System Identification of Linear and Non-Linear Dynamical Systems using Semi-definite Programming via Penalized Parabolic Relaxation Method

by

ADNAN NASIR

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2022

To my mother for her unconditional love and prayers

ACKNOWLEDGEMENTS

I would like to acknowledge the utmost help and sheer dedication of my PhD supervisor Dr. Ramtin Madani, without his continuous support and selfless commitment, this work may not be possible. Dr. Ramtin spent numerous hours helping me with the research and formulation of the problem. He was also meticulous in giving motivation and morally and ethically supporting me whenever I was down in this long journey.

I want to thank the committee members for taking time to serve in my dissertation committee, most of them were my mentors during my tenure as a PhD student. Dr. Ioannis Schizas and Dr. Ali Davoudi deserve my heartfelt appreciation and gratefulness for their valuable time and advises all along my journey.

I want to extend my appreciation to IIE and Fulbright for getting behind me financially and supporting my goal of achieving a PhD. Ms Courtney Castillo and Mr Drew Crawford were instrumental as my program advisors in helping me out throughout my five years here as a Fulbright PhD scholar.

I am grateful to my colleagues Edward Quarm Jnr, Muhammad Adil, Mohsen Kheirandeshfard and Fariba Zohrizadeh for their valuable discussions and help. I like to mention here the help and support I had from my friends Imran Syed, Taha Saeed, Naveed Muzaffar, Fahad Ahmed and Ali Azfar, who were always there for me in difficult times and made my stay here wonderful.

Finally. I would also like to acknowledge the sacrifices of my family and parents. Especially my mother who is the main source of love and support for me since my childhood.

All my achievements and honors are due to her sacrifices. I also like to thank my wife who has been patience and encouraged me to carry-on.

August 05, 2022

ABSTRACT

System Identification of Linear and Non-Linear Dynamical Systems using Semi-definite Programming via Penalized Parabolic Relaxation Method

Adnan Nasir, Ph.D.

The University of Texas at Arlington, 2022

Supervising Professor: Ramtin Madani

This document is prepared to highlight the difference between two very important fields in the realm of Convex Optimization, Machine Learning, Artificial Intelligence and many such sprouting advance fields currently impacting the whole world. The main aim of the recent SDP algorithms is to relax the NP-Hard Convex Optimization problems and transform them into amenable jobs without strong theoretical guarantees, but these algorithms tends to approximate the practical in demand applications in many of the aforementioned fields. Moreover, The lack of theoretical guarantee makes the SDP relaxation in-exact and unoptimizable in finite duration. The actual world applications most of the time desire real time optimized results, for example, in applications related to stock exchange, critical space explorations and launches, pattern and feature detection, in cyber security and many more.

Many efforts have been made to develop algorithms that can relax the SDPs, and in the same time can reach to a near optimal solution without wasting a lot of time. Many such efforts from the past and including some recently proposed algorithms are discussed in this survey, highlighting important methods used by these prominent researchers such as low rank First and Second order relaxations, with and without sparsity. Due to the demand of making these algorithms work fast on huge data sets, it is increasingly difficult to converge to optimality for any of the contemporary SDP algorithms in finite time.

We have proposed a novel SDP algorithm, that utilizes Sequential Parabolic Relaxation to further relax the non-convex constraints. The proposed algorithm not only optimizes the SDP in hand, reaching the optimal solution in a finite amount of time, it also gives a theoretical guarantee of accuracy. We have discussed the proposed algorithm emphasizing the analytical and experimental results on the System Identification and Control, and comparing its performance with state-of-the-art algorithms.

After explaining the novel convex relaxation to solve the equations governing the behavior of linear and nonlinear dynamical systems. The proposed relaxation is used for the purpose of system identification. We demonstrate significant improvements in the overall scalability in comparison with the state-of-the-art convex relaxations.For cases where the proposed relaxation is inexact, a sequential algorithm is provided which converges within a finite number of rounds. Theoretical guarantees and simulation results are presented to demonstrate the effectiveness of the proposed approach. This review will also state the mathematical proof ensuring the convergence of the algorithm in finite time. At the end, we tried to pin-point new directions and the challenges that are still ahead in this vast field of SDP optimization.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction to Semi-definite Programming

Semidefinite Programming (SDP) has been around for decades, however, their progress was not impressive initially because of the algorithms used in SDPs were slow and required considerable amount of memory space, especially for large scale optimization problems.
SDPs plays a pivotal role in solving most of the NP-Hard problems, that were considered unsolvable in the past. Thanks to a number of prominent researchers and their achievements in the field of SDPs we were able to solve these problems in efficient ways. This section sheds light on some of their major efforts.
Burer et.al [3–5] are some of initial attempts to develop a practical algorithm which may not offer strong theoretical guarantee but is able to relax the original problem to solve the SDPs. Recent advances mainly adopting the Burer's methodology, in this field has introduced new algorithms for First and Second order SDPs and has instigated plethora of opportunities in various fields of science, engineering, banking and numerous technologies. Figure 1.1 shows the classes of Convex Optimization.

Before indulging into the vast and intricate sub-class of Convex Optimization called SDPs, we need to investigate its origin. SDPs are a component of a bigger Convex Optimization Theory, which it-self is a field of Mathematical Optimization.

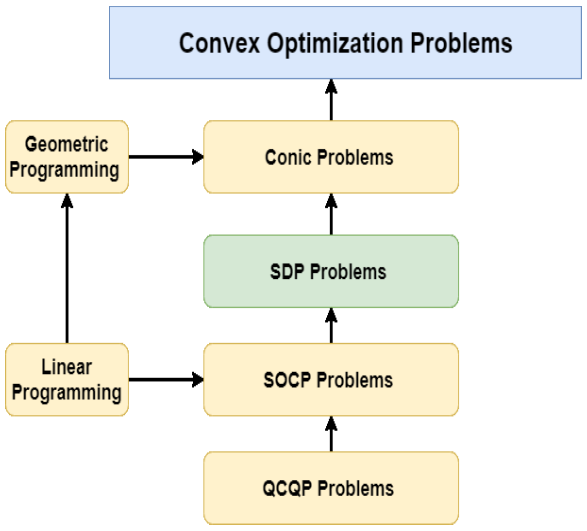

Figure 1.1. Classes of Convex Optimization.

A general Convex Optimization problem can be represented as:

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \le b_i, i = 1, 2, ..., m. \\ & f_i = b_i \end{aligned} \tag{1.1}$$

Where in equation(1.1), $f_0(x) : \mathbb{R}^n \mapsto \mathbb{R}$ is an objective function and $f_i(x) : \mathbb{R}^n \mapsto \mathbb{R}$, are constraint functions from $i = 1, 2, ..., m$. A convex Objective and Constraint functions makes the problem Convex. $f_i(x) \le b_i$ and $f_i = b_i$ are inequality and equality constraints respectively. Mathematical Optimization generally is considered as NP-hard.

SDP is quite a new addition to the convex optimization in retrospect and efficiently solves very complex optimization problems. Recent advances and mathematical breakthroughs has made it one of the most persuaded technique of current convex optimization problems. Some of the fields but not limited to Machine Learning, Artificial Intelligence, Operational Research, Combinatorial Optimization, Power, Communications, Sensing, Signal Processing etc. are utilizing the amazing power of SDP modelling.

SDPs are considered as a special case of Cone Programming, where we choose a symmetric positive semidefinite matrix to minimize a linear function subject to linear constraints.interior point methods conveniently solve the problems related to cone programming. Moreover, almost all the linear and polynomial optimization problems can be re-modelled into SDPs. The following is the Standard and inequality form semidefinite programs [6]:

$$\begin{aligned} \min_{x} \quad & \mathbf{tr(CX)} \\ \text{s.t.} \quad & \mathbf{tr(A_i X)} = \mathbf{b_i}, i = 1, 2, ..., p. \\ & \mathbf{X} \succeq \mathbf{0}, \end{aligned} \tag{1.2}$$

SDPs have linear equality constraints and a non-negativity matrix constraint on $\mathbf{X} \in \mathbf{S^n}$. Here in 1.2, C, and $A_1, ..., A_p \in \mathbf{S^n}$ [6].

By definition any algorithm that needs at minimum one derivative or gradient to optimize the problem is termed as First Order, on the same note a Second order algorithm utilizes second gradient, since, the second order methods mainly depends upon the use of Newtons steps, that require the use of Hessian which has second order derivative. Next section discusses briefly some of the recent work on first and second order methods in solving SDPs.

1.1 First-Order SDPs

First order Non-linear Programming methods are primarily motivated by Interior Point Algorithm for linear programming,using primal-dual and Lagrangian Multipliers. Conventional SDP algorithms were great in solving low dimension data problems, however, when the dimensions of the problem becomes large it was impossible to handle by these old algorithms due to huge space requirements for data storage and extraordinary time they take to produce feasible solutions.

In the SDP arena the two general first order methods has been in operation for long are **Projected sub-gradient method** and the **Method of multipliers or Augmented Lagrangian Method**.

Some recent work has made it possible to solve the large SDP problems by exploiting the sparse structure of the original problem, they can also be efficiently converged using simple numerical matrix algebra methods. Madani [7] elaborated the first order SDPs in detail and their projected applications. An excellent initial work on SDP by Vandenberghe and Boyd in 1994 [8] elaborate the topic very well.

1.2 Second-Order SDPs

Second order methods are based on Newton steps hence, referred to be second order methods. They are preconditioned conjugate gradient schemes to compute Newton direction and rely on Matrix completion to exploit sparsity of the data.

Second-order cone programming (SOCP) problems also called the Lorentz cone or the ice-cream cone, is a special case of SDPs. SOCPs are defines as a linear function which is minimized over the intersection of an affine linear manifold with the Cartesian product of second-order (Lorentz) cones. It is also noted that LPs, QPs and QCQPs can all be treated with SOCP formulations, making SOCP to lie between LP, QP and SDP [9]. However, when SOCP was formulated as SDP it increases the complexity and the size of the matrices, hence, in general this transformation is not recommended [9].

The standard form of SOCP is similar to linear program (LP) is given below:

$$\begin{aligned} \min_{X} \quad & \mathbf{f^T X} \\ \text{s.t.} \quad & \|\mathbf{A_k X + b}\|_\mathbf{2} \leq \mathbf{c_k^T X + d_k}, \quad k = 1, ...m. \end{aligned} \tag{1.3}$$

Where, $A_k \in \mathbb{R}^{n \times n}$, $b_k \in \mathbb{R}^n$, $c_k \in \mathbb{R}^n$, and $d_i \in \mathbb{R}$.

SOCPs can efficiently model the problems in the fields encompassing computer, electrical, mechanical engineering, automation and control, Machine and Deep Learning and finance [10–12].

1.3 Special Case of SDP - QCQP

In quadratic program (QP) the objective is quadratic with linear constraints, however, in quadratically constrained quadratic program (QCQP) has quadratic objective along with quadratic constraints. QCQP is thoroughly studied during the past decade and is widely considered as a special case of SOCP, it is used globally to solve problems in communication systems, signal processing and many other engineering areas [13].
As it has been seen that the QCQP is a non-convex NP-Hard problem, in order to solve these problems efficiently, Semidefinite Relaxation (SDR) techniques were employed. Generally this relaxation is obtained by reformulating the problem into a rank-one constrained optimization problem, afterwards the problem is relaxed into a convex SDP while ignoring the rank constraints. This relaxation is not 'tight', after utilizing some sort of randomization algorithm, hence, the solution is sub-optimal [14].

A general Quadratic Programming standard form is given below:

$$\begin{aligned} \min_{X \in \mathbb{S}_{\Bbbk}} \quad & \mathbf{X^T Q X + q^T X + c} \\ \text{s.t.} \quad & \mathbf{A_k X \leq b}, \end{aligned} \tag{1.4}$$

Where, $Q \in \mathbb{S}^{n \times n}$, $q \in \mathbb{R}^n$, $c \in \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$.

QCQP standard form is given below:

$$\begin{aligned} \min_{X \in \mathbb{S}_{\Bbbk}} \quad & \mathbf{X^T Q X + q^T X + c} \\ \text{s.t.} \quad & \mathbf{X^T Q_i X + q_i^T X + c_i \leq 0}, \quad i = 1, ..., m. \end{aligned} \tag{1.5}$$

Here, $Q, Q_i \in \mathbb{S}^{n \times n}$, $q, q_i \in \mathbb{R}^n$, $c, c_i \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$.

1.4 Commercial Solvers

In the past, optimization problems were divided in two classes:

- Linear programming problems that are convex and non-smooth, and were solved by Simplex methods.

- Nonlinear programming problems that are smooth, can be convex and were solved by gradient local search methods.

From 1980's due to the origination of Interior Point Methods, SOCP and SDPs, this division was redefined. Now the optimization problems were categorized into:

- Convex optimization problems, which can be solved efficiently.
- Non Convex optimization problems which cannot be solved efficiently.

However, many such non-convex optimization problems can be relaxed to behave in convex manner and these relaxations can be used to compute the bounds on the non-convex optimization. Convex Optimization is now thoroughly used in both academia and industry, many applications need solvers to get to the optimal solutions quickly. To facilitate the industry and academia many software solvers has been produced in the recent past. Many of these solvers are designed to deal with some specific problems and type of Convex Optimization under consideration.

Choosing a solver depends upon some of the following traits that fits the need of the problem at hand. First parameter is accuracy, when the problem needs to be solved accurately, it usually takes more time to reach the feasible optimal solution. Second is robustness, if the desired optimal solution needs to be reliable where the tolerance for an error is very low. Third parameter is the ability to be interfaced, it essentially means, whether the solver is compatible with the system you are using. Finally, the performance, it includes the time it will take to finish the job and how good the solver is in parallel multitasking.

A list of some of the most used commercial and free/open source solvers is presented below, this list is not comprehensive.

Solvers	**Solving Method**	**Applications**
	Open Source	
SDPA	Primal-dual interior-point method, exploits sparsity, MATLAB, C and Python platforms	SDP
SDPT3	MATLAB based, semidefinite-quadratic-linear programming, predictor-corrector primal-dual path-following method.	SDP, SOCP
CLPEX	Integer, linear and quadratic programming	LP, MILP, QP, QCQP, SOCP
SeDuMi	Primal-dual interior point methods	SDP, SOCP
Gurobi	integer, linear and quadratic programming	LP, QP, QCQP, SOCP
CVX	Matlab software for Disciplined Convex Programming based on interior point methods.	SDP, SOCP, GP, MIDCP
LOGDETPPA	MATLAB based, log-determinant semidefinite programming, semi-smooth Newton CG primal proximal point algorithm.	SDP
PENLAB	MATLAB based optimizer for nonlinear, linear SDPs. used for teaching and research purposes, based on nonlinear re-scaling method of Roman Polyak [15].	SDP
SDPLR	C-base package for solving large-scale SDPs, Runs on MATLAB, uses Burer's Method [3].	SDP
SDPNAL	MATLAB based solver for SDP's with bound constraints. Uses primal and dual methods based on, Newton-CG augmented Lagrangian, [16].	SDP
CSDP	MATLAB platform, predictor-corrector version of Primal-Dual method based on Helmberg method [17].	SDP
DSDP	MATLAB platform, Interior-point method for semidefinite programming, exploits low-rank and sparsity.	SDP, SOCP
ROME	MATLAB platform, Interior-point method for semidefinite programming, exploits low-rank and sparsity.	SDP
YALMIP	MATLAB platform, Interior-point method for semidefinite programming, exploits low-rank and sparsity.	SDP

Table 1.1. Free/Open Source Solvers, [1, 2]

Solvers	Solving Method	Applications
Commercial		
PENBMI	Quadratic objective and Bilinear matrix inequality constraints, for both dense and sparse matrices, control and optimization. MATLAB based. Exterior penalty and Interior barrier methods with the Augmented Lagrangian method.	SDP
PENSDP	Solves large-scale problems with sparse data structure. It is based on a generalized augmented Lagrangian method [18].	SDP
SCIP-SDP	Solves mixed integer SDPs, combines branch-and-bound framework of SCIP [19], with interior-point methods [20].	Mixed Integer SDP
LMILAB	MATLAB tool box, solves general Linear Matrix Inequalities (LMI) problems, based on MATLAB Executable C-MEX implementation.	LMI, SDP
CPLEX	MATLAB and Python based, solves very large, real-world optimization problems, not used for SDP problems, made on C++, Java and .Net platforms, uses many algorithms like primal, dual, Barrier(Interior Point) and network SDPs can be re-formulated into CPLEX structure.	LP, QP, QCQP, MIP
GUROBI	Uses on multiple platforms MATLAB, C++, Java, R etc. utilizes multiple algorithms, Primal, Dual, Barrier, Simplex, branch and cut, cutting planes etc.	LP, QP, MIP, QCP, SOCP, MIQCP, SDP
GAMS	Uses multiple solvers for linear, non-linear, mixed integer, complex, large scale modeling applications.	LP, MIP, QCP, MIQCP, SDP
PENNON	MATLAB based, Used for solving non-linear SDPs, based on generalized augmented Lagrangian method [18], free open source toolbox for MATLAB is available	SDP
MATLAB	linear, integer, quadratic, and nonlinear problems with Optimization Toolbox	LP, MIDCP, QP
MOSEK	MATLAB, R and Python based, solves large-scale, optimization problems. Primal and dual method and Branch, Bound and Cut algorithm, for Mixed Integer Optimization, free student version is available.	SDP, QP, QCQP, SOCP

Table 1.2. Commercial Solvers, [1,2]

CHAPTER 2

Burer's SDP Method

Interior Point methods are used to solve the SDP's in polynomial time, however, scalability is the main problem. Burer in 2002 [3] addresses this problem by solving the SDPs using some low-rank equality constraints and non-convex alternates. Burer's method was instrumental in converging to the global optima in case of some application and gives an exact solution using the block-coordinate maximization method. Burer presents a nonlinear programming algorithm for solving SDPs in standard form by change of variables X which is symmetric, positive semidefinite to a rectangular variable Y by factorization $X = YY^T$. Rank of Y is chosen minimally to improve speed. However, relaxing the SDP by assuming $X = YY^T$ makes the system non-convex, since Y is non-convex [4].

Consider the following SDP in standard form:

$$\begin{aligned} \min_{X \in \mathbb{S}_{\ltimes}} \quad & \mathbf{tr}\{\mathbf{A_0 X}\} \\ \text{s.t.} \quad & \mathbf{tr}\{\mathbf{A_k X}\} \leq \mathbf{a_k}, k = 1, 2, ..., m. \\ & \mathbf{X} \succeq \mathbf{0}, \end{aligned} \tag{2.1}$$

Here, the $n \times n$ matrices, $A_1, ..., A_m \in \mathbb{S}_{\ltimes}$ and $a_1, ..., a_m$ are given real numbers.

The factorizing method proposed by Burer at [3] and [4] uses a matrix Y of size $(n \times k)$ such that $X = YY^T$ is the SDP variable X. This formulation shown in equation (2.2) below, readily gives two main advantages over the main SDP problem shown in equation 1.2. First, it reduces the dimension of the optimizing variable and secondly, in the process, imposed the condition of positive semi-definiteness on it.

$$
\begin{aligned}
\min_{\substack{X\in\mathbb{S}_{\ltimes}\\ Y\in\mathbb{R}^{\ltimes\times\diagdown}}} \quad & \mathbf{tr}\{\mathbf{A_0X}\} \\
\text{s.t.} \quad & \mathbf{tr}\{\mathbf{A_kX}\} \le \mathbf{a_k}, \quad k = 1, 2, ..., m. \\
& \mathbf{X} = \mathbf{YY^T},
\end{aligned}
\tag{2.2}
$$

2.1 Recent Work on Burer's Method

Here, in this section we are going to analyze some of the recent and pioneer notable work on Burer's method in the period after its inception. We start from the earliest work and move towards the most recent ones.

Burer's methods uses non-convex local algorithms to solve large scale SDPs with low rank solution [3] through change of variables, it was instrumental in giving a relaxed solution for SDP problems and after that, many researchers caught on the idea and started to exploit the benefits of this relaxation. It was helpful to solve problems in many areas in combinatorics, Machine Learning, Power Flow and the list goes on. Here Burer, present a primal-dual path-following algorithm that is based on a new search direction, which, roughly speaking, is defined completely within the space of partial symmetric matrices. This work is in contrast to the work of Fukuda et al. [21] and Nakata et al. [22], in which the theory of partial positive semidefinite matrices was applied to the semidefinite programming (SDP) problem as a technique for exploiting sparsity in the data by computing primal-dual solution of the SDP problem with a duality gap less than a fraction $\epsilon > 0$ of the initial duality gap in $O(nlog(\epsilon - 1))$ iterations, where n is the size of the matrices involved.

In [23], the authors used slack variables to verify the Burer's method, however, they found that the slack variable approach will not be able to increase the efficiency of the solution but they will improve the doubly non-negative relaxation. They determined that a similar improvement on the solution is obtained by introducing some linear inequality constraints in the SDP problem. There were many first order attempts made to solve the SDP earlier on, however, Helmberg [17], is one of the first to try and solve the SDPs thorough Second Order method. His proposed method utilizing semi-definite matrices based on interior point method is crucial and become a pivotal method used by recent studies. Another one of the first to attempt second order method was Farid's [24] take on Newton's method based, Primal-dual interior-point path-following methods for semidefinite programming. However, both these papers doesn't employ sparsity by introducing a low rank solution.

Among some of the efforts made to exploit the sparsity of the a large scale SDP matrix was an attempt by Fakuda and Nakata [21, 22] in two parts. The method used by them exploits the fundamentals of positive semidefinite matrix completion. The proposed method works in two ways, first is to convert the sparse SDP of large scale positive semidefinite matrix into multiple smaller positive semidefinite matrices, this way the interior point method can be effectively applied to get the solution. Secondly, this method can be utilized directly using primal-dual interior point methods on a certain SDP. Nakata [22] also proposed clique tree approach to exploit the sparsity of the SDPs. Burer in [5], based his work on [21] and [22], to implement the theory of partial positive semidefinite matrices onto the SDP problem in order to exploit the sparsity. His work tend to improve an existing standard search direction algorithm. Burer, in 2006 [25], proposed an algorithm for optimizing the lift and project relaxations of binary integer programs introduce by Lovasz and Schrijver [26]. This papers analyzed both linear and semidefinite relaxations. The key idea is to restructure the relaxations, which isolates the complicating constraints and allows for the Lagrangian approach. The proposed algorithm is based on the enhanced sub-gradient with augmented Lagrangian. The problem was solved using the IP solver from CPLEX [27].

A first order method was proposed by Wen et al. [28], without exploiting the sparsity of the data, this algorithm uses an alternating direction dual augmented Lagrangian method (ADMM) for solving SDP problems in standard form. This method become one of the most important algorithms in solving SDPs. It utilizes sequential minimization of the augmented Lagrangian function for the dual SDP problem. The minimization was done first with respect to the dual variables corresponding to the linear constraints,and secondly, with respect to the duals slack variables. In the end the dual solution is used to update the Lagrange multipliers or the primal variables.

Masakazu [29] in order to solve large scale linear and nonlinear SDP problems, using chordal graph structure, exploits the domain space sparsity (d-space sparsity) for positive semidefinite symmetric matrix variables, and the range space sparsity (r-space sparsity) for matrix-inequality constraints in SDP problems. Sunyoung [30] also exploits sparsity through Chordal graphs, using positive semidefinite matrix completion with linear and nonlinear matrix inequalities. They also used d-space sparsity and the r-space sparsity relaxations to efficiently solve the SDPs. Sparse semidefinite programming with a chordal sparsity pattern were utilized by Yifan-Vandenberghe [31] by proposing a similar second order method based on clique decomposition theorems and Spingarn's method for equality constrained convex optimization [32].

SDP is generally NP hard, however, if there is graph sparsity, it makes the solution more efficient by guaranteeing an existence of a low-rank solution. However, we may not always have graph sparsity. If a low rank solution exists, Burer's method can be utilized to re-formulate the problem, Parrilo Parrilo-2003 proves that Burer's algorithm converges fast while analyzing systems and control problems. Chordal Graphs are very important tool in exploiting the sparsity of a large semidefinite optimization problems. Graph theory has been around since 1960's to examine sparse matrices using Cholesky factorization [33]. Vandenberghe et al. [34] proved that in the presence of graph sparsity, one can prove that low-rank solutions exist. and the graph sparsity can be exploited to solve the problem more efficiently. Chordal graphs technique is one the main methods for exploiting sparsity in large semidefinite and Convex optimization problems which have sparse positive semidefinite matrices. Chordal graph method is also used in determining the partial inverse of a sparse positive definite matrix, solving gradients and Hessians of logarithmic barriers for cones and dual cones of sparse positive semidefinite matrices and many more.

In another work by Lieder, [35], the author observes that all constraints of the relaxation associated with linear constraints of the original problem can be accumulated in a single linear constraint without changing the feasible set of either the completely positive or the semidefinite approximation.The work focuses on analysis of completely positive and semidefinite relaxations of quadratic programs with linear constraints and binary variables as presented by Burer [4].

Murat et al. [36] offers theoretical insights into Burer's [3] algorithm, it deals with the general lack of scalability of convex optimization solvers with the large dimensions of the problem. Murat analyzed the Burer and Monteiro's method in [4] which reduces the dimension of the problem by low-rank factorization, while solving this reformulated non-convex problem instead. This paper determines the theoretical convergence of the Burer's method using block-coordinate maximization algorithm which was not presented before.

Efforts were also being made to reduce the complexity of the conventional numerical algorithms for SDPs. One such attempt is by Madani in [37], utilized a first order method with sparsity to address the in-ability of the conventional convexification methods to efficiently solve large-scale SDPs. In this paper a low-complexity numerical algorithm was proposed which is a general-purpose parallelizable SDP solver for sparse SDPs, utilizing ADMMs and tree decomposition in graph theory. The performance of the proposed algorithm is tested on the optimal power flow problem for more than 13,600 nodes. Another such work by Richard [38], presents theoretical analysis of second order method with sparsity. Exploiting the sparsity of combinatorial SDP relaxations, this paper proposed dualized clique tree method to reduce complexity and consequently utilize any interior-point method to solve a large scale sparse SDP, with a guaranteed complexity of $O(n1.5L)$ time and $O(n)$ memory.

In a recent paper, [39], Pumir et all, tries to raise a question of approximating Optimal Second Order Optimal Points (SOSPs). They used some assumptions and shown that the approximate SOSP's for a randomly relaxed objective function can be approximate global optima and used this method to solve phase retrieval application. They solved equality-constrained SDPs with low-rank using following assumptions, First, the search space of the SDP is compact and second, the search space of its low-rank version is smooth. Under these assumptions, the author proved using smoothed analysis and SDP relaxations that, if the cost matrix is perturbed randomly, with high probability, approximate second-order stationary points of the perturbed low-rank problem is mapped to approximately optimal solutions of the perturbed SDP. Yang et.al [40], solves a minimization of a quadratic function with non-convex quadratic constraints using additional linear intersecting interiors ellipsoids or Convex Hull constraints. They used mixed-integer quadratic programming to solve this SDP problem.

2.2 SDP Applications

SDPs with equality constraints arise in many optimization and machine learning problems, such as Max-Cut, community detection and robust PCA. SDP covers a wide range of applications such as robust optimization, polynomial optimization, combinatorial optimization, system and control theory, financial engineering, machine learning, quantum information, electrical Power and quantum chemistry. In [41] three types of structures in SDP were discussed, a common chordal sparsity pattern that has been used in graph theory, Low rank methods used in combinatorial optimization and polynomial optimization problems. Thirdly, low dimensional invariant matrices used in truss topology optimization, particle physics, coding theory, computational geometry, and graph theory.

CHAPTER 3

Low Rank SDP via Parabolic Relaxation

Searching for a low-rank matrix within a convex set can solve many different types of non-linear optimization problems. Where, SDP relaxation is just the method of re-modeling the QCQP problems into low rank matrix optimization problems [7]. It should also be noted that, SDPs can efficiently solve in polynomial time, small to medium proportional programs using second-order interior point methods [8], but large scale SDPs being computational hard and memory hungry, cannot be handled by the contemporary algorithms. We consider the standard SDP problem presented in equation (2.1), the aim of the problem is to determine the $n \times n$ Symmetric Positive Definite Matrix $\mathbf{X}$ that minimizes the objective subject to the above inequality constraints. The problem at (2.1) is very fundamental with applications in nearly all areas of engineering, sciences and many sectors of the economy. A wide variety of computationally-hard problems can be cast in the form of SDP in (2.1) and therefore, many researchers have devoted their careers to finding efficient algorithms for solving problem in (2.1).

3.1 Problem Formulation

The state-of-art algorithms require $O(n^{6.5})$ steps to solve the problem in 2.1, numerically. In the presence of sparsity (when the vast majority of $A_0, A_1...., A_m$ elements are zero, there are algorithms that can solve semidefinite programs much faster ([34]– [22]). Unfortunately, sparsity is a very strong assumption and does not necessarily hold in practice. In this work, we seek to develop a numerical method that can solve semidefinite programs much

faster even without sparsity. Our only assumption is that the final solution is low rank. Assume that the final solution to be found is of rank r. Hence, the problem in (2.1) can be equivalently formulated as follows:
Since the above formulation in the optimization at (2.2), is non-convex, we relax the constraint $X \succeq 0$ as follows:

$$\begin{aligned}
\min_{\substack{X\in\mathbb{S}_{\ltimes}\\ Y\in\mathbb{R}^{\ltimes\times\setminus}}} \quad & \mathbf{tr(A_0X)} \\
\text{s.t.} \quad & \mathbf{tr\{A_kX\} \leq a_k}, \quad k=1,2,...,m. \\
& \mathbf{X_{ii}+X_{jj}+2X_{ij} \geq \|Y_{i:}+Y_{j:}\|_2^2}, \quad i,j=1,2,...,n. \\
& \mathbf{X_{ii}+X_{jj}-2X_{ij} \geq \|Y_{i:}-Y_{j:}\|_2^2}, \quad i,j=1,2,...,n.,
\end{aligned} \tag{3.1}$$

We start with an initial guess $\hat{Y}$, and sequentially update it by solving the following penalized parabolic relaxation:

$$\begin{aligned}
\min_{\substack{X\in\mathbb{S}_{\ltimes}\\ Y\in\mathbb{R}^{\ltimes\times\setminus}}} \quad & \mathbf{tr\{A_0X\} + J \times tr\{X - 2Y\hat{Y}^T + \hat{Y}\hat{Y}^T\}} \\
\text{s.t.} \quad & \mathbf{tr\{A_kX\} \leq a_k}, \quad k=1,2,...,m. \\
& \mathbf{X_{ii}+X_{jj}+2X_{ij} \geq \|Y_{i:}+Y_{j:}\|_2^2}, \quad i,j=1,2,...,n. \\
& \mathbf{X_{ii}+X_{jj}-2X_{ij} \geq \|Y_{i:}-Y_{j:}\|_2^2}, \quad i,j=1,2,...,n.,
\end{aligned} \tag{3.2}$$

The above-mentioned is guaranteed to find the optimal solution of the SDP at (2.1). Solving each round of problem (3.1) is very easy because the above problem is a convex quadratically-constrained quadratic program (QCQP). By solving this easy problem for a few hundred times, we can solve the SDP optimization problem that we are interested in.

We have proposed a new algorithm based on Burer's formulation and we have proven that it converges within a FINITE number of rounds.

3.2 Methodology and Algorithm

We have opted for a novel approach based on sequential parabolic relaxation on the SDP at problem at (2.2). Our approach guarantees the optimal convergence in finite amount of time as compared to the infinite itterations required to converge for the problem after standard Burer's method.

Proposed Algorithm:

Algorithm 1 Sequential Parabolic Relaxation Algorithm.

```
Result: Exact and Finite-time Convergence of SDPs
initialization  while Optimality achieved do
    update the cost function and penalty term   update Y matrix  if Convergence achieved
     then
        Optimal Cost
    else
        Repeat the process  Update the Cost and Penalty terms
    end
end
```

CHAPTER 4

System Identification of Dynamical Systems

4.1 Introduction

System identification has been studied for decades with the aim of deriving mathematical models for dynamical systems through observation of input-output data, several books and technical papers has been written on this important topic [42,43], where salient features and parameters of a system, represented by a black box were estimated to determine and control the underline system. Today almost every field of application utilize system identification methods in one form or another. The importance of system identification in various industries can be evaluated through an astounding amount of research work done on this field [44].

Dynamical systems can be classified into linear and non-linear categories. Initial linear models' approaches utilizes statistical methods such as, maximum likelihood, Bayesian, cross validation, variance, gradient correction and least squares among many [45,46]. The advantages of using state-of-the art linear system identification include, dealing with a generalized error criterion function, and are suitable for online identification, however, they are not very good for complex and or non-linear systems, and they require a full knowledge of system's input signals and states [44].

More recent methods that can deal with both linear and non-linear systems have been proposed. They are mostly based on neural networks [47], genetic algorithm [48], swarm intelligence [49], auxiliary model identification algorithm [50], multi-innovation identification algorithm [51], hierarchical identification algorithm [52] and many more. These methods present benefits for certain type of systems, but cannot be generalized in order to be used on all sort of system identification problems. For instance, neural network-based algorithms do not need a model structure, can be used on linear and non-linear systems. However, they are prone to local minimums and convergence issue and they require massive amounts of training data. Similarly, Genetic Algorithms suffer from local minimum problems and premature convergence, and they cannot identify the structure and parameters of the systems under study [44]. Other methods, such as Swarm intelligence, Auxiliary model identification, Multi-innovation identification, and Hierarchical identification give a global solution without needing too much priori knowledge of the optimization problem, but are difficult to implement, slow and computationally hard.

Another technique recently used for system identification is Convex Optimization [6]. With the advancements in convex optimization theory, these principles are now readily adapted for system identification problems, [53] is one of the recent work that has found convex optimization methods useful.

In this work, we rely on convex optimization for identifying the parameters of linear dynamical systems and we then extend our results to nonlinear systems. Convex optimization is central to the design and analysis of dynamic systems. Recent advances in convex optimization techniques [6] have opened new doors for efficient system identification. Common-practice convex relaxations primarily rely on a process referred to as *lifting* which involves the introduction of new auxilary variables [54]. some of the examples include are the semidefinite programming (SDP) [5] and second-order cone programming (SOCP) relaxations [9], as well as the diagonally-dominant (DD) and scaled diagonally-dominant (SDD) approximations [55]. Despite their effectiveness, the main limitation of these algorithms is the curse of dimensionality caused by lifting.

In this paper, we propose a novel method named sequential parabolic relaxation, which deals efficiently with the problem of lifting. The primary strength of this approach is that it relies on a modest number of auxiliary variables, which leads to a higher level of efficiency compared to the exiting methods. We prove the efficacy of our proposed method by providing mathematical proof, that guarantees an optimal solution within a short period of time using only a sample of input and state information. After giving the theoretical guarantees, we formulate linear system (a state-space control system consisting of bi-linear terms) and non-linear system (in a form of controlled permanent magnet synchronous machine **PMSM**). These problems were relaxed using our penalized parabolic relaxation method. We also compare our results by using the state-of-the art linear system identification tools in common practice, most of them are not iterative and were unable to converge, hence, further cementing the efficiency of our method.

4.2 Notations and Terminology

Throughout the thesis, the vectors and matrices are respectively shown by lower-case bold letters and upper-case bold letters. Symbols $\mathbb{R}$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$, respectively, denote the set of real scalars, real vectors of size n, and real matrices of size $n \times m$. The set of real $n \times n$ symmetric matrices and positive semidefinite matrices are shown with $\mathbb{S}_n$ and $\mathbb{S}_n^+$, respectively. Notation $\mathbf{A} \succeq 0$ means $\mathbf{A}$ is positive-semidefinite ($\mathbf{A} \succ 0$ indicates positive definite). tr$\{.\}$ and $(.)^\top$ respectively denote the trace and transpose operators, and rank$\{.\}$ denotes the rank operator. The notation $\|.\|_p$ refers to either matrix norm or vector norm depending on the context and $|.|$ indicates the absolute value operator.

4.3 Problem Formulation

4.3.1 Identification of Linear Systems

Consider a linear dynamical system whose behavior is described via the equations:

$$\mathbf{x}[\mathsf{t}+1] = \mathbf{A}\mathbf{x}[\mathsf{t}] + \mathbf{B}\mathbf{u}[\mathsf{t}] \qquad \mathsf{t} \in \mathcal{T} \tag{4.1a}$$

$$\mathbf{y}[\mathsf{t}] = \mathbf{C}\mathbf{x}[\mathsf{t}] + \mathbf{D}\mathbf{u}[\mathsf{t}] \qquad \mathsf{t} \in \mathcal{T} \tag{4.1b}$$

where $\mathcal{T} \triangleq \{1, 2, \ldots, \tau\}$ is a discrete time horizon and $\{\mathbf{x}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}\cup\{\tau+1\}}$, $\{\mathbf{y}[\mathsf{t}] \in \mathbb{R}^m\}_{\mathsf{t}\in\mathcal{T}}$, and $\{\mathbf{u}[\mathsf{t}] \in \mathbb{R}^o\}_{\mathsf{t}\in\mathcal{T}}$ denote the state, observation, and system input vectors, respectively.

Assume that $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ are unknown as well as the state vectors at snapshots $\mathcal{T}' \triangleq \{\mathsf{t}_1, \mathsf{t}_2, \ldots, \mathsf{t}_{\tau'}\} \subseteq \mathcal{T} \cup \{\tau + 1\}$. We seek to determine the unknowns based on system observation, system input, and limited knowledge of state vectors at times $(\mathcal{T}\cup\{\tau+1\})\setminus\mathcal{T}'$. This problem can be formulated as follows:

$$\begin{aligned}
&\text{find} && \mathbf{A} \in \mathbb{R}^{n\times n},\ \mathbf{B} \in \mathbb{R}^{n\times o},\ \mathbf{C} \in \mathbb{R}^{m\times n},\ \mathbf{D} \in \mathbb{R}^{m\times o}, \\
& && \{\mathbf{x}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}'} && && \text{(4.2a)} \\
&\text{subject to} && \mathbf{x}[\mathsf{t}+1] = \mathbf{A}\mathbf{x}[\mathsf{t}] + \mathbf{B}\mathbf{u}[\mathsf{t}] && \mathsf{t} \in \mathcal{T} && \text{(4.2b)} \\
& && \mathbf{y}[\mathsf{t}] = \mathbf{C}\mathbf{x}[\mathsf{t}] + \mathbf{D}\mathbf{u}[\mathsf{t}] && \mathsf{t} \in \mathcal{T} && \text{(4.2c)}
\end{aligned}$$

Problem (4.2a) – (4.2c) is non-convex and computationally challenging, due to the bi-linear terms $\mathbf{A}\mathbf{x}[\mathsf{t}]$ and $\mathbf{C}\mathbf{x}[\mathsf{t}]$. One popular approach for tackling the non-convexity of dynamical system equations is convex relaxation or approximation [56]. Despite the effectiveness of local search methods [57, 58], convex relaxation can offer major benefits including reliability and efficiency in the parameter estimation [59, 60]. For instance, in [59], it is proven that the number of feasible connected components induced by dynamical system equations can grow exponentially leading the failure of local search methods. This paper, is primarily focused on boosting the scalability of convex relaxation. We introduce a computationally-efficient parabolic relaxation to tackle constraints of the form (4.2b) and (4.2c).

4.4 State-of-the-art Relaxations and Approximations

In this section, we cover two different forms of the common-practice semidefinite programming (SDP) and second-order cone programming (SOCP) relaxations [61–63] and show that each suffer from major drawbacks compared to the proposed method.

4.4.1 Vector Formulation

Let the vector sets $\{\boldsymbol{a}_k \in \mathbb{R}^n\}_{k=1}^n$, $\{\boldsymbol{b}_k \in \mathbb{R}^o\}_{k=1}^n$, $\{\boldsymbol{c}_k \in \mathbb{R}^n\}_{k=1}^m$, and $\{\boldsymbol{d}_k \in \mathbb{R}^o\}_{k=1}^m$ represent the columns of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$, respectively, i.e.,

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{bmatrix}^\top \qquad \mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_n \end{bmatrix}^\top \tag{4.3a}$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \dots & \mathbf{c}_m \end{bmatrix}^\top \qquad \mathbf{D} = \begin{bmatrix} \mathbf{d}_1 & \mathbf{d}_2 & \dots & \mathbf{d}_m \end{bmatrix}^\top \tag{4.3b}$$

To formulate the standard conic relaxations of the problem (4.2a – (4.2c), one can cast the identification problem with respect to the new vector and matrix variables:

$$\mathbf{h} \triangleq \big[\mathbf{a}_1^\top \;\; \mathbf{a}_2^\top \;\; \dots \;\; \mathbf{a}_n^\top \;\; \mathbf{c}_1^\top \;\; \mathbf{c}_2^\top \;\; \dots \;\; \mathbf{c}_n^\top \\ \qquad\qquad \mathbf{x}^\top[\mathsf{t}_1] \;\; \dots \;\; \mathbf{x}^\top[\mathsf{t}_{\tau'}]\big]^\top \in \mathbb{R}^{n(m+\tau)} \tag{4.4a}$$

$$\mathbf{H} \triangleq \mathbf{h}\mathbf{h}^\top \tag{4.4b}$$

using which the nonlinear constrains (4.2b) and (4.2c) can be cast linearly [64]. This approach is regarded as *lifting*, as we are transitioning to a higher-dimensional space with respect to

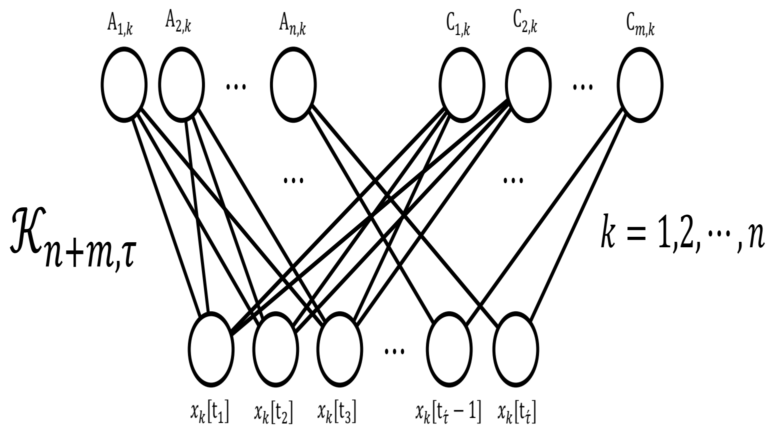

Figure 4.1. The bipartite sparsity pattern of the constraint.

the new matrix variable $\mathbf{H}$. The relations between the pair $(\mathbf{H}, \mathbf{h})$ can then be implicitly imposed as

$$\begin{bmatrix} \mathbf{H} & \mathbf{h} \\ \mathbf{h} & 1 \end{bmatrix} \in \mathcal{H} \tag{4.5}$$

where $\mathcal{H} \in \mathbb{S}_{n(m+\tau)}$ is an appropriate convex set depending on the choice of relaxation / approximation [64]. The main drawback of the aforementioned approach is the curse of dimensionality caused by lifting, even if the present sparsity of the problem is fully leveraged. Let $\mathcal{K}$ denote the sparsity graph of the quadratic constraints (4.2b) – (4.2c), whose every vertex corresponds to an element of the vector $\mathbf{h}$, and every edge corresponds a bilinear term:

$$\mathsf{A}_{ik}\,\mathsf{x}_k[\mathsf{t}] \qquad \forall (i,k,\mathsf{t}) \in \mathcal{N} \times \mathcal{N} \times \mathcal{T}' \tag{4.6a}$$

$$\mathsf{C}_{jk}\,\mathsf{x}_k[\mathsf{t}] \qquad \forall (j,k,\mathsf{t}) \in \mathcal{M} \times \mathcal{N} \times \mathcal{T}' \tag{4.6b}$$

adding up to a total of $n(n+m)\tau'$ edges, where $\mathcal{N}=\{1,\dots,n\}$ and $\mathcal{M}=\{1,\dots,m\}$. More precisely, each pair of elements in $\mathbf{h}$, are connected if and only if their product appears in either (4.2b) or (4.2c). As demonstrated in Figure 4.1, it can be easily observed that

$$\mathcal{K}=\underbrace{\mathcal{K}_{n+m,\tau'}\cup\mathcal{K}_{n+m,\tau'}\cup\cdots\cup\mathcal{K}_{n+m,\tau'}}_{n} \tag{4.7}$$

where $\mathcal{K}_{n+m,\tau'}$ denotes the complete bipartite graph with partitions of size $n+m$ and τ'. The complexity of solving cone programming relaxations is closely related to the sparsity graph of the problem [21, 22]. As shown in Table 4.1, in the case of SOCP relaxation for problem (4.2a) – (4.2c), one can leverage the sparsity of $\mathcal{K}$ by incorporating only those elements of $\mathbf{H}$ that correspond to existing edges amounting to $n(n+m)\tau'$ auxiliary variables.

The curse of dimensionality caused by lifting is further pronounced in the case of SDP relaxation that requires the incorporation of every element in $\mathbf{H}$ corresponding to the edges of an arbitrary chordal extension (Table 4.1). In the reminder of this section, we discuss an alternative conic relaxation for which a smaller number of variables are needed.

4.4.2 Matrix Formulation

Another approach for tackling problems of the form (4.2a) – (4.2c) through convex relaxation is the matrix formulation. To this end, constraints (4.2b) and (4.2c) can be cast in the following form:

$$\begin{bmatrix} \bar{\mathbf{A}} & \big[\mathbf{x}[\mathrm{t}_1{+}1]\dots\mathbf{x}[\mathrm{t}_{\tau'}{+}1]\big]-\mathbf{BU} & \mathbf{A} \\ * & \bar{\mathbf{X}} & \mathbf{X}^\top \\ * & * & \boldsymbol{I}_{n\times n} \end{bmatrix} \in \mathcal{C}_1 \tag{4.8a}$$

Table 4.1. Complexity of state-of-the-art convex relaxations versus parabolic relaxation.

	Max size of SDP constraints	**Number of SDP constraints**	**Number of new variables**
SDP relaxation of (4.4b)[*]	$(\max_{V\in\mathcal{V}}\lvert V\rvert)\times(\max_{V\in\mathcal{V}}\lvert V\rvert)$	$n\lvert\mathcal{V}\rvert$	$n(n+m)\tau'$
SOCP relaxation of (4.4b)	2×2	$n(n+m)\tau'$	$n(n+m)\tau'$
SDP relaxation of (4.8)	$(n+\max\{m,n\}+\tau')\times(n+\max\{m,n\}+\tau')$	2	$\binom{n+1}{2}+\binom{m+1}{2}+\binom{\tau'+1}{2}$
SOCP relaxation of (4.8)	2×2	$\binom{n+m+\tau'}{2}+\binom{2n+\tau'}{2}$	$\binom{n+1}{2}+\binom{m+1}{2}+\binom{\tau'+1}{2}$
Parabolic relaxation	Convex Quadratic	$2(n+m)\tau'+n+m+\tau'$	$n+m+\tau'$

[*] $(\mathcal{V},\mathcal{E})$ is an arbitrary tree decomposition of $\mathcal{K}_{n+m,\tau'}$

$$\begin{bmatrix} \bar{\mathbf{C}} & \left[\mathbf{y}[\mathrm{t}_1]\dots\mathbf{y}[\mathrm{t}_{\tau'}]\right]-\mathbf{DU} & \mathbf{C} \\ * & \bar{\mathbf{X}} & \mathbf{X}^\top \\ * & * & \boldsymbol{I}_{n\times n}. \end{bmatrix} \in \mathcal{C}_2 \tag{4.8b}$$

where $\mathbf{X} \triangleq \left[\mathbf{x}[\mathrm{t}_1]\dots\mathbf{x}[\mathrm{t}_{\tau'}]\right]$ and $\mathbf{U} = \left[\mathbf{u}[\mathrm{t}_1]\dots\mathbf{u}[\mathrm{t}_{\tau'}]\right]$, and $\bar{\mathbf{A}}$, $\bar{\mathbf{C}}$, and $\bar{\mathbf{X}}$ are auxiliary variables accounting for $\mathbf{AA}^\top$, $\mathbf{CC}^\top$ and $\mathbf{X}^\top\mathbf{X}$, respectively. Having

$$\mathcal{C}_1 = \{\boldsymbol{W}\in\mathbb{S}_{2n+\tau'} \mid \boldsymbol{W}\succeq 0, \quad \mathrm{rank}\{\boldsymbol{W}\}=n\} \tag{4.9a}$$

$$\mathcal{C}_2 = \{\boldsymbol{W}\in\mathbb{S}_{m+\tau'+n} \mid \boldsymbol{W}\succeq 0, \quad \mathrm{rank}\{\boldsymbol{W}\}=n\} \tag{4.9b}$$

makes the two formulations (4.2a) – (4.2c) and (4.8a) – (4.8b) equivalent. Additionally, various convex relaxations/approximation can be formulated via different choices for $\mathcal{C}_1 \subseteq \mathbb{S}_{2n+\tau'}$ and $\mathcal{C}_2 \subseteq \mathbb{S}_{m+\tau'+n}$, such as the common-practice semidefinite programming (SDP), second-order cone programming (SOCP) relaxations. In what follows, we pursue an alternative approach with far less complexity, involving convex quadratic constraints only.

4.5 System ID via Parabolic Relaxation

In order to formulate parabolic relaxation of the problem (4.2a) – (4.2c) we need to introduce a modest number of auxiliary variables:

- Define $\bar{\mathbf{a}} \in \mathbb{R}^n$ as the variable whose k-th element represents $\|\mathbf{a}_k\|_2^2$,
- Define $\bar{\mathbf{c}} \in \mathbb{R}^m$ as the variable whose k-th element represents $\|\mathbf{c}_k\|_2^2$,
- Define $\bar{\mathbf{x}} \in \mathbb{R}^{\tau'}$ as the variable whose k-th element represents $\|\mathbf{x}[\mathsf{t}_k]\|_2^2$,

Then problem (4.2a) – (4.2c) can be reformulated as:

$$\begin{aligned}
&\text{find} && \bar{\mathbf{a}} \in \mathbb{R}^n,\ \ \bar{\mathbf{c}} \in \mathbb{R}^m,\ \ \bar{\mathbf{x}} \in \mathbb{R}^{\tau'},\ \ \{\mathbf{x}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}'}, \\
& && \{\mathbf{a}_k \in \mathbb{R}^n\}_{k=1}^n, \{\mathbf{b}_k \in \mathbb{R}^o\}_{k=1}^n, \{\mathbf{c}_k \in \mathbb{R}^n\}_{k=1}^m, \\
& && \{\mathbf{d}_k \in \mathbb{R}^o\}_{k=1}^m, && \text{(4.10a)}
\end{aligned}$$

subject to

$$\begin{aligned}
&\bar{\mathsf{a}}_k+\bar{\mathsf{x}}_\mathsf{t}+2\mathsf{x}_k[\mathsf{t}+1]-2\mathbf{b}_k^\top\mathbf{u}[\mathsf{t}]\geq\|\mathbf{a}_k+\mathbf{x}[\mathsf{t}]\|_2^2 && \mathsf{t}\in\mathcal{T}',\ k\in\mathcal{N} && \text{(4.10b)}\\
&\bar{\mathsf{a}}_k+\bar{\mathsf{x}}_\mathsf{t}-2\mathsf{x}_k[\mathsf{t}+1]+2\mathbf{b}_k^\top\mathbf{u}[\mathsf{t}]\geq\|\mathbf{a}_k-\mathbf{x}[\mathsf{t}]\|_2^2 && \mathsf{t}\in\mathcal{T}',\ k\in\mathcal{N} && \text{(4.10c)}\\
&\bar{\mathsf{c}}_k+\bar{\mathsf{x}}_\mathsf{t}+2\mathsf{y}_k[\mathsf{t}]-2\mathbf{d}_k^\top\mathbf{u}[\mathsf{t}]\geq\|\mathbf{c}_k+\mathbf{x}[\mathsf{t}]\|_2^2 && \mathsf{t}\in\mathcal{T}',\ k\in\mathcal{M} && \text{(4.10d)}\\
&\bar{\mathsf{c}}_k+\bar{\mathsf{x}}_\mathsf{t}-2\mathsf{y}_k[\mathsf{t}]+2\,\mathbf{d}_k^\top\mathbf{u}[\mathsf{t}]\geq\|\mathbf{c}_k-\mathbf{x}[\mathsf{t}]\|_2^2 && \mathsf{t}\in\mathcal{T}',\ k\in\mathcal{M} && \text{(4.10e)}
\end{aligned}$$

$$\begin{aligned}
&\bar{\mathsf{x}}_t = \|\mathbf{x}[\mathsf{t}]\|_2^2 && \mathsf{t}\in\mathcal{T}' && \text{(4.10f)}\\
&\bar{\mathsf{a}}_k = \|\mathbf{a}_k\|_2^2 && k\in\mathcal{N} && \text{(4.10g)}\\
&\bar{\mathsf{c}}_k = \|\mathbf{c}_k\|_2^2 && k\in\mathcal{M} && \text{(4.10h)}
\end{aligned}$$

$$\begin{aligned}
&\mathsf{x}_k[\mathsf{t}+1] = \mathbf{a}_k^\top\mathbf{x}[\mathsf{t}] + \mathbf{b}_k^\top\mathbf{u}[\mathsf{t}] && \mathsf{t}\in\mathcal{T}\backslash\mathcal{T}',\quad k\in\mathcal{N} && \text{(4.10i)}\\
&\mathsf{y}_k[\mathsf{t}] = \mathbf{c}_k^\top\mathbf{x}[\mathsf{t}] + \mathbf{d}_k^\top\mathbf{u}[\mathsf{t}] && \mathsf{t}\in\mathcal{T}\backslash\mathcal{T}',\quad k\in\mathcal{M} && \text{(4.10j)}
\end{aligned}$$

It can be easily observed that problems (4.2a) – (4.2c) and (4.10a) – (4.10j) are equivalent. This is because the pair of constraints (4.10b) and (4.10c), are equivalent to

$$\begin{aligned} \mathsf{x}_k[\mathsf{t}+1] &\geq \mathbf{a}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{b}_k^\top \mathbf{u}[\mathsf{t}] - (\bar{\mathsf{x}}_t - \|\mathbf{x}[\mathsf{t}]\|_2^2) - (\bar{\mathsf{a}}_k - \|\mathbf{a}_k\|_2^2) \\ \mathsf{x}_k[\mathsf{t}+1] &\leq \mathbf{a}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{b}_k^\top \mathbf{u}[\mathsf{t}] + (\bar{\mathsf{x}}_t - \|\mathbf{x}[\mathsf{t}]\|_2^2) + (\bar{\mathsf{a}}_k - \|\mathbf{a}_k\|_2^2) \end{aligned}$$

and the pair of constraints (4.10d) and (4.10e), are equivalent to:

$$\begin{aligned} \mathsf{y}_k[\mathsf{t}] &\geq \mathbf{c}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{d}_k^\top \mathbf{u}[\mathsf{t}] - (\bar{\mathsf{x}}_t - \|\mathbf{x}[\mathsf{t}]\|_2^2) - (\bar{\mathsf{c}}_k - \|\mathbf{c}_k\|_2^2) \\ \mathsf{y}_k[\mathsf{t}] &\leq \mathbf{c}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{d}_k^\top \mathbf{u}[\mathsf{t}] + (\bar{\mathsf{x}}_t - \|\mathbf{x}[\mathsf{t}]\|_2^2) + (\bar{\mathsf{c}}_k - \|\mathbf{c}_k\|_2^2) \end{aligned}$$

Now, according to (4.10f), (4.10g) and (4.10h), the terms $(\bar{\mathsf{x}}_t - \|\mathbf{x}[\mathsf{t}]\|_2^2)$, $(\bar{\mathsf{a}}_k - \|\mathbf{a}_k\|_2^2)$, and $(\bar{\mathsf{c}}_k - \|\mathbf{c}_k\|_2^2)$ are zero which means that:

$$\mathsf{x}_k[\mathsf{t}+1] = \mathbf{a}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{b}_k^\top \mathbf{u}[\mathsf{t}] \qquad \mathsf{t} \in \mathcal{T}', \quad k \in \mathcal{N} \tag{4.13a}$$

$$\mathsf{y}_k[\mathsf{t}] = \mathbf{c}_k^\top \mathbf{x}[\mathsf{t}] + \mathbf{d}_k^\top \mathbf{u}[\mathsf{t}] \qquad \mathsf{t} \in \mathcal{T}', \quad k \in \mathcal{M} \tag{4.13b}$$

that are equivalent to (4.2b) and (4.2b).

The primary motivation behind the use of formulation (4.10a) – (4.10h) is that the only non-convex constraints in this new formulation are (4.10a), (4.10b), and (4.10c). These non-convex constraints can be readily relaxed to

$$\bar{\mathsf{x}}_t \geq \|\mathbf{x}[\mathsf{t}]\|_2^2 \qquad \bar{\mathsf{a}}_k \geq \|\mathbf{a}_k\|_2^2 \qquad \bar{\mathsf{c}}_k \geq \|\mathbf{c}_k\|_2^2 \tag{4.14}$$

We regard this as the *parabolic relaxation* of the problem (4.2a) – (4.2c). If the relaxed problem has a unique solution for which the non-convex equalities (4.10b) – (4.10h) hold true, i.e., if

$$\bar{\mathsf{x}}_t^{\mathrm{sol}} = \|\mathbf{x}^{\mathrm{sol}}[\mathsf{t}]\|_2^2 \quad \bar{\mathsf{a}}_k^{\mathrm{sol}} = \|\mathbf{a}_k^{\mathrm{sol}}\|_2^2 \quad \bar{\mathsf{c}}_k^{\mathrm{sol}} = \|\mathbf{c}_k^{\mathrm{sol}}\|_2^2 \tag{4.15}$$

then we refer to the relaxation as exact. However, like any other convex relaxation, we may encounter cases where the equalities in (4.15) are not true and the relaxation is inexact. To remedy this issue, we introduce a family of penalty functions whose minimization can help with cases where parabolic relaxation is inexact.

4.6 Sequential Penalization

Motivated by [65], we minimize a penalty function to obtain feasible points for problem (4.10a) – (4.10j). The *penalized parabolic relaxation* of the bi-linear problem (4.10a) – (4.10j) is given as

$$\underset{\substack{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\\ \{\mathbf{x}[\mathsf{t}]\}_{\mathsf{t}\in\mathcal{T}'},\\ \bar{\mathbf{a}},\bar{\mathbf{c}},\bar{\mathbf{x}}}}{\text{minimize}} \quad \eta_1\big(\mathbf{1}_n^\top\bar{\mathbf{a}} - 2\,\mathrm{tr}\big\{[\mathbf{a}_1\ldots\mathbf{a}_n]^\top\check{\mathbf{A}}\big\} + \|\check{\mathbf{A}}\|_{\mathrm{F}}^2\big)+$$

$$\eta_2\big(\mathbf{1}_n^\top\bar{\mathbf{c}} - 2\,\mathrm{tr}\big\{[\mathbf{c}_1\ldots\mathbf{c}_n]^\top\check{\mathbf{C}}\big\} + \|\check{\mathbf{C}}\|_{\mathrm{F}}^2\big)+$$

$$\eta_3\sum\nolimits_{\mathsf{t}\in\mathcal{T}}\big(\bar{\mathsf{x}}_\mathsf{t} - 2\,\check{\mathbf{x}}^\top[\mathsf{t}]\,\mathbf{x}[\mathsf{t}] + \|\check{\mathbf{x}}[\mathsf{t}]\|_2^2\big) \tag{4.16a}$$

$$\text{subject to} \qquad (4.10b), (4.10c), (4.10d), (4.10e), (4.14), (4.10i), (4.10j) \tag{4.16b}$$

where $\eta_1, \eta_2, \eta_3 > 0$ are fixed regularization parameters and $\check{\mathbf{A}} \in \mathbb{R}^{n\times n}$, $\check{\mathbf{C}} \in \mathbb{R}^{m\times n}$, and $\{\check{\mathbf{x}}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}'}$ represent an arbitrary initial guess for the solution. This approach can be

Algorithm 2 Sequential Penalized Parabolic Relaxation.

```
Require: $\check{\mathbf{A}} \in \mathbb{R}^{n\times n}$, $\check{\mathbf{C}} \in \mathbb{R}^{m\times n}$, $\{\check{\mathbf{x}}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}'}$, and $\eta_1, \eta_2, \eta_3 > 0$ repeat
  S
      until
 1: ;
      olve the penalized convex problem (4.16a) – (4.16b) to obtain ($\mathbf{A}^{\mathrm{sol}}, \mathbf{B}^{\mathrm{sol}}, \mathbf{C}^{\mathrm{sol}}, \mathbf{D}^{\mathrm{sol}}$,
    $\{\mathbf{x}^{\mathrm{sol}}[\mathsf{t}]\}_{\mathsf{t}\in\mathcal{T}'}$, $\bar{\mathbf{a}}^{\mathrm{sol}}, \bar{\mathbf{c}}^{\mathrm{sol}}, \bar{\mathbf{x}}^{\mathrm{sol}}$).
 2: set $\check{\mathbf{A}} := \mathbf{A}^{\mathrm{sol}}$, $\check{\mathbf{C}} := \mathbf{C}^{\mathrm{sol}}$, and $\check{\mathbf{x}}[\mathsf{t}] := \mathbf{x}^{\mathrm{sol}}[\mathsf{t}]$ for all $\mathsf{t} \in \mathcal{T}'$.
 3: stopping criterion is met.
 4: return ($\mathbf{A}^{\mathrm{sol}}, \mathbf{B}^{\mathrm{sol}}, \mathbf{C}^{\mathrm{sol}}, \mathbf{D}^{\mathrm{sol}}$) =0
```

applied sequentially. To this end, we propose Algorithm 2 which starts from an initial point and solves a sequence of penalized relaxations until the ground truth system is identified.

4.7 Theoretical Guarantees

In this section, we first offer some preliminary notations and definition and then we state the main theoretical result of the paper.

Definition 1. *For every* $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{C} \in \mathbb{R}^{m\times n}$, *and* $\{\mathbf{x}[\mathsf{t}] \in \mathbb{R}^n\}_{\mathsf{t}\in\mathcal{T}'}$, *define the Jacobian matrix*

$$\mathbf{J}\big(\mathbf{A}, \mathbf{C}, \mathbf{x}[\mathsf{t}_1], \ldots, \mathbf{x}[\mathsf{t}_{\mathcal{T}'}]\big) \triangleq \begin{bmatrix} \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathrm{vec}\{\mathbf{A}\}} & \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathrm{vec}\{\mathbf{B}\}} & \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathrm{vec}\{\mathbf{C}\}} & \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathrm{vec}\{\mathbf{D}\}} & \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathbf{x}[\mathsf{t}_1]} \cdots \frac{\partial \mathrm{vec}\{\mathbf{Q}\}}{\partial \mathbf{x}[\mathsf{t}_{\tau'}]} \\ \\ \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathrm{vec}\{\mathbf{A}\}} & \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathrm{vec}\{\mathbf{B}\}} & \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathrm{vec}\{\mathbf{C}\}} & \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathrm{vec}\{\mathbf{D}\}} & \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathbf{x}[\mathsf{t}_1]} \cdots \frac{\partial \mathrm{vec}\{\mathbf{R}\}}{\partial \mathbf{x}[\mathsf{t}_{\tau'}]} \end{bmatrix} \tag{4.17}$$

where the matrix functions

$$\begin{aligned} \mathbf{Q} : \mathbb{R}^{n\times n}\times\mathbb{R}^{n\times o}\times\mathbb{R}^{m\times n}\times\mathbb{R}^{m\times o}\times\mathbb{R}^{n}\times\ldots\times\mathbb{R}^{n} \quad &\rightarrow \quad \mathbb{R}^{n\times\tau} \\ \mathbf{R} : \mathbb{R}^{n\times n}\times\mathbb{R}^{n\times o}\times\mathbb{R}^{m\times n}\times\mathbb{R}^{m\times o}\times\mathbb{R}^{n}\times\ldots\times\mathbb{R}^{n} \quad &\rightarrow \quad \mathbb{R}^{m\times\tau} \end{aligned}$$

are defined as:

$$\mathbf{Q}\big(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big) \triangleq \big[\mathbf{x}[2]\ldots\mathbf{x}[\tau{+}1]\big] - \mathbf{A}\big[\mathbf{x}[1]\ldots\mathbf{x}[\tau]\big] - \mathbf{B}\big[\mathbf{u}[1]\ldots\mathbf{u}[\tau]\big] \tag{4.18a}$$

$$\mathbf{R}\big(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big) \triangleq \big[\mathbf{y}[1]\ldots\mathbf{y}[\tau]\big] - \mathbf{C}\big[\mathbf{x}[1]\ldots\mathbf{x}[\tau]\big] - \mathbf{D}\big[\mathbf{u}[1]\ldots\mathbf{u}[\tau]\big] \tag{4.18b}$$

The point $(\mathbf{A},\mathbf{C},\{\mathbf{x}[\mathrm{t}]\}_{\mathrm{t}\in\mathcal{T}'})$ *is said to satisfy the linear independence constraint qualification (LICQ) condition if the columns of* $\mathbf{J}\big(\mathbf{A},\mathbf{C},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big)$ *are linearly independent. Moreover, the singularity function* $s:\mathbb{R}^{n\times n}\times\mathbb{R}^{m\times n}\times\mathbb{R}^n\times\ldots\times\mathbb{R}^n\to\mathbb{R}$ *is defined as*

$$s\big(\mathbf{A},\mathbf{C},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big) \triangleq \begin{cases} \sigma_{\min}\big\{\mathbf{J}\big(\mathbf{A},\mathbf{C},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big)\big\} & \mathbf{J} \textit{ is full row rank} \\ 0 & \textit{otherwise,} \end{cases}$$

where $\sigma_{\min}$ *denotes the smallest singular value operator.*

The next theorem states that if the initial guess of the penalized convex relaxation (4.16a) – (4.16b) is sufficiently close to the ground truth, then the relaxation is exact and the ground truth can be recovered by solving the penalized convex relaxation.

Theorem 1. *Let* $(\overset{*}{\mathbf{A}},\overset{*}{\mathbf{B}},\overset{*}{\mathbf{C}},\overset{*}{\mathbf{D}},\{\overset{*}{\mathbf{x}}[\mathrm{t}]\}_{\mathrm{t}\in\mathcal{T}'})$ *denote a solution for the problem (4.2a) – (4.2c) that satisfies the LICQ condition. If*

$$\sqrt{\eta_1^2\|\overset{*}{\mathbf{A}}-\check{\mathbf{A}}\|_{\mathrm{F}}^2+\eta_2^2\|\overset{*}{\mathbf{C}}-\check{\mathbf{C}}\|_{\mathrm{F}}^2+\eta_3^2\textstyle\sum_{\mathrm{t}\in\mathcal{T}'}\|\overset{*}{\mathbf{x}}[\mathrm{t}]-\check{\mathbf{x}}[\mathrm{t}]\|_2^2} < \min\left\{\frac{\eta_1}{\sqrt{\tau}},\frac{\eta_2}{\sqrt{\tau}},\frac{\eta_3}{\sqrt{m+n}}\right\}\frac{s\big(\mathbf{A},\mathbf{C},\mathbf{x}[\mathrm{t}_1],\ldots,\mathbf{x}[\mathrm{t}_{\mathcal{T}'}]\big)}{2} \tag{4.19}$$

$$\mathbf{K}\big(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{x}[\mathrm{t}_1],\dots,\mathbf{x}[\mathrm{t}_{\tau'}],\bar{\mathbf{a}},\bar{\mathbf{c}},\bar{\mathbf{x}}\big)\triangleq$$
$$\begin{bmatrix} \mathrm{diag}\{\bar{\mathbf{a}}-\mathrm{diag}\{\mathbf{A}\mathbf{A}^\top\}\} & \mathbf{0}_{n\times m} & [\mathbf{x}[\mathrm{t}_1+1]\dots\mathbf{x}[\mathrm{t}_{\tau'}+1]]-\mathbf{A}[\mathbf{x}[\mathrm{t}_1]\dots\mathbf{x}[\mathrm{t}_{\tau'}]]-\mathbf{B}[\mathbf{u}[\mathrm{t}_1]\dots\mathbf{u}[\mathrm{t}_{\tau'}]] \\ * & \mathrm{diag}\{\bar{\mathbf{c}}-\mathrm{diag}\{\mathbf{C}\mathbf{C}^\top\}\} & [\mathbf{y}[\mathrm{t}_1]\dots\mathbf{y}[\mathrm{t}_{\tau'}]]-\mathbf{C}[\mathbf{x}[\mathrm{t}_1]\dots\mathbf{x}[\mathrm{t}_{\tau'}]]-\mathbf{D}[\mathbf{u}[\mathrm{t}_1]\dots\mathbf{u}[\mathrm{t}_{\tau'}]] \\ * & * & \mathrm{diag}\Big\{\bar{\mathbf{x}}-\mathrm{diag}\{[\mathbf{x}[\mathrm{t}_1]\dots\mathbf{x}[\mathrm{t}_{\tau'}]]^\top[\mathbf{x}[\mathrm{t}_1]\dots\mathbf{x}[\mathrm{t}_{\tau'}]]\}\Big\} \end{bmatrix} \tag{4.20}$$

---

then $(\overset{*}{\mathbf{A}},\overset{*}{\mathbf{B}},\overset{*}{\mathbf{C}},\overset{*}{\mathbf{D}},\{\overset{*}{\mathbf{x}}[\mathrm{t}]\}_{\mathrm{t}\in\mathcal{T}'})$ *is the unique solution of the penalized convex relaxation (4.16a) – (4.16b).*

Corollary 1. *According to Theorem 1, it can be observed that if Algorithm 2 converges to a ground truth solution, then convergence occurs within a finite number of rounds.*

Proof. Define the matrix function $\mathbf{K}$ as shown in (4.20). Constraint (4.16b) is true if and only if

$$\mathbf{K}\big(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{x}[\mathrm{t}_1],\dots,\mathbf{x}[\mathrm{t}_{\tau'}],\bar{\mathbf{a}},\bar{\mathbf{c}},\bar{\mathbf{x}}\big)\in\mathbb{F} \tag{4.21}$$

where

$$\mathbb{F}\triangleq\big\{\mathbf{F}\in\mathbb{S}_{n+m+\tau'}\mid \mathsf{F}_{ii}+\mathsf{F}_{jj}\geq 2\mathsf{F}_{ij} \quad \forall i,j\in\{1,\dots,n+m+\tau'\}\big\}. \tag{4.22}$$

In order to prove the optimality of $(\overset{*}{\mathbf{A}}, \overset{*}{\mathbf{B}}, \overset{*}{\mathbf{C}}, \overset{*}{\mathbf{D}}, \{\overset{*}{\mathbf{x}}[\mathsf{t}]\}_{\mathsf{t}\in\mathcal{T}'})$, it suffices to construct a dual certificate. To this end, define $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times\tau'}$ and $\boldsymbol{\Pi} \in \mathbb{R}^{m\times\tau'}$ as the pair of matrices satisfying:

$$\begin{bmatrix} \mathrm{vec}\{\boldsymbol{\Lambda}\} \\ \\ \mathrm{vec}\{\boldsymbol{\Pi}\} \end{bmatrix} = 2\boldsymbol{J}^{+}\big(\overset{*}{\mathbf{A}}, \overset{*}{\mathbf{C}}, \overset{*}{\mathbf{x}}[\mathsf{t}_1], \dots, \overset{*}{\mathbf{x}}[\mathsf{t}_{\tau'}]\big) \begin{bmatrix} \eta_1 \mathrm{vec}\{\overset{*}{\mathbf{A}} - \check{\mathbf{A}}\} \\ \eta_2 \mathrm{vec}\{\overset{*}{\mathbf{C}} - \check{\mathbf{C}}\} \\ \eta_3(\overset{*}{\mathbf{x}}[\mathsf{t}_1] - \check{\mathbf{x}}[\mathsf{t}_1]) \\ \vdots \\ \eta_3(\overset{*}{\mathbf{x}}[\mathsf{t}_{\tau'}] - \check{\mathbf{x}}[\mathsf{t}_{\tau'}]) \end{bmatrix} \tag{4.23}$$

We claim that the matrix

$$\boldsymbol{\Gamma} \triangleq \begin{bmatrix} \eta_1 \boldsymbol{I}_{n\times n} & \mathbf{0}_{n\times m} & \boldsymbol{\Lambda} \\ * & \eta_2 \boldsymbol{I}_{m\times m} & \boldsymbol{\Pi} \\ * & * & \eta_3 \boldsymbol{I}_{\tau'\times\tau'} \end{bmatrix} \tag{4.24}$$

qualifies as a Lagrange multiplier associated with the constraint (4.21).

Stationarity with respect to primal variables is an immediate consequence of (4.23). Additionally, according to (4.23), we have

$$\sqrt{\|\boldsymbol{\Lambda}\|_{\mathrm{F}}^2 + \|\boldsymbol{\Pi}\|_{\mathrm{F}}^2} \leq \frac{2\sqrt{\eta_1^2\|\overset{*}{\mathbf{A}} - \check{\mathbf{A}}\|_{\mathrm{F}}^2 + \eta_2^2\|\overset{*}{\mathbf{C}} - \check{\mathbf{C}}\|_{\mathrm{F}}^2 + \eta_3^2 \sum_{\mathsf{t}\in\mathcal{T}'} \|\overset{*}{\mathbf{x}}[\mathsf{t}] - \check{\mathbf{x}}[\mathsf{t}]\|_2^2}}{s\big(\mathbf{A}, \mathbf{C}, \mathbf{x}[\mathsf{t}_1], \dots, \mathbf{x}[\mathsf{t}_{\tau'}]\big)} \tag{4.25}$$

which concludes that

$$\sqrt{\|\boldsymbol{\Lambda}\|_{\mathrm{F}}^2 + \|\boldsymbol{\Pi}\|_{\mathrm{F}}^2} < \min\left\{\frac{\eta_1}{\sqrt{\tau}}, \frac{\eta_2}{\sqrt{\tau}}, \frac{\eta_3}{\sqrt{m+n}}\right\} \tag{4.26}$$

meaning that $\boldsymbol{\Gamma}$ is strictly diagonally-dominant. Since, the set of strictly diagonally-dominant matrices is the dual cone of $\mathbb{F}$, $\boldsymbol{\Gamma}$ is dual feasible and qualifies as dual certificate. $\square$

4.8 Extension to Non-Linear System Identification

In this section, we extend the proposed methodology to nonlinear systems. Consider the following dynamical equations:

$$\mathbf{x}[\mathrm{t}+1] = \mathbf{A}\,\mathbf{x}[\mathrm{t}] + \mathbf{B}\,\mathbf{u}[\mathrm{t}] + \mathbf{E}\,\mathbf{s}[\mathrm{t}] + \mathbf{F}\,\mathbf{r}[\mathrm{t}] \qquad \mathrm{t} \in \mathcal{T} \tag{4.27a}$$

$$\mathbf{y}[\mathrm{t}] = \mathbf{C}\,\mathbf{x}[\mathrm{t}] + \mathbf{D}\,\mathbf{u}[\mathrm{t}] \qquad \mathrm{t} \in \mathcal{T} \tag{4.27b}$$

$$\mathbf{r}[\mathrm{t}] = (\mathbf{G}_1\,\mathbf{x}[\mathrm{t}]) \circ (\mathbf{G}_2\,\mathbf{x}[\mathrm{t}]) \qquad \mathrm{t} \in \mathcal{T} \tag{4.27c}$$

$$\mathbf{s}[\mathrm{t}] = \mathbf{x}[\mathrm{t}] \circ \mathbf{x}[\mathrm{t}] \qquad \mathrm{t} \in \mathcal{T} \tag{4.27d}$$

where the matrices $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F})$ are unknown, and $(\mathbf{G}_1, \mathbf{G}_2)$ are known binary incidence matrices that are selected in such a way that $\mathbf{r}[\mathrm{t}]$ contains all possible monomials of type $\mathrm{x}_i[\mathrm{t}]\mathrm{x}_j[\mathrm{t}]$ $(i \neq j)$ that may appear in the dynamics. In other words,

- $\mathbf{s}[\mathrm{t}]$ encapsulates all monomials of the type $\mathrm{x}_i[\mathrm{t}]^2$,
- $\mathbf{r}[\mathrm{t}]$ encapsulates a known subset of monomials of the type $\mathrm{x}_i[\mathrm{t}]\mathrm{x}_j[\mathrm{t}]$, where $i \neq j$.

The next example gives the model of a Permanent Magnet Synchronous Machine (PMSM) in the above form.

Example 1. *Consider a PMSM machine with the following internal parameters:*

- r_{s}*: stator resistance,*
- l_{q}*: q-axis inductance,*

- l_{d}*: d-axis inductance,*
- λ_{m}*: permanent magnet flux,*
- p*: pole pairs,*
- j_{m}*: rotor inertia,*
- d_{m}*: viscous friction coefficient,*

time-domain system states:

- $i_{\mathrm{qs}}[\mathsf{t}]$*: q-axis stator current,*
- $i_{\mathrm{ds}}[\mathsf{t}]$*: d-axis stator current,*
- $w_{\mathrm{r}}[\mathsf{t}]$*: rotor electrical speed,*
- $\theta_{\mathrm{r}}[\mathsf{t}]$*: rotor electrical position,*

and time-domain input:

- $v_{\mathrm{qs}}[\mathsf{t}]$*: q-axis stator voltage,*
- $v_{\mathrm{ds}}[\mathsf{t}]$*: d-axis stator voltage,*
- $t_{\mathrm{l}}[\mathsf{t}]$*: load torque.*

The dynamics of the PMSM can be formulated as

$$\mathbf{x}[\mathsf{t}+1] = \mathbf{A}\mathbf{x}[\mathsf{t}] + \mathbf{B}\mathbf{u}[\mathsf{t}] + \mathbf{F}\mathbf{r}[\mathsf{t}] \tag{4.28a}$$

$$\mathbf{r}[t] = (\mathbf{G}_1\mathbf{x}[\mathsf{t}]) \circ (\mathbf{G}_2\mathbf{x}[\mathsf{t}]) \tag{4.28b}$$

with respect to the time-domain state and input vectors

$$\mathbf{x}[\mathsf{t}] = \begin{bmatrix} i_{\mathrm{qs}}[\mathsf{t}] & i_{\mathrm{ds}}[\mathsf{t}] & w_{\mathrm{r}}[\mathsf{t}] & \theta_{\mathrm{r}}[\mathsf{t}] \end{bmatrix}^\top \tag{4.29}$$

$$\mathbf{u}[\mathsf{t}] = \begin{bmatrix} v_{\mathrm{qs}}[\mathsf{t}] & v_{\mathrm{ds}}[\mathsf{t}] & t_{\mathrm{l}}[\mathsf{t}] \end{bmatrix}^\top \tag{4.30}$$

where

$$\mathbf{A} = \begin{bmatrix} 1-\frac{r_\mathrm{s}\delta}{l_\mathrm{q}} & 0 & -\frac{\lambda_\mathrm{m}\delta}{l_\mathrm{q}} & 0 \\ 0 & 1-\frac{r_\mathrm{s}\delta}{l_\mathrm{d}} & 0 & 0 \\ \frac{3\lambda_\mathrm{m}p^2\delta}{2j_\mathrm{m}} & 0 & 1-\frac{d_\mathrm{m}\delta}{j_\mathrm{m}} & 0 \\ 0 & 0 & 1 & \delta \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \frac{\delta}{l_\mathrm{q}} & 0 & 0 \\ 0 & \frac{\delta}{l_\mathrm{d}} & 0 \\ 0 & 0 & -\frac{p\delta}{j_\mathrm{m}} \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{F} = \begin{bmatrix} -\frac{l_\mathrm{d}\delta}{l_\mathrm{q}} & 0 & 0 \\ 0 & \frac{l_\mathrm{q}\delta}{l_\mathrm{d}} & 0 \\ 0 & 0 & \frac{3(l_\mathrm{d}-l_\mathrm{q})p^2\delta}{2j_\mathrm{m}} \\ 0 & 0 & 0 \end{bmatrix} \tag{4.31}$$

and δ is the time step, and

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{G}_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.32}$$

in section 4.8, we demonstrate the effectiveness of the proposed approach in recovering the unknown time-domain parameters of this machine.

In order to convexify the dynamical equations (4.27), we first introduce the auxiliary vectors $\{\mathbf{x}^{\mathbf{A}}[\mathrm{t}] = \mathbf{A}\mathbf{x}[\mathrm{t}] \in \mathbb{R}^n\}_{\tau\in\mathcal{T}}$, $\{\mathbf{s}^{\mathbf{E}}[\mathrm{t}] = \mathbf{E}\,\mathbf{s}[\mathrm{t}] \in \mathbb{R}^n\}_{\tau\in\mathcal{T}}$, $\{\mathbf{r}^{\mathbf{F}}[\mathrm{t}] = \mathbf{F}\,\mathbf{r}[\mathrm{t}] \in \mathbb{R}^n\}_{\tau\in\mathcal{T}}$, and

$\{\mathbf{x}^{\mathbf{C}}[\mathrm{t}] = \mathbf{C}\,\mathbf{x}[\mathrm{t}] \in \mathbb{R}^n\}_{\tau\in\mathcal{T}}$. Using these auxiliary variables, the dynamical equations (4.27) can be cast linearly in form of:

$$\mathbf{x}[\mathrm{t}+1] = \mathbf{x}^{\mathbf{A}}[\mathrm{t}] + \mathbf{B}\mathbf{u}[\mathrm{t}] + \mathbf{s}^{\mathbf{E}}[\mathrm{t}] + \mathbf{r}^{\mathbf{F}}[\mathrm{t}] \qquad \mathrm{t}\in\mathcal{T} \tag{4.33a}$$

$$\mathbf{y}[\mathrm{t}] = \mathbf{x}^{\mathbf{C}}[\mathrm{t}] + \mathbf{D}\mathbf{u}[\mathrm{t}] \qquad \mathrm{t}\in\mathcal{T} \tag{4.33b}$$

Additionally, let $\mathbf{e}_k$ and $\mathbf{f}_k$ represent the k-th column of $\mathbf{E}^\top$ and $\mathbf{F}^\top$, respectively. We then impose the following constraints to relate $\mathbf{x}^{\mathbf{A}}[\mathrm{t}]$, $\mathbf{s}^{\mathbf{E}}[\mathrm{t}]$, $\mathbf{r}^{\mathbf{F}}[\mathrm{t}]$, and $\mathbf{x}^{\mathbf{C}}[\mathrm{t}]$ and their corresponding values:

$$\bar{\mathrm{a}}_k + \bar{\mathrm{x}}_\mathrm{t} + 2\mathrm{x}^{\mathbf{A}}_k[\mathrm{t}] \geq \|\mathbf{a}_k + \mathbf{x}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34a}$$

$$\bar{\mathrm{a}}_k + \bar{\mathrm{x}}_\mathrm{t} - 2\mathrm{x}^{\mathbf{A}}_k[\mathrm{t}] \geq \|\mathbf{a}_k - \mathbf{x}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34b}$$

$$\bar{\mathrm{e}}_k + \bar{\mathrm{s}}_\mathrm{t} + 2\mathrm{s}^{\mathbf{E}}_k[\mathrm{t}] \geq \|\mathbf{e}_k + \mathbf{s}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34c}$$

$$\bar{\mathrm{e}}_k + \bar{\mathrm{s}}_\mathrm{t} - 2\mathrm{s}^{\mathbf{E}}_k[\mathrm{t}] \geq \|\mathbf{e}_k - \mathbf{s}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34d}$$

$$\bar{\mathrm{f}}_k + \bar{\mathrm{r}}_\mathrm{t} + 2\mathrm{r}^{\mathbf{F}}_k[\mathrm{t}] \geq \|\mathbf{f}_k + \mathbf{r}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34e}$$

$$\bar{\mathrm{f}}_k + \bar{\mathrm{r}}_\mathrm{t} - 2\mathrm{r}^{\mathbf{F}}_k[\mathrm{t}] \geq \|\mathbf{f}_k - \mathbf{r}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{N} \tag{4.34f}$$

$$\bar{\mathrm{c}}_k + \bar{\mathrm{x}}_\mathrm{t} + 2\mathrm{x}^{\mathbf{C}}_k[\mathrm{t}] \geq \|\mathbf{a}_k + \mathbf{x}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{M} \tag{4.34g}$$

$$\bar{\mathrm{c}}_k + \bar{\mathrm{x}}_\mathrm{t} - 2\mathrm{x}^{\mathbf{C}}_k[\mathrm{t}] \geq \|\mathbf{a}_k - \mathbf{x}[\mathrm{t}]\|_2^2 \qquad \mathrm{t}\in\mathcal{T},\quad k\in\mathcal{M} \tag{4.34h}$$

where $\bar{\mathrm{e}}_k = \|\mathbf{e}_k\|^2$, $\bar{\mathrm{f}}_k = \|\mathbf{f}_k\|^2$, $\bar{\mathrm{s}}_\mathrm{t} = \|\mathbf{s}[\mathrm{t}]\|^2$, and $\bar{\mathrm{r}}_\mathrm{t} = \|\mathbf{r}[\mathrm{t}]\|^2$. The following constraints impose the relationship between $\mathbf{r}[\mathrm{t}]$ and the corresponding monomials that it contains:

$$\mathbf{G}_1\,\mathbf{s}[\mathrm{t}] + \mathbf{G}_2\,\mathbf{s}[\mathrm{t}] + 2\,\mathbf{r}[\mathrm{t}] \geq$$

$$(\mathbf{G}_1\,\mathbf{x}[\mathrm{t}] + \mathbf{G}_2\,\mathbf{x}[\mathrm{t}]) \circ (\mathbf{G}_1\,\mathbf{x}[\mathrm{t}] + \mathbf{G}_2\,\mathbf{x}[\mathrm{t}]) \qquad \mathrm{t}\in\mathcal{T} \tag{4.35a}$$

$$\mathbf{G}_1\,\mathbf{s}[\mathrm{t}] + \mathbf{G}_2\,\mathbf{s}[\mathrm{t}] - 2\,\mathbf{r}[\mathrm{t}] \geq$$

$$(\mathbf{G}_1\,\mathbf{x}[t] - \mathbf{G}_2\,\mathbf{x}[t]) \circ (\mathbf{G}_1\,\mathbf{x}[t] - \mathbf{G}_2\,\mathbf{x}[t]) \qquad t \in \mathcal{T} \tag{4.35b}$$

Finally, the following constraints complete the proposed parabolic relaxation for nonlinear systems:

$$\bar{x}_t = \mathbf{1}^\top \mathbf{s}[t], \qquad \bar{r}_t \geq \|\mathbf{r}[t]\|_2^2 \qquad t \in \mathcal{T} \tag{4.36a}$$
$$\bar{s}_t \geq \|\mathbf{s}[t]\|_2^2, \qquad \mathbf{s}[t] \geq \mathbf{x}[t] \circ \mathbf{x}[t] \qquad t \in \mathcal{T} \tag{4.36b}$$
$$\bar{a}_k \geq \|\mathbf{a}_k\|_2^2, \quad \bar{e}_k \geq \|\mathbf{e}_k\|_2^2, \quad \bar{f}_k \geq \|\mathbf{f}_k\|_2^2 \qquad k \in \mathcal{N} \tag{4.36c}$$
$$\bar{c}_k \geq \|\mathbf{c}_k\|_2^2 \qquad k \in \mathcal{M}. \tag{4.36d}$$

If equality holds for all of the constraints in (4.36), then the relaxation is exact. The penalized parabolic relaxation for identification of nonlinear systems can cast as follows:

$$\begin{aligned}
\text{minimize} \qquad & \eta_1\big(\mathbf{1}_n^\top \bar{\mathbf{a}} - 2\,\mathrm{tr}\big\{[\mathbf{a}_1 \ldots \mathbf{a}_n]^\top \check{\mathbf{A}}\big\} + \|\check{\mathbf{A}}\|_{\mathrm{F}}^2\big)+ \\
& \eta_2\big(\mathbf{1}_n^\top \bar{\mathbf{c}} - 2\,\mathrm{tr}\big\{[\mathbf{c}_1 \ldots \mathbf{c}_n]^\top \check{\mathbf{C}}\big\} + \|\check{\mathbf{C}}\|_{\mathrm{F}}^2\big)+ \\
& \eta_3\big(\mathbf{1}_n^\top \bar{\mathbf{e}} - 2\,\mathrm{tr}\big\{[\mathbf{e}_1 \ldots \mathbf{e}_n]^\top \check{\mathbf{E}}\big\} + \|\check{\mathbf{E}}\|_{\mathrm{F}}^2\big)+ \\
& \eta_4\big(\mathbf{1}_n^\top \bar{\mathbf{f}} - 2\,\mathrm{tr}\big\{[\mathbf{f}_1 \ldots \mathbf{f}_n]^\top \check{\mathbf{F}}\big\} + \|\check{\mathbf{F}}\|_{\mathrm{F}}^2\big)+ \\
& \eta_5 \textstyle\sum_{t \in \mathcal{T}} \big(\bar{r}_t - 2\,\check{\mathbf{r}}^\top[t]\,\mathbf{r}[t] + \|\check{\mathbf{r}}[t]\|_2^2\big)+ \\
& \eta_6 \textstyle\sum_{t \in \mathcal{T}} \big(\bar{s}_t - 2\,\check{\mathbf{s}}^\top[t]\,\mathbf{s}[t] + \|\check{\mathbf{s}}[t]\|_2^2\big)+ \\
& \eta_7 \textstyle\sum_{t \in \mathcal{T}} \big(\bar{x}_t - 2\,\check{\mathbf{x}}^\top[t]\,\mathbf{x}[t] + \|\check{\mathbf{x}}[t]\|_2^2\big) && (4.37a) \\
\text{subject to} \qquad & (4.33), (4.34), (4.35), (4.36). && (4.37b)
\end{aligned}$$

As demonstrated before, this problem can be solved sequentially to obtain a feasible solution for the set of dynamical equations (4.27).

4.9 Numerical Results

In this section, we consider case studies of both large-scale and small-scale system identification problems.

4.9.1 Linear Systems

In this case study, we consider system identification problems with $n = 16$, $m = 12$, $o = 10$, $\tau = 250$, and $\hat{\mathcal{T}} = \mathcal{T} \setminus \{1, 6, 11, \ldots, 246\}$. The resulting problem is 3928-dimensional accounting for the elements of $\mathbf{A} \in \mathbb{R}^{16\times16}$, $\mathbf{B} \in \mathbb{R}^{16\times10}$, $\mathbf{C} \in \mathbb{R}^{12\times16}$, $\mathbf{D} \in \mathbb{R}^{12\times10}$, and $\{\mathbf{x}[\mathsf{t}] \in \mathbb{R}^{16}\}_{\mathsf{t}\in\mathcal{T}'}$. The ground truth matrices are chosen as follows:

- Every element of $\mathbf{A} - \boldsymbol{I}_{n\times n}$ is uniformly chosen from the interval $[-0.25, +0.25]$.
- The elements of $\mathbf{B} \in \mathbb{R}^{16\times10}$, $\mathbf{C} \in \mathbb{R}^{12\times16}$, and $\mathbf{D} \in \mathbb{R}^{12\times10}$ have zero-mean standard normal distribution.
- The elements of $\mathbf{x}[1]$ are uniformly chosen from the interval $[0.5; 1.5]$.
- For every $\mathsf{t} \in \mathcal{T}$, we have $\mathbf{u}[\mathsf{t}] = \mathbf{F}\mathbf{x}[\mathsf{t}] + \mathbf{w}[\mathsf{t}]$, where the elements of $\mathbf{w}[\mathsf{t}]$ have independent Gaussian distribution with zero mean and standard deviation 0.1. Additionally, $\mathbf{F} = -(\boldsymbol{I}_{m\times m} + \mathbf{B}^\top\mathbf{P})^{-1}\mathbf{B}^\top\mathbf{P}\mathbf{A}$ is an optimal LQR controller with $\mathbf{P}$ representing the unique positive-definite solution to the Riccati equation:

$$\mathbf{A}^\top\mathbf{P}\mathbf{A} + \boldsymbol{I}_{n\times n} - \mathbf{P} = \mathbf{A}^\top\mathbf{P}\mathbf{B}(\boldsymbol{I}_{n\times n} + \mathbf{B}^\top\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^\top\mathbf{P}\mathbf{A} \tag{4.38}$$

We generate 15 random systems and solve the resulting problem, using the sequential Algorithm 2 with the initial point $\check{\mathbf{A}} = \boldsymbol{I}_{n\times n}$ and $\check{\mathbf{x}}[\mathsf{t}] = \mathbf{0}_n$ for every $t \in \hat{\mathcal{T}}$. Figure 4.2 illustrates the convergence of the error function

$$\left(\sum_{\mathsf{t}=1}^{\tau} \|\mathbf{x}[\mathsf{t}] - \mathbf{x}^{\text{true}}[\mathsf{t}]\|_2^2\right)^{1/2} \tag{4.39}$$

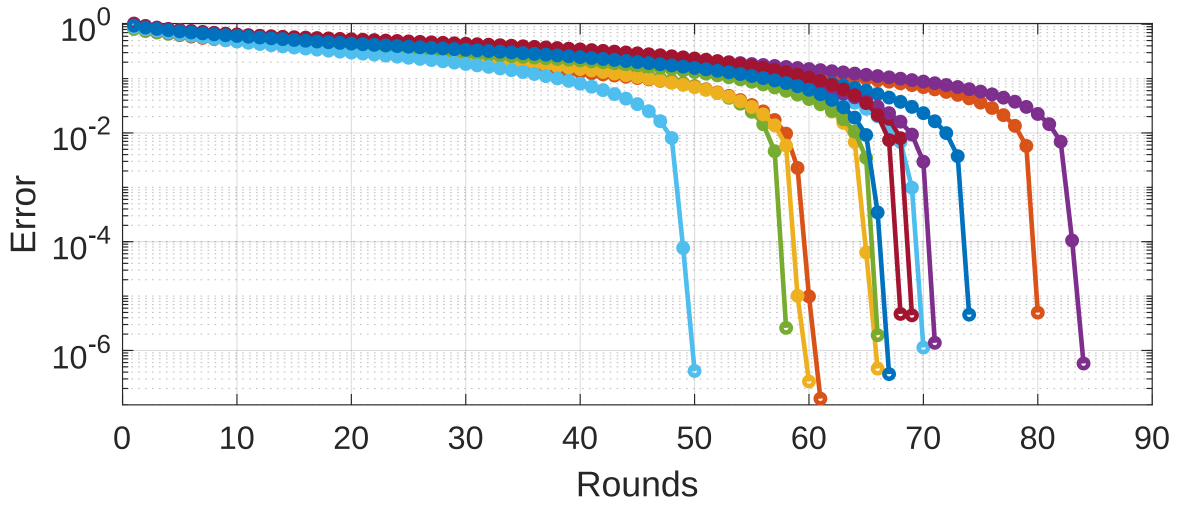

Figure 4.2. The performance of the sequential Algorithm 2 for linear system identification.

for all of the 15 experiments. Each round of the penalized parabolic is solved in less than 15 seconds.

In order to emphasize the efficacy of our proposed method compared to the fundamental and state-of-the-art linear system identification methods that are still employed for system identification, we have performed system identification on the same randomly generated systems using the methods proposed in [66]. We found that subspace system identification works really well when all the instances of the state matrix $\mathbf{x}[t]$ are well known in advance, however, when a sub-sample or the portion of the input states $\mathbf{x}[t]$ were known the method fails to converge no matter how many iterations were performed. In comparison, our proposed method by penalizing and iteratively solving the problem converges quite quickly with high accuracy.

4.9.2 Nonlinear Systems

In this part we consider a PMSM machine with the parameters $r_{\mathrm{s}} = 0.1465$, $l_{\mathrm{d}} = l_{\mathrm{q}} = 0.0001395$, $d_{\mathrm{m}} = 0.009$, $\lambda_{\mathrm{m}} = 0.005$, $p = 7$, and $j_{\mathrm{m}} = 0.0000835$. We generated time-domain signals with time steps of size $\delta = 5 \times 10^{-6}$ seconds and attempted to recover the unknown

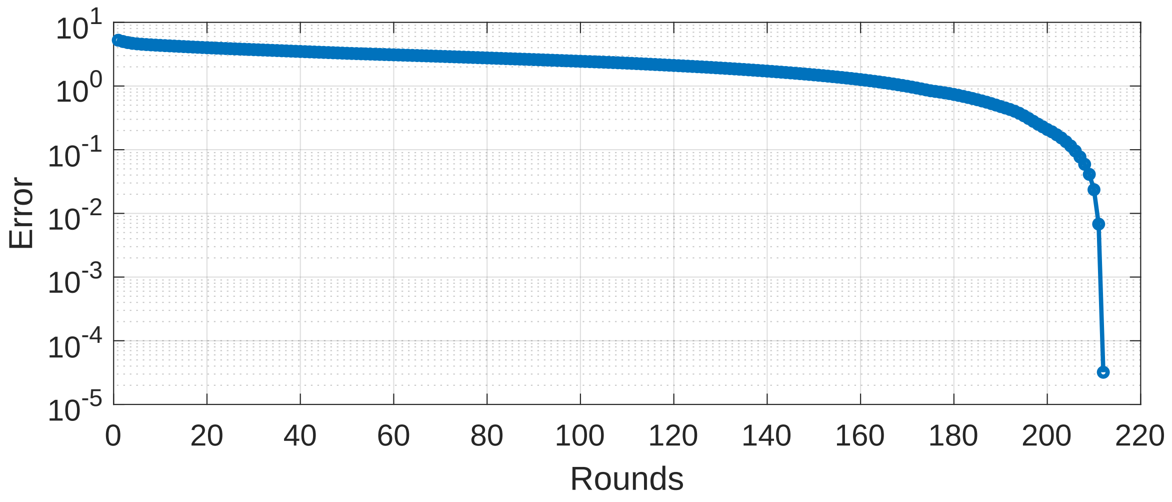

Figure 4.3. The performance of the proposed sequential approach on the PMSM nonlinear model.

parameters using the approach proposed in section 4.8. We considered a time horizon of length 20 with the following control signals:

- Each $v_{\mathrm{qs}}[\mathsf{t}]$ random Gaussian distribution with mean 12 and standard deviation 1.
- Each $v_{\mathrm{ds}}[\mathsf{t}]$ random Gaussian distribution with mean 0.25 and standard deviation 1.
- $t_{\mathrm{l}}[\mathsf{t}] = \begin{cases} 0 & \mathsf{t} < 10 \\ 0.25 & \mathsf{t} \geq 10 \end{cases}$

We assumed that the following parameters are known:

- The matrices $\mathbf{G}_1$ and $\mathbf{G}_2$.
- The last row of matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{F}$, due to the trivial relation between rotor position and rotor speed.
- The state vector $\mathbf{x}[\mathsf{t}]$ at times $\mathsf{t} \in \{1, 5, 15, 20\}$.

The convergence of the error function (4.39) is demonstrated in Figure 4.3.

CHAPTER 5

Conclusion

5.1 Future Directions

We have discussed in detail Semidefinite Programming, a sub-class of Convex Optimization with two flavors, First and Second order, with and without sparsity. The purpose of this document was to establish the usefulness of this highly researched and potentially game changing field of Optimization. The impact of a fast and near optimal optimization of intricate and intriguing Convex Optimization problems relaxed through SDPs will be deep in almost all the fields of Computer Science and Engineering, Artificial intelligence, Space, Genetics, Medicine and many other fields.

After introducing the subject in hand, detailing the recent and past prominent work, we galloped towards the modern applications that has been impacted by the recent success of SDP algorithms, these new findings has been instrumental in sprouting many more areas where these algorithms can be efficiently utilized and can solve problems previously thought unsolvable.

In the end, we spotlight our proposed novel SDP sequential parabolic relaxation technique that not only guarantees the optimal solution but also make sure it converges in finite polynomial time as compared to the peer SDP algorithms proposed recently and in the past decades. The proposed algorithm, gives a theoretical guarantee of convergence in finite polynomial time. We want to utilize our finding on the complex problems related to Power Systems' Flow Optimization and Unit Commitment. In general, the Power System's data come in huge matrices, optimizing such problems are hard and takes a lot of time and computational power to approximate the solution of a certain cost function. We will analyzed our proposed system on these real world scenarios and compare the performance, latency and accuracy of the contemporary algorithms with our proposed method.We will be trying to expanding on our proposed algorithm to cater for newly sprouting domains of Data Science, Machine Learning, Deep Learning and many more.

5.2 Conclusion

We utilized our novel algorithm on the System Identification problem. System identification is a process of finding optimized parameters of a unknown system using the input and outputs of the system, it has been around for decades. There are numerous approaches to identify the linear and non-linear systems, however, they suffer from complexity, speed and the computational hardships. This paper presents a novel parabolic relaxation with the aim of identifying linear and nonlinear dynamical systems. The proposed convex relaxation relies on a significantly smaller number of auxiliary variables leading to higher efficiency compared to the state-of-the-art methods. We also gave theoretical guarantees on the feasibility of the method in discussion. Numerical methods are provided to demonstrate the effectiveness of the proposed sequential parabolic relaxation on both linear and non-linear systems. Moreover, we formulate linear and non-linear problems and used the known state-of-the art system identification methods to determine the parameters, these known methods failed to converge when the state input is partially known. This further proves the efficacy of our proposed method.

REFERENCES

[1] YALMIP. (2018) YALMIP optimization solvers. [Online]. Available: https://yalmip.github.io/allsolvers/

[2] Wikipedia. (2018) wikipedia.org mathematical optimization softare. [Online]. Available: https://en.wikipedia.org/wiki/List_of_optimization_software

[3] S. Burer and R. Monteiro, "A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization," *MATHEMATICAL PROGRAMMING*, vol. 95, no. 2, pp. 329–357, FEB. 2003.

[4] S. Burer, R. Monteiro, and Y. Zhang, "Interior-point algorithms for semidefinite programming based on a nonlinear formulation," *COMPUTATIONAL OPTIMIZATION AND APPLICATIONS*, vol. 22, no. 1, pp. 49–79, APR. 2002.

[5] S. Burer, "Semidefinite programming in the space of partial positive semidefinite matrices," *SIAM Journal on Optimization*, vol. 14, no. 1, pp. 139 – 72, 2003//.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization.* New York, NY, USA: Cambridge University Press, 2004.

[7] R. Madani, "Computational methods for nonlinear optimization problems: Theory and applications," *Dissertation Thesis*, pp. 223 –, 2015.

[8] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, 1994.

[9] F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Mathematical Programming, Series B*, vol. 95, no. 1, pp. 3 – 51, 2003.

[10] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and Its Applications*, vol. 284, no. 1-3, pp. 193 – 228, 1998/11/15.

[11] Y. Nesterov and A. Nemirovskii, "Interior-point polynomial algorithms in convex programming," *Society for Industrial and Applied Mathematics*, 1994. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611970791

[12] R. Y. Zhang, C. Josz, and S. Sojoudi, "Conic optimization theory: Convexification techniques and numerical algorithms," *Proceedings of the American Control Conference*, vol. 2018-June, pp. 798 – 815, 2018.

[13] Z.-Q. Luo, W.-K. Ma, A. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 20 – 34, 2010.

[14] Q. Li and W. Ma, "A new low-rank solution result for a semidefinite program problem subclass with applications to transmit beamforming optimization," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3446–3450, March 2016.

[15] R. Polyak, "Modified barrier functions: Theory and methods," *Mathematical Programming*, vol. 54, no. 177–222, 1992.

[16] X. Zhao, D. Sun, and K. Toh, "A newton-cg augmented lagrangian method for semidefinite programming," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1737–1765, 2010.

[17] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz, "An interior-point method for semidefinite programming," *SIAM Journal on Optimization*, vol. 6, pp. 342–361, 1996.

[18] M. Kocvara and M. Stingl, "A generalized augmented lagrangian method for semidenite programming," 04 2019.

[19] A. Gleixner, M. Bastubbe, L. Eifler, T. Gally, G. Gamrath, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. E. Lübbecke, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, C. Schubert, F. Serrano, Y. Shinano, J. M. Viernickel, M. Walter, F. Wegscheider, J. T. Witt, and J. Witzig, "The SCIP Optimization Suite 6.0," July 2018.

[20] T. Gally, M. E. Pfetsch, and S. Ulbrich, "A framework for solving mixed-integer semidefinite programs," *Optimization Methods and Software*, vol. 33, no. 3, pp. 594–632, 2018.

[21] M. Fukuda, M. Kojima, K. Murota, and K. Nakata, "Exploiting sparsity in semidefinite programming via matrix completion i: General framework," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 647–674, 2001. [Online]. Available: https://doi.org/10.1137/S1052623400366218

[22] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota, "Exploiting sparsity in semidefinite programming via matrix completion ii: implementation and numerical results," *Mathematical Programming*, vol. 95, no. 2, pp. 303–327, Feb 2003.

[23] F. Jarre, "Burer's key assumption for semidefinite and doubly nonnegative relaxations," *Optimization Letters*, vol. 6, no. 3, pp. 593 – 9, 2012/03/, burer key assumption;semidefinite relaxations;doubly nonnegative relaxations;nonconvex quadratic programs;. [Online]. Available: http://dx.doi.org/10.1007/s11590-010-0269-8

[24] F. Alizadeh, J. Haeberly, and M. Overton, "Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 746–768, 1998. [Online]. Available: https://doi.org/10.1137/S1052623496304700

[25] S. Burer and D. Vandenbussche, "Solving lift-and-project relaxations of binary integer programs," *SIAM Journal on Optimization*, vol. 16, no. 3, pp. 726 – 750, 2006.

[26] L. Lov´asz and A. Schrijver., "Cones of matrices and set-functions and 0-1 optimization," *SIAM Journal on Optimization*, vol. 1, pp. 166–190, 1991.

[27] IBM-ILOG. (2018) CPLEX optimization studio. [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer

[28] Z. Wen, D. Goldfarb, and W. Yin, "Alternating direction augmented lagrangian methods for semidefinite programming," *Mathematical Programming Computation*, vol. 2, no. 3, pp. 203–230, Dec 2010.

[29] M. Kojima, "Exploiting structured sparsity in large scale semidefinite programming problems," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6327 LNCS, pp. 4 – 9, 2010.

[30] S. Kim, M. Kojima, M. Mevissen, and M. Yamashita, "Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion," *Mathematical Programming*, vol. 129, no. 1, pp. 33 – 68, 2011.

[31] Y. Sun, M. S. Andersen, and L. Vandenberghe, "Decomposition in conic optimization with partially separable structure," *SIAM Journal on Optimization*, vol. 24, pp. 873–897, 2014.

[32] J. E. Spingarn, "Applications of the method of partial inverses to convex programming: Decomposition," *Math. Program.*, vol. 32, pp. 199–223, 01 1985.

[33] D. Rose, "Triangulated graphs and the elimination process," *Journal of Mathematical Analysis and Applications*, vol. 32, no. 3, pp. 597 – 609, 1970/12/.

[34] L. Vandenberghe and M. S. Andersen, "Chordal graphs and semidefinite optimization," *Found. Trends Optim.*, vol. 1, no. 4, pp. 241–433, May 2015. [Online]. Available: http://dx.doi.org/10.1561/2400000006

[35] F. Lieder, "Simplified semidefinite and completely positive relaxations," *Operations Research Letters*, vol. 43, no. 6, pp. 579 – 580, 2015.

[36] M. A. Erdogdu, A. Ozdaglar, P. A. Parrilo, and N. Denizcan Vanli, "Convergence Rate of Block-Coordinate Maximization Burer-Monteiro Method for Solving Large SDPs," *arXiv e-prints*, p. arXiv:1807.04428, Jul 2018.

[37] R. Madani, A. Kalbat, and J. Lavaei, "A low-complexity parallelizable numerical algorithm for sparse semidefinite programming," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1898–1909, Dec 2018.

[38] R. Zhang and J. Lavaei, "Sparse semidefinite programs with near-linear time complexity," *2018 IEEE Conference on Decision and Control (CDC)*, pp. 1624–1631, 12 2018.

[39] T. Pumir, S. Jelassi, and N. Boumal, "Smoothed analysis of the low-rank approach for smooth semidefinite programs [arxiv]," *arXiv (USA)*, pp. 19 pp. –, 2018/06/10.

[40] B. Yang, K. Anstreicher, and S. Burer, "Quadratic programs with hollows," *Mathematical Programming*, vol. 170, no. 2, pp. 541 – 553, 2018.

[41] E. d. Klerk, "Exploiting special structure in semidefinite programming: A survey of theory and applications," *European Journal of Operational Research*, vol. 201, no. 1, pp. 1 – 10, 2010.

[42] L. Ljung, *System Identification (2nd Ed.): Theory for the User.* USA: Prentice Hall PTR, 1999.

[43] T. Söderström and P. Stoica, "Instrumental variable methods for system identification," *Journal of Circuits, Systems and Signal Processing*, vol. 21, 2002.

[44] L. Fu and P. Li, "The research survey of system identification method," *Proceedings - 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2013*, vol. 2, pp. 397 – 401, 2013.

[45] K. Astrom and P. Eykhoff, "System identification-a survey," *Automatica (UK)*, vol. 7, no. 2, pp. 123 – 62, 1971.

[46] L. Ljung, T. Chen, and B. Mu, "A shift in paradigm for system identification," *International Journal of Control*, vol. 93, no. 2, pp. 173–180, 2020.

[47] J. Stiasny, G. Misyris, and S. Chatzivasileiadis, “Physics-informed neural networks for non-linear system identification applied to power system dynamics [arxiv],” *arXiv (USA)*, p. 6, 2020/04/08.

[48] T. Kumon, T. Suzuki, M. Iwasaki, M. Matsuzaki, N. Matsui, and S. Okuma, “System identification using a genetic algorithm and its application to internal adaptive model control,” *Electrical Engineering in Japan (English translation of Denki Gakkai Ronbunshi)*, vol. 142, no. 4, pp. 45 – 55, 2003.

[49] Z. Wang, L. Zhao, and X. Luo, “Levy-particle swarm optimization intelligent search-based iterative identification for nonparametric models of bilinear systems with gaussian mixture noises,” *Transactions of the Institute of Measurement and Control*, vol. 41, no. 14, pp. 3970 – 8, 2019.

[50] F. Ding, L. Xu, D. Meng, X.-B. Jin, A. Alsaedi, and T. Hayat, “Gradient estimation algorithms for the parameter identification of bilinear systems using the auxiliary model,” *Journal of Computational and Applied Mathematics*, vol. 369, p. 112575, 2020.

[51] F. Shujun, X. Ling, D. Feng, A. Ahmed, and H. Tasawar, “Correlation analysis-based stochastic gradient and least squares identification methods for errors-in-variables systems using the multiinnovation,” *International Journal of Control, Automation and Systems*, vol. 19, 2021.

[52] F. Ding, X. Liu, and T. Hayat, “Hierarchical least squares identification for feedback nonlinear equation-error systems,” *Journal of the Franklin Institute*, vol. 357, no. 5, pp. 2958 – 2977, 2020.

[53] T. Chen, M. S. Andersen, L. Ljung, A. Chiuso, and G. Pillonetto, “System identification via sparse multiple kernel-based regularization using sequential convex optimization techniques,” *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 2933 – 2945, 2014.

[54] M. Goemans and L. Tuncel, "When does the positive semidefiniteness constraint help in lifting procedures?" *Mathematics of Operations Research*, vol. 26, no. 4, pp. 796 – 815, 2001.

[55] M. Monsalve, J. Moreno, R. Escalante, and M. Raydan, "Selective alternating projections to find the nearest sdd+ matrix," *Applied Mathematics and Computation*, vol. 145, no. 2-3, pp. 205 – 20, 2003.

[56] A. A. Ahmadi and A. Majumdar, "Dsos and sdsos optimization: Lp and socp-based alternatives to sum of squares optimization," *2014 48th Annual Conference on Information Sciences and Systems, CISS 2014*, pp. Department of Electrical Engineering at Princeton University –, 2014.

[57] A. Majumdar, G. Hall, and A. A. Ahmadi, "A Survey of Recent Scalability Improvements for Semidefinite Programming with Applications in Machine Learning, Control, and Robotics," *arXiv e-prints*, p. arXiv:1908.05209, Aug 2019.

[58] J. Sun, "Provable nonconvex methods and algorithms (collection of papers)," https://sunju.org/research/nonconvex/,2019, accessed: 11-25-2019.

[59] H. Feng and J. Lavaei, "On the exponential number of connected components for the feasible set of optimal decentralized control problems," *2019 American Control Conference (ACC)*, pp. 1430 – 7, 2019.

[60] L. Vandenberghe, "Convex optimization techniques in system identification," *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 16, no. PART 1, pp. 71 – 76, 2012.

[61] A. Barzegar, D. K. Molzahn, and R. Su, "A method for quickly bounding the optimal objective value of an opf problem using a semidefinite relaxation and a local solution," *Electric Power Systems Research*, vol. 177, 2019.

[62] Z. Tian and W. Wu, "Recover feasible solutions for socp relaxation of optimal power flow problems in mesh networks," *IET Generation, Transmission and Distribution*, vol. 13, no. 7, pp. 1078 – 1087, 2019.

[63] D. Molzahn and I. Hiskens, "Mixed sdp/socp moment relaxations of the optimal power flow problem," *2015 IEEE Eindhoven PowerTech*, pp. 6 pp. –, 2015.

[64] R. Madani, S. Sojoudi, G. Fazelnia, and J. Lavaei, "Finding low-rank solutions of sparse linear matrix inequalities using convex optimization," *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 725 – 758, 2017.

[65] M. Kheirandishfard, F. Zohrizadch, M. Adil, and R. Madani, "Convex relaxation of bilinear matrix inequalities part ii: Applications to optimal control synthesis," *Proceedings of the IEEE Conference on Decision and Control*, vol. 2018-December, pp. 75 – 82, 2018.

[66] W. Favoreel, B. De Moor, and P. Van Overschee, "Subspace state space system identification for industrial processes," *Journal of Process Control*, vol. 10, pp. 149 – 155, 2000.

BIOGRAPHICAL STATEMENT

Adnan Nasir was born in Karachi, Pakistan. He received his B.E. degree in Electronics Engineering from the NED University of Engineering and Technology, Karachi, Pakistan, in 2002, and the M.E. degree in Electrical Engineering in the field of Wireless Sensor Networks from Nanyang Technological University (NTU), Singapore, in 2015. He worked towards his Ph.D. degree from 2017 to 2022, with the University of Texas at Arlington. His research interests include development of numerical algorithms for optimization and control, machine learning and data science.