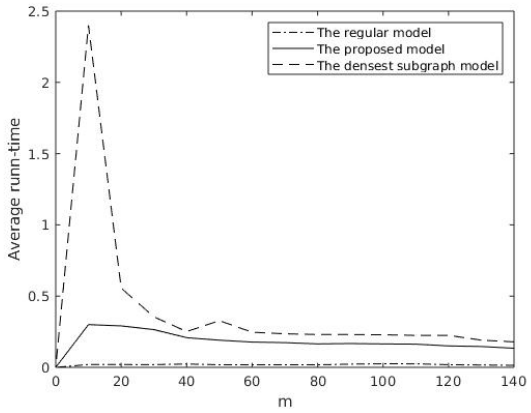


(c) $N = 1000$, $M = 800$, and fixed $m = 150$

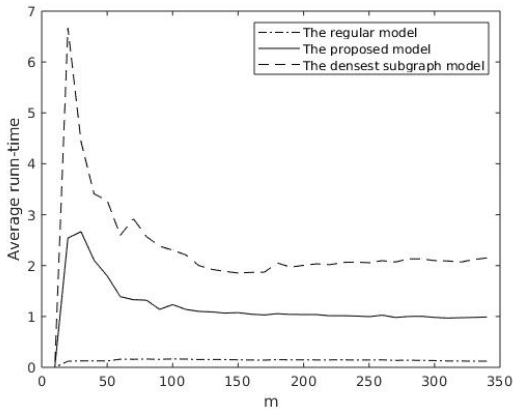
Figure 7.20: The average number of iterations for each problem with varied n.

Figures 7.19 and 7.20 show that our algorithm for (5.12)-(5.13) produces better errors with a smaller number of iterations than DSA, on average.

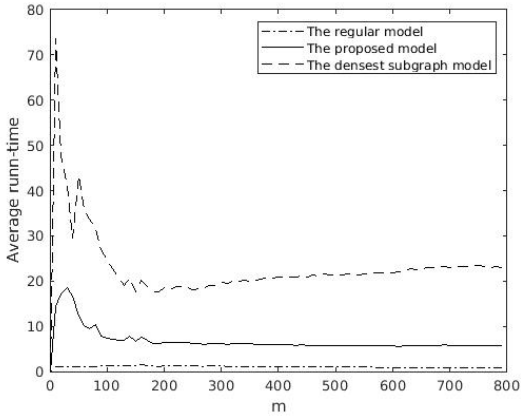
The total runtime and average runtime per iteration needed by our algorithm for (5.12)-(5.13) and DSA are summarized in Figures 7.21 and 7.22.



(a) $N = 200,\ M = 150$, and fixed $n = 100$

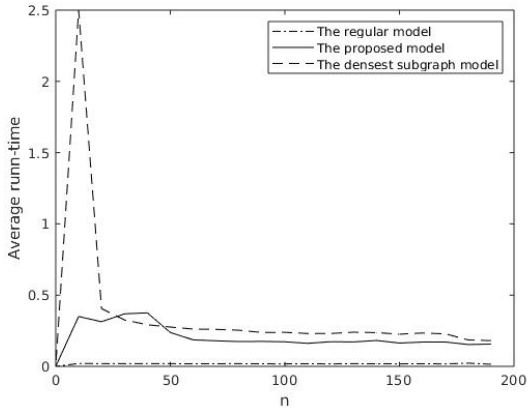


(b) $N = 500,\ M = 350$, and fixed $n = 250$

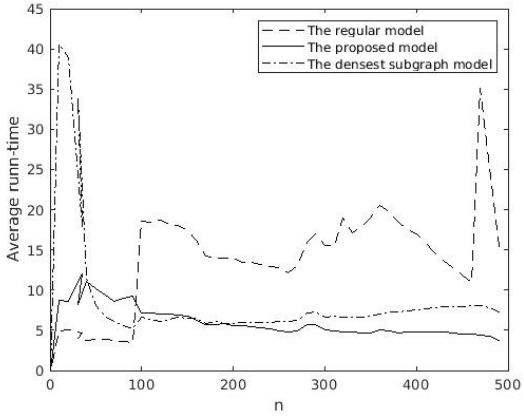


(c) $N = 1000,\ M = 800$, and fixed $n = 600$

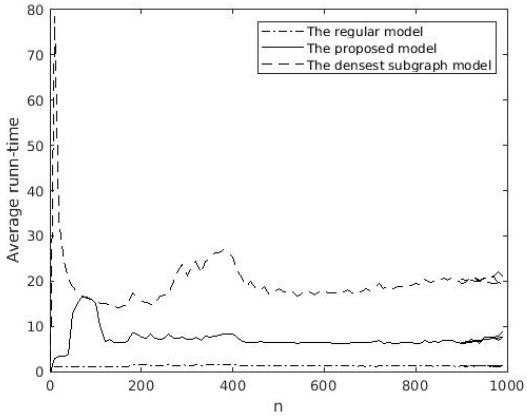
Figure 7.21: Average runtime for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$

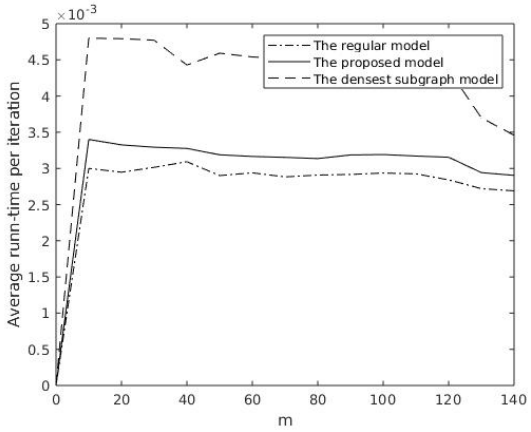


(c) $N = 1000$, $M = 800$, and fixed $m = 150$

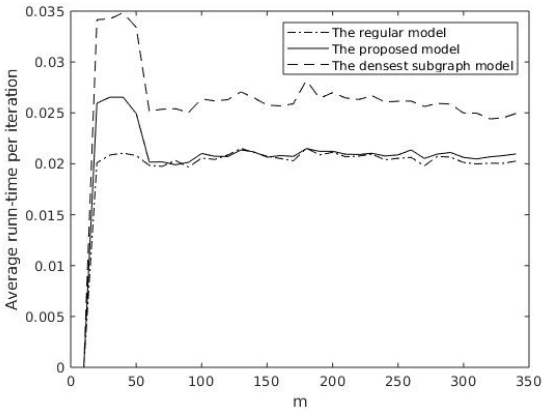
Figure 7.22: Average runtime for each problem with varied n.

Figures 7.21 and 7.22 show the average runtime for all (N, n) and (M, m) pairs for the problems considered, the averages are taken over 15 runs on each problem. These figures indicate that our algorithm for (5.12)-(5.13) outperforms DSA in finding the optimal solution for both cases when m and n being varied.

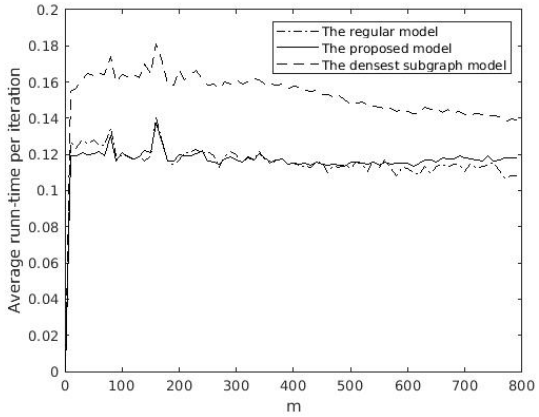
Figures 7.23 and 7.24 show the average runtime per iteration for the all the problems considered, where our algorithm for (5.12)-(5.13) performs slightly better, on average.



(a) $N = 200$, $M = 150$, and fixed $n = 100$

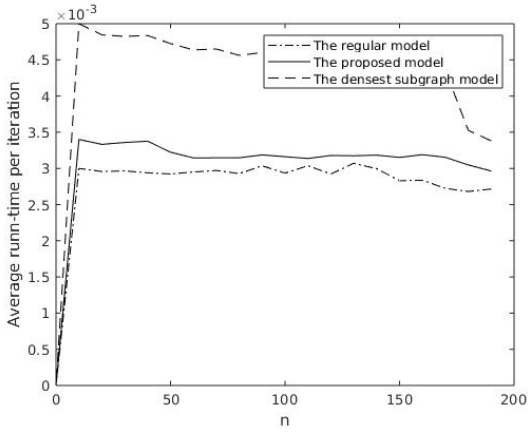


(b) $N = 500$, $M = 350$, and fixed $n = 250$

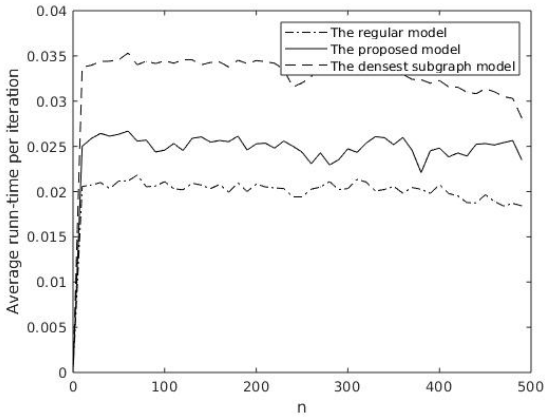


(c) $N = 1000$, $M = 800$, and fixed $n = 600$

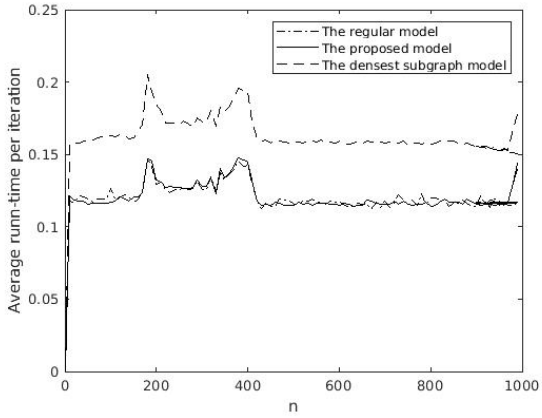
Figure 7.23: Average runtime per iteration for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



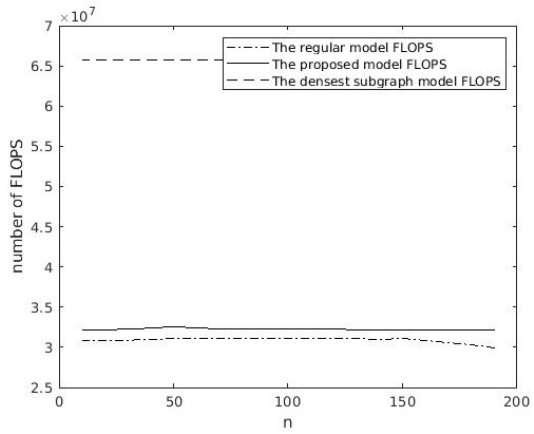
(b) $N = 500$, $M = 350$, and fixed $m = 200$



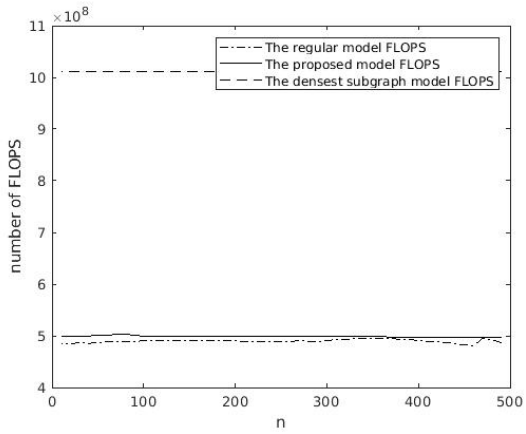
(c) $N = 1000$, $M = 800$, and fixed $m = 150$

Figure 7.24: Average runtime per iteration for each problem with varied n.

To clarify the scaling of our proposed approach, we sketch the number of FLOPS (Floating Point Operations per Second) needed per iteration. Figure 7.25 demonstrates the average number of FLOPS needed per iteration for $N = 200$ and $N = 500$.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$

Figure 7.25: Average number of FLOPS per iteration.

Figure 7.25 shows that $\mathcal{O}(N^3)$ FLOPS are required per iteration. It also shows that our proposed algorithm needed less number of FLOPS than DSA.

7.4.2 Bi-cliques in Random Graphs via NN

We have also performed experiments on random bipartite graphs where the edges are assigned with probability p. These results are presented in the table below, where nm is the size of the maximum bi-clique obtained by our algorithm. We have used the same stopping condition in (5.28), where $M = M_b$, but calculated the errors using the condition

$$Error = \left| \sum_{i=1}^{N} \sum_{j=1}^{M} (L_*)_{ij} - \|L_*\| \right|, \tag{7.3}$$

since we have shown earlier that $\|L_*\| = \sigma_1 = \sqrt{nm}$, where L_* represents the low-rank matrix (corresponds to the bi-clique) obtained by our algorithm. Our suggested formula (7.3) can measure the recovery of maximum bi-clique based on nodes in the bi-clique.

The solution (L, S) is initialized as in the case of planted bi-clique. A total of 48 runs have been performed by considering a number of (N, M) pairs. Results presented in each row of Table 7.7 are obtained for a single run. We have used two values for N, i.e., $N = 200, 500$ and varied M in

creating the pair (N, M) for this experiment. We have used $p = 0.85$ for generating $M_b \in \mathbb{R}^{200\times M}$ and $p = 0.88$ for $M_b \in \mathbb{R}^{500\times M}$ for each M considered. We have used higher values for p to ascertain that a reasonable size of maximum bi-clique, (n, m), is formed in each random graph. Results obtained show the perfect recovery of maximum bi-cliques having little or no errors.

Table 7.5: Random graph with fixed $N = 200$

(N, M)	(n, m)	CPU time	Error
$(200, 170)$	$(133, 119)$	1.7462	0
	$(113, 119)$	1.7383	$3.9790e^{-13}$
	$(124, 104)$	1.6812	$4.2633e^{-14}$
	$(111, 106)$	1.6811	$7.1054e^{-14}$
$(200, 145)$	$(111, 18)$	1.2597	$2.8422e^{-14}$
	$(132, 107)$	1.5079	$1.4211e^{-14}$
	$(108, 103)$	1.4105	$1.4211e^{-14}$
	$(101, 101)$	1.3989	$2.844e^{-14}$
$(200, 100)$	$(139, 82)$	0.7479	$2.2737e^{-13}$
	$(126, 91)$	0.8779	$9.9476e^{-14}$
	$(118, 70)$	0.7656	0
	$(95, 86)$	0.7811	$1.1369e^{-13}$
$(200, 77)$	$(96, 67)$	0.4822	$7.1154e^{-14}$
	$(96, 74)$	0.5032	$5.6042e^{-14}$
	$(137, 67)$	0.5321	$1.5632e^{-13}$
	$(123, 72)$	0.5036	$7.1050e^{-14}$
$(200, 45)$	$(108, 45)$	0.2810	$1.4211e^{-14}$
	$(85, 40)$	0.2751	$3.5527e^{-14}$
	$(80, 45)$	0.2661	$7.1053e^{-14}$
	$(95, 38)$	0.2887	0
$(200, 20)$	$(72, 20)$	0.1338	0
	$(43, 18)$	0.1259	0
	$(56, 17)$	0.1294	$7.1054e^{-14}$
	$(80, 19)$	0.1135	$2.1316e^{-14}$

Table 7.6: Random graph with fixed $N = 500$

(N, M)	(n, m)	CPU time	Error
$(500, 350)$	$(330, 283)$	16.6764	0
	$(414, 327)$	22.5961	$1.7053e^{-13}$
	$(451, 329)$	22.0645	$1.1369e^{-14}$
	$(447, 338)$	20.7785	$5.6843e^{-14}$
$(500, 265)$	$(461, 257)$	14.5689	$3.9790e^{-13}$
	$(476, 263)$	15.0924	$5.1159e^{-13}$
	$(468, 258)$	12.8741	$5.1158e^{-13}$
	$(460, 258)$	13.7202	$4.5475e^{-13}$
$(500, 155)$	$(315, 149)$	3.4211	$1.9875e^{-13}$
	$(298, 146)$	3.3781	$1.7153e^{-13}$
	$(349, 148)$	3.8173	$8.5265e^{-14}$
	$(297, 144)$	3.2799	$1.1369e^{-14}$
$(500, 100)$	$(123, 71)$	1.3621	$1.2412e^{-14}$
	$(107, 70)$	1.2230	$1.4211e^{-14}$
	$(122, 74)$	1.1793	$1,4211e^{-14}$
	$(136, 67)$	1.1785	$1.1369e^{-13}$
$(500, 70)$	$(140, 59)$	0.7183	$2.9433e^{-14}$
	$(139, 59)$	0.7111	$2.9433e^{-14}$
	$(160, 52)$	0.7243	$1.7053e^{-13}$
	$(121, 63)$	0.6915	$2.8421e^{-14}$
$(500, 20)$	$(114, 18)$	0.1380	$2.8511e^{-14}$
	$(120, 18)$	0, 1510	$2.1316e^{-14}$
	$(94, 15)$	0.1418	$7.1055e^{-15}$
	$(128, 18)$	0.1625	$7.1055e^{-15}$

Table 7.7: Maximum bi-clique in random graphs via NN

Four random bipartite graphs are generated for each (N, M) pair. The corresponding recovered bi-clique sizes are shown under the column under (n, m).

7.5 The Maximum Edge Bi-clique Problem via TNN

In this section, we evaluate the performance of our TNN-based matrix decomposition model for the MEB problem as we consider two different problems: PEBP and MEBP.

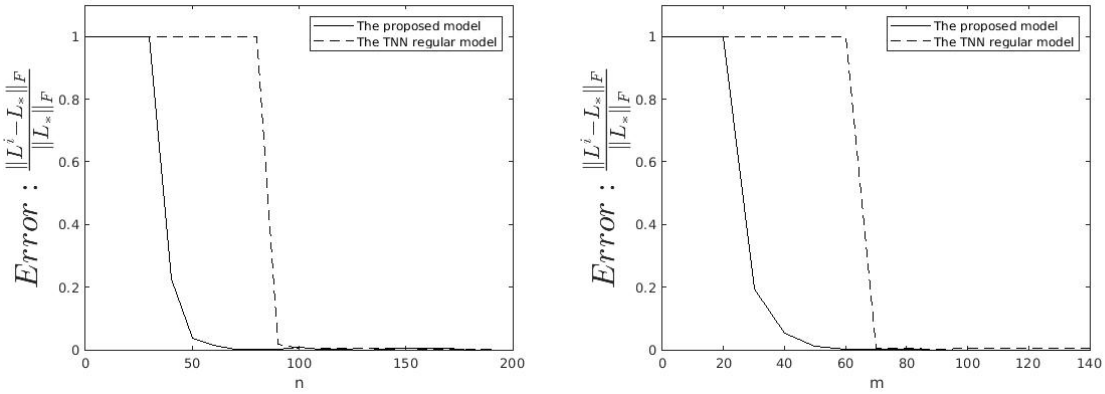
7.5.1 Planted Bi-cliques via TNN

We have implemented **Algorithm** 6 on both TNN-based models, and performed experiments on graphs with planted bi-cliques. We have compared the proposed model (5.19)-(5.20) with the TNN-based regular model (5.16)-(5.18) for all the problems considered in this section. We have generated the matrix M_b in the same way as in Section 7.2.1. The relative errors have been calculated using (7.1).

We have used bipartite graphs of sizes 200×150, 500×350, and 1000×800. For each value of n and m the procedures below of generating the bipartite graph M_b is repeated 15 times. In each case we have used the procedure in each M_b. For a bi-clique of size nm, we fixed n at one value and set $m = 10, 20, ..., M - 10$, therefore, the number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 210, 510, and 1185, respectively. We also fix the value of m at one value and let $n = 10, 20, ..., N - 10$, therefore, the number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 285, 735, and 1485, respectively. Hence the total number of problems considered for graph of sizes 200×150, 500×350, and 1000×800 are 495, 1245, and 2535, respectively, and hence the total number of test runs was 4410.

We first start with a bipartite graph of size 200×150 where the input matrix M_b is constructed as above. For the bi-clique of size $n \times m$, we fix the value of n to be 100 and vary the value of m; we have used $m = 10, 20, ..., M - 10$. We also fix the value of m to be 70 and vary the value of n; we have used $n = 10, 20, ..., N - 10$.

Figure 7.26 shows average errors obtained, where the y-axis indicates the average of relative errors; this average is determined over 15 runs on each problem. The value n or m in the x-axis denotes the varied part of the size of the planted bi-clique while the other part remained fixed.

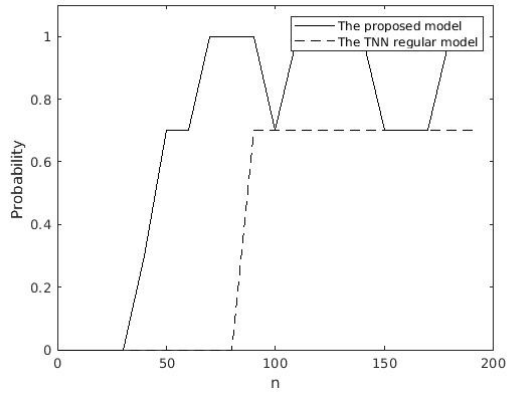


(a) $N = 200, M = 150$, and fixed $m = 70$ **(b)** $N = 200, M = 150$, and fixed $n = 100$

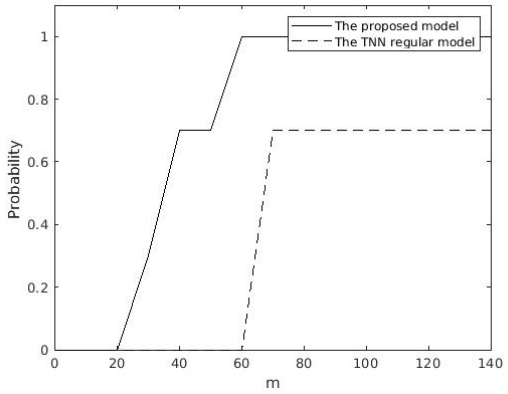
Figure 7.26: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{200\times 150}$.

Figure 7.26 shows that the ADMM algorithm for the TNN-based regular model (5.16)-(5.18) has worse errors than that of the proposed model for all pairs (N, M) and (n, m). In addition, the ADMM algorithm for the TNN-based regular model (5.19)-(5.20) shows that error does improve for higher values of n and m, i.e., for the easier problems.

We now present the probability of recovery for each problem where by recovery we mean that the obtained solution by **Algorithm** 6 has average error less than 10^{-8} for the TNN-based model (5.19)-(5.20) and about 10^{-4} for the TNN-based regular model (5.16)-(5.18). The TNN-based regular model (5.16)-(5.18) fails to produce errors less than 10^{-4}. This is demonstrated in Figure 7.27 for all N, M, n and m. Figure 7.27 shows both algorithms are comparable, noting the fact that ADMM for TNN-based model (5.19)-(5.20) produces smaller errors than the TNN-based regular model (5.16)-(5.18).



(a) $N = 200, M = 150$, and fixed $m = 70$

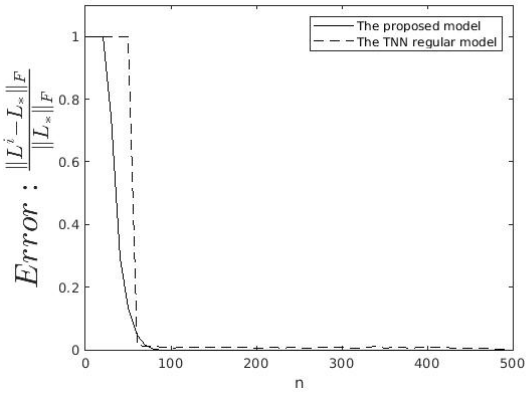


(b) $N = 200, M = 150$, and fixed $n = 100$

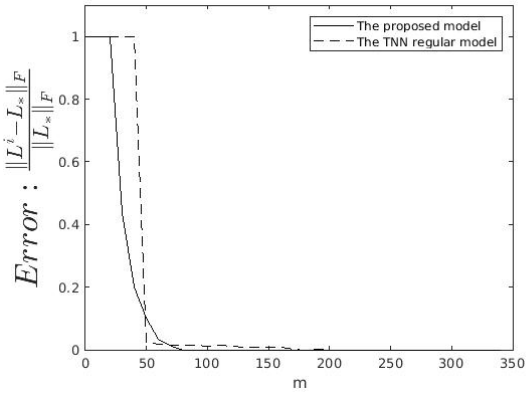
Figure 7.27: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{200\times150}$.

The average recovery probabilities for bi-cliques of sizes nm are presented in Figure 7.27a (respectively Figure 7.27b) correspond to the results in Figure 7.26a (respectively 7.26b).

We now perform numerical studies using larger values for N, M, n, and m. The relative error values we shown in Figure 7.28 and Figure 7.29 for problems corresponding to $M_b \in \mathbb{R}^{500\times350}$ and $M_b \in \mathbb{R}^{1000\times800}$, respectively.

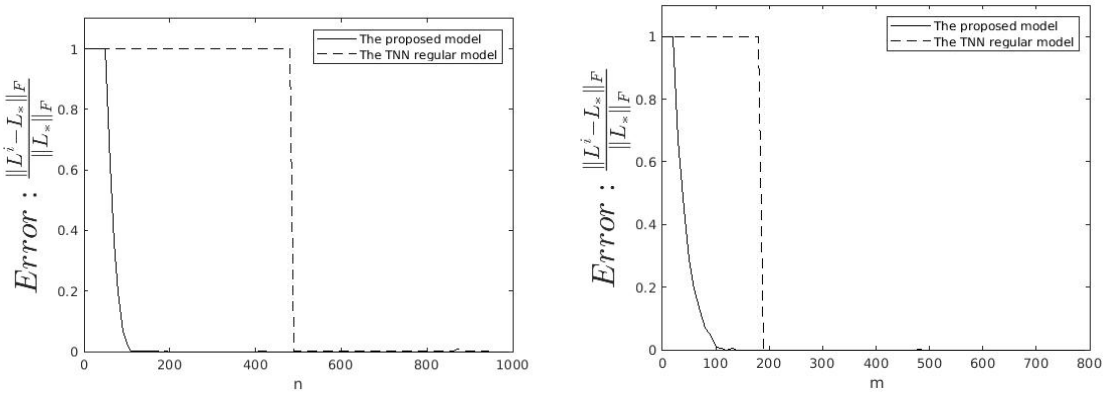


(a) $N = 500, M = 350$, and fixed $m = 200$



(b) $N = 500, M = 350$, and fixed $n = 250$

Figure 7.28: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{500\times350}$.

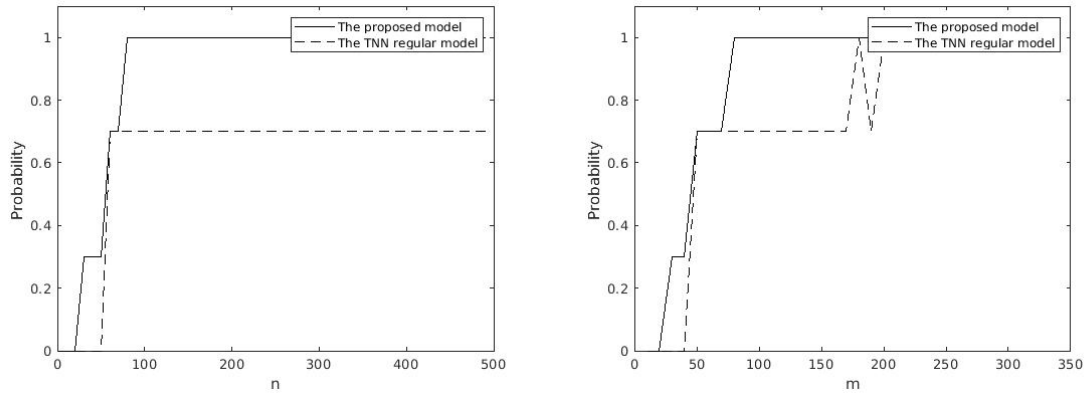


(a) $N = 1000, M = 800$, and fixed $m = 150$ **(b)** $N = 1000, M = 850$, and fixed $n = 600$

Figure 7.29: The average recovery errors for bi-clique of size nm, $M_b \in \mathbb{R}^{1000\times 800}$.

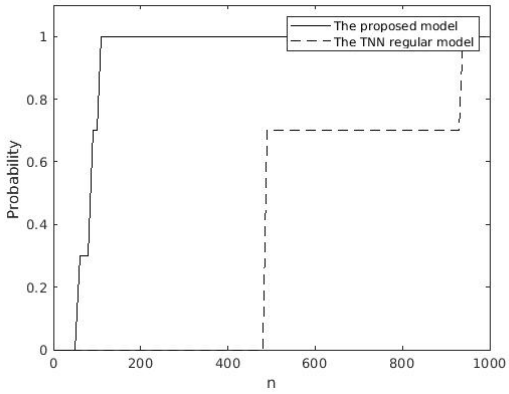
Figures 7.28 and 7.29 show that while ADMM for the proposed model (5.19)-(5.20) produces small errors than the TNN-based regular model (5.16)-(5.18).

The average recovery probabilities for bi-cliques of size nm using $M_b \in \mathbb{R}^{500\times 350}$ and $M_b \in \mathbb{R}^{1000\times 800}$ are demonstrated in Figure 7.30 and 7.31, respectively. Figure 7.28a (respectively Figure 7.28b) corresponds to the results in Figure 7.30a (respectively 7.30b). Similarly, Figure 7.29a (respectively Figure 7.29b) corresponds to the results in Figure 7.31a (respectively 7.31b).

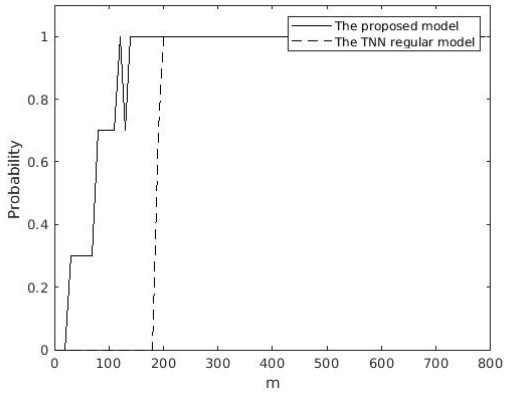


(a) $N = 500, M = 350$, and fixed $m = 200$ **(b)** $N = 500, M = 350$, and fixed $n = 250$

Figure 7.30: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{500\times 350}$.



(a) $N = 1000, M = 800$, and fixed $m = 150$

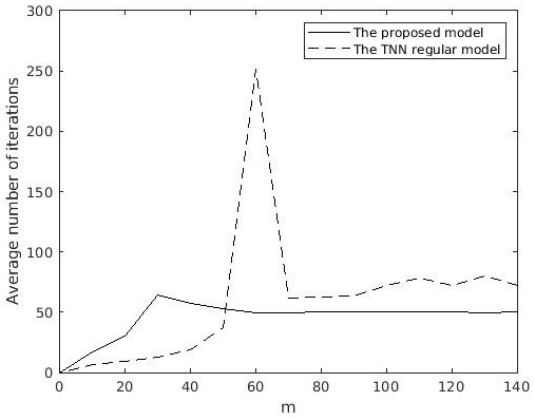


(b) $N = 1000, M = 800$, and fixed $n = 600$

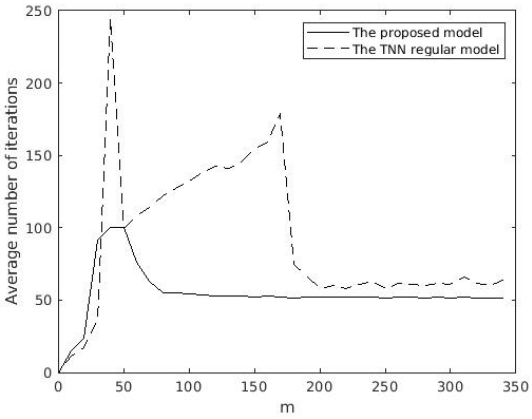
Figure 7.31: The average recovery probabilities for bi-clique of size nm, $M_b \in \mathbb{R}^{1000\times 800}$.

Results summarized in Figures 7.30 and 7.31 show the perfect recovery of planted bi-cliques with our algorithm with probability equals to one for most of the considered cases. On the other hand, the TNN-based regular model (5.16)-(5.18) fails to obtain perfect recovery for a number of problems as illustrated in Figures 7.30 and 7.31.

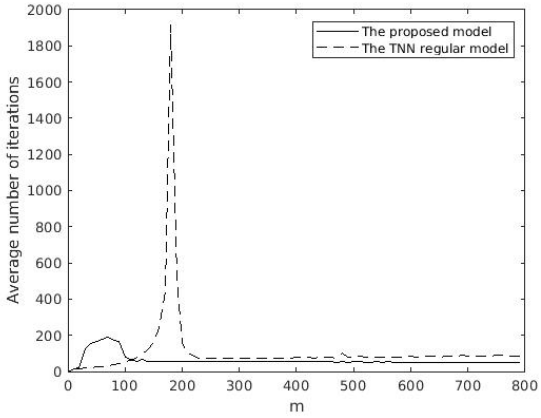
In Figure 7.32 we present the average number of iterations needed by **Algorithm** 6 for producing average error of 10^{-8} for the TNN-based model (5.19)-(5.20) and 10^{-5} for the TNN-based regular model (5.16)-(5.18).



(a) $N = 200,\ M = 150$, and fixed $n = 100$

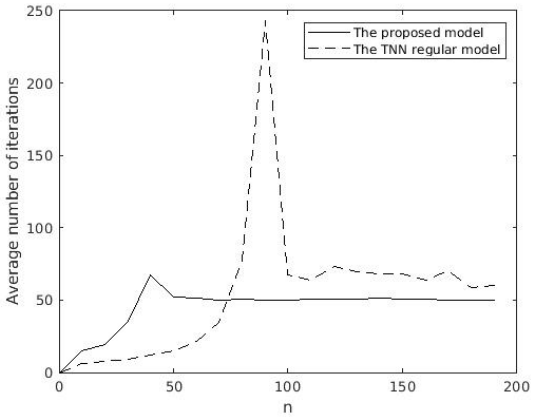


(b) $N = 500,\ M = 350$, and fixed $n = 250$

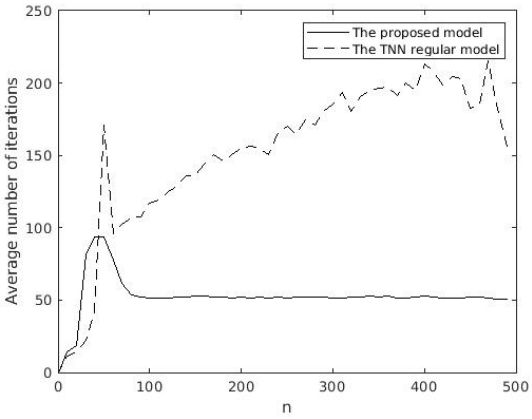


(c) $N = 1000,\ M = 800$, and fixed $n = 600$

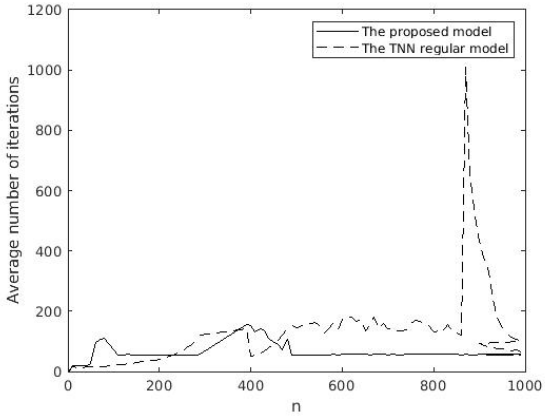
Figure 7.32: The average number of iterations for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$

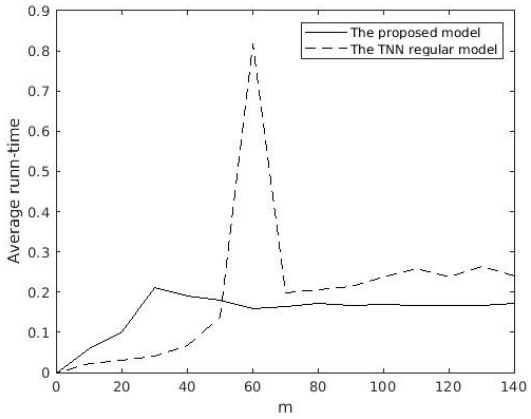


(c) $N = 1000$, $M = 800$, and fixed $m = 150$

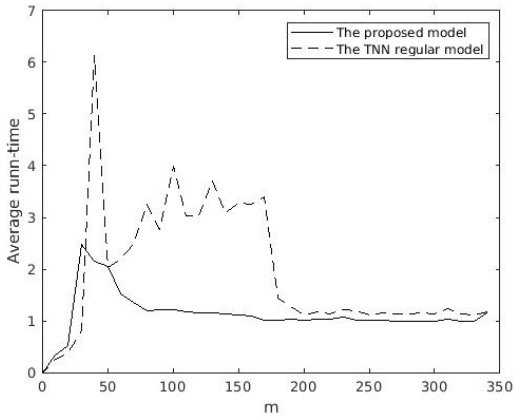
Figure 7.33: The average number of iterations for each problem with varied n.

Figures 7.32 and 7.33 show that our algorithm for the TNN-based model (5.19)-(5.20) produces better errors with a smaller number of iterations than the TNN-based regular model (5.16)-(5.18), on average.

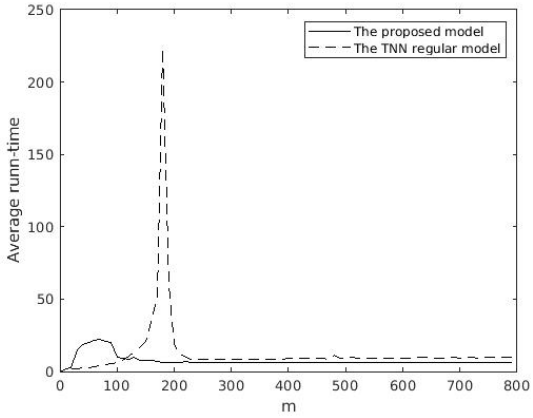
The total runtime and average runtime per iteration needed by our algorithm for the TNN-based model and the TNN-based regular model are summarized in the following figures.



(a) $N = 200,\ M = 150$, and fixed $n = 100$

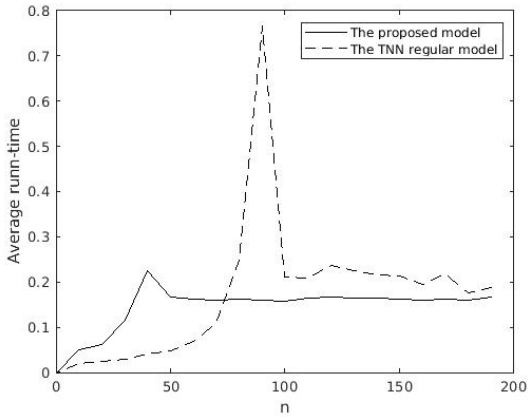


(b) $N = 500,\ M = 350$, and fixed $n = 250$

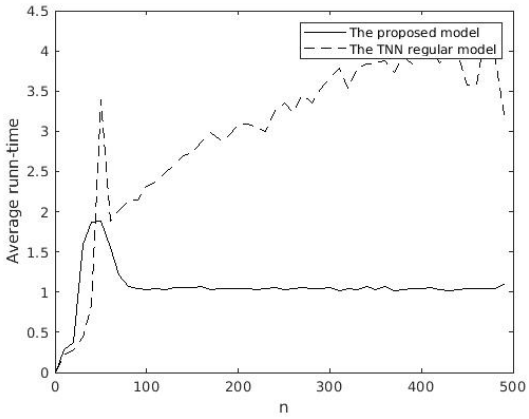


(c) $N = 1000,\ M = 800$, and fixed $n = 600$

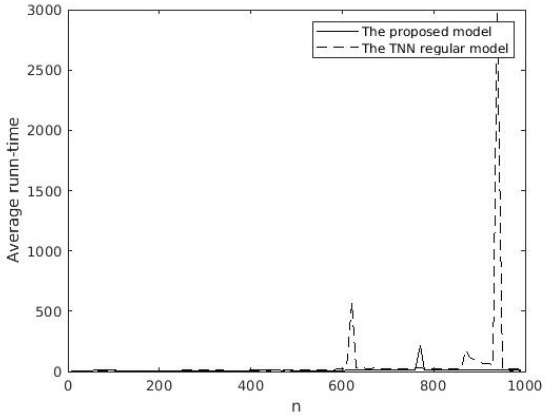
Figure 7.34: Average runtime for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$

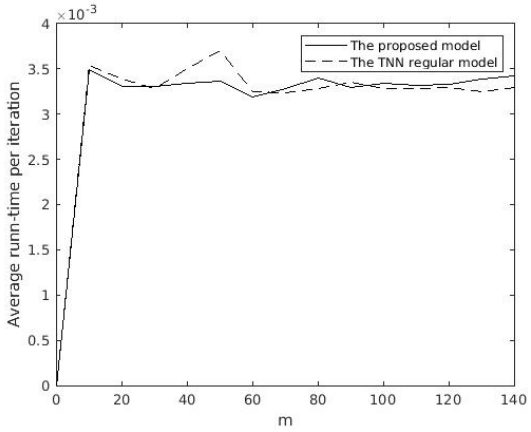


(c) $N = 1000$, $M = 800$, and fixed $m = 150$

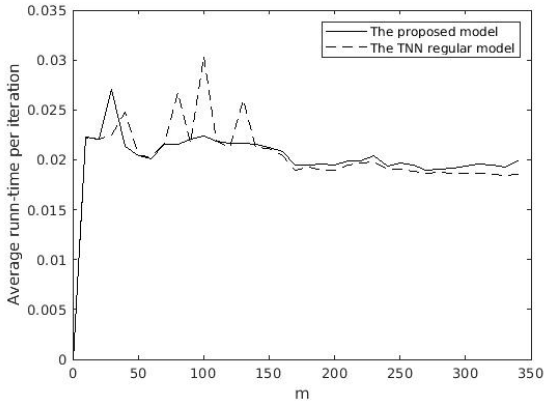
Figure 7.35: Average runtime for each problem with varied n.

Figures 7.34 and 7.35 show the average runtime for all (N, n) and (M, m) pairs for the problems considered, the averages are taken over 15 runs on each problem. These figures indicate that our algorithm for the TNN-based model (5.19)-(5.20) outperforms the TNN-based regular model (5.16)-(5.18) in finding the optimal solution for both cases when m and n being varied.

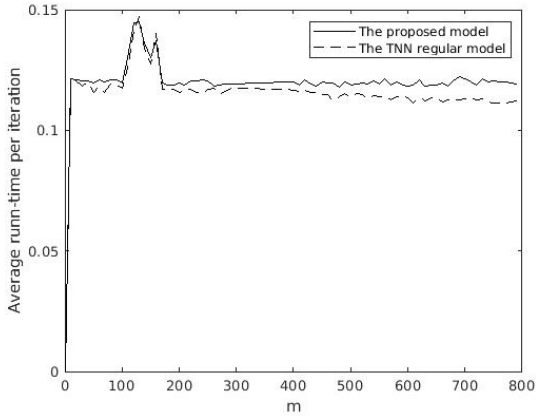
Figures 7.36 and 7.37 show the average runtime per iteration for the all the problems considered, where our algorithm for the TNN-based model performs slightly better, on average.



(a) $N = 200$, $M = 150$, and fixed $n = 100$

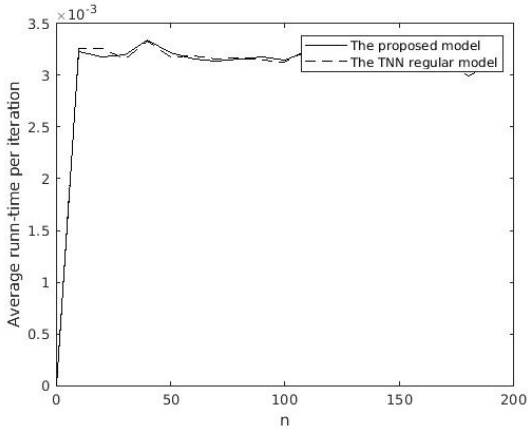


(b) $N = 500$, $M = 350$, and fixed $n = 250$

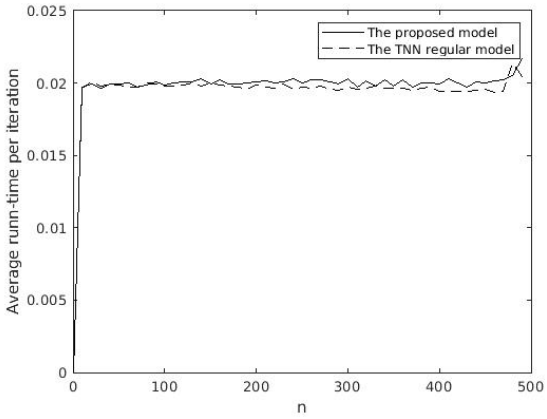


(c) $N = 1000$, $M = 800$, and fixed $n = 600$

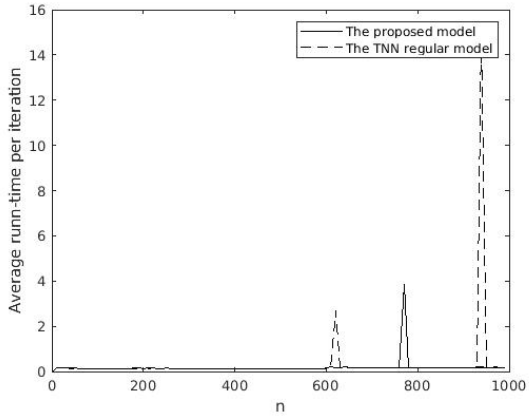
Figure 7.36: Average runtime per iteration for each problem with varied m.



(a) $N = 200$, $M = 150$, and fixed $m = 70$



(b) $N = 500$, $M = 350$, and fixed $m = 200$



(c) $N = 1000$, $M = 800$, and fixed $m = 150$

Figure 7.37: Average runtime per iteration for each problem with varied n.

7.5.2 Bi-cliques in Random Graphs via TNN

We have also performed experiments on random bipartite graphs where the edges are assigned with probability p. These results are presented in the table below, where nm is the size of the maximum bi-clique obtained by our algorithm, **Algorithm** 6. We have used the same stopping condition in (5.28), where $M = M_b$, but calculated the errors using the condition (7.3).

The solution (L, S) is initialized as in the case of planted bi-clique. A total of 48 runs have been performed by considering a number of (N, M) pairs. Results presented in each row of Table 7.10 are obtained for a single run. We have used two values for N, i.e., $N = 200, 500$ and varied M in

creating the pair (N, M) for this experiment. We have used $p = 0.85$ for generating $M_b \in \mathbb{R}^{200\times M}$ and $p = 0.88$ for $M_b \in \mathbb{R}^{500\times M}$ for each M considered. We have used higher values for p to ascertain that a reasonable size of maximum bi-clique, (n, m), is formed in each random graph. Results obtained show the perfect recovery of maximum bi-cliques having little or no errors.

Table 7.8: Random graph with fixed $N = 200$

(N, M)	(n, m)	CPU time	Error
$(200, 170)$	$(193, 169)$	0.6411	$6.6147e^{-07}$
	$(190, 166)$	0.6659	$4.2591e^{-07}$
	$(196, 161)$	0.6380	$7.7311e^{-07}$
	$(194, 168)$	0.6216	$3.8315e^{-07}$
$(200, 145)$	$(188, 143)$	0.4402	$1.7013e^{-06}$
	$(184, 141)$	0.4851	$2.6751e^{-07}$
	$(185.3449, 145)$	2.1788	0.7709
	$(191, 100)$	0.2316	$5.1018e^{-06}$
$(200, 100)$	$(194, 99)$	0.2328	$3.1379e^{-07}$
	$(191, 100)$	0.2504	$4.2093e^{-07}$
	$(188, 100)$	0.2647	$1.2640e^{-06}$
	$(95, 86)$	0.7811	$1.1369e^{-13}$
$(200, 77)$	$(198, 77)$	0.1835	$6.4592e^{-07}$
	$(190, 77)$	0.1435	$2.5082e^{-07}$
	$(191, 77)$	0.1659	$4.9869e^{-07}$
	$(189, 77)$	0.1686	$9.1106e^{-07}$
$(200, 45)$	$(192, 45)$	0.2353	$7.2352e^{-07}$
	$(187, 45)$	0.0895	$4.9194e^{-07}$
	$(194, 45)$	0.0802	$1.1894e^{-07}$
	$(193, 45)$	0.0816	$7.0117e^{-07}$

Table 7.9: Random graph with fixed $N = 500$

(N, M)	(n, m)	CPU time	Error
$(500, 350)$	$(361, 292)$	9.3826	$4.9620e^{-08}$
	$(379, 319)$	9.7824	$6.0027e^{-07}$
	$(342, 315)$	9.9261	$9.3536e^{-07}$
	$(366, 316)$	9.0191	$5.1989e^{-07}$
$(500, 265)$	$(391, 254)$	5.0065	$1.6554e^{-07}$
	$(380, 257)$	5.3655	$1.7086e^{-07}$
	$(405, 262)$	5.7250	$4.8119e^{-07}$
	$(396, 258)$	4.9212	$5.1253e^{-07}$
$(500, 155)$	$(487, 155)$	1.0469	$3.4728e^{-07}$
	$(493, 155)$	0.9647	$5.1299e^{-07}$
	$(489, 155)$	0.9743	$4.6342e^{-07}$
	$(492, 155)$	1.0018	$3.4146e^{-07}$
$(500, 100)$	$(498, 100)$	0.3388	$3.7339e^{-07}$
	$(496, 100)$	0.3596	$5.7898e^{-07}$
	$(496.4523, 100)$	1.4303	0.3734
	$(488, 100)$	0.3021	0.1226
$(500, 70)$	$(497, 70)$	0.1829	$1.4693e^{-07}$
	$(493, 70)$	0.1876	$6.6557e^{-07}$
	$(497, 70)$	0.1757	$2.2593e^{-07}$
	$(499, 70)$	0.1735	$2.9478e^{-07}$

Table 7.10: Maximum bi-clique in random graphs via TNN

Four random bipartite graphs are generated for each (N, M) pair. The corresponding recovered bi-clique sizes are shown under the column under (n, m).

Chapter 8

Conclusion

Considering the importance of the Maximum Clique (MC) problem and the Maximum Edge Bi-clique (MEB) problem, this thesis is dedicated to solving the Planted Clique Problem (PCP), the Planted Edge Bi-clique Problem (PEBP), and the Maximum Clique Problem (MCP) and the Maximum Edge Bi-clique Problem (MEBP) in random graphs. We have suggested mathematical models and the corresponding ADMM (Alternating Direction Method of Multipliers) algorithms for the PCP, PEBP, MCP, and MEBP that differ from the known matrix decomposition model in that it produces naturally integer solution required. We could achieve this by approximating the ℓ_0-norm using a non-convex function, when deriving our mathematical models. It has been shown that the proposed ADMM algorithm converges to the optimal solution. We have also suggested a second matrix decomposition model using the truncated nuclear norm and the weighted ℓ_1-norm. For solving the problem, we have developed an ADMM-based two-step algorithm for solving the truncated nuclear norm based model.

For the nuclear norm based matrix decomposition model, we have established conditions that guarantee the uniqueness and the recovery of the solution based on the assumptions of specific standard identifiability conditions for the sparse and low-rank components. Moreover, we have derived a tight bound of the dual matrix that certifies the optimality conditions of our proposed model.

Moreover, our computational investigations suggest that our proposed ADMM algorithms, for both models, produce almost intensive results for the regularization parameter $\lambda = \frac{\alpha}{\sqrt{N}}$, where N denotes the number of vertices in the given graph and α is a parameter which can be estimated. Our approaches produce much superior solutions quality when compared to other known approaches. This has been possible due to the dynamic nature of our mathematical models. Our algorithms require no input from the user other than the adjacency matrix of the input graph. In addition, the algorithms are implemented at easy without needing any external solvers. Although the algo-

rithms have been proposed for the PCP and PEBP, it has been tested on the MCP and MEBP using random graphs with almost error-free results. We have also suggested a new expression for error calculations. In addition, we have tested our models for real-world graphs, with successful results.

Furthermore, we have compared the results of the densest sub-graph algorithm (and the regular matrix decomposition model) to our nuclear norm-based matrix decomposition proposed algorithm and have shown that the latter outperforms better. We have also compared the truncated nuclear norm-based matrix decomposition model to the truncated nuclear norm based regular model where the first outperforms the later. Our proposed algorithms completely recover the planted clique, bi-clique, and MC and the MEB in random and real-world graphs with high probability. To conclude, our proposed approaches can be applied to graph-based low-rank problems since our algorithms recover a 0-1 solution that is suitable for graph-based problems.

Below, we list a few research directions which we would investigate in future.

- Is it possible to achieve better exact recovery guarantees for the truncated nuclear norm based matrix decomposition model than the one we have proposed? This will be investigated.
- Apply our proposed approaches for the MCP and the MEBP to real-world applications, such as in the area of bioinformatics.
- We would like to develop theoretical results on the truncated nuclear norm based model.
- We would like to investigate why the algorithm for the optimization problem of the nuclear norm based model require number of iterations less than the truncated nuclear norm model.

Bibliography

[1] Achlioptas, D. (2003). Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687.

[2] Alexe, G., Alexe, S., Crama, Y., Foldes, S., Hammer, P. L., and Simeone, B. (2004). Consensus algorithms for the generation of all maximal bicliques. *Discrete Applied Mathematics*, 145(1):11–21.

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

[4] Ambühl, C., Mastrolilli, M., and Svensson, O. (2011). Inapproximability results for maximum edge biclique, minimum linear arrangement, and sparsest cut. *SIAM Journal on Computing*, 40(2):567–596.

[5] Ames, B. (2011). *Convex relaxation for the planted clique, biclique, and clustering problems*. PhD thesis, University of Waterloo.

[6] Ames, B. P. (2015). Guaranteed recovery of planted cliques and dense subgraphs by convex relaxation. *Journal of Optimization Theory and Applications*, 167(2):653–675.

[7] Ames, B. P. and Vavasis, S. A. (2011). Nuclear norm minimization for the planted clique and biclique problems. *Mathematical Programming*, 129(1):69–89.

[8] Amin, A. and Hakimi, S. (1972). Upper bounds on the order of a clique of a graph. *SIAM Journal on Applied Mathematics*, 22(4):569–573.

[9] Andrade, D. V., Resende, M. G., and Werneck, R. F. (2012). Fast local search for the maximum independent set problem. *Journal of Heuristics*, 18(4):525–547.

[10] Arias-Castro, E. and Verzelen, N. (2014). Community detection in dense random networks. *The Annals of Statistics*, 42(3):940–969.

[11] Babel, L. and Tinhofer, G. (1990). A branch and bound algorithm for the maximum clique problem. *Zeitschrift für Operations Research*, 34(3):207–217.

[12] Battiti, R. and Protasi, M. (2001). Reactive local search for the maximum clique problem 1. *Algorithmica*, 29(4):610–637.

[13] Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

[14] Beezer, R. A. (2015). *A first course in linear algebra*. Independent.

[15] Belachew, M. T. and Gillis, N. (2017). Solving the maximum clique problem with symmetric rank-one non-negative matrix approximation. *Journal of Optimization Theory and Applications*, 173(1):279–296.

[16] Benlic, U. and Hao, J.-K. (2013). Breakout local search for maximum clique problems. *Computers & Operations Research*, 40(1):192–206.

[17] Blum, C., Djukanovic, M., Santini, A., Jiang, H., Li, C.-M., Manyà, F., and Raidl, G. R. (2021). Solving longest common subsequence problems via a transformation to the maximum clique problem. *Computers & Operations Research*, 125:105089.

[18] Bombina, P. and Ames, B. (2020). Convex optimization for the densest subgraph and densest submatrix problems. In *SN Operations Research Forum*, volume 1, pages 1–24. Springer.

[19] Bomze, I. M., Budinich, M., Pardalos, P. M., and Pelillo, M. (1999). The maximum clique problem. In *Handbook of Combinatorial Optimization*, pages 1–74. Springer.

[20] Bomze, I. M., Frommlet, F., and Locatelli, M. (2010). Copositivity cuts for improving sdp bounds on the clique number. *Mathematical Programming*, 124(1):13–32.

[21] Borwein, J. and Lewis, A. S. (2010). *Convex analysis and nonlinear optimization: theory and examples*. Springer Science & Business Media.

[22] Boyd, S., Boyd, S. P., and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

[23] Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122.

[24] Bruckstein, A. M., Donoho, D. L., and Elad, M. (2009). From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81.

[25] Budinich, M. and Budinich, P. (2006). A spinorial formulation of the maximum clique problem of a graph. *Journal of Mathematical Physics*, 47(4):043502.

[26] Bulo, S. R. and Pelillo, M. (2009). New bounds on the clique number of graphs based on spectral hypergraph theory. In *International Conference on Learning and Intelligent Optimization*, pages 45–58. Springer.

[27] Cai, J.-F., Candès, E. J., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982.

[28] Cai, J.-F. and Osher, S. (2013). Fast singular value thresholding without singular value decomposition. *Methods and Applications of Analysis*, 20(4):335–352.

[29] Cai, S. (2015). Balance between complexity and quality: Local search for minimum vertex cover in massive graphs. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[30] Cai, S., Su, K., and Sattar, A. (2011). Local search with edge weighting and configuration checking heuristics for minimum vertex cover. *Artificial Intelligence*, 175(9-10):1672–1696.

[31] Cai, T., Cai, T. T., and Zhang, A. (2016). Structured matrix completion with applications to genomic data integration. *Journal of the American Statistical Association*, 111(514):621–633.

[32] Candes, E., Demanet, L., Donoho, D., and Ying, L. (2006). Fast discrete curvelet transforms. *Multiscale Modeling & Simulation*, 5(3):861–899.

[33] Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37.

[34] Candes, E. J. and Plan, Y. (2010). Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936.

[35] Candès, E. J. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717.

[36] Candès, E. J., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509.

[37] Candes, E. J., Romberg, J. K., and Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223.

[38] Candes, E. J. and Tao, T. (2005). Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215.

[39] Candès, E. J. and Tao, T. (2010). The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080.

[40] Candes, E. J., Wakin, M. B., and Boyd, S. P. (2008). Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905.

[41] Cao, F., Chen, J., Ye, H., Zhao, J., and Zhou, Z. (2017). Recovering low-rank and sparse matrix based on the truncated nuclear norm. *Neural Networks*, 85:10–20.

[42] Carraghan, R. and Pardalos, P. M. (1990). An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9(6):375–382.

[43] Chandraker, M., Agarwal, S., Kahl, F., Nistér, D., and Kriegman, D. (2007). Autocalibration via rank-constrained estimation of the absolute quadric. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[44] Chandrasekaran, V., Sanghavi, S., Parrilo, P. A., and Willsky, A. S. (2009). Sparse and low-rank matrix decompositions. *IFAC Proceedings Volumes*, 42(10):1493–1498.

[45] Chandrasekaran, V., Sanghavi, S., Parrilo, P. A., and Willsky, A. S. (2011). Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596.

[46] Chen, G. and Teboulle, M. (1994). A proximal-based decomposition method for convex minimization problems. *Mathematical Programming*, 64(1):81–101.

[47] Chen, X. and Zhou, W. (2010). Convergence of reweighted l1 minimization algorithms and unique solution of truncated lp minimization. *Department of Applied Mathematics, The Hong Kong Polytechnic University*.

[48] Cramton, P., Shoham, Y., Steinberg, R., et al. (2004). Combinatorial auctions. Technical report, University of Maryland, Department of Economics-Peter Cramton.

[49] Dasgupta, S. and Gupta, A. (2003). An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65.

[50] Daubechies, I., DeVore, R., Fornasier, M., and Güntürk, C. S. (2010). Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 63(1):1–38.

[51] De Klerk, E. and Pasechnik, D. V. (2002). Approximation of the stability number of a graph via copositive programming. *SIAM Journal on Optimization*, 12(4):875–892.

[52] Delgado, R. A., Agüero, J. C., and Goodwin, G. C. (2014). A rank-constrained optimization approach: Application to factor analysis. *IFAC Proceedings Volumes*, 47(3):10373–10378.

[53] Demmel, J. (2000). Singular value decomposition. In *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, pages 135–147. SIAM.

[54] Dias, V. M., de Figueiredo, C. M., and Szwarcfiter, J. L. (2007). On the generation of bicliques of a graph. *Discrete Applied Mathematics*, 155(14):1826–1832.

[55] Dong, J., Xue, Z., Guan, J., Han, Z.-F., and Wang, W. (2018). Low rank matrix completion using truncated nuclear norm and sparse regularizer. *Signal Processing: Image Communication*, 68:76–87.

[56] Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306.

[57] Du, D. and Pardalos, P. M. (1998). *Handbook of combinatorial optimization*, volume 4. Springer Science & Business Media.

[58] Dvijotham, K. and Fazel, M. (2010). A nullspace analysis of the nuclear norm heuristic for rank minimization. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3586–3589. IEEE.

[59] Eckstein, J. and Fukushima, M. (1994). Some reformulations and applications of the alternating direction method of multipliers. In *Large Scale Optimization*, pages 115–134. Springer.

[60] Esser, E. (2009). Applications of lagrangian-based alternating direction methods and connections to split bregman. *CAM Report*, 9:31.

[61] Etzion, T. and Ostergard, P. R. (1998). Greedy and heuristic algorithms for codes and colorings. *IEEE Transactions on Information Theory*, 44(1):382–388.

[62] Fahle, T. (2002). Simple and fast: Improving a branch-and-bound algorithm for maximum clique. In *European Symposium on Algorithms*, pages 485–498. Springer.

[63] Fathi, M. and Bevrani, H. (2019). *Optimization in electrical engineering*. Springer.

[64] Fazel, M. (2002). *Matrix rank minimization with applications*. PhD thesis, Stanford University.

[65] Fazel, M., Hindi, H., and Boyd, S. (2004). Rank minimization and applications in system theory. In *Proceedings of the 2004 American Control Conference*, volume 4, pages 3273–3278. IEEE.

[66] Fazel, M., Hindi, H., and Boyd, S. P. (2001). A rank minimization heuristic with application to minimum order system approximation. In *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, volume 6, pages 4734–4739. IEEE.

[67] Fazel, M., Hindi, H., and Boyd, S. P. (2003). Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 3, pages 2156–2162. IEEE.

[68] Feige, U. and Krauthgamer, R. (2000). Finding and certifying a large hidden clique in a semirandom graph. *Random Structures & Algorithms*, 16(2):195–208.

[69] Feige, U. and Ron, D. (2010). Finding hidden cliques in linear time. In *Discrete Mathematics and Theoretical Computer Science*, pages 189–204. Discrete Mathematics and Theoretical Computer Science.

[70] Foucart, S. and Lai, M.-J. (2009). Sparsest solutions of underdetermined linear systems via ℓq-minimization for $0 < q \leq 1$. *Applied and Computational Harmonic Analysis*, 26(3):395–407.

[71] Fukushima, M. (1992). Application of the alternating direction method of multipliers to separable convex programming problems. *Computational Optimization and Applications*, 1(1):93–111.

[72] Gabay, D. and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40.

[73] Ganesh, A., Lin, Z., Wright, J., Wu, L., Chen, M., and Ma, Y. (2009). Fast algorithms for recovering a corrupted low-rank matrix. In *2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 213–216. IEEE.

[74] Gao, W., Goldfarb, D., and Curtis, F. E. (2020). Admm for multiaffine constrained optimization. *Optimization Methods and Software*, 35(2):257–303.

[75] Gillis, N. and Glineur, F. (2008). Nonnegative factorization and the maximum edge biclique problem. *ArXiv Preprint arXiv:0810.4225*.

[76] Gillis, N. and Glineur, F. (2014). A continuous characterization of the maximum-edge biclique problem. *Journal of Global Optimization*, 58(3):439–464.

[77] Glowinski, R. and Le Tallec, P. (1989). *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*. SIAM.

[78] Golub, G. H. and Van Loan, C. F. (1996). Matrix computations. johns hopkins studies in the mathematical sciences.

[79] Gómez Ravetti, M. and Moscato, P. (2008). Identification of a 5-protein biomarker molecular signature for predicting alzheimer's disease. *PloS One*, 3(9):e3111.

[80] Gross, D. (2011). Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566.

[81] Gross, D., Liu, Y.-K., Flammia, S. T., Becker, S., and Eisert, J. (2010). Quantum state tomography via compressed sensing. *Physical Review Letters*, 105(15):150401.

[82] Grosso, A., Locatelli, M., and Della Croce, F. (2004). Combining swaps and node weights in an adaptive greedy approach for the maximum clique problem. *Journal of Heuristics*, 10(2):135–152.

[83] Guturu, P. and Dantu, R. (2008). An impatient evolutionary algorithm with probabilistic tabu search for unified solution of some np-hard problems in graph and set theory via clique finding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(3):645–666.

[84] Han, L., Bi, S., and Pan, S. (2016). Two-stage convex relaxation approach to least squares loss constrained low-rank plus sparsity optimization problems. *Computational Optimization and Applications*, 64(1):119–148.

[85] Harary, F. and Ross, I. C. (1957). A procedure for clique detection using the group matrix. *Sociometry*, 20(3):205–215.

[86] Harvey, N. J. A. (2005). *Deterministic network coding by matrix completion*. PhD thesis, Massachusetts Institute of Technology.

[87] Hosseinian, S., Fontes, D. B., and Butenko, S. (2020). A lagrangian bound on the clique number and an exact algorithm for the maximum edge weight clique problem. *INFORMS Journal on Computing*, 32(3):747–762.

[88] Hu, Y., Zhang, D., Ye, J., Li, X., and He, X. (2012). Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2117–2130.

[89] Huang, S. and Wolkowicz, H. (2018). Low-rank matrix completion using nuclear norm minimization and facial reduction. *Journal of Global Optimization*, 72(1):5–26.

[90] Huang, Y.-M., Yan, H.-Y., Wen, Y.-W., and Yang, X. (2018). Rank minimization with applications to image noise removal. *Information Sciences*, 429:147–163.

[91] Hungerford, J. T. and Rinaldi, F. (2019). A general regularized continuous formulation for the maximum clique problem. *Mathematics of Operations Research*, 44(4):1161–1173.

[92] Ipsen, I. C. (2009). *Numerical matrix analysis: Linear systems and least squares*. SIAM.

[93] Jin, Y. and Hao, J.-K. (2015). General swap-based multiple neighborhood tabu search for the maximum independent set problem. *Engineering Applications of Artificial Intelligence*, 37:20–33.

[94] Johnson, D. (1993). The second dimacs implementation challenge. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*.

[95] Johnson, D. S. (1985). The np-completeness column: an ongoing guide. *Journal of Algorithms*, 6(3):434–451.

[96] Johnson, D. S. and Trick, M. A. (1996a). *Cliques, coloring, and satisfiability: second DIMACS implementation challenge, October 11-13, 1993*, volume 26. American Mathematical Soc.

[97] Johnson, D. S. and Trick, M. A. (1996b). Introduction to the second dimacs challenge: Cliques, coloring, and satisfiability. *Cliques, coloring, and satisfiability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26:1–7.

[98] Kang, Z., Peng, C., Cheng, J., and Cheng, Q. (2015a). Logdet rank minimization with application to subspace clustering. *Computational Intelligence and Neuroscience*, 2015.

[99] Kang, Z., Peng, C., and Cheng, Q. (2015b). Robust pca via nonconvex rank approximation. In *2015 IEEE International Conference on Data Mining*, pages 211–220. IEEE.

[100] Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer.

[101] Katayama, K., Hamamoto, A., and Narihisa, H. (2004). Solving the maximum clique problem by k-opt local search. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 1021–1025.

[102] Katayama, K., Hamamoto, A., and Narihisa, H. (2005). An effective local search for the maximum clique problem. *Information Processing Letters*, 95(5):503–511.

[103] Keshavan, R. H., Montanari, A., and Oh, S. (2010). Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998.

[104] Kloks, T. and Kratsch, D. (1995). Computing a perfect edge without vertex elimination ordering of a chordal bipartite graph. *Information Processing Letters*, 55(1):11–16.

[105] Knuth, D. E. (1993). The sandwich theorem. *ArXiv Preprint math/9312214*.

[106] Koetter, R. and Médard, M. (2003). An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795.

[107] Konc, J. and Janezic, D. (2007). An improved branch and bound algorithm for the maximum clique problem. *Proteins*, 4(5).

[108] Kontogiorgis, S. and Meyer, R. R. (1998). A variable-penalty alternating directions method for convex optimization. *Mathematical Programming*, 83(1):29–53.

[109] Kuhn, H. and Tucker, A. (1951). Nonlinear programming in proceedings of 2nd berkeley symposium (pp. 481–492). *Berkeley: University of California Press. [Google Scholar]*.

[110] Kumlander, D. (2005). Problems of optimization: an exact algorithm for finding a maximum clique optimized for dense graphs. In *Proceedings-Estonian Academy of Sciences Physics Mathematics*, volume 54, page 79. Estonian Academy Publishers; 1999.

[111] Lai, M.-J. and Wang, J. (2011). An unconstrained ℓ_q minimization with $0 \leq q \leq 1$ for sparse solution of underdetermined linear systems. *SIAM Journal on Optimization*, 21(1):82–101.

[112] Lewis, A. S. (2003). The mathematics of eigenvalue optimization. *Mathematical Programming*, 97(1):155–176.

[113] Li, C.-M. and Quan, Z. (2010). An efficient branch-and-bound algorithm based on maxsat for the maximum clique problem. In *Twenty-fourth AAAI Conference on Artificial Intelligence*.

[114] Lin, Z., Chen, M., and Ma, Y. (2010). The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *ArXiv Preprint arXiv:1009.5055*.

[115] Liu, Y.-J., Sun, D., and Toh, K.-C. (2012). An implementable proximal point algorithmic framework for nuclear norm minimization. *Mathematical Programming*, 133(1):399–436.

[116] Liu, Z. and Vandenberghe, L. (2010). Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256.

[117] Lovász, L. (1979). On the shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7.

[118] Lu, C., Feng, J., Chen, Y., Liu, W., Lin, Z., and Yan, S. (2016). Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5249–5257.

[119] Lyu, B., Qin, L., Lin, X., Zhang, Y., Qian, Z., and Zhou, J. (2020). Maximum biclique search at billion scale. *Proceedings of the VLDB Endowment*.

[120] Magnússon, S., Weeraddana, P. C., Rabbat, M. G., and Fischione, C. (2015). On the convergence of alternating direction lagrangian methods for nonconvex structured optimization problems. *IEEE Transactions on Control of Network Systems*, 3(3):296–309.

[121] Malod-Dognin, N., Andonov, R., and Yanev, N. (2010). Maximum cliques in protein structure comparison. In *International Symposium on Experimental Algorithms*, pages 106–117. Springer.

[122] Manurangsi, P. (2017). Inapproximability of maximum edge biclique, maximum balanced biclique and minimum k-cut from the small set expansion hypothesis. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[123] Marx, D. (2004). Graph colouring problems and their applications in scheduling. *Periodica Polytechnica Electrical Engineering (Archives)*, 48(1-2):11–16.

[124] Maslov, E., Batsyn, M., and Pardalos, P. M. (2014). Speeding up branch and bound algorithms for solving the maximum clique problem. *Journal of Global Optimization*, 59(1):1–21.

[125] Meka, R., Jain, P., and Dhillon, I. S. (2009). Guaranteed rank minimization via singular value projection. *ArXiv Preprint arXiv:0909.5457*.

[126] Mohan, K. and Fazel, M. (2010). New restricted isometry results for noisy low-rank recovery. In *2010 IEEE International Symposium on Information Theory*, pages 1573–1577. IEEE.

[127] Mohri, M. and Talwalkar, A. (2011). Can matrix coherence be efficiently and accurately estimated? In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 534–542. JMLR Workshop and Conference Proceedings.

[128] Motzkin, T. S. and Straus, E. G. (1965). Maxima for graphs and a new proof of a theorem of turán. *Canadian Journal of Mathematics*, 17:533–540.

[129] Mu, Y., Dong, J., Yuan, X., and Yan, S. (2011). Accelerated low-rank visual recovery by random projection. In *CVPR 2011*, pages 2609–2616. IEEE.

[130] Nesterov, Y. (2013). Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161.

[131] Nussbaum, D., Pu, S., Sack, J.-R., Uno, T., and Zarrabi-Zadeh, H. (2012). Finding maximum edge bicliques in convex bipartite graphs. *Algorithmica*, 64(2):311–325.

[132] Omer, S. and Ali, M. (2021a). Large scale planted bi-clique problem via matrix decomposition. Unpublished paper.

[133] Omer, S. and Ali, M. (2021b). Large scale planted clique problem via matrix decomposition. Unpublished paper.

[134] Omer, S. and Ali, M. (In prep.). Large scale planted clique problem via matrix decomposition and truncated nuclear norm.

[135] Orsi, R., Helmke, U., and Moore, J. B. (2006). A newton-like method for solving rank constrained linear matrix inequalities. *Automatica*, 42(11):1875–1882.

[136] Östergård, P. R. (2002). A fast algorithm for the maximum clique problem. *Discrete Applied Mathematics*, 120(1-3):197–207.

[137] Otazo, R., Candes, E., and Sodickson, D. K. (2015). Low-rank plus sparse matrix decomposition for accelerated dynamic mri with separation of background and dynamic components. *Magnetic Resonance in Medicine*, 73(3):1125–1136.

[138] Oyelade, J., Isewon, I., Oladipupo, F., Aromolaran, O., Uwoghiren, E., Ameh, F., Achas, M., and Adebiyi, E. (2016). Clustering algorithms: their application to gene expression data. *Bioinformatics and Biology Insights*, 10:BBI–S38316.

[139] Oymak, S., Mohan, K., Fazel, M., and Hassibi, B. (2011). A simplified approach to recovery conditions for low rank matrices. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 2318–2322. IEEE.

[140] Papailiopoulos, D. S. and Dimakis, A. G. (2012). Interference alignment as a rank constrained rank minimization. *IEEE Transactions on Signal Processing*, 60(8):4278–4288.

[141] Pardalos, J. and Resende, M. (1999). On maximum clique problems in very large graphs. *DIMACS Series*, 50:119–130.

[142] Pardalos, P. M. and Phillips, A. (1990). A global optimization approach for solving the maximum clique problem. *International Journal of Computer Mathematics*, 33(3-4):209–216.

[143] Pardalos, P. M. and Rodgers, G. P. (1992). A branch and bound algorithm for the maximum clique problem. *Computers & Operations Research*, 19(5):363–375.

[144] Pardalos, P. M. and Rosen, J. B. (1987). *Constrained global optimization: algorithms and applications*, volume 268. Springer.

[145] Parikh, N. and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239.

[146] Pattabiraman, B., Patwary, M. M. A., Gebremedhin, A. H., Liao, W.-k., and Choudhary, A. (2015). Fast algorithms for the maximum clique problem on massive graphs with applications to overlapping community detection. *Internet Mathematics*, 11(4-5):421–448.

[147] Pattillo, J., Youssef, N., and Butenko, S. (2012). Clique relaxation models in social network analysis. In *Handbook of Optimization in Complex Networks*, pages 143–162. Springer.

[148] Peeters, R. (2003). The maximum edge biclique problem is np-complete. *Discrete Applied Mathematics*, 131(3):651–654.

[149] Pirim, H., Ekşioğlu, B., Perkins, A. D., and Yüceer, Ç. (2012). Clustering of high throughput gene expression data. *Computers & Operations Research*, 39(12):3046–3061.

[150] Pullan, W. (2006). Phased local search for the maximum clique problem. *Journal of Combinatorial Optimization*, 12(3):303–323.

[151] Qian, W. and Cao, F. (2019). Adaptive algorithms for low-rank and sparse matrix recovery with truncated nuclear norm. *International Journal of Machine Learning and Cybernetics*, 10(6):1341–1355.

[152] Rahmani, M. and Atia, G. K. (2017). High dimensional low rank plus sparse matrix decomposition. *IEEE Transactions on Signal Processing*, 65(8):2004–2019.

[153] Recht, B. (2011). A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(12).

[154] Recht, B., Fazel, M., and Parrilo, P. A. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501.

[155] Recht, B., Xu, W., and Hassibi, B. (2008). Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. In *2008 47th IEEE Conference on Decision and Control*, pages 3065–3070. IEEE.

[156] Recht, B., Xu, W., and Hassibi, B. (2011). Null space conditions and thresholds for rank minimization. *Mathematical Programming*, 127(1):175–202.

[157] Régin, J.-C. (2003). Using constraint programming to solve the maximum clique problem. In *International Conference on Principles and Practice of Constraint Programming*, pages 634–648. Springer.

[158] Rockafellar, R. T. (2015). *Convex analysis*. Princeton University Press.

[159] Rossi, R. A. and Ahmed, N. K. (2015). The network data repository with interactive graph analytics and visualization. In *Twenty-ninth AAAI Conference on Artificial Intelligence*.

[160] San Segundo, P., Matia, F., Rodriguez-Losada, D., and Hernando, M. (2013). An improved bit parallel exact maximum clique algorithm. *Optimization Letters*, 7(3):467–479.

[161] Sciacca, L. J. and Evans, R. J. (1995). Multidimensional inverse problems in ultrasonic imaging. In *Control and Dynamic Systems*, volume 69, pages 1–48. Elsevier.

[162] Shahinpour, S. (2013). *Optimization-Based Network Analysis with Applications in Clustering and Data Mining*. PhD thesis.

[163] Shahinpour, S., Shirvani, S., Ertem, Z., and Butenko, S. (2017). Scale reduction techniques for computing maximum induced bicliques. *Algorithms*, 10(4):113.

[164] Soleimani-Pouri, M., Rezvanian, A., and Meybodi, M. R. (2014). An ant based particle swarm optimization algorithm for maximum clique problem in social networks. In *State of the Art Applications of Social Network Analysis*, pages 295–304. Springer.

[165] Solnon, C. and Fenet, S. (2004). Investigating aco capabilities for solving the maximum clique problem. *Rapport de recherché RR-LIRIS-2004-021, Soumis á Journal of Heuristics*.

[166] Sözdinler, M. and Özturan, C. (2018). Finding maximum edge biclique in bipartite networks by integer programming. In *2018 IEEE International Conference on Computational Science and Engineering (CSE)*, pages 132–137. IEEE.

[167] Starck, J.-L., Murtagh, F., and Fadili, J. M. (2010). *Sparse image and signal processing: wavelets, curvelets, morphological diversity*. Cambridge University Press.

[168] Strickland, D. M., Barnes, E., and Sokol, J. S. (2005). Optimal protein structure alignment using maximum cliques. *Operations Research*, 53(3):389–402.

[169] Sturm, J. F. (1999). Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653.

[170] Sun, C. and Dai, R. (2017). Rank-constrained optimization and its applications. *Automatica*, 82:128–136.

[171] Sun, W., Liu, C., Li, J., Lai, Y. M., and Li, W. (2014). Low-rank and sparse matrix decomposition-based anomaly detection for hyperspectral imagery. *Journal of Applied Remote Sensing*, 8(1):083641.

[172] Suyudi, M., Mamat, M., Bon, A., et al. (2018). Branch and bound algorithm for finding the maximum clique problem.

[173] Tao, M. and Yuan, X. (2011). Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization*, 21(1):57–81.

[174] Tayachi, D. and Khemiri, M. (2018). Solving the maximum clique problem using a hybrid particle swarm optimization algorithm. *International Journal of Operations Research and Information Systems (IJORIS)*, 9(4):21–35.

[175] Thai, M. T. and Pardalos, P. M. (2011). *Handbook of optimization in complex networks: theory and applications*, volume 57. Springer Science & Business Media.

[176] Thompson, R. (1978). Matrix type metric inequalities. *Linear and Multilinear Algebra*, 5(4):303–319.

[177] Toh, K.-C., Todd, M. J., and Tütüncü, R. H. (1999). Sdpt3—a matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1-4):545–581.

[178] Toh, K.-C. and Yun, S. (2010). An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 6(615-640):15.

[179] Tomita, E. and Kameda, T. (2007). An efficient branch-and-bound algorithm for finding a maximum clique with computational experiments. *Journal of Global Optimization*, 37(1):95–111.

[180] Tomita, E. and Seki, T. (2003). An efficient branch-and-bound algorithm for finding a maximum clique. In *International Conference on Discrete Mathematics and Theoretical Computer Science*, pages 278–289. Springer.

[181] Tomita, E., Sutani, Y., Higashi, T., Takahashi, S., and Wakatsuki, M. (2010). A simple and faster branch-and-bound algorithm for finding a maximum clique. In *International Workshop on Algorithms and Computation*, pages 191–203. Springer.

[182] Tropp, J. A. (2015). An introduction to matrix concentration inequalities. *ArXiv Preprint arXiv:1501.01571*.

[183] Tseng, P. (1997). Alternating projection-proximal methods for convex programming and variational inequalities. *SIAM Journal on Optimization*, 7(4):951–965.

[184] Tsourakakis, C., Bonchi, F., Gionis, A., Gullo, F., and Tsiarli, M. (2013). Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 104–112.

[185] Vandenberghe, L. and Boyd, S. (1996). Semidefinite programming. *SIAM Review*, 38(1):49–95.

[186] Vershynin, R. (2010). Introduction to the non-asymptotic analysis of random matrices. *ArXiv Preprint arXiv:1011.3027*.

[187] Watson, G. A. (1992). Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications*, 170:33–45.

[188] Wen, X., Chen, W.-N., Lin, Y., Gu, T., Zhang, H., Li, Y., Yin, Y., and Zhang, J. (2016). A maximal clique based multiobjective evolutionary algorithm for overlapping community detection. *IEEE Transactions on Evolutionary Computation*, 21(3):363–377.

[189] Wright, J., Ganesh, A., Rao, S. R., Peng, Y., and Ma, Y. (2009). Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *NIPS*, volume 58, pages 289–298.

[190] Wu, Q. (2013). *The maximum clique problems with applications to graph coloring*. PhD thesis, Université d'Angers.

[191] Wu, Q. and Hao, J.-K. (2012). Coloring large graphs based on independent set extraction. *Computers & Operations Research*, 39(2):283–290.

[192] Wu, Q. and Hao, J.-K. (2013). An adaptive multistart tabu search approach to solve the maximum clique problem. *Journal of Combinatorial Optimization*, 26(1):86–108.

[193] Xue, Z., Dong, J., Zhao, Y., Liu, C., and Chellali, R. (2019). Low-rank and sparse matrix decomposition via the truncated nuclear norm and a sparse regularizer. *The Visual Computer*, 35(11):1549–1566.

[194] Yannakakis, M. (1978). Node-and edge-deletion np-complete problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, pages 253–264.

[195] Yuan, X. and Yang, J. (2009). Sparse and low-rank matrix decomposition via alternating direction methods. *Preprint*, 12(2).

[196] Zhang, D., Hu, Y., Ye, J., Li, X., and He, X. (2012). Matrix completion by truncated nuclear norm regularization. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2192–2199. IEEE.

[197] Zhang, Y., Guo, J., Zhao, J., and Wang, B. (2016). Robust principal component analysis via truncated nuclear norm minimization. *Journal of Shanghai Jiaotong University (Science)*, 21(5):576–583.

[198] Zhao, X.-Y., Sun, D., and Toh, K.-C. (2010). A newton-cg augmented lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4):1737–1765.

[199] Zheng, C.-H., Yuan, L., Sha, W., and Sun, Z.-L. (2014). Gene differential coexpression analysis based on biweight correlation and maximum clique. In *BMC Bioinformatics*, volume 15, pages 1–7. Springer.

[200] Zhou, G. (2015). *Rank-constrained optimization: A Riemannian manifold approach*. PhD thesis, The Florida State University.