Rank-Constrained Optimization: Algorithms and Applications

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Chuangchuang Sun, BS

Graduate Program in Aeronautical and Astronautical Engineering

The Ohio State University

2018

Dissertation Committee:

Ran Dai, Advisor

Mrinal Kumar

Manoj Srinivasan

Wei Zhang

Abstract

Matrix rank related optimization problems comprise rank-constrained optimization problems (RCOPs) and rank minimization problems, which involve the matrix rank function in the objective or the constraints. General RCOPs are defined to optimize a convex objective function subject to a set of convex constraints and rank constraints on unknown rectangular matrices. RCOPs have received extensive attention due to their wide applications in signal processing, model reduction, and system identification, just to name a few. Noteworthily, The general/nonconvex quadratically constrained quadratic programming (QCQP) problem with wide applications is a special category of RCOP. On the other hand, when the rank function appears in the objective of an optimization problem, it turns out to be a rank minimization problem (RMP), classified as another category of nonconvex optimization. Applications of RMPs have been found in a variety of areas, such as matrix completion, control system analysis and design, and machine learning.

At first, RMP and RCOPs are not isolated. Though we can transform a RMP into a RCOP by introducing a slack variable, the reformulated RCOP, however, has an unknown upper bound for the rank function. That is a big shortcoming. Provided that a given matrix upper bounded is a prerequisite for most of the algorithms when solving RCOPs, we first fill this gap by demonstrating that RMPs can be equivalently transformed into RCOPs by introducing a quadratic matrix equality constraint. Furthermore, the wide application of RMPs and RCOPs attract extensive studies aiming at developing efficient optimization

algorithms to solve the challenging matrix-related optimization problems. Motivated by the limitations of existing algorithms, we aim to develop effective and efficient algorithms to solve rank-related optimization problems. Two classes of algorithms and their variants are proposed here.

The first algorithm, based on the alternating minimization, is named iterative rank minimization (IRM). The IRM method, with each sequential problem formulated as a convex optimization problem, aims to solve general RCOPs, where the constrained rank could be any assigned integer number. Although IRM is primarily designed for RCOPs with rank constraints on positive semidefinite matrices, a semidefinite embedding lemma is introduced to extend IRM to RCOPs with rank constraints on general rectangular matrices. Moreover, The proposed IRM method is applicable to RCOPs with rank inequalities constrained by upper or lower bounds, as well as rank equality constraints. Convergence of IRM is proved via the duality theory and the Karush-KuhnTucker conditions. Furthermore, the conversion from RMPs to RCOPs and the proposed IRM method are applied to solve cardinality minimization problems and cardinality-constrained optimization problems, which are handled as special cases of RMPs and RCOPs, respectively. To verify the effectiveness and improved performance of the proposed IRM method, representative applications, including matrix completion, system identification, output feedback stabilization, structured H_2 controller design, and cardinality minimization problems, UAV path planning, network identification, are presented with comparative results. Moreover, for sparse QCQPs, a variant of IRM is developed to enhance the efficiency by exploiting the sparsity.

The second algorithm, based on the framework of alternating direction method of multipliers (ADMM), aims to get simple subproblem solutions, even in closed forms. We reformulate the rank constraint by matrix factorization and thus lead to the Burer-Monteiro

form to achieve simpler subproblems. Despite the nonconvex constraint, we prove the ADMM iterations converge to a stationary point in two types of formulations under mild assumptions. Additionally, recent work suggests that in this latter form, when the matrix factors are wide enough, local optimum with high probability is also the global optimum. To demonstrate the scalability of our algorithm, we include results for MAX-CUT, community detection, and image segmentation benchmark and simulated examples. A variant of our algorithm reformulates the original RCOP using an approximate representation. The approximate formulation and the introduced bilinear constraint make each subproblem of ADMM more computationally-efficient. In particular cases, the subproblems can even be solved in a closed form. We then prove the global convergence of our proposed algorithm to a local minimizer of the approximate problem. Another variant focuses on solving QCQPs with both equality and inequality constraints. Based on inexact augmented Lagrangian method, the subproblem also admits a simple closed-form solution. By exploiting the structure of specific problems, such as network module detection, the computational efficiency can be further improved.

Moreover, we also examine the simultaneous design of the network topology and the corresponding edge weights in presence of a cardinality constraint on the edge set. Network properties of interest for this design problem lead to optimization formulations with convex objectives, convex constraint sets, and cardinality constraints. This class of problems is referred to as cardinality-constrained optimization problem (CCOP); the cardinality constraint generally makes CCOPs NP-hard. In this work, a customized ADMM algorithm aiming to improve the scalability of the solution strategy for large-scale CCOPs is proposed. This algorithm utilizes the special structure of the problem formulation to obtain closed-form solutions for each iterative step of the corresponding ADMM updates. We also provide the

convergence proof of the proposed customized ADMM to a stationary point under certain conditions.

In conclusion, this dissertation developed the algorithmic frameworks to solve RCOPs and RMPs with convergence analysis. Comparative simulation results with the state-of-art algorithms are provided to validate the efficacy and effectiveness of proposed algorithms.

Dedicated to my parents

Acknowledgments

I would like to appreciate whoever helped me in this incredible journey. First and foremost, I would like to thank my advisor Dr. Ran Dai, for her support, encouragement throughout all this PhD program. Her technical and editorial suggestions are essential to this dissertation and I learnt a lot from her on academic research. Also, my thanks goes to my committee members, Dr. Mrinal Kumar, Dr. Manoj Srinivasan and Dr. Wei Zhang as well as my previous committee members back at Iowa State. Their insightful questions greatly help me to improve the quality of this dissertation. Moreover, I owe a thanks to the administrative staffs both at Ohio State and Iowa State, who have always been helpful in the administrative processes.

The friendship with the colleagues in our lab is also much appreciated. It has been a pleasure to work with Yue Zu, Changhuang Wan, Nathaniel Kingry and Yen-chen Liu. The meaningful discussions also contribute to this dissertation. Moreover, I also appreciate the friendship with the kind people during my stay at both Ames, IA and Columbus, OH. That has been a great memory.

Last but not the least, my deepest gratitude goes to my family, especially my parents Shikui Sun and Yufang Liu. Their unconditional love and support not only in this PhD program but ever since my birth, shape me the one who I am today. Surely this dissertation will not be possible without their support. Also, I wold like to thank my grandparents, my aunt, my sister and my brother, for everything.

Vita

August 2018 ..... PhD,
Aeronautical and Astronautical Engineering,
The Ohio State University, USA.

June 2013 ..... Bachelor of Science,
Beihang University.

Publications

Research Publications

Sun, Chuangchuang, and Ran Dai. "Topology Design and Identification for Dynamic Networks." Cooperative Control of Multi-Agent Systems: Theory and Applications (2017): 209.

Sun, Chuangchuang, Ran Dai and Mehran Mesbahi. "Weighted Network Design with Cardinality Constraints via Alternating Direction Method of Multipliers." IEEE Transactions on Control of Network Systems, 2018, DOI: 10.1109/TCNS.2018.2789726

Sun, Chuangchuang, and Ran Dai. "Rank-constrained optimization and its applications." Automatica 82 (2017): 128-136.

Sun, Chuangchuang, Yen-chen Liu and Ran Dai. "Two Approaches for Path Planning of Unmanned Aerial Vehicles with Avoidance Zones." Journal of Guidance, Control, and Dynamics (2017) 1:8.

Ran Dai and Chuangchuang Sun, “Path Planning of Spatial Rigid Motion with Constrained Attitude”, Journal of Guidance, Control, and Dynamics, Vol. 38, No. 8, 2015, pp. 1356-1365.

Wang, Dangxiao, Jing Guo, Chuangchuang Sun, Mu Xu, and Yuru Zhang. "A Flexible Concept for Designing Multiaxis Force/Torque Sensors Using Force Closure Theorem." Instrumentation and Measurement, IEEE Transactions on62, no. 7 (2013): 1951-1959.

Du, Jianbin, and Chuangchuang Sun. "Reliability-based vibro-acoustic microstructural topology optimization." Structural and Multidisciplinary Optimization 55.4 (2017): 1195-1215

Wang, Xiaohui, Changhuang Wan, Chuangchuang Sun, Renwei Xia. "An optimization algorithm for multi-objective optimization problem by using envelope-dual method." Procedia Engineering 67 (2013): 457-466.

Sun, Chuangchuang, and Ran Dai. "An Efficient Module Detection Algorithm for Large Scale Complex Networks." American Control Conference (ACC), 2018. IEEE, 2018.

Sun, Chuangchuang, and Ran Dai. "A Customized ADMM for Rank-Constrained Optimization Problems with Approximate Formulations." Decision and Control (CDC), 2017 IEEE 56th Annual Conference on. IEEE, 2017.

Sun, Chuangchuang, and Ran Dai. "A decomposition method for nonconvex quadratically constrained quadratic programs." American Control Conference (ACC), 2017. IEEE, 2017

Sun, Chuangchuang, and Ran Dai. "An iterative approach to rank minimization problems." Decision and Control (CDC), 2015 IEEE 54th Annual Conference on. IEEE, 2015.

Sun, Chuangchuang, Ran Dai, and Ping Lu. "Solving polynomial optimal control problems via iterativeconvex optimization." AIAA Guidance, Navigation, and Control Conference. 2016.

Sun, Chuangchuang, and Ran Dai. "Spacecraft attitude control under constrained zones via quadratically constrained quadratic programming." AIAA Guidance, Navigation, and Control Conference. 2015.

Sun, Chuangchuang, and Ran Dai. "Identification of network topology via quadratic optimization." American Control Conference (ACC), 2015. IEEE, 2015.

Zu, Yue, Chuangchuang Sun, and Ran Dai. "Distributed estimation for spatial rigid motion based on dual quaternions." Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on. IEEE, 2014.

Du, Jianbin, and Chuangchuang Sun. "Reliability-based microstructural topology design with respect to vibro-acoustic criteria." 11th World Congress on Structural and Multidisciplinary Optimization, Sydney, Australia 2015.

Fields of Study

Major Field: Aeronautical and Astronautical Engineering

Studies in Optimization algorithms and applications: Ran Dai

Table of Contents

Page

List of Tables

List of Figures

Chapter 1: Introduction

1.1 General Rank-Constrained Optimization

Rank-constrained optimization problems (RCOPs) are to minimize a convex function subject to a convex set of constraints and rank constraints on unknown matrices. They have received extensive attention due to their wide applications in signal processing, model reduction, and system identification, just to name a few [109, 132, 133, 198]. Although some special RCOPs can be solved analytically [79, 136], they are NP-hard in most of the cases. Existing methods for RCOPs mainly focus on alternating projection based methods [49, 51, 81] and combined linearization and factorization algorithms [88, 146] with application to factor analysis, etc. However, these iterative approaches depend on the initial guess and fast convergence cannot be guaranteed. In addition, a Newton-like method [161] has been proposed to search for a feasible solution of RCOPs with application to a feedback stabilization problem. A Riemannian manifold optimization method [203] has been applied to solve large-scale Lyapunov matrix equations by finding a low-rank approximation. Also, [201] proposes to use the toolbox BARON to solve a RCOP. There are alternative approaches for solving specially formulated RCOPs. For example, a greedy convex smoothing algorithm has been designed to optimize a convex objective function subject to only one rank constraint [176].

1.2 Rank Minimization Problems

When the rank function in constraints of RCOPs appears as the objective of an optimization problem, it turns to be a rank minimization problem (RMP), classified as a category of nonconvex optimization. Applications of RMPs have been found in a variety of areas, such as matrix completion [36, 148, 169], control system analysis and design [57, 63, 64, 144, 145], and machine learning [142, 168]. The wide application of RMPs attracts extensive studies aiming at developing efficient optimization algorithms.

Due to the discontinuous and nonconvex nature of the rank function, most of the existing methods solve relaxed or simplified RMPs by introducing an approximate function, such as log-det or nuclear norm heuristic methods [62, 65]. The heuristic methods minimize a relaxed convex function instead of the exact rank function over a convex set, which is computationally favorable. They generally generate a solution with lower rank, even a minimum rank solution in special cases [169]. The relaxed formulation with convex objective and constraints does not require the initial guess and global optimality is guaranteed for the relaxed formulation. When the unknown matrix is constrained to be positive semidefinite, relaxation of RMPs using a trace function is equivalent to the relaxed formulation using a nuclear norm function based on the fact that the trace of a positive semidefinite matrix equals to its nuclear norm [145]. For cases when the unknown matrix is not positive semidefinite, work in [62] introduces a semidefinite embedding lemma to extend the trace heuristic method to general cases.

However, a relaxed function cannot represent the exact rank function and performance of the heuristic method is not guaranteed. Other heuristic methods, e.g., the iterative reweighted least square algorithm [148] which iteratively minimizes the reweighted Frobenius norm of the matrix, cannot guarantee the minimum rank solution either. The uncertainty of the

performances in heuristic methods stems from the fact that these methods are minimizing a relaxed function and generally there is a gap between the relaxed objective and the exact one. Other methods for RMPs include the alternating projections and its variations [81, 117, 161], linearization [58, 88], and augmented Lagrangian method [61]. These methods, similar to existing iterative methods for RCOPs, depend on initial guess, which generally leads to slow convergence to just a feasible solution.

After reviewing the literature, we come to a conclusion that more efficient approaches that are applicable for general ROCPs/RMPs with advantages in terms of convergence rate, robustness to initial guess, and performance of cost function, are required to solve RCOPs and RMPs. To our knowledge, there is few literature addresses equivalent conversion from RMPs into RCOPs [52, 182]. This paper describes a novel representation of RMPs in the form of RCOPs and proposes a uniform approach to both RCOPs and reformulated RMPs. Therefore, instead of solving two classes of nonconvex optimization problems separately, the uniform formulation and approach significantly reduces the required efforts for solving two types of challenging problems.

1.3 Quadratically Constrained Quadratic Programming

The general/noncovex quadratically constrained quadratic programming (QCQP) problem has recently attracted significant interests due to its wide applications. For example, any polynomial problems of optimizing a polynomial objective function while satisfying polynomial inequality constraints can be reformulated as general QCQP problems [31, 48]. In addition, we can find QCQP applications in the areas of maxcut problems [53], production planning [171], signal processing [128], sensor network localizations [20], and optimal power flow [100, 116], just to name a few.

Convexification and relaxation techniques have been commonly used when solving nonconvex optimization problems [3, 59, 153]. Efforts toward solving nonconvex QCQP problems have been pursued in two directions, obtaining a bound on the optimal value and finding a feasible solution. For simplicity, the QCQPs discussed below represent general/nonconvex QCQPs. Extensive relaxation methods have been investigated to obtain a bound on the optimal value of a QCQP. The linear relaxation approach introduces extra variables to transform the quadratic objective and constraints into bilinear terms, which is followed by linearization of the bilinears [5, 167]. The final linear formulation reaches a bound on the QCQP optimal value with fast convergence, but low accuracy. The semidefinite programming (SDP) relaxation introduces a rank one matrix to replace the quadratic objective and constraints with linear matrices equalities/inequalities. However, the nonlinear rank one constraint on the unknown matrix is substituted by semidefinite relaxation. In general, the SDP relaxation reaches a tighter bound on the optimal value than that obtained from linear relaxation [48]. A detailed discussion of various relaxation approaches and the comparison of their relative accuracy is provided in [10].

However, finding a bound on the optimal value of QCQP does not imply generating an optimal solution, not even a feasible one. One of the efforts for obtaining a feasible solution utilizes an iterative linearization approach to gradually improve the objective value [48]. However, this method does not provide any guarantee of convergence. Another approach is to generate randomized samples and solve the QCQP on average of the distribution. However, the randomization approach does not apply to problems with equality constraints and the optimality is not guaranteed. Branch and bound (BNB) method has been frequently utilized to search for the optimal solution of nonconvex problems [6,44,122]. Although BNB can lead to global optimal solution, the searching procedure is time consuming, especially

for large scale optimization problems. Recent work in [178] proposes that the structure of the QCQP problems can be changed based on graph theory to obtain a low-rank solution which greatly reduces the gap between the exact solution and the relaxed one. Furthermore, works in [115] generates a series of SDPs to solve polynomial optimization problems, which is applicable to small scale QCQPs.

After reviewing the literature, we come to a conclusion that a more efficient approach is required to solve QCQP problems. In our previous work of [42, 184], an iterative rank minimization (IRM) method has been proposed to solve homogeneous QCQPs. Inspired by the SDP relaxation, the IRM method focuses on finding the unknown rank one matrix by gradually minimizing the rank of the unknown matrix. This paper explores the problem to inhomogeneous QCQPs and focuses on proof of local convergence to a local optimum. Each iteration of IRM is formulated as a convex problem with semidefinite constraints.

This dissertation is based on my previous publications [47, 185–196, 218].

Chapter 2: Iterative Rank Minimization

2.1 Problem Formulation

2.1.1 Rank-Constrained Optimization Problems

A general RCOP to optimize a convex objective subject to a set of convex constraints and rank constraints can be formulated as follows

$$\begin{array}{ll} \min_X & f(X) \\ s.t. & X \in \mathscr{C}, \ \ \mathbf{rank}(X) \lesseqgtr r, \end{array} \tag{1.1}$$

where $f(X)$ is a convex function, $\mathscr{C}$ is a convex set, and $X \in \mathbb{R}^{m\times n}$ is a general rectangular matrices set. Without loss of generality, it is assumed that $m \leq n$. The sign $\lesseqgtr$ include all types of rank constraints, including upper and lower bounded rank inequality constraints and rank equality constraints. Although lower bounded rank inequality constraints and rank equality constraints do not have as many practical applications compared to the upper bounded rank inequality constraints, they are included here for completeness. Because the existing and proposed approaches for RCOPs require the to-be-determined matrix to be a positive semidefinite matrix, it is then necessary to convert the rank constraints on rectangular matrices into corresponding ones on positive semidefinite matrices.

Lemma 2.1.1. *(Lemma 1. in [63]) Let $X \in \mathbb{R}^{m\times n}$ be a given matrix, then* $\mathbf{rank}(X) \leq r$ *if and only if there exists matrices $Y = Y^T \in \mathbb{R}^{m\times m}$ and $Z = Z^T \in \mathbb{R}^{n\times n}$ such that*

$$\mathbf{rank}(Y)+\mathbf{rank}(Z)\leq 2r, \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq \boldsymbol{0}.$$

However, Lemma (2.1.1) is not applicable to lower bounded rank inequality constraints. As a result, a new lemma is introduced to extend the above semidefinite embedding lemma to all types of rank constraints. Before that, we first describe a proposition which is involved in proof of the new lemma.

Proposition 2.1.2. *$Z = X^T X$ is equivalent to*
$\mathbf{rank}\Big(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \Big) \leq m$, *where $Z \in \mathbb{S}^n$, $X \in \mathbb{R}^{m\times n}$, and $I_m \in \mathbb{R}^{m\times m}$ is an identify matrix.*

Proof. Given that the rank of a symmetric block matrix is equal to the rank of a diagonal block plus the rank of its Schur complement, we have the following relationship, $\mathbf{rank}\Big(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \Big) \leq m \Leftrightarrow \mathbf{rank}(I_m)+\mathbf{rank}(Z-X^T X)\leq m \Leftrightarrow m+\mathbf{rank}(Z-X^T X)\leq m \Leftrightarrow$ $\mathbf{rank}(Z-X^T X)=0 \Leftrightarrow Z=X^T X$. □

Remark: When $Z = X^T X$, it indicates that $\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \succeq \mathbf{0}$ holds for $I_m \succ \mathbf{0}$ and its Schur complement is a zero matrix.

Next, we give the extended semidefinite embedding lemma below.

Lemma 2.1.3. *Let $X \in \mathbb{R}^{m\times n}$ be a given matrix. Then $\mathbf{rank}(X) \leq r(=r, \geq r)$ if and only if there exists a matrix $Z \in \mathbb{S}^n$ such that*

$$\mathbf{rank}(Z) \leq r(=r, \geq r, resp.) \ \& \ \mathbf{rank}\Big(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \Big) \leq m.$$

Proof. From Proposition (2.1.2), $\mathbf{rank}\Big(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \Big) \leq m \Leftrightarrow Z = X^T X$. It is known that $\mathbf{rank}(Z) = \mathbf{rank}(X)$ when $Z = X^T X$. Hence the proof is complete. □

Remark: For an upper bounded rank inequality constraint, $\mathbf{rank}(X) \leq r$, Lemma 2.1.3 can be interpreted as a special case of Lemma 2.1.1 by setting Y in Lemma 2.1.1 be I_m and $\mathbf{rank}(Z) = \mathbf{rank}(X)$ and replacing $2r$ by $m+r$. Lemma 2.1.3 for this case transforms the rank inequality on rectangular matrix into two rank inequalities on semidefinite matrices while Lemma 2.1.1 equivalently transforms it into one rank inequality and one semidefinite constraint. For the case of rank equality constraint, $\mathbf{rank}(X) = r$, since the semidefinite constraint in Lemma 2.1.1 indicates that $\mathbf{rank}(Y) \geq \mathbf{rank}(X)$ and $\mathbf{rank}(Z) \geq \mathbf{rank}(X)$, Lemma 2.1.1 for this case leads to two rank constraints and one semidefinite constraint, expressed as

$$\mathbf{rank}(Y) = r,\ \mathbf{rank}(Z) = r,\ \text{ and } \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq \mathbf{0}. \tag{1.2}$$

The general semidefinite Lemma in 2.1.3 for this case has the same form of expressions compared to the case with upper bounded rank inequality. Additionally, for a lower bounded rank inequality constraint, $\mathbf{rank}(X) \geq r$, Lemma 2.1.1 is not applicable based on the fact that $\mathbf{rank}(Y) \geq \mathbf{rank}(X)$ and $\mathbf{rank}(Z) \geq \mathbf{rank}(X)$ always leads to $\mathbf{rank}(X) \leq r$. However, the uniform formulation stated in Lemma 2.1.3 is applicable to all cases.

Consequently, based on Lemma 2.1.3, problem in (1.1) with rank constraints on rectangular matrices can be equivalently transformed into the following formulation with rank constraints on positive semidefinite matrices,

$$\begin{array}{ll} \min_{X,Z} & f(X) \\ s.t. & X \in \mathscr{C},\ \mathbf{rank}(Z) \leq r (= r, \geq r, resp.) \\ & \mathbf{rank}\Big(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix} \Big) \leq m, \end{array} \tag{1.3}$$

where $Z \in \mathbb{S}^n$ is a newly introduced auxiliary matrix. The above formulation can be summarized as a RCOP with convex objective, a set of convex constraints, and rank constraints on semidefinite matrices. For simplicity, the following discussion of RCOP considers rank constraints on semidefinite matrices only.

2.1.2 Rank Minimization Problems and Reformulation

A RMP to minimize a rank function within a convex set is formulated as

$$\begin{array}{ll} \min_X & \mathbf{rank}(X) \\ s.t. & X \in \mathscr{C}, \end{array} \tag{1.4}$$

where $X \in \mathbb{R}^{m\times n}$ is an unknown matrix and $\mathscr{C}$ is a convex set. Without loss of generality, we assume $m \leq n$ in the above formulation provided that $\mathbf{rank}(X) = \mathbf{rank}(X^T)$. The matrix rank function is discontinuous and highly nonlinear. In the following, we introduce an equivalent conversion to reformulate RMPs as RCOPs.

Based on the fact that the trace of a projection matrix is the dimension of the target space, the mathematical expression of this statement is in the form of

$$P(X) = X(X^TX)^{-1}X^T, \mathbf{trace}(P(X)) = \mathbf{rank}(X),$$

where $P(X) \in \mathbb{R}^{m\times m}$ is the projection matrix and its trace is equivalent to the rank of X if X^TX is nonsingular. For singular cases, a small regularization parameter, ε, is introduced to reformulate $P(X)$ as $P_\varepsilon(X) = X(X^TX + \varepsilon I_n)^{-1}X^T$. It has been verified that $\mathbf{trace}(P_\varepsilon(X))$ can approximately represent $\mathbf{rank}(X)$ at any prescribed accuracy as long as ε is properly given [217]. Since $\mathbf{trace}(P_\varepsilon(X))$ is continuous and differentiable with respect to X, the rank function can be replaced by $\mathbf{trace}(P_\varepsilon(X))$ and RMP in (1.4) is rewritten as

$$\begin{array}{lll} \min_{X,Y} & & \mathbf{trace}(Y) \\ s.t. & X \in \mathscr{C}, & Y \succeq X(X^TX + \varepsilon I_n)^{-1}X^T, \end{array} \tag{1.5}$$

where $Y \in \mathbb{S}^m$ is a slack symmetric matrix. The new formulation in (1.5) is equivalent to (1.4) based on the fact that if (Y^*, X^*) is an optimal solution pair to (1.5), its matrix inequality constraint will be active such that $Y^* = X^*((X^*)^TX^* + \varepsilon I_n)^{-1}(X^*)^T$ since we want to minimize $\mathbf{trace}(Y)$ where Y is the upper bound of $X(X^TX + \varepsilon I_n)^{-1}X^T$. Hence, X^* is an optimum to (1.4).

In addition, by using Schur complement to convert the nonlinear matrix inequality in (1.5) into a linear matrix equality by introducing a new matrix $Z \in \mathbb{S}^n$, problem in (1.5) can be equivalently transformed into the following representation,

$$\begin{aligned} \min_{X,Y,Z} \quad & \mathbf{trace}(Y) \\ s.t. \quad & X \in \mathscr{C},\ \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq \mathbf{0},\ Z = X^T X. \end{aligned} \tag{1.6}$$

The new formulation does not require handling matrix inverse operation and eliminates the regularization parameter, ε, in the semedefinite constraint of (1.5). However, the quadratic equality constraint, $Z = X^T X$, is nonconvex. The first step is to transform the quadratic equality constraint, $Z = X^T X$, into a rank constraint.

From proposition (2.1.2), $Z = X^T X$ can be equivalently transformed into $\mathbf{rank}(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix}) \leq m$. Combining the formulation in (1.6), RMPs are converted into RCOPs in the form of

$$\begin{aligned} \min_{X,Y,Z} \quad & \mathbf{trace}(Y) \\ s.t. \quad & X \in \mathscr{C},\ \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq \mathbf{0},\ \mathbf{rank}(\begin{bmatrix} I_m & X \\ X^T & Z \end{bmatrix}) \leq m. \end{aligned} \tag{1.7}$$

The above problem, with a convex objective function, a convex set of constraints, and a rank constraint on semidefinite matrix, is classified as a standard RCOP.

2.1.3 The Reformulation of QCQP to RCOP

The general QCQP problems can be expressed in the form of

$$\begin{aligned} J &= \min_{\mathbf{x}} \mathbf{x}^T Q_0 \mathbf{x} + {a_0}^T \mathbf{x} \\ s.t. \quad & \mathbf{x}^T Q_j \mathbf{x} + {a_j}^T \mathbf{x} \leq c_j,\ \forall\ j = 1, \ldots, m \\ & l_{\mathbf{x}} \leq \mathbf{x} \leq u_{\mathbf{x}}, \end{aligned} \tag{1.8}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the unknown vector to be determined, $Q_j \in \mathbb{S}^{n \times n}$, $j = 0, \ldots, m$, is an arbitrary symmetric matrix, $c_j \in \mathbb{R}$, $j = 1, \ldots, m$, and $a_j \in \mathbb{R}^n$, $j = 0, \ldots, m$. Moreover, $l_{\mathbf{x}} \in \mathbb{R}^n$ and $u_{\mathbf{x}} \in \mathbb{R}^n$ are the lower and upper bounds on $\mathbf{x}$, respectively. It can be seen that equality

constraints can be expressed equivalently by two inequalities and thus they are omitted here. Since Q_j, $j = 0,\ldots,m$, are not necessarily positive semidefinite, problem in (3.25) is generally classified as nonconvex and NP-hard, requiring global search for its optimal solution.

The above QCQP problem with inhomogeneous quadratic function can be transformed into homogenous ones by introducing a new variable $t \in \mathbb{R}$ and a new quadratic constraint $t^2 = 1$ by the following formulation

$$\begin{aligned} J = \min \begin{bmatrix} \mathbf{x}^T & t \end{bmatrix} \begin{bmatrix} Q_0 & a_0/2 \\ a_0^T/2 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} \\ s.t. \quad \begin{bmatrix} \mathbf{x}^T & t \end{bmatrix} \begin{bmatrix} Q_j & a_j/2 \\ a_j^T/2 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} + c_j \leq 0, \forall j = 1,...,m, \\ l_{\mathbf{x}} \leq \mathbf{x} \leq u_{\mathbf{x}}, \\ t^2 = 1. \end{aligned} \tag{1.9}$$

Then $\mathbf{x}^*/t^*$ will be the solution of the original problem stated in (3.39) while $(\mathbf{x}^*, t^*)$ is the solution pair of (1.9). In addition, linear constraints in (3.39) can be rewritten in the above quadratic form as well by setting $Q = \mathbf{0}$. The homogeneous QCQP problem is formulated as

$$\begin{aligned} J = \min \mathbf{x}^T Q_0 \mathbf{x} \\ s.t. \quad \mathbf{x}^T Q_j \mathbf{x} \leq c_j, \ \forall \ j = 1,\ldots,m. \end{aligned} \tag{1.10}$$

Based on this fact, any inhomogeneous QCQP can be transformed into a homogeneous one. Without loss of generality, the following approach to solve nonconvex QCQP problems focuses on homogeneous QCQPs.

In order to solve the nonconvex QCQP in (3.25), the semidefinite relaxation method is firstly introduced to find a tight lower bound on the optimal objective value. By applying interior point method, the relaxed formulation can be solved via a SDP solver [202]. After

introducing a rank one positive definite matrix $X = \mathbf{x}\mathbf{x}^T$, the nonconvex QCQP problem in (3.25) is equivalent to

$$\begin{aligned} & J = \min_X \langle X, Q_0 \rangle \\ s.t. \quad & \langle X, Q_j \rangle \leq c_j, \ \forall \ j = 1, \ldots, m \\ & X = \mathbf{x}\mathbf{x}^T, \end{aligned} \tag{1.11}$$

where '$\langle \cdot \rangle$' denotes the inner product of two matrices, i.e., $\langle A, B \rangle = \mathbf{trace}(A^T B)$. Furthermore, (1.11) is equivalent to the following RCOP

$$\begin{aligned} & J = \min_X \langle X, Q_0 \rangle \\ s.t. \quad & \langle X, Q_j \rangle \leq c_j, \ \forall \ j = 1, \ldots, m \\ & X \succeq \mathbf{0}, \end{aligned} \tag{1.12}$$

$$\mathbf{rank}(X) = 1. \tag{1.13}$$

2.1.4 Cardinality Minimization and Cardinality Constrained Optimization Problems

Cardinality minimization and cardinality-constrained optimization problems have extensive applications in system control and sensing. Existing approaches use surrogate models, e.g., weighted l_1 norm, to approximately represent the cardinality function [35, 78, 86, 214]. By reformulating the two types of problems as RCOPs, the proposed approach can be applied to solve both of them.

The general cardinality minimization problem is formulated as

$$\begin{array}{ll} \min_{\mathbf{x}} & \mathbf{Card}(\mathbf{x}) \\ s.t. & \mathbf{x} \in \mathscr{C}, \end{array} \tag{1.14}$$

where $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$ and $\mathscr{C}$ is a convex set. Based on the fact that cardinality of a vector $\mathbf{x}$ is equal to the rank of a diagonal matrix with diagonal elements $\mathbf{x}$, the cardinality function

can be equivalently reformulated as $\mathbf{Card}(x) = \mathbf{rank}(\mathbf{diag}(\mathbf{x}))$. For a vector $\mathbf{x}$, which is not necessarily element-wise positive, $\mathbf{rank}(\mathbf{diag}(\mathbf{x}))$ is not a positive semidefinite matrix in general. Then a cardinality minimization problem can be formulated as a RMP in the form of

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{y},\mathbf{z}} & \sum \mathbf{y} \\ s.t. & \mathbf{x} \in \mathscr{C}, \begin{bmatrix} \mathbf{diag}(\mathbf{y}) & \mathbf{diag}(\mathbf{x}) \\ \mathbf{diag}(\mathbf{x}) & \mathbf{diag}(\mathbf{z}) \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{rank}\Big(\begin{bmatrix} I_n & \mathbf{diag}(\mathbf{x}) \\ \mathbf{diag}(\mathbf{x}) & \mathbf{diag}(\mathbf{z}) \end{bmatrix} \Big) \leq n, \end{array} \tag{1.15}$$

where $\mathbf{y} \in \mathbb{R}^{\mathbf{n}}$ and $\mathbf{z} \in \mathbb{R}^{\mathbf{n}}$. Moreover, to improve computational efficiency, the last two constraints in (1.15) can be divided into multiple constraints in smaller scale, such that

$$\begin{bmatrix} \mathbf{y_i} & \mathbf{x_i} \\ \mathbf{x_i} & \mathbf{z_i} \end{bmatrix} \succeq \mathbf{0}, \mathbf{rank}\Big(\begin{bmatrix} 1 & \mathbf{x_i} \\ \mathbf{x_i} & \mathbf{z_i} \end{bmatrix} \Big) \leq 1, \forall i = 1, \ldots, n.$$

Similarly, the general cardinality-constrained optimization problems with cardinality constraints on the unknown vector are formulated as

$$\begin{array}{ll} \min_{\mathbf{x}} & f(\mathbf{x}) \\ s.t. & \mathbf{x} \in \mathscr{C}, \ \mathbf{Card}(\mathbf{x}) \leq \mathbf{r}, \end{array} \tag{1.16}$$

where $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$, $f(\mathbf{x})$ is a convex function, and $\mathscr{C}$ is a convex set. As a general vector $\mathbf{x}$ is not necessarily element-wise positive, problem in (1.16) can be reformulated as

$$\begin{array}{ll} \min_{\mathbf{x},\mathbf{y},\mathbf{z}} & f(\mathbf{x}) \\ s.t. & \mathbf{x} \in \mathscr{C}, \begin{bmatrix} \mathbf{diag}(\mathbf{y}) & \mathbf{diag}(\mathbf{x}) \\ \mathbf{diag}(\mathbf{x}) & \mathbf{diag}(\mathbf{z}) \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{rank}\Big(\begin{bmatrix} \mathbf{diag}(\mathbf{y}) & \mathbf{0} \\ \mathbf{0} & \mathbf{diag}(\mathbf{z}) \end{bmatrix} \Big) \leq 2r. \end{array} \tag{1.17}$$

Through the above conversion, both the cardinality minimization problems and cardinality-constrained optimization problems are formulated as RCOPs. In the following, we will describe the IRM approach to solve the general RCOPs.

2.2 An Iterative Rank Minimization Approach

2.2.1 IRM Approach to Rank Constrained Optimization Problems

Given the above discussion, we formulate the general RCOPs as follows,

$$\begin{array}{ll} \min_X & f(X) \\ s.t. & X \in \mathscr{C},\, X \succeq \mathbf{0},\, \mathbf{rank}(X) \leq r. \end{array} \tag{2.18}$$

For simplicity, the lower bounded rank inequality constraints and rank equality constraints are omitted in problem (2.18) and the following formulations. However, the method proposed for solving RCOPs is applicable to all cases. Satisfying the rank constraint for an unknown matrix is computationally complicated. Existing methodologies for rank-constrained problems are mainly focused on matrix factorization and/or linearization [58,81], which have slow convergence rate and are sensitive and dependent on the existence of initial guess.

In fact, the number of nonzero eigenvalues of a matrix is identical to its rank. For an unknown square matrix $U \in \mathbb{S}^n_+$, it is not feasible to examine its eigenvalues before it is determined. We focus on the fact that when a matrix rank is r, it has r nonzero eigenvalues. Therefore, instead of making constraint on the rank, we focus on constraining the eigenvalues of U such that the $n-r$ eigenvalues of U are all zeros. The eigenvalue constraints on matrices have been used for graph design [175] and are applied here for rank-constrained problems. Before addressing the detailed approach for rank-constrained problems, we first provide necessary observations that will be used subsequently in the approach.

Proposition 2.2.1. *The $(r+1)$th largest eigenvalue λ_{n-r} of matrix $U \in \mathbb{S}^n_+$ is no greater than e if and only if $eI_{n-r} - V^T U V \succeq \mathbf{0}$, where I_{n-r} is the identity matrix with a dimension*

of $n-r$, $V \in \mathbb{R}^{n\times(n-r)}$ are the eigenvectors corresponding to the $n-r$ smallest eigenvalues of U.

Proof. Assume the eigenvalues of U is sorted in descending orders in the form of $[\lambda_n, \lambda_{n-1}, \ldots, \lambda_1]$. Since the Rayleigh quotient of an eigenvector is its associated eigenvalue, then $eI_{n-r} - V^T UV$ is a diagonal matrix with diagonal elements set as $[e-\lambda_{n-r}, e-\lambda_{n-r-1}, \ldots, e-\lambda_1]$. Therefore $e \geq \lambda_{n-r}$ if and only if $eI_{n-r} - V^T UV \succeq \mathbf{0}$. □

Corollary 2.2.2. *When $e = 0$ and U is a positive semidefinite matrix, $\mathbf{rank}(U) \leq r$ holds if and only if $eI_{n-r} - V^T UV \succeq \mathbf{0}$. $\mathbf{rank}(U) = r$ holds if and only if $eI_{n-r} - V^T UV \succeq \mathbf{0}$ and $V_p^T UV_p \succ 0$. $\mathbf{rank}(U) \geq r$ holds if and only if $V_p^T UV_p \succ 0$. The matrices $V \in \mathbb{R}^{n\times(n-r)}$ and $V_p \in \mathbb{R}^{n\times 1}$ are the eigenvectors corresponding to the $n-r$ smallest eigenvalues and $n-r+1$ smallest eigenvalue of U, respectively.*

However, for problem (2.18), before we solve X, we cannot obtain the exact V matrix, i.e. the eigenvectors of X. Therefore an iterative method is proposed to solve (2.18) by gradually approaching the constrained rank. At each step k, we will solve the following semidefinite programming problem formulated as

$$\begin{array}{cc} \min_{X_k, e_k} & f(X_k) + w_k e_k \\ s.t. & X_k \in \mathscr{C}, X_k \succeq \mathbf{0} \\ & e_k I_{n-r} - V_{k-1}^T X_k V_{k-1} \succeq \mathbf{0} \\ & e_k \leq e_{k-1}, \end{array} \tag{2.19}$$

where $w_k = w_0 t^k$ is the weighting factor at iteration k. w_k is increasing with the increment of k when $t > 1$ and $w_0 > 0$ are given parameters. $V_{k-1} \in \mathbb{R}^{n\times(n-r)}$ are the orthonormal eigenvectors corresponding to the $n-r$ smallest eigenvalues of X_{k-1} solved at previous iteration $k-1$. At the first iteration where $k = 1$, e_0 is the $(n-r)$th smallest eigenvalue of X_0 described below. Furthermore, we will prove that each sequential problem is feasible with the additional constraint, $e_k \leq e_{k-1}$, in the convergence analysis section. It is known that

the existing SDP solver based on interior point method has computational time complexity in order of $O(m(n^2m+n^3))$ for a SDP problem with m linear constraints and a linear matrix inequality of dimension n. As a result, the extra linear matrix inequality $e_kI_{n-r} - V_{k-1}^T X_k V_{k-1} \succeq 0$ in the subproblem leads to additional linear constraints at the order above $O(n^2)$. Therefore, the time complexity of each iteration is lower bounded by $O(n^6)$.

At each step, we are trying to optimize the original objective function and at the same time minimize parameter e such that when $e = 0$, the rank constraint on X is satisfied. The weighting factor, w_k, acts as a regularization factor and increasing its values at each step will enforce the $r+1$th largest eigenvalue to gradually reduce to zero. With this regularization factor, our algorithm not simply switches between searching for the unknown matrix and then its eigenvectors, it also drives the slack variable e_k to quickly reduce to zero. Furthermore, the additional constraint, $e_k \leq e_{k-1}$, guarantees that e_k is monotonically decreasing in the iterative algorithm. The above approach is repeated until $e \leq \varepsilon$, where ε is a small threshold for the stopping criteria. Each iteration is solved via an existing semidefinite programming (SDP) solver based on an interior point method, which is applicable to small and medium size SDP problems [181]. It is straightforward to extend (2.19) to problems with multiple rank constraints. For brevity, the simplest version with one rank constraint is described here.

In addition, an initial starting point, V_0, is required at the first iteration $k = 1$. It is intuitive to use the relaxed solution by dropping the last constraint and the penalty term in the objective function in (2.19) for starting point. Under this assumption, X_0 is obtained via

$$\begin{array}{ll} \min_{X_0} & f(X_0) \\ s.t. & X_0 \in \mathscr{C}, X_0 \succeq \mathbf{0}, \end{array} \tag{2.20}$$

and V_0 are the eigenvectors corresponding to $n-r$ smallest eigenvalues of X_0. However, under special cases, adding the penalty term in the objective function will not change the solution found from the semidefinite relaxation formulation, e.g., the penalty term is identical

to the objective function, which makes the penalty term and the semidefinite constraint become inactive. A new initial value will be used in the IRM algorithm if the one from the semidefinite relaxation problem does not converge to the constrained-rank. The IRM approach is summarized in Algorithm 1 for general RCOPs.

Algorithm 1 Iterative Rank Minimization for Solving (2.18)

Input: Problem information $\mathscr{C}$, w_0, t, ε_1, ε_2, k_{max}
Output: with local minimum X^*
begin

1. **Initialize** Set $k = 0$, solve the relaxed problem in (2.20) to obtain V_0 from X_0 via eigenvalue decomposition. set $k = k+1$
2. **while** $k \leq k_{max}$ & $e_k \geq \varepsilon_1$ || $|(f(X_k) - f(X_{k-1}))/f(X_{k-1})| \geq \varepsilon_2$
3. Solve sequential problem (2.19) and obtain X_k, e_k
4. Update V_k from X_k via eigenvalue decomposition and set $k = k+1$
5. Update w_k via $w_k = w_{k-1} * t$
6. **end while**

end

2.2.2 IRM Approach to RMPs

Combing the algorithm for RCOPs in the above subsection, the RMPs reformulated as RCOPs in (1.7) can be solved iteratively and the sequential problem at iteration k is formulated as

$$\begin{array}{cc} \min_{X_k, Y_k, Z_k, e_k} & \mathbf{trace}(Y_k) + w_k e_k \\ s.t. & X_k \in \mathscr{C} \\ & \begin{bmatrix} Y_k & X_k \\ X_k^T & Z_k \end{bmatrix} \succeq \mathbf{0}, \begin{bmatrix} I_m & X_k \\ X_k^T & Z_k \end{bmatrix} \succeq \mathbf{0} \\ & e_k I_n - V_{k-1}^T \begin{bmatrix} I_m & X_k \\ X_k^T & Z_k \end{bmatrix} V_{k-1} \succeq \mathbf{0} \\ & e_k \leq e_{k-1}, \end{array} \tag{2.21}$$

where $V_{k-1} \in \mathbb{R}^{(m+n)\times n}$ are the eigenvectors corresponding to the n smallest eigenvalues of $\begin{bmatrix} I_m & X_{k-1} \\ X_{k-1}^T & Z_{k-1} \end{bmatrix}$ solved at previous iteration $k-1$.

Since the heuristic methods, e.g., trace heuristic method, are often considered as a good candidate for approximate solutions of RMPs. They are adopted here to obtain the starting point, V_0, by solving the relaxed X_0. The trace heuristic method for RMPs formulated in (1.4) is listed below by applying the aforementioned semidefinite embedding lemma for general unknown matrix $X \in \mathbb{R}^{m\times n}$ [64],

$$\begin{array}{cc} \min_{X_0,Y_0,Z_0} & \mathbf{trace}(Y_0) + \mathbf{trace}(Z_0) \\ s.t. & X_0 \in \mathscr{C}, \begin{bmatrix} Y_0 & X_0 \\ X_0^T & Z_0 \end{bmatrix} \succeq \mathbf{0}. \end{array} \tag{2.22}$$

Providing this starting point, problem (1.7) can be solved by the proposed IRM algorithm where the sequential problem at each iteration is formulated in (2.21).

2.2.3 Convergence Analysis of IRM

In the following, we provide the convergence analysis of the proposed IRM method for solving general RCOPs. We first introduce one preparatory lemma.

Lemma 2.2.3. *(Corollary 4.3.37 in [98]) For a given matrix $X \in \mathbb{S}^n$ with eigenvalues in increasing order, denoted as $[\lambda_1^X, \lambda_2^X, \ldots, \lambda_n^X]$, when $m \leq n$, one has $\lambda_m^X \leq \lambda_{\max}(V^T X V)$ for any $V \in \{V | V \in \mathbb{R}^{n\times m}, V^T V = I_m\}$. Moreover, equality holds when columns of V are the eigenvectors of X associated with the m smallest eigenvalues.*

Assumption 2.2.4. *Problem in (2.18) is feasible with interior points [28] and the relaxed problem (2.20) is bounded.*

Proposition 2.2.5. *(Local Convergence) When Assumption 2.2.4 holds, the sequential problem of (2.19) is feasible. Moreover, when $\|e_k - e^*\| \to 0$, i.e., $\lambda_{\max}(V_{k-1}^T X_k V_{k-1}) \to 0$,*

where (X_k, e_k) *is the optimum pair of (2.19) at iteration* k *and* $e^* = 0$*, the sequence* $\{e_k\}$ *converges to* 0 *with at least a sublinear convergence rate.*

Proof. Variable e_k introduced in (2.19) is a slack variable. Then e_k obtained at each step k satisfies $e_k = \lambda_{\max}(V_{k-1}^T X_k V_{k-1}) \geq \lambda_{n-r}^{X_k} = \lambda_{\max}(V_k^T X_k V_k)$, where the inequality follows Lemma 2.2.3. According to Assumption 2.2.4, the relaxed formulation in (2.20) has an optimal solution, denoted by X_0. For the first iteration $k = 1$, the pair (X_0, e_0) is a feasible solution for the sequential problem at $k = 1$, where e_0 is the $r+1$th largest eigenvalue of X_0. For any $k \geq 1$, the solution pair (X_k, e_k) obtained at iteration k is a feasible solution pair for the sequential problem at iteration $k+1$. It is because X_k satisfies the original constraints $X_k \in \mathscr{C}$ and $X_k \succeq 0$ at iteration k, then it will satisfy the same set of constraints at iteration $k+1$. Furthermore, since e_k satisfies $e_k I_{n-r} - V_{k-1}^T X_k V_{k-1} \succeq 0$ at iteration k and $e_k \geq \lambda_{\max}(V_k^T X_k V_k)$, the inequality $e_{k+1} I_{n-r} - V_k^T X_{k+1} V_k \succeq 0$ at iteration $k+1$ will be satisfied if (X_{k+1}, e_{k+1}) is set to be (X_k, e_k). Therefore, the sequential problem of (2.19) has at least one pair of feasible solution and thus it is feasible. Consequently, from the last constraint of (2.19), we get that $e_{k+1} \leq e_k$ and it can be rearranged as $\frac{|e_{k+1} - e^*|}{|e_k - e^*|} \leq 1$. This concludes the proof of at least a sublinear convergence rate of sequence $\{e_k\}$. □

Next, we prove that when $\lim_{k \to +\infty} e_k = e^* = 0$, the corresponding solution X_k from (2.19) is a local optimum of the original problem (2.18). Some definitions will be introduced here, including $\mathscr{C} = \{X | \mathbf{g}(X) \leq \mathbf{0}\}$, $\mathscr{C}' = \mathscr{C} \cap \{X | X \in \mathbb{S}^n, X \succeq 0\}$, $\mathscr{P}_k = \{P | P \in \mathbb{R}^{n \times r}, V_{k-1}^T P = \mathbf{0}\}$ and $\Gamma_k = \{X | X = PP^T, P \in \mathscr{P}_k\}$, where $\mathbf{g}(X) : \mathbb{S}^n \to \mathbb{R}^p$ are convex functions defining $\mathscr{C}$. Then we have the following assumption.

Assumption 2.2.6. *When* $\lim_{k \to +\infty} e_k = 0$*, each sequential problem has only one optimum, that is,* $X_k \in \mathscr{C}' \cap \Gamma_k$ *is the only minimizer of sequential problem at iteration k.*

Proposition 2.2.7. *When* $\lim_{k\to+\infty} e_k = 0$ *and Assumption 2.2.6 holds, the corresponding solution* $\lim_{k\to+\infty} X_k = X^*$*, where* X^* *is a local minimizer of (2.18).*

Proof. Since $\lim_{k\to+\infty} e_k = 0$, it implies $V_{k-1}^T X_k V_{k-1}$

$= \mathbf{0}$. Then there exists $P_k \in \mathscr{P}_k$ such that $X_k = P_k P_k^T$, which implies the rank of X_k is no greater than r. As V_k are the orthonormal eigenvectors of $X_k = P_k P_k^T$, we have $V_k^T P_k = 0$, i.e., $\mathbf{span}\{V_k\} = (\mathscr{P}_k)^\perp$. As a result, $\mathscr{P}_{k+1} = (\mathbf{span}\{V_k\})^\perp = ((\mathscr{P}_k)^\perp)^\perp = \mathscr{P}_k$, which leads to $\Gamma_{k+1} = \Gamma_k$. It implies the feasible regions for the sequential problem at iterations k and $k+1$ are the same. From Assumption 2.2.6, $\{X_k\}$ converges to a unique solution.

For the sequential problem (2.19) at iteration k, the optimality condition of the prime-dual problem pair are written as

$$\begin{array}{c} \nabla f(X) + \sum_{j=1}^p \lambda_k^{(j)} \nabla \mathbf{g}_j(X) + V_{k-1} S_k^{(1)} V_{k-1}^T - S_k^{(2)} = \mathbf{0} \\ \lambda_k^{(j)} \mathbf{g}_j(X) = 0,\ \lambda_k^{(j)} \geq 0,\ \forall\ j = 1,\ldots,p \\ S_k^{(1)}(e_k I_{n-r} - V_{k-1}^T X_k V_{k-1}) = \mathbf{0},\ S_k^{(2)} X_k = \mathbf{0} \\ X_k \succeq \mathbf{0}, S_k^{(1)} \succeq \mathbf{0},\ S_k^{(2)} \succeq \mathbf{0} \end{array} \tag{2.23}$$

where $\lambda_k = [\lambda_k^{(1)},\ldots,\lambda_k^{(p)}]^T, S_k^{(1)}, S_k^{(2)}$ are the dual variables. Problem (2.18) is equivalent to

$$\min_{X\in\mathbb{S}^n, P\in\mathbb{R}^{n\times r}} f(X),\ \ s.t.\ X \in \mathscr{C},\ X = PP^T,$$

whose optimality conditions are

$$\begin{array}{c} \nabla f(X) + \sum_{j=1}^p \mu^j \nabla \mathbf{g}_j(X) - S = \mathbf{0},\ (S + S^T)P = \mathbf{0} \\ \mu^{(j)} \mathbf{g}_j(X) = 0,\ \mu^j \geq 0,\ \forall\ j = 1,\ldots,p \end{array} \tag{2.24}$$

where $\mu = [\mu^{(1)},\ldots,\mu^{(p)}]^T$ and S are the associated dual variables. When $\lim_{k\to+\infty} e_k = 0$, it implies $V_{k-1}^T X_k V_{k-1} = 0$, where columns of V_{k-1} are orthonormal and rank of X_k is no more than r due to Lemma 2.2.3. As a result, there exists a $P_k \in \mathbb{R}^{n\times r}$ such that $X_k = P_k P_k^T$, $V_{k-1}^T P_k = 0$ and $S_k^{(2)} P_k = 0$ from (2.23). By setting $\mu = \lambda_k$, $S = -(V_{k-1} S_k^{(1)} V_{k-1}^T - S_k^{(2)}) \in \mathbb{S}^n$,

the prime-dual pair (μ, S, X_k, P_k) satisfies (2.24), which indicates that (X_k, P_k) is a local minimizer of the original problem (2.18). If a local optimum of (2.18) is denoted by X^*, combined with the conclusion that $\{X_k\}$ converges to a unique solution, we get $\lim_{k\to+\infty} X_k = X^*$. □

2.3 IRM for sparse QCQP

2.3.1 General Homogeneous QCQPs

A general homogeneous QCQP problem can be expressed in the form of

$$\begin{aligned} J &= \min_{\mathbf{x}} \mathbf{x}^T Q_0 \mathbf{x} \\ s.t. \quad & \mathbf{x}^T Q_j \mathbf{x} \leq c_j, \ \forall\ j = 1, \ldots, m, \end{aligned} \tag{3.25}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the unknown vector to be determined, $Q_j \in \mathbb{S}^n$, $j = 0, \ldots, m$, is an arbitrary symmetric matrix, and $c_j \in \mathbb{R}$, $j = 1, \ldots, m$. Since Q_j, $j = 0, \ldots, m$, are not necessarily positive semidefinite, problem in (3.25) is generally classified as nonconvex and NP-hard, requiring global search for its optimal solution. Any inhomogeneous QCQPs with linear terms can be reformulated as homogeneous ones by defining an extended vector $\tilde{\mathbf{x}} = [\mathbf{x}^T, t]^T$ as well as a new quadratic constraint $t^2 = 1$ [128]. For example, an inhomogeneous quadratic function can be transformed into an equivalent homogeneous one in form of

$$\begin{aligned} \mathbf{x}^T Q_j \mathbf{x} + {a_j}^T \mathbf{x} - c_j &= \begin{bmatrix} \mathbf{x}^T & t \end{bmatrix} \begin{bmatrix} Q_j & a_j/2 \\ a_j^T/2 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} - c_j \\ &= \tilde{\mathbf{x}}^T Q_j' \tilde{\mathbf{x}} - c_j, \end{aligned}$$

where $Q_j' = \begin{bmatrix} Q_j & a_j/2 \\ a_j^T/2 & 0 \end{bmatrix}$. Therefore, instead of solving inhomogeneous QCQPs, we can solve its equivalent homogeneous formulation expressed in (3.25). Without loss of generality, the following approach to nonconvex QCQP problems focuses on homogeneous QCQPs only.

2.3.2 Transformation of QCQPs into Rank-One Constrained Optimization Problems

By introducing a rank-one positive semidefinite matrix $X = \mathbf{x}\mathbf{x}^T$, problem in (3.25) can be rewritten as [202]

$$\begin{aligned} & J = \min_{X,\mathbf{x}} \langle X, Q_0 \rangle \\ s.t. \quad & \langle X, Q_j \rangle \leq c_j, \ \forall \ j = 1, \ldots, m, \\ & X = \mathbf{x}\mathbf{x}^T, \end{aligned} \tag{3.26}$$

where '$\langle \cdot \rangle$' denotes the inner product of two matrices, i.e., $\langle A, B \rangle = \mathbf{trace}(A^T B)$. However, the rank-one constraint on X is nonlinear and it is computationally complicated to find the exact rank-one solution. One existing approach is to relax the rank-one constraint by a semidefinite constraint, denoted as $X \succeq \mathbf{x}\mathbf{x}^T$. The semidefinite relaxation approach generally yields a tighter lower bound on the optimal value of (3.25) than the one obtained from linearization relaxation technique [10]. However, the relaxation method will not generate an optimal solution of the unknown variables $\mathbf{x}$, not even an infeasible solution in most cases.

The new approach introduced in this paper focuses on finding the unknown matrix satisfying rank-one constraint and the first step is to equivalently transform the rank-one constraint, $X = \mathbf{x}\mathbf{x}^T$, into multiple small scale quadratic equality constraints.

Definition 2.3.1. *For $N = \{1, \ldots, n\}$, $\alpha_p \subseteq N$, $p = 1, 2, \ldots, P$, is a complete decomposition of N if for any $1 \leq l_1, l_2 \leq n$, there exist a α_p, $p = 1, 2, \ldots, P$ such that $\{l_1, l_2\} \subseteq \alpha_p$.*

As a result, problem in (3.26) can be equivalently transformed into

$$\begin{aligned} & J = \min_{X,\mathbf{x}} \langle X, Q_0 \rangle \\ s.t. \quad & \langle X, Q_j \rangle \leq c_j, \ \forall \ j = 1, \ldots, m \end{aligned} \tag{3.27}$$

$$X_{\alpha_p} = \mathbf{x}_{\alpha_p}\mathbf{x}_{\alpha_p}^T,\ p = 1,2,\ldots,P,$$

where X_{α_p} is a principle submatrix of X consisting entries of X with rows and columns indexed by α_p and x_{α_p} is a subset of x with index set α_p. Problem in (3.31) is equivalent to (3.26) if $\alpha_p \subseteq N$, $p = 1,2,\ldots,P$, is a complete decomposition of N. To make it clear, a simple example of decomposition is demonstrated below.

Example: For $n = 3$, $\alpha_1 = \{1,2\}$, $\alpha_2 = \{2,3\}$ and $\alpha_3 = \{1,3\}$ are a complete decomposition of $N = \{1,2,3\}$. When $n = 3$, problem in (3.26) is equivalent to

$$J = \min_X \langle X, Q_0 \rangle \tag{3.28}$$

$$\begin{aligned} s.t.\quad & \langle X, Q_j \rangle \le c_j,\ \forall\ j = 1,\ldots,m, \\ & \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \end{bmatrix} \\ & \begin{bmatrix} X_{22} & X_{23} \\ X_{32} & X_{33} \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} \begin{bmatrix} x_2 & x_3 \end{bmatrix} \\ & \begin{bmatrix} X_{11} & X_{13} \\ X_{31} & X_{33} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} \begin{bmatrix} x_1 & x_3 \end{bmatrix}, \end{aligned}$$

where $X = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix}$ and $\mathbf{x} = [x_1, x_2, x_3]^T$.

Noteworthily, for sparse problems, i.e., problems with parameter matrices, $Q_j, \forall j = 0,\ldots,m$, being sparse, the complete decomposition will be less complex compared to the problems with dense parameter matrices. Figure 2.1 illustrates examples of the complete decomposition for problems with certain sparsity patterns. Mathematically, we define a matrix $A \in \mathbb{S}^n$ with $A_{ts} = 1$ if there exists a $j = 0,\ldots,m$ such that $(Q_j)_{ts} \neq 0$ and $A_{ts} = 0$ otherwise. Then (3.26) can be further simplified as

$$\begin{aligned} & J = \min_{X,\mathbf{x}} \langle X, Q_0 \rangle \\ s.t.\quad & \langle X, Q_j \rangle \le c_j,\ \forall\ j = 1,\ldots,m \end{aligned} \tag{3.29}$$

$$X_{ts} = \mathbf{x}_t \mathbf{x}_s, \forall A_{ts} \neq 0.$$

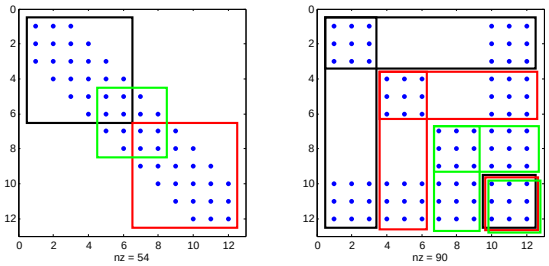


Figure 2.1: Examples of complete decomposition for two problems with typical sparsity patterns: band (left) and block-arrow (right)

Through the above decomposition, the rank-one constraint on X is equivalently transformed into multiple quadratic equality constraints on submatrices of X. However, when all decomposed matrices, X_{α_p}, are rank-one matrices, it does not imply that $X_{\alpha_p} = \mathbf{x}_{\alpha_p} \mathbf{x}_{\alpha_p}^T$, $p = 1, 2, \ldots, P$, due to the coupled elements of x. For example, X_{11} obtained from the first and third submatrices in (3.28) are not necessarily equivalent to each other if we simply have rank-one constraint on both $\begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}$ and $\begin{bmatrix} X_{11} & X_{13} \\ X_{31} & X_{33} \end{bmatrix}$. To establish equivalent transformation, rank-one constraints are considered for extended submatrices such that

$$\mathbf{rank}\left(\begin{bmatrix} X_{\alpha_p} & \mathbf{x}_{\alpha_p} \\ (\mathbf{x}_{\alpha_p})^T & 1 \end{bmatrix}\right) = 1, \forall\, p = 1, 2, \ldots, P. \tag{3.30}$$

Consequently, the original optimization problem with single rank-one constraint in (3.27) is converted into an equivalent optimization problem with multiple rank-one constraints on small size submatrices. The equivalent formulation is written as

$$J = \min_X \langle X, Q_0 \rangle \tag{3.31}$$

$$s.t. \qquad \langle X, Q_j \rangle \leq c_j, \ \forall\, j = 1, \ldots, m$$

$$\mathbf{rank}(\begin{bmatrix} X_{\alpha_p} & \mathbf{x}_{\alpha_p} \\ (\mathbf{x}_{\alpha_p})^T & 1 \end{bmatrix}) = 1, \forall\, p = 1,2,\ldots,P.$$

2.3.3 Iterative Rank Minimization Approach

The new formulation of QCQP in (3.31) is to minimize a linear function under linear constraints and rank constraints, which is classified as a RCOP. Existing methods for solving RCOPs are subject to sensitive initial guess and uncertainty of convergence. An iterative approach with guaranteed convergence is proposed here to search for a local optimal solution that satisfies all of the decomposed rank-one constraints. In fact, for a rank-one matrix, the only nonzero eigenvalue is its largest one, which indicates the second largest eigenvalue of the rank-one matrix is zero. Therefore, a semidefinite constraint is introduced to put a upper bound on the second largest eigenvalue of the to-be-determined matrix.

Proposition 2.3.2. *For a nonzero positive semidefinite matrix, $U \in \mathbb{S}^n_+$, its rank is no more than one if and only if there exists $V \in \mathbb{R}^{n\times(n-1)}, V^TV = I_{n-1}$ such that $V^TXV = 0$, where I_{n-1} is an identity matrix with dimension $n-1$.*

Proof. Assuming the eigenvalues of U are sorted in increasing order, $[\lambda_1^U, \lambda_2^U, \ldots, \lambda_n^U]$, then the following inequality holds (Corollary 4.3.39 in [98]),

$$\lambda_1^U + \lambda_2^U + \ldots + \lambda_{n-1}^U \leq \mathbf{trace}(V^TUV) \tag{3.32}$$

for any $V \in \mathbb{R}^{n\times(n-1)}$ and $V^TV = I_{n-1}$. When V are the eigenvectors corresponding to the $n-1$ smallest eigenvalues of U, the equality holds for (3.32). For the sufficiency proof, when the rank of U is no more than one and V are the eigenvectors corresponding to the $n-1$ smallest eigenvalues of U, $V^TUV = \mathbf{0}_{(n-1)\times(n-1)}$, which indicates $V^TXV = 0$. To prove the necessity, we start from $V^TXV = 0$. It implies that $0 = \mathbf{trace}(V^TXV) \geq$

$(\lambda_1^U + \lambda_2^U + \ldots + \lambda_{n-1}^U) \geq 0$. Then it leads to $\lambda_1^U = \lambda_2^U = \ldots = \lambda_{n-1}^U = 0$, which indicates rank of U is no more than one. □

Based on the above discussion, we will substitute the rank one constraint, $\mathbf{rank}(\begin{bmatrix} X_{\alpha_p} & \mathbf{x}_{\alpha_p} \\ (\mathbf{x}_{\alpha_p})^T & 1 \end{bmatrix}) = 1$, by the semidefinite constraint, $r^{(p)} I_{|\alpha_p|} - (V^p)^T \begin{bmatrix} X^{(p)} & \mathbf{x}^{(p)} \\ (\mathbf{x}^{(p)})^T & 1 \end{bmatrix} V^p \succeq 0$, where $r^{(p)} = 0$, $X^{(p)} = X_{\alpha_p}$ and $\mathbf{x}^{(p)} = \mathbf{x}_{\alpha_p}$ are simplified notations, and $V^p \in \mathbb{R}^{(|\alpha_p|+1)\times|\alpha_p|}$ are the eigenvectors corresponding to the $|\alpha_p|$ smallest eigenvalues of $\begin{bmatrix} X^{(p)} & \mathbf{x}^{(p)} \\ (\mathbf{x}^{(p)})^T & 1 \end{bmatrix}$. In the following, we denote $\tilde{X}^{(p)} = \begin{bmatrix} X^{(p)} & \mathbf{x}^{(p)} \\ (\mathbf{x}^{(p)})^T & 1 \end{bmatrix}$ for simplicity. However, before we solve $\tilde{X}^{(p)}$, we cannot obtain the exact $V^{(p)}$ matrix, thus an iterative rank minimization (IRM) method is proposed to gradually minimize the rank of $\tilde{X}^{(p)}$. At each step k, we will solve the following semidefinite programming (SDP) problem formulated as

$$
\begin{aligned}
& J_k = \min_{X_k, \mathbf{x}_k, \mathbf{r}_k} \langle X, Q_0 \rangle + w_k |\mathbf{r}_k|_1 \qquad (3.33) \\
s.t. \quad & \langle X_k, Q_j \rangle \leq c_j, \ \forall\ j = 1, \ldots, m \\
& \tilde{X}_k^{(p)} \succeq 0, \ p = 1, 2, \ldots, P \\
& r_k^{(p)} I_{|\alpha_p|} - (V_{k-1}^{(p)})^T \tilde{X}_k^{(p)} V_{k-1}^{(p)} \succeq 0, \ p = 1, 2, \ldots, P,
\end{aligned}
$$

where w_k is an increasing weighting factor for $|\mathbf{r}_k|_1$ with $\mathbf{r}_k = [r_k^{(1)}, r_k^{(2)}, \ldots, r_k^{(P)}]$ in kth iteration and $V_{k-1}^{(p)}$ are the eigenvectors corresponding to the α_p smallest eigenvalues of $\tilde{X}^{(p)}$ solved at previous iteration $k-1$. In each iteration, we are trying to optimize the original objective function and at the same time minimize 1-norm of the newly introduced parameter $\mathbf{r}_k$ such that when $r_k^{(p)} = 0$, the rank one constraint on $\tilde{X}_k^{(p)}$, $p = 1, 2, \ldots, P$, is satisfied. Meanwhile, since $\tilde{X}_k^{(p)}$ is constrained to be positive semidefinite, the term $(V_{k-1}^{(p)})^T \tilde{X}_k^{(p)} V_{k-1}^{(p)}$ is positive semidefinite as well, which implies that the value of $r_k^{(p)}$ is nonnegative in order to satisfy $r_k^{(p)} I_{|\alpha_p|} - (V_{k-1}^{(p)})^T \tilde{X}_k^{(p)} V_{k-1}^{(p)} \succeq 0$ in (3.33). The above approach is repeated until

$|\mathbf{r}_k|_1 \leq \varepsilon$, where ε is a small threshold for stopping criteria. Once all of the submatrices $\tilde{X}^{(p)}$ satisfy rank-one constraints, the original matrix X is rank one and the converged solution $\mathbf{x}$ satisfies $X = \mathbf{x}\mathbf{x}^T$.

The IRM algorithm is summarized below.

Algorithm 1: Iterative Rank Minimization
Input: Parameters $Q_0, Q_j, c_j,\ j = 1,\ldots,m,\ w_1, \alpha_p, k_{\max}$
Output: Unknown rank one matrix X and unknown state vector $\mathbf{x}$
begin

1. **initialize** Set $k = 0$, solve the SDP relaxation of (3.26) to find $\tilde{X}_0^{(p)}$ and obtain $V_0^{(p)}$ from $\tilde{X}_0^{(p)}$, set $k = k+1$
2. **while** $|\mathbf{r}_k|_1 > \varepsilon$ && $k \leq k_{\max}$
3. Solve problem (3.33) and obtain $X_k, \mathbf{x}_k, \mathbf{r}_k$
4. Update $V_k^{(p)}$ from $\tilde{X}_k^{(p)}$, $p = 1,2,\ldots,P$
5. $k = k+1$, update $w_{k+1} \geq w_k$
6. **end while**
7. Find X and $\mathbf{x}$

end

Remark 2.3.3. *For an SDP problem with m linear constraints and a linear matrix inequality of dimension n, the computational time complexity is $O(m(n^2m+n^3))$ when solved by the interior point method [24]. Without decomposition, for original $X \in \mathbb{S}^n$, the additional SDP constraint $r_k I_{n-1} - V_{k-1}^T X_k V_{k-1} \succeq 0$ in (3.33) will introduce $m' = O(n^2)$ additional linear constraints. Then the computational time complexity becomes $O(n^6)$ and the corresponding memory is $O(m'^2)$, which is impractical for large scale QCQPs. Therefore, decomposing the original rank-one constraint into multiple ones in small scale reduces size of each SDP constraints in (3.33).*

Further analysis on sufficient conditions for local convergence of IRM and convergence rate is discussed below.

2.4 Conclusions

This paper established a uniform formulation for rank-constrained optimization problems (RCOPs) and rank minimization problems. An iterative rank minimization (IRM) method is proposed to solve general RCOPs. The sublinear convergence of IRM to a local optimum is proved through the duality theory and the Karush-Kuhn-Tucker conditions. We apply IRM to several representative applications, including system identification, output feedback stabilization problem, structured H_2 controller design, and cardinality minimization problems. Comparative results are provided to verify the effectiveness and improved performance of IRM in terms of convergence rate, robustness to initial guess, and cost function. Future work will focus on seeking the global optimal solution for RCOPs.

Chapter 3: Application of IRM

3.1 Applications of IRM for RCOPs

In this section, four representative applications are solved via the proposed IRM method. They are system identification, output feedback stabilization, structured H_2 controller design, and cardinality minimization problems. To verify the effectiveness and improved performance of IRM, results obtained from IRM are compared with those obtained from existing approaches. All simulation is run on a desktop computer with a 3.50 GHz processor and a 16.0 RAM.

3.1.1 System Identification Problem

Considering a linear time-invariant system with input $u \in \mathbb{R}^m$ and output $y \in \mathbb{R}^p$, we want to identify the linear system through T samples $w_d := (u_d, y_d) \in (\mathbb{R}^m \times \mathbb{R}^p)^T$. This identification problem is equivalent to finding a full row rank matrix $R = [R_0, R_1, \ldots, R_l]$ such that

$$\begin{aligned} R_0 w(t) + R_0 w(t) + \cdots + R_l w(t+l) = 0, \\ \forall\, t = 1, 2, \ldots, T-l, \end{aligned} \tag{1.1}$$

where l is the lag of the identified model. Expression in (1.1) can be rewritten as $R\mathscr{H}_{l+1,T-l}(w) = 0$, where $\mathscr{H}_{l+1,T-l}(w) = \begin{bmatrix} w(1) & w(2) & \cdots & w(T-l) \\ w(2) & w(3) & \cdots & w(T-l+1) \\ \vdots & \vdots & & \vdots \\ w(l+1) & w(l+2) & \cdots & w(T) \end{bmatrix}$. Since the row dimension of R is equal to the number of the outputs, p, of the model [207], the rank of $\mathscr{H}_{l+1,T-l}(w)$ is constrained by p. As a result, the identification problem is equivalent to the following low-rank approximation problem expressed as,

$$\begin{array}{ll} \min_w & \|w - w_d\|_F^2 \\ s.t. & \textbf{rank}(\mathscr{H}_{l+1,T-l}(w)) \leq r, \end{array}$$

where $r = (m+p)(l+1) - p$.

Using the benchmark system identification problems in database DAISY [149], we solve the above identification problem via IRM and compare with four existing methods, including subspace identification method (SUBID), deterministic balanced subspace algorithm (DETSS) [137], prediction error method (PEM) [125] and the software for Structured Low-Rank Approximation (SLRA) [134], named STLS. All results from the four existing method can be found in [135]. The comparative results are demonstrates in Table (3.1) in terms of the relative misfit percentage, $M_{rel} = 100\|w - w_d\|_F / \|w_d\|_F$. The comparison verifies that identification results from IRM lead to a significant reduction on fitting error at the cost of more computational time.

3.1.2 Structured H_2 Controller Design

Consider the following LTI system,

$$\begin{array}{c} \dot{x} = Ax + B_1 w + B_2 u \\ z = C_1 x + D_{12} u, \;\; y = C_2 x, \end{array} \tag{1.2}$$

where $x \in \mathbb{R}^n$ is the state variable, $w \in \mathbb{R}^{n_w}$ is the input noise, $u \in \mathbb{R}^m$ is control input, $z \in \mathbb{R}^{n_z}$ is the output to be regulated, and $y \in \mathbb{R}^p$ is the measured output. The goal here is to

Table 3.1: Comparative results for the system identification problem, where $\Delta M_{rel}(\%)$ represents the percentage decrement of M_{rel} for IRM compared to SLRA and 'dist. col.' represents 'distillation column'

NO.	Dataset name in [149]	T	m	p	l	SUBID M_{rel}	DETSS M_{rel}	PEM M_{rel}	STLS M_{rel}	IRM M_{rel}
1	dist. col.	90	5	3	1	0.0089	0.0306	0.0505	0.0029	0.0021
2	dist. col. n20	90	5	3	1	0.4309	0.1187	1.8574	0.0448	0.0257
3	dist. col. n30	90	5	3	1	0.4357	0.1848	7.3600	0.0522	0.0322
NO.	Dataset name in [149]							ΔM_{rel} (%)	STLS Time(s)	IRM Time(s)
1	dist. col.							27.6	0.6010	1086
2	dist. col. n20							42.6	0.1841	407
3	dist. col. n30							38.3	0.6761	449

design a static output feedback controller, $u = Ky$, to minimize the H_2 norm while placing all eigenvalues of the closed-loop system in a stability region described by $\mathscr{D}(p,q,r) = \{s \in \mathbb{C} | p + qs + q^* s^* + r|s|^2 < 0\}$, where q^* and s^* are conjugate variables of q and s, respectively. Additionally, structure constraints on K are considered. For example, if control input u_i does not depend on the measured output y_j, element K_{ij} is set as 0. Thus, $K \in \mathscr{K}$ is used to denote the structure constraint.

In the simulation example, the stability region is denoted by $\mathscr{D}(-1,0,1)$. Work in [109] formulates the structured H_2 controller design problem as a RCOP in the form of

$$\begin{array}{ll} \min_{P,\mu,W,K} & \mathbf{trace}(B_1^T P B_1) \\ s.t. & P \succ \mathbf{0}, \mathbf{W} \succ \mathbf{0}, \mu > \mathbf{0}, \mathbf{K} \in \mathscr{K} \\ & \begin{bmatrix} pP & qP & \mathbf{0} & A_{cl}^T & C_{cl}^T \\ q^*P & rP & \mathbf{0} & -I_n & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{n_z} & \mathbf{0} & -I_{n_z} \\ A_{cl} & -I_n & \mathbf{0} & \mu I_n & \mathbf{0} \\ C_{cl} & \mathbf{0} & -I_{n_z} & \mathbf{0} & \mu I_{n_z} \end{bmatrix} \prec W \\ & \mathbf{rank}(W) = n + n_z, \end{array} \tag{1.3}$$

where $A_{cl} = A + B_2KC_2, C_{cl} = C_1 + D_{12}KC_2$. To verify the improved performance of IRM in solving this type of problem, the penalty function method (PFM) introduced in [109] is used to solve the same problem. The simulation example assumes that $m = 5$, $n = 8$, $n_w = 5$, $n_z = 2$, and $p = 5$. In addition, elements of matrices A, B_1, B_2, C_1, C_2 and D_{12} are randomly generated within a uniform distribution in the range of $[-1, 1]$. Under these assumptions, 50 cases are generated among which 43 have a feasible solution, as the random matrices generation does not guarantee a feasible solution for this type of problem. We claim that for any case if neither IRM or PFM generates a feasible solution, then this case is infeasible. Moreover, we apply a commercial solver LMIRANK based on [161] to further verify the feasibility. It is verified that LMIRANK also fails for all the 7 cases when IRM or PFM cannot generate a feasible solution.

Among the 43 feasible cases, IRM converges for all of the cases while PFM converges for 19 cases. In addition, IRM requires 51 iterations on average to achieve convergence while PFM requires 135 iterations on average. Moreover, among the 19 cases where both methods converge, IRM outperforms PFM in terms of objective value for 15 cases. For the remaining 4 cases, IRM yields slightly larger objective value. Comparison of objective values for the 19 cases is shown in Fig. 3.1. And comparison of penalty term, objective value, and weighting factors for results of one selected case generated from both methods are demonstrated in Fig. 3.2, where penalty term for IRM is e_k with w_k assigned as its corresponding weighting factor, as described in (2.19). For PFM, the penalty term is $\mathbf{trace}(V_{k-1}^T X_k V_{k-1})$ with $\mu_k = \tau\mu_{k-1}$ assigned as its corresponding weighting factor and more details can be found from [109]. Moreover, to verify robustness of IRM in terms of local convergence under random initial values, 50 simulation results are generated using random initial values for the same selected case in Fig. 3.2. All results from IRM converge while none of them converges using random

initial values for PFM. It indicates that random initial guess will not affect the convergence property of IRM. However using random initial guess, instead of initial from relaxation solution, will not improve convergence of PFM. Figure 3.3 demonstrates the objective value distribution of the 50 results with a covariance of 0.29. In summary, IRM demonstrates improved performance in terms of convergence rate, robustness, and cost function values for most of the cases.

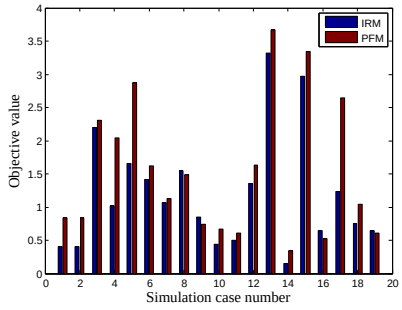


Figure 3.1: Comparison of objective values for 19 converging cases of IRM and PFM.

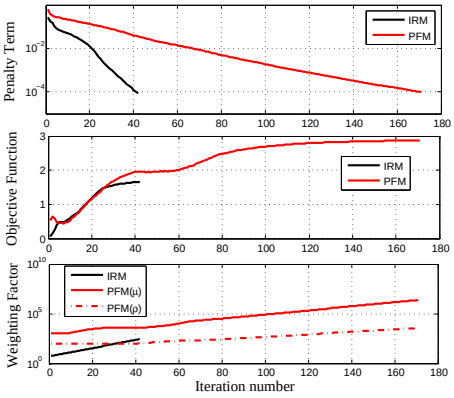


Figure 3.2: Comparison of penalty term, objective value, and weighting factors for results of one selected case generated from IRM and PFM .

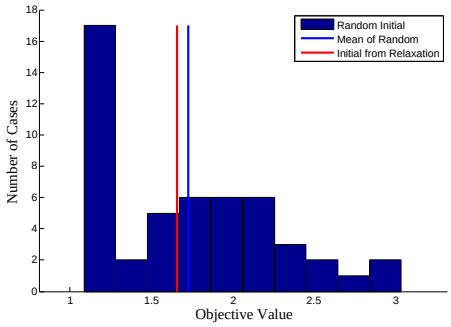


Figure 3.3: Objective value distribution for simulation results using 50 random initial values.

3.1.3 Cardinality Minimization Problem

As discussed in Section 2.3, cardinality minimization problems are classified as a special category of RMP and thus can be solved via IRM. The problem we consider here is formulated as

$$\begin{array}{ll} \min_x & \mathbf{Card}(x) \\ s.t. & Ax \leq b, \|x\|_\infty \leq x^U, \end{array} \tag{1.4}$$

where $x \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$ and $x^U \in \mathbb{R}_+$. For this specific cardinality optimization problem in (1.4), it can be transformed into the following mixed-integer linear programming problem [18]

$$\begin{array}{ll} \min_{x,z} & \mathbf{1}^T z \\ s.t. & Ax \leq b,\ \|x_i\| \leq x^U z_i, \\ & z_i \in \{0,1\},\ \forall i = 1,\ldots,n, \end{array} \tag{1.5}$$

to obtain its global optimal solution via the mixed-integer linear programming solver. We solve (1.4) via IRM and l_1 norm relaxation method and compare both results with the global optimal solution obtained by solving (1.5). Parameters in this example are set as $m = 12$ and $n = 10$. The comparison results for 10 simulation cases are demonstrated in Fig. (3.4). Among them, IRM achieves global optimum for 4 cases and the cardinality value obtained

from IRM for each case is less equal than the corresponding one computed from l_1 relaxation method.

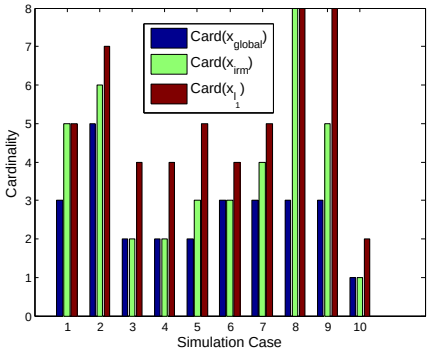


Figure 3.4: Comparison of cardinality value obtained from global optimum, IRM and l_1 relaxation method.

3.2 Applications of IRM for RMPs

3.2.1 Matrix Completion Problem

Matrix completion problems require to reconstruct a matrix given a set of entries to improve a predefined performance index. One commonly used performance index is the rank of the reconstructed matrix and lower rank is generally preferred. Methodologies in matrix completion have been applied in machine learning [2], i.e., low-rank covariance matrix recovery with partial of the entries obtained from observation. Another application of matrix completion is in the recommendation system to recommend preferred items to customers according to their shopping/rating records. In general, information in these systems is incomplete and matrix completion is required to find the missing information. Assume matrix X has a set of given entries, denoted as $X_{I_k,J_k} = v_k$, the matrix completion

problem is formulated as

$$J = \min_X \mathbf{rank}(X)$$
$$s.t. \quad X_{I_k,J_k} = v_k, k = 1,\ldots,p, \tag{2.6}$$

where p is the cardinality of the given entries.

A particular example, image reconstruction, is used here to test IRM and compare it with the existing methods. Three red letters 'ISU' on a blue background, shown in Fig. 3.5, has 21 rows and 36 columns (756 elements) and two different nonzero numbers are used to represent the red and blue color. The matrix representing the real image has 5 distinct rows and thus its rank is 5. Intuitively, the cardinality of the given entries, denoted as p in (2.6), will affect the recovery result. Therefore, the image reconstruction is simulated for different number of given entries, varying from $p = 100$ to $p = 700$. In the simulation, we use the MATLAB function 'randperm' to randomly generate the index of the given entries. IRM shows its advantages in three aspects. Firstly, Fig. 3.6 demonstrates the relative error of recovered matrices from different methods where IRM has the smallest relative error. The relative error is defined by the relative Frobenius norm, denoted as, $\|X - X_0\|_F / \|X_0\|_F$. Secondly, it is verified in Fig. 3.7 that IRM can always generate a matrix with lower or equivalent rank compared to the best solution obtained from other methods, especially in the cases when less number of entries are given. Thirdly, although trace heuristic method and log-det method can recover the matrix when p is relatively large, i.e. $p = 500$, the quality of reconstructed image is poor when p is relatively small. However, IRM can reconstruct higher quality images even with fewer number of given entries, as demonstrated in Fig. 3.8.

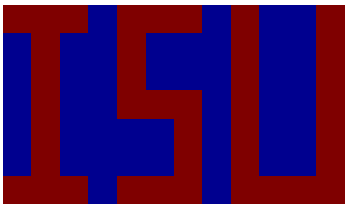


Figure 3.5: Test image. The corresponding matrix has 21×36 elements and the rank is 5.

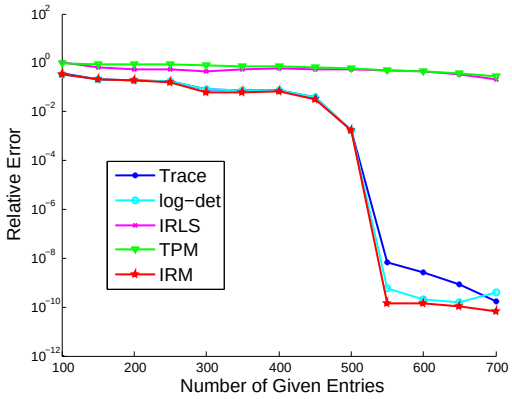


Figure 3.6: Relative error comparison of 5 methods under different number of given entries (the relative error is in logarithmic scale).

3.2.2 Output Feedback Stabilization Problem

Consider the following linear time-invariant (LTI) continuous dynamical system

$$\dot{x} = Ax + Bu,\ y = Cx, \tag{2.7}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^p$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{p\times n}$. For system (2.7) and a given number $k \leq n$, the stabilizing control law is defined as

$$\dot{z} = A_K z + B_K u,\ u = C_K z + D_K y, \tag{2.8}$$

where $z \in \mathbb{R}^k$, $A_k \in \mathbb{R}^{k\times k}$, $B_K \in \mathbb{R}^{k\times m}$, $C_K \in \mathbb{R}^{m\times k}$, and $D_K \in \mathbb{R}^{m\times p}$. Existence of the kth order control law can be determined by solving a RMP with linear matrix constraints [57].

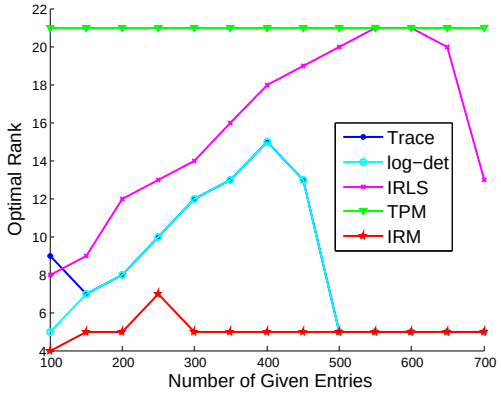


Figure 3.7: Comparison of optimal rank from five methods under different numbers of given entries.

Lemma 3.2.1. *(Corollary 1. in [57]) There exists a stabilizing output feedback law of order k if and only if the global minimum objective of the program*

$$\begin{aligned} \min_{R,S,\gamma} \quad & \mathbf{rank}\left(\begin{bmatrix} \gamma R & I_n \\ I_n & \gamma S \end{bmatrix}\right) \\ s.t. \quad & AR + RA^T \prec BB^T \\ & A^T S + SA \prec C^T C \\ & \gamma > 0, \begin{bmatrix} \gamma R & I_n \\ I_n & \gamma S \end{bmatrix} \succeq \boldsymbol{0}, \end{aligned} \tag{2.9}$$

is less than or equal to $n+k$, where $R \in \mathbb{S}^n$ and $S \in \mathbb{S}^n$.

By multiplying the first two linear matrix inequalities by γ and letting $W_1 = \gamma R$, $W_2 = \gamma S$, problem in (2.9) is converted to

$$\begin{aligned} \min_{W_1,W_2,\gamma} \quad & \mathbf{rank}\left(\begin{bmatrix} W_1 & I_n \\ I_n & W_2 \end{bmatrix}\right) \\ s.t. \quad & AW_1 + W_1 A^T \prec \gamma BB^T \\ & A^T W_2 + W_2 A \prec \gamma C^T C \\ & \gamma > 0, \begin{bmatrix} W_1 & I_n \\ I_n & W_2 \end{bmatrix} \succeq \mathbf{0}, \end{aligned} \tag{2.10}$$

where constraints are linear functions with respect to W_1, W_2, and γ. Using the above converted formulation, the stabilizing output feedback law can be determined via the

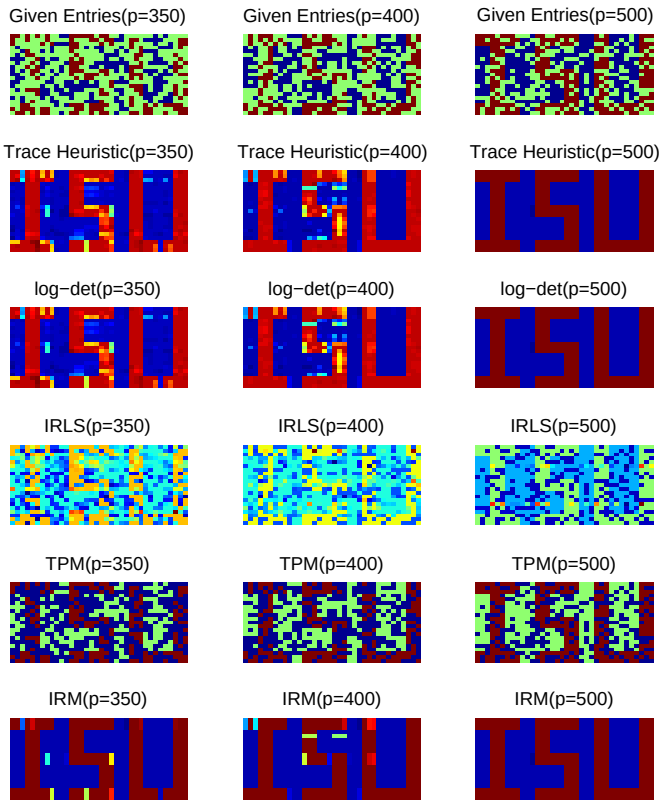


Figure 3.8: Comparison of matrix completion results from five methods with the number given entries $p = 350$, 400, and 500.

proposed IRM approach. In the simulation, we assume $n = 10$, $m = 4$, $p = 5$ and randomly generate matrices A, B, and C for 100 simulation cases. We compare the results obtained from five methods, including IRM, nuclear norm heuristic method (denoted as Trace), log-det heuristic method (denoted as log-det), the iterative reweighted least square (IRLS) method, and trace penalty method (TPM). Among the 100 cases, IRM obtains lower rank solution than the other methods for 79 cases. In addition, IRM generates the same lowest rank solution compared to the best solution of the other method for 18 cases. However, IRM cannot find the best solution for 3 cases. Although the IRM method converges to a local

optimum, the results indicate that it yields significantly improved performance in solving most of the RMPs. Figure 3.9 provides comparison of the computed rank obtained from the five methods for the 100 cases. Moreover, for one of the 100 cases, Table 3.2 shows the corresponding computation time, number of iteration, and optimal rank obtained from the five methods. It is obvious that IRM consumes more computation time to achieve a lower rank solution compared to the four existing methods. The trade-off between computation time and optimal value can be illustrated therein. Figure 3.10 demonstrates the value of e_k at each iteration of IRM for the same case, which verifies that convergence to the constrained rank is achieved within a few iterations under a stopping threshold $\varepsilon = 5e-5$.

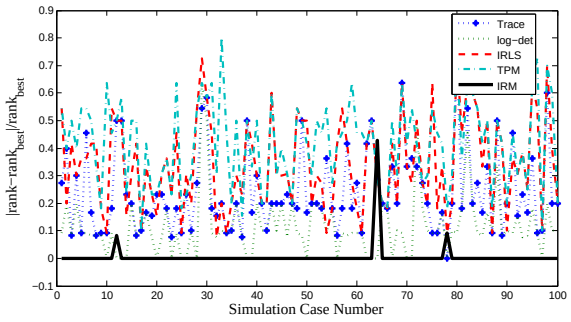


Figure 3.9: Comparison of rank value, where rank_{best} represents the lowest rank value obtained from the five methods.

Table 3.2: Comparison of computation time, number of iteration, and computed rank for one case of the output feedback stabilization problem.

Method	Time (s)	No Iter.	Time per Iter.	Rank
Trace	0.429	1	0.429	14
log-det	3.196	10	0.319	13
TPM	1.789	1	1.789	18
IRLS	7.163	20	0.358	17
IRM	57.862	16	3.616	12

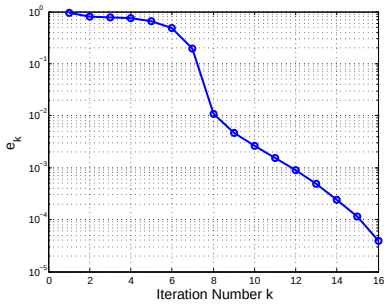


Figure 3.10: IRM convergence history for one case of the output feedback stabilization problem.

3.3 Application of IRM for QCQP

3.3.1 Polynomial Optimal Control Problems

Studies on optimal control can be traced back to several centuries ago [30]. For certain types of optimal control problems with linear affine equality dynamics, convex performance index and constraints, they can be converted to convex optimization problems and be solved through the interior point method [28]. The computing time increases polynomially with regard to the dimension of the unknown variable space. However, optimal control problems are nonconvex in general and thus NP-hard to solve.

The class of optimal control problem we are examine is polynomial optimal control problems (POCPs) where the performance index, dynamics, and constraints are polynomials with respect to the control and states variables. The applications of POCPs can be found in many areas. One typical example is the path planning problems for aerospace and robotic missions under constraints, i.e., no-flying-zone and/or collision avoidance [23]. In these problems, the constraints such as the collision avoidance can be formulated as quadratic inequalities in terms of the Euclidean distance. POCPs have also been applied in orbital transfer and launch ascent problems [124, 140]. More applications of POCPs can be found in [82]. After employing discretization technique, a POCP is converted into a polynomial programming problem. Based on the fact that every polynomial program can be expressed as an equivalent Quadratically Constrained Quadratic Programming (QCQP) by introducing extra variables and quadratic constraints, Consequently, we will just focus on the QCQP rather than polynomial programming for simplicity.

The POCP can be formulated as follows

$$\begin{aligned} & J = \min \varphi(\mathbf{y}(t_f)) + \int_{t_0}^{t_f} \mathbf{L}(\mathbf{y},\mathbf{u},t)dt \\ s.t. \quad & \dot{\mathbf{y}}(t) = \mathbf{f}(\mathbf{y},\mathbf{u},t) \\ & \mathbf{g}(\mathbf{y},\mathbf{u},t) \leq 0 \\ & \phi(\mathbf{y}(t_0)) \leq 0 \\ & \psi(\mathbf{y}(t_f)) \leq 0, \end{aligned} \tag{3.11}$$

where $\mathbf{y} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^m$ is the control input, $\mathbf{g}$ is a sets of functionals representing the constraints on the states and controls, and ϕ and ψ are functions representing the constraints on the boundary points. Specially, all the functionals/functions in (3.11) are polynomials and thus the problem is named as POCP. In this paper, only fixed-final-time

is considered, as the free-final-times problems can be converted to fixed-final-time ones. For example, by introduce a new unknown variable t_f, the normalized time is determined by $\tau = (t - t_0)/t_f$ and terminates at $\tau(t_f) = 1$. Such conversion leads to a fixed-final-time problem. With the new unknown variable t_f, the order of the polynomials representing the dynamics will add by one. But they are still polynomials and can be classified as (3.11). Moreover, in some special cases [124], the fixed-final-time problems might yield the same solution as free-final-time when the terminal time is chosen sufficiently large.

After discretization, problem in (3.11) can be transformed into be a polynomial programming problem in the form of

$$\begin{aligned} & J = \min_{\mathbf{x}} p_0(\mathbf{x}) \\ s.t. \quad & p_i(\mathbf{x}) \leq 0, i \in \mathscr{I}, \end{aligned} \tag{3.12}$$

where $\mathbf{x}$ is the collection of the states $\mathbf{y}$ and control inputs $\mathbf{u}$ at all discrete time steps and $\mathscr{I}$ is the index set of the constraints. In following, we will describe how to transform a polynomial programming to a QCQP [48].

The major approach to achieve this goal is to introduce new variables and quadratic constraints. For instance, to reduce the maximum order of the polynomials, the following inequality,

$$x^{2n} + (\cdots) \leq 0,$$

can be transformed into

$$u^n + (\cdots) \leq 0, u = x^2.$$

Similarly, the product of multiple terms can be replaced as well. For example,

$$xyz + (\cdots) \leq 0$$

can be converted into

$$ux + (\cdots) \leq 0, u = yz.$$

Through the two conversions introduced above, the problem formulated in (3.12) can be transformed into a QCQP. In the following, we will focus on solving QCQPs.

In this section, the proposed IRM is applied to solve a representative optimal control problem, optimal launch ascent problem (OLAP), to verify the feasibility and effectiveness of the new algorithm.

OLAP is to minimize fuel consumption for orbital transfer of a launch vehicle during the ascent of the upper stage subject to finite thrust and finite burn time. This problem has nonlinear dynamics and terminal constraints. Thus it is used to demonstrate that IRM is capable of solving highly nonlinear optimal control problems with complicated terminal constraints. The mathematical problem formulation is given below.

Let τ be the thrust-acceleration vector of the vehicle and define it as

$$\tau = \frac{\mathbf{T}}{\mathbf{m}}, \tag{3.13}$$

where T is the thrust vector and m is the current mass of the vehicle normalized by the initial mass. The dynamics of McCue's problem is formulated as

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{V} \\ \dot{\mathbf{V}} &= -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} + \tau \\ \dot{m} &= -\frac{1}{I_{sp}g_0}m\sigma, \end{aligned} \tag{3.14}$$

where $\mathbf{r}$, $\mathbf{V}$ and m are the position vector, velocity vector, and the mass of the vehicle, respectively. In addition, g_0 is the gravitational acceleration at R_0 and I_{sp} is the specific impulse of the rocket engine. Due to the large magnitude range of the state and control

variables, they are normalized to obtain numerical stability. For example, $\mathbf{r}$ and $\mathbf{V}$ are normalized by R_0 and $\sqrt{g_0 R_0}$, respectively, and equation in (3.14) is differentiated with respect to time normalized by $\sqrt{R_0/g_0}$. The state vector is represented by $\mathbf{x} = [\mathbf{r}^T, \mathbf{V}^T, m]^T \in \mathbb{R}^7$ and the control vector is $\mathbf{u} = [\tau^T, \sigma]^T \in \mathbb{R}^4$. Two constraints on the thrust acceleration are expressed as

$$\begin{aligned} 0 \leq m\sigma \leq T_{max} \\ \|\tau\| \leq \sigma. \end{aligned} \tag{3.15}$$

The performance index representing the propellant consumption is determined by

$$J = \int_{t_0}^{t_f} \sigma dt. \tag{3.16}$$

The final orbit is one with $e = 0.05$, perigee altitude of 400 km, and $i = 51.6^o$. Also, the insertion point to the target orbit is required to be at the perigee, i.e., $\nu = 0^o$. The rest of the orbital elements, i.e., the arguments of the ascending node and the argument of perigee are set as free. The constraints on the terminal states to approach the target orbit are expressed as

$$\begin{aligned} \mathbf{r}_f^T \mathbf{r}_f - (r_f^*)^2 = 0 \\ \mathbf{V}_f^T \mathbf{V}_f - (\mathbf{V_f^*})^\mathbf{2} = \mathbf{0} \\ [0 \;\; 0 \;\; 1](\mathbf{r}_f \times \mathbf{V}_f) = h_f^* cos(i^*) \\ \mathbf{r}_f^T \mathbf{V}_f = 0. \end{aligned} \tag{3.17}$$

where $\mathbf{r}_f^*$ and $\mathbf{V}_f^*$ are the required radius (corresponding to an altitude of 400 km) and velocity at perigee of the specific orbit, which can be computed from the three given orbital elements of the final orbit. The third constraint in (3.17) specifies the final inclination,

$i^* = 51.6^o$ deg, where $h_f^* = r_f^* V_f^*$ is the magnitude of the angular momentum for the final orbit at the perigee. The last constraint in (3.17) defines that the flight path angle at the perigee is zero. The other simulation parameters are listed in the following. The initial mass of the launch vehicle is $70,000$kg, the initial thrust-to-weight ratio of the upper stage is 1.45, and the flight time is 800s. Moreover, the orbital elements of the initial orbit are determined, i.e., $a_0 = 15,374.75km, e_0 = 0.58, i_0 = 50.2^o, \Omega_0 = \omega_0 = \nu_0 = 0^o$. As stated before, new variables need to be introduced to transform the above POCP into a QCQP. Let

$$\begin{aligned} R^2 &= \|\mathbf{r}\|_2^2 \\ N_2 &= R^2 \\ N_1 &= N_2 R, \end{aligned} \tag{3.18}$$

where R, N_1, N_2 are introduced are at each discrete time step. Consequently, the new state vector is denoted by $\mathbf{x'}^T = (\mathbf{x}^T, R, N_1, N_2)$. If the trajectory is discretized by N nodes, we will have $14N$ unknown variables. In addition, (3.17) and (3.18) will be quadratic constraints. Then the OLAP is cast as a QCQP through the above transformations.

Noteworthily, there is an implicit constraint on the state of OLAP. To simply the Earth as a point of mass, we need to add a constraint to guarantee that the distance between the launch vehicle and the center of earth is no less than the radius of the earth. Mathematically, it can be expressed by

$$||\mathbf{r}|| \geq \mathbf{R_0},$$

in the form of a nonconvex quadratic constraint.

OLAP aims to find the optimal control inputs to minimize the propellant consumption (i.e., maximize the final mass) subject to dynamics and terminational constraints.

Applying the IRM algorithm described above, we obtain the following results. The terminal position and velocity vectors are $\mathbf{r}_f = []^T \times 10^3 km$ and $\mathbf{V}_f = []^T km/s$, respectively, and from transformation, the other two unknown orbital elements are $\Omega_f =^\circ$ and $\omega_f =^\circ$. Additionally, it can be verified that all of the terminal constraints are satisfied. Moreover, the convergence history of the second largest eigenvalue of the rank-one matrix is given in figure (3.11). It indicates that λ_{n-1} converges to zeros in a few iterations, which means that the rank-one constraint is strictly satisfied. Figure (3.12) and figure (3.13) demonstrate the time histories of the position and velocity vector. In addition, Figure (3.14) shows the time history of the mass of the space vehicle. The terminal mass is $1561.3kg$ and the overall propellant consumption is $438.7kg$, which indicates a signification amount of fuel reduction compared to the fuel consumption, $644.47kg$, obtained in [124].

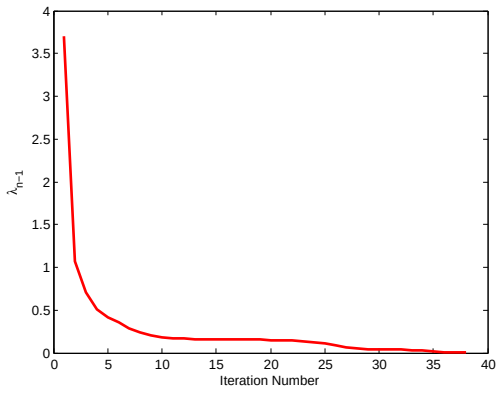


Figure 3.11: The second largest eigenvalue versus the iteration number.

3.3.2 Spacecraft Attitude Control Under Constrained Zones

Reorientation maneuvers of spacecraft have been frequently executed in space missions. To prevent sensitive instruments equipped on spacecraft from exposure to some celestial

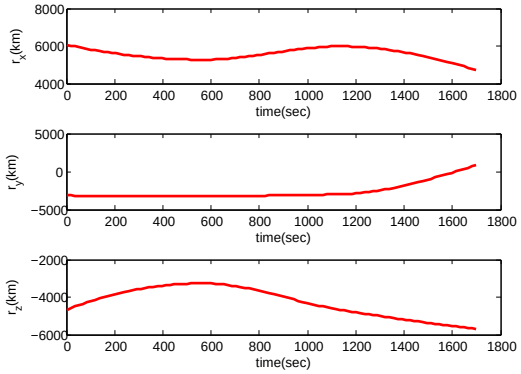


Figure 3.12: Position vector versus time.

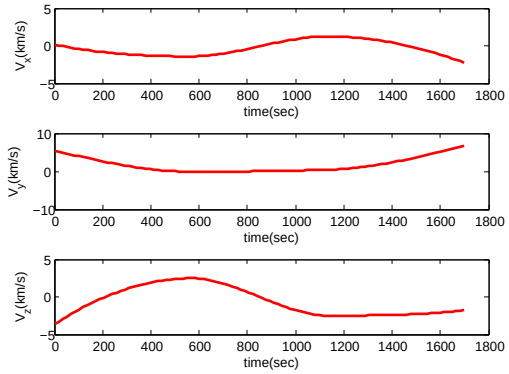


Figure 3.13: Velocity vector versus time.

objects, there are specified forbidden zones during the reorientation procedure. For example, infrared telescope and interferometer must be protected to avoid being directly exposed to the sun or other bright objects. Since fuel consumption is concerned during the space missions, a fuel saving attitude control strategy incorporating attitude constraints plays an important role toward mission success.

Although the reorientation problems without attitude constraints has been widely investigated [199, 206], articles with attitude constraints can only be found in a limited number

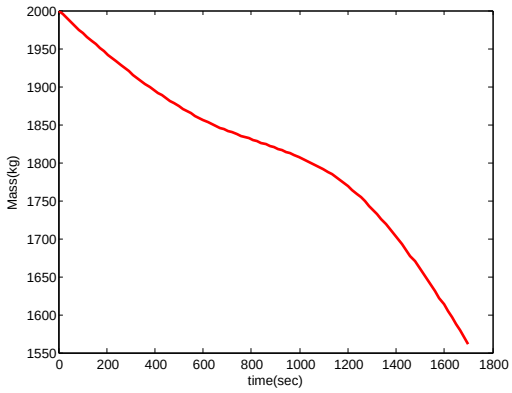


Figure 3.14: Mass of the vehicle versus time.

of literatures. For example, McInnes solved constrained attitude control problem using an artificial potential function [141]. However, due to the involvement of Euler angle representations, singularities may be generated when planning the attitude path. Quaternions have been introduced as mathematical tools for calculation involving three-dimensional (3D) rotations to avoid singularity and reduce expensive computational load created by Euler angle expressions [87]. As alternative and powerful tools for representing object orientation, quaternions have been acknowledged as playing indispensable role in dynamical systems due to their unambiguous, unencumbered, and computationally-efficient features [113]. In this paper, we use unit quaternions to represent attitude dynamics and constraints

An alternative approach for optimal attitude control with constraints is to employ heuristic search algorithms, which are quite time-consuming [41, 71, 111]. Spindler [179], Hablani [84], and Frakes et al. [70] have proposed geometry based approaches to handle this type of problem. Their methods rely on the geometric relations between direction of instrument's boresight and the celestial object, which leads to a feasible attitude trajectory. However, their method is not applicable to the cases where multiple forbidden zones are

considered. Recently, Lee and Mesbahi have solved optimal attitude control problem with multiple forbidden zones via logarithmic barrier potential formulation and nonlinear programming (NLP) solver [118]. However, NLP method generally generates local optimal solutions, which are dependant on the initial guess of unknown variables.

In this paper, we reformulate the optimal attitude control problem with constraints as a nonconvex quadratically constrained quadratic programming (QCQP) problem based on unit quaternions. Efforts on solving nonconvex QCQP problem have been focused on finding the bounds on the optimal values by linear or semidefinite relaxation [4]. Although randomization and linearization have been used to find the approximate solution, none of them guarantees the optimality of the solution. Another approach is the branch and bound method which can find the global optimal solution of nonconvex QCQP [180]. However, when size of the problem increases, this method is time consuming. In this paper, based on the semidefinite relaxation, we transfer the original QCQP problem into a semidefinite programming (SDP) problem with rank one constraint on the unknown symmetric matrix. We then focus on finding this rank one matrix by using an iterative rank minimization (IRM) approach.

3.3.2.1 Problem Formulation

The optimal spacecraft attitude control problem considered in this paper is to find the optimal control torque $\mathbf{u} \in \mathbb{R}^3$ to maneuver the spacecraft with minimum control efforts while satisfying a set of constraints over time interval $t \in [t_0, t_f]$. The constraints include boundary conditions, rotational dynamics, unit quaternion kinematics, and attitude forbidden zones.

Firstly, we assume the initial and terminal conditions of the spacecraft angular velocity in the body frame, denoted as $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^T \in \mathbb{R}^3$, and attitude orientation represented

by unit quaternions, denoted as $\mathbf{q} = [q_1, q_2, q_3, q_4]^T \in \mathbb{R}^4$, $\|\mathbf{q}\| = 1$, are given. The rotational dynamics of a rigid body is expressed as

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \mathbf{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \mathbf{u}, \tag{3.19}$$

where $\mathbf{J} = diag(J_1, J_2, J_3)$ represents the moment of inertia matrix of the spacecraft in the body frame. The unit quaternion kinematics is given by

$$\dot{\mathbf{q}} = \frac{1}{2}\Omega\mathbf{q}, \tag{3.20}$$

where $\Omega(t) = \begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix}$. In addition, the magnitude of the control torque and the angular velocity is constrained by $|\mathbf{u}| \leq \beta_u$, $|\boldsymbol{\omega}| \leq \beta_\omega$, $\forall t \in [t_0, t_f]$.

Last but not the least, the attitude is constrained by a set of forbidden zones. Orientation of spacecraft within those zones will expose certain type of sensitive instruments to bright celestial objects, such as the Sun. Consequently, the instruments may be damaged or have low accuracy. Figure 3.15 demonstrates a forbidden zone example for a sensor installed on the spacecraft. The attitude constraint excluding from a forbidden zone is composed of the boresight vector $\mathbf{y}$ and constrained angle θ. For a specified unit vector $\mathbf{x}$, if $\mathbf{x}$ is expected to be outside the sight of the sensor, we will have

$$\cos(\theta) \geq \mathbf{x} \cdot \mathbf{y}. \tag{3.21}$$

Work in [110] has formulated this type of attitude constraints as

$$\mathbf{q}^T M_f \mathbf{q} = \mathbf{q}^T \begin{bmatrix} A & b \\ b^T & d \end{bmatrix} \mathbf{q} \leq 0, \tag{3.22}$$

where

$$A = \mathbf{x}\mathbf{y}^T + \mathbf{y}\mathbf{x}^T - (\mathbf{x}^T\mathbf{y} + \cos\theta)\mathbf{I}_3$$

$$b = -\mathbf{x} \times \mathbf{y}, d = \mathbf{x}^T \mathbf{y} - \cos\theta. \tag{3.23}$$

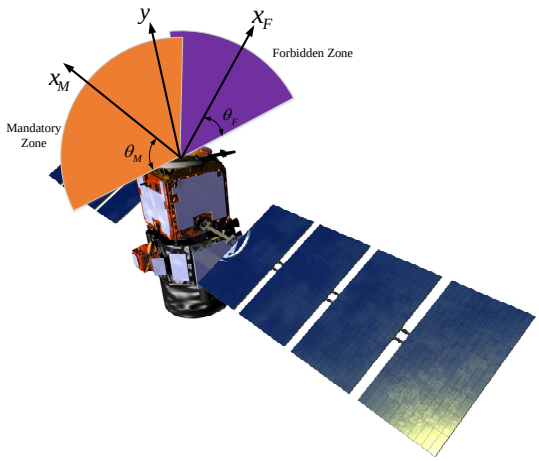


Figure 3.15: Example of a attitude forbidden zone.

In summary, the optimal control problem to minimize total control efforts for for spacecraft reorientation with constraints can be formulated as

$$\min_{\mathbf{u},\omega,\mathbf{q}} \int_{t_0}^{t_f} \mathbf{u}^T \mathbf{u}\, dt \tag{3.24}$$

$$s.t. \qquad \mathbf{J}\dot{\boldsymbol{\omega}}(t) = \mathbf{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \mathbf{u}$$

$$\dot{\mathbf{q}}(t) = \tfrac{1}{2}\Omega(t)\mathbf{q}(t)$$

$$\|\mathbf{q}\| = 1$$

$$\mathbf{q}^T M_{f_l} \mathbf{q} \leq 0, l = 1, \cdots, n,$$

$$\mathbf{q}^T M_{m_s} \mathbf{q} \geq 0, s = 1, \cdots, h,$$

$$|u| \leq \beta_u, |\boldsymbol{\omega}| \leq \beta_\omega$$

$$\boldsymbol{\omega}(t_0) = \boldsymbol{\omega}_0, \boldsymbol{\omega}(t_f) = \boldsymbol{\omega}_f, \mathbf{q}(t_0) = \mathbf{q}_0, \mathbf{q}(t_f) = \mathbf{q}_f$$

where $\boldsymbol{\omega}_0$ and $\mathbf{q}_0$ represent initial condition of angular velocity and attitude orientation, respectively, $\boldsymbol{\omega}_f$ and $\mathbf{q}_f$ represent final condition of angular velocity and attitude orientation, respectively. n and h represents the number of forbidden zones and mandatory zones, respectively.

3.3.2.2 Discretization

The first step toward finding the optimal solution is to utilize the discretization technique to reformulate the above continuous NLP problem as a QCQP problem. The QCQP formulation will allow us to employ recent breakthroughs in semidefinite programming to obtain global optimal solution, which will be described in §IV.

In the discretization procedure, the continuous equations in (3.22)-(3.24) are discretized into a series of segments represented by $\mathbf{z}(k) = [\mathbf{u}(k)^T, \boldsymbol{\omega}(k)^T, \mathbf{q}(k)^T]^T \in \mathbb{R}^{10}$ at each node k, $k = 0, \cdots, N$. The time step between two adajance nodes is denoted as $\Delta t = (t_f - t_0)/N$. By applying the trapezoidal integration rule, the differential equations in (3.19) and (3.20) can be integrated as

$$\mathbf{J}\frac{\boldsymbol{\omega}(k+1)-\boldsymbol{\omega}(k)}{\Delta t} = \frac{1}{2}(\mathbf{J}\boldsymbol{\omega}(k+1)\times\boldsymbol{\omega}(k+1)+\mathbf{J}\boldsymbol{\omega}(k)\times\boldsymbol{\omega}(k)) + \frac{1}{2}(\mathbf{u}(k+1)+\mathbf{u}(k)), \tag{3.25}$$

$$\frac{\mathbf{q}(k+1)-\mathbf{q}(k)}{\Delta t} = \frac{1}{4}(\Omega(k+1)\mathbf{q}(k+1)+\Omega(k)\mathbf{q}(k)). \tag{3.26}$$

More explicitly, combining the expanded terms of $\mathbf{J}$, $\boldsymbol{\omega}$ and $\mathbf{u}$, the above two equations can be rewritten as

$$\begin{aligned}
J_1\frac{\omega_1(k+1)-\omega_1(k)}{\Delta t} - \frac{1}{2}(J_2-J_3)[\omega_2(k+1)\omega_3(k+1)+\omega_2(k)\omega_3(k)] &= \frac{1}{2}[u_1(k+1)+u_1(k)] \\
J_2\frac{\omega_2(k+1)-\omega_2(k)}{\Delta t} - \frac{1}{2}(J_3-J_1)[\omega_3(k+1)\omega_1(k+1)+\omega_3(k)\omega_1(k)] &= \frac{1}{2}[u_2(k+1)+u_2(k)] \\
J_3\frac{\omega_3(k+1)-\omega_3(k)}{\Delta t} - \frac{1}{2}(J_1-J_2)[\omega_1(k+1)\omega_2(k+1)+\omega_1(k)\omega_2(k)] &= \frac{1}{2}[u_3(k+1)+u_3(k)],
\end{aligned} \tag{3.27}$$

and

$$\begin{aligned}
\frac{q_1(k+1)-q_1(k)}{\Delta t} &= \frac{1}{4}[\omega_3(k)q_2(k)-\omega_2(k)q_3(k)+\omega_1(k)q_4(k) \\
&\quad +\omega_3(k+1)q_2(k+1)-\omega_2(k+1)q_3(k+1)+\omega_1(k+1)q_4(k+1)] \\
\frac{q_2(k+1)-q_2(k)}{\Delta t} &= \frac{1}{4}[-\omega_3(k)q_1(k)+\omega_1(k)q_3(k)+\omega_2(k)q_4(k) \\
&\quad -\omega_3(k+1)q_1(k+1)+\omega_1(k+1)q_3(k+1)+\omega_2(k+1)q_4(k+1)] \\
\frac{q_3(k+1)-q_3(k)}{\Delta t} &= \frac{1}{4}[\omega_2(k)q_1(k)-\omega_1(k)q_2(k)+\omega_3(k)q_4(k) \\
&\quad +\omega_2(k+1)q_1(k+1)-\omega_1(k+1)q_2(k+1)+\omega_3(k+1)q_4(k+1)] \\
\frac{q_4(k+1)-q_4(k)}{\Delta t} &= \frac{1}{4}[-\omega_1(k)q_1(k)-\omega_2(k)q_2(k)-\omega_3(k)q_3(k) \\
&\quad -\omega_1(k+1)q_1(k+1)-\omega_2(k+1)q_2(k+1)-\omega_3(k+1)q_3(k+1)].
\end{aligned}$$

After discretization, $\mathbf{z} = [\mathbf{z}(0)^T, \cdots, \mathbf{z}(k)^T, \cdots, \mathbf{z}(N)^T]^T \in \mathbb{R}^{10(N+1)}$ is the collection of unknown variable vector to be determined. At each time step, there are 8 quadratic equality constraints, including the 7 equation from Eq. (3.27) and Eq. (3.28) and a $norm-1$ constraint of the unit quaternion. In addition, there are $n+h$ quadratic inequality constraints, where n is the number of the forbidden zones and h is the number of the mandatory zones.

Consequently, the optimal control problem can be reformulated as

$$\begin{aligned}
&J = \min_{\mathbf{z}} \mathbf{z}^T M_u \mathbf{z} \\
s.t. \quad &\mathbf{z}^T M_{\omega_i} \mathbf{z} + p_{\omega_i}^T \mathbf{z} + r_{\omega_i} = 0, i = 1, \cdots, 3(N+1) \\
&\mathbf{z}^T M_{\omega \mathbf{q}_j} \mathbf{z} + p_{\omega \mathbf{q}_i}^T \mathbf{z} + r_{\omega \mathbf{q}_i} = 0, j = 1, \cdots, 4(N+1) \\
&\mathbf{z}^T M_{\mathbf{q}_k} \mathbf{z} = 1, k = 1, \cdots, N+1 \\
&\mathbf{z}^T M_{f_l} \mathbf{z} \leq 0, l = 1, \cdots, n(N+1) \\
&\mathbf{z}^T M_{m_s} \mathbf{z} \geq 0, s = 1, \cdots, h(N+1) \\
&|\mathbf{z}| \leq U_{\mathbf{z}}
\end{aligned} \tag{3.29}$$

$$\boldsymbol{\omega}(0) = \boldsymbol{\omega}_0, \boldsymbol{\omega}(N) = \boldsymbol{\omega}_f, \mathbf{q}(0) = \mathbf{q}_0, \mathbf{q}(N) = \mathbf{q}_f.$$

The first constraints are from Eq. (3.27) while the second is from Eq. (3.28). The third is the norm-one constraint on the unit quaternion. in addition, the forth is the forbidden zone constraint while the fifth is the mandatory zone constraint. The last two are the magnitude constraints and the boundary condition constraints, respectively.

In the following, two simulation cases are given. One scenario has four forbidden zones and no mandatory zones while the other has three forbidden zones and one mandatory zone. We also compare the result from the proposed method with those obtained from the commercial nonlinear programming solver developed by Tomlab [95] to verify the effectiveness of IRM method.

3.3.2.3 Scenario 1 (four forbidden zones)

In the scenario, the objective is to reorient its telescope attitude with minimum total control efforts while avoiding the four forbidden zones in the rotational configuration space. The four forbidden zones are selected randomly without overlapping with each other. Noteworthily, both initial and terminal attitude should be chosen outside of these specified four zones, to prevent violation of the constraints. The spacecraft is assumed to carry a light-sensitive telescope with a fixed boresight $\mathbf{y}$, defined as $\mathbf{y} = [0, 0, 1]^T$, in the spacecraft body frame. The other simulation parameters are given in Table 3.3. The objective value from IRM is 41.2474 while the one from TomLab is 114.6850, which indicates that IRM is more effectiveness than TomLab in terms of achieving optimality values.

Figures 3.17 to 3.19 demonstrate the time history of control torque, angular velocity and unit quaternion, respectively, from both methods. Moreover, Figure 3.20 presents the trajectory of the telescope pointing vector in the constrained 3-dimensional space. It can

Table 3.3: Simulation Parameters of scenario one

Parameter	Value
J	diag[54,63,59] $kg{\cdot}m^2$
$\|\omega_i\|, i = 1,2,3$	$\leq$ 0.3 rad/s
$\|u_i\|, i = 1,2,3$	$\leq$ 3 rad/s^2
t_f	=20s
Initial Attitude $\mathbf{q}_{t_0}$	$[0.6085,-0.6300,-0.2369,-0.4204]^T$
Terminal Attitude $\mathbf{q}_{t_f}$	$[-0.0238,0.7127,-0.2734,-0.6455]^T$
Instrument boresight	$[0,0,1]^T$
Forbidden zone 1	$\mathbf{x}_1 = [0.1632,-0.9863,0.0250]^T$, $\theta_1 = 40^\circ$
Forbidden zone 2	$\mathbf{x}_2 = [0.0734,0.6938,0.7164]^T$, $\theta_2 = 40^\circ$
Forbidden zone 3	$\mathbf{x}_3 = [-0.5000,0.6634,-0.5567]^T$, $\theta_3 = 30^\circ$
Forbidden zone 4	$\mathbf{x}_4 = [-0.0677,-0.4628,-0.8839]^T$, $\theta_4 = 20^\circ$

be easily seen that the trajectory from IRM is smoother and feasible for implementation compared with that obtained from TomLab solver. Furthermore, Figure 3.16 gives the convergence history of the second largest eigenvalue of matrix $\begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix}$ at each iteration. It's shown that λ_{n-1}, which is represented by r in Eq. (3.33), quickly reduces to zero within a few steps. Figure 3.16 indicates that we obtain a rank one matrix of X within a few iterative steps.

3.3.2.4 Scenario 2 (three forbidden zones and one mandatory zone)

In the scenario, the objective is to reorient the spacecraft with minimum total control efforts while preventing its telescope pointing vector from the three forbidden zones and keeping the antenna in the mandatory zone. The three forbidden zones are selected randomly

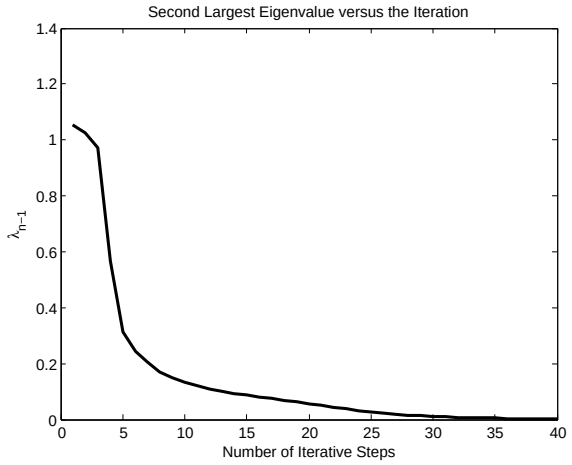


Figure 3.16: The second largest eigenvalue λ_{n-1} at each iterative step(scenario 1).

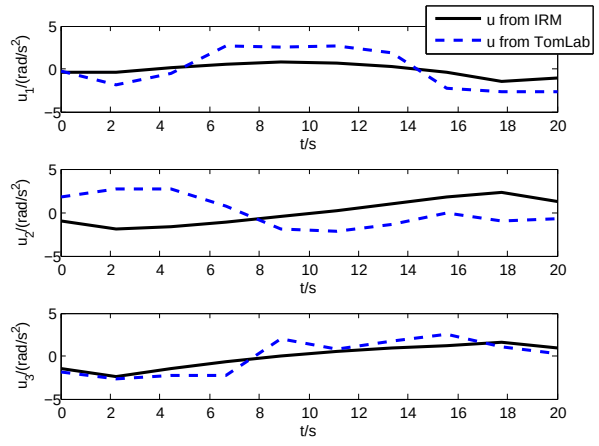


Figure 3.17: Time history of control torque(scenario 1).

without overlapping with each other but may overlap with the mandatory zone. Noteworthily, both initial and terminal attitude should be chosen properly to prevent violation of the attitude

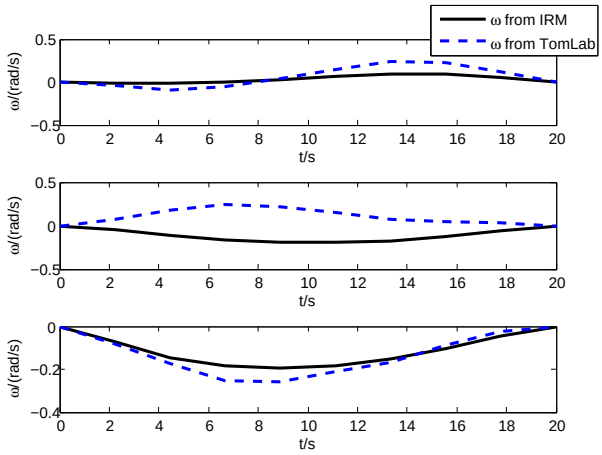


Figure 3.18: Time history of angular velocity(scenario 1).

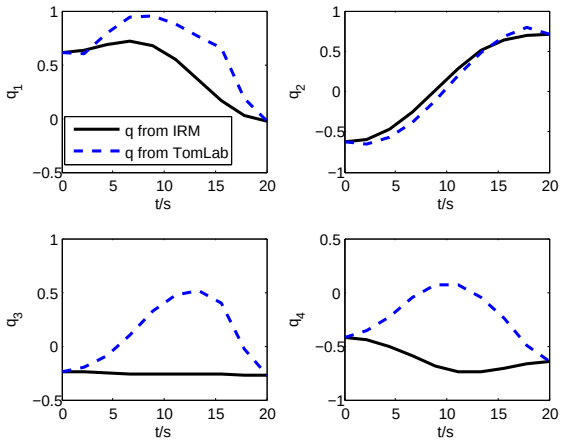


Figure 3.19: Time history of unit quaternion(scenario 1).

constraints. The spacecraft is assumed to carry a light-sensitive telescope with a fixed boresight $\mathbf{y}_F$, defined as $\mathbf{y}_F = [0,1,0]^T$, while the boresight of the antenna is $\mathbf{y}_M = [0,0,1]^T$,

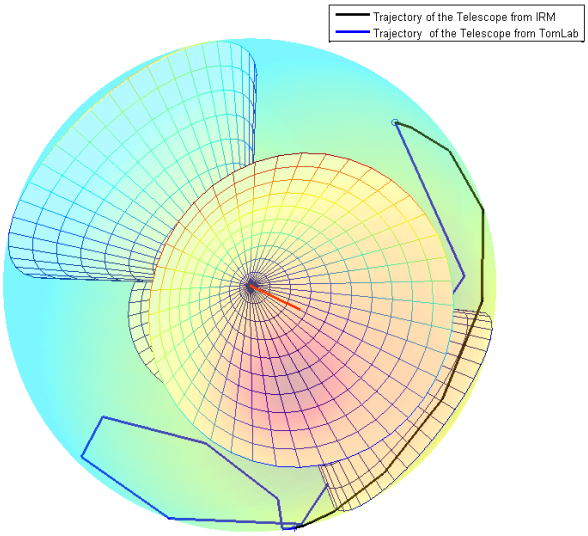


Figure 3.20: Trajectory of the telescope pointing vector in 3-dimensional space. The blue star represents the initial orientation while the blue circle the terminal(scenario 1).

both in the spacecraft body frame. The other simulation parameters are given in Table 3.4.The objective value from IRM is 23.7270, while the one calculated by TomLab solver is 31.7932.

Figures 3.22 to 3.24 demonstrate the time history of control torque, angular velocity and unit quaternion, respectively, from both methods. Moreover, Figure 3.25 presents the trajectory of the telescope and antenna pointing vector in the constrained 3-dimensional space. Again, trajectory from IRM is much smoother and reasonable in the second case.

Furthermore, Figure 3.21 gives the convergence history of the second largest eigenvalue of matrix $\begin{bmatrix} 1 & x^T \\ x & X \end{bmatrix}$ at each iteration. It's shown that λ_{n-1}, which is represented by r in Eq. (3.33), quickly reduces to zero within a few steps. Figure 3.21 indicates that we obtain a rank one matrix of X within a few iterative steps.

Table 3.4: Simulation Parameters of scenario two

Parameter	Value
J	diag[54,63,59] $kg{\cdot}m^2$
$\|\omega_i\|, i = 1,2,3$	$\leq$ 0.3 rad/s
$\|u_i\|, i = 1,2,3$	$\leq$ 2 rad/s^2
t_f	=20s
Initial Attitude $\mathbf{q}_{t_0}$	$[0.81744,0.51592,-0.11618,-0.22831]^T$
Terminal Attitude $\mathbf{q}_{t_f}$	$[0.27536,-0.50637,-0.78252,-0.23542]^T$
antenna boresight	$[0,0,1]^T$
telescope boresight	$[0,1,0]^T$
Mandatory zone 1	$\mathbf{x}_1 = [-0.8138, 0.5483, -0.1926]^T$, $\theta_1 = 70^\circ$
Forbidden zone 1	$\mathbf{x}_2 = [0,-1,0]^T$, $\theta_2 = 40^\circ$
Forbidden zone 2	$\mathbf{x}_3 = [0, 0.8192, 0.5736]^T$, $\theta_3 = 30^\circ$
Forbidden zone 3	$\mathbf{x}_4 = [-0.1220, -0.1397, -0.9827]^T$, $\theta_4 = 20^\circ$

3.3.3 UAV Path Planning

Unmanned Aerial Vehicles (UAVs) have been widely used in the battle field for military operations, such as target attacking, surveillance and reconnaissance. The goal is to replace the roles of manned aircraft in a more efficient and less risky fashion. The advances in autonomous control, signal processing, communication, etc. have enabled the UAVs to make onboard decisions in certain operations without human interactions. However, the requirement for autonomy and decision making will be more challenging when operations are performed in hostile environments, i.e., areas with radar systems. In such scenarios, the susceptibility of UAVs in hostile environments raises requirements for flight path planning to avoid or reduce the possibility of detection by adversarial radars.

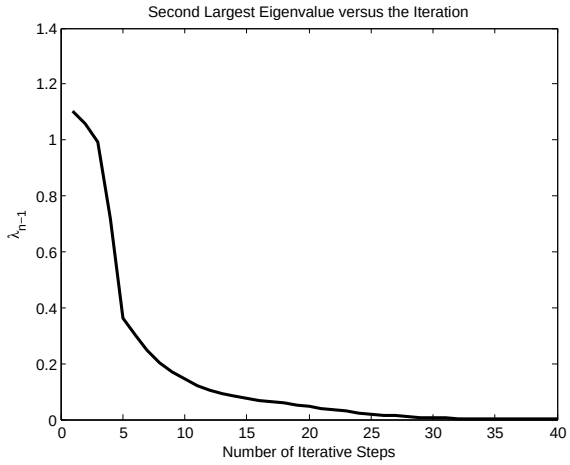


Figure 3.21: The second largest eigenvalue λ_{n-1} at each iterative step(scenario 2).

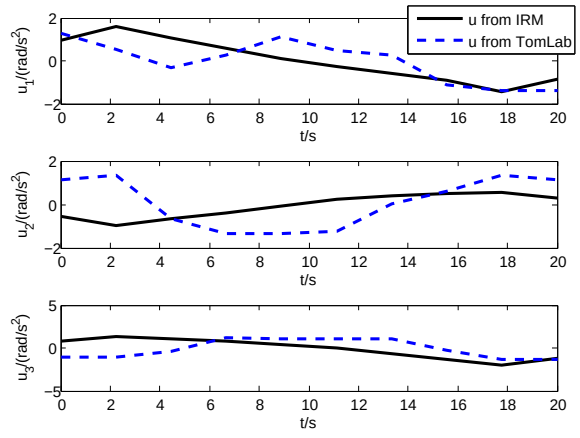


Figure 3.22: Time history of control torque(scenario 2).

Due to the importance of UAV path planning in achieving mission success, many methodologies have been developed, especially for problems with obstacle avoidance

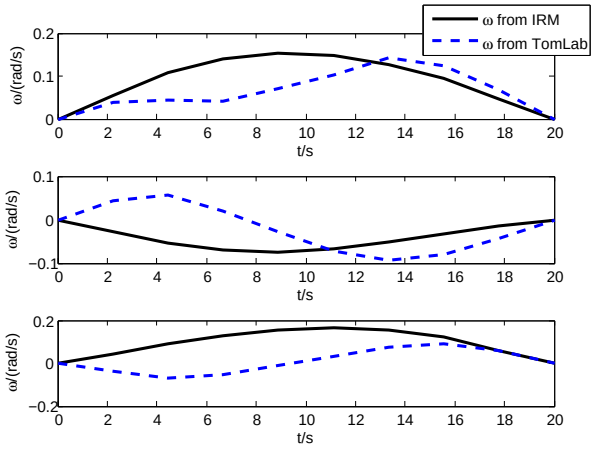


Figure 3.23: Time history of angular velocity(scenario 2).

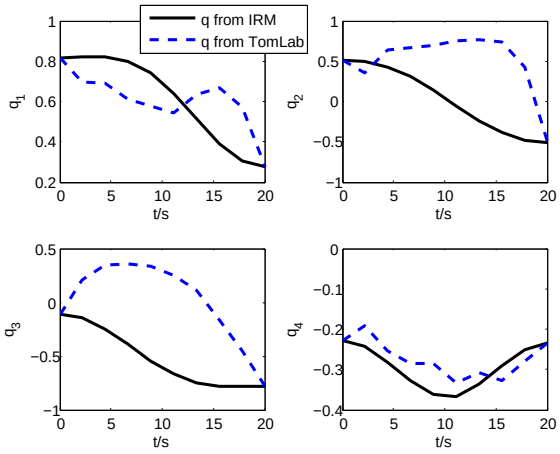


Figure 3.24: Time history of unit quaternion(scenario 2).

constraints [38, 103, 139]. Early work in path planning, such as artificial potential function [25, 108] and randomized sampling algorithms [13, 107, 112, 166], focuses on searching

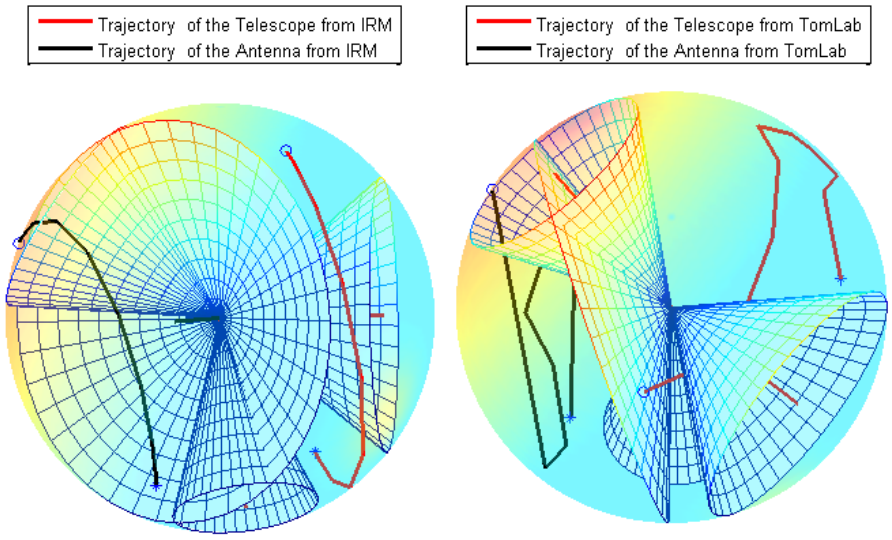


Figure 3.25: Trajectory of the telescope pointing vector(red) and the antenna point vector(black) in 3-dimensional space. The blue star represents the initial orientation while the blue circle the terminal. The cone with black boresight vector represents the mandatory zone while the rest 3 are the forbidden zones(scenario 2).

a feasible path without optimizing the desired path performance, i.e., path length or travel time. Due to the nonconvex nature of obstacle avoidance constraints, the constrained path planning problem to optimize the desired path performance is generally formulated as a nonlinear programming (NLP) problem which can be solved via an NLP solver [80]. However, the solution of an NLP problem is dependent on the initial guess and convergence to a local optimum is generally not guaranteed. When multiple avoidance zones are considered, it is even difficult to find a feasible solution using an NLP. On the other hand, an optimized heuristic method, such as the optimized Rapidly-Exploring Random Tree algorithm, named

as RRT*, has been developed to search for obstacle avoiding paths with asymptotic optimality property [105, 106]. However, the RRT* algorithm itself does not include the flight kinematic constraints.

Inspired by the computational efficiency of convex optimization techniques [28], one approach is proposed to search for the minimum-time path for a UAV flying through a specified area with fixed starting and ending points and multiple avoidance zones, including circular and elliptical zones. The approach is to reformulate the path planning problem as a general/nonconvex Quadratically Constrained Quadratic Programming (QCQP) problem where an iterative rank minimization (IRM) method proposed in our previous work [43, 183] is applied to solve the path planning problem. The IRM method with each iteration formulated as a convex optimization problem has guaranteed convergence to local optimum. Different from the NLP approach, an initial guess of the planned path is not required in IRM.

3.3.3.1 Problem Formulation

The problem of solving a single UAV passing through hostile environments with avoidance zones is illustrated in Fig. 3.30, where the UAV has assigned starting and ending points, denoted as triangles. The avoidance zones have pre-defined locations and shapes. Depending on the requirements of flight mission, the performance index can be assigned accordingly, i.e., minimum flight time.

The flight kinematics of a UAV in two-dimensional space is represented by a single control model in the form of

$$\begin{aligned} \dot{x} &= V\cos\theta \\ \dot{y} &= V\sin\theta \end{aligned} \tag{3.30}$$

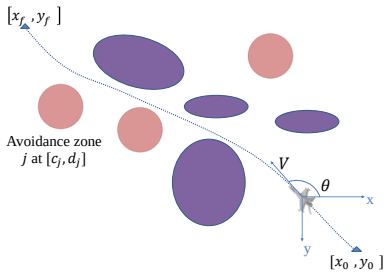


Figure 3.26: Illustration of UAV path planning problem with avoidance zones.

$$\dot{\theta} = u$$

$$|u| \leq u_{max},$$

where x and y are the coordinates, V is the specified cruise speed, θ is the heading angle, and u_{max} is the maximum rate of change of the heading angle. The starting and ending points are specified as $[x_0, y_0]$ and $[x_f, y_f]$. The avoidance zones are represented by ellipses formulated as,

$$\left(\frac{x - c_j}{a_j}\right)^2 + \left(\frac{y - d_j}{b_j}\right)^2 \geq 1, \forall\, j = 1, \ldots, m', \tag{3.31}$$

where $[c_j, d_j]$ represents the center of avoidance zone j, $j = 1, \ldots, m'$, (a_j, b_j) are pre-defined parameters, i.e., the semi-major/semi-minor axes of the elliptical zones, and m' is the number of avoidance zones. The above function can create different elliptical shapes modeling a range of avoidance zones with specified centers. A special case is a circular-shaped zone by setting $a_j = b_j$. For minimum time of flight, the performance index is $J = \int_{t_0}^{t_f} 1 dt$. Consequently, the minimum time path planning problems can be formulated as

$$J = \min_u \int_{t_0}^{t_f} 1 dt$$

$$s.t. \qquad x(t_0) = x_0, y(t_0) = y_0, x(t_f) = x_f, y(t_f) = y_f,$$

$$\dot{x} = V\cos\theta$$

$$\dot{y} = V\sin\theta \tag{3.32}$$

$$\left(\frac{x-c_j}{a_j}\right)^2 + \left(\frac{y-d_j}{b_j}\right)^2 \geq 1, \forall\, j = 1,\ldots,m', \forall x, y, t_0 \leq t \leq t_f$$

$$\dot{\theta} = u$$

$$|u| \leq u_{max}.$$

The above path planning problems formulated by nonlinear equations are difficult to solve. The indirect method requires deriving the necessary conditions for optimality based on Hamiltonian and Euler Lagrange equations. In most cases, the difficulty of guessing the initial adjoint variables and the trigonometric equations involved in the problem formulation make the indirect method infeasible. The direct method, such as using collocation and an NLP cannot guarantee fast convergence to a local optimal solution, or even convergence to a feasible solution, when highly nonlinear equations are included in the constraints and/or an initial guess of the solution is randomly selected. Therefore, two distinct approaches are proposed below to solve the UAV path planning problem posed in (3.32) to improve the computational performance.

The first step in the numerical optimization approach is to convert the nonlinear optimization problem formulated in (3.32) into a general QCQP problem, where the objective is a quadratic function and the constraints are quadratic equalities or inequalities. An iterative convex optimization method is then introduced to solve the general QCQP problem. The novelty of QCQP formulation and its associated iterative method is that it does not involve linearization procedures in the formulation and optimization approach such that it will present errors generated from linearization of a highly nonlinear model.

3.3.3.2 Reformulation of the Path Planning Problems as QCQP Problems

The single conbtrol model described in (3.30) for UAV path planning includes trigonometric functions, which are highly nonlinear and may generate singular matrices in computational operations. The first step is to reformulate the above nonlinear optimization problems as general QCQP problems via a discretization method.

A continuous flight path can be discretized into a series of segments represented by coordinates $[x_h, y_h]$, $h = 1, \ldots, H$, at each node, where H is the number of discrete nodes. By discretization, the change rate of the coordinates can be approximately determined by two adjacent nodes,

$$\dot{x} = \frac{x_{h+1} - x_h}{\Delta t} = V \cos\theta, h = 1, \ldots, H, \tag{3.33}$$

$$\dot{y} = \frac{y_{h+1} - y_y}{\Delta t} = V \sin\theta, h = 1, \ldots, H, \tag{3.34}$$

where Δt is the uniform time interval between two adjacent nodes. The above two equations can be synthesized as

$$(x_{h+1} - x_h)^2 + (y_{h+1} - y_h)^2 = V^2 (\Delta t)^2. \tag{3.35}$$

Differentiating (3.33)-(3.34) leads to

$$\ddot{x} = \frac{x_{h+2} + x_h - 2x_{h+1}}{(\Delta t)^2} = -V\dot{\theta}\sin\theta, h = 1, \ldots, H,$$

$$\ddot{y} = \frac{y_{h+2} + y_h - 2y_{h+1}}{(\Delta t)^2} = V\dot{\theta}\cos\theta, h = 1, \ldots, H.$$

Combining with $\dot{\theta} = u$ and $|u| \leq u_{max}$, the above equations can be synthesized as

$$\dot{\theta}^2 = \left(\frac{x_{h+2} + x_h - 2x_{h+1}}{V(\Delta t)^2}\right)^2 + \left(\frac{y_{h+2} + y_h - 2y_{h+1}}{V(\Delta t)^2}\right)^2 \leq u_{max}^2. \tag{3.36}$$

By introducing an additional variable $t' = \Delta t^2$, the above constraint can be reformulated as a quadratic inequality in the form of

$$(x_{h+2} + x_h - 2x_{h+1})^2 + (y_{h+2} + y_h - 2y_{h+1})^2 \leq V^2 u_{max}^2 t'^2, h = 1, ..., H-2. \tag{3.37}$$

Meanwhile, the avoidance zones are originally formulated as quadratic inequalities. Based on the above reformulation, the path planning problems can be generalized as a nonconvex QCQP problem in the form of

$$
\begin{aligned}
& J = \min_{x,y,\Delta t,t'} (H-1)\Delta t \\
s.t. \quad & x_1 = x_0, y_1 = y_0, x_H = x_f, y_H = y_f, \\
& (x_{h+1}-x_h)^2 + (y_{h+1}-y_h)^2 = V^2(\Delta t)^2, h = 1,...,H-1, \\
& (x_{h+2}+x_h-2x_{h+1})^2 + (y_{h+2}+y_h-2y_{h+1})^2 \le V^2 u_{max}^2 t'^2, h = 1,...,H-2, \\
& \left(\frac{x_h-c_j}{a_j}\right)^2 + \left(\frac{y_h-d_j}{b_j}\right)^2 \ge 1, \forall\, j = 1,\ldots,m', h = 1,\ldots,H, \\
& t' = (\Delta t)^2.
\end{aligned} \tag{3.38}
$$

where x, y are coordinates at all discrete nodes. Together with Δt and t', they are unknown variables to be determined in the new formulation.

Based on the above reformulation, the path planning problem with avoidance zones can be generalized as a nonconvex QCQP problem in the form of

$$
\begin{aligned}
& J = \min_{\mathbf{x}} \mathbf{x}^T Q_0 \mathbf{x} + {a_0}^T \mathbf{x} \\
s.t. \quad & \mathbf{x}^T Q_j \mathbf{x} + {a_j}^T \mathbf{x} \le c_j, \ \forall\ j = 1,\ldots,m \\
& l_{\mathbf{x}} \le \mathbf{x} \le u_{\mathbf{x}},
\end{aligned} \tag{3.39}
$$

where $\mathbf{x} \in \mathbb{R}^n$ is the unknown vector to be determined, $Q_j \in \mathbb{S}^{n\times n}$, $j = 0,\ldots,m$, is an arbitrary symmetric matrix, $c_j \in \mathbb{R}$, $j = 1,\ldots,m$, and $a_j \in \mathbb{R}^n$, $j = 0,\ldots,m$. Moreover, $l_{\mathbf{x}} \in \mathbb{R}^n$ and $u_{\mathbf{x}} \in \mathbb{R}^n$ are the lower and upper bounds on $\mathbf{x}$, respectively. Since $Q_j\,(j = 0,\ldots,m)$ is not necessarily a positive definite matrix, the problem in (3.39) is classified as NP-hard.

In this section, a group of simulation cases are presented to demonstrate the advantages and limitations of the two proposed approaches. In the first case, one avoidance

zone is considered and the path planning problem for this simple case has an analytical solution as a reference to verify the accuracy of proposed approaches. Other cases consider a group of cluttered avoidance zones, which makes the path planning problem more difficult. In both types of cases, the parameters in the IRM method are set as $[x_0, y_0] = [0,0]^T, [x_f, y_f] = [10,10]^T, V = 1, u_{max} = 0.2, w = 1.5$, and $\delta = 1e-4$, which is the threshold for the convergence of r. All cases use 20 discrete nodes to represent the planned path. The corresponding parameters in the refined RRT* method are set as $d_{min} = 0.3$ and $\theta = 150°$. All of the simulation cases are run on a desktop computer with a 3.50 GHz processor and of 16 GB RAM.

3.3.3.3 Case with One Avoidance Zone

The single avoidance zone considered here is a circle centered at $[5,5]$ with radius of 5. The analytical solution for this case yields a minimum flight time of $f_{benchmark} = 17.85$. The optimal solution from IRM is $f_{IRM} = 17.84$. The relative error of the solution from IRM is only 0.066% compared to the benchmark value, which verifies the high accuracy of IRM. Three nonlinear programming solvers, conSolve, NPSOL and SNOPT, embedded in the commercial software, TOMLAB [95], are applied to solve case one. Moreover, we use three different categories of initial guesses for the three solvers, including special guess, i.e., setting all variables to be zero, solution from SDP relaxation, and random initial values. The only combination that generates a local optimum for case one is the SNOPT solver with the special initial guess. The rest combinations cannot generate a smooth or feasible trajectory. The flight time from NLP is 17.84. Therefore, the comparative results indicate that IRM outperforms NLP in generating a convergent solution that does not depend on initial guess. The paths obtained from IRM and NLP are demonstrated in Fig. 3.27(a). It takes 10 iterations for the IRM with threshold $\delta = 1e-4$ converging to the optimal

solution. The overall computation time for IRM is 202.90 seconds. To illustrate the accuracy of simulation results in terms of number of discrete nodes for the IRM method, Table 1 presents the objective value determined from IRM method using different number of discrete nodes for case one. When $H = 20$, the relative error of IRM compared to the analytical solution is only 0.066%. Therefore, in the following simulation cases, 20 nodes are used for discretization.

Table 3.5: Comparison of flight time determined from IRM method using different number of discrete nodes for case one, where the analytical solution yields a flight time of $f_{\text{benchmark}} = 17.85$ seconds.

Number of discrete nodes	10	15	20	25
f_{IRM}	17.80	17.86	17.84	17.85
$\lvert f_{\text{IRM}} - f_{\text{benchmark}} \rvert / f_{\text{benchmark}} \times 100\%$	0.27	0.063	0.066	0.0029

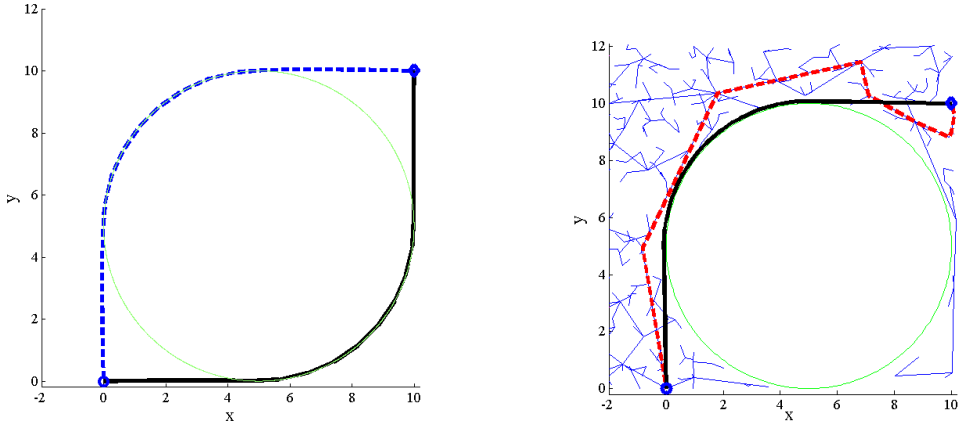
The planned paths from RRT* and refined RRT* are shown in Fig. 3.27(b). The original path from RRT* (dash lines) includes sharp turns at connections of sampling trees. Through refinement steps, the refined path with flight time of $f_{refined} = 18.00$ is very close to the planned path generated from IRM. It takes 0.95 seconds to generate the coarse path from RRT*. The refinement steps take additional 25.15 seconds to refine the coarse path. Although the refined RRT* does not yield the same high accuracy compared to IRM, it has significant reduction on computation time in this simple example. The results and parameters of case one from the above four methods are listed in Table 2.

3.3.3.4 Cases with Multiple Avoidance Zones

A group of cluttered avoidance zones in the shape of circles and ellipses are considered in the second case. The optimal solution from IRM is $f_{IRM} = 14.75$ while the one from

Table 3.6: Comparison for simulation results for case one.

Algorithm	Parameter	time (sec)	f
NLP	SNOPT	1.52	17.84
IRM	$N = 20, w = 1.5, \delta = 1e-4$	202.90	17.84
RRT*	$d_{min} = 0.3$	0.95	N/A
Refined RRT*	$\theta = 150^{\circ}$	26.10	18.00



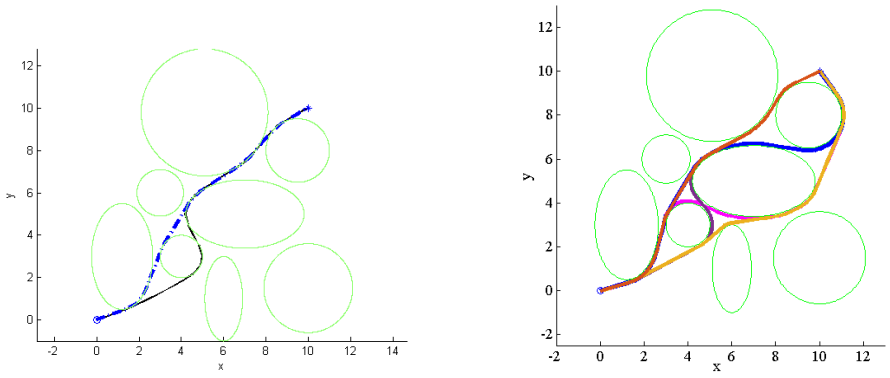
(a) Planned paths generated from IRM (solid) and NLP (dash) for case one.

(b) Planned paths generated from RRT* (dash) and refined RRT* (solid) for case one.

Figure 3.27: Comparative results of planned path for case one.

NLP is $f^*_{NLP} = 16.02$. Figure 3.28(a) shows the planned paths from IRM and NLP for this case. Again, the only combination that leads to a feasible solution for the NLP method is the SNOPT with special initial guess. However, as described below, results from the IRM and RRT* methods outperform the best results from existing NLP solvers in terms of the flight time. The IRM converges to the optimal solution within 15 iterations and the overall computational time is 1065.77 seconds. Different from case one where the paths from refined RRT* lead to a semi-uniform result with significantly small differences, paths

generated for case two fall into several branches, as shown in Fig. 3.28(b). Among 350 simulation runs, only 5.43% simulations lead to a shortest path with $f_{refined} = 14.93$, which is consistent with the one obtained from IRM. However, most of the simulations, with the probability of 79.71%, generate flight paths yielding $f_{refined}$ around 17.96 with ± 0.001 difference. The remaining 14.85% simulations generate results with even larger $f_{refined}$. Although the random process is assumed to sample the distribution uniformly, it is observed that the RRT is biased to generate more samples in larger spaces [40]. Thus the sampling trees are not likely to distribute vertices in narrow spaces, leading to the low opportunity of producing the shortest outcome in case two. It takes an average of 10.88 seconds to find the coarse path from RRT* and the average computation time to obtain the final path considering refinement steps is 193.29 seconds. Compared with the simulation time of IRM, the refined RRT* requires much less computation time without guarantee of optimality.



(a) Planned paths generated from IRM (dash) and NLP (solid) for case two.

(b) Planned paths generated from RRT* (dash) and refined RRT* (solid) for case two.

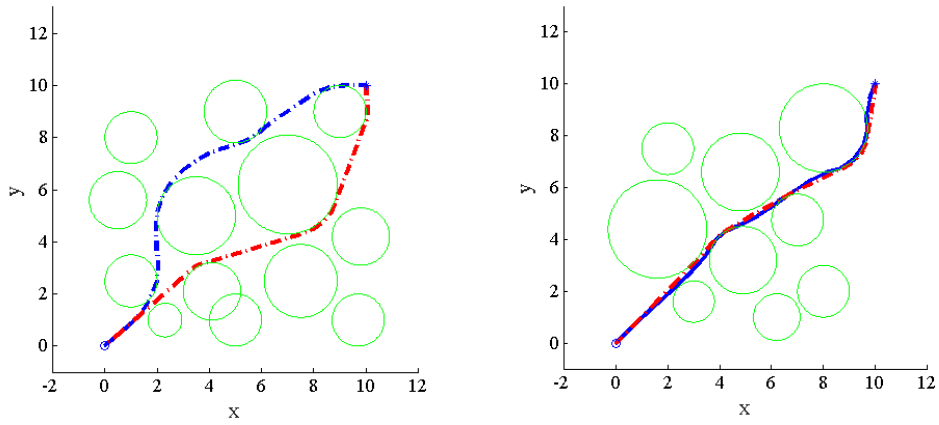
Figure 3.28: Comparative results of planned path for case two.

To further verify advantages of the proposed methods, results of four additional cases are demonstrated in Fig. 3.29, where overlap between avoidance zones are considered. For each case, the refined RRT* execute 15 simulation runs and the path yielding best performance is provided in the corresponding plot to compare with those obtained from IRM. Among the four cases, two of them generate very similar results and the remaining two lead to different paths where flying time resorting from the IRM is slighter shorter than the corresponding results from the best result of the refined RRT*. The summary of objective value and simulation time for cases two to six are listed in Table 3.

Table 3.7: Comparison of objective value and simulation time for cases two-four.

	f_{IRM}	$f_{refined}$ (15 runs)	time of IRM(s)	time of Refined RRT*(s)
Case 2	14.75	14.93	1065.77	193.29
Case 3	15.73	15.92	1481.27	605.98
Case 4	14.86	15.29	1879.19	184.39
Case 5	14.43	14.52	424.83	212.24
Case 6	14.52	15.42	1124.73	188.90

All example cases indicate that the numerical optimization approach using the IRM method achieves high performance with guaranteed convergence. However, at each iteration of IRM, the additional semidefinite constraint, $r_k I_{n-1} - V_{k-1} X_k V_{k-1} \succeq 0$ in (3.33), introduces $(n-1) \times (n-1)$ linear constraints, which results in significant increases of computational time compared to the semidefinite relaxation formulated in (3.26). The heuristic search based on refined RRT*, on the other hand, has faster convergence without guarantee of optimality. In real applications, according to the mission definition, one of the approaches can be selected to meet specific mission planning requirement.



(a) Case Three

(b) Case Four

(c) Case Five

(d) Case Six

Figure 3.29: Planned paths generated from IRM (solid) and best result of refined RRT* (solid) for cases three to six.

3.3.4 Identification Of Network Topology

A priori knowledge of network structure/topology is essential in many types of multi-agent involved missions, where interaction among agents is determined by the underlying network structure. For example, successful control of leader-follower network toward desired states requires identifying node indices of leaders, connectivity between leaders and followers, and interaction relationship among followers [39]. In fact, obtaining network

topology is a prerequisite for operations such as spacecraft formation, mobile robot rendezvous, unmanned aerial vehicle flocking [15]. In this paper, we address the problem of network topology identification (NTI) by tracking input and output data observed in discrete time sequence.

Extensive work has been developed in the area of identification for a linear time-invariant (LTI) system which is controllable and observable. The well-known Kung [114] and subspace methods [204] find 'similar' state space matrices that match the response between input and output. Such 'black box' model constructed by input-output data is applied in this paper for solving NTI problems. We assume the dynamics of each agent in a connected network is governed by the consensus protocol. Thus the linear transfer matrix of LTI is defined by the negative graph Laplacian that directly reflects the network topology. However, finding 'similar' state-space matrices is far away from the objective of identifying the exact network topology.

Previous work in [151] used the 'black-box' setup to establish the generating function of graph Laplacian by observing input-output data from selected network nodes. Work in [150], meanwhile, used an integrated spectral characterization of graphs and similarity transformation approach to find an approximate graph Laplacian. Reconstruction of tree-like networks and sparse networks can be found in recent work of [8, 138, 152, 172]. However, mapping the exact graph Laplacian from 'similar' state-space matrices constructed by input-output data is very challenging. The major reason comes from the involvement of unknown binary variables representing the edge set of the network. For a network with given number of nodes, the number of network topology configurations is an exponential function about its number of nodes. It will be extremely time consuming to find the exact graph Laplacian

that has the closest spectra to the 'similar' state transfer matrix while keeping the similarity properties.

In this paper, we reformulate the similarity transformation between graph Laplacian and 'similar' state space matrices as bilinear constraints. The NTI problem is then transformed as an optimization problem with bilinear constraints, as well as constraints on the Laplacian structure. We further generalize the NTI problem as a nonconvex quadratically constrained quadratic programming (QCQP) problem, classified as NP-hard.

3.3.4.1 LTI System Identification

We start with introducing the black-box approach of identifying a LTI system by observing input-output data. Consider a linear continuous system in state-space format

$$\begin{array}{c} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{array}, \tag{3.40}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^{r_\mathscr{I}}$ is the input vector, $y \in \mathbb{R}^{r_\mathscr{O}}$ is the output vector, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times r_\mathscr{I}}$, and $C \in \mathbb{R}^{r_\mathscr{O} \times n}$ are the state-space matrices. Through discrete integration, the continuous linear differential equation in (3.40) can be integrated at discrete sampling time t_k, $k = 1, \ldots, K$, and reformulated as discrete LTI system in the format

$$\begin{array}{c} x(t_{k+1}) = A_d x(t_k) + B_d u(t_k) \\ y(t_k) = C_d x(t_k) \end{array}, \tag{3.41}$$

where $A_d = e^{t_k A}$, $B_d = (\int_0^{t_k} e^{At} dt)B$, and $C_d = C$. For a given A_d, $A = 1/t_k \log_M A_d$, where $\log_M$ denotes the matrix logarithm. By propagation, the input-output expression for discrete LTI at sampling sequence p is expressed as

$$y(p) = H(p)u(p), \tag{3.42}$$

where $H(p) = \Gamma(p)\Omega(p)$ denotes the Hankel matrix and matrices $\Gamma(p)$ and $\Omega(p)$ are defined by the A_d, B_d, C_d in (3.41) as $\Gamma(p) = \begin{bmatrix} C_d \\ C_d A_d \\ \vdots \\ C_d A_d^{p-1} \end{bmatrix}$ and $\Omega(p) = \begin{bmatrix} B_d & A_d B_d & \dots & A_d^{p-1} B_d \end{bmatrix}$, respectively. By tracking a series of input and output data at discrete sampling time, the Hankel matrix can be constructed via impulse response parameters [114] or lease-square estimation method [204]. There are many methods to reconstruct the extended controllability and observability matrices, $\Gamma(p)$ and $\Omega(p)$, from the Hankel matrix. For example, a commonly used one is the singular value decomposition method [114]. Results of this identification procedure leads to the realization of 'similar' state-space matrices set (A_T, B_T, C_T) that represents the identified LTI system

$$\begin{aligned} \dot{x}(t) &= A_T x(t) + B_T u(t) \\ y(t) &= C_T x(t) \end{aligned} \tag{3.43}$$

while satisfying the input-output response. Furthermore, from Kalman's theorem [104], the set (A, B, C) and (A_T, B_T, C_T) are linearly related by a nonsingular matrix $T \in \mathbb{R}^{n \times n}$ such that

$$A_T = TAT^{-1}, \quad B_T = TB, \quad C = CT^{-1}. \tag{3.44}$$

This system identification procedure can be applied to the NTI problem where the state-space transfer matrix is solely determined by the network topology.

3.3.4.2 Consensus Protocol of Dynamic Networks

We consider a undirected network $\mathscr{G} = (V, E)$ with vertex set $V = \{1, 2, \dots, n\}$ and edge set E consisting of two element subsets of V. We use nodes or agents interchangeably with vertices. The connection among the nodes in the undirected network $\mathscr{G} = (V, E)$ is expressed by the entries of the adjacency matrix with $[\mathscr{A}(\mathscr{G})]_{ij} = 1$ when $v_i, v_j \in E$ and $[\mathscr{A}(\mathscr{G})]_{ij} = 0$

otherwise. Since the adjacency matrix for a graph on n nodes, $\mathscr{A}(\mathscr{G})$, is symmetric, we use a set of binary variables comprised of $n(n-1)/2$ elements to determine off-diagonal entries of the $\mathscr{A}(\mathscr{G})$. We note that the diagonals are simply zeros. Using such a framework, we assign binary variable a_{ij} to represent the element $[\mathscr{A}(\mathscr{G})]_{ij}$ in $\mathscr{A}(\mathscr{G})$ with $a_{ij} = a_{ji}$, $(i \neq j)$ and a_{ii} is set to be zero. The degree matrix $\Delta(\mathscr{G})$ of the graph can also be expressed in terms of the binary variables a_{ij} as $\Delta(\mathscr{G})_{ii} = \sum_{j=1}^{n} a_{ij}$ and $\Delta(\mathscr{G})_{ij} = 0\,(i \neq j)$. Therefore, the Laplacian $\mathscr{L}(\mathscr{G}) = \Delta(\mathscr{G}) - \mathscr{A}(\mathscr{G})$ is completely determined by these binary variables.

Assuming nodes belong to the set $\mathscr{I} \in \mathscr{V}$ with cardinality $r_{\mathscr{I}}$ are selected to receive infused control signals u. Meanwhile, nodes set $\mathscr{O} \in \mathscr{V}$ with cardinality $r_{\mathscr{O}}$ is used to observe output response y. Recall that when x_i denotes the state of dynamic agent i in the connected network $\mathscr{G}$, the consensus protocol of the overall system with input u is represented by

$$\dot{x}(t) = -\mathscr{L}(\mathscr{G})x(t) + Bu(t), \tag{3.45}$$

which will drive each agent to the consensus set $\mathscr{C} = \{x \in \mathbb{R}^n \mid x_i = x_j, \forall v_i, v_j \in V\}$ by exchanging state information with connected agents in the specified network $\mathscr{G}$. $B \in \mathbb{R}^{n \times r_{\mathscr{I}}}$ is the control matrix with element $B_{i,j}$ set as one if node i, $i = 1, \ldots, n$, is selected as the j'th input, where $j = 1, \ldots, r_{\mathscr{I}}$. For the output, we have

$$y(t) = Cx(t), \tag{3.46}$$

where $C \in \mathbb{R}^{r_{\mathscr{O}} \times n}$, is the observation matrix with element $C_{j,i}$ set as one if node i, $i = 1, \ldots, n$, is selected as the j'th output, where $j = 1, \ldots, r_{\mathscr{O}}$. An illustrative example is demonstrated in Fig. 3.30.

Here we assume the LTI system governed by (3.45) and (3.46) is controllable and observable. Furthermore, the network to be identified is connected. Under these assumptions, the system identification procedure introduced above is applied to construct the 'similar'

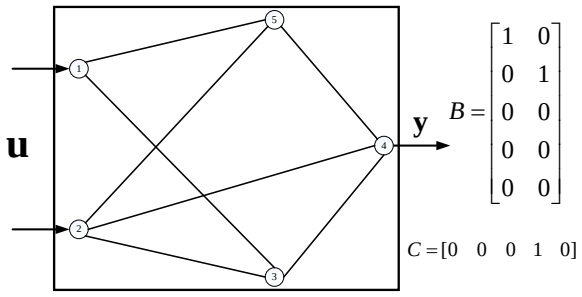


Figure 3.30: Illustrative example of NTI concept.

state-space matrices set (A_T, B_T, C_T). The detailed steps of constructing (A_T, B_T, C_T) can be referred to [150]. However, the system identification procedure does not lead to the exact A, B, C matrices of the original system, nor does the matrix A_T share the graph Laplacian properties. Our focus is to find the exact graph Laplacian from the obtained (A_T, B_T, C_T) set such that the final state-space matrices $(-\mathscr{L}(\mathscr{G}), B, C)$ satisfy the linear similarity relationship stated in (3.44) while constraining the graph Laplacian $\mathscr{L}(\mathscr{G})$ with the defined structure.

3.3.4.3 Problem Formulation

In summary, the NTI problem is to find $(-\mathscr{L}(\mathscr{G}), B, C)$ from constructed matrices set (A_T, B_T, C_T) obtained through system identification procedure. The desired matrix $-\mathscr{L}(\mathscr{G})$, sharing the same spectrum of A_T, has the minimum distance from A_T in the Frobenius norm [169]. In addition, the matrices set $(-\mathscr{L}(\mathscr{G}), B, C)$ is constrained by the similarity relationship and $\mathscr{L}(\mathscr{G})$ is constrained by Laplacian structure. The above statement of NTI can be summarized as a constrained optimization problem, written as

$$J = \min_{A,T} \|A - A_T\|_F^2$$

$$s.t. \qquad A_T = TAT^{-1}$$

$$
\begin{aligned}
B_T &= TB \\
C_T &= CT^{-1} \\
A_{i,j} &= A_{j,i} \in \{0,1\}, \forall v_i, v_j \in V, i \neq j \\
A\mathbf{1} &= \mathbf{0},
\end{aligned}
\tag{3.47}
$$

where $A = -\mathscr{L}(\mathscr{G})$. The above formulation including both binary variables in A and continuous variables in T (which is nonsingular as mentioned before) under nonlinear constraints is a mixed-integer nonlinear optimization problem. In addition, as A_T is supposed to be similar to a negative laplacian matrix of a graph, it must have the same eigenstructure, namely, all eigenvalues should be negative except the one which is zero. This can be used to check the feasibility of (3.47) or the system identification procedures. For a given matrix $\Phi \in \mathbb{R}^{n\times n}$, its Frobenius norm is determined by $\|\Phi\|_F := \sqrt{\mathbf{trace}(\Phi^T\Phi)} = \left(\sum_{i=1}^{n}\sum_{j=1}^{n}\Phi_{i,j}^2\right)^{\frac{1}{2}}$. Furthermore, the binary variables $A_{i,j}$ can be constrained via quadratic functions $A_{i,j}(A_{i,j}-1) = 0, \forall v_i, v_j \in V, i \neq j$. With these transformation, problem in (3.47) can be transformed into a quadratic optimization problem with quadratic and linear constraints in the form of

$$
\begin{aligned}
& J = \min_{A,T} \sum_{i=1}^{n}\sum_{j=1}^{n}(A_{i,j} - A_{T\,i,j})^2 \\
s.t. \quad & TA - A_T T = \mathbf{0} \\
& B_T = TB \\
& C_T T = C \\
& A_{i,j}(A_{i,j}-1) = 0, \forall v_i, v_j \in V, i \neq j \\
& A_{i,j} = A_{j,i} \\
& A\mathbf{1} = \mathbf{0}.
\end{aligned}
\tag{3.48}
$$

The first three constraints are from (3.44) and the following three are the properties of a graph Laplacian matrix. It's obvious that the quadratic constraints such as the binary variable constraints and bilinear constraints in (3.48) is nonconvex. Solving the NTI problem is then transformed as solving a nonconvex QCQP problem. An iterative approach is proposed in the following section to gradually approach the optimal solution.

3.3.4.4 Simulation Examples

In this section, we provide two simulation examples to identify the topology of two graphs, composed of 5 and 6 nodes, respectively. In the six-node case, from system identification procedure introduced in §3.3.4.1, the 'similar' state-space matrices set (A_T, B_T, C_T) is listed below, $A_T =$

$$\begin{bmatrix} -2.873 & -0.720 & 0.183 & -0.480 & 1.594 & 0.393 \\ -0.720 & -4.227 & -1.197 & -0.076 & -0.927 & -0.723 \\ 0.183 & -1.197 & -5.17 & 0.139 & 0.368 & 0.351 \\ -0.479 & -0.077 & 0.139 & -4.625 & 0.726 & 0.678 \\ 1.594 & -0.927 & 0.368 & 0.726 & -2.813 & 0.908 \\ 0.398 & -0.723 & 0.351 & 0.678 & 0.908 & -2.325 \end{bmatrix},$$

$B_T = \begin{bmatrix} 0.5152 & -0.0319 & -0.3207 \\ -0.0319 & 0.9979 & -0.0211 \\ -0.3207 & -0.0211 & 0.7878 \\ -0.2928 & -0.0193 & -0.1937 \\ -0.5515 & -0.0363 & -0.3649 \\ -0.4906 & -0.0323 & -0.3246 \end{bmatrix}$, and $C_T = B_T'$. We define the nodes with indices $i = 1, 2, 3$ as the selected input and output nodes. Under this assumption, matrices B and C are set as $B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$ and $C = B^T$, respectively.

By formulating the problem in the format described in (3.48) and applying the IRM method, the optimal transformation matrix T and the graph Laplacian are found as

$$T_{opt} =$$

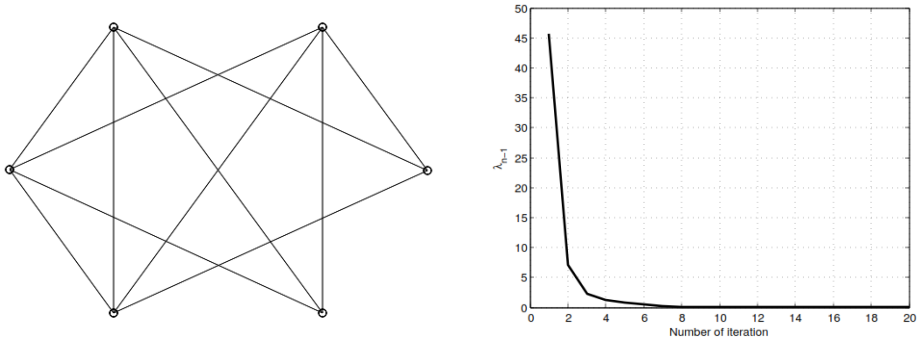
$$\begin{bmatrix} 0.515 & -0.031 & -0.320 & -0.292 & -0.551 & -0.490 \\ -0.031 & 0.997 & -0.021 & -0.019 & -0.036 & -0.032 \\ -0.320 & -0.021 & 0.787 & -0.193 & -0.364 & -0.324 \\ -0.292 & -0.019 & -0.193 & 0.823 & -0.333 & -0.296 \\ -0.551 & -0.036 & -0.364 & -0.333 & 0.372 & -0.558 \\ -0.490 & -0.032 & -0.324 & -0.296 & -0.558 & 0.503 \end{bmatrix},$$

$$L_{opt} = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 & 0 \\ -1 & 4 & 0 & -1 & -1 & -1 \\ -1 & 0 & 4 & -1 & -1 & -1 \\ 0 & -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & -1 & 4 & 0 \\ 0 & -1 & -1 & -1 & 0 & 3 \end{bmatrix}.$$

Comparing with the original network demonstrated in Fig. 3.31(a), it is verified that the network topology obtained from the proposed identification procedure is identical to the original one. In addition, the history of the second smallest eigenvalue at each iteration of the IRM algorithm is shown in Fig. 3.31(b). It indicates that λ_{n-1}, which is represented by r in (3.33), quickly reduces to zero within 18 steps, where the stopping threshold is set as $\delta = 10^{-4}$. This fact verifies that we obtain a rank one matrix of X within a few iterative steps.

Finding the network topology for five-node case is similar to the previous one. We define nodes with indices $i = 1,2$ as the input and output nodes. Following the same procedure, we can get the following results, $T_{opt} = \begin{bmatrix} 0.6262 & -0.0830 & -0.2022 & -0.4726 & 0.5752 \\ -0.6125 & 0.1835 & -0.6447 & -0.0626 & 0.4124 \\ 0.2034 & 0.1060 & 0.0804 & 0.8262 & 0.5064 \\ -0.3620 & -0.7848 & 0.3524 & -0.0587 & 0.3524 \\ -0.2452 & 0.5766 & 0.6404 & -0.2794 & 0.3378 \end{bmatrix}$ and $L_{opt} = \begin{bmatrix} 3 & -1 & 0 & -1 & -1 \\ -1 & 4 & -1 & -1 & -1 \\ 0 & -1 & 2 & -1 & 0 \\ -1 & -1 & -1 & 4 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix}$. Again, the graph Laplacian obtained above is consistent with the original graph provided in Fig. 3.32(a). Furthermore, the solution of the IRM method converges to the optimal solution within 16 steps, as demonstrated in Fig. 3.32(b).

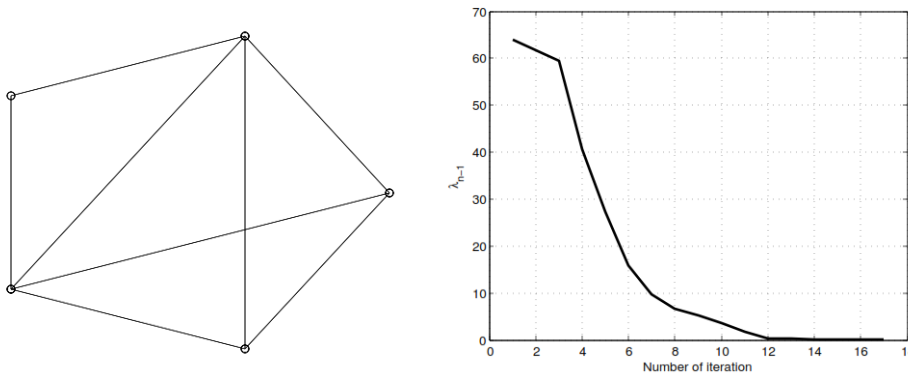
We have run more than one hundred simulation cases which uniformly yield converging results within a few iteration steps. To save space, the above two cases are provided with detailed results. The calculation time is dependant on number of the network nodes, the stopping threshold, and the performance of the SDP solver. For example, to identify a five-nodes network topology, we have 31 unknown variables in each iterative step. It takes the SDP solver around 10 seconds to calculate the solution at each step on an Lenovo T430 laptop with intel i7 CPU and 8GB RAM. However, to examine all tree graph with five nodes, it will take more than 10 minutes to finish the exhaustive search.



(a) Original graph of 6 Nodes

(b) λ_{n-1} at each iteration step

Figure 3.31: Network identification for six-node case.



(a) Original graph of 5 Nodes (b) λ_{n-1} at each iteration step

Figure 3.32: Network identification for five-node case.

3.4 Application of IRM for decomposition for sparse QCQP

To verify the feasibility and efficiency of the proposed decomposed formulation and associated IRM method, the mixed-boolean quadratic programming problems are considered here as simulation examples. Each iterative problem formulated in (3.33) is solved via the SDP solver, SeDuMi [181]. All of the simulation is run on a desktop computer with a 3.50 GHz processor and a 16 GB RAM.

The mixed-boolean quadratic programming problems are formulated as,

$$\begin{aligned} & J = \min_{\mathbf{x}} \mathbf{x}^T Q_0 \mathbf{x} \\ s.t. \quad & \mathbf{x}^T Q_i \mathbf{x} + c_i \leq 0, \ \forall\, i = 1, \ldots, m, \\ & \mathbf{x}_j \in \{1, -1\}, \ \forall\, j \in N_I, \end{aligned} \tag{4.49}$$

where $\mathbf{x} \in \mathbb{R}^n$, m is the number of inequality constraints, N_I is the index set of the integer variables. The matrices Q_0 and Q_i, $i = 1, \ldots, m$, are randomly generated and not necessarily positive semidefinite. Here the problem is sparse in the band pattern as defined before. Since the bivalent constraint on integer variables, $\mathbf{x}_j$, can be expressed as a quadratic equality

constraint in the form of $(\mathbf{x}_j + 1)(\mathbf{x}_j - 1) = 0$, $j \in N_I$, problem (4.49) can be converted to a nonconvex QCQP problem which can be solved by the proposed IRM method. The threshold for stopping criteria of IRM is set as $\varepsilon = 1e-6$.

We first set $n = 60$ and compare the results from IRM to those obtained from a commercial mixed-integer nonlinear programming solver, 'minlpBB', which applies branch and bound method to search for optimal solutions [94]. Moreover, we compare the results obtained from the centralized formulation in (3.26) and two types of complete decomposition formulation, i.e., $D_1 : |\alpha_p| = \frac{n}{2}$, $P = 6$ and $D_2 : |\alpha_p| = \frac{n}{3}$, $P = 15$. Note that we assume the cardinality of each α_p is the same for simplicity. 20 random cases are generated and solved via IRM, including three types of formulation, and a commercial solver, 'minlpBB'. For each case, the objective value obtained from both methods are recorded in Fig. 3.33. After comparison, the objective value obtained from IRM, including three types of formulation is always smaller than the corresponding one computed from 'minlpBB' for all of the 20 cases. The average computational time of each iteration (seconds)/the average iteration number for those methods are 'minlpBB': 5.17/1, IRM/centralized: 98.92/10.75, IRM/D_1: 41.16/10.05, and IRM/D_2: 27.59/10.95. In average, each iteration of D_2 decomposition method reduces 73.11% of the computational time compared to the centralized one. It verifies that the decomposed formulation significantly reduces the computational time compared to the centralized formulation. Furthermore, the value of $r_k^{(p)}$, representing the second largest eigenvalue of the unknown matrix $\tilde{X}^{(p)}$, at each iteration is demonstrated in Fig. 3.34 for one selected case. As r_k converges to a number close to zero within a few iterations, Fig. 3.34 verifies the convergence of $\tilde{X}^{(p)}$ in the IRM method to a rank one matrix. r_k for remaining cases also converge to zero and are not displayed here to save space.

Moreover, to demonstrate the reduction of computational time using the decomposition formulation for QCQPs at different scale, we solve (4.49) with n varying from 30 to 120. Table 4.1 demonstrates that the computational time of each iteration can be significantly reduced by the decomposition formulation compared to the centralized one. Especially for QCQPs with $n \geq 90$, it is impractical to solve the centralized SDP subproblem using SeDuMi due to the large scale.

However, for a complete decomposition α_p, $p = 1, \ldots, P$, the increment of P, does not always lead to further reduced computational time. Although size of each LMI is reduced, there is increment of $\sum_{p=1}^{P} |\alpha_p|$ which leads to increased number of LMIs. Therefore, for the above simulation cases, further reducing the size of decomposed submatrices will not significantly reduce the computational time.

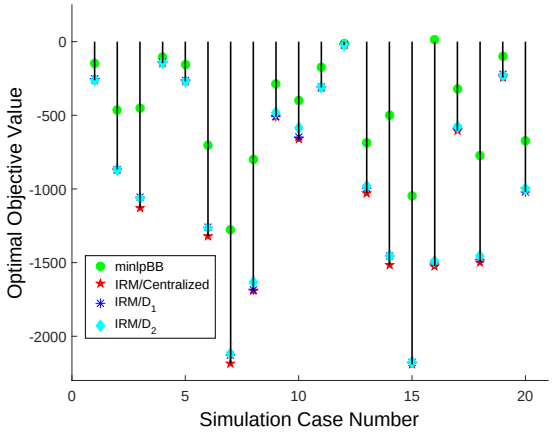


Figure 3.33: Comparative results between minlpBB and IRM for 20 cases.

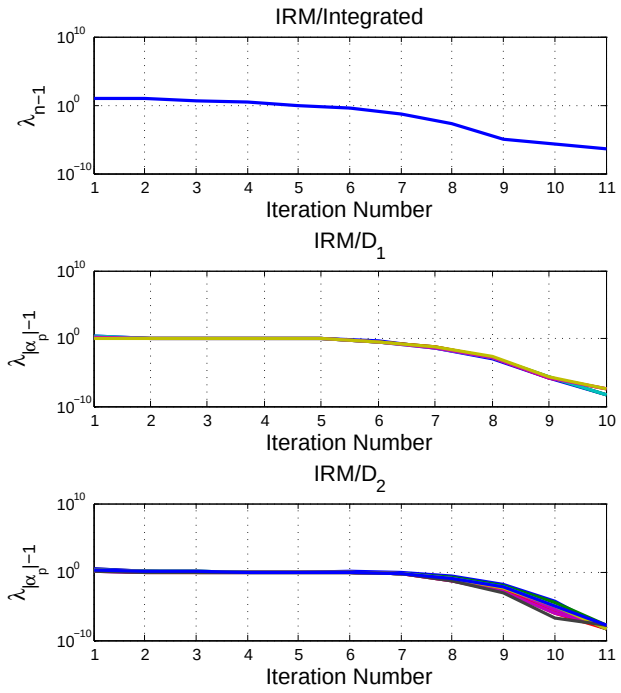


Figure 3.34: Value of elements in vector $\mathbf{r}_k$ at each iteration from the above three formulation.

Table 3.8: Performance comparison of three methods for solving problem (4.49) with different scales

n	Method	Objective	Time(sec) per Ite./Ite. No
n = 30	minlpBB	-148.41	1.97/1
	IRM/centralized	-154.94	2.19/8
	IRM/decomposed	-154.94	0.55/7
n = 60	minlpBB	-806.3	32.29/1
	IRM/centralized	-1126.6	105.5/10
	IRM/decomposed	-1126.6	3.21/9
n = 90	minlpBB	-1069.5	795.46/1
	IRM/centralized	N/A	N/A
	IRM/decomposed	-1601.1	3.63/8
n = 120	minlpBB	-1753.2	1582.2/1
	IRM/centralized	N/A	N/A
	IRM/decomposed	-2559.7	13.0/8
n = 200	minlpBB	N/A	N/A
	IRM/centralized	N/A	N/A
	IRM/decomposed	-4887.6	134.4/8

Chapter 4: ADMM for Rank Constrained Optimization

4.1 Customized ADMM for RCOP with Approximate Formulations

4.1.1 Introduction

Rank-Constrained Optimization Problems (RCOPs) are to minimize a convex objective function subject to a set of convex constraints and a matrix rank constraint. Since a matrix rank represents the dimension of space spanned by its vectors in any basis, it is of high interests in many fields when building a parsimonious system model. Application of RCOPs can be found in system identification, model reduction, and signal processing [191, 198]. In addition, a wide range of optimization problems can be equivalently reformulated as RCOPs, such as polynomial programming problems [193] which can be equivalently converted into a rank-one constrained optimization problem. However, the nonconvexity and nonlinearity of the rank constraint make it challenging to solve this type of nonconvex optimization problems.

One type of commonly used algorithm for solving RCOPs is the alternating projection-based method [81] which heavily depends on the initial guess for obtaining a convergent solution with relatively low computation cost. Alternatively, work in [161] proposed a Newton-like method to search for a feasible solution of RCOP with application to a feedback stabilization problem. For some special cases with one rank constraint, a greedy

convex smoothing algorithm has been designed [176]. However, these methods are generally time-consuming, especially for solving large scale RCOPs. When iterative approaches are employed, convergence with a fast rate is not a guarantee. After reviewing the aforementioned algorithms for RCOPs, we come to a conclusion that an efficient algorithm which is applicable to large-scale RCOPs with guaranteed global convergence is highly desired.

Recently, Alternating Direction Method of Multipliers (ADMM) has been widely used in various large-scale optimization-related fields, including but not limited to, machine learning, computer vision, and signal processing [173, 214]. A comprehensive survey of ADMM and its applications have been discussed in [27]. Due to the simple implementation and high computational performance in solving a great number of convex optimization problems, extensive research focusing on convergence analysis of ADMM has been investigated. For example, when applying ADMM in the separable two-block convex problems, it has proved convergence under certain mild assumptions [17, 96]. When extending ADMM to solve a convex problem with multiple separable blocks, work in [121] proves its linear global convergence under assumptions of strong convexity. However, the convergence of ADMM for solving nonconvex problems is still limited and open. Work in [97] analyzes the convergence of ADMM for solving consensus and sharing problem with a sufficiently large penalty parameter. However, the nonconvexity only lies in the objective function and the coupling constraint is still linear. Other work in [130] and [85] also investigates the convergence property of ADMM for some special cases.

ADMM has been applied to solving RCOPs. Work in [27] (Chapter 9.1) proposes a Singular Value Decomposition (SVD) based algorithm under the framework of ADMM. Due to the SVD computation involved in every iteration of the algorithm, it is time consuming. Furthermore, convergence of ADMM in solving this specific nonconvex optimization

problem has not been examined. As a result, this paper first introduces a set of auxiliary variables to decompose the convex constraints and the matrix rank constraints. Then, a penalty term associated with a sufficiently large weighting factor is added to the objective function to drive the auxiliary variables toward the original variables. With the auxiliary variables and the penalty term, we reformulate the original RCOP using an approximate representation. In addition, to avoid the SVD computation, we introduce a bilinear matrix equality constraint to replace the rank constraint based on the matrix factorization theory. The approximate formulation and the introduced bilinear constraint make each subproblem of ADMM more computationally-efficient. In particular cases, the subproblems can even be solved in a closed form. We then prove the global convergence of our proposed algorithm to a local minimizer of the approximate problem. To demonstrate the feasibility and efficiency of the proposed method, simulation results are presented with comparative results.

The rest of the paper is organized as follows. In Section II, the problem formulation of RCOP and the approximate representation are described. The customized ADMM algorithm and the closed form solution of some specific cases are given in Section III. In Section IV, we prove the global convergence of the proposed algorithm. Simulation results are demonstrated in Section V and we conclude the paper with a few remarks in Section VI.

4.1.1.1 Preliminaries

Some notations used throughout this paper are introduced in this section. The set of $n \times n$ symmetric matrices is denoted by $\mathbb{S}^n$ and the set of $n \times n$ positive semidefinite (definite) matrices is denoted by $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$). The notation $X \succeq \mathbf{0}$ ($X \succ \mathbf{0}$) means that the matrix $X \in \mathbb{S}^n$ is positive semidefinite (definite). The Frobenius norm of X is denoted by $\|X\|_F$ and the rank of X is denoted by $\mathbf{rank}(X)$. The operator $\mathrm{Proj}_{\mathscr{C}}(\bullet)$ represents the projection of "$\bullet$"

on the set $\mathscr{C}$. For two matrices X_1 and X_2 in appropriate dimensions, $\langle X_1, X_2 \rangle$ denotes the standard matrix inner product.

4.1.2 Problem Formulation

A rank-constrained optimization problem can be expressed in the form of

$$\begin{aligned} \min_{X} \quad & \langle C, X \rangle \\ s.t. \quad & X \in \mathscr{C} \\ & \mathbf{rank}(X) \leq r, \end{aligned} \tag{1.1}$$

where $X \in \mathbb{R}^{m \times n}$ is the unknown matrix to be determined, $\mathscr{C}$ is a convex feasible set, $C \in \mathbb{R}^{m \times n}$ and $r \in \mathbb{N}$ are constants. When the convex objective, e.g., $f_0(X)$ is not linear, we can introduce a new variable t and a new constraint $f_0(X) \leq t$ to replace $f_0(X)$ by t as the new objective. In this way, the new objective will be linear again and can be rewritten in the form of (1.1) without loss of generality. We start the reformulation of (1.1) with the following lemma.

Lemma 4.1.1. *For $X \in \mathbb{R}^{m \times n}$, $\mathbf{rank}(X) \leq r$ (we assume $r \leq \min(m,n)$ here and in the following) is equivalent to that there exist $F \in \mathbb{R}^{m \times r}$ and $G \in \mathbb{R}^{r \times n}$ such that $X = FG$. Moreover, for $X \in \mathbb{S}^n$, $\mathbf{rank}(X) \leq r$ with $X \succeq \mathbf{0}$ is equivalent to that there exist $F \in \mathbb{R}^{n \times r}$ such that $X = FF^T$.*

Providing Lemma 4.1.1 and introducing an auxiliary matrix $Y \in \mathbb{R}^{m \times n}$, problem in (1.1) is equivalent to

$$\begin{aligned} \min_{X,Y,F,G} \quad & \langle C, X \rangle \\ s.t. \quad & X \in \mathscr{C} \end{aligned} \tag{1.2}$$

$$Y = FG$$

$$X = Y,$$

where $F \in \mathbb{R}^{m\times r}$ and $G \in \mathbb{R}^{r\times n}$. The motivation of introducing Y will be stated in the algorithm and the convergence analysis sections. In the following, we define $\mathbf{1}_{\mathscr{X}}(X,Y)$ as an indicator function on the subset $\mathscr{X} = \{(X,Y)|X = Y, \text{where} X,Y \in \mathbb{R}^{m\times n}\}$, expressed as

$$\mathbf{1}_{\mathscr{X}}(X,Y) = \begin{cases} 0, (X,Y) \in \mathscr{X} \\ +\infty, (X,Y) \notin \mathscr{X} \end{cases}.$$

As a result, we approximate the indicator function as $\mathbf{1}_{\mathscr{X}}(X,Y) \approx \frac{\rho_2}{2}\|X-Y\|_F^2$ with ρ_2 selected as a positive and sufficiently large number. With the approximate indicator function, we can approximate (1.2) in the following form

$$\begin{aligned} \min_{X,Y,F,G} \quad & \langle C,X\rangle + \tfrac{\rho_2}{2}\|X-Y\|_F^2 \\ s.t. \quad & X \in \mathscr{C} \\ & Y = FG. \end{aligned} \tag{1.3}$$

In (1.3), with a sufficiently large penalty coefficient, it is equivalent to the original problem (1.1). Moreover, the only nonconvex part is the coupling bilinear constraint and we will develop a customized ADMM to solve it in the following section.

4.1.3 A Customized ADMM

In this section, we will outline the general algorithm as well as its implementation in some special cases.

4.1.3.1 ADMM Framework

To solve the approximate problem via ADMM, we first build the augmented Lagrangian function of (1.3), expressed as

$$\begin{aligned} \mathscr{L}(X,Y,F,G;\Lambda) &= \langle C,X\rangle + \frac{\rho_2}{2}\|X-Y\|_F^2 \\ &+ \langle \Lambda, Y-FG\rangle + \frac{\rho_1}{2}\|Y-FG\|_F^2, \end{aligned} \tag{1.4}$$

where $\Lambda \in \mathbb{R}^{m\times n}$ is the dual variable and $\rho_1 > 0$ is a constant. Then by employing the classical ADMM framework, we partition the variables into two sets, (X,F) and (Y,G). To solve (1.3), the algorithm alternates between those two variable sets followed by the update of the dual variable Λ. The subproblems of ADMM at step $k+1$ for (1.3) include

$$(X,F)^{k+1} := \arg\min_{X\in\mathscr{C},F} \mathscr{L}(X,Y^k,F,G^k;\Lambda^k), \tag{1.5a}$$

$$(Y,G)^{k+1} := \arg\min_{Y,G} \mathscr{L}(X^{k+1},Y,F^{k+1},G;\Lambda^k), \tag{1.5b}$$

$$\Lambda^{k+1} := \Lambda^k + \rho_1(Y^{k+1} - F^{k+1}G^{k+1}), \tag{1.5c}$$

where ρ_1 is the step size.

The above steps demonstrate the advantages of introducing the auxiliary variable Y. Firstly in (3.16a), searching for the optimal X and F is separable since terms in the objective function and constraints related to the two matrices are not coupled. This makes the subproblem much easier to solve, especially when the constraint set $X \in \mathscr{C}$ is involved. Searching for the optimal F is to solve an unconstrained convex quadratic programming problem so that we can find the optimal F by simply applying the first order optimality condition which is expressed as a linear matrix equality. Secondly in (3.16b), although the pair (Y,G) is coupled, the subproblem in (3.16b) is still a convex quadratic optimization problem and thus can be solved by finding the joint first order condition which is also

a linear matrix equality. The detailed procedures for solving the subproblem are shown in the following subsections. In one word, the introduction of Y contributes to a more computationally efficient approach when solving the subproblems of ADMM. Thirdly, this approach also plays an important role that leads to the global convergence, which will be stated in the convergence analysis section.

The above proposed algorithm also demonstrates advantages over the traditional Alternating Minimization (AM) method. When AM is applied to problem (1.1), it also introduces the bilinear term $X = FG$ to substitute X in both the objective and constraints and then alternatively searches for F and G by fixing one of them. However, each iteration of AM method changes the original constraints in problem (1.1), which may lead to an infeasible subproblem. For example, when searching for F while fixing G, the constraint $FG^k \in \mathscr{C}$ is not equivalent to $X \in \mathscr{C}$ and may be infeasible. However, formulation in (3.16) avoids this issue by remaining $X \in \mathscr{C}$ in the original format.

The customized ADMM algorithm is summarized as follows in Algorithm 1.

Algorithm 1: Customized ADMM for problem (1.3)
Input: Problem parameters $C, \mathscr{C}, \rho_1, \rho_2$ and initial point (Y^0, G^0, Λ^0)
Output: Local minimizer (X, Y, F, G) to (1.3)
begin

1. Set $k = 0$
2. **while** not convergent
3. Update of X and F

$$\begin{aligned}&(X,F)^{k+1} := \\ &\arg\min_{X\in\mathscr{C},F} \quad \langle C, X\rangle + \tfrac{\rho_2}{2}\|X - Y^k\|_F^2 \\ &\quad + \langle \Lambda^k, Y^k - FG^k\rangle + \tfrac{\rho_1}{2}\|Y^k - FG^k\|_F^2\end{aligned} \tag{1.6}$$

4. Update Y and G

$$\begin{aligned}&(Y,G)^{k+1} := \\ &\quad \arg\min_{Y,G} \quad \tfrac{\rho_2}{2}\|X^{k+1} - Y\|_F^2 \\ &+ \langle \Lambda^k, Y - F^{k+1}G\rangle + \tfrac{\rho_1}{2}\|Y - F^{k+1}G\|_F^2\end{aligned} \tag{1.7}$$

5. Update Λ

$$\Lambda^{k+1} := \Lambda^k + \rho_1(Y^{k+1} - F^{k+1}G^{k+1}) \tag{1.8}$$

6. $k = k + 1$
7. **end while**
8. Find X, Y, F, G

end

4.1.3.2 Subproblem Solution for Cases with Equality Affine Constraints

For the two subproblems in Algorithm 1, (3.16) is unconstrained and convex with regard to the variables Y and G while (1.6) is to optimize over a general convex feasible set $\mathscr{C}$. As stated above, searching for the pair (X, F) in (1.6) is separable. For a random convex

feasible region $\mathscr{C}$,

$$X^{k+1} = \text{Proj}_{\mathscr{C}}(Y^k - \frac{C}{\rho_2}). \tag{1.9}$$

However, (1.9) does not generally have a closed form solution except for some special cases. Here we will show how to solve it analytically when $\mathscr{C}$ is defined as equality affine constraints. In other words, $\mathscr{C} := \{X \in \mathbb{R}^{m\times n} | \mathscr{A}(X) = b\}$, where the operator $\mathscr{A}(\bullet) : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is defined as

$$\mathscr{A}(X) = \big[\langle A_1, X\rangle, \langle A_2, X\rangle, \ldots, \langle A_p, X\rangle\big]^T, \tag{1.10}$$

with $A_i \in \mathbb{R}^{m\times n}, \forall\, i = 1,2,\ldots,p$. Moreover, we denote $\text{vec}(X) \in \mathbb{R}^{mn}$ as vectorization of X by stacking the columns of X on top of one another. Based on the operation of vectorization, $\mathscr{A}(X)$ can be rewritten as

$$\mathscr{A}(X) = A\ \text{vec}(X),$$

where $A = \big[\text{vec}(A_1), \text{vec}(A_2), \ldots, \text{vec}(A_p)\big]^T \in \mathbb{R}^{p\times mn}$. Moreover, without loss of generality we can assume $p \leq mn$ and A is a matrix of full row rank provided that we can simply keep the most linearly independent rows without changing $\mathscr{C}$. In addition, we define the adjoint operator $\mathscr{A}^*(\mu) : \mathbb{R}^p \to \mathbb{S}^n$ as $\mathscr{A}^*(\mu) = \sum_{i=1}^p A_i\mu_i$ and the operator $\mathscr{A}\mathscr{A}^*(\bullet) : \mathbb{R}^p \to \mathbb{R}^p$ as $\mathscr{A}\mathscr{A}^*(\mu) = (AA^T)\mu$. These operators will be used later in the context.

Recall (1.6) and for simplicity we ignore the superscript for the ADMM iteration index. Then the optimality conditions of this quadratic programming with equality affine constraints can be written as

$$\begin{aligned} C + \rho_2(X - Y) - \mathscr{A}^*(\mu) &= \mathbf{0} \\ -\Lambda G^T + \rho_1(FG - Y)G^T &= \mathbf{0} \\ \mathscr{A}(X) &= b, \end{aligned} \tag{1.11}$$

where $\mu \in \mathbb{R}^p$ is the dual variable for (1.6). Solving (1.11) will obtain the updated (X, F) as follows

$$\begin{aligned} \mu &= (\mathscr{A}\mathscr{A}^*)^{-1}\left[\rho_2 b + \mathscr{A}(C - \rho_2 Y)\right] \\ X &= -\frac{1}{\rho_2}(C - \mathscr{A}^*(\mu) - \rho_2 Y) \\ F &= (\Lambda G^T + \rho_1 Y G^T)(\rho_1 G G^T)^{-1}. \end{aligned} \tag{1.12}$$

Similarly, we can solve the unconstrained (3.16) by examining its optimality conditions, expressed as

$$\begin{aligned} G &= (\rho_1\rho_2 F^T F)^{-1}\Big(\rho_2 F^T(\Lambda + \rho_1 X)\Big) \\ Y &= -\tfrac{1}{\rho_1+\rho_2}(\Lambda - \rho_2 X - \rho_1 F G). \end{aligned} \tag{1.13}$$

For a rank constraint with $r > 1$, computing the matrix inverse in (1.12) and (1.13) is straightforward. When $r = 1$, problem (1.1) represents a broad range of nonconvex optimization problems, such as quadratically constrained quadratic programming [184, 190], and the matrix inverse degenerates to the inverse of a scalar. Note that in (1.12), the solution of μ is well defined due to that AA^T is invertible. In addition, as A is a given parameter set, computing $(AA^T)^{-1}$ is only required once and then cached before the ADMM iteration begins. However, it might occur that G (or F) is not of full row (or collum) rank such that the update of F (or G) in (1.12) (or (1.13)) will be ill-defined. In that case, we will approximate GG^T (or $F^T F$) by adding εI where $\varepsilon > 0$ is a small scalar and I is an identity matrix of dimension r.

4.1.3.3 Algorithm for Cases with $X \succeq \mathbf{0}$

For cases with $\mathscr{C} \subseteq \{X \in \mathbb{S}^n | X \succeq \mathbf{0}\}$ considered in (1.1), though we can solve it in the aforementioned algorithm, the fact that problem (1.6) is a quadratic programming with linear matrix inequality will greatly decrease the efficiency of Algorithm 1. By applying Lemma

(4.1.1), the semidefinite constraint on X with $\mathbf{rank}(X) \leq r$ can be equivalently transformed into $X = FF^T$, where $F \in \mathbb{R}^{n\times r}$. Based on the equivalent transformation, subproblem in (1.2) can be revised in the following form by adding one more equality constraint $F = G^T$ to replace the semidefinite constraint in (1.6). Similarly, by introducing an approximate indicator function for the constraint $F = G^T$, the approximate problem can be reformulated as

$$\begin{aligned} \min_{X,Y,F,G} \quad & \langle C, X\rangle + \tfrac{\rho_2}{2}\|X - Y\|_F^2 + \tfrac{\rho_3}{2}\|F - G^T\|_F^2 \\ s.t. \quad & X \in \mathscr{C} \\ & Y = FG. \end{aligned} \tag{1.14}$$

Then similar to the customized ADMM in Algorithm 1, we could use the same framework to solve the newly formulated problem (1.14). Moreover, in this case both (1.6) and (3.16) are strongly convex with regard to F and G, thus the corresponding update of F and G, unlike (1.12) and (1.13), are guaranteed to be well defined.

4.1.4 Convergence Analysis

In this section, inspired by the concept in [85,97], we aim to prove the global convergence of Algorithm 1 under mild conditions. Although we focus on problem (1.3) in the following discussion, the proof is also applicable to problem (1.14). At first, the following lemma is introduced.

Lemma 4.1.2. *For two adjacent iterations of (3.16), we have* $\|\Lambda^{k+1} - \Lambda^k\|_F^2 \leq 2\rho_2^2(\|X^{k+1} - X^k\|_F^2 + \|Y^{k+1} - Y^k\|_F^2)$.

Proof. The first order optimality for Y in (3.16) is that

$$\rho_2(Y^{k+1} - X^{k+1}) + \Lambda^k + \rho_1(Y^{k+1} - F^{k+1}G^{k+1}). \tag{1.15}$$

Combining (4.28) and (3.10), we get

$$-\Lambda^{k+1} = \rho_2(Y^{k+1} - X^{k+1}), \tag{1.16}$$

which leads to

$$\|\Lambda^{k+1} - \Lambda^k\|_F^2 \leq 2\rho_2^2(\|X^{k+1} - X^k\|_F^2 + \|Y^{k+1} - Y^k\|_F^2).$$

□

Remark: From Lemma 6.4.3, another advantage of introducing the auxiliary variable Y and the penalty $\frac{\rho_2}{2}\|X - Y\|_F^2$ is demonstrated. It makes X carry the (local) constraint $X \in \mathscr{C}$ while Y appears in the coupling (rank) constraint $Y = FG$. As a result, the $Y-$optimization is unconstrained and thus we can apply the first order optimality condition in (4.28). Then combining the update of the dual variables, it builds the equality relationship between the prime and dual variables. This conclusion is critical to prove the main result below.

Another preparatory lemma is introduce below.

Lemma 4.1.3. *For the following convex optimization problem with a quadratic objective,*

$$\begin{aligned} \min_x \quad & f(x) = \tfrac{1}{2}x^T Ax + b^T x \\ s.t. \quad & x \in \mathscr{C}, \end{aligned} \tag{1.17}$$

we have that $f(x^*) - f(x) \leq -\frac{1}{2}(x^* - x)^T \nabla^2 f(x)(x^* - x)$ *where* x^* *is the minimizer of (1.17).*

Proof. Staring with Taylor expansion, we have

$$\begin{aligned} f(x^*) \quad - \quad & f(x) \\ = \quad & \langle f(x), x^* - x\rangle + \frac{1}{2}(x^* - x)^T \nabla^2 f(x)(x^* - x) \\ = \quad & \langle f(x^*), x^* - x\rangle - \frac{1}{2}(x^* - x)^T \nabla^2 f(x)(x^* - x) \end{aligned}$$

$$\leq \quad -\frac{1}{2}(x^*-x)^T\nabla^2 f(x)(x^*-x), \tag{1.18}$$

where the first equality follows that the second order Taylor expansion for a quadratic function is exact, the second equality can be verified provided $\nabla f(x) = Ax+b, \nabla^2 f(x) = A$, and the third inequality follows the optimality condition of (1.17) [28]. □

Assumption 4.1.4. *The relaxation problem of (1.1) by dropping the rank constraint, i.e.,* $\min_{X\in\mathscr{C}}\langle C,X\rangle$*, is bounded.*

To simplify the notation, we denote $P^k = (X,Y,F,G)^k$ as a collection of the primal variables in the following context.

Lemma 4.1.5. *The augmented Lagrangian function is lower bounded and monotonically non-increasing in the iterations of Algorithm 1 with* $\rho_1 > 4\rho_2$*. Mathematically, it is stated that there exists* $c_1 > 0, c_2 > 0, c_3 > 0, c_4 > 0$ *such that* $\mathscr{L}(P^{k+1};\Lambda^{k+1}) - \mathscr{L}(P^k;\Lambda^k) \leq -c_1\|X^{k+1}-X^k\|_F^2 - c_2\|(F^{k+1}-F^k)G^k\|_F^2 - c_3\|Y^{k+1}-Y^k\|_F^2 - c_4\|F^{k+1}(G^{k+1}-G^k)\|_F^2$.

Proof. From (1.6) and Lemma 4.1.3, we have

$$\begin{aligned}&\mathscr{L}(X^{k+1},Y^k,F^{k+1},G^k;\Lambda^k) - \mathscr{L}(P^k;\Lambda^k) \leq \\ &-\frac{\rho_2}{2}\|X^{k+1}-X^k\|_F^2 - \frac{\rho_1}{2}\|(F^{k+1}-F^k)G^k\|_F^2.\end{aligned} \tag{1.19}$$

Similarly, from (3.16), it leads to

$$\begin{aligned}&\mathscr{L}(P^{k+1};\Lambda^k) - \mathscr{L}(X^{k+1},Y^k,F^{k+1},G^k;\Lambda^k) \leq \\ &-\frac{\rho_1+\rho_2}{2}\|Y^{k+1}-Y^k\|_F^2 - \frac{\rho_1}{2}\|F^{k+1}(G^{k+1}-G^k)\|_F^2.\end{aligned} \tag{1.20}$$

Moreover, we have

$$\mathscr{L}(P^{k+1};\Lambda^{k+1}) - \mathscr{L}(P^{k+1};\Lambda^k)$$

$$
\begin{aligned}
&= \langle \Lambda^{k+1} - \Lambda^k, Y^{k+1} - F^{k+1}G^{k+1} \rangle \\
&= \frac{1}{\rho_1} \|\Lambda^{k+1} - \Lambda^k\|_F^2 \qquad (1.21) \\
&\leq \frac{2\rho_2^2}{\rho_1} (\|X^{k+1} - X^k\|_F^2 + \|Y^{k+1} - Y^k\|_F^2),
\end{aligned}
$$

where the first equality follows the definition of $\mathscr{L}$, the second follows (3.10), and the inequality follows Lemma 6.4.3. By reorganizing the above inequalities, we have

$$
\begin{aligned}
& \mathscr{L}(P^{k+1}; \Lambda^{k+1}) - \mathscr{L}(P^k; \Lambda^k) \\
= \; & \mathscr{L}(X^{k+1}, Y^k, F^{k+1}, G^k; \Lambda^k) - \mathscr{L}(P^k; \Lambda^k) \\
+ \; & \mathscr{L}(P^{k+1}; \Lambda^k) - \mathscr{L}(X^{k+1}, Y^k, F^{k+1}, G^k; \Lambda^k) \\
+ \; & \mathscr{L}(P^{k+1}; \Lambda^{k+1}) - \mathscr{L}(P^{k+1}; \Lambda^k). \qquad (1.22)
\end{aligned}
$$

Based on (4.30)-(4.32), (4.33) can be rewritten as

$$
\begin{aligned}
& \mathscr{L}(P^{k+1}; \Lambda^{k+1}) - \mathscr{L}(P^k; \Lambda^k) \\
\leq \; & -\left(\frac{\rho_2}{2} - \frac{2\rho_2^2}{\rho_1} \right) \|X^{k+1} - X^k\|_F^2 \\
& -\frac{\rho_1}{2} \|(F^{k+1} - F^k)G^k\|_F^2 \\
& -\left(\frac{\rho_1 + \rho_2}{2} - \frac{2\rho_2^2}{\rho_1} \right) \|Y^{k+1} - Y^k\|_F^2 \\
& -\frac{\rho_1}{2} \|F^{k+1}(G^{k+1} - G^k)\|_F^2. \qquad (1.23)
\end{aligned}
$$

With $\rho_1 > 4\rho_2 > 0$, we have $c_1 = \frac{\rho_2}{2} - \frac{2\rho_2^2}{\rho_1} > 0$, $c_2 = \frac{\rho_1}{2} > 0$, $c_3 = \frac{\rho_1+\rho_2}{2} - \frac{2\rho_2^2}{\rho_1} > 0$ and $c_4 = \frac{\rho_1}{2} > 0$.

Next, we will verify that $\mathscr{L}(P^k; \Lambda^k)$ is lower bounded.

$$
\begin{aligned}
& \mathscr{L}(P^k; \Lambda^k) \\
= \; & \langle C, X^k \rangle + \frac{\rho_2}{2} \|X^k - Y^k\|_F^2
\end{aligned}
$$

$$
\begin{aligned}
& +\langle \Lambda^k, Y^k - F^k G^k \rangle + \frac{\rho_1}{2}\|Y^k - F^k G^k\|_F^2 \\
\overset{(4.28)}{=} \quad & \langle C, X^k \rangle + \frac{\rho_2}{2}\|X^k - Y^k\|_F^2 \\
& +\rho_2 \langle X^k - Y^k, Y^k - F^k G^k \rangle + \frac{\rho_1}{2}\|Y^k - F^k G^k\|_F^2 \\
= \quad & \langle C, X^k \rangle + \frac{\rho_1 - \rho_2}{2}\|Y^k - F^k G^k\|_F^2 \\
& +\frac{\rho_2}{2}\|X^k - F^k G^k\|_F^2 \\
> \quad & -\infty,
\end{aligned} \tag{1.24}
$$

where the last inequality comes from Assumption 4.1.4. This concludes the proof. □

Theorem 4.1.6. *Algorithm 1 will globally converge to a Karush-–Kuhn-–Tucker (KKT) point of (1.3) with* $\rho_1 > 4\rho_2$.

Proof. From (4.34) and (1.24), it has been verified that the sequence $\{\mathscr{L}(P^k; \Lambda^k)\}$ is non-increasing and lower bounded. Thus the sequence is convergent. As a result, from (4.34) we have

$$
\begin{array}{c}
X^{k+1} - X^k \to 0, (F^{k+1} - F^k) G^k \to 0 \\
Y^{k+1} - Y^k \to 0, F^{k+1}(G^{k+1} - G^k) \to 0.
\end{array} \tag{1.25}
$$

As $\{X^k\}$ and $\{Y^k\}$ converge, (4.28) leads to that $\{\Lambda^k\}$ converges. That indicates the prime feasibility is achieved. In other word, $Y^{k+1} - F^{k+1}G^{k+1} = 0$ indicates the rank constraint is satisfied, though $\{F^k\}$ and $\{G^k\}$ are not necessary convergent. Once the prime feasibility is satisfied, it can be shown that Algorithm 1 reaches a KKT point of (1.3) by simply checking its optimality conditions. □

4.1.5 Simulation Results

This section applies the proposed algorithm to two representative cases. One is the affine rank-constrained optimization problem and the other is the well-known image denoising

problem. Comparative results are presented to verify improved computational efficiency of the proposed algorithm. All of the simulation is run on a standard desktop computer with a 3.5 GHz processor and a 16 GB memory.

4.1.5.1 Affine rank constrained problem

We first start with the the affine rank-constrained optimization problem, expressed as

$$\begin{aligned} \min_{X} \quad & \langle C, X \rangle \\ s.t. \quad & \mathscr{A}(X) = b \\ & X \succeq \mathbf{0} \\ & \mathbf{rank}(X) \leq r. \end{aligned} \tag{1.26}$$

We solve the above problem of different scales via Algorithm 1 (denoted by ADMM-Alg.1) by the variant formulation (1.14). For comparison, we apply the ADMM based on SVD in [27] (denoted as ADMM-SVD here) with some modifications considering that problem in (1.26) has other constraints besides the rank constraint. Both algorithms use the same parameter settings, especially for the initial values, which are randomly generated. Moreover, we also solve a relaxed version of (1.26) by dropping the semidefinite constraint with the formulation in (1.3). The stopping criterion assumes $\frac{\|X_k - X_{k-1}\|_F}{\|X_k\|_F} \leq 1 \times 10^{-3}$ and $\frac{\|X_k - Y_k\|_F}{\|X_k\|_F} \leq 1.5 \times 10^{-2}$. We add $\frac{\|F_k - G_k^T\|_F}{\|F_k\|_F} \leq 1 \times 10^{-2}$ only for the SDP case.

Comparative results of two cases of different scales are provided in Table 4.1, which demonstrate the significant improvement of computational efficiency of the proposed algorithm. The column σ_{r+1}/σ_r shows the accuracy of the approximation. When SDP is considered in the first case, even though the scale of the problem is $n = 50$, ADMM-SVD requires much more computational time due to the SDP subproblem and SVD operation involved in each iteration. The efficiency of Algorithm 1 is further verified in Figure 4.1 for

Table 4.1: Performance comparison of ADMM-Alg.1 and ADMM-SVD for problem (1.26) with $n = 50$ and SDP in the first case and $n = 500$ w/o SDP in the second case

Problem	Method	Obj.	Time(s)	Ite. #	$\frac{\sigma_{r+1}}{\sigma_r}$
(1.26)	ADMM-Alg.1	-27.71	0.0636	697	0.0143
w/ SDP	ADMM-SVD	-26.14	67.27	22	0
(1.26)	ADMM-Alg.1	-147.84	12.1541	1544	0.0021
w/o SDP	ADMM-SVD	-91.39	135.14	1871	0.00055

larger scale problems. In addition, Figure 6.6 shows the monotonic non-increasing trend of the augmented Lagrangian function of one case. Note that in the figure, the value of the augmented Lagrangian function is normalized as $(\mathscr{L}_k - \mathscr{L}_{\min})/|\mathscr{L}_{\min}|$, where $\mathscr{L}_{\min}$ is the function value in the last iteration when the algorithm terminates. In addition, the relative prime feasibility of $\frac{\|Y_k - F_k G_k\|_F}{\|Y_k\|_F}$, the approximation error $\frac{\|X_k - Y_k\|_F}{\|X_k\|_F}$, and $\frac{\|F_k - G_k^T\|_F}{\|F_k\|_F}$ (only for SDP case) is shown in Figure 6.5.

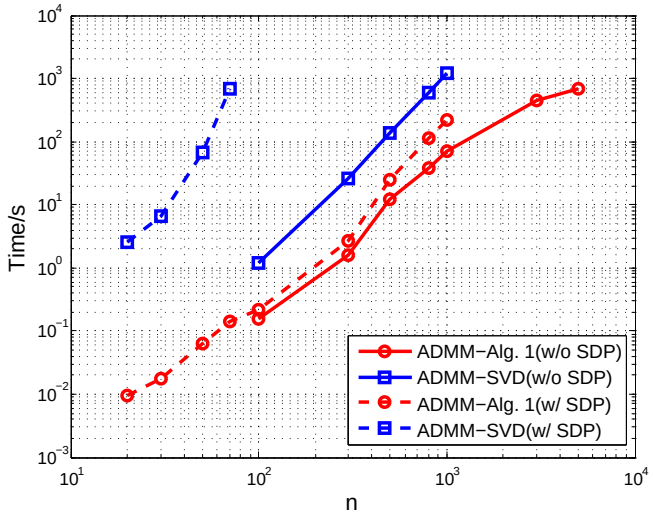


Figure 4.1: Comparison of computation time for solving (1.26) using ADMM-Alg.1 and ADMM-SVD for cases with and without SDP.

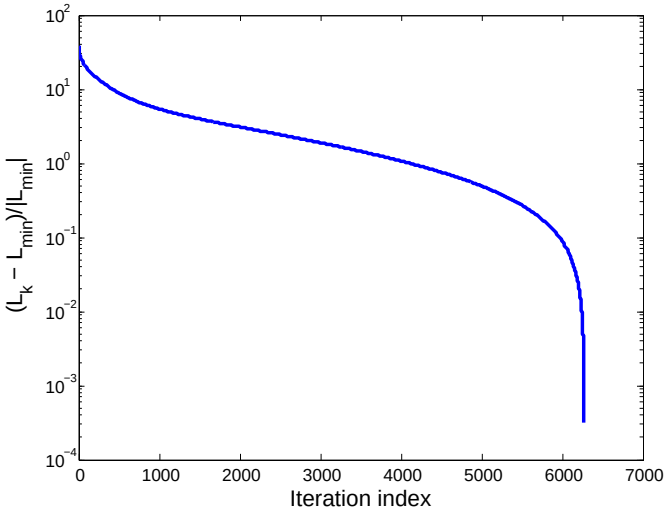


Figure 4.2: Convergence history of the augmented Lagrangian function value.

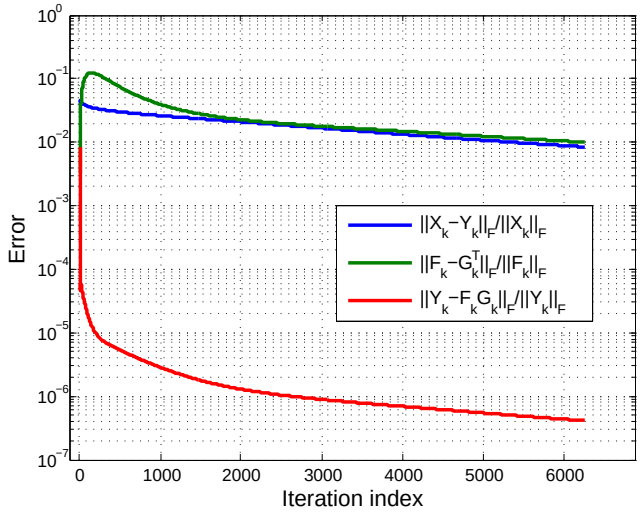


Figure 4.3: Convergence of the relative prime feasibility of the coupling constraint and approximation error.

4.1.5.2 Image Denoising

Image denoising problem aims to recover a corrupted image with partial pixels missing and the given ones are interfered by noises. For an 8 bit grayscale image, each pixel is

represented by an integer in the interval $[0,255]$ to carry intensity information. In addition, as the matrix representing image information is usually of low rank, the image denoising problem can be converted into a low rank matrix completion problem. Mathematically, it can be formulated as

$$\begin{aligned} \min_{X\in\mathbb{R}^{m\times n}} \quad & \tfrac{1}{2}\|P_\Omega(X-X_0)\|_F^2 \\ s.t. \quad & \mathbf{rank}(X)\leq r, \end{aligned} \tag{1.27}$$

where X_0 is the corrupted information matrix and Ω is the index set of the available noised observation. Moreover, $P_\Omega(X)(i,j)=X_{i,j}$ if $(i,j)\in\Omega$ and $P_\Omega(X)(i,j)=0$ otherwise. In (1.27), there are two factors defining how the image is corrupted. One is how many entries of X_0 are missing and the other is how to model the noise. As a result, we define a coefficient $\alpha=|\Omega|/(mn)$ to represent the ratio of the observed entries of X. In addition, we assume the noise is in Gaussian distribution, i.e., $N\sim\mathscr{N}(0,\sigma^2 I)$, where σ is the standard deviation.

Algorithm 1 is applied to two simulation examples and the results are compared to those obtained from the algorithm in [101]. In Figure 4.4, the simulation results validate that our proposed algorithm outperforms the comparative one in terms of producing a high quality denoised image. As a result, it verifies the advantages of the proposed algorithm in image denoising problems.

4.1.5.3 Conclusions

This paper proposes a new algorithm to approximately solve the Rank-Constrained Optimization Problems (RCOPs) via a customized Alternating Direction Method of Multipliers (ADMM). The proposed algorithm does not require singular value decomposition operations when solving RCOPs, which makes it different from existing ADMM-based algorithms

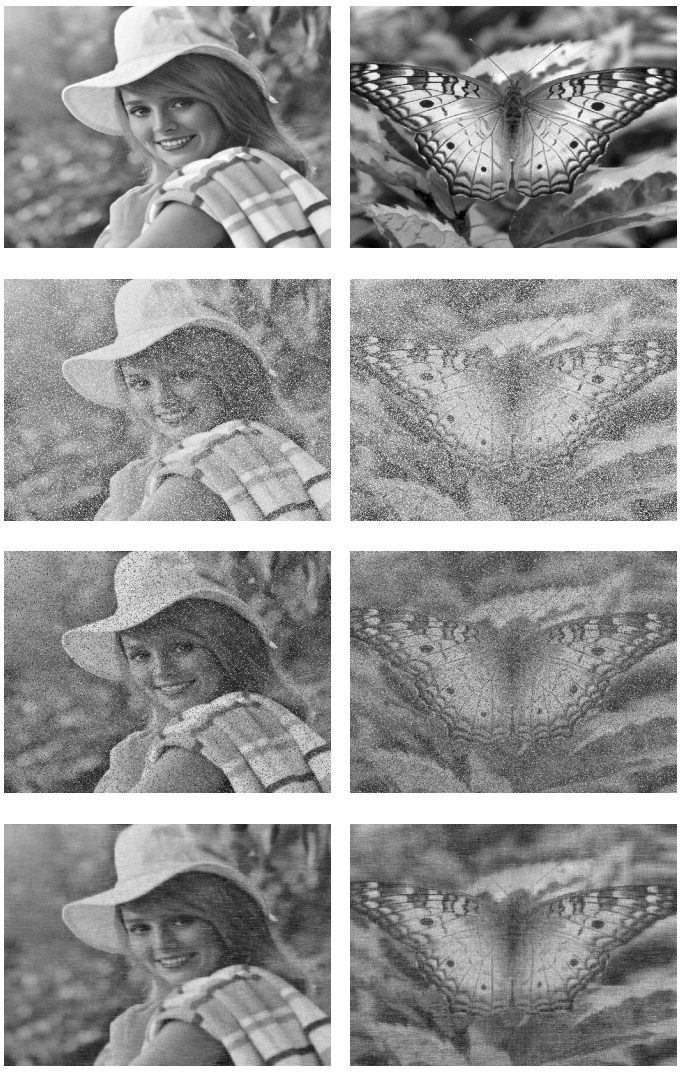


Figure 4.4: Each column represents one simulation example. Left: $\alpha = 0.8, \sigma = 10$; Right: $\alpha = 0.7, \sigma = 20$. Each column from top to bottom: original image, corrupted image, denoised image via algorithm in [101], denoised image via Algorithm 1.

and significantly improves its computational efficiency. More importantly, another contribution is the global convergence proof of the customized ADMM to a stationary point of the approximate problem of RCOP. Numerical simulation results validate the improved

efficiency of the proposed algorithm. Future research will investigate the convergence rate of the customized ADMM algorithm.

4.2 Customized ADMM for RCOP with exact Formulations

4.2.1 Introduction

A number of important problems arising in graph applications can be formulated as a linear binary optimization problem, for different choices of C:

$$\text{minimize} \qquad x^T C x \tag{2.28}$$

$$\text{subject to} \quad x \in \{+1, -1\}^n \tag{2.29}$$

Here, C is an $n \times n$ symmetric matrix, where n is the number of nodes in an undirected graph. For different choices of C, problem (2.28) may correspond to the MAX-CUT problem in combinatorics, the 2-community detection problem in machine learning, or many other combinatorial graph problems.

Exactly solving combinatorial problems of this nature is a very difficult problem (it is in general NP-Hard), and even the best branch-and-bound methods do not scale well beyond a few thousands of nodes.

4.2.1.1 Notation

Define $\mathbb{S}^n$ the space of $n \times n$ symmetric matrices. For an undirected graph with n nodes, we define $A \in \mathbb{S}^n$ the adjacency matrix of the graph, where $A_{ij} > 0$ is the strictly positive weight of the edge between nodes i and j, and $A_{ij} = 0$ if there is no edge between i and j. Define $\mathbf{1}$ as the vector of all 1's.

4.2.1.2 Community detection

The problem of community detection is to identify node clusters in undirected graphs that are more likely to be connected with each other than with nodes outside the cluster. This prediction is useful in many graphical settings, such as interpreting online communities through social network or linking behavior, interpreting molecular or neuronal structure in biology or chemistry, finding disease sources in epidemiology, and many more. There are many varieties and methodologies in this field, and it would be impossible to list them all, though a very comprehensive overview is given in [69].

The stochastic binary model [92] is one of the simplest generative models for this application. Given a graph with n nodes and parameters $0 < q < p < 1$, the model partitions the nodes into two communities, and generates an edge between nodes in a community with probability p and nodes in two different communities with probability q. Following the analysis in [1], we can define $C = \frac{p+q}{2}\mathbf{1}\mathbf{1}^T - A$, and the solution to (2.28) gives a solution to the community detection problem with sharp recovery guarantees.

4.2.1.3 MAX-CUT

While the community detection problem aims to find two densely connected clusters in a graph, the MAX-CUT problem attempts to find two clusters with dense interconnections. For an undirected graph with n nodes and (possibly weighted) edges between some pairs of nodes, a *cut* is defined as a partition of the n nodes, and the *cut value* is the sum of the weights of the severed edges. The MAX-CUT problem is to find the cut in a given graph that gives the maximum cut value. When $C = (A - \mathbf{diag}(A\mathbf{1}))/4$, the solution to (2.28) is exactly the maximum cut.

Many methods exist to find the MAX-CUT of a graph. In general, combinatorial methods can be solved using branch-and-bound schemes, using a linear relaxation of (2.28) as a bound [12, 50], where the binary constraint is replaced with $0 \leq (x+1)/2 \leq 1$. Historically, these "polyhedral methods" were the main approach to find exact solutions of the MAX-CUT problem. Though this is an NP-Hard problem, if the graph is sparse enough, branch-and-bound converges quickly even for very large graphs [50]. However, when the graph is not very sparse, the linear relaxation is loose, and finding efficient branching mechanisms is challenging, causing the algorithm to run slowly.

The MAX-CUT problem can also be approximated by one pass of the linear relaxation (with bound $\frac{f_{\text{relax}}}{f_{\text{exact}}} \geq 2 \times \#\text{edges}$) [165]. A tighter approximation can be found with the semidefinite relaxation (see next section), which is also used for better bounding in branch-and-bound techniques [11, 34, 90, 170]. In particular, the rounding algorithm of [77] returns a feasible $\hat{x}$ given optimal Z, and is shown in expectation to satisfy $\frac{x^T C x}{\hat{x}^T C \hat{x}} \geq 0.878$. For this reason, the semidefinite relaxation for problems of type (2.28) are heavily studied (*e.g.* [72, 89, 164]).

4.2.1.4 Semidefinite relaxation

To find the semidefinite relaxation of problem (2.28), we first reformulate it equivalently using a change of variables $Z = xx^T$:

$$\begin{array}{ll} \underset{Z \in \mathbb{S}^n}{\text{minimize}} & \mathbf{trace}(CZ) \\ \text{subject to} & \mathbf{diag}(Z) = 1 \\ & Z \succeq \mathbf{0} \\ & \mathbf{rank}(Z) \leq 1. \end{array} \tag{2.30}$$

We refer to this formulation as the rank constrained semidefinite program (RCSDP). Without the rank constraint, (5.39) is the usual semidefinite (convex) relaxation used in MAX-CUT

approximations; we refer to the semidefinite relaxation as the SDR. Solving the SDR exactly can be done in polynomial time (*e.g.*using an interior point method); however, the per-iteration complexity can be large ($O(n^6)$ for interior point method and $O(n^3)$ for most first-order methods) limiting its scalability.

Consequently, a reformulation based on matrix factorization has been proposed by [32, 33], which solves the SDR with a factorization $Z = RR^T$, $R \in \mathbb{R}^{n\times r}$:

$$\begin{array}{ll} \underset{R\in\mathbb{R}^{n\times r}}{\text{minimize}} & \mathbf{trace}(R^T C R) \\ \text{subject to} & \mathbf{diag}(RR^T) = \mathbf{1} \end{array} \tag{2.31}$$

which we will call the factorized SDP (FSDP). Note that FSDP is not convex; however, because the search space is compact, if r is large enough such that $r(r+1)/2 > n$, then the global optimum of the FSDP (2.31) is the global optimum of the SDR [14, 163]. Furthermore, a recent work [26] shows that almost all local optima of FSDP are also global optima, suggesting that any stationary point of the FSDP is also a reasonable approximation of (2.28).

To solve (2.31), [32] put forward an algorithm based on augmented Lagrangian method to find local optima of (2.31); unsurprisingly, due to the later obsrvation of Boumel et. al, this method empirically found global optima almost all of the time. However, solving (2.31) is still numerically burdensome; in the augmented Lagrangian term, the objective is quartic in R, and is usually solved using an iterative numerical method, such as L-BFGS.

Another way of finding a low-rank solution to the SDR is to use the nuclear norm as a surrogate for the rank penalty [36,169,200]. This method is popular because of its theoretical guarantees and experimental success. A difficulty with nuclear norm minimization is in choosing the penalty parameter; in general this is done through cross-validation. Also, when

the rank function appears as a hard constraint in the optimization problem, the corresponding bound for the surrogate model is ambiguous.

4.2.2 ADMM for nonconvex optimization

We propose using the Alternagting Direction Method of Multipliers (ADMM) to solve vector and SDP-based matrix reformulations of (2.28), as a scalable alternative to the prior methods mentioned here. Recently, the Alternating Direction Method of Multipliers (ADMM) [56, 73, 76, 123] has been widely applied and investigated in various fields; see the survey [27] and the references therein. Specifically, it is favored because it is easy to apply to a distributed framework, and has been shown to converge quickly in practice. For convex optimization problems, ADMM can be shown to converge to a global optima in several ways: as a series of contractions of monotone operators [56] or as the minimization of a global potential function [27].

However, in general there is a lack of theoretical justification for ADMM on nonconvex problems despite its good numerical performance. Almost all works concerning ADMM on nonconvex problems investigate when nonconvexity is in the objective functions (*i.e.* [97, 119, 130, 205], and also [127, 213] for matrix factorization) Under a variety of assumptions (*e.g.* convergence or boundedness of dual objectives) they are shown to convergence to a KKT stationary point.

In comparison, relatively fewer works deal with nonconvex constraints. [102] tackles polynomial optimization problems by minimizing a general objective over a spherical constraint $\|x\|_2 = 1$, [99] solves general QCQPs, and [177] solves the low-rank-plus-sparse matrix separation problem. In all cases, they show that all limit points are also KKT stationary points, but do not show that their algorithms will actually converge to the limit

points. In this work, we investigate a class of nonconvex constrained problems, and show with much milder assumptions that the sequence always converges to a KKT stationary point.

4.2.3 Vector Form

We begin by considering a simple reformulation of (2.28) using only vector variables

$$\begin{array}{ll} \underset{x,y\in\mathbb{R}^n}{\text{minimize}} & x^T C x \\ \text{subject to} & y \in \{-1,1\}^n \\ & x = y. \end{array}$$

This is similar to the nonconvex formulations proposed in [27], Chapter 9.1.

To solve (2.32) via ADMM, we first form the augmented Lagrangian

$$\mathscr{L}(x,y;\mu) = x^T C x + \langle \mu, x-y\rangle + \frac{\rho}{2}\|x-y\|_F^2, \tag{2.32}$$

where $\mu \in \mathbb{R}^n$ is the corresponding dual variable and $\rho > 0$ is the weighting factor associated with the penalty term. Under the ADMM framework, problem in (2.32) can be solved by alternating between the two primal variables x and y, and then updating the dual variable μ. The ADMM for (2.32) is summarized in Algorithm 2. Note that unlike typical ADMM, the penalty parameter ρ in Algorithm 2 is increasing with the iteration index, which will facilitate the convergence.

4.2.3.1 Solution for Subproblems in ADMM

For every step k in ADMM of Algorithm 3, the updating procedure is composed of three subproblems, expressed in (3.8)-(3.10).The update for y in (3.8) is just rounding

$$y_i^{k+1} = \mathbf{sign}(x^k + \mu^k/\rho^k)_i,$$

Algorithm 2 ADMM for solving (2.32)

Inputs: $C \in \mathbb{S}^n$, $\rho_0 > 0$, $\alpha > 1$, tol $\varepsilon > 0$
Outputs: Local optimum (x, y) of (2.32)
Initialize: $x^0, y^0; \mu^0$ as random vectors
for $k = 1 \ldots$ **do**
    Update y^k as the solution to

$$\begin{array}{ll} \min\limits_{y} & \|x^k - y + \frac{\mu^k}{\rho^k}\|_F^2 \\ \text{s.t.} & y \in \{-1, 1\}^n \end{array} \tag{2.33}$$

    Update x^k the minimizer of

$$x^T C x + \frac{\rho^k}{2}\|x - y^{k+1} + \frac{\mu^k}{\rho^k}\|_F^2 \tag{2.34}$$

    Update μ via

$$\begin{array}{rcl} \mu^{k+1} & = & \mu^k + \rho^k(x^{k+1} - y^{k+1}) \\ \rho^{k+1} & = & \alpha\rho^k \end{array} \tag{2.35}$$

    if $\|x^k - y^k\| \leq \varepsilon$ **then**
        break
    end if
end for

and is computationally cheap. However, updating x is more cumbersome. The update of x is the solution to a linear system

$$2Cx + \mu^k + \rho^k(x - y^{k+1}) = 0. \tag{2.36}$$

and $x^k = (\rho^k I + 2C)^{-1}(-\mu^k + \rho^k y^{k+1})$. If $\rho^k = \rho$ is constant, then one can avoid inverting at each iteration by storing $(\rho^k I + 2C)^{-1}$ and using matrix multiplication at each step. For changing ρ^k, one can store the eigenvalue decomposition of $C = U_C \Lambda_C U_C^T$, and compute for each ρ^k

$$(\rho^k I + 2C)^{-1} = U_C(2\Lambda_C + \rho^k I)^{-1} U_C^T.$$

Although this formulation seems simple, when n is large, the initial eigenvalue decomposition or matrix inverse can incur significant overhead. Still, despite these limitations, this vector-based method can be shown to converge under very mild conditions.

4.2.3.2 Convergence Analysis

We will now show that Algorithm 2 converges under very mild conditions; specifically, the augmented Lagrangian monotonically decreases.

Definition 4.2.1. *$x^T Cx$ is L_1-smooth function, that is, any x_1, x_2, $\|2Cx_1 - 2Cx_2\| \leq L_1\|x_1 - x_2\|$.*

Definition 4.2.2. *$L_H \in \mathbb{R}_+$ is the smallest number such that $-L_H I \preceq 2C$, where $2C$ is the Hessian of the objective function of* (2.28)*. In other words, the Hessian is lower bounded.*

Assumption 4.2.3. *The value $x^T Cx$ is lower bounded over $x \in \{-1, 1\}^n$.*

Theorem 4.2.4. *Given the sequence $\{\rho^k\}$ such that*

$$(\rho^k)^2 - L_H\rho^k - (\alpha + 1)L_1^2 > 0, \rho^k > L_H, \alpha > 0$$

under Algorithm 2 the augmented Lagrangian monotonically decreases

$$\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1})-\mathscr{L}(y^k,x^k;\mu^k)\leq -c_1^k\|x^{k+1}-x^k\|_F^2$$

with $c_1^k=\frac{\rho^k-L_H}{2}-\frac{(\alpha+1)L_1^2}{2\rho^k}>0$. *With Assumption 6.4.1 and* $\rho^k>L_1$, *the augmented Lagrangian* $\mathscr{L}(y^k,x^k;\mu^k)$ *is lower bounded and convergent. Therefore,* $\{y^k,x^k,\mu^k\}$ *converge to* $\{y^*,x^*,\mu^*\}$ *a KKT solution of* (2.32) *and* $x^*=y^*$.

Proof. We first prove the following lemma.

Lemma 4.2.5. *If Assumption 6.4.1 holds, for two adjacent iterations of Algorithm 2 we have*

$$\|\mu^{k+1}-\mu^k\|_2^2\leq L_1^2\|x^{k+1}-x^k\|_2^2. \tag{2.37}$$

Proof. Recall the first order optimality conditions for x in (2.36)

$$2Cx^{k+1}+\mu^k+\rho^k(x^{k+1}-y^{k+1})=0. \tag{2.38}$$

Combining with (3.10), we get

$$2Cx^{k+1}+\mu^{k+1}=0. \tag{2.39}$$

As (4.28) holds for each iteration k, combining it with the definition of L_1 proves the lemma. □

Remark: We update x after y so that we can apply the optimality condition (4.28).

Next we will show that the augmented Lagrangian (2.32) is monotonically decreasing and lower bounded.

Lemma 4.2.6. *The augmented Lagrangian* (2.32) *is monotonically decreasing; that is,*

$$\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1};\rho^{k+1})-\mathscr{L}(y^k,x^k;\mu^k;\rho^k)$$
$$\leq c_1^k\|x^{k+1}-x^k\|_F^2. \quad (2.40)$$

where $c_1^k=\frac{\rho^k-L_H}{2}-\frac{(\alpha+1)L_1^2}{2\rho^k}>0$.

Proof. For the update of y in (3.8), we have

$$\mathscr{L}(y^{k+1},x^k;\mu^k;\rho^k)-\mathscr{L}(y^k,x^k;\mu^k;\rho^k)\leq 0, \quad (2.41)$$

based on the fact that problem (3.8) is globally minimized. Using $\{y,\mu,\rho\}=\{y^{k+1},\mu^k,\rho^k\}$, for the update of (3.9), we have

$$\begin{aligned}\mathscr{L}(y,x^{k+1};\mu;\rho)-\mathscr{L}(y,x^k;\mu;\rho)&\\ \leq\langle\nabla_x\mathscr{L}(y,x^{k+1};\mu;\rho),x^{k+1}-x^k\rangle&\\ -\frac{\rho^k-L_H}{2}\|x^{k+1}-x^k\|_2^2&\\ \leq-\frac{\rho^k-L_H}{2}\|x^{k+1}-x^k\|_2^2,& \quad (2.42)\end{aligned}$$

where the first inequality follows that $\mathscr{L}(y,x;\mu;\rho)$ is strongly convex with respect to x given $\rho>L_H$, and the second inequality follows from the optimality condition of (3.9).

In addition, using $\{x,y\}=\{x^{k+1},y^{k+1}\}$ we have

$$\begin{aligned}&\mathscr{L}(y,x;\mu^{k+1};\rho^{k+1})-\mathscr{L}(y,x;\mu^k;\rho^k)\\ =\;&\langle\mu^{k+1}-\mu^k,x-y\rangle+\frac{\rho^{k+1}-\rho^k}{2}\|x-y\|_F^2\\ =\;&\frac{\alpha+1}{2\rho^k}\|\mu^{k+1}-\mu^k\|_2^2\end{aligned} \quad (2.43)$$

where the first equality follows the definition of $\mathscr{L}$ and the second follows (3.10). Combining with Lemma (6.4.3)

$$\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1};\rho^{k+1})-\mathscr{L}(y^{k+1},x^{k+1};\mu^k;\rho^k)$$

$$\leq \frac{(\alpha+1)L_1^2}{2\rho^k}\|x^{k+1}-x^k\|_F^2, \tag{2.44}$$

Moreover, it can be easily verified that

$$\begin{aligned}
&\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1};\rho^{k+1})-\mathscr{L}(y^k,x^k;\mu^k;\rho^k)\\
&\qquad=\mathscr{L}(y^{k+1},x^k;\mu^k;\rho^k)-\mathscr{L}(y^k,x^k;\mu^k;\rho^k)\\
&\qquad+\mathscr{L}(y^{k+1},x^{k+1};\mu^k;\rho^k)-\mathscr{L}(y^{k+1},x^k;\mu^k;\rho^k)\\
&\qquad+\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1};\rho^{k+1})-\mathscr{L}(y^{k+1},x^{k+1};\mu^k;\rho^k).
\end{aligned} \tag{2.45}$$

By incorporating (4.30), (4.31) and (4.32), (4.33) can be rewritten as

$$\begin{aligned}
&\mathscr{L}(y^{k+1},x^{k+1};\mu^{k+1};\rho^{k+1})-\mathscr{L}(y^k,x^k;\mu^k;\rho^k)\\
&\qquad\leq-\frac{\rho^k-L_H}{2}\|x^{k+1}-x^k\|_F^2+\frac{(\alpha+1)L_1^2}{2\rho^k}\|x^{k+1}-x^k\|_F^2\\
&\qquad\leq-\left(\frac{\rho^k-L_H}{2}-\frac{(\alpha+1)L_1^2}{2\rho^k}\right)\|x^{k+1}-x^k\|_F^2.
\end{aligned}$$

With $(\rho^k)^2-L_H\rho^k-(\alpha+1)L_1^2>0,\rho^k>0$, we have $c_1=\frac{\rho^k-L_H}{2}-\frac{(\alpha+1)L_1^2}{2\rho^k}>0$.

□

Lemma 4.2.7. *If the objective of* (2.32) *is lower-bounded over* $x\in\{-1,1\}^n$ *(assumption* (6.4.1)*), then the augmented Lagrangian* (2.32) *is lower bounded.*

Proof. For notation ease, we denote $f(x)=x^TCx$. From the L_1-Lipschitz continuity of $\nabla_x f(x)$, it follows that

$$f(y)\leq f(x)+\langle\nabla f(x),y-x\rangle+\frac{L_1}{2}\|x-y\|_F^2 \tag{2.46}$$

for any x and y. By definition

$$\mathscr{L}(y^k,x^k;\mu^k;\rho^k)=f(x^k)+\langle\mu^k,x^k-y^k\rangle+\frac{\rho^k}{2}\|x^k-y^k\|_F^2$$

$$= f(x^k) - \langle \nabla f(x^k), x^k - y^k \rangle + \frac{\rho^k}{2}\|x^k - y^k\|_F^2 \tag{2.47}$$

$$\geq f(y^k) + \frac{\rho^k - L_1}{2}\|x^k - y^k\|_F^2, \tag{2.48}$$

where (2.47) follows from (4.28) and (4.35) follows from (2.46). From the boundedness of the objective (Assumption 6.4.1) and the assumption $\rho^k > L_1$, it follows that $\mathscr{L}(y^k, x^k; \mu^k; \rho^k)$ is bounded below for all k. □

Since the sequence $\{\mathscr{L}(z^k, x^k; \mu^k)\}$ is monotonically decreasing and lower bounded, then the sequence $\{\mathscr{L}(z^k, x^k; \mu^k)\}$ converges. Combining with Lemma 6.4.3 we have

$$\|x^{k+1} - x^k\| \to 0, \tag{2.49}$$

which indicates that the sequence $\{x^k\}$ converges to a limit point (x^*). Moreover, as $\{x^k\}$ converges, the sequence $\{\mu^k\}$ also converges to μ^* based on (4.29). From (3.10), $\mu^{k+1} - \mu^k = \rho^k(x^{k+1} - y^{k+1})$ and so $x^{k+1} - y^{k+1} \to 0$; thus $\{y^k\}$ converges to a limit point y^* and $x^* = y^*$. Additionally, note that $x^* = y^*$ implies $\nabla_x \mathscr{L}(y^*, x^*; \mu^*; \rho) = \nabla_y \mathscr{L}(y^*, x^*; \mu^*; \rho) = 0$ and $x^* = y^* \in \{-1, 1\}$ are feasible. Thus Theorem 6.4.4 is proven.

□

Remark: Convergence is also guaranteed under a constant penalty coefficient $\rho_k \equiv \rho^0$ $(\alpha = 1)$ satisfying $(\rho^0)^2 - L_H\rho^0 - 2L_1^2 > 0$. However, in implementation, we find empirically that increasing $\{\rho^k\}$ from a relatively small ρ^0 can encourage convergence to more useful global minima.

Corollary 4.2.8. *With the prerequisites in Theorem 6.4.4 satisfied and* $(y^*, x^*; \mu^*)$ *as the accumulation point, we have that*

i. $\mu_i^* x_i^* \geq -\rho^*$ *with* $x^* \in \{-1,1\}^n$.

ii. x^* *is a stationary point of the optimization problem* $\min_x x^T Cx + (\mu^*)^T x$.

In particular for a convex function objective $f(x) = x^T Cx$ *(*$C \succeq 0$*),* $\mu^* = 0$ *indicates that the constraint* $x \in \{-1,1\}^n$ *is automatically satisfied; that is, the minimizer of* $f(x)$ *is the exact minimizer of* (2.28).

We now prove Corollary 4.2.8.

Proof. i) At the accumulation point, the update of x^* is expressed as

$$x^* = y^* = \text{Proj}_{\{-1,1\}^n}(x^* + \tfrac{u_i^*}{\rho^*}). \tag{2.50}$$

Since $x_i^* = \textbf{sign}(\rho^* x_i^* + u_i^*)$, therefore

$$x_i^*(\rho^* x_i^* + u_i^*) = \rho^* + x_i^* u_i^* \geq 0.$$

ii) Given $x^* = y^*$, then from (2.36) it follows that

$$\nabla_x f(x^*) + \mu^* = 0, \tag{2.51}$$

or equivalently,

$$x^* = \arg\min_x f(x) + x^T \mu^*.$$

If, additionally, $\mu^* = 0$, then x^* is also the minimizer of the unconstrained problem; that is

$$\arg\min_x f(x) + x^T \mu^* = \arg \min_{x \in \{-1,1\}} f(x) = \arg\min_x f(x)$$

suggesting that the combinatorial constraints $x \in \{-1,1\}$ are not necessary, and the convex relaxation is exact; therefore x^* is the global minimum of (2.28).

□

4.2.4 PSD Matrix Form

We now present our second reformulation of (2.28), using ideas from the SDR and FSDP:

$$\begin{array}{ll} \underset{X,Y,Z}{\text{minimize}} & \mathbf{trace}(CZ) \\ \text{subject to} & \mathbf{diag}(Z) = \mathbf{1} \\ & Z = XY^T \\ & X = Y. \end{array} \tag{2.52}$$

Here the matrix variables are $Z \in \mathbb{S}^n$, and X, $Y \in \mathbb{R}^{n\times r}$. The extra variable Y and the constraint $X = Y$ are introduced to transform the quadratic constraint in (2.31) to bilinear, which will simplify the subproblems significantly. Two cases of r are of interest. When $r = 1$, (2.52) is equivalent to (2.28), with $X = x$. When, $r = \lceil \sqrt{2n} \rceil$, then $\frac{r(r+1)}{2} > n$ and the global optimum of (2.52) is with high probability that of the SDR, based on the results of [26].

To solve (2.52) via ADMM, we build the augmented Lagrangian function as

$$\begin{aligned} \mathscr{L}(Z,X,Y;\Lambda_1,\Lambda_2) \;=\; & \mathbf{trace}(CZ) + \langle \Lambda_2, X - Y\rangle \\ & + \langle \Lambda_1, Z - XY^T\rangle + \frac{\rho}{2}\|X - Y\|_F^2 \\ & + \frac{\rho}{2}\|Z - XY^T\|_F^2 \end{aligned} \tag{2.53}$$

where $\Lambda_1 \in \mathbb{R}^{n\times n}$ and $\Lambda_2 \in \mathbb{R}^{n\times r}$ are the dual variables corresponding to the two coupling constraints, respectively. Similar as to the vector case, problem (2.52) is solved by alternating minimization between two primal variable sets, Y and (Z,X), and then updating the dual variable Λ_1 and Λ_2. This sequence is summarized in Algorithm 3 below.

Algorithm 3 ADMM for solving (2.52)

Inputs: $C \in \mathbb{S}^n$, $\rho_0 > 0$, $\alpha > 1$, tol $\varepsilon > 0$
Initialize: $Z^0, X^0; \Lambda_1^0, \Lambda_2^0$ as random matrices
Outputs: Z, $X = Y$
for $k = 1 \ldots$ **do**
    Update Y^{k+1} the minimizer of

$$\|Z^k - X^k Y^T + \frac{\Lambda_1^k}{\rho^k}\|_F^2 + \|X^k - Y + \frac{\Lambda_2^k}{\rho^k}\|_F^2 \tag{2.54}$$

    Update $(Z,X)^{k+1}$ as the solutions of

$$\begin{array}{ll} \min\limits_{X,Z} & \mathscr{L}(Z,X,Y^{k+1};\Lambda_1^k,\Lambda_2^k;\rho^k) \\ \text{s.t.} & \mathbf{diag}(Z) = \mathbf{1} \end{array} \tag{2.55}$$

    where $\mathscr{L}$ is as defined in (2.53)
    Update Λ_1, Λ_2 and ρ via

$$\begin{aligned} \Lambda_1^{k+1} &= \Lambda_1^k + \rho^k (Z^{k+1} - X^{k+1}(Y^{k+1})^T) \\ \Lambda_2^{k+1} &= \Lambda_2^k + \rho^k (X^{k+1} - Y^{k+1}) \\ \rho^{k+1} &= \alpha \rho^k \end{aligned} \tag{2.56}$$

    if $\max\{\|X^k - Y^k\|, \|Z^k - X^k (Y^k)^T\|\} \leq \varepsilon$ **then**
        break
    end if
end for

4.2.4.1 Solution for Subproblems in ADMM

For notation ease, we omit the iteration index k. To update Y in (2.55), we solve the first order optimality condition, which reduces to the following linear system:

$$Y = \left(\frac{1}{\rho}(\Lambda_1^T X + \Lambda_2) + Z^T X + X\right)(I + X^T X)^{-1}.$$

For $r = \lceil \sqrt{2n} \rceil$, the size of the matrix to be inverted is of the order $O(\sqrt{n})$; for $r = 1$, it is a scalar and inversion is trivial.

To update (Z, X) in (2.55), we similarly need to solve an equality-constrained quadratic problem. Define the linear map $\mathscr{A}(Z) = \mathbf{diag}(Z)$ selecting the diagonal of a symmetric matrix, and its adjoint operator $\mathscr{A}^*(v) = \mathbf{Diag}(v)$ producing a diagonal matrix from a vector. Then the optimality conditions of (2.55) are

$$\begin{aligned} C - \mathscr{A}^*(v) + \Lambda_1 + \rho(Z - XY^T) &= 0 \\ \Lambda_2 - \Lambda_1 Y + \rho(XY^T - Z)Y + \rho(X - Y) &= 0 \\ \mathscr{A}(X) &= \mathbf{1}, \end{aligned}$$

where $v \in \mathbb{R}^m$ is the dual variable for the local constraint $\mathscr{A}(X) = b$.

Given Y, solving for v reduces to solving

$$Gv = \rho(b - \mathscr{A}(DY^T)) + \mathscr{A}[(C + \Lambda_1)(I + YY^T)]$$

where

$$D = \frac{1}{\rho}(\Lambda_1 Y - \Lambda_2) + Y, \quad G = \mathscr{A}^*(\mathscr{A}(I + YY^T)).$$

Since G is a diagonal positive definite matrix then finding G^{-1} is computationally simple. Then

$$X = BY + D, \quad \text{and} \quad Z = XY^T + B,$$

where

$$B = -\frac{1}{\rho}(C - \mathscr{A}^*(v) + \Lambda_1).$$

Note that the complexity of all inversions are $O(r^3)$, and are particularly simple if $r = 1$. In that case, the complexity is dominated by the matrix multiplications.

4.2.4.2 Convergence Analysis

Assumption 4.2.9. *The sequences* $\{\mathbf{trace}(CZ^k)\}$ *and* $\{\Lambda_1^k, \Lambda_2^k\}$ *are bounded in norm.*

Theorem 4.2.10. *If Assumption 4.2.9 holds, and given a sequence* $\{\rho^k\}$ *such that*

$$\rho^k > 0, \sum \frac{\rho^{k+1}}{(\rho^k)^2} < \infty, \text{ and } \sum \frac{1}{\rho^k} < \infty,$$

then $\{Z^k, X^k, Y^k\}$ *generated by Algorithm 3 will globally converge to a stationary point of (2.52).*

Proof. To simplify notation, we first collect the primal and dual variables $P^k = (Z, X, Y)^k$ and $D^k = (\Lambda_1, \Lambda_2)^k$.

Lemma 4.2.11. $\mathscr{L}(P^k; D^k; \rho^k)$ *is bounded.*

Proof. Recall the definition of $\mathscr{L}(P^k; D^k; \rho^k)$

$$\begin{aligned}\mathscr{L}(P^k; D^k; \rho^k) = \mathbf{trace}(CZ^k) & \\ + \frac{\rho^k}{2}\|Z^k - X^k(Y^k)^T + \frac{\Lambda_1^k}{\rho^k}\|_F^2 & \\ + \frac{\rho^k}{2}\|X^k - Y^k + \frac{\Lambda_2^k}{\rho^k}\|_F^2 & \\ - \frac{1}{2\rho^k}\left(\|\Lambda_1^k\|_F^2 + \|\Lambda_2^k\|_F^2\right) > -\infty & \quad (2.57)\end{aligned}$$

where the inequality follows the boundedness of $\{\mathbf{trace}(CZ^k)\}$ and $\{\Lambda_1^k, \Lambda_2^k\}$ in Assumption 4.2.9. □

Lemma 4.2.12.

$$\nabla^2 \mathscr{L}_Y \succeq \rho^k I$$

and

$$\nabla^2 \mathscr{L}_{(X,Z)} \succeq \rho^k \left(1 - \frac{\sqrt{\lambda_N^2 + 4\lambda_N} - \lambda_N}{2}\right) I.$$

Proof. Given the definition of $\mathscr{L}$, we can see that the Hessian

$$\nabla^2 \mathscr{L}_Y = \rho^k (M + I) \succeq \rho^k I$$

where

$$M = \textbf{blkdiag}((X^k)^T(X^k), (X^k)^T(X^k), ...) \succeq 0.$$

For (X,Z), we have

$$\nabla^2_{(X,Z)} \mathscr{L}_k = \rho^k \begin{bmatrix} I + NN^T & -N \\ -N^T & I \end{bmatrix}$$

where $N = \textbf{blkdiag}((Y^{k+1})^T, \ldots, (Y^{k+1})^T) \in \mathbb{R}^{nr \times n^2}$. Note that for block diagonal matrices, $\|N\|_2 = \|Y^{k+1}\|_2$. Since $I \succ \mathbf{0}$ and its Schur complement $(I + NN^T) - NN^T \succ \mathbf{0}$, then $\nabla^2_{(X,Z)} \tilde{\mathscr{L}}_k \succ \mathbf{0}$ and equivalently $\lambda_{\min}(\nabla^2_{(X,Z)} \mathscr{L}_k) > 0$.

To find the smallest eigenvalue $\lambda_{\min}(\nabla^2_{(X,Z)} \mathscr{L}_k)$, it suffices to find the largest $\sigma > 0$ such that

$$\begin{aligned} H_2 = (\rho^k)^{-1} \nabla^2_{(X,Z)} \tilde{\mathscr{L}}_k - \sigma I \\ = \begin{bmatrix} (1-\sigma)I + NN^T & -N \\ -N^T & (1-\sigma)I \end{bmatrix} \succeq \mathbf{0}. \quad (2.58) \end{aligned}$$

Using the same Schur Complement trick, we want to find the largest $\sigma > 0$ where $(1-\sigma)I \succeq 0$ and

$$H_3 = (1-\sigma)I + NN^T(1 - (1-\sigma)^{-1})) \succeq 0.$$

Since this implies $1 - \sigma > 0$, then together with $\sigma > 0$, it must be that $1 - (1-\sigma)^{-1} < 0$.

Therefore, defining $\lambda_N = \|Y^{k+1}\|_2^2$,

$$\lambda_{\min}(H_3) = (1-\sigma) + \lambda_N(1-(1-\sigma)^{-1}).$$

We can see that $(1-\sigma)\lambda_{\min}(H_3)$ is a convex function in $(1-\sigma)$, with two zeros at

$$1-\sigma = \frac{\pm\sqrt{\lambda_N^2+4\lambda_N}-\lambda_N}{2}.$$

In between the two roots, $\lambda_{\min}H_3 < 0$. Since the smaller root cannot satisfy $1-\sigma > 0$, we choose

$$\sigma_{\max} = 1 - \frac{\sqrt{\lambda_N^2+4\lambda_N}-\lambda_N}{2} > 0$$

as the largest feasible σ that maintains $\lambda_{\min}(H_3) \geq 0$.

As a result,

$$\lambda_{\min}(\nabla^2_{(X,Z)}\mathscr{L}) = \rho^k\sigma_{\max} = \rho^k\left(1 - \frac{\sqrt{\lambda_N^2+4\lambda_N}-\lambda_N}{2}\right).$$

□

We now prove Thm. 4.2.10.

Proof. For the update of Y, taking $\{D,\rho\} = \{D^k,\rho^k\}$, for the update of Y in (2.54), we have

$$\begin{aligned}
&\mathscr{L}(Z^k,X^k,Y^{k+1};D^k;\rho^k) - \mathscr{L}(P^k;D^k;\rho^k) \\
&\qquad \leq \langle \nabla_Y \mathscr{L}(Z^k,X^k,Y^{k+1};D^k;\rho^k), Y^{k+1}-Y^k\rangle \\
&\qquad\quad - \frac{\lambda_{\min}(\nabla^2_Y\bar{\mathscr{L}_k})}{2}\|Y^{k+1}-Y^k\|_F^2 \\
&\qquad\qquad \leq -\frac{\rho^k}{2}\|Y^{k+1}-Y^k\|_F^2 \quad (2.59)
\end{aligned}$$

with $\nabla^2_Y \bar{\mathscr{L}}_k = \nabla^2_{\mathrm{vec}(Y)} \mathscr{L}(Z^k, X^k, Y^{K+1}; D^k; \rho^k) \succeq \rho^k I$.

which follows from the ρ^k-strong convexity of $\mathscr{L}$ with respect to Y, and the optimality of Y^{k+1}.

For the update of (Z, X) in (2.54), we have

$$\begin{aligned}
&\mathscr{L}(P^{k+1}; D^k; \rho^k) - \mathscr{L}(Z^k, X^k, Y^{k+1}; D^k; \rho^k) \\
&\qquad \leq \langle \nabla_Z \mathscr{L}(P^{k+1}; D^k; \rho^k), Z^{k+1} - Z^k \rangle \\
&\qquad + \langle \nabla_X \mathscr{L}(P^{k+1}; D^k; \rho^k), X^{k+1} - X^k \rangle \\
&\quad - \frac{\lambda_{\min}(\nabla^2_{(X,Z)} \tilde{\mathscr{L}}_k)}{2} \left(\|Z^{k+1} - Z^k\|_F^2 + \|X^{k+1} - X^k\|_F^2 \right) \\
&\qquad \leq - \frac{\lambda_{\min}(\nabla^2_{(X,Z)} \tilde{\mathscr{L}}_k)}{2} \left(\|Z^{k+1} - Z^k\|_F^2 + \|X^{k+1} - X^k\|_F^2 \right), \quad (2.60)
\end{aligned}$$

where the first inequality follows from the strong convexity of $\mathscr{L}(Z, X, Y^k; \Lambda_1^k, \Lambda_2^k)$ with respect to (X, Z) with $\nabla^2_{(X,Z)} \tilde{\mathscr{L}}_k = \nabla^2_{\mathrm{vec}(X,Z)} \mathscr{L}(P^{k+1}; D^k; \rho^k)$. In addition, the second inequality follows the optimality condition of (2.55). Note that $\lambda_{\min}(\nabla^2_{(X,Z)} \tilde{\mathscr{L}}) > 0$ given $\mathscr{L}(Z, X, Y^k; \Lambda_1^k, \Lambda_2^k)$ is strongly convex with regard to (X, Z).

In addition for the update of the dual variables and the penalty coefficient, we have

$$\begin{aligned}
&\mathscr{L}(P^{k+1}; D^{k+1}; \rho^{k+1}) - \mathscr{L}(P^{k+1}; D^k; \rho^k) \\
&\qquad = \langle {\Lambda_1}^{k+1} - {\Lambda_1}^k, Z^{k+1} - X^{k+1} (Y^{k+1})^T \rangle \\
&\qquad + \langle {\Lambda_2}^{k+1} - {\Lambda_2}^k, X^{k+1} - Y^{k+1} \rangle \\
&\quad + \frac{\rho^{k+1} - \rho^k}{2} (\|Z^{k+1} - X^{k+1} (Y^{k+1})^T\|_F^2) \\
&\qquad + \frac{\rho^{k+1} - \rho^k}{2} (\|X^{k+1} - Y^{k+1}\|_F^2) \\
&\qquad = \frac{\rho^{k+1} + \rho^k}{2(\rho^k)^2} \left(\|{\Lambda_1}^{k+1} - {\Lambda_1}^k\|_F^2 + \|{\Lambda_2}^{k+1} - {\Lambda_2}^k\|_F^2 \right) \quad (2.61)
\end{aligned}$$

where the first equality follows the definition of $\mathscr{L}$, the second follows (2.56). Moreover, it can be easily verified that

$$\begin{aligned}
&\mathscr{L}(P^{k+1};D^{k+1};\rho^{k+1})-\mathscr{L}(P^k;D^k;\rho^k)\\
&\qquad=\mathscr{L}(Z^k,X^k,Y^{k+1};D^k;\rho^k)-\mathscr{L}(P^k;D^k;\rho^k)\\
&\qquad+\mathscr{L}(P^{k+1};D^k;\rho^k)-\mathscr{L}(Z^k,X^k,Y^{k+1};D^k;\rho^k)\\
&\qquad+\mathscr{L}(P^{k+1};D^{k+1};\rho^{k+1})-\mathscr{L}(P^{k+1};D^k;\rho^k). \quad (2.62)
\end{aligned}$$

By incorporating (2.60), (2.59) and (2.61), (2.62) can be rewritten as

$$\begin{aligned}
&\mathscr{L}(P^{k+1};D^{k+1};\rho^{k+1})-\mathscr{L}(P^k;D^k;\rho^k)\\
&\qquad\leq -c_1^k\|Z^{k+1}-Z^k\|_F^2-c_2^k\|X^{k+1}-X^k\|_F^2\\
&\qquad\qquad-c_3^k\|Y^{k+1}-Y^k\|_F^2\\
&\qquad+\frac{\rho^{k+1}+\rho^k}{2(\rho^k)^2}\left(\|{\Lambda_1}^{k+1}-{\Lambda_1}^k\|_F^2+\|{\Lambda_2}^{k+1}-{\Lambda_2}^k\|_F^2\right). \quad (2.63)
\end{aligned}$$

with $c_1^k=c_2^k=\frac{\lambda_{\min}(\nabla^2_{(Z,X)}\tilde{\mathscr{L}}_k)}{2}>0$ as given in Lemma 4.2.12 and $c_3^k=\frac{\rho^k}{2}>0$.

By telescoping, the summation

$$\begin{aligned}
&\mathscr{L}(P^K;D^K;\rho^K)-\mathscr{L}(P^0;D^0;\rho^0)\\
=&\sum_{k=0}^{K-1}\mathscr{L}(P^{k+1};D^{k+1};\rho^{k+1})-\mathscr{L}(P^k;D^k;\rho^k)\\
\leq&\sum_{k=0}^{K-1}\frac{\rho^{k+1}+\rho^k}{2(\rho^k)^2}\left(\|\Lambda_1^{k+1}-\Lambda_1^k\|_F^2+\|\Lambda_2^{k+1}-\Lambda_2^k\|_F^2\right)\\
&-\sum_{k=0}^{K-1}c^k\Big(\|Z^{k+1}-Z^k\|_F^2+\|X^{k+1}-X^k\|_F^2\\
&+\|Y^{k+1}-Y^k\|_F^2\Big),
\end{aligned}$$

with $c^k=\min(c_1^k,c_2^k,c_3^k)>0$.

For the first term,

$$\begin{aligned} 0 &\leq \sum_{k=0}^{K-1} \frac{\rho^{k+1}+\rho^k}{2(\rho^k)^2}\left(\|\Lambda_1^{k+1}-\Lambda_1^k\|_F^2+\|\Lambda_2^{k+1}-\Lambda_2^k\|_F^2\right) \\ &\leq 4\sum_{k=0}^{K-1} \frac{\rho^{k+1}+\rho^k}{(\rho^k)^2}B \end{aligned}$$

where from Assumption 4.2.9 we have that $\{\Lambda_1^k, \Lambda_2^k\}$ is bounded; that is there exists $B < \infty$ where $B \geq \max\{\|\Lambda_1^k\|_F^2, \|\Lambda_2^k\|_F^2\}$ for all k. Since additionally $\sum \frac{\rho^{k+1}+\rho^k}{2(\rho^k)^2} \leq \sum \frac{\rho^{k+1}}{(\rho^k)^2} < +\infty$, then

$$0 \leq \sum_{k=0}^{K-1} \frac{\rho^{k+1}+\rho^k}{2(\rho^k)^2}\left(\|\Lambda_1^{k+1}-\Lambda_1^k\|_F^2+\|\Lambda_2^{k+1}-\Lambda_2^k\|_F^2\right) \leq +\infty.$$

Since $\mathscr{L}(P^k; D^k; \rho^k)$ is bounded, then it follows that

$$0 \leq \sum_{k=0}^{K-1} c^k\Big(\|Z^{k+1}-Z^k\|_F^2+\|X^{k+1}-X^k\|_F^2 + \|Y^{k+1}-Y^k\|_F^2\Big) \leq +\infty.$$

Since additionally $\sum_k \rho_k = +\infty$, this immediately yields $Z^{k+1}-Z^k \to 0, X^{k+1}-X^k \to 0, Y^{k+1}-Y^k \to 0$.

Moreover,

$$\sum_{k=1}^{\infty} \frac{1}{\rho^k} < \infty \iff \frac{1}{\rho^k} \to 0 \iff \rho^k \to \infty.$$

Combined with Assumption 4.2.9, this implies that

$$Z^{k+1} - X^{k+1}(Y^{k+1})^T = \frac{1}{\rho^k}(\Lambda_1^{k+1}-\Lambda_1^k) \to 0,$$

$$X^{k+1} - Y^{k+1} = \frac{1}{\rho^k}(\Lambda_2^{k+1}-\Lambda_2^k) \to 0.$$

Therefore the limit points X^*, Y^*, and Z^* are all feasible, and simply checking the first optimality condition will verify that this accumulation point is a stationary point of (2.52). The proof is complete.

□

□

Corollary 4.2.13. *If $r \geq \lceil \sqrt{2n} \rceil$ and the stationary point of Algorithm 3 converges to a second order critical point of* (2.52)*, then it is globally optimal for the convex relaxation of* (2.31) *[26].*

Unfortunately, the extension of KKT stationary points to global minima is not yet known when $\frac{r(r+1)}{2} < n$ (*i.e.*, $r = 1$). However, our empirical results suggest that even when $r = 1$, often a local solution to (2.52) well-approximates the global solution to (2.28).

4.2.5 Numerical Results

From the convergence proofs, we have shown that our algorithms converge to a KKT stability point, and if $r = \lceil \sqrt{2n} \rceil$, with high probability achieves the global solution of the SDR. In this section, we give empirical evidence that they are in addition good solutions to (2.28), our original combinatorial problem. We show this through three applications: community detection of the stochastic block model, MAX-CUT, and image segmentation.

We evaluate four methods for solving (2.28):

1. SD: the solution to the SDR rounded to a binary vector using a Goemans-Williamson style rounding [77] technique;

2. V: the binary vector reformulation (2.32) solved via ADMM;

3. MR1: the matrix reformulation (2.52) with $r = 1$, solved via ADMM;

4. MRR: the matrix reformulation (2.52) with $r = \lceil \sqrt{2n} \rceil$, solved via ADMM, and rounded to a binary vector using a nonsymmetric version of the Goemans-Williamson style rounding [77] technique.

4.2.5.0.1 Rounding For both the SDP and MRR methods, we need to round the solutions to a vector. For the SDP method, we first do an eigenvalue decomposition $X = Q\Lambda Q^T$ and form a factor $F = Q\Lambda^{1/2}$ where the diagonal elements of Λ are in decreasing magnitude order. Then we scan $k = 1,\ldots,n$ and find $x_{k,t} = \mathbf{sign}(F_k z_t)$ for trials $t = 1,\ldots 10$. Here, F_k contain the first k columns of F, and each element of z_t is drawn i.i.d from a normal Gaussian distribution. We keep $x_r = x_{k,t}$ that minimizes $x_r^T C x_r$. For the MRR method, we repeat the procedure using a factor $F = U\Sigma^{1/2}$ where $X = U\Sigma V^T$ is the SVD of X. For MR1 and V, we simply take $x_r = \mathbf{sign}(x)$ as the binary solution.

4.2.5.0.2 Computer information The following simulations are performed on a Dell Poweredge R715 with an AMD Opteron(tm) processor, with 64 GB of RAM and 24 cores. It is running Ubuntu 16.04 (Xenial) with Matlab 2017a.

4.2.5.1 Community detection

In this section we evaluate C as defined for community detection. Figure 4.5 gives a sample evolution for each method with $n = 2500$, $m = n/2$, and $p = 10q = 0.04$. Although the vector method is simple and has a fast per-iteration rate, it has significant initial overhead and slow overall progress. In comparison, the two matrix methods MR1 and MRR are significantly faster. The SDR method has a slow per iteration rate, but suggests good answers in only a few iterations. However, this plot does not take into account rounding overhead time (see figure 4.6), which is significant for both the SDR and MRR methods.

Table 4.2.5.1 gives the average recovery rate for each method on the community detection problem for the stochastic block model. The results are surprisingly consistent, and suggest that solving (2.28) using our approaches all have great promise for general community detection problems.

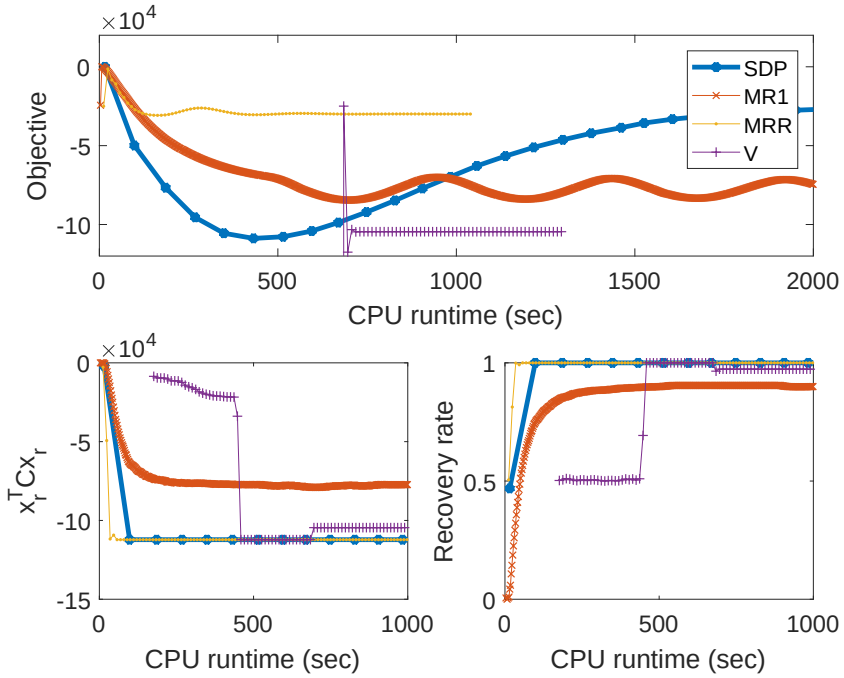


Figure 4.5: Sample evolution for $n = 2500$, showing the objective value $\mathbf{trace}(CX)$, best objective value using $x_r = \mathbf{sign}(x)$, and recovery rate using x_r.

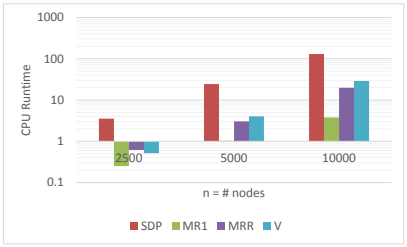
4.2.5.2 MAX-CUT

Table 4.3 gives the best MAX-CUT values using best-of-random-guesses and our approaches over four examples from the 7th DIMACS Implementation Challenge in 2002.[i] Our solutions are very close to the best known solutions.

4.2.5.3 Image segmentation

Both community detection and MAX-CUT can be used in image segmentation, where each pixel is a node and the similarity between pixels form the weight of the edges. Generally, solving (2.28) for this application is not preferred, since the number of pixels in even a

[i]See `http://dimacs.rutgers.edu/Workshops/7thchallenge/`. Problems downloaded from `http://www.optsicom.es/maxcut/`



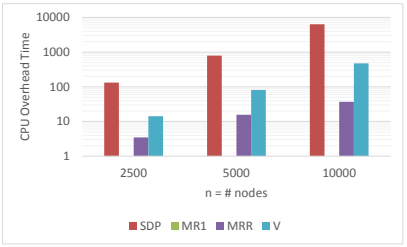


Figure 4.6: Top: CPU per-iteration runtime for four methods. Bottom: CPU runtime for overhead operations (rounding for SDP and MRR and initial matrix inversion or SVD for V). MR1 gives a vector output and has no additional overhead.

n	m	p	S	V	MR1	MRR
1000	100	0.1	1	1	1	1
1000	500	0.1	1	1	1	1
2500	250	0.04	1	1	1	1
2500	1250	0.04	1	1	1	1.00
5000	500	0.02	1	1	1	1.00
5000	2500	0.02	1	1	1	1.00
10000	1000	0.01	1	1	1	1
10000	5000	0.01	1	1	1	1.00

Table 4.2: **Correct community recovery rate for stochastic block model.** Result after 10 iterations for S, MR1, and MRR methods and 50 iterations for V, averaged over 10 trials for each method. n = number of nodes in graph. m = number of nodes in smaller of two communities. p = percentage of edges within communities, and $q = p/10$ the percentage of edges between communities. 1 = perfect recovery, 1.00 = rounds to 1.

	g3-15	g3-8	pm3-15-50	pm3-8-50
n	3375	512	3375	512
# edges	20250	3072	20250	3072
sparsity	0.18%	1.2%	0.18%	1.2%
BK	281029888	41684814	2964	454
R	12838418	4816306	170	62
MR1	255681256	36780180	1990	312
V	202052290	8213762	2016	330

Table 4.3: MAX-CUT values for graphs from the 7th DIMACS Challenge. BK = best known. R = best of 1000 random guesses. MR1 = matrix formulation, $r = 1$. V = vector formulation.

moderately sized image is extremely large. However, because of our fast methods, we successfully performed image segmentation on several thumbnail-sized images, in figure 4.7.

The C matrix is composed as follows. For each pixel, we compose two feature vectors: f_c^{ij} containing the RGB values and f_p^{ij} containing the pixel location. Scaling f_c^{ij} by some weight c, we form the concatenated feature vector $f^{ij} = [f_c^{ij}, cf_p^{ij}]$, and form the weighted adjacency matrix as the squared distance matrix between each feature vector $A_{(ij),(kl)} = \|f^{ij} - f^{kl}\|_2^2$. For MAX-CUT, we again form $C = A - \mathbf{Diag}(A\mathbf{1})$ as before. For community detection, since we do not have exact p and q values, we use an approximation as $C = a\mathbf{1}\mathbf{1}^T - A$ where $a = \frac{1}{n^2}\mathbf{1}^T A\mathbf{1}$ the mean value of A. Sweeping C and ρ_0, we give the best qualitative result in figure 4.7.

4.2.6 Conclusion

We present two methods for solving quadratic combinatorial problems using ADMM on two reformulations. Though the problem has a nonconvex constraint, we give convergence results to KKT solutions under much milder conditions than previously shown. From this, we give empirical solutions to several graph-based combinatorial problems, specifically MAX-CUT and community detection; both can can be used in additional downstream applications, like image segmentation.

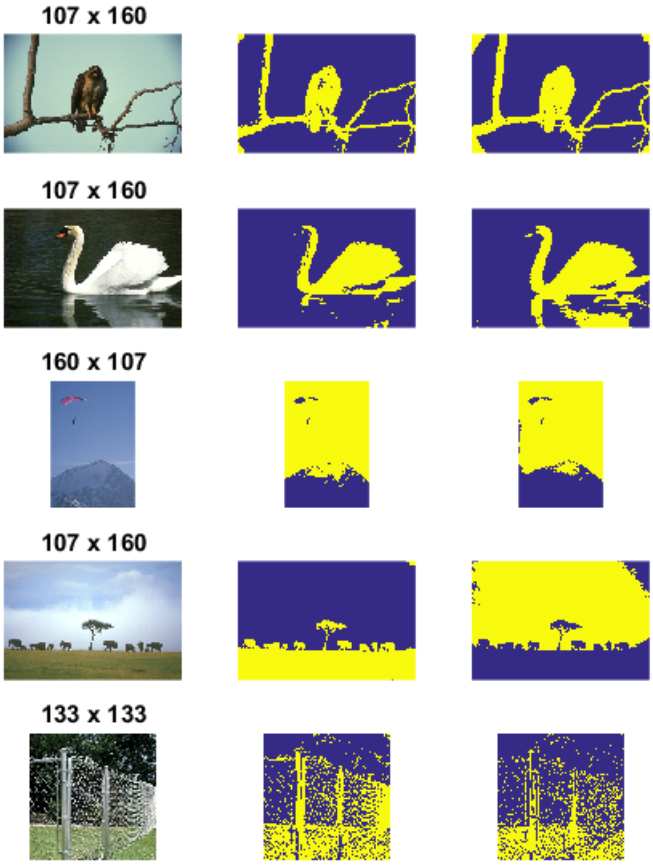


Figure 4.7: Image segmentation. The center (right) column are the best MAX-CUT (community detection) results.

Chapter 5: Inexact Augmented Lagrangian Method for QCQP

5.1 Introduction

Many real world complex systems with multiple subsystems in various disciplines, ranging from sociology, biology, engineering, information technology, just to name a few, can be abstracted as networks. In those systems, instead of manipulating the system as a whole, e.g., through decentralized control, people wish to find the modules of a networks to suppress the complexity. Besides, it is beneficial to better understand the interactions of nodes, as well as, the underlying property of the complex networks. Module detection is to group the nodes into modules such that the intra-module connectivity is obviously denser that the inter-module one. Consequently, module detection has attracted great attentions in the study of complex networks.

Traditional approaches for solving network module detection problems typically include graph partitioning [22] and hierarchical clustering methods [93]. The main disadvantage for the former is the unreliable iterative bisectioning to split the graph and for the latter is the uncertainties to choose a partition among many options to better represent the module structure. Moreover, the divisive algorithms [156] has been developed to select edges according to their contribution importance to a specific performance index. In addition,

methodologies based on partitional clustering and spectral clustering have been proposed [54, 129].

One challenge in network module detection problem is to quantify the module structure in a network. Newman in [156] proposed the notion *modularity*, which is subsequently accepted as the benchmark performance index for the module networks despite the resolution limit [68]. After introducing the concept of modularity, efforts on solving the module detection problems have been focused on maximizing the modularity of a network via optimization approaches. Furthermore, work in [158] demonstrates the equivalence between the method of modularity maximization and the method of maximum likelihood. Following the definition of modularity and justification of the model, Newman developed a series of algorithms to detect the number of modules (prerequisite of many existing algorithms) in a network [157] and to find these modules, including the matrix eigenvector method [155] and the greedy algorithm [154]. Variants of greedy algorithms have also been developed: see [21]. Other modularity optimization based algorithms or techniques include genetic algorithms [197], simulated annealing [83], extremal optimization [55], and etc. More details are addressed in the comprehensive survey [67]. On the other hands, the network module detection problem is also formulated as a Mixed Integer Quadratic Programming [212] problem and solved via commercial optimization solver, such as CPLEX. However, existing algorithms are subject to low efficiency when solving large-scale module detection problems. It has been recognized that the heuristic algorithms, such as genetic algorithm and simulated annealing, are computationally demanding. Furthermore, due to the NP-hardness of MIQP, the corresponding algorithms in the worst-case are exponential complexity and most are inefficient [66, 186, 191].

Based on the NP-hard nature of the modularity maximization [29], it is unrealistic to develop an efficient optimization algorithm to search for its global optimal solution within polynomial computation time. As a tradeoff or compromise, we aim to develop a local optimization algorithm with reasonable computation time. We first reformulate the module detection problem as a MIQP, which is equivalent to the one formulated in [212]. The new formulation is more compact and concise, i.e., with fewer number of variables and constraints, which facilitates developing a time and memory efficient algorithm that does not require employing commercial solvers. Algorithms for MIQPs generally include two categories. One is global optimization technique, such as branch and bound [66]. The other is convex relaxations, such as semidefinite relaxation [162]. Though relaxation techniques, in most cases, can only find a lower bound and not even a feasible one, there are some theoretical guarantee for special cases like the max-cut problems [77]. While the former is not scalable, solving a semidefinite programming for the latter is also computationally prohibitive in medium and large-scale problems.

In this paper, an algorithm based on inexact Augmented Lagrangian Method (ALM) is developed to solve the network module detection problem formulated as an MIQP. In the proposed inexact ALM, we partition the variables into two blocks, which leads to a closed form solution of each subproblem. In addition, we adequately exploit the special structure and sparsity of the problem settings to simplify the closed form solutions and save memory. The rest of the paper is organized as follows. In §II, the problem formulation of network module detection as an MIQP is described. The algorithm framework and the subproblem solutions are addressed in §III. Simulation results on various real-world databases with varying scales are presented in §IV. We conclude the paper with a few remarks in §V.

5.1.1 Preliminary

For a matrix $X \in \mathbb{R}^{m\times n}$, X_{ij} denotes its entry on the ith row and jth column. Similarly, $X_{i,j_1:j_2}$ represent the entries in ith row and from j_1th column to j_2th column. $X_{:,i}$ represents the $i$the column of X. Moreover, we denote $x := \text{vec}(X) \in \mathbb{R}^{mn}$ as vectorization of X by stacking the columns of X on top of one another in order. Correspondingly, $X := \text{mat}(x)$ is the reverse operation. For two matrices X and Y, $X \otimes Y$ represents the Kronecker matrix product. $\|X\|_F$ represents the Frobenius norm. $\textbf{blkdiag}(X_1,\ldots,X_n)$ represents a block diagonal matrix with square matrices $X_1,\ldots,X_n$ on the diagonal. For two matrices X and Y with the same dimensions, $X \circ Y$ represents the element-wise multiplication. Correspondingly, $X \oslash Y$ represents the element-wise division. For a vector x, $\textbf{Diag}(x)$ represents a diagonal matrix with x as the diagonal entries.

5.2 Problem Formulation

Modularity, the quantity to measure the difference between the intra-community and inter-community connection, is first defined in [156]. Its formulation is as follows

$$Q = \frac{1}{2m}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j), \tag{2.1}$$

where A is the adjacency matrix, k_i is the degree of node i, n is the number of the ndoes and m is the number of the total edges. In this paper, we focus on undirected networks. Moreover, C_i and C_j are the communities to which vertex i and j belong to, respectively. Then $\delta(C_i, C_j)$ is defined as

$$\delta(C_i, C_j) = \begin{cases} 1, & C_i = C_j \\ 0, & C_i \neq C_j \end{cases}. \tag{2.2}$$

As we aim to detect the community in the network by maximizing the modularity function, it will be computationally favorable if $\delta(C_i, C_j)$ can be represented by a continuously

differentiable function. For a two-community network, a bivalent vector, $s \in \{-1,1\}^n$, is introduced to define $\delta(C_i,C_j) = \frac{1}{2}(s_i s_j + 1)$ with $s_i = -1$ ($s_i = 1$, resp.) representing that vertex i is within the designated community (the other community, resp.). However, for a network with possible k communities ($k > 2$), a single column vector like s is insufficient to describe the relationship. Correspondingly, we define a binary matrix $X \in \{0,1\}^{k\times n}$ such that

$$X_{ij} = \begin{cases} 1, \text{ if node } \ j \in \text{community } i \\ 0, \text{ otherwise} \end{cases}.$$

Moreover, as we exclude the cases with overlapping communities, i.e., one vertex belongs to one and only one community, it adds the following constraint on X,

$$\sum_{i=1}^{k} X_{ij} = 1, \forall j = 1,\ldots,n. \tag{2.3}$$

Consequently, $\delta(C_i,C_j) = X_{:,i}^T X_{:,j}$ represents its exact definition in (2.2), where $X_{:,i}$ is the ith column of matrix X. For notation ease, we define $B \in \mathbb{S}^n$ such that $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$. Accordingly, the modularity in (2.1) can be rewritten as

$$\begin{aligned} Q &= \frac{1}{2m}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(A_{ij} - \frac{k_i k_j}{2m}\right)X_i^T X_j \\ &= \frac{1}{2m}x^T\Big(B \otimes I_k\Big)x, \end{aligned} \tag{2.4}$$

where $x \in \mathbb{R}^{nk} := \text{vec}(X)$ is a vectorization of X. Furthermore, the equality constraint in (2.3) can be rewritten with regard to x in the form of

$$\sum_{l=k(j-1)+1}^{kj} x_l = 1, \forall j = 1,\ldots,n \iff Ex = \mathbf{1}, \tag{2.5}$$

with $E \in \mathbb{R}^{n\times nk}$, $E_{j,k(j-1)+1:kj} = 1$ and zeros otherwise for $j = 1,\ldots,n$.

Maximizing modularity in network module detection means to search for the best module structure to suppress complexity that the intra-module connectivity is much denser that the

inter-module one. As a summary, the network module detection problem can be formulated as

$$\begin{aligned} \min_{x} \quad & x^T M x \\ \text{subject to} \quad & Ex = \mathbf{1} \\ & x \in \{0,1\}^{nk}, \end{aligned} \tag{2.6}$$

with $M = -\frac{1}{2m}\left(B \otimes I_k\right)$. Problem (2.6) is a Mixed Integer Quadratic Programming (MIQP) problem and generally NP-hard. Although the network module detection problem formulated in (2.6) is equivalent to the one in [212], the former is in a more compact form. The new formulation facilitates generating the proposed algorithm for solving (2.6), which is described in the following sections.

5.3 Network Module Detection Algorithms

5.3.1 Classical Alternating Direction Method of Multipliers

Given binary variables are involved in (2.6), we start with decoupling the binary constraints on x from it linear equality constraint, $Ex = \mathbf{1}$. An auxiliary variable y is introduced and problem in (2.6) is reformulated as

$$\begin{aligned} \min_{x,y} \quad & x^T M x \\ \text{subject to} \quad & Ex = \mathbf{1} \\ & y \in \{0,1\}^{nk} \\ & x = y. \end{aligned} \tag{3.7}$$

In this way, y is used to carry the binary constraints so that it can be decoupled from the other set of constraints, i.e., the linear equality constraint. It is intuitive to introduce the

classical Alternating Direction Method of Multipliers (ADMM) framework to solve the above problem as ADMM is well known for partitioning variables into desired subsets to simplify computation [189]. In the ADMM framework, the variables in (3.7) are partitioned into two blocks, x and y, and solved alternatively in sequence. When fixing one of the blocks and solving the other, the subproblems in each sequence of ADMM are straightforward. To be more specific, one subproblem is linearly constrained quadratic programming while the other is a projection onto the bivalent constraint. However, though ADMM does work well for some nonconvex problems, the general convergence is not theoretically guaranteed. For this specific problem, the classical ADMM has poor convergence when solving (3.7). Accordingly, a time and storage efficient algorithm, based on the inexact augmented Lagrangian method, is proposed to solve the MIQP problem in (2.6).

5.3.2 The Inexact ALM Algorithm

We first reformulate the binary constraint using a continuous quadratic function, expressed as

$$x_i \in \{0,1\} \iff x^T H_i x - e_i^T x = 0, \forall i = 1,\ldots,n \tag{3.8}$$

where $(H_i)_{ii}$ is 1 and 0 for other entries and e_i is a unit vector with the ith entry being 1 and 0 otherwise. Subsequently, by introducing an auxiliary variable y, (2.6) can be rewritten as

$$\begin{aligned} \min_{x} \quad & x^T M y \\ \text{subject to} \quad & Ex = \mathbf{1} \\ & x^T H_i y - e_i^T x = 0, i = 1,\ldots,n \\ & x = y. \end{aligned} \tag{3.9}$$

Similar to the classical ADMM, introducing y in the above formulation is to simplify the subproblem in each iteration, which will be explained later. In this way, the binary constraint is a coupling one getting both partitions of variables involved, which is different from that in (3.7). To solve (3.9) via the augmented Lagrangian method, the augmented Lagrangian function of (3.7) is constructed first, expressed as

$$\begin{aligned}\mathscr{L}(x,y;\nu,\lambda,\mu) &= x^T My \\ &+\langle \nu, Ex-\mathbf{1}\rangle + \langle \lambda, x^T Hy - e^T x\rangle + \langle \mu, x-y\rangle \\ &+\frac{\rho_1}{2}\|Ex-\mathbf{1}\|_F^2 + \frac{\rho_2}{2}\|x^T Hy - e^T x\|_F^2 + \frac{\rho^k}{2}\|x-y\|_F^2,\end{aligned} \tag{3.10}$$

where $\nu \in \mathbb{R}^n, \lambda \in \mathbb{R}^{nk}$ and $\mu \in \mathbb{R}^{nk}$ are the dual variables of the corresponding constraints and $\rho_1 > 0, \rho_2 > 0, \rho_3 > 0$ are the penalty coefficients for the augmented terms can be adjusted along the iterations. Unlike the augmented Lagrangian function in ADMM, here each constraint, including the local one $Ex = \mathbf{1}$ is also contained. Moreover, for notation simplicity, $x^T Hy - e^T x : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is defined as

$$x^T Hy - e^T x := [x^T H_1 y - e_1^T x, \ldots, x^T H_n y - e_n^T x]^T.$$

In the traditional augmented Lagrangian method, the dual variable sets x and y need be updated simultaneously. However, due to the fact that $\mathscr{L}(x,y;\nu,\lambda,\mu)$ (with the dual variables fixed) is a nonconvex forth order polynomial function, seeking for the global optimum is computationally complicated. However, $\mathscr{L}(x,y;\nu,\lambda,\mu)$ is bi-convex with respect to x and y. Then instead of updating x and y simultaneously, an inexact ALM is proposed that also partitions the variables into two sets, x and y and update one with the other fixed. To solve (3.9), the algorithm alternates between these two variable sets followed by the update of the dual variables (ν,λ,μ). With $(x^h, y^h; \nu^h, \lambda^h, \mu^h)$ obtained at step h, the

subproblems of the inexact ALM at step $h+1$ for (3.9) include

$$x^{h+1} := \arg\min_{x} \mathscr{L}(x, y^h; \nu^h, \lambda^h, \mu^h) \tag{3.11a}$$

$$y^{h+1} := \arg\min_{y} \mathscr{L}(x^{h+1}, y; \nu^h, \lambda^h, \mu^h) \tag{3.11b}$$

$$\nu^{h+1} := \nu^h + \rho_1^h(Ex^{h+1} - \mathbf{1}) \tag{3.11c}$$

$$\lambda^{h+1} := \lambda^h + \rho_2^h((x^{h+1})^T H y^{h+1} - e^T x^{h+1}) \tag{3.11d}$$

$$\mu^{h+1} := \mu^h + \rho_3^h(x^{h+1} - y^{h+1}). \tag{3.11e}$$

5.3.3 Subproblem Solutions

In (3.16), updating x^{h+1} using (3.16a) is to solve an unconstrained strongly convex optimization problem. As a result, the first order optimality conditions listed below will lead to its global optimal solution,

$$\begin{aligned}\frac{\partial \mathscr{L}}{\partial x} &= Cy + E^T\nu + \sum_{i=1}^{n} \lambda_i(H_i y - e_i) + \mu \\ &+\rho_1 E^T(Ex - \mathbf{1}) + \rho_2 \sum_{i=1}^{n} (H_i y - e_l)(H_i y - e_i)^T x \\ &+\rho_3(x - y) = 0.\end{aligned} \tag{3.12}$$

Note that in the above equation, the algorithm step index is ignored for simplicity. Furthermore, the computation cost in (3.12) can be significantly reduced by exploiting the sparsity of H_i as well as e_i. From the quadratic function definition in (3.8), some terms in (3.12) can be simplified as

$$\begin{aligned}&\sum_{i=1}^{n} \lambda_i(H_i y - e_i) = \lambda \circ (y - \mathbf{1}) \\ &\sum_{i=1}^{n} (H_i y - e_i)(H_i y - e_i)^T = \mathbf{Diag}\Big((y - \mathbf{1}) \circ (y - \mathbf{1})\Big).\end{aligned} \tag{3.13}$$

Now the focus is to solve the linear equality derived from (3.12). For conciseness, we rewrite it as $G_x x = b_x$, where $G_x = \rho_1 E^T E + \rho_2 \mathbf{Diag}\Big((y - \mathbf{1}) \circ (y - \mathbf{1})\Big) + \rho_3 I$ and

$b_x = -\Big(Cy + E^T\nu + \lambda \circ (y - \mathbf{1}) + \mu - \rho_1 E^T\mathbf{1} - \rho_3 y\Big)$. For large-scale network module detection problems, the dimension of $G_k \in \mathbb{S}^{nk}$ will increase dramatically. The direct matrix inverse is not applicable here due to its computational complexity (at least $O((nk)^{2.373})$ so far). Similarly, the backslash (left division) is also time consuming due to the same reason (complexity of $O((nk)^2)$). Subsequently, this motivates us to fully exploit the problem data structure and sparsity. As G_k is the sum of three components, it shows that the last two are diagonal and the first satisfies $E^TE = I_n \otimes \mathbf{1}_{k\times k} = \mathbf{blkdiag}(\mathbf{1}_{k\times k}, \ldots, \mathbf{1}_{k\times k}, \ldots, \mathbf{1}_{k\times k}) \in \mathbb{R}^{nk\times nk}$. In other words, G_x is a block diagonal matrix with n square blocks of size k. As $k \ll n$ in realistic problem settings, x can be solved via block-wise back slash. In one word, (3.13) can be solved block-wisely as,

$$\begin{aligned} x_{\mathscr{I}_i} = \Big(\rho_1 \mathbf{1}_{k\times k} + \rho_2 \mathbf{Diag}\big((y_{\mathscr{I}_i} - \mathbf{1}) \circ (y_{\mathscr{I}_i} - \mathbf{1})\big) + \rho_3 I_k\Big) \\ \backslash (b_x)_{\mathscr{I}_i}, \forall i = 1, \ldots, n \end{aligned} \tag{3.14}$$

where $\mathscr{I}_i = \big((i-1)k + 1 : ik\big)$ is the index set. There are two advantages in solving (3.13) both computation-wisely and memory-wisely. For the former, with complexity $O(nk^2)$, it avoids large-scale matrix inversion ($\geq O((nk)^{2.373})$) or left division ($O((nk)^2)$), and full size matrix multiplication by taking advantage of the block diagonal structure. Recall $k \ll n$, this reduction is significant. As for the latter, instead of caching $G_x \in \mathbb{S}^{nk}$, it suffices to just store $y \in \mathbb{R}^{nk}$, which substantially decreases the memory burden.

In a parallel scheme, the optimum of (3.16b) can be achieved by solving

$$\begin{aligned} &\frac{\partial \mathscr{L}}{\partial y} = Cx + \sum_{i=1}^{n} \lambda_i H_i x - \mu \\ &+\rho_2 \sum_{i=1}^{n} (H_i x)\Big((H_i x)^T y - e_i^T x\Big) + \rho_3 (y - x) = 0. \end{aligned} \tag{3.15}$$

Analogously, we have

$$\sum_{i=1}^{n} \lambda_i H_i x = \lambda \circ x$$
$$\sum_{i=1}^{n} (H_i x)\Big((H_i x)^T y - e_i^T x\Big) = \mathbf{Diag}(x \circ x)y - (x \circ x).$$

Since the Hessian of (3.16b) is a nonsingular diagonal matrix, its inverse operation is straightforward. So far, the ease of solving the subproblems validates the advantages of introducing the auxiliary variable y, the constraint $x = y$, and the quadratic constraints. Though (3.9) is less compact compared to (2.6), it transforms the quadratic constraint into bilinear and binary variables into continuous ones. Subsequently, the augmented term in the augmented Lagrangian function is quadratic and convex instead of quartic and concave, which leads to closed form solution of two subproblems of (3.9). Moreover, replacing the original objective $x^T Mx$ by $x^T My$ is to avoid the presence of the dense M in the Hessian matrix in the x-update procedure. Otherwise, G_x will not be a block diagonal matrix and the aforementioned efficient method for $x-$update is no longer applicable. These are the key strategies to enhance the algorithm efficiency and make it promising for large-scale problems.

Recall the definition of $M = -\frac{1}{2m}\big(B \otimes I_k\big) \in \mathbb{S}^{nk}$, which can cause another memory issue. Note that in each iteration, M appears in both subproblems, (3.16a) and (3.16b). However, for any two matrices A_1, B_1 and a vector v in appropriate dimension, we have the following property $(B_1 \otimes A_1)v = \mathrm{vec}(A_1 \mathrm{mat}(v) B_1^T)$. As a result, instead of storing M and execute the full size matrix multiplication, the implementation below in $x-$update and analogously for $y-$update is executed

$$Mx = -\frac{1}{2m}\big(B \otimes I_k\big)x = -\frac{1}{2m}\mathrm{vec}\big(\mathrm{mat}(x)B^T\big).$$

In an analogous way, due to its special structure, the computation related to E can also be simplified such that it is not necessary to cache E either. Updating the Lagrange multipliers in (3.16c)-(3.11e) is straightforward. So far, we end solving the subproblems involved in the inexact ALM algorithm and it is summarized below. For conciseness, we keep M and E therein.

Algorithm 1: Inexact augmented Lagrangian algorithm for problem (3.9)
Input: Problem parameters $n,k,A,B,C,E,\rho_1,\rho_2,\rho_3$, $\alpha_1>0,\alpha_2>0,\alpha_3>0$ and initial point $(x^0,y^0,\nu^0,\lambda^0,\mu^0)$
Output: Local minimizer (x,y) to (3.9)
begin

1. Set $h=0$
2. **while** not convergent
3. Update x via (3.14)

$$x_{\mathscr{I}_i}=\Big(\rho_1^h\mathbf{1}_{k\times k}+\rho_2^h\mathbf{Diag}\big((y_{\mathscr{I}_i}-\mathbf{1})\circ(y_{\mathscr{I}_i}-\mathbf{1})\big)+\rho_3^h I_k\Big) \backslash (b_x)_{\mathscr{I}_i},\forall i=1,\ldots,n$$

4. Update y via solving (3.15)

$$\begin{aligned} b_y &= Cx^{h+1}+\lambda^h\circ x^{h+1}-\mu^h-\rho_2^h x^{h+1}\circ x^{h+1} \\ &\quad -\rho_3^h x^{h+1} \\ H_y &= \rho_2^h x^{h+1}\circ x^{h+1}+\rho_3^h\mathbf{1}_{nk\times 1} \\ y_{h+1} &= -b_y\oslash H_y \end{aligned} \tag{3.16}$$

5. Update ν,λ,μ via (3.16c)-(3.11e)

$$\begin{aligned} \nu^{h+1} &= \nu^h+\rho_1^h(Ex^{h+1}-\mathbf{1}) \\ \lambda^{h+1} &= \lambda^h+\rho_2^h((x^{h+1})^T Hy^{h+1}-e^T x^{h+1}) \\ \mu^{h+1} &= \mu^h+\rho_3^h(x^{h+1}-y^{h+1}). \end{aligned} \tag{3.17}$$

6. Update ρ_1,ρ_2,ρ_3

$$\rho_1^{h+1}=\alpha_1\rho_1^h,\rho_2^{h+1}=\alpha_2\rho_2^h,\rho_3^{h+1}=\alpha_3\rho_3^h$$

end

Again, the proposed formulation in (2.6) is equivalent to the one developed in [212], but more compact with fewer number of variables and constraints. The new formulation enables a computationally efficient algorithm (Algorithm 1) for the network module detection problem with closed form solutions for subproblems in the inexact ALM algorithm. Rather than solving the network module detection problem as a general mixed integer nonlinear programming in [212], our proposed formulation and algorithm aim to improve scalability for large-scale problems.

5.4 Simulation

In this section, we apply the proposed algorithm to multiple real-world databases with varying scales. For comparison, we apply the greedy algorithm from [21] to verify the improved computational efficiency of the proposed algorithm. All codes are written in MATLAB and all simulation is run on a standard desktop computer with a 3.60 GHz processor and a 32.0 RAM.

Both algorithms are applied the original network module detection problem formulated in (2.6). To test the robustness, for the small scale problems, we run both of the algorithms for three times. Note that random initials are generated for Algorithm 1. The stopping criterion is $\frac{\|x_k - y_k\|_F}{\|x_k\|_F} \leq 1 \times 10^{-3}$ and $\alpha_i \geq 1, i = 1,2,3$ are with small real values. As a result, the comparative results of two algorithms are provided in Table 5.1. For small scale problems, the greedy algorithm outperforms Algorithm 1 in efficiency. However, in larger scale ones, Algorithm 1 conversely demonstrates the significant improvement of computational efficiency. In Table 5.1, Q^* represents the best known modularity of the corresponding network [212], $Q_G, Q_{ALM}(\max), Q_{ALM}(\text{median})$ represent the modularity from greedy algorithm, the maximum and median of modularity from multiple trials of

Algorithm 1, respectively. Moreover, T_G and T_{ALM} represent the running time (in seconds) of greedy algorithm and Algorithm 1, respectively, and 'relative' is the relative difference of the modularity from two algorithms, defined as $\frac{Q_{ALM}(\max)-Q_G}{Q_G}$. Given the relative difference, it is also worth mentioning that our algorithm remains at comparable modularity value while improving the efficiency. Moreover, from the comparison of the maximum and median values from multiple trials of Algorithm 1, it verifies that Algorithm 1 is robust to random initials. Additionally, the computation time of our algorithm is less sensitive than that of the greedy algorithm, which makes it more promising for larger scale networks.

To better illustrate the comparison, Figure 5.1 demonstrates the time versus the number of network nodes for both algorithms. Moreover, we use the football database as an example to demonstrate the module detection result from Algorithm 1. Figure 5.2 shows that the coupling constraints converge to prime feasibility in a few iterations. Figure 5.3 provides the network after grouping the nodes into modules. It is obvious that the inter-module connectivity is more sparse than the intra-module connectivity. Similarly, such information can be validated in Fig. 5.4, which shows the permuted adjacency matrix. The nearly-block diagonal matrix proves the effectiveness of Algorithm 1 for solving network module detection problems from another perspective.

Table 5.1: Comparison between the Algorithm 1 and the greedy algorithm.

Network	node	edge	Q^*	Q_G	Q_{ALM}(max)
karate	34	78	0.420	0.4188	0.4118
dolphins	62	159	0.529	0.5188	0.5074
lesmis	77	254	0.560	0.5556	0.5249
polbooks	105	441	-	0.4986	0.4903
adjnoun	112	425	-	0.2906	0.2495
football	115	613	-	0.6046	0.5979
email	1,133	10,903	-	0.5406	0.5159
power	4,941	6,594	-	0.7490	0.7491
hep-th	8,361	15,751	-	0.8113	0.6531
astro-ph	16,706	121,251	-	0.6951	0.6114
cond-mat	16,726	47,594	-	0.8076	0.6201

	Q_{ALM}(median)	T_G/s	T_{ALM}/s	relative
karate	0.4003	0.0045	0.0340	-0.0
dolphins	0.4902	0.0076	0.0398	-0.0220
lesmis	0.5241	0.0078	0.0516	-0.0551
polbooks	0.4651	0.0077	0.0768	-0.0167
adjnoun	0.2257	0.0149	0.1278	-0.1416
football	0.5921	0.0160	0.0722	-0.0110
email	0.5016	0.4626	1.3594	-0.0457
power	0.7447	100.2569	11.5610	0.00
hep-th	0.6508	395.4231	41.1734	-0.195
astro-ph	0.6114	583.7437	111.3529	-0.1204
cond-mat	0.6201	1176.4	111.2393	-0.2321

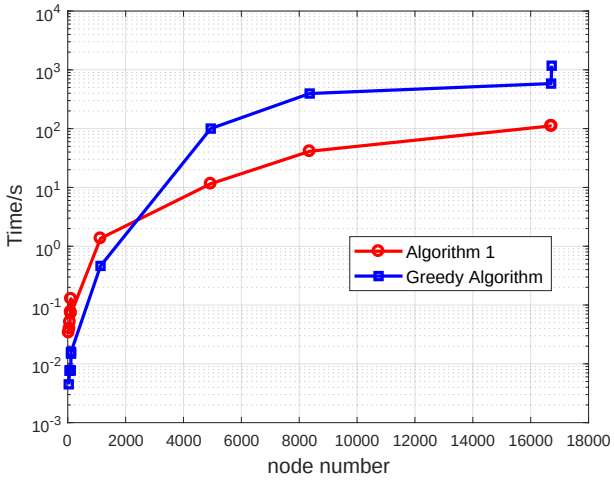


Figure 5.1: Time comparison between Algorithm 1 and the greedy method.

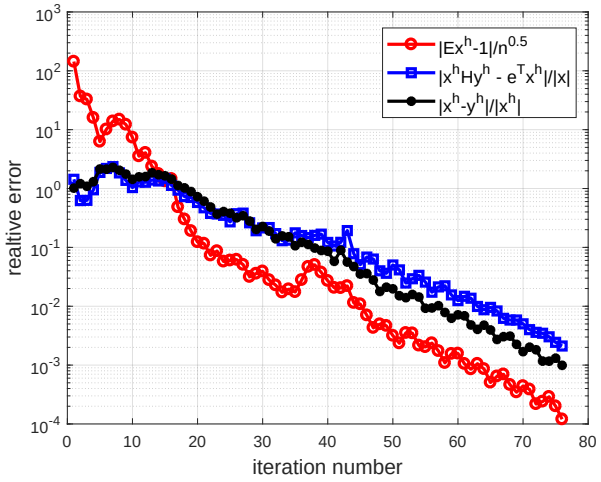


Figure 5.2: Convergence of the relative prime feasibility of the three constraints.

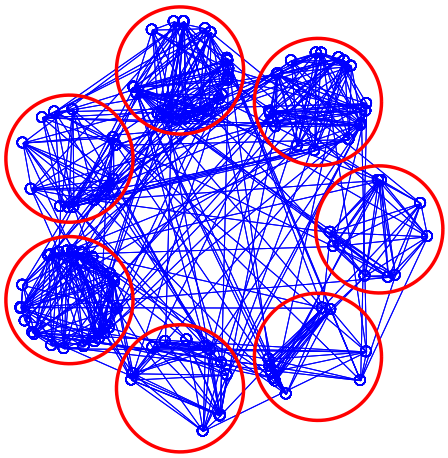


Figure 5.3: Module structure representation of the network from the module detection result. Each red circles contains nodes in the same module.

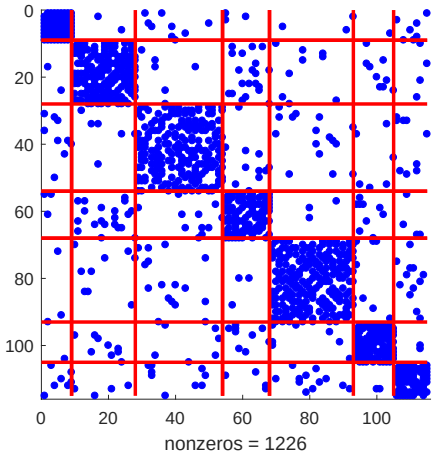


Figure 5.4: The (permuted) nearly-block diagonal adjacency matrix of the network. Each block corresponds to one module.

5.5 Conclusions

This paper aims to solve the module detection problem for large-scale complex networks to maximize modularity. We first formulate the network module detection problem as a Mixed Integer Quadratic Programming (MIQP) in a compact form. An time and memory efficient algorithm based on inexact Augmented Lagrangian Method (ALM) is developed to MIQPs. As the subproblems generated from the inexact ALM are of simple forms, each of the them leads to a closed form solution. Additionally, the special structure and sparsity of the problem formulation are utilized to to simplify the closed form solutions. Comparative results from the proposed algorithm and the greedy algorithm on solving real-world database examples are presented. Simulation results validate the efficiency of the proposed algorithm compared to the greedy algorithm for medium to large-scale problems while maintaining desired network modularity values.

5.6 Acknowledgements

The authors would like to thank Mark Newman for all of the databases we use here (`http://www-personal.umich.edu/~mejn/netdata/`). Also, we want to thank Antoine Scherrer for the MATLAB codes implementing greedy algorithm for comparison in this paper (`https://perso.uclouvain.be/vincent.blondel/research/louvain.html`).

Chapter 6: Weighted Network Design with Cardinality Constraints via Alternating Direction Method of Multipliers

6.1 Introduction

Network structure is essential in many classes of multiagent systems, where interaction among agents is determined by the underlying network. For example, successful control of a leader-follower network toward assigned states requires identifying leader nodes, leaders and followers subnetworks connectivity, and interactions among the followers [39]. In fact, designing the network topology is a prerequisite for many applications, such as spacecraft formations, mobile robot rendezvous, unmanned aerial vehicle flocking, among many others [9, 16, 159]. Related works in the area of network design have been pursued predominately for allocating edge weights. For geometric networks, efforts have been focused on determining the location of mobile nodes to establish connectivity [45] or maintain a connected network [215] while optimizing the system performance index, such as minimum control efforts for optimal trajectory generation. On the other hand, for non-geometric networks, extensive research has focused on the optimal edge weight design for networks with pre-specified topologies. For example, the work reported in [210] aims to design edge weights to obtain fast convergence of averaging algorithms. This problem is cast as a Semidefinite Programming (SDP) problem and two customized algorithms are

proposed that improve computational efficiency when compared to the commonly used heuristic methods. The work reported in [174] presents a procedure for designing the nodes and edge weights of a given network to satisfy the Laplacian eigenvalue constraints such as minimizing the gap between the largest and smallest eigenvalues. Other examples include allocating the (symmetric) graph edge weights to minimize the steady-state mean-square deviation [211] and assigning edge weights for a graph with a given topology [75]. Meanwhile, topology design of an unweighted network, i.e., designing binary edge weights, can be found in [46]. In general, the optimal network design problems presented in the existing literature focus on either allocating edge weights for the network with a given topology or topology design for unweighted networks.

In applications such as ad-hoc communication networks, the cost of constructing a connected network is proportional to the number of links. A network with more communication links, in general, implies higher implementation and operational costs. Thus, considering cardinality constraint on the edge set has economic impacts and benefits. Due to the cardinality constraint, the network topology design problem can be cast as an optimal resource allocation problem, where edges are treated as the limited resource and the budget corresponds to the total edge weights. The work presented in this paper focuses on the combinatorial problem of determining the edge weights and the network topology simultaneously.

In particular, the objective of this work is to develop a computationally efficient algorithm for simultaneous topology design and the corresponding edge weights for non-geometric undirected networks while satisfying a cardinality constraint on the edge set. Such problems are referred as Cardinality-Constrained Optimization Problems (CCOPs) that have extensive applications in system control and sensing [74]. For an unweighted network with boolean

variables representing the edge set, cardinality of the edge set is simply the total number of edges. By allowing allocating the limited edges, designing both topology and edge weights offers more flexibility to optimize the desired performance index than only allowing the assignment of edge weights for a network with a given topology [210, 211], or designing the topology of an unweighted graph [46]. Therefore, the simultaneous design of the network weight and topology represents a more general class of network design problems and holds the promise of generating a desired network with a better performance. For example in the consensus protocol, the algebraic connectivity of the graph, representing the convergence rate to the agreement set [143], can further be enhanced by the simultaneous allocation of the edge set and the corresponding weights. Meanwhile, the cardinality constraint for network synthesis can be applied in designing a sparse network, requiring less control efforts and/or lower developing cost in realistic applications [201]. Specifically, two problems are of a particular interest due to their wide range of applications: maximizing the algebraic connectivity of the network and minimizing the effective resistance of the graph [75].

In order to address cardinality constraints, a surrogate model, i.e., l_1 norm, is typically employed to approximately represent the cardinality function [35,78,86,214]. In general, this so-called l_1 regularization is able to promote sparsity in the feasible solution [37,60,120,147]. However, when the cardinality function appears in the constraint (as in CCOPs), such surrogate models are no longer applicable, as it becomes ambiguous to find appropriate bounds for the surrogate functions. In particular, an approximate model cannot guarantee satisfying the cardinality constraint. Other methods, such as heuristic search [208, 209] and branch-and-bound [19, 46], have been applied to small-scale network topology design problems with edge constraints; both procedures prove to be computational costly.

In the context of the existing literature on CCOPs, an efficient and effective algorithm that is applicable to large-scale CCOPs with guaranteed global convergence is highly desirable. Recently, Alternating Direction Method of Multipliers (ADMM) has been widely used for large-scale optimization problems, in areas such as machine learning, computer vision, and signal processing [173, 214]. A comprehensive survey of ADMM and its applications have been discussed in [27]. Due to the simple implementation and favorable computational performance in solving a wide range of convex optimization problems, extensive research on its convergence has been done. For example, when applying ADMM in the separable two-block convex problems, it has guaranteed convergence under certain mild assumptions [17,96]. When extending ADMM to solve a convex problem with multiple separable blocks, the work in [121] proves its global linear convergence. ADMM has also been applied to solving CCOPs [27] (Chapter 9.1). However, finding the solution for each iterative update in the ADMM can be time consuming in this case, especially when these updates involve solving Linear Matrix Inequalities (LMIs). In this work, in order to improve computational efficiency of ADMM in the context of CCOPs, we utilize the special structure of the weighted network design problem and search for closed-form solutions for iterative steps in the ADMM update.

Analysis on the convergence of ADMM for solving nonconvex problems has not been throughly investigated. Work in [97] analyzes the convergence of ADMM for solving consensus and coordination problems with a sufficiently large penalty parameter in the augmented Lagrangian function. However, the nonconvexity only lies in the objective function without local constraints. Other researchers have investigated the convergence property of ADMM for some special problem instances [85, 130]. In this paper, convergence of ADMM for solving a rather specific nonconvex optimization problem with a nonconvex

constraint (induced by the cardinality constraint) is examined. A customized ADMM algorithm is then developed and implemented for the corresponding weighted network design problems. In order to illustrate the improved scalability and robustness of the proposed algorithm, problem instances at different scales and random initializations of the algorithm are examined. Moreover, simulation results are compared to existing methods that are generally applicable to small-scale network design problems. The main contribution of the paper is the development of a computationally efficient algorithm for solving cardinality-constrained network design problems with guaranteed convergence to a local optimum.

The rest of the paper is organized as follows. §II introduces two classes of weighted network design problems: maximizing the algebraic connectivity of graphs and minimizing the effective resistance in electrical networks. The ADMM framework and its extension to solving the specific network design problems are then discussed in §III; this is then followed by the convergence proof of the algorithm in §IV. Simulation results are presented in §V. We conclude the paper with a few remarks in §VI.

6.1.1 Preliminaries

The notation used throughout this paper is introduced in this subsection. The set of $n \times n$ symmetric matrices is denoted by $\mathbb{S}^n$ and the set of $n \times n$ positive semidefinite (definite) matrices is denoted by $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$). The notation $X \succeq \mathbf{0}$ ($X \succ \mathbf{0}$) designates that the matrix $X \in \mathbb{S}^n$ is positive semidefinite (definite). The Frobenius norm of X is denoted by $\|X\|_F$ and the cardinality of vector $\mathbf{x}$ is denoted by $\mathbf{Card}(\mathbf{x})$. The operator $\mathrm{Proj}_{\mathscr{C}}$ represents the projection of its argument onto the set $\mathscr{C}$. For two matrices X_1 and X_2 with appropriate dimensions, $\langle X_1, X_2 \rangle$ denotes their inner product; $\mathbf{diag}(X)$ represents a vector whose components are the diagonal elements of matrix X while $\mathbf{Diag}(\mathbf{x})$ represents

a diagonal matrix whose diagonal components are the elements of the vector $\mathbf{x}$. Similarly, $\mathbf{blkdiag}(X_1,\ldots,X_n)$ represents a block-diagonal matrix with $X_1,\ldots,X_n$ on the diagonal. The notation $\mathrm{vec}(X)$ represents the vectorization of a matrix X by stacking its columns on one another from left to right. The spectra of the square matrix X is represented by $\mathrm{eig}(X)$.

6.2 Problem Formulation

The number of edges in a network is often used as a criterion for evaluating the cost of constructing a connected network. When given a limited cost for this construction expressed by the cardinality constraint on the edge set, a wide range of network topology design problems can be formulated as a CCOP. Specifically, two representative examples of interest are presented in this section. Consider an undirected, weighted network $\mathscr{G} = (V,E)$ with vertex set $V = \{1,2,\ldots,n\}$ and edge set E consisting of two element subsets of V. The Laplacian matrix of $\mathscr{G}$ is defined as $\mathscr{L}(\mathscr{G}) = \mathscr{A}\,\mathbf{diag}(\mathbf{g})\mathscr{A}^T$, where $\mathscr{A} \in \mathbb{R}^{n\times m}$ is the incidence matrix, $\mathbf{g} \in \mathbb{R}^m$ is the vector for weights of the edge set, and $m = n(n-1)/2$. Note that for $v_i, v_j \in V$, if $\{v_i, v_j\} \notin E$, the corresponding edge weight is zero.

6.2.1 Network Design to Maximize Algebraic Connectivity

For a multi-agent system, the consensus protocol on the network is represented as

$$\dot{\mathbf{x}} = -\mathscr{L}(\mathscr{G})\mathbf{x}, \tag{2.1}$$

deriving the states of each agent to consensus, i.e., to the set $\mathscr{C} = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{x}_i = \mathbf{x}_j, \forall\, v_i, v_j \in V\}$; this is accomplished by exchanging state information with agents in the specified (connected) network $\mathscr{G}$. In (2.1), $\mathscr{L}(\mathscr{G})$ is the positive semi-definite graph Laplacian. As such, one can sort its eigenvalues in the non-decreasing order as $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. It is known that for a connected network $\mathscr{G}$, the (undirected) consensus protocol converges to

consensus with a rate of convergence that is dictated by the algebraic connectivity of the graph, namely, $\lambda_2(\mathscr{G})$ [143].

With $\lambda_2(\mathscr{L})$ chosen as the objective to be maximized, our goal is to design the weighted topology of a connected network subject to cardinality constraint on the edge set. Mathematically, this type of network topology design problem is formulated as

$$\begin{aligned} \max_{\mathbf{g}} \quad & \lambda_2(\mathscr{L}) \\ s.t. \quad & \mathbf{g} \geq 0, \lambda_2(\mathscr{L}) > 0 \\ & \mathbf{1}^T \mathbf{g} = g_t \\ & \mathbf{Card}(\mathbf{g}) \leq r, \end{aligned} \tag{2.2}$$

where g_t is the overall weight of the edge set, r is the upper bound of cardinality of $\mathbf{g}$. Note that λ_2 in (2.2) is not an explicit function of the design variable $\mathbf{g}$. In addition, computing the second smallest eigenvalue of a to-be-determined matrix is non-trivial. A similarity transform is introduced below to transform the network connectivity constraint into an LMI constraint.

Lemma 6.2.1. *For a graph Laplacian $\mathscr{L}(\mathscr{G})$, $\alpha > \lambda_2$ if and only if $\mathscr{L}(\mathscr{G}) + \alpha \mathbf{1}\mathbf{1}^T/n \succeq \lambda_2(\mathscr{L})\mathbf{I}$, where $\mathbf{I} \in \mathbb{R}^{n \times m}$ is the identity matrix.*

Lemma 6.2.1 has been used in the literature, including our previous work [46]. Based on the LMI representation of the connectivity constraint, the original weighted network design problem to maximize algebraic connectivity is reformulated as

$$\begin{aligned} \min_{\mathbf{g}, \lambda_2} \quad & -\lambda_2 \\ s.t. \quad & \mathbf{g} \geq 0, \lambda_2 > 0 \\ & \mathbf{1}^T \mathbf{g} = g_t \end{aligned} \tag{2.3}$$

$$\mathbf{Card}(\mathbf{g}) \leq r$$

$$\mathscr{L}(\mathscr{G}) + \alpha \mathbf{1}\mathbf{1}^T/n \succeq \lambda_2 \mathbf{I},$$

where λ_2 is handled as an unknown variable to be determined together with $\mathbf{g}$; in this formulation α is selected as a large scalar. As the objective is to maximize λ_2 and the term $\mathscr{L}(\mathscr{G}) + \alpha \mathbf{1}\mathbf{1}^{\mathbf{T}}/n$ acts as an upper bound on λ_2, formulation in (2.3) is equivalent to the original problem (2.2). The objective function and constraints, except for the cardinality constraint, are convex. Thus problem in (2.3) is classified as a CCOP.

6.2.2 Network Design to Minimize Total Effective Resistance

In an electrical network, effective resistance between a pair of nodes in a connected weighted graph is the voltage difference between these nodes to induce a one Amp current flow between them, when the weights are viewed as conductances. In the electrical circuit analysis, the total effective resistance is defined as the sum of the effective resistance between all different pairs of nodes in the circuit. Effective resistance has often been used as a measure of responsiveness of a network to an external input, as well as a robustness measure due to node or edge removal in the network. In [75] it is shown that designing the edge weights of a network with a given topology (without concern for the cardinality of the edge set) is a convex optimization problem. In this paper, we consider designing both network topology and edge weights with an upper bound on $\mathbf{Card}(\mathbf{g})$. Based on the formulation of edge weight design problem for minimizing the total effective resistance in [75], the corresponding problem with cardinality constraint assumes the form,

$$\begin{aligned} \min_{\mathbf{g},Y} \quad & n\mathbf{trace}(Y) \\ s.t. \quad & \mathbf{1}^T\mathbf{g} = g_t, \mathbf{g} \geq 0 \end{aligned}$$

$$\begin{bmatrix} \mathscr{L} + \mathbf{11^T/n} & I \\ I & Y \end{bmatrix} \succeq \mathbf{0} \qquad (2.4)$$
$$\mathbf{Card}(\mathbf{g}) \leq r,$$

where $Y \in \mathbb{S}^n$ is a slack matrix and the other variables are defined similar to those in (2.2). It is clear that the weighted network topology design problem formulated in (2.4) is again a CCOP.

For the two classes of network design problems discussed above, the l_1-regularization to approximately represent the cardinality of the edge set can be inadequate. One reason for this is that the cardinality constraint appears as an inequality in both problems; as such an approximate representation does not guarantee satisfying the exact inequality constraint. In addition, there is an equality constraint on the l_1 norm of the variable set, i.e, $\mathbf{1}^T\mathbf{g} = g_t$. Therefore, approximating the cardinality function of the edge set via l_1-regularization does not apply to problems considered here. Moreover, the presence of the LMI constraints makes the optimization impractical for large-scale problems. For example, for an LMI with linear constraints in the order of $O(n^2)$, the computational time for an interior-point based method is in the order of $O(n^6)$ [24].

6.3 Algorithm: Framework and Procedures

In this section we examine a computational approach for CCOPs based on an ADMM framework.

6.3.1 ADMM for General CCOPs

Let us start with the general CCOP expressed in the form of

$$\min_{\mathbf{x}} \quad f(\mathbf{x})$$

$$\begin{aligned} s.t. \quad & \mathbf{x} \in \mathscr{C} \\ & |\mathbf{x}|_0 \leq r, \end{aligned} \tag{3.5}$$

where $\mathbf{x} \in \mathbb{R}^p$ is the unknown variable, $f(\mathbf{x})$ is a convex and differentiable function, $r \leq p \in \mathbb{N}$, $|\mathbf{x}|_0$ is the pseudo norm representing the number of nonzero elements in $\mathbf{x}$, and $\mathscr{C}$ is a convex and nonempty feasible set.

To decouple the constraints, the first step introduces an auxiliary variable vector $\mathbf{z}$ and reformulates the general CCOP in (3.5) as

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & f(\mathbf{x}) \\ s.t. \quad & \mathbf{x} \in \mathscr{C} \\ & |\mathbf{z}|_0 \leq r \\ & \mathbf{x} = \mathbf{z}. \end{aligned} \tag{3.6}$$

The augmented Lagrangian of (3.6) is then

$$\mathscr{L} = f(\mathbf{x}) + \langle \boldsymbol{\mu}, \mathbf{x} - \mathbf{z} \rangle + \frac{\rho}{2} \|\mathbf{x} - \mathbf{z}\|_2^2, \tag{3.7}$$

where $\boldsymbol{\mu} \in \mathbb{R}^p$ is the corresponding dual variable and ρ is the weighting factor associated with the penalty term. Under the ADMM framework, problem in (3.6) can be solved by alternating between two variables, $\mathbf{z}$ and $\mathbf{x}$, followed by updating the dual variable $\boldsymbol{\mu}$. More explicitly, the ADMM for (3.6) is summarized in Algorithm 1 below.

Algorithm 1: ADMM for solving (3.6)
Input: Problem data $f(\mathbf{x}), r, \rho$ and initial point $(\mathbf{x}^0, \boldsymbol{\mu}^0)$
Output: Local optimum $(\mathbf{x}, \mathbf{z})$ of (3.6)
begin

1. Set $k = 0$
2. **while** not convergent
3. Update $\mathbf{z}$ via

$$\mathbf{z}^{k+1} := \arg \min_{|\mathbf{z}|_0 \leq r} \|\mathbf{x}^k - \mathbf{z} + \boldsymbol{\mu}^k/\rho\|_F^2 \tag{3.8}$$

4. Update $\mathbf{x}$ via

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x} \in \mathscr{C}} f(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}^{k+1} + \frac{\boldsymbol{\mu}^k}{\rho}\|_F^2 \tag{3.9}$$

5. Update $\boldsymbol{\mu}$ via

$$\boldsymbol{\mu}^{k+1} := \boldsymbol{\mu}^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}) \tag{3.10}$$

6. $k = k + 1$
7. **end while**
8. Determine $\mathbf{x}, \mathbf{z}$

end

Remark: The work reported in [27] (Chapter 9.1) proposes an algorithm, based on ADMM, for solving problems with a convex objective and cardinality constraints. However, for more general CCOPs, the problem not only includes cardinality constraints but also other convex constraints, such as the convex constraint set $\mathscr{C}$ in (3.5). In addition, the algorithm presented here is slightly different from [27]; in particular, for reasons that will become apparent in the convergence proof, the order of updates for $\mathbf{x}$ and $\mathbf{z}$ has been interchanged.

6.3.2 ADMM Subproblems

For every step k in the ADMM of Algorithm 1, the update is composed of three subproblems, expressed in (3.8)-(3.10). For updating $\mathbf{z}$ in (3.8), one has $\mathbf{z}^{k+1} = \text{Proj}_{\{\mathbf{z}|\ |\mathbf{z}|_0 \leq r\}}(\mathbf{x}^k + \boldsymbol{\mu}^k/\rho)$. To satisfy the cardinality constraint $|\mathbf{z}|_0 \leq r$, $\mathbf{z}^{k+1}$ is obtained by keeping the r largest magnitude entries in $\mathbf{z}$ and setting the rest to zero [27].

The updating of $\mathbf{x}$ in (3.9) is to solve a strongly convex problem for its global optimum. For the case where $f(\mathbf{x})$ is linear with regard to $\mathbf{x}$, i.e., $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$, the updating of $\mathbf{x}$ in (3.9) within a random convex feasible region $\mathscr{C}$ is determined by

$$\mathbf{x}^{k+1} = \text{Proj}_{\mathscr{C}}(\mathbf{z}^{k+1} - (\boldsymbol{\mu}^k + \mathbf{c})/\rho), \tag{3.11}$$

which does not generally have a closed form solution. However, in some special cases, e.g, $\mathscr{C}$ defined by affine constraints, determining the analytical solution becomes feasible. For example, when $\mathscr{C} := \{\mathbf{x} \in \mathbb{R}^p | A\mathbf{x} = \mathbf{b}\}$ with $A \in \mathbb{R}^{q\times p}$ and $\mathbf{b} \in \mathbb{R}^q$, the update of $\mathbf{x}$ in (3.11) is a strongly convex quadratic programming problem with equality linear constraints. As a result, the updated $\mathbf{x}$ can be obtained by solving the following linear equations,

$$\begin{bmatrix} I_{p\times p} & A^T \\ A & \mathbf{0}_{q\times q} \end{bmatrix} \begin{bmatrix} \mathbf{x}^{k+1} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{z}^{k+1} - (\boldsymbol{\mu}^k + \mathbf{c})/\rho \\ \mathbf{b} \end{bmatrix}, \tag{3.12}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^q$ is the dual variable associated with the linear equality constraint. Since the above matrix $\begin{bmatrix} I_{p\times p} & A^T \\ A & \mathbf{0}_{q\times q} \end{bmatrix}$ is invertible, its inverse only needs to be computed once and then cached for the ADMM iterations and subsequent updates.

However, a closed form solution of (3.9) is not always available for general CCOPs, such as the weighted network design problems formulated in (2.3) and (2.4). To reduce the computation cost for updating $\mathbf{x}$, we propose an approximate updating method that transforms the constrained optimization problem into an unconstrained one. By assuming

that $\mathscr{C}$ in (3.5) can be rewritten in the form of a finite number of constraints, we define $\mathscr{C} = \{\mathbf{x} \in \mathbb{R}^p | \mathbf{h}(\mathbf{x}) \leq 0\}$, where $\mathbf{h}(\mathbf{x}) : \mathbb{R}^p \to \mathbb{R}^l$ are continuously differentiable functions. For simplicity, we omit the equality constraints here, as an equality constraint can be written as a pair of inequality constraints. To remove the constraints, we introduce a quadratic penalty function $P(\mathbf{x})$ defined as

$$P(\mathbf{x}) = \beta \sum_{j=1}^{l} \big(\max(\mathbf{h}_j(\mathbf{x}), 0)\big)^2,$$

where $\beta > 0$ is the penalty weighting factor which is selected as a sufficiently large number. By considering the constraints as penalty terms, the original problem in (3.5) is replaced by the following one,

$$\begin{aligned} \min_{x} \quad & f(\mathbf{x}) + P(\mathbf{x}) \\ s.t. \quad & |\mathbf{x}|_0 \leq r. \end{aligned}$$

Through the above conversion, the subproblem in (3.9) is converted into a convex unconstrained problem that can be solved more efficiently.

6.3.3 A Customized ADMM Algorithm

For the network design problems stated in Section II, the constraints in (2.3) and (2.4) can be generalized in the form of $\mathscr{C} = \{\mathbf{x} | A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0, \mathscr{B}(\mathbf{x}) + B_0 \succeq \mathbf{0}\}$. This uniform representation can be obtained by adopting a new notation. For problem (2.3), by setting $\mathbf{x} = [\mathbf{g}^T, \lambda_2]^T$, the components in the LMI constraint $\mathscr{L}(\mathscr{G}) + \alpha \mathbf{1}\mathbf{1}^T/n \succeq \lambda_2 \mathbf{I}$ can be represented as $\mathscr{B}(\mathbf{x}) = \mathscr{L}(\mathscr{G}) - \lambda_2 \mathbf{I}$ and $B_0 = \alpha \mathbf{1}\mathbf{1}^T/n$. Similarly in problem (2.4), by setting $\mathbf{x} = [\mathbf{g}^T, (\mathrm{vec}(Y))^T]^T$, its components in the LMI constraint

$$\begin{bmatrix} \mathscr{L} + \mathbf{1}\mathbf{1}^{\mathbf{T}}/\mathbf{n} & I \\ I & Y \end{bmatrix} \succeq \mathbf{0}$$

can be represented as $\mathscr{B}(\mathbf{x}) = \mathbf{blkdiag}(\mathscr{L}(\mathscr{G}), Y)$ and

$$B_0 = \begin{bmatrix} \mathbf{1}\mathbf{1}^T/n & I \\ I & \mathbf{0} \end{bmatrix}.$$

The linear vector constraints for both problems can be defined accordingly. Thus, a uniform representation for both network design problems can then be obtained as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\ s.t. \quad & A\mathbf{x} = \mathbf{b}, \mathscr{B}(\mathbf{x}) + B_0 \succeq \mathbf{0} \\ & \mathbf{x} \geq 0, |\mathbf{x}|_0 \leq r, \end{aligned} \tag{3.13}$$

where $\mathscr{B} : \mathbb{R}^p \to \mathbb{S}^q$ is a linear operator and $B_0 \in \mathbb{S}^q$. Note that the nonnegativity and cardinality constraints are not imposed on all variables involved. For example, in (2.3), λ_2 is not involved in the cardinality constraint while in (2.4), Y is involved in neither type of constraints. However, this will not effect the proposed algorithms, which will be addressed subsequently.

For problem (3.13), Algorithm 1 can be applied directly without modifications. However, due to the complexity of the set $\mathscr{C}$ in (3.13), a closed-form solution for the subproblem (3.9) is non-trivial. Thus, this step requires employing a convex optimization solver, such as the interior point method [91] to find the optimal $\mathbf{x}$ in each iteration of Algorithm 1. In order to improve the computational efficiency when iteratively solving subproblems in Algorithm 1, as applied to the network design problems, a customized ADMM is now proposed.

First, new auxiliary variables S and S_1 are introduced to reformulate problem (3.13) as

$$\begin{aligned} \min_{\mathbf{z},\mathbf{x},S,S_1} \quad & \mathbf{c}^T\mathbf{x} \\ s.t. \quad & A\mathbf{x} = \mathbf{b}, \mathscr{B}(\mathbf{x}) + B_0 + S = \mathbf{0} \\ & \mathbf{z} \geq 0, |\mathbf{z}|_0 \leq r \end{aligned} \tag{3.14}$$

$$\mathbf{x} = \mathbf{z}, S = S_1, S_1 \preceq \mathbf{0}.$$

The corresponding augmented Lagrangian function is now constructed as

$$\begin{aligned}\mathscr{L}_1 = \mathbf{c}^T\mathbf{x} + \langle \boldsymbol{\mu}, \mathbf{x} - \mathbf{z}\rangle + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}\|_F^2 \\ + \langle \Lambda, S - S_1\rangle + \frac{\rho}{2}\|S - S_1\|_F^2. \end{aligned} \tag{3.15}$$

When applying ADMM to problem (3.14), the iterative procedure will alternate between solving for $\mathscr{X} = (\mathbf{z}, S_1)$ and $\mathscr{Y} = (\mathbf{x}, S)$ followed by updating the dual variables $D = (\boldsymbol{\mu}, \Lambda)$. Based on the augmented Lagrangian in (3.21), each subproblem for updating the corresponding variables is expressed as

$$\begin{aligned}\mathscr{X}^{k+1} &:= \arg\min_{\mathscr{X}} \mathscr{L}_1(\mathscr{X}, \mathscr{Y}^k; D^k) \\ &= \arg\min_{\mathbf{z}} \mathscr{L}_1(\mathscr{X}, \mathscr{Y}^k; D^k) \\ &\quad + \arg\min_{S_1} \mathscr{L}_1(\mathscr{X}, \mathscr{Y}^k; D^k) \end{aligned} \tag{3.16a}$$

$$\mathscr{Y}^{k+1} := \arg\min_{\mathbf{x}, S} \mathscr{L}_1(\mathscr{X}^{k+1}, \mathscr{Y}; D^k) \tag{3.16b}$$

$$\Lambda^{k+1} := \Lambda^k + \rho(S^{k+1} - S_1^{k+1}) \tag{3.16c}$$

$$\boldsymbol{\mu}^{k+1} := \boldsymbol{\mu}^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}) \tag{3.16d}$$

The above updates omit the constraints on $\mathscr{X}$ and $\mathscr{Y}$ in the formulation for simplicity. In the following, we aim to develop the explicit closed-form solutions for all subproblems in (3.16).

For (3.16a), searching for S_1 is projection onto negative semidefinite matrices. The optimum of S_1 in (3.16a) can be obtained via the following proposition [131].

Proposition 6.3.1. *The optimal solution to minimize the following quadratic function*

$$S_1^* = \arg\min_{S_1 \in \mathbb{S}^n} \|S_1 - S_0\|_F^2, \; s.t. \;\; S_1 \preceq \boldsymbol{0},$$

can be obtained via

$$S_1^* = U \bullet \mathbf{Diag}(\min(\mathbf{diag}(\Sigma), 0)) \bullet U^T,$$

where U and Σ are the matrix of unit eigenvectors and eigenvalues of S_0, respectively. Specifically, in the S_1-optimization of (3.16a) at step k, $S_0 = S^k + \Lambda^k/\rho$.

As for the $\mathbf{z}-$update, recall the clarification provided after (3.13), underscoring that the nonnegativity and cardinality constraints are not imposed on all variables involved. Since the $\mathbf{z}-$update is a projection problem, taking (2.3) for example, variables $\mathbf{g}$ involved in both constraints are uncoupled with λ_2 that is only involved in the nonnegativity constraint. Similarly in (2.4), the variable $\mathbf{g}$ is decoupled from Y. Consequently, the projection of these variables is independent and projection on cardinality or nonnegativity constraint individually is straightforward. However, the challenge in updating $\mathbf{z}$ in (3.16a) is the projection onto the intersection of the two sets. One projection is onto the nonnegative reals ($\Omega_1 = \{\mathbf{z} \,|\, \mathbf{z} \geq 0\}$) and the other is with respect to the cardinality constraint $\Omega_2 = \{\mathbf{z} \,|\, |\mathbf{z}|_0 \leq r\}$. Mathematically, the $\mathbf{z}-$update involves solving the optimization problem,

$$\begin{aligned} \min_{\mathbf{z}} \quad & \|\mathbf{x}^k - \mathbf{z} + \boldsymbol{\mu}^k/\rho\|_F^2 \\ s.t. \quad & \mathbf{z} \geq 0, |\mathbf{z}|_0 \leq r. \end{aligned} \tag{3.17}$$

The projection of a vector $\mathbf{y} \in \mathbb{R}^p$ onto each set is determined by

$$\begin{aligned} \text{Proj}_{\Omega_1}(\mathbf{y}) &= \max(\mathbf{y}, 0) \\ (\text{Proj}_{\Omega_2}(\mathbf{y}))_i &= \begin{cases} \mathbf{y}_i, i \in \mathscr{I} \\ 0, \text{otherwise} \end{cases}, \end{aligned} \tag{3.18}$$

where $\mathscr{I}$ is the index set of the r largest magnitude entries of $\mathbf{y}$. Based on the closed-form solution of the projection on either Ω_1 or Ω_2, the projection on $\Omega_1 \cap \Omega_2$ is obtained via

$$\mathbf{z}^{k+1} = \text{Proj}_{\Omega_2}(\text{Proj}_{\Omega_1}(\mathbf{z}^k + \boldsymbol{\mu}^k/\rho)). \tag{3.19}$$

Solution of the $\mathbf{z}-$update derived above is a special case of the alternating projection method that finds the projection of a point at the intersection of two sets. The alternating projection step (3.19) however converges after one iteration. Furthermore, the proposition stated below shows that $\mathbf{z}^{k+1}$ obtained from (3.19) finds the global optimum of the problem (3.17) with a concave Ω_2.

Proposition 6.3.2. *Given* $\mathbf{d} \in \mathbb{R}^p$, $Proj_{\Omega_2}(Proj_{\Omega_1}(\mathbf{d}))$ *characterizes the global optimum of* $\min_{\mathbf{z}\in\Omega_1\cap\Omega_2} \|\mathbf{z}-\mathbf{d}\|_F^2$.

Proof. It suffices to consider $\mathbf{d}_i \neq 0$ for all $1 \leq i \leq p$. With $\mathbf{z}$ being the global optimum, define $\mathscr{S} = \{i \,|\, \mathbf{z}_i > 0\}$ and $\mathscr{S}^{\complement} = \{i \,|\, \mathbf{z}_i = 0\}$. Then for all $1 \leq i \leq p$, only one of the two conditions is true: $i \in \mathscr{S}$ and $i \in \mathscr{S}^{\complement}$. Accordingly, the objective can be rewritten as $\sum_{i\in\mathscr{S}^{\complement}} |\mathbf{d}_i|^2 + \sum_{i\in\mathscr{S}} |\mathbf{z}_i - \mathbf{d}_i|^2$. For a possible choice of $\mathscr{S}$, there are two rules ordered by priority:

i) $\forall\, \mathbf{d}_i < 0$, one has $\mathbf{z}_i = 0$, i.e., $i \in \mathscr{S}^{\complement}$. This is because if $\mathbf{d}_i < 0$ and $\mathbf{z}_i > 0$, we can always have a $\tilde{\mathbf{z}}$ such that $\tilde{\mathbf{z}}_i = 0$ and $\tilde{\mathbf{z}}_{p/i} = \mathbf{z}_{p/i}$ to make the objective smaller without violating the constraints.

ii) With i), the problem degenerates to the projection of $\tilde{\mathbf{d}}$ onto Ω_2 where $\tilde{\mathbf{d}} = \text{Proj}_{\Omega_1}(\mathbf{d})$.

Combing i) and ii) leads to $\mathbf{z} = \text{Proj}_{\Omega_2}(\tilde{\mathbf{d}}) = \text{Proj}_{\Omega_2}(\text{Proj}_{\Omega_1}(\mathbf{d}))$, concluding the proof. □

Moreover, it is worth mentioning that the order of projection onto the two sets in (3.19) does make a difference on the result, which is explained by the priority order of i) and ii) in the above proof.

For (3.16b), the $(\mathbf{x}, S)-$optimization is the projection onto the affine equality constraints set, which can follow the same scheme expressed in (3.12) to solve a linear equation

formulated as

$$\begin{bmatrix} I & \tilde{A}^T \\ \tilde{A} & \mathbf{0}_{q\times q} \end{bmatrix} \begin{bmatrix} \mathbf{x}_S^{k+1} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{z}^{k+1} - (\boldsymbol{\mu}^k + \mathbf{c})/\rho \\ \text{vec}(S_1^{k+1} - \Lambda^k/\rho) \\ \mathbf{b} \\ \text{vec}(-B_0) \end{bmatrix}, \tag{3.20}$$

where $\mathbf{x}_S = [\mathbf{x}^T, \text{vec}(S)^T]^T \in \mathbb{R}^{p+q^2}$, $\tilde{A} = \begin{bmatrix} A & \mathbf{0} \\ \tilde{B} & I_{q^2} \end{bmatrix}$ with $\tilde{B}\mathbf{x} = \text{vec}(\mathscr{B}(\mathbf{x}))$.

However, for large scale problems, solving the linear problem in (3.20) requires extensive memory allocation and computational time. As a result, we aim to exploit the special structure and sparsity of the weighted network design problems in (2.3) and (2.4) to reduce the corresponding computational effort. For illustration purposes, the weighted network design problem in (2.3) to maximize the algebraic connectivity is taken as a problem instance here. Thus the unknown variables include $\mathbf{x} = [\mathbf{g}^T, \lambda_2]^T \in \mathbb{R}^{m+1}$ and $S \in \mathbb{S}^n$. The dimensions of the other involved matrix variables are $\mathscr{B}(\mathbf{x}) \in \mathbb{S}^n$ and $B_0 \in \mathbb{S}^n$.

First, since all matrices, $\mathscr{B}(\mathbf{x})$, S, and B_0, are symmetric, we only need to consider their upper triangular entries when formulating the related constraints. Second, as $\mathscr{B}(\mathbf{x})$ comes from the Laplacian matrix in (2.3), the upper off-diagonal entries in S can be represented by finding S_{ij} from $-\mathbf{g}_k + (B_0)_{ij} + S_{ij} = 0$ for $i \neq j$, where k is the index of $\mathbf{g}$ for the corresponding elements in $\mathscr{L}_{ij}$. These two facts reduce the number of constraints in $\mathscr{B}(x) + B_0 + S = \mathbf{0}$ as well as the number of unknown entries in S (from $m = n(n-1)/2$ to n). As a result, (3.16b) can be simplified in the following form

$$\begin{aligned} \min_{\mathbf{x},s} \quad & \tfrac{1}{2}\|\mathbf{x} + \mathbf{c}_1\|_F^2 + \|\mathbf{g} + \mathbf{c}_2\|_F^2 + \tfrac{1}{2}\|\mathbf{s} + \mathbf{c}_3\|_F^2 \\ s.t. \quad & A\mathbf{x} = \mathbf{b} \\ & B\mathbf{x} + \mathbf{s} = -\mathbf{b}_0, \end{aligned} \tag{3.21}$$

where $\mathbf{c}_1 = -\big(\mathbf{z}^{k+1} - (\boldsymbol{\mu}^k + \mathbf{c})/\rho\big)$, $\mathbf{c}_2 = -(B_0 + S_1^{k+1} - \Lambda^k/\rho)_{\mathscr{T}}$ with $\mathscr{T}$ representing the indices of off-diagonal entries in a square matrix, and $\mathbf{c}_3 = -\mathbf{diag}(S_1^{k+1} - \Lambda^k/\rho)$. Moreover,

$B\mathbf{x} = \mathbf{diag}(\mathscr{B}(\mathbf{x}))$, $\mathbf{s} = \mathbf{diag}(S)$ and $\mathbf{b}_0 = \mathbf{diag}(B_0)$. Note that the coefficient in front of $\|\mathbf{g}+\mathbf{c}_2\|_F^2$ is 1 instead of $\frac{1}{2}$ due to the fact that S is symmetric and the lower triangular entries are identical to the upper triangular ones. The optimum of (3.21) can be obtained via solving the first order optimality conditions,

$$\begin{aligned} &Q_x\mathbf{x} + A^T\boldsymbol{\nu}_1 + B^T\boldsymbol{\nu}_1 = -\mathbf{c}_x \\ &Q_s\mathbf{s} + \boldsymbol{\nu}_2 = -\mathbf{c}_s \\ &A\mathbf{x} = \mathbf{b} \\ &B\mathbf{x} + \mathbf{s} = -\mathbf{b}_0, \end{aligned} \tag{3.22}$$

where Q_x ($\mathbf{c}_x$) and Q_s ($\mathbf{c}_s$) are the Hessian (coefficient vectors of the first order term) in the objective function with respect to $\mathbf{x}$ and $\mathbf{s}$, respectively, and $\boldsymbol{\nu}_1$ and $\boldsymbol{\nu}_2$ are the associated Lagrangian multipliers.

By eliminating $\mathbf{x}$ and $\mathbf{s}$ in (3.22), these equations simplify as

$$\begin{aligned} &\mathbf{b} + AQ_x^{-1}A^T\boldsymbol{\nu}_1 + AQ_x^{-1}B^T\boldsymbol{\nu}_2 = -AQ_x^{-1}\mathbf{c}_x \\ &\mathbf{b}_0 + BQ_x^{-1}A^T\boldsymbol{\nu}_1 + BQ_x^{-1}B^T\boldsymbol{\nu}_2 + Q_s^{-1}\boldsymbol{\nu}_2 \\ &= -BQ_x^{-1}\mathbf{c}_x - Q_s^{-1}\mathbf{c}_s, \end{aligned} \tag{3.23}$$

where the Hessian matrices, Q_x and Q_s, are diagonal matrices with positive diagonal elements. Consequently, the first order optimality conditions can be efficiently solved instead of computing inversion of the large-scale block matrix in (3.20). In addition, since $B \in \mathbb{R}^{n\times(m+1)}$ is a large-scale sparse matrix with the ratio of nonzero entries in the order of $O(1/n)$), only nonzero elements are stored to save memory allocation. Furthermore, the sparse matrix multiplication technique to reduce computational cost is applied when solving (3.22). Figure 6.1 provides an example of the sparsity pattern when $n = 20$. In fact, the

computation in (3.22) can be performed and cached before the ADMM algorithm is initiated to avoid duplicate computations in the ADMM iterations.

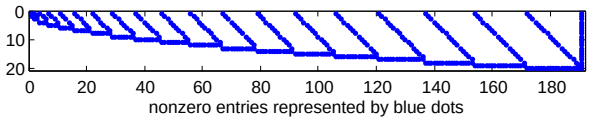


Figure 6.1: Sparsity pattern of B when $n = 20$.

So far, we have found closed-form solutions for all of the subproblems in (3.16) with reduced number of variables and constraints in order to improve computational efficiency. A summary of the customized ADMM algorithm for the specific weighted network design problems is outlined as follows.

Algorithm 2: A Customized ADMM for solving (3.14)
Input: Problem information $f(\mathbf{x}), r, \rho$ and initial point $(\mathbf{x}^0, S^0, \boldsymbol{\mu}^0, \boldsymbol{\Lambda}_0)$
Output: Local minimizer $(\mathbf{x}, \mathbf{z}, S, S_1)$ to (3.14)
begin

1. Perform preliminary calculations in solving the first order conditions in (3.22)
2. Set $k = 0$
3. **while** not convergent
4. Update $\mathscr{X} = (\mathbf{z}, S_1)$ via

$$\begin{aligned} \mathbf{z}^{k+1} &= \mathrm{Proj}_{\Omega_2}(\mathrm{Proj}_{\Omega_1}(\mathbf{x}^k + \boldsymbol{\mu}^k/\rho)) \\ [U, \Sigma] &= \mathrm{eig}(S^k + \Lambda^k/\rho) \\ S_1^{k+1} &= U \bullet \mathbf{Diag}(\min(\mathbf{diag}(\Sigma), 0)) \bullet U^T \end{aligned} \tag{3.24}$$

5. Update $\mathscr{Y} = (\mathbf{x}, S)$ via
   solving the first order conditions in (3.22).
6. Update Λ and $\boldsymbol{\mu}$ via (3.16c) and (3.16d), respectively.
7. $k = k + 1$
8. **end while**
9. Determine $\mathbf{x}, \mathbf{z}, S, S_1$

end

6.3.4 Extension to Directed Graphs

For the design of general weighted directed graphs with cardinality constraint, Algorithm 1 can be applied directly. For Algorithm 2, as it is customized for solving the specific problem in (2.3), it is not applicable to general problems. To extend the developed techniques to directed version of problem (2.3) with a asymmetric real Laplacian matrix whose eigenvalues might be complex, we first need to define the notion of algebraic connectivity used for directed graphs. In this paper, we adopt the definition proposed in [7].

Definition 6.3.3. *(Definition 1 in [7]) For a weighted directed graph $\mathcal{G}$ with Laplacian matrix $\mathscr{L}$, the generalized algebraic connectivity $\tilde{\lambda}(\mathscr{L})$ is defined as*

$$\tilde{\lambda}(\mathscr{L}) = \min_{\lambda_i(\mathscr{L})\neq 0, \lambda_i(\mathscr{L})\in\Lambda_{\mathscr{L}}} Re(\lambda_i(\mathscr{L})), \tag{3.25}$$

where $\Lambda_{\mathscr{L}}$ is the spectrum of $\mathscr{L}$ and $Re(\bullet)$ represents the real part of a complex number.

Intuitively, the algebraic connectivity for directed graphs is quantified by the smallest real part of the nonzero eigenvalues. This generalized definition is also applicable to undirected graphs; the reader is referred to [7] for further justification. Moreover, as we consider strongly connected directed graphs in this subsection, $\mathscr{L}$ has only one zero eigenvalue associated with an eigenvector with entries as all ones. Then the inequality from Lemma 6.2.1 can be introduced once again to deal with the zero eigenvalue, as $\mathbf{1}$ is in the null space of the (in-degree) Laplacian. That is,

$$\mathrm{Re}(\lambda_1(\mathscr{L} + \alpha \mathbf{1}\mathbf{1}^T/n)) = \tilde{\lambda}(\mathscr{L}),$$

where $\lambda_1(\bullet)$ denotes the eigenvalue of a square matrix with the smallest real part.

However, as $\mathscr{L} + \alpha \mathbf{1}\mathbf{1}^T/n$ is asymmetric with complex eigenvalues, reformulating the network synthesis into a linear matrix inequality is not straightforward. Given Theorem 10.28 in [216], for any matrix M we have

$$\lambda_1\Big(\frac{M+M^T}{2}\Big) \leq \mathrm{Re}(\lambda_1(M)). \tag{3.26}$$

As a result, $\lambda_1(\frac{M+M^T}{2})$ can be used as a surrogate of $\mathrm{Re}(\lambda_1(M))$. Then taking $M = \mathscr{L} + \alpha \mathbf{1}\mathbf{1}^T/n$ in (3.26), the algebraic connectivity maximization problem for directed graphs can be formulated as

$$\max_{\mathbf{g},\lambda_1} \qquad \lambda_1$$

$$
\begin{aligned}
s.t. \qquad & \mathbf{g} \geq 0, \lambda_1 > 0 \\
& \mathbf{1}^T \mathbf{g} = g_t \\
& \mathbf{Card}(\mathbf{g}) \leq r \\
& \tfrac{\mathscr{L}(\mathscr{G}) + \mathscr{L}^T(\mathscr{G})}{2} + \alpha \mathbf{1}\mathbf{1}^T / n \succeq \lambda_1 \mathbf{I}.
\end{aligned} \tag{3.27}
$$

Due to Lemma 2 in [160], for a strongly connected directed graph, all nontrivial eigenvalues of the Laplacian matrix have positive real parts, i.e., $\tilde{\lambda}(\mathscr{L}) > 0$. Consequently, we have $\lambda_1 > 0$ in order to guarantee the strong connectivity. Moreover, as the off-diagonal entries of $\frac{\mathscr{L}(\mathscr{G}) + \mathscr{L}^T(\mathscr{G})}{2}$ are no longer single variables, to make the Hessian diagonal, the substitution in (3.21) is not applicable. However, the analogous problem to (3.21) for the directed graphs can still be solved as a quadratic programming with equality linear constraints with a diagonal Hessian and thus the customized ADMM can still be applied to (3.27).

6.4 Convergence Analysis

In this section, we aim to prove the global convergence of Algorithms 1 and 2 under fairly mild conditions. We mainly focus on the convergence proof of Algorithm 1 as it is more general than the customized ADMM in Algorithm 2. Based on the convergence analysis of Algorithm 1, an extension to Algorithm 2 is then discussed with additional analysis on convergence conditions and approximate updates. Recall that $\mathscr{C} = \{\mathbf{x} \in \mathbb{R}^p \,|\, \mathbf{h}(\mathbf{x}) \leq 0\}$ and $\boldsymbol{\lambda}^{k+1}$ is denoted as the Lagrange multiplier vector associated with the constraints $\mathbf{h}(\mathbf{x}) \leq 0$ in the optimization problem (3.9). The following two assumptions are now introduced.

Assumption 6.4.1. *$\nabla f(\mathbf{x})$ is Lipschitz continuous; that is, there exists $0 < L_1 < \infty$ such that for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathscr{C}$, $\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L_1 \|\mathbf{x}_1 - \mathbf{x}_2\|$.*

Assumption 6.4.2. *In k and $k+1$ iterations of Algorithm 1 with solution $\mathbf{x}^k$ and $\mathbf{x}^{k+1}$, respectively, there exists $0 \leq L_2 < \infty$ such that $\|\sum \boldsymbol{\lambda}_i^{k+1} \nabla \mathbf{h}_i(\mathbf{x}^{k+1}) - \sum \boldsymbol{\lambda}_i^k \nabla \mathbf{h}_i(x^k)\| \leq L_2 \|\mathbf{x}^{k+1} - \mathbf{x}^k\|$.*

Lemma 6.4.3. *If Assumptions 6.4.1 and 6.4.2 hold, for the two adjacent iterations of Algorithm 1, we have $\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|_F^2 \leq 2(L_1^2 + L_2^2)\, \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_F^2$.*

Proof. The first order optimality conditions for $\mathbf{x}$ in (3.9) is expressed as

$$\begin{aligned} &\nabla f(\mathbf{x}^{k+1}) + \boldsymbol{\mu}^k + \rho(\mathbf{x}^{k+1} - \mathbf{z}^{k+1}) \\ &+ \sum \boldsymbol{\lambda}_i^{k+1} \nabla \mathbf{h}_i(\mathbf{x}^{k+1}) = 0. \end{aligned}$$

Combining with (3.10), we obtain

$$\nabla f(\mathbf{x}^{k+1}) + \boldsymbol{\mu}^{k+1} + \sum \boldsymbol{\lambda}_i^{k+1} \nabla \mathbf{h}_i(\mathbf{x}^{k+1}) = 0. \tag{4.28}$$

As (4.28) holds for each k iteration, combining with Assumptions 6.4.1 and 6.4.2 leads to

$$\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|_F^2 \leq 2(L_1^2 + L_2^2) \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_F^2. \tag{4.29}$$

□

Remark: For the approximate update, $\nabla \mathbf{h}(\mathbf{x})$ appears in the objective function and $\boldsymbol{\lambda}^k$ will be replaced by β in the penalty function. Then the only condition required is that $\nabla \mathbf{h}(\mathbf{x})$ is Lipschitz continuous, which is similar to Assumption 6.4.1 in regards to $\nabla f(\mathbf{x})$. Also, for a convex $f(\mathbf{x})$, let $L_f^l \geq 0$ be the largest possible constant that satisfies $L_f^l I_n \preceq \nabla^2 f(\mathbf{x})$ for any $\mathbf{x} \in \text{dom}(f)$.

Proposition 6.4.4. *If Assumptions 6.4.1 and 6.4.2 hold, the augmented Lagrangian function is monotonically decreasing for the iterations in Algorithm 1 with $\rho^2 + L_f^l \rho - 4(L_1^2 + L_2^2) > 0$ and $\rho > 0$. Mathematically, there exists $c_1 > 0$ such that $\mathscr{L}(P^{k+1}; D^{k+1}) - \mathscr{L}(P^k; D^k) \leq -c_1 \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_F^2$.*

Proof. For the update of $\mathbf{z}$ in (3.8), we have

$$\mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^k;\boldsymbol{\mu}^k)-\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)\leq 0, \tag{4.30}$$

based on the fact that problem (3.8) is globally minimized. For the update of (3.9), we have

$$\begin{aligned}
& \mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^{k+1};\boldsymbol{\mu}^k)-\mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^k;\boldsymbol{\mu}^k)\\
\leq\ & \langle\nabla_{\mathbf{x}}\mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^{k+1};\boldsymbol{\mu}^k),\mathbf{x}^{k+1}-\mathbf{x}^k\rangle\\
& -\frac{\rho+L_f^l}{2}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2\\
\leq\ & -\frac{\rho+L_f^l}{2}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2,
\end{aligned} \tag{4.31}$$

where the first inequality follows that $\mathscr{L}(\mathbf{z},\mathbf{x};\boldsymbol{\mu})$ is strongly convex with regard to $\mathbf{x}$ and L_f^l is the corresponding strong convexity parameter of $f(\mathbf{x})$. In addition, the second inequality follows the first order optimality condition for update of $\mathbf{x}$ with the convex definition of $\mathscr{C}$. Moreover, we have

$$\begin{aligned}
& \mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^{k+1};\boldsymbol{\mu}^{k+1})-\mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^{k+1};\boldsymbol{\mu}^k)\\
=\ & \langle\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k,\mathbf{x}^{k+1}-\mathbf{x}^{k+1}\rangle\\
=\ & \frac{1}{\rho}\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k\|_F^2\\
\leq\ & \frac{2(L_1^2+L_2^2)}{\rho}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2,
\end{aligned} \tag{4.32}$$

where the first equality follows the definition of $\mathscr{L}$, the second follows (3.10), and the inequality follows Lemma 6.4.3. Furthermore, it can be verified that

$$\begin{aligned}
& \mathscr{L}(P^{k+1};D^{k+1})-\mathscr{L}(P^k;D^k)\\
=\ & \mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^k;D^k)-\mathscr{L}(P^k;D^k)\\
& +\mathscr{L}(P^{k+1};D^k)-\mathscr{L}(\mathbf{z}^{k+1},\mathbf{x}^k;D^k)
\end{aligned}$$

$$+\mathscr{L}(P^{k+1};D^{k+1})-\mathscr{L}(P^{k+1};D^{k}). \tag{4.33}$$

By incorporating (4.30), (4.31) and (4.32), (4.33) can be rewritten as

$$\begin{aligned}
& \mathscr{L}(P^{k+1};D^{k+1})-\mathscr{L}(P^{k};D^{k}) \\
\leq\ & -\frac{\rho+L_f^l}{2}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2+\frac{2(L_1^2+L_2^2)}{\rho}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2 \\
\leq\ & -\left(\frac{\rho+L_f^l}{2}-\frac{2(L_1^2+L_2^2)}{\rho}\right)\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_F^2.
\end{aligned} \tag{4.34}$$

With $\rho^2+L_f^l\rho-4(L_1^2+L_2^2)>0$ and $\rho>0$, we have $c_1=\frac{\rho+L_f^l}{2}-\frac{2(L_1^2+L_2^2)}{\rho}>0$. □

In the following, we will prove that $\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)$ is lower bounded and thus convergent.

Assumption 6.4.5. *The sequences $\{f(\mathbf{x}^k)\}$ and $\{\boldsymbol{\mu}^k\}$ are bounded over $\mathscr{C}$.*

Proposition 6.4.6. *If Assumptions 6.4.2 and 6.4.5 hold, the augmented Lagrangian function $\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)$ is lower bounded and convergent.*

Proof. Recall the definition of $\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)$, we have

$$\begin{aligned}
& \mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)=f(\mathbf{x}^k)+\langle\boldsymbol{\mu}^k,\mathbf{z}^k-\mathbf{x}^k\rangle+\frac{\rho}{2}\|\mathbf{x}^k-\mathbf{z}^k\|_F^2 \\
=\ & f(\mathbf{x}^k)+\frac{\rho}{2}\|\mathbf{x}^k-\mathbf{z}^k+\boldsymbol{\mu}^k/\rho\|_F^2-\frac{1}{2\rho}\|\boldsymbol{\mu}^k\|_F^2 \\
>\ & -\infty,
\end{aligned} \tag{4.35}$$

where the inequality follows Assumption 6.4.5. With the sequence $\{\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)\}$ verified to be non-increasing (Proposition 6.4.4) and lower bounded, it leads to the conclusion that it is convergent. □

Due to the relatively restrictive nature of Assumptions 6.4.2 and 6.4.5, we provide some justifications in the following. First, assumptions here are less restrictive compared to

existing works for convergence proof, where the convergence of the dual variables (e.g., [130]) and/or the primal variables (e.g., [102]) are pre-assumed. Moreover, if Assumption 6.4.2 fails, the monotonic decrease of the augmented Lagrangian function in (3.7) may not hold. However, with the bounded dual variable $\boldsymbol{\mu}^k$ (from Assumption 6.4.5) and an increasing penalty coefficient $\rho^k \to \infty$, we can still reach the conclusion that the norm of the successive difference of the primal variable is square summable. In other words, $\sum_k (c_k \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_F^2) < +\infty$ with $c_k > 0$, which indicates the convergence of $\{\mathbf{x}^k\}$. For Assumption 6.4.5, the condition on the boundedness of $\{f(\mathbf{x}^k)\}$ over $\mathscr{C}$ is relatively mild in practice. Thus the main restrictive part is the boundedness of $\{\boldsymbol{\mu}^k\}$. Additionally, if 6.4.5 or both assumptions do not hold, then we might impose the more restrictive assumptions in [130] or ensure a local convergence property instead of the global one.

Remark: Here we will discuss similarities of Algorithm 1 and Algorithm 2 in the convergence analysis, i.e., Propositions 6.4.4 and 6.4.6. In principle, both algorithms consist of two subproblems, with the first being nonconvex and the second being strongly convex. To be more specific, in Algorithm 1, the first subproblem is a projection onto the cardinality constraint and the second is to solve a general semidefinite programming problem. On the other hand, the first subproblem in Algorithm 2 includes projections of two sets of variables, one onto the intersection of the cardinality and nonnegative constraints and the other onto the positive semidefinite cone. The second subproblem is a quadratic programming with equality linear constraints. The mutual property shared by two algorithm is that all the subproblems are solved globally despite the nonconvexity. Moreover, though Algorithm 2 has two sets of consensus constraint, i.e, $\mathbf{x} = \mathbf{z}, S = S_1$, it makes no difference from Algorithm 1 with $\mathbf{x} = \mathbf{z}$, as their coefficients are identity matrices. As a result, with similar assumptions, the convergence analysis of Algorithms 1 applies to Algorithm 2.

Proposition 6.4.7. *Algorithm 1 will globally converge if the prerequisites in Propositions 6.4.4 and 6.4.6 are satisfied. Specifically,*

i) $\lim_{k\to\infty}\{\mathbf{z}^k,\mathbf{x}^k,\boldsymbol{\mu}^k\}=\{\mathbf{z}^*,\mathbf{x}^*,\boldsymbol{\mu}^*\}$ *and* $\mathbf{x}^*=\mathbf{z}^*$. *Moreover,* $(\boldsymbol{\mu}^*)_{\mathscr{I}}=0$ *and* $(|\boldsymbol{\mu}^*/\rho|)_{n/\mathscr{I}}\leq \min(|\mathbf{x}^*|_{\mathscr{I}})$, *where* $\mathscr{I}$ *represents the index set of the r largest elements of* $|\mathbf{x}^*|$; $n/\mathscr{I}$ *is the complement of* $\mathscr{I}$.

ii) $\mathbf{x}^*$ *is the global optimum of the convex optimization problem* $\min_{\mathbf{x}\in\mathscr{C}} f(\mathbf{x})+(\boldsymbol{\mu}^*)^T\mathbf{x}$. *In particular,* $\boldsymbol{\mu}^*=0$ *indicates that in (3.5) the cardinality constraint can be ignored. That is* $\min_{\mathbf{x}\in\mathscr{C}} f(\mathbf{x})$ *has an optimum* $\mathbf{x}^*$ *which satisfies* $|\mathbf{x}^*|_0\leq r$. *In other words, the relaxation is exact.*

Proof. i) Proposition 6.4.6 indicates that the sequence $\{\mathscr{L}(\mathbf{z}^k,\mathbf{x}^k;\boldsymbol{\mu}^k)\}$ converges. Combining (4.34), it leads to

$$\mathbf{x}^{k+1}-\mathbf{x}^k\to 0, \tag{4.36}$$

which indicates that the sequence $\{\mathbf{x}^k\}$ converges. Moreover, as $\{\mathbf{x}^k\}$ converges, the sequence $\{\boldsymbol{\mu}^k\}$ also converges based on (4.29). From (3.10), $\boldsymbol{\mu}^{k+1}=\boldsymbol{\mu}^k+\rho(\mathbf{x}^{k+1}-\mathbf{z}^{k+1})$ and $\mathbf{x}^{k+1}-\mathbf{z}^{k+1}\to 0$. In other words, $\{\mathbf{z}^k\}$ converges to $\mathbf{z}^*$ which is equivalent to $\mathbf{x}^*=\mathbf{z}^*$. Hence, $\lim_{k\to\infty}\{\mathbf{z}^k,\mathbf{x}^k,\boldsymbol{\mu}^k\}=\{\mathbf{z}^*,\mathbf{x}^*,\boldsymbol{\mu}^*\}$ and $\mathbf{x}^*=\mathbf{z}^*$. In the limit, the update for $\mathbf{x}^*$ is expressed as

$$\mathbf{x}^*=\mathbf{z}^*=\mathrm{Proj}_{\{\mathbf{z}|\ |\mathbf{z}|_0\leq r\}}(\mathbf{x}^*+\boldsymbol{\mu}^*/\rho). \tag{4.37}$$

Then we have $(\mathbf{x}^*)_{\mathscr{I}}=(\mathbf{x}^*+\boldsymbol{\mu}^*/\rho)_{\mathscr{I}}$ and $(\mathbf{x}^*)_{n/\mathscr{I}}=0$. Consequently, $(\boldsymbol{\mu}^*)_{\mathscr{I}}=0$ and $(|\boldsymbol{\mu}^*/\rho|)_{n/\mathscr{I}}\leq\min(|\mathbf{x}^*|_{\mathscr{I}})$. This concludes the proof of part i) in Proposition 6.4.7.

ii) The update for $\mathbf{x}$ in (3.9) is to solve a convex problem. The first order optimality condition of (3.9) is expressed as

$$\nabla f(\mathbf{x}^*)+\boldsymbol{\mu}^*+\sum_i \boldsymbol{\lambda}_i^*\nabla\mathbf{h}_i(\boldsymbol{\mu}^*)=0, \tag{4.38}$$

which indicates that $\mathbf{x}^*$ is the global optimum of the convex optimization problem $\min_{\mathbf{x}\in\mathscr{C}} f(\mathbf{x}) + (\boldsymbol{\mu}^*)^T\mathbf{x}$. Intuitively, $\boldsymbol{\mu}^* = 0$ indicates that $\mathbf{x}^*$, the optimum of the relaxed problem in (3.5), denoted as $\min_{\mathbf{x}\in\mathscr{C}} f(\mathbf{x})$, is the exact solution of the original problem when $|\mathbf{x}^*|_0 \leq r$ is satisfied. Taking $r = n$ for instance, the cardinality constraint is inherently redundant and the relaxed solution is guaranteed to be exact. On the other hand, we have $\boldsymbol{\mu}^* = 0$ ($|\mathscr{I}| = n$) from part i) one can derive the same conclusion that the relaxation is exact by following the results of part ii). □

Remark: The second statement of i) is slightly different in Algorithm 2 due to different projection sets.

6.5 Simulation

In this section, the customized ADMM algorithm is applied to the weighted network design problem formulated in (2.3). Comparative results from existing methods are presented to verify the feasibility and improved efficiency of the proposed method. All simulation results are computed on a standard desktop computer with a 3.6 GHz processor and a 32.0 GB memory in MATLAB environment. The computational time could be reduced by a magnitude of ten to hundred when the algorithm is implemented in C/C++.

For comparison purposes, simulation results for average weighted networks are also presented to emphasize the advantage of assigning edge weights as well as designing network topology. The problem for average weighted network is named as 'AveWeighted' and formulated as

$$\begin{aligned} \min_{\mathbf{g},\lambda_2} \quad & -\lambda_2 \\ s.t. \quad & \mathbf{g} \in \{0,1\}^m, \lambda_2(\mathscr{L}) > 0 \end{aligned} \tag{5.39}$$

$$\mathbf{1}^T\mathbf{g} \leq r$$

$$\mathscr{L}(\mathscr{G}) + \alpha\mathbf{1}\mathbf{1}^{\mathbf{T}}/n \succeq \lambda_{\mathbf{2}}\mathbf{I},$$

where $m = n(n-1)/2$ and n is the number of vertices. Moreover, an equivalent formulation of (2.3) for weighted network in the form of Mixed-Integer Semidefinite Programming (MISDP) is represented by

$$\begin{aligned} \min_{\mathbf{g},\lambda_2,\mathbf{y}} \quad & -\lambda_2 \\ s.t. \quad & \mathbf{g} \geq 0, \lambda_2(\mathscr{L}) > 0 \\ & \mathbf{1}^T\mathbf{g} = g_t, \mathbf{1}^T\mathbf{y} \leq r \\ & \mathbf{y} \in \{0,1\}^m, 0 \leq \mathbf{g} \leq g_t\mathbf{y} \\ & \mathscr{L}(\mathscr{G}) + \alpha\mathbf{1}\mathbf{1}^{\mathbf{T}}/n \succeq \lambda_{\mathbf{2}}\mathbf{I}. \end{aligned} \tag{5.40}$$

For all cases, the overall edge weight is set as $g_t = r$.

The first case considered is in a relatively small scale with $n = 6$ and the upper bound the cardinality constraint r is set as $r = \text{round}(0.7m)$ with $m = n(n-1)/2$. There are $\binom{m}{r}$ possible configurations satisfying the cardinality constraint for a network with n vertices and cardinality constraint of r on the edge set. For medium or large-scale network topology design problems, the Exhaustive Search (ES) method is computationally expensive and not feasible to find an optimal solution within practical computational time. For this small-scale case with $n = 6$, it is feasible to find the global optimal solution from ES. In addition, a Branch and Bound (BnB) based method embedded in YALMIP [126] is applied to obtain the global optimum using formulations in (5.39) and (5.40).

Table 6.1 shows the comparative results from five methods, including ES, MISDP, AveWeighted, Algorithm 1 and Algorithm 2, annotated as ES, AveW., MISDP, Alg. 1 and Alg. 2, respectively. While the comparison to the first three algorithms aims to justify the

results, that to Algorithm 1 means to validate that our customized Algorithm 2 does make improvements against Algorithm 1 from the classic ADMM framework. ES and MISDP both obtain global optimal solutions for this small-scale problem with $n = 6$. However, ES is already computationally expensive even for a problem with only six nodes. Although MISDP based on BnB saves computational time compared to ES, it is unable to solve any problems in slightly larger scale. Also in Table 6.1, it shows that for $n = 15$, BnB fails to find the global minimum with an already large maximum iteration number ($5,000$ as we used in simulation). The computational time of Algorithm 2 is significantly reduced in the same case compared to all the rest including Algorithm 1. Correspondingly, Algorithm 2 is more efficient and promising for large-scale problems. To verify the robustness of Algorithm 2 in terms of initial inputs, 200 trials are simulated for both Algorithm 1 and 2 with identical settings and randomly generated initial inputs. The mean value of λ_2 for 200 trails with random initial inputs from Algorithms 2 outperforms 89% of the feasible solutions obtained from ES while the median is only 6% smaller than the global optimum. Moreover, Algorithm 2 intuitively outperforms Algorithm 1 in efficiency despite more iterations as the latter has to solve general semidefinite programming in each iteration. Moreover, the former also achieves better performance index statistically in Table 6.1. Additionally, the λ_2 value comparison between AveWeighted and weighted networks shows the advantage of optimally designing the network topology and allocating the overall assigned edge weights simultaneously. Furthermore, the network design results with corresponding edge weights from different methods and the histogram of λ_2 for 200 trails from Algorithm 1 and 2 are demonstrated in Figure 6.2, where a wider edge represents larger designed edge weight (not in proportion in order to make the difference clear). Again, the skewness of the histogram

of Algorithm 2 onto good performance index indicates its insensitivity to the initial guesses while that does not hold for Algorithm 1.

To demonstrate the influence of r on the five methods when designing a network with same number of nodes, simulation cases for $n = 6$ are provided with varying r bounded by $r \geq 5$ to guarantee connectivity. The algebraic connectivity and the computation time are shown in Fig. 6.3. The simulation results from five methods indicate that Algorithm 2 yields comparable results with reduced computational time. In addition, the efficiency of Algorithm 1 and 2 are insensitive to r while the rest are not guaranteed, especially for ES.

Table 6.1: Results comparison of different algorithms for $n = 6$ and $n = 15$ with $r = \text{round}(0.7m)$ in both cases (* The branch and bound algorithm for AveWeight and MISDP fail to find the global minimum for $n = 15$ even with a large number of maximum iterations).

$n = 6$					
Method	ES	AveW.	MISDP	Alg. 1	Alg. 2
λ_2	3.702	3.00	3.702	max = 3.696 mean = 2.924 med. = 2.901	max = 3.609 mean = 3.365 med. = 3.459
Time/s	217.3	3.31	51.49	mean = 13.91	mean = 0.041
Iter. #	-	-	-	mean = 65.3	mean = 197.6
$n = 15$					
Method	ES	AveW.	MISDP	Alg. 1	Alg. 2
λ_2	-	7.457*	7.729*	max = 9.121 mean = 8.414 med. = 8.413	max = 9.541 mean = 9.182 med. = 9.309
Time/s	-	1.59e4	3.02e4	mean = 22.06	mean = 0.14
Iter. #	-	-	-	mean = 66.40	mean = 441.99

To verify the scalability of Algorithm 2, the algorithm is applied to solve problems at different scales with n ranging from 100 to 1500. Given the scalability issues for the other four algorithms, we exclude reporting their implementations in this paper. When $n = 1500$,

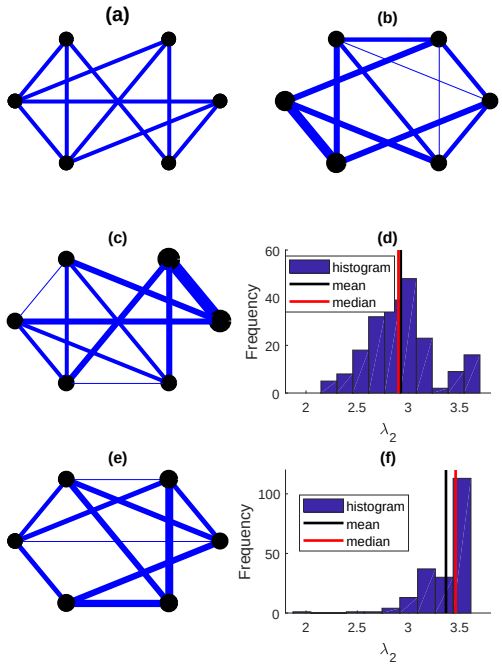


Figure 6.2: Network topology and weights from (a) AveWeighted, (b) MISDP, (c) Algorithm 1; (d) histogram of λ_2 for 200 trails of Algorithm 1, (e) Algorithm 2, (f) histogram of λ_2 for 200 trails of Algorithm 2.

the number of overall edges is more than one million $(1,124,250)$. Figure 6.4 shows the computational time versus n. When $n = 1500$, the total computational time is only about 3 minutes, among which the customized ADMM iterations only takes less than 90 seconds. This validates the superior computational efficiency of the proposed Algorithm 2. Moreover, Figure 6.5 demonstrates the convergence of the relative prime feasibility of the coupling constraints, and Figure 6.6 shows the monotonic decreasing of the augmented Lagrangian function value as proved in this work. Finally, the weighted network design results for the case with $n = 20$ is given in Figure 6.7 for illustration.

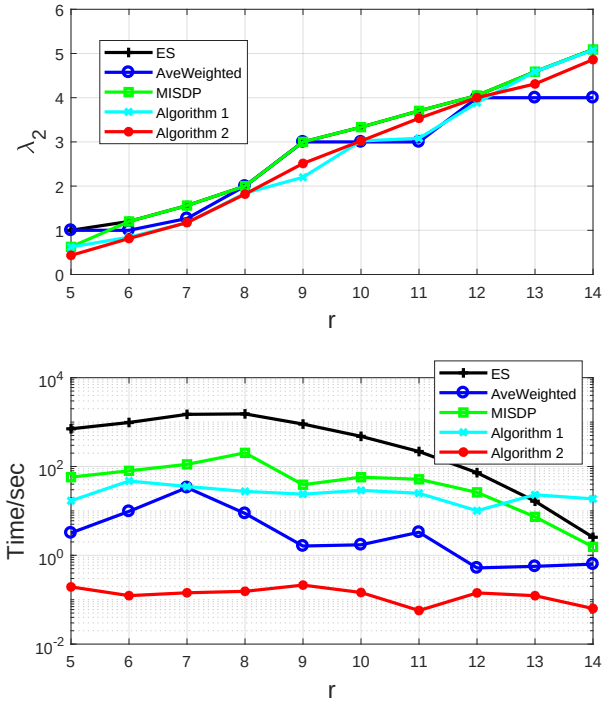


Figure 6.3: Objective (above) and computation time (below) comparison of the five algorithms for $n = 6$ with varying r. Note that BnB fails to obtain the global optimum for MISDP with $r = 5$.

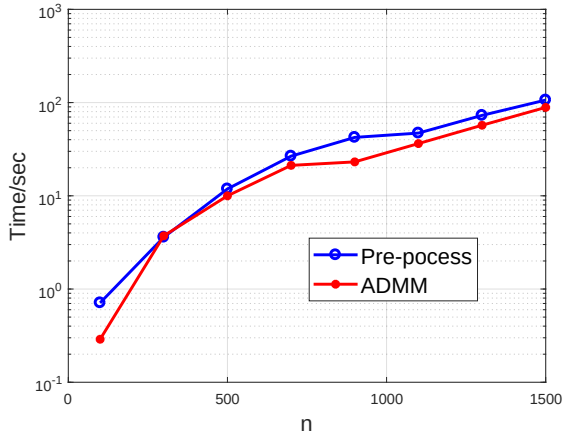


Figure 6.4: Computational performance of Algorithm 2 with respect to n.

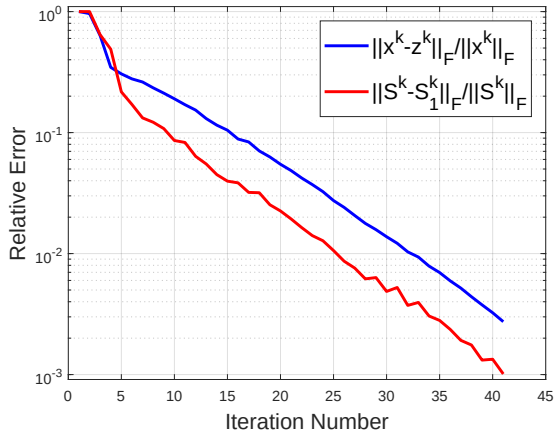


Figure 6.5: Convergence of the relative prime feasibility of the coupling constraints.

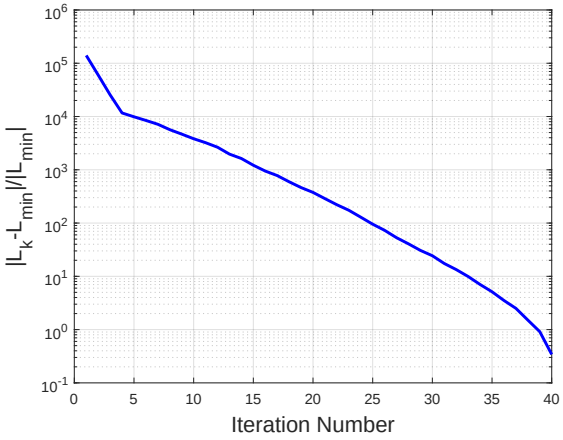


Figure 6.6: Convergence history of the augmented Lagrangian function value.

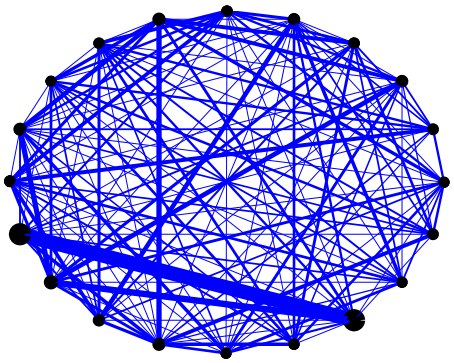


Figure 6.7: Network topology and weighs for a case with $n = 20$.

6.6 Conclusions

This paper develops a computationally efficient method to design network topology and edge weights to optimize certain network properties while considering cardinality constraint on the edge set. The weighted network design problem with a cardinality constraint is formulated as a general cardinality constrained optimization problem (CCOP) and a customized Alternating Direction Method of Multipliers (ADMM) algorithm is proposed for its solution. The proposed algorithm takes advantages of the special structure of the problem to find closed-form solutions for iterative updates with linear matrix inequalities. Additionally, we prove that the proposed ADMM globally converges to a stationary point for a general cardinality constrained optimization problem under certain conditions. Simulation results illustrate that the customized ADMM significantly improves the computational efficiency and scalability of the proposed approach.

Chapter 7: Future Work

In this chapter, we present possible extensions and improvements to our proposed techniques.

It is desirable to develop distributed algorithms for nonconvex optimization, such as low-rank optimization. Distributed optimization can decompose an optimization problem into multiple simpler subproblems which can be solved in parallel. As a result, it is widely investigated in various areas such as signal processing and machine learning. Recent years have witnessed a booming interest on distributed optimization for solving convex programming. Usually it is the case that a multi-agent system works cooperatively to optimize the global objective function, which is the sum of the local objectives only known to the specific agents. In contrast, work on nonconvex distributed optimization is relatively limited. The convergence property of a distributed projected stochastic gradient algorithm for non-convex optimization is analyzed with the nonconvexity lying in the objective. However, in a number of applications, that is not the case. For example in machine learning, Distributed Machine Learning (D-ML) refers to multi-agent machine learning algorithms, which aims to work with large datasets and improve efficiency. The necessity of D-ML for the large-scale network of agents arises mainly in two aspects. First, the decentralized storage of the massive data and the parallel computing to learn the complex models make distributed machine learning necessary in terms of efficiency or even feasibility.

Secondly, in decentralized systems, where an individual agent has only a partial view of the system and cannot get access to the global information, learning methods have to be implemented in parallel on individual agents followed by the communication. Consequently, the application of distributed machine learning can be found in domains like distributed control, resource management and multi-robot coordination, perception and planning in a dynamic environment. For example, the robot team is required to cover a certain area, with sensing the environment partially and communication to the neighbor teammates. Then each robot is required to plan its own efficient path to jointly ensure the coverage. After reviewing the state-of-the-art of the distributed optimization algorithms to train D-ML models, the major gap is due to the frequently encountered non-convexity. There are a few issues waiting to be addressed. At first, the critical issue is the convergence and maybe further convergence rate. Existing work analyzes the convergence property of a non-convex distributed algorithm with the non-convexity mainly lying in the objective in the literature. However, we aim to deal with a more general case here with non-convexity possibly in the coupling constraints involving multiple agents. Moreover, as D-RL requires communication between agents, the message conveyed might be missing or delayed and the network might be time-varying. Correspondingly, stochastic and asynchronous nonconvex distributed optimization will be investigated to address such issues. Moreover, we also aim to develop distributed control algorithm for the multi-robot system such that the system will work to cooperatively accomplish a certain mission, with each UAV learning to plan its own path in real time. In conclusion, the proposed research will make an impactful difference in terms of both theoretical guarantee and real-world applications.

Chapter 8: Conclusion

In this chapter, we summarize the status of the work presented in this dissertation.

This dissertation is devoted to solving matrix rank(vector cardinality)-related optimization problems. Firstly, we bridged the gap by demonstrating that rank minimization problems (RMPs) can be equivalently transformed into rank-constrained optimization problems (RCOPs) by introducing a quadratic matrix equality constraint. This means that RMP is a special case of RCOP with an upper bounded rank value.

Secondly, we proposed an algorithm based on alternating minimization, named iterative rank minimization (IRM). The IRM method, with each subproblem formulated as a convex optimization problem, aimed to solve general RCOPs, including special cases such as quadratically constrained quadratic programming. Combined with the semidefinite embedding lemma, IRM can deal with rank constraints on general rectangular matrices. For completeness, the IRM method is also extended to solve RCOPs with rank inequalities constrained by upper or lower bounds, as well as rank equality constraints. As a special case of alternating minimization, the convergence of IRM was proved. Furthermore, the proposed IRM method is applied to solve cardinality minimization problems and cardinality-constrained optimization problems (CCOPs), which are handled as special cases of RMPs and RCOPs, respectively. To verify the effectiveness and improved performance of the proposed IRM method, representative applications, including matrix completion, system

identification, output feedback stabilization, structured H_2 controller design, and cardinality minimization problems, UAV path planning, network identification, were presented with comparative results. Additionally, for sparse QCQP, we develop a variant of IRM to enhance the efficiency by exploiting the sparsity.

Thirdly, we proposed a second algorithm, based on the framework of alternating direction method of multipliers (ADMM) aims to get simple subproblem solutions or even in closed form. By matrix factorization theory, we reformulate the rank constraint to the Burer-Monteiro form for simpler subproblems. Despite the nonconvex rank constraint, the ADMM iterates are proved to converge to a stationary point in both formulations, under mild assumptions. Additionally, as recent work suggests, when the matrix factors in this latter form satisfy some checkable conditions, local optimum with high probability is also the global optimum. To validate the efficacy, results for MAX-CUT, community detection, and image segmentation benchmark and simulated examples were provided. In a variant of the algorithm, an approximate representation is used to reformulate the original RCOP. The approximate formulation and the introduced bilinear constraint make each subproblem of ADMM more computationally-efficient, as the trade-off of approximation. In particular cases, the subproblems can even admit a closed form. The global convergence of the proposed algorithm to a local minimizer of the approximate problem is proved. Based on inexact augmented Lagrangian method, another variant focused on solving QCQPs with both equality and inequality constraints. Noteworthily, the subproblem still admits a simple closed-form solution. By fully exploiting the structure of the specific problem, such as network module detection, the efficiency can be further improved.

Fourthly, the simultaneous design of the network topology and the corresponding edge weights in presence of a cardinality constraint on the edge set is also investigated. Network

properties of interest for this design problem led to optimization formulations with a nonconvex cardinality constraint. This class of problems is referred to as CCOPs and the presence of the cardinality constraint generally makes CCOPs NP-hard. In this work, we proposed a customized ADMM algorithm aiming to improve the scalability of the solution strategy for large-scale CCOPs. This algorithm utilized the special structure of the problem formulation to obtain closed-form solutions for each iterative step of the corresponding ADMM updates. We also provided the convergence proof of the proposed customized ADMM to a stationary point under certain conditions.

In conclusion, this dissertation developed algorithmic frameworks to solve RCOPs and RMPs with convergence analysis. Comparative simulation results with the state-of-art algorithms validated the efficacy and effectiveness of the proposed algorithms.

Bibliography

[1] Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1):471–487, 2016.

[2] Jacob Abernethy, Francis Bach, Theodoros Evgeniou, and Jean-Philippe Vert. Low-rank matrix factorization with attributes. *arXiv preprint cs/0611124*, 2006.

[3] Behcet Acikmese and Lars Blackmore. Lossless convexification of a class of optimal control problems with non-convex control constraints. *Automatica*, 47(2):341–347, 2011.

[4] F. A. Al-Khayyal, C. Larsen, and T. Van Voorhis. A relaxation method for nonconvex quadratically constrained quadratic programs. *Journal of Global Optimization*, 6:215–230, 1995.

[5] Faiz A Al-Khayyal, Christian Larsen, and Timothy Van Voorhis. A relaxation method for nonconvex quadratically constrained quadratic programs. *Journal of Global Optimization*, 6(3):215–230, 1995.

[6] IP Androulakis, CD Maranas, and CA Floudas. αbb: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7(4):337–363, 1995.

[7] Mohammad Mehdi Asadi, Mohammad Khosravi, Amir G Aghdam, and Stephane Blouin. Generalized algebraic connectivity for asymmetric networks. In *American Control Conference*, pages 5531–5536, 2016.

[8] Mustafa Ayazoglu, Mario Sznaier, and Necmiye Ozay. Blind identification of sparse dynamic networks and applications. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 2944–2950. IEEE, 2011.

[9] Tucker Balch and Ronald C Arkin. Behavior-based formation control for multirobot teams. *Robotics and Automation, IEEE Transactions on*, 14(6):926–939, 1998.

[10] Xiaowei Bao, Nikolaos Sahinidis, and Mohit Tawarmalani. Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons. *Mathematical Programming*, 129:129–157, 2011.

[11] Xiaowei Bao, Nikolaos V Sahinidis, and Mohit Tawarmalani. Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons. *Mathematical programming*, 129(1):129–157, 2011.

[12] Francisco Barahona, Martin Grötschel, Michael Jünger, and Gerhard Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988.

[13] Jérôme Barraquand, Lydia Kavraki, Rajeev Motwani, Jean-Claude Latombe, Tsai-Yen Li, and Prabhakar Raghavan. A random sampling scheme for path planning. In *Robotics Research*, pages 249–264. Springer, 1996.

[14] Alexander I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13(2):189–202, 1995.

[15] Al W. Beard, Jonathan Lawton, and Fred Y. Hadaegh. A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, 9:777–790, 2001.

[16] Randal W Beard, Jonathan Lawton, Fred Y Hadaegh, et al. A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, 9(6):777–790, 2001.

[17] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

[18] D. Bertsimas. Statistics and machine learning via a modern optimization lens. In *INFORMS Annual Meeting. Catonsville, MD: The Institute for Operations Research and the Management Sciences*, 2014.

[19] Dimitris Bertsimas and Romy Shioda. Algorithm for cardinality-constrained quadratic optimization. *Computational Optimization and Applications*, 43(1):1–22, 2009.

[20] Pratik Biswas, Tzu-Chen Lian, Ta-Chung Wang, and Yinyu Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks (TOSN)*, 2(2):188–220, 2006.

[21] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[22] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308, 2006.

[23] Kevin P Bollino, L Ryan Lewis, Pooya Sekhavat, and I Michael Ross. Pseudospectral optimal control: a clear road for autonomous intelligent path planning. In *Proceedings of the AIAA InfoTech at Aerospace Conference and Exhibit*, pages 1228–1241, 2007.

[24] Brian Borchers. CSDP, AC library for semidefinite programming. *Optimization methods and Software*, 11(1-4):613–623, 1999.

[25] Scott Bortoff et al. Path planning for uavs. In *American Control Conference, 2000. Proceedings of the 2000*, volume 1, pages 364–368, 2000.

[26] Nicolas Boumal, Vlad Voroninski, and Afonso Bandeira. The non-convex Burer-Monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems*, pages 2757–2765, 2016.

[27] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[28] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[29] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. Maximizing modularity is hard. *arXiv preprint physics/0608255*, 2006.

[30] Arthur E Bryson and Yu-Chi Ho. *Applied optimal control: optimization, estimation, and control*. Taylor & Francis, 1975.

[31] Samuel Burer and Hongbo Dong. Representing quadratically constrained quadratic programs as generalized copositive programs. *Operations Research Letters*, 40(3):203–206, 2012.

[32] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

[33] Samuel Burer and Renato DC Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.

[34] Samuel Burer and Dieter Vandenbussche. A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. *Mathematical Programming*, 113(2):259–282, 2008.

[35] EJ Candès, MB Wakin, and S Boyd. Enhancing sparsity by reweighted l_1 minimization. *Journal for Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

[36] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[37] Emmanuel J Candes and Terence Tao. Decoding by linear programming. *Information Theory, IEEE Transactions on*, 51(12):4203–4215, 2005.

[38] Yongcan Cao. Uav circumnavigating an unknown target under a gps-denied environment with range-only measurements. *Automatica*, 55:150–158, 2015.

[39] Airlie Chapman, Ran Dai, and Mehran Mesbahi. Network topology design for uav swarming with wind gusts. 2011.

[40] Howie Choset. Robotic motion planning: Rrt's, November 2007. Visited on 2015-09-27.

[41] P. Cui, W. Zhong, and H. Cui. Onboard spacecraft slew-planning by heuristic state-space search and optimization. *International Conference on Mechatronics and Automation*, pages 2115–2119, 2007.

[42] R. Dai. Three-dimensional aircraft path planning based on nonconvex quadratic optimization. In *American Control Conference, Accepted*, 2014.

[43] R. Dai and C. Sun. Path planning of spatial rigid motion with constrained attitude. *Journal of Guidance, Control, and Dynamics*, 38:1356–1365, 2015.

[44] Ran Dai, Unsik Lee, Saghar Hosseini, and Mehran Mesbahi. Optimal path planning for solar-powered uavs based on unit quaternions. In *Decision and Control, 2012 IEEE 51st Annual Conference on*, pages 3104–3109, 2012.

[45] Ran Dai, Joshua Maximoff, and Mehran Mesbahi. Optimal trajectory generation for establishing connectivity in proximity networks. *Aerospace and Electronic Systems, IEEE Transactions on*, 49(3):1968–1981, 2013.

[46] Ran Dai and Mehran Mesbahi. Optimal topology design for dynamic networks. In *Decision and Control and European Control Conference, 50th IEEE Conference on*, pages 1280–1285, 2011.

[47] Ran Dai and Chuangchuang Sun. Path planning of spatial rigid motion with constrained attitude. *Journal of Guidance, Control, and Dynamics*, 38(8):1356–1365, 2015.

[48] Alexandre d'Aspremont and Stephen Boyd. Relaxations and randomized methods for nonconvex qcqps. *EE392o Class Notes, Stanford University*, 2003.

[49] Jon Dattorro. *Convex optimization & Euclidean distance geometry*. Meboo Publishing USA, 2015.

[50] Caterina De Simone, Martin Diehl, Michael Jünger, Petra Mutzel, Gerhard Reinelt, and Giovanni Rinaldi. Exact ground states of ising spin glasses: New experimental results with a branch-and-cut algorithm. *Journal of Statistical Physics*, 80(1-2):487–496, 1995.

[51] R. A. Delgado, J. C. Agüero, and G. C. Goodwin. A rank-constrained optimization approach: Application to factor analysis. In *19th IFAC World Congress*, 2014.

[52] R. A. Delgado, J. C. Agüero, and G. C. Goodwin. A novel representation of rank constraints for real matrices. *Linear Algebra and its Applications*, 496:452–462, 2016.

[53] M. Diehl. Formulation of closed-loop min-max mpc as a quadratically constrained quadratic program. *Automatic Control, IEEE Transactions on*, 52(2):339–343, Feb 2007.

[54] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

[55] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104, 2005.

[56] Jonathan Eckstein and Dimitri P Bertsekas. On the Douglas Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992.

[57] Laurent El Ghaoui and Pascal Gahinet. Rank minimization under lmi constraints: A framework for output feedback problems. 1993.

[58] Laurent El Ghaoui, Francois Oustry, and Mustapha AitRami. A cone complementarity linearization algorithm for static output-feedback and related problems. *Automatic Control, IEEE Transactions on*, 42(8):1171–1176, 1997.

[59] E. Elhamifar and R. Vidal. Block-sparse recovery via convex optimization. *Signal Processing, IEEE Transactions on*, 60(8):4094–4107, Aug 2012.

[60] Mohammad Fardad and Mihailo R Jovanovic. On the design of optimal structured and sparse feedback gains via sequential convex programming. In *American Control Conference*, pages 2426–2431, 2014.

[61] Bassem Fares, Pierre Apkarian, and Dominikus Noll. An augmented lagrangian method for a class of lmi-constrained problems in robust control theory. *International Journal of Control*, 74(4):348–360, 2001.

[62] Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

[63] Maryam Fazel, Haitham Hindi, and S Boyd. Rank minimization and applications in system theory. In *American Control Conference, Proceedings of the 2004*, volume 4, pages 3273–3278, 2004.

[64] Maryam Fazel, Haitham Hindi, and Stephen P Boyd. A rank minimization heuristic with application to minimum order system approximation. In *American Control Conference, Proceedings of the 2001*, volume 6, pages 4734–4739, 2001.

[65] Maryam Fazel, Haitham Hindi, and Stephen P Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *American Control Conference, 2003. Proceedings of the 2003*, volume 3, pages 2156–2162. IEEE, 2003.

[66] Roger Fletcher and Sven Leyffer. Numerical experience with lower bounds for miqp branch-and-bound. *SIAM Journal on Optimization*, 8(2):604–616, 1998.

[67] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[68] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[69] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1–44, 2016.

[70] J.P. Frakes, D.A. Henretty, T.W. Flatley, F.L. Markley, J. San, and E.G. Lightsey. Sampex science pointing with velocity avoidance. *Space Mechanics*, 1:949–966, 1992.

[71] E. Frazzoli, E. Dahleh, M. A.and Feron, and R. P. Kornfeld. A randomized attitude slew planning algorithm for autonomous spacecraft. *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2001.

[72] Tetsuya Fujie and Masakazu Kojima. Semidefinite programming relaxation for nonconvex quadratic programs. *Journal of Global Optimization*, 10(4):367–380, 1997.

[73] Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976.

[74] Jianjun Gao and Duan Li. Cardinality constrained linear-quadratic optimal control. *IEEE Transactions on Automatic Control*, 56(8):1936–1941, 2011.

[75] Arpita Ghosh, Stephen Boyd, and Amin Saberi. Minimizing effective resistance of a graph. *SIAM review*, 50(1):37–66, 2008.

[76] Roland Glowinski and A Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 9(R2):41–76, 1975.

[77] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[78] Tom Goldstein and Stanley Osher. The split Bregman method for L1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

[79] G. H. Golub, A. Hoffman, and G. W. Stewart. A generalization of the eckart-young-mirsky matrix approximation theorem. *Linear Algebra and its applications*, 88:317–327, 1987.

[80] Qi Gong, Ryan Lewis, and Michael Ross. Pseudospectral motion planning for autonomous vehicles. *Journal of Guidance, Control, and Dynamics*, 32(3):1039–1045, 2009.

[81] Karolos M Grigoriadis and Robert E Skelton. Low-order control design for LMI problems using alternating projection methods. *Automatica*, 32(8):1117–1125, 1996.

[82] Michael J Grimble. *Robust industrial control: optimal design approach for polynomial systems*. Prentice-Hall, Inc., 1994.

[83] Roger Guimera, Marta Sales-Pardo, and Luís A Nunes Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101, 2004.

[84] H. B. Hablani. Attitude commands avoiding bright objects and maintaining communication with ground station. *AIAA Journal of Guidance, Control, and Dynamics*, 22(6):759–767, 1999.

[85] Davood Hajinezhad, Tsung-Hui Chang, Xiangfeng Wang, Qingjiang Shi, and Mingyi Hong. Nonnegative matrix factorization using ADMM: Algorithm and convergence analysis. In *Acoustics, Speech and Signal Processing, IEEE International Conference on*, pages 4742–4746, 2016.

[86] Elaine T Hale, Wotao Yin, and Yin Zhang. A fixed-point continuation method for l_1-regularized minimization with applications to compressed sensing. *CAAM TR07-07, Rice University*, 43:44, 2007.

[87] William Rowan Hamilton. Ii. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 25(163):10–13, 1844.

[88] Arash Hassibi, Jonathan How, and Stephen Boyd. A path-following method for solving bmi problems in control. In *American Control Conference. Proceedings of the 1999*, volume 2, pages 1385–1389, 1999.

[89] Christoph Helmberg. *Semidefinite programming for combinatorial optimization*. Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2000.

[90] Christoph Helmberg and Franz Rendl. Solving quadratic (0, 1)-problems by semidefinite programs and cutting planes. *Mathematical programming*, 82(3):291–315, 1998.

[91] Christoph Helmberg, Franz Rendl, Robert J Vanderbei, and Henry Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, 1996.

[92] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[93] Petter Holme, Mikael Huss, and Hawoong Jeong. Subnetwork hierarchies of biochemical pathways. *Bioinformatics*, 19(4):532–538, 2003.

[94] Kenneth Holmström. Tomlab–an environment for solving optimization problems in matlab. In *Proceedings for the Nordic Matlab Conference'97*, 1997.

[95] Kenneth Holmström. The tomlab optimization environment in matlab. 1999.

[96] Mingyi Hong and Zhi-Quan Luo. On the linear convergence of the alternating direction method of multipliers. *Mathematical Programming*, pages 1–35, 2012.

[97] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

[98] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge university press, 2012.

[99] Kejun Huang and Nicholas D Sidiropoulos. Consensus-ADMM for general quadratically constrained quadratic programming. *IEEE Transactions on Signal Processing*, 64(20):5297–5310, 2016.

[100] R.A. Jabr. Radial distribution load flow using conic programming. *Power Systems, IEEE Transactions on*, 21(3):1458–1459, Aug 2006.

[101] Hui Ji, Chaoqiang Liu, Zuowei Shen, and Yuhong Xu. Robust video denoising using low rank matrix completion. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1791–1798. IEEE, 2010.

[102] Bo Jiang, Shiqian Ma, and Shuzhong Zhang. Alternating direction method of multipliers for real and complex polynomial optimization models. *Optimization*, 63(6):883–898, 2014.

[103] Timothy R Jorris and Richard G Cobb. Three-dimensional trajectory optimization satisfying waypoint and no-fly zone constraints. *Journal of Guidance, Control, and Dynamics*, 32(2):551–572, 2009.

[104] Rudolf Emil Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial & Applied Mathematics, Series A: Control*, 1(2):152–192, 1963.

[105] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[106] Sertac et al Karaman. Anytime motion planning using the rrt*. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483. IEEE, 2011.

[107] Lydia E Kavraki, Petr Švestka, Jean-Claude Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566–580, 1996.

[108] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

[109] Seog-Joo Kim and Young-Hyun Moon. Structurally constrained h_2 and h_∞ control: A rank-constrained LMI approach. *Automatica*, 42(9):1583–1588, 2006.

[110] Y. Kim and M. Mesbahi. Quadratically constrained attitude control via semidefinite programming. *IEEE Transactions on Automatic Control*, pages 731–735, 2004.

[111] .R. P. Kornfeld. On-board autonomous attitude maneuver planning for planetary spacecraft using genetic algorithms. *AIAA Conference on Guidance, Navigation and Control*, 2003.

[112] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Robotics and Automation, IEEE International Conference on*, volume 2, pages 995–1001, 2000.

[113] Jack B Kuipers. *Quaternions and rotation sequences*, volume 66. Princeton university press Princeton, 1999.

[114] Sun-Yuan Kung, KS Arun, and DV Rao. State-space and singular-value decomposition-based approximation methods for the harmonic retrieval problem. *JOSA*, 73(12):1799–1811, 1983.

[115] Jean B Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[116] J. Lavaei and S.H. Low. Zero duality gap in optimal power flow problem. *Power Systems, IEEE Transactions on*, 27(1):92–107, Feb 2012.

[117] K. Lee and Y. Bresler. Admira: Atomic decomposition for minimum rank approximation. *Information Theory, IEEE Transactions on*, 56(9):4402–4416, 2010.

[118] U. Lee and M. Mesbahi. Spacecraft reorientation in presence of attitude constraints via logarithmic barrier potentials. *American Control Conference*, pages 450–455, 2011.

[119] Guoyin Li and Ting Kei Pong. Global convergence of splitting methods for nonconvex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460, 2015.

[120] Fu Lin, Mohammad Fardad, and Mihailo R Jovanovic. Design of optimal sparse feedback gains via the alternating direction method of multipliers. *Automatic Control, IEEE Transactions on*, 58(9):2426–2431, 2013.

[121] Tianyi Lin, Shiqian Ma, and Shuzhong Zhang. On the global linear convergence of the admm with multiblock variables. *SIAM Journal on Optimization*, 25(3):1478–1497, 2015.

[122] Jeff Linderoth. A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs. *Mathematical Programming*, 103(2):251–282, 2005.

[123] Pierre-Louis Lions and Bertrand Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979.

[124] Xinfu Liu and Ping Lu. Solving nonconvex optimal control problems by convex optimization. *Journal of Guidance, Control, and Dynamics*, 37(3):750–765, 2014.

[125] L. Ljung. System identification theory for the user, 1999.

[126] Johan Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Computer Aided Control Systems Design, IEEE International Symposium on*, pages 284–289, 2004.

[127] Songtao Lu, Mingyi Hong, and Zhengdao Wang. A nonconvex splitting method for symmetric nonnegative matrix factorization: Convergence analysis and optimality. *IEEE Transactions on Signal Processing*, 2017.

[128] Zhi-quan Luo, Wing-kin Ma, AM-C So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *Signal Processing Magazine, IEEE*, 27(3):20–34, 2010.

[129] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[130] Sindri Magnússon, Pradeep Chathuranga Weeraddana, Michael G Rabbat, and Carlo Fischione. On the convergence of alternating direction Lagrangian methods for nonconvex structured optimization problems. *IEEE Transactions on Control of Network Systems*, 3(3):296–309, 2016.

[131] Jérôme Malick and Hristo S Sendov. Clarke generalized jacobian of the projection onto the cone of positive semidefinite matrices. *Set-Valued Analysis*, 14(3):273–293, 2006.

[132] I. Markovsky. Structured low-rank approximation and its applications. *Automatica*, 44(4):891–909, 2008.

[133] I. Markovsky. *Low rank approximation: algorithms, implementation, applications*. Springer Science & Business Media, 2011.

[134] I. Markovsky and K. Usevich. Software for weighted structured low-rank approximation. *Journal of Computational and Applied Mathematics*, 256:278–292, 2014.

[135] I. Markovsky and S. Van Huffel. High-performance numerical algorithms and software for structured total least squares. *Journal of Computational and Applied Mathematics*, 180(2):311–331, 2005.

[136] I. Markovsky and S. Van Huffel. Left vs right representations for solving weighted low-rank approximation problems. *Linear algebra and its applications*, 422(2):540–552, 2007.

[137] I. Markovsky, J. C. Willems, P. Rapisarda, and B. L. De Moor. Algorithms for deterministic balanced subspace identification. *Automatica*, 41(5):755–766, 2005.

[138] Donatello Materassi and Giacomo Innocenti. Topological identification in networks of dynamical systems. *Automatic Control, IEEE Transactions on*, 55(8):1860–1871, 2010.

[139] Massimiliano Mattei and Luciano Blasi. Smooth flight trajectory planning in the presence of no-fly zones and obstacles. *Journal of guidance, control, and dynamics*, 33(2):454–462, 2010.

[140] GARY A McCuE. Quasilinearization determination of optimum finite-thrust orbital transfers. *AIAA Journal*, 5(4):755–763, 1967.

[141] C. R. McInnes. Large angle slew meneuvers with autonomous sun vector avoidance. *AIAA Journal of Guidance, Control and Dynamics*, 17(4):875–877, 1994.

[142] Raghu Meka, Prateek Jain, Constantine Caramanis, and Inderjit S Dhillon. Rank minimization via online learning. In *Proceedings of the 25th International Conference on Machine learning*, pages 656–663. ACM, 2008.

[143] M Mesbahi and M Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010.

[144] Mehran Mesbahi. On the rank minimization problem and its control applications. *Systems & control letters*, 33(1):31–36, 1998.

[145] Mehran Mesbahi and George P Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *Automatic Control, IEEE Transactions on*, 42(2):239–243, 1997.

[146] G. Meyer. *Geometric optimization algorithms for linear regression on fixed-rank matrices*. PhD thesis, University of Liege, Belgium, 2011.

[147] Sepideh Hassan Moghaddam and Mihailo R Jovanovic. Topology design for stochastically-forced consensus networks. *IEEE Transactions on Control of Network Systems, pusblished online*, 2017.

[148] Karthik Mohan and Maryam Fazel. Iterative reweighted algorithms for matrix rank minimization. *The Journal of Machine Learning Research*, 13(1):3441–3473, 2012.

[149] B. De Moor. *Daisy: Database for the identification of systems*, (accessed 2016-06-24). `http://homes.esat.kuleuven.be/~smc/daisy/daisydata.html`.

[150] M. Nabi-Abdolyousefi and M. Mesbahi. Sieve method for consensus-type network tomography. *IET Control Theory & Applications*, 6(12):1926–1932, 2012.

[151] M. Nabi-Abdolyyousefi and M. Mesbahi. Network identification via node knock-out. In *49th IEEE Conference on Decision and Control*, pages 2239–2244, 2010.

[152] Domenico Napoletani and Timothy D Sauer. Reconstructing the topology of sparsely connected dynamical networks. *Physical Review E*, 77(2):026103, 2008.

[153] S. Narasimhan and R. Rengaswamy. Plant friendly input design: Convex relaxation and quality. *Automatic Control, IEEE Transactions on*, 56(6):1467–1472, June 2011.

[154] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[155] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[156] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[157] Mark EJ Newman and Gesine Reinert. Estimating the number of communities in a network. *Physical review letters*, 117(7):078301, 2016.

[158] MEJ Newman. Equivalence between modularity optimization and maximum likelihood methods for community detection. *Physical Review E*, 94(5):052315, 2016.

[159] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *Automatic Control, IEEE Transactions on*, 51(3):401–420, 2006.

[160] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[161] Robert Orsi, Uwe Helmke, and John B Moore. A newton-like method for solving rank constrained linear matrix inequalities. *Automatica*, 42(11):1875–1882, 2006.

[162] Jaehyun Park and Stephen Boyd. A semidefinite programming method for integer convex quadratic minimization. *Optimization Letters*, pages 1–20, 2017.

[163] Gábor Pataki. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998.

[164] Svatopluk Poljak, Franz Rendl, and Henry Wolkowicz. A recipe for semidefinite relaxation for (0, 1)-quadratic programming. *Journal of Global Optimization*, 7(1):51–73, 1995.

[165] Svatopluk Poljak and Zsolt Tuza. The expected relative error of the polyhedral approximation of the MAX-CUT problem. *Operations Research Letters*, 16(4):191–198, 1994.

[166] Yuan-Qing Qin, De-Bao Sun, Ning Li, and Yi-Gang Cen. Path planning for mobile robot using the particle swarm optimization with mutation operator. In *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, volume 4, pages 2473–2478. IEEE, 2004.

[167] Andrea Qualizza, Pietro Belotti, and François Margot. Linear programming relaxations of quadratically constrained quadratic programs. In *Mixed Integer Nonlinear Programming*, pages 407–426. Springer, 2012.

[168] B. Recht. A simpler approach to matrix completion. *The Journal of Machine Learning Research*, 12:3413–3430, 2011.

[169] Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[170] Franz Rendl, Giovanni Rinaldi, and Angelika Wiegele. A branch and bound algorithm for MAX-CUT based on combining semidefinite and polyhedral relaxations. In *IPCO*, volume 4513, pages 295–309. Springer, 2007.

[171] David P Rutenberg and Timothy L Shaftel. Product design: Subassemblies for multiple markets. *Management Science*, 18(4-part-i):B–220, 1971.

[172] Borhan M Sanandaji, Tyrone L Vincent, and Michael B Wakin. Exact topology identification of large-scale interconnected dynamical systems from compressive observations. In *American Control Conference (ACC), 2011*, pages 649–656. IEEE, 2011.

[173] Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in ad hoc wsns with noisy links—part i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350–364, 2008.

[174] S Yusef Shafi, Murat Arcak, and Laurent El Ghaoui. Designing node and edge weights of a graph to meet Laplacian eigenvalue constraints. In *Communication, Control, and Computing, 48th Annual Allerton Conference on*, pages 1016–1023, 2010.

[175] S Yusef Shafi, Murat Arcak, and Laurent El Ghaoui. Graph weight design for laplacian eigenvalue constraints with multi-agent systems applications. In *Decision and Control and European Control Conference, 2011 50th IEEE Conference on*, pages 5541–5546. IEEE, 2011.

[176] Shai Shalev-Shwartz, Alon Gonen, and Ohad Shamir. Large-scale convex minimization with a low-rank constraint. *arXiv preprint arXiv:1106.1622*, 2011.

[177] Yuan Shen, Zaiwen Wen, and Yin Zhang. Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software*, 29(2):239–263, 2014.

[178] Somayeh Sojoudi and Javad Lavaei. Exactness of semidefinite relaxations for nonlinear optimization problems with underlying graph structure. *SIAM Journal on Optimization*, 24(4):1746–1778, 2014.

[179] K. Spindler. New methods in on-board attitude control. *Advanced in the Astronautical Sciences*, 100(2):111–124, 1998.

[180] B. Stephen and M. Jacob. Branch and bound methods, 2003.

[181] Jos F Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999.

[182] C. Sun and R. Dai. An iterative approach to rank minimization problems. In *Decision and Control, IEEE 54st Annual Conference on*, 2015.

[183] C. Sun and R. Dai. An iterative approach to rank minimization problems. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 3317–3323, 2015.

[184] Chuangchuang Sun and Ran Dai. Identification of network topology via quadratic optimization. In *American Control Conference*, July 2015.

[185] Chuangchuang Sun and Ran Dai. Identification of network topology via quadratic optimization. In *American Control Conference (ACC), 2015*, pages 5752–5757. IEEE, 2015.

[186] Chuangchuang Sun and Ran Dai. An iterative approach to rank minimization problems. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 3317–3323. IEEE, 2015.

[187] Chuangchuang Sun and Ran Dai. Spacecraft attitude control under constrained zones via quadratically constrained quadratic programming. In *AIAA Guidance, Navigation, and Control Conference*, page 2010, 2015.

[188] Chuangchuang Sun and Ran Dai. An iterative method for nonconvex quadratically constrained quadratic programs. *arXiv preprint arXiv:1609.02609*, 2016.

[189] Chuangchuang Sun and Ran Dai. A customized admm for rank-constrained optimization problems with approximate formulations. In *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*, pages 3769–3774. IEEE, 2017.

[190] Chuangchuang Sun and Ran Dai. A decomposition method for nonconvex quadratically constrained quadratic programs. In *American Control Conference (ACC), 2017*, pages 4631–4636. IEEE, 2017.

[191] Chuangchuang Sun and Ran Dai. Rank-constrained optimization and its applications. *Automatica*, 82:128–136, 2017.

[192] Chuangchuang Sun and Ran Dai. Topology design and identification for dynamic networks. *Cooperative Control of Multi-Agent Systems: Theory and Applications*, page 209, 2017.

[193] Chuangchuang Sun, Ran Dai, and Ping Lu. Solving polynomial optimal control problems via iterative convex optimization. In *AIAA Guidance, Navigation, and Control Conference*, page 0371, 2016.

[194] Chuangchuang Sun, Ran Dai, and Mehran Mesbahi. Weighted network design with cardinality constraints via alternating direction method of multipliers. *IEEE Transactions on Control of Network Systems*, 2018.

[195] Chuangchuang Sun, Yen-Chen Liu, Ran Dai, and David Grymin. Two approaches for path planning of unmanned aerial vehicles with avoidance zones. *Journal of Guidance, Control, and Dynamics*, pages 1–8, 2017.

[196] Chuangchuang Sun, Yifan Sun, and Ran Dai. Admm for combinatorial graph problems. *arXiv preprint arXiv:1805.10678*, 2018.

[197] Mursel Tasgin, Amac Herdagdelen, and Haluk Bingol. Community detection in complex networks using genetic algorithms. *arXiv preprint arXiv:0711.0491*, 2007.

[198] J. M. Ten Berge and H. A. Kiers. A numerical approach to the approximate and the exact minimum rank of a covariance matrix. *Psychometrika*, 56(2):309–315, 1991.

[199] M. P. Tsiotras, M. Corless, and J. M. Longuski. Novel approach to the attitude control of axi-symmetric spacecraft. *Automatica*, 31(8):1099–1112, 1995.

[200] Madeleine Udell, Corinne Horn, Reza Zadeh, Stephen Boyd, et al. Generalized low rank models. *Foundations and Trends® in Machine Learning*, 9(1):1–118, 2016.

[201] Gabriel Urrutia, Ramón A Delgado, and Juan C Agüero. Low-order control design using a novel rank-constrained optimization approach. In *Australian Control Conference, 2016*, pages 38–42.

[202] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

[203] B. Vandereycken. *Riemannian and multilevel optimization for rank-constrained matrix problems*. PhD thesis, Katholieke Universiteit Leuven, 2010.

[204] Mats Viberg. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 31(12):1835–1851, 1995.

[205] Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of ADMM in nonconvex nonsmooth optimization. *arXiv preprint arXiv:1511.06324*, 2015.

[206] B. Wie and P.M. Barba. Quaternion feedback for spacecraft large angle maneuvers. *AIAA Journal of Guidance, Control and Dynamics*, 8(3):360–365, 1985.

[207] J. C. Willems. Paradigms and puzzles in the theory of dynamical systems. *IEEE Transactions on automatic control*, 36(3):259–294, 1991.

[208] Maria Woodside-Oriakhi, C Lucas, and John E Beasley. Heuristic algorithms for the cardinality constrained efficient frontier. *European Journal of Operational Research*, 213(3):538–550, 2011.

[209] Zhengping Wu and Renming Wang. The Consensus in Multi-Agent System with Speed-Optimized Network. *International Journal of Modern Physics B*, 23(10):2339–2348, 2009.

[210] Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004.

[211] Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46, 2007.

[212] Gang Xu, Laura Bennett, Lazaros G Papageorgiou, and Sophia Tsoka. Module detection in complex networks using integer optimisation. *Algorithms for Molecular Biology*, 5(1):36, 2010.

[213] Yangyang Xu, Wotao Yin, Zaiwen Wen, and Yin Zhang. An alternating direction algorithm for matrix completion with nonnegative factors. *Frontiers of Mathematics in China*, 7(2):365–384, 2012.

[214] Wotao Yin, Stanley Osher, Donald Goldfarb, and Jerome Darbon. Bregman iterative algorithms for l_1-minimization with applications to compressed sensing. *SIAM Journal on Imaging sciences*, 1(1):143–168, 2008.

[215] M. M. Zavlanos and G. J. Pappas. Potential fields for maintaining connectivity of mobile networks. *IEEE Transactions on Robotics*, 23(4):812–816, 2007.

[216] Fuzhen Zhang. *Matrix theory: basic results and techniques*. Springer Science & Business Media, 2011.

[217] Yun-Bin Zhao. An approximation theory of matrix rank minimization and its application to quadratic equations. *Linear Algebra and its Applications*, 437(1):77–93, 2012.

[218] Yue Zu, Chuangchuang Sun, and Ran Dai. Distributed estimation for spatial rigid motion based on dual quaternions. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 334–339. IEEE, 2014.