Non-convex Quadratically Constrained Quadratic Programming: Hidden Convexity, Scalable Approximation and Applications

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Aritra Konar

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Nicholas D. Sidiropoulos, Advisor

SEPTEMBER, 2017

Acknowledgements

First and foremost, I wish to thank my advisor, Professor Nikos Sidiropoulos for providing me the opportunity to work with him. His immense knowledge, invaluable feedback, and dedication to research and teaching has served as an inspiration to me and has motivated me to strive to excel. I would like to thank him for all the time he invested in molding and developing my research, writing and presentation skills. It has truly been an honor and a privilege being his student.

I am also thankful to Professor Georgios Giannakis, Professor Jarvis Haupt, and Professor Shuzhong Zhang for agreeing to serve on my defense committee and am grateful for their support and valuable feedback. Furthermore, I am indebted to the efforts of the following professors at the University of Minnesota: Jarvis Haupt, Arindam Banerjee, Yousef Saad, Zhi-Quan Luo and Soheil Mohajer; their passion for teaching made an indelible impression on me in terms of what I aspire to be as a teacher. I would also like to thank Dr. Sandip Pal at Variable Energy Cyclotron Center, Kolkata, for introducing me to research and for teaching me the value of being rigorous and organized while conducting research.

My sincerest thanks go to my (former and current) labmates Omar Mehanna, Balasubramanian Gopalakrishnan, Kejun Huang, Xiao Fu, John Tranter, Artem Mosesov, Cheng Qian, Bo Yang, Charilaos Kanatsoulis, Ahmad Zamzam, Nikos Kargas, Faisal Almutairi, Panos Alevizos, Yunmei Shi and Mohamed Salah for our discussions on a broad variety of topics and for their help and support in easing the pressures of graduate life. Additional thanks also goes out to my friends Akshay Soni, Swayambhoo Jain, Tanvi Sharma, Panos Traganitis, Donghoon Lee, Yanning Shen, Vassilis Ioannidis and Dimitris Berberidis for making life in Minneapolis more enjoyable, as well as for all the fun times had at conferences. I am also very grateful to Akshay Soni and Swayambhoo Jain for serving as my unofficial mentors in my first year of graduate school. A word of thanks goes to my roommate Gaurav Chopra for putting up with me for all these years and for being a dependable friend. I would also like to give thanks to Ruoyu Sun, Meisam Razaviyayn and Miguel Vasquez for giving me the chance to collaborate and for helping me out in the course of my research. Finally, I would like to thank my friends Rohit

and Soumyajit, and my cousins Sutirtha and Shawon for all their support and encouragement during the course of my graduate studies.

Lastly, I would like to extend my deepest gratitude to my family: my parents Asokananda and Sutapa Konar, my sister Sudakshina, my brother-in-law Partho, and my nieces Parthivee and Sudiksha, whose unconditional love and unwavering support provides me with the strength to continue on my journey in life.

Dedication

To my family.

Abstract

Quadratically Constrained Quadratic Programming (QCQP) constitutes a class of computationally hard optimization problems that have a broad spectrum of applications in wireless communications, networking, signal processing, power systems, and other areas. The QCQP problem is known to be NP–hard in its general form; only in certain special cases can it be solved to global optimality in polynomial-time. Such cases are said to be convex in a hidden way, and the task of identifying them remains an active area of research. Meanwhile, relatively few methods are known to be effective for general QCQP problems. The prevailing approach of Semidefinite Relaxation (SDR) is computationally expensive, and often fails to work for general non-convex QCQP problems. Other methods based on Successive Convex Approximation (SCA) require initialization from a feasible point, which is NP-hard to compute in general.

This dissertation focuses on both of the above mentioned aspects of non-convex QCQP. In the first part of this work, we consider the special case of QCQP with Toeplitz-Hermitian quadratic forms and establish that it possesses hidden convexity, which makes it possible to obtain globally optimal solutions in polynomial-time. The second part of this dissertation introduces a framework for efficiently computing feasible solutions of general quadratic feasibility problems. While an approximation framework known as *Feasible Point Pursuit-Successive Convex Approximation* (FPP-SCA) was recently proposed for this task, with considerable empirical success, it remains unsuitable for application on large-scale problems. This work is primarily focused on speeding and scaling up these approximation schemes to enable dealing with large-scale problems. For this purpose, we reformulate the feasibility criteria employed by FPP-SCA for minimizing constraint violations in the form of non-smooth, non-convex penalty functions. We demonstrate that by employing judicious approximation of the penalty functions, we obtain problem formulations which are well suited for the application of first-order methods (FOMs). The appeal of using FOMs lies in the fact that they are capable of efficiently exploiting various forms of problem structure while being computationally lightweight. This endows our approximation algorithms the ability to scale well with problem dimension. Specific applications in wireless communications and power grid system optimization considered to illustrate the efficacy of our FOM based approximation schemes. Our experimental results reveal the surprising effectiveness of FOMs for this class of hard optimization problems.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and motivation

Quadratically Constrained Quadratic Programming (QCQP) constitutes an important class of computationally hard optimization problems with a broad spectrum of applications ranging from wireless communications and networking, (e.g., $\{\pm 1\}$ multiuser detection [1], multicast beamforming [2–4], and the MAXCUT problem [5]), to radar (e.g., robust adaptive radar detection [6], and optimum coded waveform design [7]), signal processing (e.g., parametric model-based power spectrum sensing [8], phase retrieval [9], unimodular quadratic programming [10]), power systems (e.g., optimal power flow [11] and power system state estimation [12]), and financial engineering [13]. In its general form, the QCQP problem can be expressed as

$$\min_{\mathbf{x}\in\mathbb{R}^N} \quad \mathbf{x}^T\mathbf{A}_0\mathbf{x} \tag{1.1a}$$

$$\text{s.t.} \quad \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m,\ \forall\ m \in [M_I] \tag{1.1b}$$

$$\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m = 0,\ \forall\ m \in [M_E] \tag{1.1c}$$

where $[M_I] := \{1, \cdots, M_I\}$ and $[M_E] := \{1, \cdots, M_E\}$ represent the inequality and equality constraint sets respectively. The matrices $\{\mathbf{A}_m\}_{m=1}^{M_I}$ and $\{\mathbf{C}_m\}_{m=1}^{M_E}$ are assumed to be symmetric (without loss of generality), while $\{b_m\}_{m=1}^{M_I}$ and $\{d_m\}_{m=1}^{M_E}$ are real numbers. We assume that the cost function (1.1a) is convex; i.e., $\mathbf{A}_0 \succeq 0$ (positive semidefinite). In the special case where $\mathbf{A}_m \succeq \mathbf{0},\ \forall m \in [M_I]$ and $M_E = 0$ (i.e., the equality constraints are absent), (1.1) is a convex optimization problem which can be solved to global optimality in polynomial-time using interior-point methods (IPMs) [14]. However, the general case of (1.1) is a non-convex optimization problem due to the presence of the quadratic equality constraints (1.1c) and the inequality constraints (1.1b) possibly involving indefinite/negative semidefinite matrices, and is known to

be NP–hard [15]. In fact, for an arbitrary instance of (1.1), even establishing (in)feasibility is NP–hard. Only in certain cases, involving a small number of non-convex constraints (e.g., see [16–25]) or special problem structure, or both (e.g., see [26–32]), can (1.1) be solved to global optimality in polynomial-time. We say that such instances possess the property of *hidden convexity*. Identifying such special instances which can be optimally solved in polynomial-time continues to be an active area of research.

For the general case of (1.1), the use of approximation algorithms is well motivated for the purpose of obtaining high quality, albeit sub-optimal solutions in polynomial-time. The prevailing approach is Semidefinite Relaxation (SDR) [33, 34]: upon defining $\mathbf{X} = \mathbf{x}\mathbf{x}^T$ and utilizing the cyclic property of the trace operator, (1.1) can be equivalently recast in the form of the following rank constrained Semidefinite Programming (SDP) problem.

$$\begin{aligned}
\underset{\mathbf{X}\in\mathbb{R}^{N\times N}}{\text{min.}} \quad & \text{Trace}(\mathbf{A}_0\mathbf{X}) && \text{(1.2a)}\\
\text{s.t.} \quad & \text{Trace}(\mathbf{A}_m\mathbf{X}) \leq b_m,\ \forall\ m \in [M_I] && \text{(1.2b)}\\
& \text{Trace}(\mathbf{C}_m\mathbf{X}) = d_m,\ \forall\ m \in [M_E] && \text{(1.2c)}\\
& \mathbf{X} \succeq \mathbf{0}, && \text{(1.2d)}\\
& \text{Rank}(\mathbf{X}) = 1 && \text{(1.2e)}
\end{aligned}$$

Upon dropping the non-convex rank constraint, we obtain the following rank-relaxed SDP problem

$$\begin{aligned}
\underset{\mathbf{X}\in\mathbb{R}^{N\times N}}{\text{min.}} \quad & \text{Trace}(\mathbf{A}_0\mathbf{X}) && \text{(1.3a)}\\
\text{s.t.} \quad & \text{Trace}(\mathbf{A}_m\mathbf{X}) \leq b_m,\ \forall\ m \in [M_I] && \text{(1.3b)}\\
& \text{Trace}(\mathbf{C}_m\mathbf{X}) = d_m,\ \forall\ m \in [M_E] && \text{(1.3c)}\\
& \mathbf{X} \succeq \mathbf{0} && \text{(1.3d)}
\end{aligned}$$

whose solution yields a lower bound on the optimal value of the cost function of (1.1). Note that (1.3) is the Lagrangian bi-dual of (1.1) [33], and can be solved efficiently using modern IPMs, at a worst case computational complexity of $O(N^{6.5})$ [35]. If the optimal solution of (1.3) is rank-1, then its principal component is the globally optimal solution for (1.1). However, solving (1.3) does not solve the original NP–hard problem (1.1) in general, i.e., the rank of the optimal solution of (1.3) is generally higher than 1. Hence, SDR is usually followed by a post-processing *randomization* step, in order to convert the optimal solution of (1.3) into an approximate feasible solution for (1.1). When $M_E = 0$ and all the matrices $\{\mathbf{A}_m\}_{m=1}^{M_I}$ are negative semidefinite, then any randomly generated point can be scaled up to satisfy the constraints of the QCQP (1.1);

finding a feasible solution using randomization is easy in this case, and the challenge is to find one that is close to optimum, see [2, 34]. In the general setting where $M_E > 0$, $\{\mathbf{A}_m\}_{m=1}^{M_I}$ are all indefinite, or when one deals with two-sided positive semidefinite constraints, SDR with randomization often fails to find a point that satisfies the constraints in (1.1). Hence, the overall effectiveness of SDR in obtaining approximate solutions for the general case of (1.1) appears to be limited – not to mention the potentially very high complexity incurred in solving the relaxed problem (1.3) in SDP form.

An alternative approach is to employ *successive convex approximation* (SCA), which is a general framework for approximating non-convex problems. Its application to non-convex QCQP is sometimes called the *convex-concave procedure* [15]: for the inequality constraints (1.1b), each quadratic term is separated into convex and concave parts, and the latter is replaced by a convex (usually linear) approximation around a feasible point. Meanwhile, each equality constraint in (1.1c) is simply converted into a pair of inequalities and the same approximation procedure is used. The resulting convex problem is solved to obtain the next iterate, which also serves as the approximation point for the subsequent iteration. Thus, one approximates the non-convex problem by solving a series of convex problems. Under certain technical assumptions, convergence of the procedure to a stationary point of (1.1) can be established [36–39]. The main drawback of this approach is that it requires initialization with a feasible point, which is also NP–hard to compute for general non-convex QCQP.

Hence, one can conclude that determining a feasible point of a non-convex QCQP problem is the critical step for any approximation algorithm to succeed. Recently, an algorithmic approach known as *Feasible Point Pursuit (FPP)-SCA* [40,41] was proposed specifically for this task. FPP-SCA uses SCA together with auxiliary slack variables to approximate the feasibility problem by a sequence of convex subproblems. The algorithm works with *any* choice of initialization, as the slack variables guarantee that each SCA subproblem is feasible at every step. Empirically, FPP-SCA demonstrates very good performance in attaining feasibility for general non-convex QCQP problems (QCQPs). Nevertheless, the algorithm is not without its drawbacks. For one, it is required to iteratively solve a sequence of convex optimization problems via IPMs, which can be computationally very demanding. In addition, eigen-decomposition of all the quadratic constraint matrices is required, followed by storing the positive and negative definite components in memory.

In addition to the above, a consensus-ADMM (C-ADMM) algorithm for general non-convex QCQPs has been proposed in [42], which can also be used for directly computing a feasible point. In this approach, (1.1) is decomposed into multiple QCQP subproblems, each with a single constraint, which ensures that each subproblem can be solved to global optimality. Thereafter,

consensus is used to enforce agreement amongst the solutions of the subproblems. The per-iteration complexity of C-ADMM is much lower than that of FPP-SCA, but the drawback is that C-ADMM can be very memory intensive, since it uses local copies of the global optimization variable (one for each constraint).

Due to their inherently large computational and memory footprint, FPP-SCA and C-ADMM are unsuitable for application on quadratic feasibility problems in large dimensions and/or with a large number of constraints. Hence, in spite of their empirical successes, these algorithms have limited application for use in large-scale scenarios, which are becoming more and more common. The major contribution of this thesis is on addressing these shortcomings by proposing high performance, scalable approximation algorithms for feasibility pursuit. A key ingredient in our algorithmic approach is the use of problem formulations whose structure is well suited for the application of first-order methods (FOMs). The appeal of using FOMs lies in the fact that they have minimal memory and computational requirements relative to other optimization schemes, which makes them well-suited for application on large-scale problems. Hence, in this thesis, we adopt a FOM based optimization approach for general quadratic feasibility problems, instead of resorting to FPP-SCA or C-ADMM. Our interest in pursuing this approach is partially motivated from recent work which established that FOMs work remarkably well (under certain conditions) for many important non-convex problems arising in low-rank matrix regression and structured matrix factorization [43–48], as well as generalized phase retrieval [49–51]. In addition, we also identify a special case of (1.1) which can be solved to global optimality via SDR.

1.2 Thesis Outline and Contributions

Chapter 2 considers non-convex QCQP with Toeplitz-Hermitian quadratics, and shows that it possesses hidden convexity: it can always be solved to global optimality in polynomial-time via SDR followed by a spectral factorization step. Under the additional assumption that the matrices are circulant, it is shown that the QCQP can be equivalently reformulated as a linear program, which can be solved very efficiently. These results have been reported in [52].

In subsequent chapters, we focus on FOM based approaches for efficiently computing feasible solutions for the general case of (1.1). Chapter 3 considers a special class of non-convex quadratic feasibility problems defined by the intersection of a convex, compact set and a system of quadratic inequalities with negative semidefinite matrices. In order to compute a feasible solution (when one exists), we pose the problem in the form of minimizing the point-wise maximum of a finite number of concave quadratic functions over a convex set. An SCA algorithm with provable convergence to the set of stationary points is then proposed for the problem. Departing from the standard approach of using IPMs to solve each SCA subproblem, we exploit

the structure inherent in each subproblem and apply specialized FOMs for efficiently computing solutions. Multicast beamforming is considered as an application example to showcase the effectiveness of the proposed algorithms, which achieve a very favorable performance-complexity trade-off relative to the existing state of the art. These results appear in [53, 54].

Chapter 4 considers the most general case of the quadratic feasibility problem where the goal is to find a feasible solution for a general system of quadratic inequalities and equalities defined by (1.1b)-(1.1c). For the purpose of developing low-complexity feasibility pursuit algorithms, we propose to do away with SCA altogether and use a modified reformulation of the optimization criterion employed by FPP-SCA, which is well-suited for direct application of FOMs. Our approach features low computational and memory requirements, which makes it well-suited for application to large-scale problems. While *a priori* it may appear that these benefits come at the expense of technical sophistication, rendering our approach too simple to even merit consideration for a non-convex and NP–hard problem, we provide compelling empirical evidence to the contrary. Experiments on synthetic as well as real world instances of non-convex QCQPs reveal the surprising effectiveness of FOMs compared to more established and sophisticated alternatives. These results have been published in [55, 56].

A summary of contributions and directions for future research is provided in Chapter 5.

1.3 Notational Conventions

Throughout this thesis, we adopt the following notation. Superscript ^{H} is used to denote the Hermitian transpose of a vector/matrix, while ^{T} denotes plain transposition. Capital boldface is reserved for matrices, while vectors are denoted by small boldface. Scalars are represented in the normal face, while calligraphic font is used to denote sets. We use the shorthand $[N]$ to represent the set of numbers $\{1, \cdots, N\}$. The set of natural numbers is indicated by $\mathbb{N}$. We denote the $N-$dimensional real Euclidean space by $\mathbb{R}^N$, while $\mathbb{R}^N_+$ and $\mathbb{R}^N_{++}$ represent the corresponding non-negative and positive orthants respectively. Meanwhile, the complex Euclidean space is denoted by $\mathbb{C}^N$. The convex hull of a finite number of points $\{\mathbf{x}_i\}_{i=1}^N$ is denoted as $\text{conv}(\mathbf{x}_i | \forall\, i \in [N])$. The domain of a function $f : \mathbb{R}^N \to \mathbb{R}$ is defined as $\text{dom} f = \{\mathbf{x} \in \mathbb{R}^N | f(\mathbf{x}) < +\infty\}$. If a function $f(.)$ is differentiable, its gradient and hessian are denoted by $\nabla f(.)$ and $\nabla^2 f(.)$ respectively. For convex, non-differentiable f, the subdifferential set at a point $\mathbf{x}$ is represented by $\partial f(\mathbf{x})$. For general f (differentiable or non-differentiable), the directional derivative at a point $\mathbf{x}$ in the direction $\mathbf{d}$ is given by $f'(\mathbf{x}; \mathbf{d})$. For a set $\mathcal{X} \subset \mathbb{R}^N$, we use $\bar{\mathcal{X}}$ to denote its closure and $\partial\mathcal{X}$ to represent its boundary. Finally, the empty set is denoted by $\emptyset$.

Chapter 2

Hidden Convexity in QCQP with Toeplitz-Hermitian Quadratics

2.1 Introduction

In this section, we consider the special case of the homogeneous QCQP problem

$$\min_{\mathbf{x}\in\mathbb{C}^N} \quad \mathbf{x}^H\mathbf{A}_0\mathbf{x} \tag{2.1a}$$

$$\text{s.t.} \quad \mathbf{x}^H\mathbf{A}_m\mathbf{x} \leq b_m,\ \forall\ m \in [M] \tag{2.1b}$$

where all the matrices $\{\mathbf{A}_m\}_{m=0}^M$ possess Toeplitz-Hermitian structure. No additional structure is assumed, except $\mathbf{A}_0 \succeq 0$ (so that we always have a valid minimization problem). Upon employing SDR for (2.1), we obtain the following rank-relaxed SDP problem

$$\min_{\mathbf{X}\in\mathbb{C}^{N\times N}} \quad \text{Trace}(\mathbf{A}_0\mathbf{X}) \tag{2.2a}$$

$$\text{s.t.} \quad \text{Trace}(\mathbf{A}_m\mathbf{X}) \leq b_m,\ \forall\ m \in [M] \tag{2.2b}$$

$$\mathbf{X} \succeq \mathbf{0} \tag{2.2c}$$

We now show that for this special case of QCQP

1. (In)feasibility of an arbitrary instance can always be established in polynomial-time using SDR; and
2. If the instance is feasible, then SDR can be used to obtain a globally optimal solution in polynomial-time as well.

Our proof uses the Toeplitz structure of the matrices to show the tightness of SDR, although simply solving SDR for this special case of the QCQP problem does not return a rank-1 solution in general. Instead, we use a relaxed SDP formulation for (2.1) based on representation of finite autocorrelation sequences (FAS) via Linear Matrix Inequalities (LMIs) to show the existence of a rank-1 solution, which is also shown to be equivalent to SDR. A rank-reduction technique based on spectral factorization is used to convert the higher rank solution of SDR into a feasible rank-1 solution with the same cost. The proof of tightness does not depend on the number of constraints M. The implication is that non-convex QCQP with Toeplitz-Hermitian quadratics is not NP–Hard, but in fact exactly solvable in polynomial time.

To the best of our knowledge, our work is the first to show that *any* non-convex QCQP with Toeplitz-Hermitian quadratics can be solved optimally. Special cases have been previously considered in [57,58], but none of them settled the *general* non-convex Toeplitz-Hermitian QCQP problem. Toeplitz quadratic minimization subject to Toeplitz *equality* constraints was considered in [57, p. 30] (each equality corresponds to a pair of inequalities with the same Toeplitz-Hermitian matrix). Another special case was investigated in [58], which considered positive-semidefinite Toeplitz-Hermitian quadratics and a special QCQP structure arising in multi-group multicast beamforming. In [58], the proof of existence of an optimal rank-1 solution uses the Caratheodory parametrization of a covariance matrix [59, p. 181], which is only valid for *positive-semidefinite* Toeplitz matrices. Our work can be considered as an extension of this result, since we prove the existence of an optimal rank-1 solution for *indefinite* Toeplitz matrices. In [58], the optimal solution is obtained from the solution of the SDP relaxation based on the LMI representation of FAS. We show that this problem is equivalent to SDR, which is cheaper computationally, and demonstrate how an optimal rank-1 solution can be obtained, which also solves the original non-convex QCQP problem.

Additionally, when all matrices are circulant (a special class of Toeplitz matrices), we further show that the QCQP problem (2.1) can be equivalently reformulated as a Linear Programming (LP) problem, which can again be solved to global optimality in polynomial-time, at a far lower computational complexity compared to the SDR approach.

2.2 QCQPs with Toeplitz Quadratic Forms

Consider a special case of (2.1) where the Hermitian matrices $\{\mathbf{A}_m\}_{m=0}^{M}$ are also Toeplitz. Each $\mathbf{A}_m$ can then be expressed as

$$\mathbf{A}_m = \sum_{k=-p}^{p} a_{m,k}\boldsymbol{\Theta}_k \tag{2.3}$$

where $p = N-1$, $\mathbf{\Theta}_k \in \mathbb{R}^{N\times N}$ is an elementary Toeplitz matrix with ones on the k^{th} diagonal and zeros elsewhere ($k=0 \leftrightarrow$ main diagonal, $k>0 \leftrightarrow$ super-diagonals, and $k<0 \leftrightarrow$ sub-diagonals), while $a_{m,k}$ represents the entries along the k^{th} diagonal; i.e., we have $A_m(i,j) = a_{m,k},\ \forall\ j-i=k \in \mathcal{K}$, where $\mathcal{K} := \{-p, \cdots, 0, \cdots, p\}$. Note that due to the Hermitian property, $a_{m,k} = a^*_{m,-k}, \forall\ k \in \mathcal{K}$. Using (2.3), we can express each quadratic term in (2.1) as $\mathbf{x}^H\mathbf{A}_m\mathbf{x} = \text{Re}(\mathbf{a}_m^T\mathbf{r})$, where $\mathbf{a}_m = [a_{m,0}, 2a_{m,1}\cdots, 2a_{m,p}]^T \in \mathbb{C}^N$, $\mathbf{r} = [r(0), \cdots, r(p)]^T \in \mathbb{C}^N$, $r(k) = \text{Trace}(\mathbf{\Theta}_k\mathbf{X})$, and $\mathbf{X} = \mathbf{x}\mathbf{x}^H$. Overall, we can now express (2.1) as

$$\min_{\mathbf{X},\mathbf{r}} \quad \text{Re}(\mathbf{a}_0^T\mathbf{r}) \tag{2.4a}$$

$$\text{s.t.} \quad \text{Re}(\mathbf{a}_m^T\mathbf{r}) \le b_m,\ \forall\ m \in [M] \tag{2.4b}$$

$$r(k) = \text{Trace}(\mathbf{\Theta}_k\mathbf{X}), \forall\ k \in \mathcal{K}_+ \tag{2.4c}$$

$$\mathbf{X} \succeq \mathbf{0}, \tag{2.4d}$$

$$\text{Rank}(\mathbf{X}) = 1 \tag{2.4e}$$

where $\mathcal{K} \supseteq \mathcal{K}_+ := \{0, \cdots, p\}$. Note that (2.4c) and (2.4d) $\Rightarrow r^*(k) = r(-k)$. Upon dropping the rank constraint, we obtain the following convex SDP relaxation.

$$\min_{\mathbf{X},\mathbf{r}} \quad \text{Re}(\mathbf{a}_0^T\mathbf{r}) \tag{2.5a}$$

$$\text{s.t.} \quad \text{Re}(\mathbf{a}_m^T\mathbf{r}) \le b_m,\ \forall\ m \in [M] \tag{2.5b}$$

$$r(k) = \text{Trace}(\mathbf{\Theta}_k\mathbf{X}), \forall\ k \in \mathcal{K}_+ \tag{2.5c}$$

$$\mathbf{X} \succeq \mathbf{0} \tag{2.5d}$$

Proposition 1. *For Toeplitz $\{\mathbf{A}_m\}_{m=0}^M$, problems* (2.2) *and* (2.5) *are equivalent.*

Proof. For any $\mathbf{X}, \mathbf{r}$ satisfying (2.5c) and (2.5d)

$$\text{Re}(\mathbf{a}_m^T\mathbf{r}) = a_{m,0}r(0) + 2\sum_{k=1}^{p}\text{Re}(a_{m,k}r(k)) \tag{2.6a}$$

$$= \sum_{k=-p}^{p} a_{m,k}r(k) \tag{2.6b}$$

$$= \sum_{k=-p}^{p} a_{m,k}\text{Trace}(\mathbf{\Theta}_k\mathbf{X}) \tag{2.6c}$$

$$= \text{Trace}((\sum_{k=-p}^{p} a_{m,k}\mathbf{\Theta}_k)\mathbf{X}) \tag{2.6d}$$

$$= \text{Trace}(\mathbf{A}_m\mathbf{X}) \tag{2.6e}$$

therefore we may replace every instance of $\text{Re}(\mathbf{a}_m^T\mathbf{r})$ in (2.5) (including the cost function $\leftrightarrow m = 0$) with $\text{Trace}(\mathbf{A}_m\mathbf{X})$. Then it becomes evident that $\mathbf{r}$ is completely determined by $\mathbf{X}$ via (2.5c).

Thus we may drop (2.5c) and simply compute $\mathbf{r}$ from the optimal $\mathbf{X}$. What remains is precisely (2.2). □

We next show that feasibility of any instance of (2.1) can always be checked in polynomial time by checking the feasibility of (2.5). Furthermore, if feasibility of (2.1) is established, then the optimal solution of (2.5) can always be transformed into a globally optimal solution of (2.1). Since (2.5) is equivalent to (2.2), a solution to (2.1) can also be obtained from a solution of (2.2), as we will soon show. Although solving (2.2) is more computationally efficient compared to (2.5) (since it has fewer variables and constraints), it is more convenient to establish the proof of the following claims by considering (2.5).

Proposition 2. *For Toeplitz* $\{\mathbf{A}_m\}_{m=0}^{M}$*, (in)feasibility of* (2.5) *is equivalent to (in)feasibility of* (2.1)*. Furthermore, if* (2.1) *is feasible, then it can be solved to global optimality in polynomial-time.*

Proof. Taken together, constraints (2.4c), (2.4d), (2.4e) constitute the LMI parametrization of the autocorrelation sequence $r(k)$ of an MA process of order p, and it has been shown in [60, Appendix A] that (2.4e) is redundant, in the sense that the set of feasible $(\mathbf{X}, \mathbf{r})$ defined by (2.4c), (2.4d), and (2.4e) is the same as that defined by (2.4c) and (2.4d) only. If a solution $\hat{\mathbf{X}}, \hat{\mathbf{r}}$ of (2.5) has $\text{Rank}(\hat{\mathbf{X}}) > 1$, then there exists a rank-1 matrix $\bar{\mathbf{X}}$ which defines the same sequence $\hat{\mathbf{r}}$, and such a rank-1 matrix can be obtained by determining a spectral factor $\bar{\mathbf{x}}$ of $\hat{\mathbf{r}}$ using a spectral factorization algorithm (e.g., see [61]) and setting $\bar{\mathbf{X}} = \bar{\mathbf{x}}\bar{\mathbf{x}}^H$. Spectral factorization is non-unique, but we only need one (e.g., the minimum phase) factor. □

The implication is that for the special case of (2.1) considered here, with Toeplitz-Hermitian quadratic forms, the problem is not NP–hard (as is general QCQP with non-convex Hermitian quadratic forms), but is in fact exactly solvable in polynomial-time using convex programming, followed by a simple post-processing step.

A globally optimal solution of (2.1) can also be obtained by solving (2.2) first, followed by defining the autocorrelation sequence $r(k) = \text{Trace}(\mathbf{\Theta}_k \mathbf{X}_{\text{opt}})\ \forall\ k \in \mathcal{K}$, where $\mathbf{X}_{\text{opt}}$ denotes the optimal solution of (2.2). Since solving (2.2) is equivalent to solving (2.5), determining a spectral factor of $\{r(k)\}_{k=-p}^{p}$ will yield the optimal solution to (2.1). This is the preferred approach since solving (2.2) is more computationally efficient compared to (2.5). Thus, for QCQPs with non-convex Toeplitz-Hermitian quadratic forms, the solution of SDR (which is not rank-1 in general), can always be converted into an optimal rank-1 solution via spectral factorization; SDR is tight.

2.3 QCQPs with Circulant Quadratic Forms

We now consider a more special case of (2.1) where the matrices $\{\mathbf{A}_m\}_{m=0}^M$ are circulant. Although circulant matrices are a subset of the set of Toeplitz matrices, and hence the results of the previous section apply, we show that by exploiting the circulant structure, the QCQP problem (2.1) can be equivalently reformulated as a LP problem which can again be solved to global optimality, at a lower complexity cost as compared to solving the SDP (2.2). Circulant matrices are diagonalized by the discrete Fourier Transform (DFT) matrix, i.e., we can write $\mathbf{A}_m = \mathbf{F}\mathbf{\Lambda}_m\mathbf{F}^H$, where $\mathbf{F} \in \mathbb{C}^{N\times N}$ is the unitary DFT matrix

$$\mathbf{F} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w_N & w_N^2 & \cdots & w_N^{N-1} \\ 1 & w_N^2 & w_N^4 & \cdots & w_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w_N^{N-1} & w_N^{2(N-1)} & \cdots & w_N^{(N-1)(N-1)} \end{bmatrix} \tag{2.7}$$

where $w_N = e^{-j\frac{2\pi}{N}}$ is the N^{th} root of unity, and $\mathbf{\Lambda}_m$ is a diagonal matrix of the eigenvalues of $\mathbf{A}_m$ obtained by taking the DFT of the first row of $\mathbf{A}_m$. Hence, each quadratic term in (2.1b) can be expressed as

$$\mathbf{x}^H\mathbf{A}_m\mathbf{x} = \mathbf{x}^H\mathbf{F}\mathbf{\Lambda}_m\mathbf{F}^H\mathbf{x} \tag{2.8a}$$

$$= \mathbf{y}^H\mathbf{\Lambda}_m\mathbf{y} \tag{2.8b}$$

$$= \sum_{i=1}^{N} \lambda_m(i)|y(i)|^2 \tag{2.8c}$$

with obvious notation. Define $z(i) = |y(i)|^2, \forall\ i \in [N]$, $\mathbf{z} = [z(1), \cdots, z(N)]^T$, and $\boldsymbol{\lambda}_m = [\lambda_m(1), \cdots, \lambda_m(N)]^T$. Then, we have

$$\mathbf{x}^H\mathbf{A}_m\mathbf{x} = \boldsymbol{\lambda}_m^T\mathbf{z},\ \forall\ m \in [M] \tag{2.9}$$

Similarly for the objective, we have $\mathbf{x}^H\mathbf{A}_0\mathbf{x} = \boldsymbol{\lambda}_0^T\mathbf{z}$. Putting everything together, we obtain the following formulation

$$\min_{\mathbf{z}\in\mathbb{R}^n} \quad \boldsymbol{\lambda}_0^T\mathbf{z} \tag{2.10a}$$

$$\text{s.t.} \quad \boldsymbol{\lambda}_m^T\mathbf{z} \le b_m,\ \forall\ m \in [M] \tag{2.10b}$$

$$\mathbf{z} \ge \mathbf{0} \tag{2.10c}$$

which is a LP problem in $\mathbf{z}$. Thus, by exploiting the fact that all circulant matrices are diagonalized by the *same* eigen-basis, we can equivalently reformulate the non-convex QCQP problem

(2.1) as the LP problem (2.10), which can be solved to global optimality very efficiently. If $\mathbf{z}_{\text{opt}}$ is the optimal solution of (2.10), then we define $\mathbf{y}_{\text{opt}}$ as $y_{opt}(i) = \sqrt{z_{opt}(i)}, \forall\ i \in [n]$. Since $\mathbf{F}$ is unitary, an optimal solution $\mathbf{x}_{\text{opt}}$ for (2.1) can be obtained as $\mathbf{x}_{\text{opt}} = \mathbf{F}\mathbf{y}_{\text{opt}}$. Again, the optimal solution is not unique since, from the definition of $\mathbf{z}$, all phase information about $\mathbf{y}$ is irrelevant.

2.4 Numerical Results

In order to illustrate our claims, we carried out the following simulation experiments in MATLAB on a 64-bit desktop with 8 GB RAM and a 3.40 GHz Intel CORE i7 processor. YALMIP was chosen as the modeling language and the MOSEK solver was used to solve the optimization problems.

In our first experiment, we consider a problem with $N = 10$ complex dimensions, $M \in \{20, 40, 60, 80, 100\}$, and all the matrices are Toeplitz-Hermitian. The following procedure was used to create each instance of the QCQP problem (2.1). Generating the Toeplitz-Hermitian constraint matrices $\{\mathbf{A}_m\}_{m=1}^{M}$ only requires specification of the first row of each matrix. Every such row is drawn randomly and independently from a complex, circularly symmetric Gaussian distribution with zero mean and covariance matrix equal to identity. The Hermitian part of each matrix was finally taken to obtain $\mathbf{A}_m$. An initial point $\mathbf{p}$ was randomly generated, while the each of the values $\{b_m\}_{m=1}^{M}$ were sampled randomly from a Gaussian distribution $b_m \sim \mathcal{N}(\mathbf{p}^H\mathbf{A}_m\mathbf{p}, 1)$. If $\mathbf{p}^H\mathbf{A}_m\mathbf{p} > b_m$, then both sides of the inequality are multiplied by -1 to obtain $\leq$ inequalities. In order to generate $\mathbf{A}_0$, a Gaussian random vector was randomly drawn from the previous distribution, and its one-sided deterministic autocorrelation sequence was calculated, which was then used to specify the first row of $\mathbf{A}_0$. This technique ensures that $\mathbf{A}_0$ is Toeplitz-Hermitian, and also positive semi-definite. In order to solve (2.1), the SDR problem (2.2) was solved first, followed by a spectral factorization step to obtain a globally optimal solution of (2.1). The results, depicted in Table 2.1, were obtained after averaging over 1000 Monte-Carlo trials.

The table reports the average difference between the cost of the rank-1 solution obtained from SDR followed by spectral factorization and the lower bound obtained from the (higher-rank) solution of SDR alone; plus the average execution time of the algorithm. It was observed that, in all instances, a feasible solution was obtained, at a very modest computational effort, which is extremely close to the SDR lower bound.

In our second experiment, we consider the case of the QCQP problem (1) where all the matrices are Circulant-Hermitian. The number of complex dimensions was set to be $N = 20$ and the constraints $M \in \{25, 50, 75, 100, 125\}$. In order to generate each of the constraint matrices $\{\mathbf{A}_m\}_{m=1}^{M}$, a random vector of eigen-values was independently sampled from a uniform

distribution in the interval $[-10, 10]$. Upon forming a diagonal matrix of the eigen-values, followed by pre- and post-multiplication by the unitary DFT matrix (as defined in (2.7)), we obtain $\mathbf{A}_m$. The right hand sides $\{b_m\}_{m=1}^M$ were generated and the sign of each inequality was fixed in the same manner as described in our previous experiment. $\mathbf{A}_0$ was also synthesized in a similar fashion as each $\{\mathbf{A}_m\}_{m=1}^M$, except that the vector of eigen-values was drawn randomly from a uniform distribution on the interval $[0, 20]$, in order to ensure positive semi-definiteness. Each problem instance was solved by both the LP approach and SDR followed by spectral factorization. The results are summarized in Tables 2.2 and 2.3.

M	20	40	60	80	100
Avg. Loss (dB $\times 10^{-5}$)	6.37	6.92	6.06	5.97	9.53
Execution Time (secs)	0.114	0.154	0.192	0.231	0.268

Table 2.1: Results using SDR + Spectral Factorization for Toeplitz quadratic QCQPs.

M	25	50	75	100	125
Avg. Loss (dB $\times 10^{-5}$)	71.3	6.32	8.28	2.32	3.63
Execution Time (secs)	0.228	0.274	0.327	0.386	0.451

Table 2.2: Results using SDR + Spectral Factorization for Circulant quadratic QCQPs.

M	25	50	75	100	125
Avg. Loss (dB $\times 10^{-6}$)	0.28	1.22	1.42	1.24	1.22
Execution Time (secs)	0.059	0.098	0.138	0.171	0.214

Table 2.3: Results using Linear Programming for Circulant quadratic QCQPs.

Each table reports the average loss incurred in the cost function when the cost of the solutions obtained via the respective methods are compared to the SDR lower bound, along with the average execution times. From the tables, it is seen that the obtained solutions from both methods are indeed the globally optimal solutions of the non-convex QCQP problem (2.1), as evidenced by the very low average loss. As expected, the LP approach is considerably faster as compared to the SDR method. In addition, it was empirically observed that the solution of the former method was more accurate as compared to that of the latter method, in the sense that the average loss was typically an order of magnitude smaller. Overall, this illustrates the benefit of using the LP approach over the SDR method for solving non-convex QCQPs with Circulant-Hermitian quadratic forms.

2.5 Conclusion

We considered QCQPs with Toeplitz-Hermitian quadratics and proved that they are exactly solvable in polynomial-time via SDP relaxation followed by spectral factorization. For circulant-Hermitian quadratics, it was shown that the QCQP problem can be reformulated as a LP problem, which can be solved very efficiently. Numerical experiments illustrated the main claims.

Chapter 3

Fast Feasibility Pursuit for a class of non-convex QCQP

3.1 Introduction

In this section, we consider the case of the following quadratic feasibility problem

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{find}} \quad \mathbf{x} \tag{3.1a}$$

$$\text{s.t.} \quad \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq -1, \forall\ m \in [M] \tag{3.1b}$$

where $\mathbf{A}_m \preceq \mathbf{0},\ \forall\ m \in [M]$, and $\mathcal{X} \subset \mathbb{R}^N$ is a convex, compact set. Note that (3.1) is a non-convex problem for which it is NP–hard to establish (in)feasibility in general. For the purpose of obtaining feasible solutions (when one exists), we recast (3.1) as

$$\min_{\mathbf{x}\in\mathcal{X}} \max_{m\in[M]} \mathbf{x}^T\mathbf{A}_m\mathbf{x} \tag{3.2}$$

Define $f(\mathbf{x}) := \max\limits_{m\in[M]} u_m(\mathbf{x})$, where $u_m(\mathbf{x}) := \mathbf{x}^T\mathbf{A}_m\mathbf{x},\ \forall\ m \in [M]$, and let $\mathbf{x}^*$ denote an optimal solution of (3.2). If $f(\mathbf{x}^*) \leq -1$, then feasibility of (3.1) is established and $\mathbf{x}^*$ corresponds to a feasible solution of (3.1). Otherwise, if $f(\mathbf{x}^*) > -1$, (3.1) is infeasible. However, (3.2) is a non-convex problem, which cannot be solved to global optimality in general. Even then, we note that the set $\mathcal{F} := \{\mathbf{x} \in \mathcal{X} | f(\mathbf{x}) \leq -1\}$ is equivalent to the feasible set of (3.1), i.e., any *sub-optimal* solution of (3.2) which lies in $\mathcal{F}$ is also feasible for (3.1). This observation motivates the following approach: although it is hard to solve (3.2) optimally, it suffices to determine *any* $\mathbf{x} \in \mathcal{F}$ to establish feasibility of (3.1). Thus, one can design approximation algorithms for (3.2), with the goal of obtaining sub-optimal solutions which lie in the set $\mathcal{F}$. We will adopt this

approach, utilizing the framework of SCA to design an approximation algorithm for feasibility pursuit. However, the non-convex nature of (3.2) implies that our approach is not guaranteed to yield solutions which lie in $\mathcal{F}$, even when (3.1) is known to be feasible (i.e., $\mathcal{F}$ is non-empty). At best, we hope to design an approximation algorithm which consistently yields feasible solutions for a large number of feasible instances of (3.1).

Our approach for developing fast SCA algorithms can be summarized as follows: at each iteration we construct a non-smooth, convex surrogate function of the non-convex cost function of (3.2) by locally linearizing each quadratic component about the current iterate. On replacing the cost function with its convex surrogate, at each iteration, we obtain a non-smooth, convex optimization subproblem. The solution of each subproblem is then used as the point about which we construct a convex surrogate function in the next iteration. We establish the global convergence of the iterates generated by the resulting SCA algorithm to the set of d-stationary solutions of (3.2).

The overall complexity of this algorithm depends on the cost incurred in solving each non-smooth, convex subproblem. The prevailing approach for solving each subproblem is to first convert it into an equivalent smooth representation and then use general purpose convex optimization solvers, which utilize interior-point methods, for obtaining solutions. However, such a smooth reformulation has undesirable consequences from a computational standpoint, since it introduces additional constraints. In addition, when the number of variables is large, then using interior-point methods to solve each subproblem can become very computationally expensive.

In order to reduce the complexity of solving each subproblem, we explore the idea of using FOMs for efficiently solving each subproblem in its non-smooth representation. It is shown that the cost function of each subproblem possesses special structure in the form of being expressible as the maximization of a bilinear function over a convex set. This is the key step, since this form of structure can be exploited by the *Nesterov smoothing* technique [62] and Nemirovski's *saddle point reformulation* approach [63] to allow the development of specialized FOMs for efficiently solving each SCA subproblem. These methods possess the desirable property of low per-iteration complexity and an improved iteration complexity over standard methods for non-smooth, convex-optimization (e.g., projected subgradient methods). In addition to these two methods, we also propose using an inexact version of the classical Alternating Direction Method of Multipliers (ADMM) algorithm [64–66] for solving each SCA subproblem, which drops expensive, exact variable updates in favor of computationally cheaper, albeit inexact updates. In order to test the performance of the proposed FOM-based SCA techniques, we apply them to the problem of designing a max-min fair multicast beamformer [2].

3.2 Preliminaries

We will use the following standard definitions from convex analysis [67].

- Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ denote a single-valued, convex, lower semi-continuous function such that $\text{dom} f \neq \emptyset$. Then, the *Fenchel conjugate* of f is defined as

$$f^*(\mathbf{y}) = \sup_{\mathbf{x} \in \mathbb{R}^n} \mathbf{x}^T \mathbf{y} - f(\mathbf{x}) \tag{3.3}$$

  Note that $f^*(\mathbf{y})$ is also convex since it corresponds to the point-wise supremum of linear functions in $\mathbf{y}$.

- Consider a multi-valued function $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$. F is said to be *monotone* if

$$(F(\mathbf{x}) - F(\mathbf{y}))^T (\mathbf{x} - \mathbf{y}) \geq 0, \forall\ \mathbf{x}, \mathbf{y} \in \text{dom} F \tag{3.4}$$

  For the special case of $n = 1$, (3.4) implies that F is a monotonically increasing function. Thus (3.4) can be interpreted as an extension of the concept of monotonicity to the general case of $n \geq 1$. As an example, the gradient of a convex, differentiable function is monotone.

- A multi-valued function F is said to be *strongly monotone* with parameter $\sigma \in \mathbb{R} > 0$ if

$$(F(\mathbf{x}) - F(\mathbf{y}))^T (\mathbf{x} - \mathbf{y}) \geq \sigma \|\mathbf{x} - \mathbf{y}\|_2^2, \forall\ \mathbf{x}, \mathbf{y} \in \text{dom} F \tag{3.5}$$

  As an example, the gradient of a strongly convex, differentiable function is strongly monotone.

- Let $\mathbf{y} \in \mathbb{R}^n$ and $\mathcal{X} \subseteq \mathbb{R}^n$ be a set. Then the distance of the point $\mathbf{y}$ from the set $\mathcal{X}$ is defined as

$$d(\mathbf{y}, \mathcal{X}) = \inf_{\mathbf{x} \in \mathcal{X}} \|\mathbf{y} - \mathbf{x}\|_2 \tag{3.6}$$

3.3 Successive Convex Approximation

SCA is an iterative procedure, where in each iteration, a convex surrogate function which serves as a tight, global upper bound of $f(\mathbf{x})$, is minimized. Starting from a feasible point $\mathbf{x}^{(0)} \in \mathcal{X}$, at every iteration $n \in \mathbb{N}$, we construct a convex majorization function of $f(\mathbf{x})$ about the current iterate $\mathbf{x} = \mathbf{x}^{(n)}$ in the following manner. Since $u_m(\mathbf{x})$ is concave $\forall\ m \in [M]$, on locally linearizing $u_m(\mathbf{x})$ about the point $\mathbf{x} = \mathbf{x}^{(n)}$, we obtain the following upper bound.

$$\begin{aligned} u_m(\mathbf{x}) &\leq u_m(\mathbf{x}^{(n)}) + \nabla u_m(\mathbf{x}^{(n)})^T (\mathbf{x} - \mathbf{x}^{(n)}) \\ &= (2\mathbf{A}_m \mathbf{x}^{(n)})^T \mathbf{x} - (\mathbf{x}^{(n)})^T \mathbf{A}_m \mathbf{x}^{(n)} \\ &= (\mathbf{c}_m^{(n)})^T \mathbf{x} + d_m^{(n)}, \forall\ m \in [M] \end{aligned} \tag{3.7}$$

where $\mathbf{c}_m^{(n)} := 2\mathbf{A}_m\mathbf{x}^{(n)} \in \mathbb{R}^N$, and $d_m^{(n)} := -(\mathbf{x}^{(n)})^T\mathbf{A}_m\mathbf{x}^{(n)} \in \mathbb{R}$, $\forall\ m \in [M]$. Following this step, we define the function $v(\mathbf{x}, \mathbf{x}^{(n)}) := \max\limits_{m\in[M]} \mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)}$, which possesses the following properties.

(A1) $v(\mathbf{x}, \mathbf{x}^{(n)})$ is a convex function in $\mathbf{x}$ (being piecewise linear).

(A2) $v(\mathbf{x}, \mathbf{x}^{(n)})$ is continuous in $(\mathbf{x}, \mathbf{x}^{(n)})$, but non-differentiable in $\mathbf{x}$.

(A3) $v(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) = f(\mathbf{x}^{(n)}),\ \forall\ \mathbf{x}^{(n)} \in \mathcal{X}$

(A4) $v(\mathbf{x}, \mathbf{x}^{(n)}) \geq f(\mathbf{x}),\ \forall\ \mathbf{x}, \mathbf{x}^{(n)} \in \mathcal{X}$

Properties (A3) and (A4) together imply that the piecewise linear function $v(\mathbf{x}, \mathbf{x}^{(n)})$ is an upper bound of the original function $f(\mathbf{x})$ which is tight at $\mathbf{x} = \mathbf{x}^{(n)}$. In every iteration n, we replace $f(\mathbf{x})$ by its piecewise linear approximation about $\mathbf{x}^{(n)}$ to obtain the non-smooth, convex subproblem

$$\min_{\mathbf{x}\in\mathcal{X}}\ \max_{m\in[M]}\ \mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)} \tag{3.8}$$

The overall algorithm is given by

Algorithm 1 : SCA

Initialization: Starting point $\mathbf{x}^{(0)} \in \mathcal{X}$, set $n := 0$.

Repeat

- Compute $\mathbf{x}^{(n+1)} \in \arg\min\limits_{\mathbf{x}\in\mathcal{X}} v(\mathbf{x}, \mathbf{x}^{(n)})$
- Set $n := n + 1$.

Until termination criterion is met

Note that a feasible starting point can always be determined since, by our assumption, $\mathcal{X}$ is a convex set. Furthermore, we have the following chain of inequalities

$$f(\mathbf{x}^{(n+1)}) \leq v(\mathbf{x}^{(n+1)}, \mathbf{x}^{(n)}) \leq v(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) = f(\mathbf{x}^{(n)}),\ \forall\ n \in \mathbb{N} \tag{3.9}$$

The first inequality stems from (A4), whereas the second inequality follows since $\mathbf{x}^{(n+1)}$ is an optimal solution of $v(\mathbf{x}, \mathbf{x}^{(n)})$, and the last equality is due to (A3). As a result, it can easily be seen that the algorithm produces a sequence of iterates with monotonically non-increasing cost. In addition, we have the following result, which establishes that the algorithm is also provably convergent.

Proposition 3. *The iterates generated by SCA globally converge to the set of d-stationary solutions of* (3.2)*; i.e., we have*

$$\lim_{n\to\infty} d(\mathbf{x}^{(n)}, \mathcal{X}^*) = 0$$

where $\mathcal{X}^*$ *is the set of d-stationary solutions of* (3.2)*.*

Proof. The proof is deferred to Appendix A.1. □

The main computational cost incurred is in solving a subproblem of the form (3.8) at each iteration. Since (3.8) does not have a closed form solution, we must resort to iterative methods for solving each subproblem. The standard procedure involves transforming (3.8) into its epigraph representation, which yields the following smooth optimization problem

$$\min_{t\in\mathbb{R},\, \mathbf{x}\in\mathcal{X}} \quad t \tag{3.10a}$$

$$\text{s.t.} \quad \mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)} \leq t, \forall\ m \in [M] \tag{3.10b}$$

Problem (3.10) can be solved using interior point methods at a worst case computational complexity of $O(N^3\sqrt{M+|\mathcal{X}|})$ [68], where $|\mathcal{X}|$ is the smallest number of constraints required to define $\mathcal{X}$. Coupled with the fact that each such problem has to be solved multiple times, it is clear that the overall cost incurred is expensive and can become a serious computational burden for large M or (especially) N. In hindsight, the high computational cost stems from equivalently reformulating (3.8) as a smooth optimization problem (3.10). This motivates the development of low-complexity alternatives for efficiently solving each non-smooth subproblem (3.8), which are described in the following section.

3.4 First-Order based Algorithms for SCA

As a first step, we impose the following condition regarding the convex set $\mathcal{X}$: we assume that $\mathcal{X}$ is *simple*; i.e., the Euclidean projection operator onto $\mathcal{X}$ can be computed very efficiently. Given this, one may consider using projected sub-gradient methods [69, 70] for solving (3.8), which are well suited for minimizing non-differentiable convex functions subject to simple convex constraints. These methods possess the desirable property of having low per-iteration complexity in contrast to interior-point methods. The main drawback of using such methods is their slow convergence rate, which ultimately limits the attainable accuracy. Indeed, if the constraint set is compact, then for appropriately chosen step-sizes, the number of iterations required to obtain an ϵ-optimal solution [1] is upper bounded by $O(\frac{1}{\epsilon^2})$. Furthermore, from results in Information-Based Complexity Theory [71], it is known that the number of iterations required to construct

[1] Here, we define ϵ-optimality in terms of the cost function.

an ϵ-optimal solution by a FOM, with knowledge of the value and subgradients of the cost function only, is no less than $O(\frac{1}{\epsilon^2})$.

Since the iteration outer bound of projected subgradient methods matches this lower bound, they are optimal in a minimax sense, which would seem to indicate that one cannot devise a FOM with a faster convergence rate for solving problems of the form (3.8). However, it is important to stress that projected subgradient does not utilize any specific structure the problem may possess. This implies that it may indeed be possible to devise FOMs which explicitly exploit problem structure to achieve ϵ-optimality in fewer iterations. Note that this observation is not a contradiction of the results of [71], since such FOMs are not oblivious to problem-specific structure.

We now demonstrate that the cost function of (3.8) possesses specific structure which can be judiciously exploited. First, we define the matrix $\mathbf{C}^{(n)} := [\mathbf{c}_1^{(n)}, \cdots, \mathbf{c}_M^{(n)}]^T \in \mathbb{R}^{M \times N}$ and the vector $\mathbf{d} = [d_1^{(n)}, \cdots, d_M^{(n)}]^T \in \mathbb{R}^M$. Then, $v(\mathbf{x}, \mathbf{x}^{(n)})$ can be equivalently expressed as

$$v(\mathbf{x}, \mathbf{x}^{(n)}) = \max_{m \in [M]} \ \mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)} \tag{3.11a}$$

$$= \max_{\substack{\mathbf{y} \geq \mathbf{0}, \mathbf{1}^T\mathbf{y}=1, \\ \mathbf{y} \in \mathbb{R}^M}} (\mathbf{C}^{(n)}\mathbf{x} + \mathbf{d}^{(n)})^T\mathbf{y} \tag{3.11b}$$

To see that this holds, note that (3.11b) corresponds to maximizing a linear function over the $M-$dimensional probability simplex. Hence, the maximum is attained at one of the vertices of the simplex, which are given by the canonical basis vectors of $\mathbb{R}^M$. From the definition of $\mathbf{C}^{(n)}$ and $\mathbf{d}^{(n)}$, it is then evident that the equivalence holds. Defining $\phi^{(n)}(\mathbf{x}, \mathbf{y}) := (\mathbf{C}^{(n)}\mathbf{x} + \mathbf{d}^{(n)})^T\mathbf{y}$ and Δ_M as the $M-$dimensional probability simplex, (3.8) can be equivalently reformulated as

$$\min_{\mathbf{x} \in \mathcal{X}} \max_{\mathbf{y} \in \Delta_M} \ \phi^{(n)}(\mathbf{x}, \mathbf{y}) \tag{3.12}$$

This special problem structure is the cornerstone of our algorithmic approach for solving (3.12), as it is well suited for the application of several specialized FOMs. We now discuss methods available in the existing optimization literature which are capable of solving problems of the form (3.12) efficiently.

3.4.1 Smoothing via Conjugation

We first point out that one can consider using the *log-sum-exp* function as a smooth surrogate for the cost function of (3.8), since log-sum-exp can be interpreted as a differentiable approximation of the point-wise maximum function [14, p. 72]. We now show that it is possible to rigorously derive a more general result.

In his seminal work [62], Nesterov considered the following problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad q(\mathbf{x}) \tag{3.13a}$$

$$\text{s.t.} \quad \mathbf{x} \in \mathcal{C} \tag{3.13b}$$

where $\mathcal{C} \subseteq \mathbb{R}^n$ is a simple, compact, convex set and $q : \mathbb{R}^n \to \mathbb{R}$ is a non-differentiable, Lipschitz continuous convex function which admits the following representation

$$\begin{aligned} q(\mathbf{x}) &= \sup_{\mathbf{y}\in\operatorname{dom} p} ((\mathbf{C}\mathbf{x}+\mathbf{d})^T\mathbf{y} - p(\mathbf{y})) \\ &= p^*(\mathbf{C}\mathbf{x}+\mathbf{d}) \end{aligned} \tag{3.14}$$

where $\mathbf{C} \in \mathbb{R}^{m\times n}, \mathbf{d} \in \mathbb{R}^m$, $p : \mathbb{R}^m \to \mathbb{R}$ is a closed, convex function with bounded domain and $p^*(\mathbf{x})$ denotes the Fenchel conjugate of $p(\mathbf{y})$. Hence, $q(\mathbf{x})$ belongs to the class of non-differentiable convex functions which can be expressed as Fenchel conjugates of other convex functions. Nesterov's technique for solving (3.13) is based on *smoothing*-constructing a smooth approximation of $q(\mathbf{x})$ which possesses a Lipschitz continuous gradient, followed by minimizing the approximation by an accelerated FOM [72]. We now succinctly summarize the details of this technique as presented in [73].

Let $r(\mathbf{y})$ be a continuous, strongly convex function defined over the closure of the domain of p such that $\inf_{\mathbf{y}\in\operatorname{dom} p} r(\mathbf{y}) = 0$. Then, consider the function

$$\begin{aligned} q_\mu(\mathbf{x}) &= \sup_{\mathbf{y}\in\operatorname{dom} p} ((\mathbf{C}\mathbf{x}+\mathbf{d})^T\mathbf{y} - p(\mathbf{y}) - \mu r(\mathbf{y})) \\ &= (p+\mu r)^*(\mathbf{C}\mathbf{x}+\mathbf{d}) \end{aligned} \tag{3.15}$$

where $(p+\mu r)^*(\mathbf{x})$ is the Fenchel conjugate of the strongly convex function $(p+\mu r)(\mathbf{y})$ and $\mu \in \mathbb{R}_{++}$. Nesterov established that the function $q_\mu(\mathbf{x})$ possesses the following properties.

(B1) $q_\mu(\mathbf{x})$ is well defined and differentiable at all $\mathbf{x}$, and $\nabla q_\mu(\mathbf{x})$ is Lipschitz continuous with Lipschitz constant $L_\mu \propto \frac{1}{\mu}$.

(B2) $q_\mu(\mathbf{x}) \le q(\mathbf{x}) \le q_\mu(\mathbf{x}) + \mu D$, where $D := \sup_{\mathbf{y}\in\operatorname{dom} p} r(\mathbf{y})$.

Hence, $q_\mu(\mathbf{x})$ can be interpreted as a smooth approximation of $q(\mathbf{x})$, where μ is a parameter which controls the level of smoothing. Replacing $q(\mathbf{x})$ by $q_\mu(\mathbf{x})$ in (3.13), we obtain the smooth optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad q_\mu(\mathbf{x}) \tag{3.16a}$$

$$\text{s.t.} \quad \mathbf{x} \in \mathcal{C} \tag{3.16b}$$

which can be solved to a numerical accuracy of ϵ_μ in $O(\sqrt{\frac{L_\mu}{\epsilon_\mu}})$ iterations using an accelerated FOM [62, 72]. It can also be shown that we have

$$q(\mathbf{x}) - q^*(\mathbf{x}) \leq q_\mu(\mathbf{x}) - q_\mu^*(\mathbf{x}) + \mu D \tag{3.17}$$

where $q^*(\mathbf{x})$ and $q_\mu^*(\mathbf{x})$ denote the optimal values of (3.13) and (3.16) respectively. If we define $\epsilon_\mu = q_\mu(\mathbf{x}) - q_\mu^*(\mathbf{x})$, then from (3.17) it is evident that an ϵ-optimal solution of (3.13) can be obtained by solving (3.16) to a numerical accuracy of $\epsilon_\mu = \epsilon - \mu D$; i.e., the smooth approximation (3.16) is solved to a higher degree of accuracy than the original non-smooth problem (3.13).

The role of the smoothing parameter μ is now discussed. Clearly, a smaller μ results in less smoothing which corresponds to a more accurate approximation of $q(\mathbf{x})$, but results in more iterations required to solve (3.16) via an accelerated FOM since L_μ is large. On the other hand, a larger μ produces a more smooth approximation (since L_μ is small), but results in having to solve (3.16) to a higher degree of accuracy in order to obtain an ϵ-optimal solution of (3.13). Overall, for a given ϵ, if we choose $\mu = \frac{\epsilon}{2D}$, it can be shown that using an accelerated FOM to solve (3.16), one can obtain an ϵ-optimal solution of (3.13) in no more than $O(\frac{1}{\epsilon})$ iterations, which represents an order of magnitude improvement over the $O(\frac{1}{\epsilon^2})$ iterations required by subgradient methods.

From (3.11), it is clear that Nesterov's smoothing technique can be applied to solve (3.12). Define the function $r(\mathbf{y}) := \sum_{m=1}^{M} y(m) \log y(m) + \log M$, which is continuous and strongly convex everywhere over Δ_M. Then, the smooth approximation of $v(\mathbf{x}, \mathbf{x}^{(n)})$ is given by

$$\begin{aligned} v_\mu(\mathbf{x}, \mathbf{x}^{(n)}) &= \sup_{\mathbf{y} \in \Delta_M} (\mathbf{C}^{(n)}\mathbf{x} + \mathbf{d}^{(n)})^T \mathbf{y} - \mu r(\mathbf{y}) \\ &= \mu \log\left(\sum_{m=1}^{M} \exp\left(\frac{\mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)}}{\mu} \right) \right) - \mu \log M \end{aligned} \tag{3.18}$$

where $\mu \in \mathbb{R}_+$ is the smoothing parameter. Note that if we set $\mu = 1$ and neglect the last term, we re-obtain the log-sum-exp function. Replacing $v(\mathbf{x}, \mathbf{x}^{(n)})$ by $v_\mu(\mathbf{x}, \mathbf{x}^{(n)})$ in (3.12), our optimization problem becomes

$$\min_{\mathbf{x} \in \mathcal{X}} \ \log\left(\sum_{m=1}^{M} \exp\left(\frac{\mathbf{c}_m^{(n)T}\tilde{\mathbf{x}} + d_m^{(n)}}{\mu} \right) \right) \tag{3.19}$$

which is a smooth optimization problem, and can be solved using an accelerated FOM. Utilizing Nesterov's smoothing technique to solve each SCA subproblem, the overall SCA algorithm is given by

Algorithm 2 : Nesterov SCA

Initialization: Starting point $\mathbf{x}^{(0)} \in \mathcal{X}$, set $n := 0$.

Repeat

- Compute $\mathbf{x}^{(n+1)} \in \arg\min_{\mathbf{x}\in\mathcal{X}} v_\mu(\mathbf{x}, \mathbf{x}^{(n)})$ using an accelerated FOM.
- Set $n := n + 1$.

Until termination criterion is met.

The per-iteration cost of an accelerated FOM is dominated by the cost of forming the gradient (since all projections can be computed in closed form), which incurs a modest cost of $O(MN)$ flops in our case. In order to obtain further savings in computation, at each SCA iteration n, one can warm-start the accelerated FOM with the current iterate $\mathbf{x}^{(n)}$.

3.4.2 Convex-Concave Saddle Point Reformulation

The following technique is attributed to Nemirovski [63]. Let $\mathcal{X} \subset \mathbb{R}^n, \mathcal{Y} \subset \mathbb{R}^m$ be convex, compact sets and let $\phi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ be a continuous function which is convex in $\mathbf{x} \in \mathbb{R}^n$ and concave in $\mathbf{y} \in \mathbb{R}^m$. Define the function $g(\mathbf{x}) = \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y})$ and consider the following problem

$$\min_{\mathbf{x}\in\mathcal{X}} g(\mathbf{x}) = \min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y}) \tag{3.20}$$

From Sion's minimax equality theorem [74], we have that

$$\min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y}) = \max_{\mathbf{y}\in\mathcal{Y}} \min_{\mathbf{x}\in\mathcal{X}} \phi(\mathbf{x}, \mathbf{y}) \tag{3.21}$$

which implies that the optimal solution $\mathbf{z}^* = (\mathbf{x}^*, \mathbf{y}^*) \in \mathcal{X} \times \mathcal{Y} := \mathcal{Z}$ of (3.20) corresponds to a saddle point of $\phi(\mathbf{x}, \mathbf{y})$, i.e., we have

$$\phi(\mathbf{x}^*, \mathbf{y}) \leq \phi(\mathbf{x}^*, \mathbf{y}^*) \leq \phi(\mathbf{x}, \mathbf{y}^*) \;\forall\; (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y} \tag{3.22}$$

Given a candidate solution $\bar{\mathbf{z}} = (\bar{\mathbf{x}}, \bar{\mathbf{y}}) \in \mathcal{Z}$, its degree of suboptimality can be evaluated via the following primal-dual gap

$$\epsilon_{\text{sad}}(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \max_{\mathbf{y}\in\mathcal{Y}} \phi(\bar{\mathbf{x}}, \mathbf{y}) - \min_{\mathbf{x}\in\mathcal{X}} \phi(\mathbf{x}, \bar{\mathbf{y}}) \tag{3.23}$$

Using (3.23), we can obtain the following inequality

$$\begin{aligned} g(\mathbf{x}) - \min_{\mathbf{x}\in\mathcal{X}} g(\mathbf{x}) &= \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y}) - \min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y}) \\ &\leq \max_{\mathbf{y}\in\mathcal{Y}} \phi(\mathbf{x}, \mathbf{y}) - \min_{\mathbf{x}\in\mathcal{X}} \phi(\mathbf{x}, \mathbf{y}) \\ &= \epsilon_{\text{sad}}(\mathbf{x}, \mathbf{y}) \end{aligned} \tag{3.24}$$

Hence, the $\mathbf{x}$ component of a point $(\mathbf{x}, \mathbf{y}) \in \mathcal{Z}$ for which $\epsilon_{\text{sad}}(\mathbf{x}, \mathbf{y}) \leq \epsilon$ also corresponds to an ϵ-optimal solution of (3.20). The overall problem of determining a saddle point $\mathbf{z}^* = (\mathbf{x}^*, \mathbf{y}^*) \in \mathcal{Z}$ of $\phi(\mathbf{x}, \mathbf{y})$ can be cast as solving the associated variational inequality

$$\Psi(\mathbf{z}^*)^T(\mathbf{z} - \mathbf{z}^*) \geq 0, \ \forall \ \mathbf{z} \in \mathcal{Z} \tag{3.25}$$

where we define the *saddle-point operator* Ψ as

$$\Psi(\mathbf{z}) := \Psi(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{y}) \\ -\nabla_{\mathbf{y}} \phi(\mathbf{x}, \mathbf{y}) \end{bmatrix} \tag{3.26}$$

In addition, it can be shown that Ψ is monotone and Lipschitz continuous on $\mathcal{Z}$ (see [63]). The field of variational inequalities is a well studied subject which finds application in diverse areas (see [75] and references therein) and can be analyzed using tools from fixed-point theory. Hence, one can attempt to solve (3.25) by a fixed-point iteration, such as the *generalized* projected gradient method [76]. However, the convergence of this method is not guaranteed (see [76] for counter-examples), unless Ψ is *strongly* monotone. This restriction can be overcome by using a modified version of the method, known as the *extragradient* algorithm [77], which only requires the assumption of monotonicity and Lipschitz continuity of Ψ to guarantee convergence to the solution of (3.25).

The idea of classical projected gradient descent was extended to non-Euclidean geometries by the *Mirror Descent* algorithm [78], [79], which uses a distance generating function to exploit the specific geometry of the constraint set. In [63], Nemirovski proposed a variant of the Mirror Descent algorithm, known as the *Mirror-Prox* algorithm, for solving variational inequalities of the form (3.25), which can be interpreted as a generalization of the extragradient algorithm to non-Euclidean geometries. We now summarize the details of the Mirror-Prox algorithm as presented in [80, Section 5.2.3].

Let the sets $\mathcal{X}$ and $\mathcal{Y}$ be endowed with norms $\|.\|_{\mathcal{X}}$ and $\|.\|_{\mathcal{Y}}$ respectively. Assume that $\phi(\mathbf{x}, \mathbf{y})$ is $(\beta_{11}, \beta_{12}, \beta_{21}, \beta_{22})$-smooth in the following sense.

$$\|\nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{y}) - \nabla_{\mathbf{x}} \phi(\mathbf{x}', \mathbf{y})\|_{\mathcal{X},^*} \leq \beta_{11} \|\mathbf{x} - \mathbf{x}'\|_{\mathcal{X}}, \tag{3.27a}$$

$$\|\nabla_{\mathbf{y}} \phi(\mathbf{x}, \mathbf{y}) - \nabla_{\mathbf{y}} \phi(\mathbf{x}, \mathbf{y}')\|_{\mathcal{Y},^*} \leq \beta_{22} \|\mathbf{y} - \mathbf{y}'\|_{\mathcal{Y}}, \tag{3.27b}$$

$$\|\nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{y}) - \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{y}')\|_{\mathcal{X},^*} \leq \beta_{12} \|\mathbf{y} - \mathbf{y}'\|_{\mathcal{Y}}, \tag{3.27c}$$

$$\|\nabla_{\mathbf{y}} \phi(\mathbf{x}, \mathbf{y}) - \nabla_{\mathbf{y}} \phi(\mathbf{x}', \mathbf{y})\|_{\mathcal{Y},^*} \leq \beta_{21} \|\mathbf{x} - \mathbf{x}'\|_{\mathcal{X}}, \tag{3.27d}$$

$$\forall \ \mathbf{z} = (\mathbf{x}, \mathbf{y}) \in \mathcal{Z}, \ \mathbf{z}' = (\mathbf{x}', \mathbf{y}') \in \mathcal{Z}$$

where $\|.\|_{\mathcal{X},^*}$ and $\|.\|_{\mathcal{Y},^*}$ denote the dual norms of $\|.\|_{\mathcal{X}}$ and $\|.\|_{\mathcal{Y}}$ respectively. Define $\Phi_{\mathcal{X}}(\mathbf{x})$ to be a *mirror map* for $\mathcal{X}$, which possesses the following properties [80, Section 4.1]

(C1) $\Phi_{\mathcal{X}} : \mathcal{D}_{\mathcal{X}} \to \mathbb{R}$, where $\mathcal{D}_{\mathcal{X}} \subset \mathbb{R}^n$ is a non-empty, convex open set which contains $\mathcal{X}$ in its closure (i.e., $\mathcal{X} \subset \overline{\mathcal{D}}_{\mathcal{X}}$) and $\mathcal{X} \cap \mathcal{D}_{\mathcal{X}} \neq \emptyset$.

(C2) $\lim_{\mathbf{x} \to \partial \mathcal{D}_{\mathcal{X}}} \|\nabla \Phi_{\mathcal{X}}(\mathbf{x})\| \to \infty$

(C3) $\Phi_{\mathcal{X}}(\mathbf{x})$ is strongly convex and continuously differentiable on $\mathcal{D}_{\mathcal{X}}$.

(C4) The *Bregman Divergence* associated with $\Phi_{\mathcal{X}}$ is defined as

$$D_{\Phi_{\mathcal{X}}}(\mathbf{x}, \mathbf{x}') := \Phi_{\mathcal{X}}(\mathbf{x}) - \Phi_{\mathcal{X}}(\mathbf{x}') - \nabla \Phi_{\mathcal{X}}(\mathbf{x}')^T(\mathbf{x} - \mathbf{x}'), \forall\ \mathbf{x}, \mathbf{x}' \in \mathcal{D}_{\mathcal{X}} \tag{3.28}$$

Similarly, define $\Phi_{\mathcal{Y}}(\mathbf{y})$ to be a mirror map for $\mathcal{Y}$. We now consider the mirror map $\Phi(\mathbf{z}) = \Phi(\mathbf{x}, \mathbf{y}) = \Phi_{\mathcal{X}}(\mathbf{x}) + \Phi_{\mathcal{Y}}(\mathbf{y})$ for $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, defined on $\mathcal{D} = \mathcal{D}_{\mathcal{X}} \times \mathcal{D}_{\mathcal{Y}}$. Define $\beta := \max(\beta_{11}, \beta_{12}, \beta_{21}, \beta_{22})$ and $\alpha := \frac{1}{2\beta}$. The Mirror-Prox algorithm is then given by the following steps

---

Algorithm 3 : Mirror Prox

Initialization: Define $\mathbf{z}_j = [\mathbf{x}_j^T, \mathbf{y}_j^T]^T, \mathbf{w}_j = [\mathbf{u}_j^T, \mathbf{v}_j^T]^T$, $\Psi(\mathbf{z}_j) = [\nabla_{\mathbf{x}} \phi(\mathbf{x}_j, \mathbf{y}_j)^T, -\nabla_{\mathbf{y}} \phi(\mathbf{x}_j, \mathbf{y}_j)^T]^T$, and $\Psi(\mathbf{w}_j) = [\nabla_{\mathbf{x}} \phi(\mathbf{u}_j, \mathbf{v}_j)^T, -\nabla_{\mathbf{y}} \phi(\mathbf{u}_j, \mathbf{v}_j)^T]^T$ for $j \geq 0$. Let $\mathbf{z}_0 = \arg\min_{\mathbf{z} \in \mathcal{Z} \cap \mathcal{D}} \Phi(\mathbf{z})$. Set $j := 0, \mathbf{w}_0 = \mathbf{z}_0$.

Repeat

1. $\nabla \Phi(\mathbf{w}'_{j+1}) = \nabla \Phi(\mathbf{z}_j) - \alpha \Psi(\mathbf{z}_j)$
2. $\mathbf{w}'_{j+1} = \nabla \Phi^{-1}(\nabla \Phi(\mathbf{z}_j) - \alpha \Psi(\mathbf{z}_j))$
3. $\mathbf{w}_{j+1} = \arg\min_{\mathbf{z} \in \mathcal{Z} \cap \mathcal{D}} D_{\Phi}(\mathbf{z}, \mathbf{w}'_{j+1})$
4. $\nabla \Phi(\mathbf{z}'_{j+1}) = \nabla \Phi(\mathbf{z}_j) - \alpha \Psi(\mathbf{w}_{j+1})$
5. $\mathbf{z}'_{j+1} = \nabla \Phi^{-1}(\nabla \Phi(\mathbf{z}_j) - \alpha \Psi(\mathbf{w}_{j+1}))$
6. $\mathbf{z}_{j+1} = \arg\min_{\mathbf{z} \in \mathcal{Z} \cap \mathcal{D}} D_{\Phi}(\mathbf{z}, \mathbf{z}'_{j+1})$
7. Set $j := j + 1$.

Until termination criterion is met.

Note that the functional $\nabla \Phi^{-1}$ exists and is well defined since the gradient of a strongly convex function is strongly monotone. In addition, the existence and uniqueness of the minimizers of steps 3) and 6) follow from properties (C2) and (C3) of mirror maps, respectively. The overall algorithm consists of two iterations of Mirror Descent. The first three steps of the algorithm (i.e., going from $\mathbf{z}_j$ to $\mathbf{w}_{j+1}$) correspond to a single step of Mirror Descent, whereas in the subsequent three steps, a similar procedure is followed, albeit with a slight difference; the algorithm again

starts from $\mathbf{z}_j$ (instead of $\mathbf{w}_{j+1}$), but uses an operator evaluation at $\mathbf{w}_{j+1}$ to obtain $\mathbf{z}_{j+1}$. If the mirror maps of $\mathcal{X}$ and $\mathcal{Y}$ are chosen to be $\frac{1}{2}\|\mathbf{x}\|_2^2$ and $\frac{1}{2}\|\mathbf{y}\|_2^2$ respectively, then it can be shown that Mirror-Prox reduces to the extragradient algorithm of [77]. In [63], Nemirovski established convergence of the ergodic average of the iterates $(\mathbf{x}_j, \mathbf{y}_j)$ generated by the algorithm. To be more specific, he proved the following sub-optimality bound in terms of the primal-dual gap.

$$\epsilon_{\text{sad}}\left(\frac{1}{T}\sum_{j=0}^{T-1}\mathbf{x}_j, \frac{1}{T}\sum_{j=0}^{T-1}\mathbf{y}_j\right) \leq O\left(\frac{1}{T}\right) \tag{3.29}$$

Combining this result with (3.24) implies an iteration outer bound of $O(\frac{1}{\epsilon})$ for guaranteeing convergence to an ϵ-optimal solution of (3.20).

Note that problem (3.12) fits the framework proposed by Nemirovski, since it corresponds to a smooth, *bilinear* saddle-point reformulation of the non-smooth problem (3.8). Define $\Phi(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|_2^2$ and $\Phi(\mathbf{y}) = \sum_{m=1}^{M} y(m)\log y(m)$ to be the mirror maps for the sets $\mathcal{X}$ and $\mathcal{Y} := \{\mathbf{y} \in \Delta_M\}$, respectively. Then, the mirror map $\Phi(\mathbf{z})$ defined $\forall\ \mathbf{z} = (\mathbf{x}, \mathbf{y}) \in \mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ is given by

$$\begin{aligned}\Phi(\mathbf{z}) = \Phi(\mathbf{x}, \mathbf{y}) &= \Phi(\mathbf{x}) + \Phi(\mathbf{y}) \\ &= \frac{1}{2}\|\mathbf{x}\|_2^2 + \sum_{m=1}^{M} y(m)\log y(m)\end{aligned} \tag{3.30}$$

from which it follows that

$$\nabla\Phi(\tilde{\mathbf{z}}) = \begin{bmatrix} \tilde{\mathbf{x}} \\ \log y(1) + 1 \\ \vdots \\ \log y(M) + 1 \end{bmatrix}, \nabla^{-1}\Phi(\mathbf{z}) = \begin{bmatrix} \mathbf{x} \\ \exp(y(1) - 1) \\ \vdots \\ \exp(y(M) - 1) \end{bmatrix} \tag{3.31}$$

Furthermore, the Bregman Divergence associated with $\Phi(\mathbf{z})$ can be expressed as

$$\begin{aligned}D_\Phi(\mathbf{z}, \mathbf{z}') &= \Phi(\mathbf{z}) - \Phi(\mathbf{z}') - \nabla\Phi(\mathbf{z}')^T(\mathbf{z} - \mathbf{z}') \\ &= \frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|_2^2 + \sum_{m=1}^{M} y(m)\log\frac{y(m)}{y'(m)} - \sum_{m=1}^{M}(y(m) - y'(m))\end{aligned} \tag{3.32}$$

where $\mathbf{z} = (\mathbf{x}, \mathbf{y}), \mathbf{z}' = (\mathbf{x}', \mathbf{y}') \in \mathcal{D}$. Thus, the non-Euclidean projection problem

$$\begin{aligned}&\min_{\mathbf{z}\in\mathcal{Z}\cap\mathcal{D}} D_\Phi(\mathbf{z}, \mathbf{z}') \\ =&\min_{\substack{\mathbf{x}\in\mathcal{X},\\ \mathbf{y}\in\Delta_M}} \frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|_2^2 + \sum_{m=1}^{M} y(m)\log\frac{y(m)}{y'(m)} - \sum_{m=1}^{M}(y(m) - y'(m))\end{aligned} \tag{3.33}$$

can be decomposed into the pair of problems

$$\min_{\mathbf{x}\in\mathcal{X}} \|\mathbf{x}-\mathbf{x}'\|_2^2 \tag{3.34a}$$

$$\min_{\mathbf{y}\in\Delta_M} \sum_{m=1}^{M} y(m)\log\frac{y(m)}{y'(m)} - \sum_{m=1}^{M}(y(m)-y'(m)) \tag{3.34b}$$

The first problem is an Euclidean projection onto $\mathcal{X}$, which can be solved in closed form, while the second problem involves a non-Euclidean projection onto the $M-$dimensional probability simplex, where "distances" are measured using the unnormalized Kullback-Leibler (KL) divergence. This problem admits a simple closed form solution [80, p. 302] given by

$$\mathbf{y} = \begin{cases} \mathbf{y}', & \mathbf{y}' \in \Delta_M \\ \frac{\mathbf{y}'}{\|\mathbf{y}'\|_1}, & \text{otherwise} \end{cases} \tag{3.35}$$

Furthermore, note that

$$\Psi(\mathbf{z}) = \begin{bmatrix} \nabla_{\mathbf{x}}\phi^{(n)}(\mathbf{x},\mathbf{y}) \\ -\nabla_{\mathbf{y}}\phi^{(n)}(\mathbf{x},\mathbf{y}) \end{bmatrix} = \begin{bmatrix} (\mathbf{C}^{(n)})^T\mathbf{y} \\ -(\mathbf{C}^{(n)}\mathbf{x}+\mathbf{d}^{(n)}) \end{bmatrix} \tag{3.36}$$

where the superscript n denotes the iteration index of the outer SCA loop. Finally, from (3.27), it can be verified that for a fixed n, we have $\beta_{11}=0, \beta_{22}=0, \beta_{12}=\beta_{21}=L$, where $L=\max_{m\in\mathcal{M}}\|\mathbf{c}_m^{(n)}\|_2$ is the Lipschitz constant of the functions $\mathbf{c}_m^{(n)T}\mathbf{x}+d_m^{(n)}, \forall\, m\in[M]$. Thus, we obtain the step size $\alpha=\frac{1}{2L}$.

It now only remains to solve (3.12) according to the steps of the Mirror-Prox algorithm (Algorithm 3) with the mirror maps and saddle-point operator (along with the associated quantities) as defined above. The cost of each iteration of the Mirror-Prox algorithm is dominated by the formation of the saddle-point operator $\Psi(\tilde{\mathbf{z}})$, which requires only $O(MN)$ flops; all projections are again closed form operations. The overall SCA algorithm is now given by

---

Algorithm 4 : Mirror-Prox SCA

Initialization: Starting point $\mathbf{x}^{(0)}\in\mathcal{X}$, set $n:=0$.

Repeat

- Compute $\mathbf{x}^{(n+1)}\in\arg\min_{\mathbf{x}\in\mathcal{X}}\max_{\mathbf{y}\in\Delta_M}\phi^{(n)}(\mathbf{x},\mathbf{y})$ using the Mirror-Prox algorithm.
- Set $n:=n+1$.

Until termination criterion is met.

3.4.3 Alternating Direction Method of Multipliers

We now propose an alternative low complexity method for solving each subproblem of SCA. Define the indicator function of the constraint set $\mathcal{X}$ as

$$I_{\mathcal{X}}(\mathbf{x}) := \begin{cases} 0, \ \mathbf{x} \in \mathcal{X} \\ \infty, \ \text{otherwise} \end{cases} \tag{3.37}$$

Then, consider the following equivalent reformulation of (3.8)

$$\min_{\mathbf{x}\in\mathbb{R}^N} \ v(\mathbf{x}, \mathbf{x}^{(n)}) + I_{\mathcal{X}}(\mathbf{x}) \tag{3.38a}$$

$$= \min_{\mathbf{x}\in\mathbb{R}^N} \ \omega(\mathbf{C}^{(n)}\mathbf{x}, \mathbf{x}^{(n)}) + I_{\mathcal{X}}(\mathbf{x}) \tag{3.38b}$$

where we have defined $\omega(\mathbf{z}, \mathbf{x}^{(n)}) := \max_{m\in[M]} \{z(m) + d^{(n)}(m)\}$ and $\mathbf{z} \in \mathbb{R}^M$. Thus, the constrained minimization problem (3.8) is equivalent to minimizing the sum of two non-smoooth, convex functions. In order to ease the burden of notation, we suppress the explicit dependence of $\omega(.,.)$ on $\mathbf{x}^{(n)}$ and equivalently express (3.38b) as

$$\min_{\mathbf{x}\in\mathbb{R}^N} \ \omega(\mathbf{C}^{(n)}\mathbf{x}) + I_{\mathcal{X}}(\mathbf{x}) \tag{3.39}$$

Defining $\mathbf{z} := \mathbf{C}^{(n)}\mathbf{x}$, we obtain the problem

$$\min_{\mathbf{x}\in\mathbb{R}^N, \mathbf{z}\in\mathbb{R}^M} \ \omega(\mathbf{z}) + I_{\mathcal{X}}(\mathbf{x}) \tag{3.40a}$$

$$\text{s.t.} \qquad \mathbf{C}^{(n)}\mathbf{x} - \mathbf{z} = \mathbf{0} \tag{3.40b}$$

which is in a form suitable for the Alternating Direction Method of Multipliers (ADMM) [64–66]; a method which combines the benefits of dual decomposition and augmented Lagrangian techniques into a simple but powerful algorithm. The advantage of ADMM is that it enables cost functions (which may be non-smooth) and constraints to be handled separately via variable splitting. This can yield very efficient updates that are amenable to distributed implementation, while requiring mild conditions for achieving convergence. The augmented Lagrangian of (3.40) is given by

$$L_\rho(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}) = \omega(\mathbf{z}) + I_{\mathcal{X}}(\mathbf{x}) + \boldsymbol{\lambda}^T(\mathbf{C}^{(n)}\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{C}^{(n)}\mathbf{x} - \mathbf{z}\|_2^2 \tag{3.41}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^M$ is the dual variable and ρ is the penalty parameter of the augmented Lagrangian. The ADMM updates for a given subproblem (3.40) are then given by

$$\begin{aligned}
\mathbf{x}_{j+1}^{(n)} &:= \arg\min_{\mathbf{x}} \; L_\rho(\mathbf{x}, \mathbf{z}_j^{(n)}, \boldsymbol{\lambda}_j^{(n)}) \\
&= \arg\min_{\tilde{\mathbf{x}}} \; I_{\mathcal{X}}(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{C}^{(n)}\mathbf{x} - \mathbf{z}_j^{(n)} + \tilde{\boldsymbol{\lambda}}_j^{(n)}\|_2^2 \qquad (3.42a) \\
\mathbf{z}_{j+1}^{(n)} &:= \arg\min_{\mathbf{z}} \; L_\rho(\mathbf{x}_{j+1}^{(n)}, \mathbf{z}, \boldsymbol{\lambda}_j^{(n)}) \\
&= \arg\min_{\mathbf{z}} \; \omega(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{z} - \mathbf{C}^{(n)}\mathbf{x}_{j+1}^{(n)} - \tilde{\boldsymbol{\lambda}}_j^{(n)}\|_2^2 \\
&= \mathrm{prox}_{\frac{\omega}{\rho}}(\mathbf{C}^{(n)}\mathbf{x}_{j+1}^{(n)} + \tilde{\boldsymbol{\lambda}}_j^{(n)}) \qquad (3.42b) \\
\tilde{\boldsymbol{\lambda}}_{j+1}^{(n)} &:= \tilde{\boldsymbol{\lambda}}_j^{(n)} + \mathbf{C}^{(n)}\mathbf{x}_{j+1}^{(n)} - \mathbf{z}_{j+1}^{(n)} \qquad (3.42c)
\end{aligned}$$

where the subscript $j \in \mathbb{N}$ is the ADMM iteration counter, the superscript n is the iteration counter of SCA, $\tilde{\boldsymbol{\lambda}} := \frac{1}{\rho}\boldsymbol{\lambda}$ represents the scaled dual variable and in (3.42b), we have defined the proximal operator [81] of a convex, proper, closed function $f : \mathbb{R}^n \to \mathbb{R}$ as

$$\mathrm{prox}_{\frac{f}{\rho}}(\mathbf{x}) = \arg\min_{\mathbf{y}} \; f(\mathbf{y}) + \frac{\rho}{2}\|\mathbf{y} - \mathbf{x}\|_2^2 \qquad (3.43)$$

The update of $\mathbf{z}$ can be computed efficiently since the proximal operator of $\omega(.)$ can be evaluated via a bisection search (refer to Appendix A.2). Although the proximal operator of $I_{\mathcal{X}}(.)$ can be evaluated in closed form (being the Euclidean projection operator for the simple set $\mathcal{X}$, which can be computed in closed form), the update of $\mathbf{x}$ has to be solved numerically due to the presence of the matrix $\mathbf{C}^{(n)}$ (unless $\mathbf{C}^{(n)}$ is the identity matrix or is orthogonal, neither of which hold in our case), which is undesirable from a computational complexity standpoint. Thus, we propose to use an inexact version of ADMM, known as *Linearized* ADMM (L-ADMM) [81, 82], which is specifically designed to solve problems of the form (3.39) using the proximal operators of $\omega(.)$ and $I_{\mathcal{X}}(.)$ to update the primal variables. The variable updates for the L-ADMM algorithm are given by

$$\begin{aligned}
\mathbf{x}_{j+1}^{(n)} &:= \mathrm{prox}_{\eta I_{\mathcal{X}}}(\mathbf{x}_j^{(n)} - \eta\rho\mathbf{C}^{(n)T}(\mathbf{C}^{(n)}\mathbf{x}_j^{(n)} - \mathbf{z}_j^{(n)} + \tilde{\boldsymbol{\lambda}}_j^{(n)})) \\
&:= \mathrm{proj}_{\mathcal{X}}(\mathbf{x}_j^{(n)} - \eta\rho\mathbf{C}^{(n)T}(\mathbf{C}^{(n)}\mathbf{x}_j^{(n)} - \mathbf{z}_j^{(n)} + \tilde{\boldsymbol{\lambda}}_j^{(n)})) \qquad (3.44a) \\
\mathbf{z}_{j+1}^{(n)} &:= \mathrm{prox}_{\frac{\omega}{\rho}}(\mathbf{C}^{(n)}\mathbf{x}_{j+1}^{(n)} + \tilde{\boldsymbol{\lambda}}_j^{(n)}) \qquad (3.44b) \\
\tilde{\boldsymbol{\lambda}}_{j+1}^{(n)} &:= \tilde{\boldsymbol{\lambda}}_j^{(n)} + \mathbf{C}^{(n)}\mathbf{x}_{j+1}^{(n)} - \mathbf{z}_{j+1}^{(n)} \qquad (3.44c)
\end{aligned}$$

where the parameters η and ρ are chosen to satisfy $0 < \eta\rho \leq \frac{1}{\|\mathbf{C}^{(n)}\|_2^2}$ [81, p. 158]. Note that the L-ADMM algorithm differs from standard ADMM in the update of $\mathbf{x}$ only, which now involves evaluating the projection onto the set $\mathcal{X}$ and can be computed in closed form. In L-ADMM, the standard update for $\mathbf{x}$ is modified by replacing the term $\frac{\rho}{2}\|\mathbf{C}^{(n)}\mathbf{x} - \mathbf{z}_j\|_2^2$ in the augmented

Lagrangian $L_\rho(\mathbf{x}, \mathbf{z}_j, \boldsymbol{\lambda}_j)$ (3.41) by $\rho(\mathbf{C}^{(n)T}\mathbf{C}^{(n)}\mathbf{x}_j - \mathbf{C}^{(n)T}\mathbf{z}_j)^T\mathbf{x} + \frac{\eta}{2}\|\mathbf{x}-\mathbf{x}_j\|_2^2$, i.e., linearization of $\frac{\rho}{2}\|\mathbf{C}^{(n)}\mathbf{x}-\mathbf{z}_j\|_2^2$ about $\mathbf{x}_j$ plus a quadratic regularization term. The result can be rearranged in the form of a proximal operator as in (3.44a).

In [83], the following convergence result of L-ADMM can be found. The authors reformulated the optimality condition of (3.40) into a variational inequality of the form

$$\text{find } \mathbf{w}^* \in \Omega \tag{3.45a}$$

$$\text{s.t. } \theta(\mathbf{u}) - \theta(\mathbf{u}^*) + (\mathbf{w}-\mathbf{w}^*)^T F(\mathbf{w}^*) \geq 0, \forall\, \mathbf{w} \in \Omega \tag{3.45b}$$

where we have defined $\mathbf{u} := [\mathbf{x}^T, \mathbf{z}^T]^T, \mathbf{w} := [\mathbf{x}^T, \mathbf{z}^T, \boldsymbol{\lambda}^T]^T$,
$\theta(\mathbf{u}) := \omega(\mathbf{z}) + I_{\mathcal{X}}(\mathbf{x})$, $\Omega := \mathbb{R}^N \times \mathbb{R}^M \times \mathbb{R}^M$ and $F(\mathbf{w}) = [-(\mathbf{C}^{(n)T}\boldsymbol{\lambda})^T, \boldsymbol{\lambda}^T, (\mathbf{C}^{(n)}\mathbf{x}-\mathbf{z})^T]^T$. Let $\bar{\mathbf{w}}_j := \frac{1}{T+1}\sum_{t=0}^{T}\mathbf{w}_j$ where $\mathbf{w}_j := [\mathbf{x}_{j+1}^T, \mathbf{z}_{j+1}^T, \boldsymbol{\lambda}_{j+1}^T]^T$ (here we drop the superscript n for ease of notation). Then, the number of iterations required so that

$$\theta(\mathbf{u}) - \theta(\bar{\mathbf{u}}_j) + (\mathbf{w}-\bar{\mathbf{w}}_j)^T F(\bar{\mathbf{w}}_j) \geq -\epsilon, \forall\, \mathbf{w} \in \Omega \tag{3.46}$$

is $O(\frac{1}{\epsilon})$ (in an ergodic sense) in the worst case. Meanwhile, analysis of the per iteration cost of L-ADMM reveals that all the required matrix-vector multiplications incur a cost of $O(MN)$ flops. The update of $\tilde{\mathbf{x}}$ is in closed form, while for the update of $\tilde{\mathbf{z}}$, computing the proximal operator of $\omega(.)$ via bisection search requires $O(M\log_2(\frac{D}{\epsilon_b}))$ operations, where D is the initial bisection interval and ϵ_b is the desired length of the final interval. Hence, it follows that L-ADMM can be used to solve each SCA subproblem efficiently. The overall SCA algorithm is given by

Algorithm 5 : L-ADMM SCA

Initialization: Starting point $\mathbf{x}^{(0)} \in \mathcal{X}$, set $n := 0$.

Repeat

- Compute $\mathbf{x}^{(n+1)} \in \arg\min_{\mathbf{x}\in\mathbb{R}^N} \omega(\mathbf{C}^{(n)}\mathbf{x}, \mathbf{x}^{(n)}) + I_{\mathcal{X}}(\mathbf{x})$ according to the L-ADMM updates (3.44).
- Set $n := n + 1$.

Until termination criterion is met.

If we define $\mathbf{p}^{(n)} := \mathbf{x}^{(n)}$ (where the superscript n is the SCA iteration counter), then the n^{th} L-ADMM subproblem can be warm-started by initializing $\mathbf{x}_1^{(n)} = \mathbf{p}^{(n)}$, $\mathbf{z}_1^{(n)} = \mathbf{z}_1^{(n-1)}, \tilde{\boldsymbol{\lambda}}_1^{(n)} = \tilde{\boldsymbol{\lambda}}_1^{(n-1)}$ (here the subscript 1 denotes the L-ADMM iteration counter). For the very first iteration of SCA, we use $\mathbf{z}_1^{(0)} = \mathbf{C}^{(0)}\mathbf{p}^{(0)}, \boldsymbol{\lambda}_1^{(0)} = \mathbf{0}$.

3.5 Application: Single Group Multicast Beamforming

3.5.1 Problem Formulation

In recent years, physical layer multicasting via transmit beamforming has emerged as a powerful technique for efficient audio and video streaming in multiuser, multiple-antenna wireless communications systems. Multicast beamforming exploits channel state information at the transmitter to assign different weights to the elements of a multiple antenna array in order to steer power in the directions of subscribers while limiting interference caused to other users and systems [2]. It is currently a part of the Evolved Multimedia Broadcast Multicast Service (eMBMS) in the Long-Term Evolution (LTE) standard. In the single-group multicasting scenario, all users are interested in the same data stream from the transmitter (Tx), with the result that the maximum common data rate is determined by the minimum received signal-to-noise ratio (SNR). Hence, the problem of maximizing the common data rate can be formulated as maximizing the minimum received SNR subject to transmit power constraints (max-min fair multicast beamforming).

More formally, consider a downlink transmission scenario where a single base station (BS) equipped with N antennas wishes to transmit common information to a group of M single-antenna users over the same frequency band. Assuming perfect channel state information (CSI) is available at the BS, the goal of multicast beamforming is to exploit this knowledge and the spatial diversity offered by the multiple transmit antennas to steer transmitted power towards the group of desired users while limiting leakage to nearby co-channel users and systems [2]. Let $\mathbf{w} \in \mathbb{C}^N$ denote the desired beamforming vector and $\mathbf{h}_m \in \mathbb{C}^N$ denote the downlink channel between the BS and the m^{th} user, which is modeled as complex, random vector that is flat in frequency and quasi-static in time. Using a unit-norm beamformer, the BS transmits a zero-mean , unit-variance, common information bearing signal x to all M users. The corresponding received signal at the m^{th} user is given by

$$y_m = \mathbf{h}_m^H \mathbf{w} x + z_m, \ \forall\, m \in [M] \tag{3.47}$$

where z_m is zero-mean, wide-sense stationary additive noise at the m^{th} receiver with variance σ_m^2, and is assumed to be independent of x and $\mathbf{h}_m$. The received signal-to-noise ratio (SNR) at the m^{th} user is then given by $\frac{|\mathbf{h}_m^H \mathbf{w}|^2}{\sigma_m^2} = \mathbf{w}^H \mathbf{R}_m \mathbf{w}$, where $\mathbf{R}_m := \frac{\mathbf{h}_m \mathbf{h}_m^H}{\sigma_m^2} \succeq \mathbf{0}, \forall\, m \in [M]$. Since all users are required to decode the same information bearing signal, the maximum common data rate attainable is determined by the minimum SNR. The objective is to maximize the minimum received SNR subject to unit-norm transmit sum power constraints, which leads to the following

max-min fair formulation

$$\max_{\mathbf{w}\in\mathbb{C}^N} \min_{m\in[M]} \mathbf{w}^H\mathbf{R}_m\mathbf{w} \tag{3.48a}$$

$$\text{s.t.} \quad \|\mathbf{w}\|_2 \leq 1 \tag{3.48b}$$

which is a non-convex QCQP problem. When $M \geq N$, (3.48) is NP–hard [2] in the worst case. Another variant of (3.48) seeks to minimize the transmitted power subject to user-specific quality of service (QoS) guarantees which are formulated in terms of minimum received SNR per user. From an optimization point of view, the two formulations are essentially equivalent [2].

Several algorithms have been proposed for obtaining approximate solutions to (3.48), ranging from SDR followed by randomization [2], to alternating maximization [84], and SCA [85, 86] (applied to the QoS version), which exhibit the best overall performance. Several low complexity algorithms also exist [87–89], although they are unable to match the performance of SCA. In [90], it was proposed to approximate (3.48) by a proportionally fair formulation and a first-order based *Multiplicative Update* (MU) algorithm was introduced, which was demonstrated to attain performance comparable to that of SCA at much lower complexity. This algorithm is the current state-of-the-art for solving (3.48) in the traditional multicasting scenario (i.e., when $M \geq N$).

Recently, massive MIMO [91, 92], which refers to the concept of equipping cellular base stations with a very large number of transmit antennas, has emerged as a very promising paradigm for possible implementation in a future 5G wireless broadband standard [93, 94]. When considering the multicasting problem in the context of such a scenario (i.e., when $M < N$), it is more practical to replace the transmit sum power constraint (3.48b) by per antenna power constraints (PAPCs), since every antenna in a large scale array will be equipped with its own power amplifier, whose linearity is the performance limiting factor. Hence the single-group multicasting problem in the massive MIMO setting can be expressed as [95]

$$\max_{\mathbf{w}\in\mathbb{C}^N} \min_{m\in[M]} \mathbf{w}^H\mathbf{R}_m\mathbf{w} \tag{3.49a}$$

$$\text{s.t.} \quad |w(i)|^2 \leq P_i, \ \forall\ i \in [N] \tag{3.49b}$$

where P_i denotes the power constraint for the i^{th} transmit antenna, $\forall\ i \in [N]$. Although (3.49) is a non-convex optimization problem, we are currently not aware of any result which establishes it as being NP–hard. Note that (2.5) is a QCQP problem of the form (2.1), since computing the projection onto the set of constraints (3.49b) decouples into computing the projection of each element of $\mathbf{w}$ onto its corresponding element-wise constraint, which admits a closed form solution. When N is large (i.e., of the order of hundreds), then solving (3.49) via SDR is very computationally demanding.

We now show that both (3.48) and (3.49) can be expressed in the form of (3.2). Towards

this end, we express both max-min fair formulations as

$$\underset{\mathbf{w}\in\mathcal{W}}{\text{max.}} \; \min_{m\in[M]} \; \mathbf{w}^H\mathbf{R}_m\mathbf{w} \tag{3.50}$$

where $\mathcal{W} \subset \mathbb{C}^N$ represents the desired power constraints. Note that in this case, the focus is not in determining feasibility, but rather, deriving benefit in maximizing the minimum SNR utility function. We now reformulate (3.50) in terms of real variables. Define $\tilde{\mathbf{w}} := [\mathbf{w}_r^T, \mathbf{w}_i^T]^T \in \mathbb{R}^{2N}$, $\mathbf{w}_r = \text{Re}\{\mathbf{w}\}, \mathbf{w}_i = \text{Im}\{\mathbf{w}\}$ and the matrices $\tilde{\mathbf{R}}_m \in \mathbb{R}^{2N\times 2N}$ as

$$\tilde{\mathbf{R}}_m = \begin{bmatrix} \text{Re}\{\mathbf{R}_m\} & -\text{Im}\{\mathbf{R}_m\} \\ \text{Im}\{\mathbf{R}_m\} & \text{Re}\{\mathbf{R}_m\} \end{bmatrix}, \; \forall\, m \in [M] \tag{3.51}$$

Note that $\mathbf{R}_m \succeq \mathbf{0}$ if and only if $\tilde{\mathbf{R}}_m \succeq \mathbf{0}$, $\forall\; m \in [M]$. Hence, (3.50) can be equivalently expressed in terms of real variables as

$$\underset{\tilde{\mathbf{w}}\in\tilde{\mathcal{W}}}{\text{max.}} \; \min_{m\in[M]} \; \tilde{\mathbf{w}}^T\tilde{\mathbf{R}}_m\tilde{\mathbf{w}} \tag{3.52}$$

where $\tilde{\mathcal{W}}$ is the representation of the feasible set $\mathcal{W}$ in terms of real variables. Furthermore, defining $\bar{\mathbf{R}}_m := -\tilde{\mathbf{R}}_m$, $\forall\, m \in [M]$, (3.52) can be equivalently expressed as

$$\underset{\tilde{\mathbf{w}}\in\tilde{\mathcal{W}}}{\text{min.}} \; \max_{m\in[M]} \; \tilde{\mathbf{w}}^T\bar{\mathbf{R}}_m\tilde{\mathbf{w}} \tag{3.53}$$

which is now in the form as (3.2). Hence, the SCA algorithms described in the previous section can be applied on (3.53) for the purpose of obtaining high quality solutions efficiently.

3.5.2 Numerical Results

In order to benchmark the performance of our proposed low-complexity SCA algorithms, we implemented a standard SCA algorithm where each subproblem was cast as a SoCP problem and solved with the MOSEK solver [96] in MATLAB using the modeling language YALMIP [97] as a parser. We implemented the Nesterov SCA algorithm in MATLAB using the optimization toolbox TFOCS [98] to solve each SCA subproblem via the accelerated FOM described in [72]. The Mirror-Prox SCA and L-ADMM SCA algorithms were implemented in MATLAB by straightforward coding. The Nesterov SCA and L-ADMM SCA make use of the warm-starting strategies described previously. For Nesterov SCA, we set the smoothing parameter $\mu = 1e^{-4}$, while in L-ADMM, we set $\epsilon_b = 1e^{-6}$, and, in each SCA iteration, we let $\eta = \frac{1}{\rho\|\mathbf{C}^{(n)}\|_2^2}$. The value of ρ used depended on the scenario under consideration (i.e., traditional multicast or massive MIMO multicast). In both scenarios, the downlink channels $\{\mathbf{h}_m\}_{m=1}^M$ were modeled as random vectors drawn from a complex, circularly symmetric, normal distribution with zero mean and identity covariance matrix and the noise variance was set to be 1 for all users. The SCA algorithms were

all initialized from the same starting point and run for a maximum of 20 iterations. For the FOM-based SCA algorithms, each subproblem was solved using 1000 iterations. All experiments were carried out on a Windows desktop with 4 Intel i7 cores and 8GB of RAM.

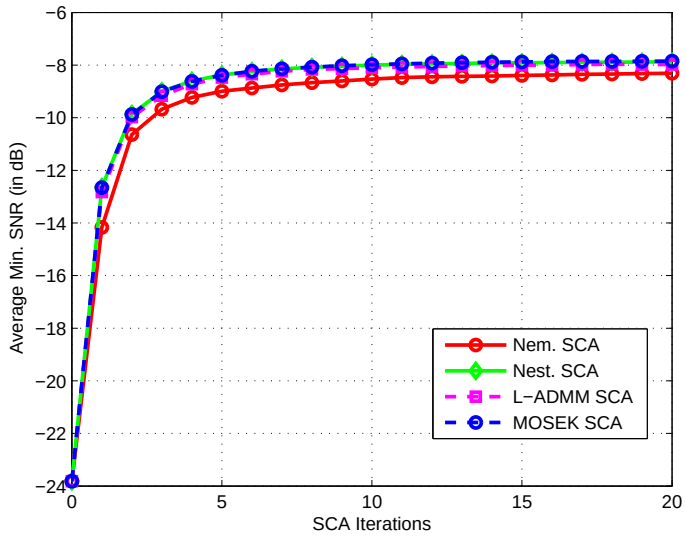


Figure 3.1: Comparison of average max-min SNR attained for $N = 10$ antennas, $M = 200$ users (traditional multicasting).

In a preliminary simulation, we considered a traditional multicasting scenario with $N = 10$ transmit antennas and $M = 200$ users. In this case, we set $\rho = 0.1$ in the L-ADMM method. For initializing our SCA algorithms, we considered the problem of maximizing the *average* SNR, which can serve as a reasonable starting point for further refinement [88]. In [99], Lopez demonstrated that the average SNR maximization problem in a multicasting scenario reduces to computing the principal eigenvector of the normalized channel correlation matrix $\mathbf{H} = \sum_{m=1}^{M} \frac{\mathbf{h}_m \mathbf{h}_m^H}{\sigma_m^2}$, and can be determined via the power method. Using the Lopez initialization as a starting point for our SCA algorithms, the results obtained after averaging over 200 channel realizations are depicted in Figure 3.1, which plots the average minimum SNR in dB as a function of the SCA iteration index. It is observed that the FOM-based SCA algorithms attain the same performance as the standard SCA algorithm which uses the MOSEK solver to solve each subproblem. The timing results are summarized in Table 3.1. Taken together, we observe that the Nemirovski SCA algorithm, which uses the Mirror-Prox algorithm to solve each SCA subproblem, exhibits the best overall performance in terms of speed and max-min SNR objective

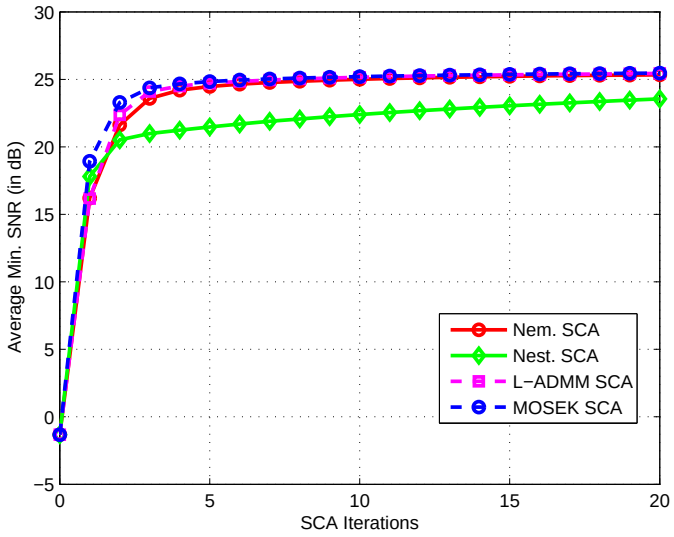


Figure 3.2: Comparison of average max-min SNR attained for $N = 200$ antennas, $M = 50$ users (massive MIMO multicasting).

function. The Nesterov SCA and L-ADMM algorithms exhibit slightly improved performance in terms of the objective value attained, but are more expensive compared to Nemirovski SCA; however, they are still less expensive compared to the standard SCA algorithm.

We also carried out a similar experiment for massive MIMO multicasting, with $N = 200$ antennas and 50 users. In this scenario, we replaced the sum power constraint by the PAPCs. We set $P_i = 0.33, \forall\, i \in \{1, \cdots, N\}$. A starting point that satisfies the PAPCs was randomly generated and was used to initialize the SCA algorithms. The value of ρ in the L-ADMM method was set to 0.01. All results were averaged over 200 channel realizations. The performance with respect to the value of the objective function attained is shown in Figure 3.2, while the timing results are depicted in Table 3.2. It can be seen that Nemirovski SCA and L-ADMM SCA match the performance of standard SCA in terms of average minimum SNR attained, but at much lower complexity (Nemirovski SCA in particular). Nesterov SCA did not perform as well in this regime with respect to the objective value attained, and was also the most expensive amongst the FOM-based SCA algorithms.

From these initial experiments, it is evident that using fast FOMs to solve the SCA subproblems allows us to effect a very favorable performance-complexity tradeoff, i.e., we attain the same performance as that of an interior-point method based SCA algorithm, but at much lower

Method	Stand. SCA	Nest. SCA	Nem. SCA	L-ADMM SCA
Avg. Time	10.24s	6.77s	1.94s	4.91s

Table 3.1: Timing results for traditional multicasting

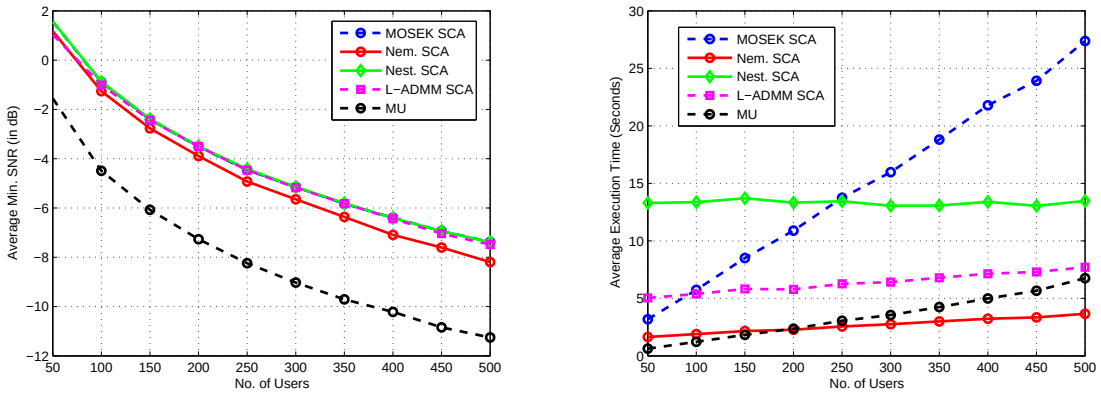
Method	Stand. SCA	Nest. SCA	Nem. SCA	L-ADMM SCA
Avg. Time	10.24s	6.77s	1.94s	4.91s

Table 3.2: Timing results for traditional multicasting

complexity.

We also carried out a more comprehensive experiment for both multicasting scenarios. First, we considered a traditional multicasting scenario where we fixed the number of transmit antennas $N = 25$ and increased the number of users M from 50 to 500. The algorithm parameters were set to be the same as previously indicated. We also added the MU algorithm in [90], which uses proportional fairness as a surrogate for max-min fairness, for comparison. Lopez initialization was again used for all algorithms. The MU algorithm was run for 1000 iterations. All results were obtained by averaging across 200 channel realizations for each value of M. The average minimum SNR attained (in dB) as a function of the number of users M and the timing results are depicted in Figure 3.3. From the figures, it is observed that Nesterov SCA and L-ADMM SCA methods always attain the same average minimum SNR as standard SCA, with Nemirovski SCA being only slightly worse. In terms of execution time, it is observed that as the number of users is increased, the time taken by standard SCA increases considerably (by almost an order of magnitude), while the execution times of the FOM-based SCA methods remains relatively constant.

Next, a massive MIMO multicasting scenario was considered with $M = 50$ users, and the number of transmit antennas N was increased from 50 to 500. The power budget of each antenna was set to be $P_i = 0.25, \forall\, i = \{1, \cdots, N\}$. We used the same choice of algorithm parameters for massive MIMO multicast as described previously. As an additional performance benchmark, we appropriately modified the MU algorithm to handle PAPCs (see Appendix A.3 for details). A randomly generated, feasible, starting point was used to initialize all the algorithms. The MU algorithm was run for 200 iterations in this case. All generated results were averaged across 200 channel realizations. The average minimum SNR attained is shown in Figure 5, while the average execution times are depicted in Figure 6. The figures demonstrate the state-of-the-art performance and computational gains offered by our proposed algorithms. The Nemirovski SCA and L-ADMM SCA algorithms attain the same performance as that of standard SCA but at significantly lower complexity. While the Nesterov SCA algorithm initially exhibits the best performance (when $N \leq 150$), it falls off as N is increased further. In terms of complexity, it is



(A) (B)

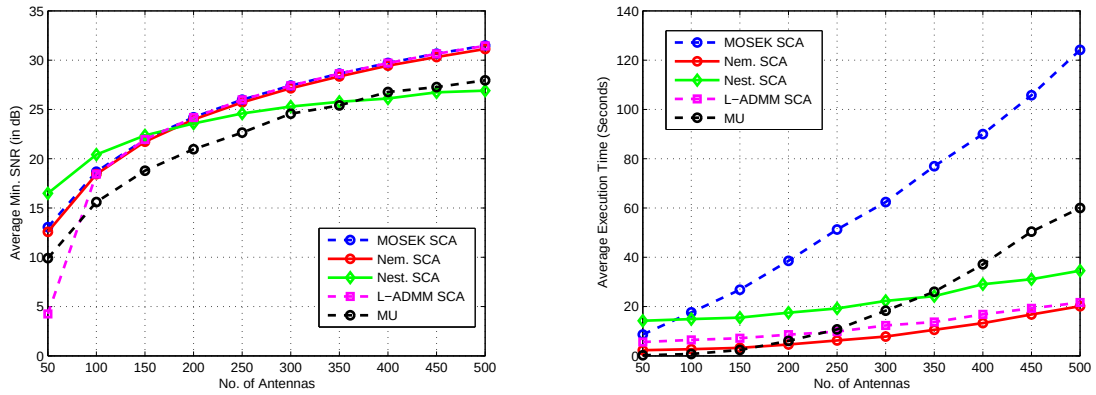
Figure 3.3: Traditional multicasting with $N = 25$ Tx antennas: (A) Average minimum SNR vs M (B) Average Execution Time vs M.

also slower than the other FOM-based SCA algorithms. Overall, as the number of antennas is increased, the timing curves for the proposed SCA algorithms increase very gracefully compared to that of the standard SCA (showcasing the drawback of using interior-point methods for solving large problems) and the MU algorithms. The Nemirovski SCA algorithm effects the best performance-complexity tradeoff in this regime. [2]

3.6 Conclusions

We considered a special class of non-convex quadratic feasibility problems defined by the intersection of a convex, compact set and a system of quadratic inequalities with negative semidefinite matrices. We expressed our feasibility criterion as minimizing the point-wise maximum of homogeneous, concave quadratics over a simple convex set. The development of SCA algorithms was pursued for obtaining high quality approximate solutions of this problem at low complexity. Our approach involves iteratively solving a sequence of convex approximations of the non-convex problem. Each subproblem is formulated as a non-smooth convex optimization problem, and solved using specialized FOMs which leverage the structure inherent in each subproblem to efficiently compute solutions at low overall complexity. This endows the algorithms with the ability to scale well to problems in high dimensions with a large number of constraints. The proposed algorithms were applied to the problem of single-group multicast beamforming. Simulations

[2] In this case, the MU algorithm does not scale very well to large dimensions. Hence, we defer from using it for last-mile refinement using Nemirovski SCA.



(A) (B)

Figure 3.4: massive MIMO multicasting with $M = 50$ users: (A) Average minimum SNR vs M (B) Average Execution Time vs M.

demonstrated that the algorithms offer substantial computational savings while attaining the same performance as standard SCA algorithms using interior-point methods to solve each SCA subproblem. These results are borne out of theoretical worst-case complexity considerations, which prove complexity reduction. Careful implementation of these algorithms in an appropriate lower-level programming language has the potential for deployment in real-time applications.

Chapter 4

Fast Feasibility Pursuit for general non-convex QCQP

4.1 Introduction

In this chapter, we consider the most general case of the quadratic feasibility problem, which can be expressed as

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{find}} \quad \mathbf{x} \tag{4.1a}$$

$$\text{s.t.} \quad \mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m \leq 0,\ \forall\ m \in [M_I] \tag{4.1b}$$

$$\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m = 0,\ \forall\ m \in [M_E] \tag{4.1c}$$

where $\mathcal{X} \subseteq \mathbb{R}^N$ is a simple, closed, convex set, while M_I and M_E represent the number of inequality and equality constraints respectively. The matrices $\{\mathbf{A}_m\}_{m=1}^{M_I}$ and $\{\mathbf{C}_m\}_{m=1}^{M_E}$ are assumed to be symmetric (without loss of generality), while $\{b_m\}_{m=1}^{M_I}$ and $\{d_m\}_{m=1}^{M_E}$ are real numbers. The general case of (4.1) is a non-convex optimization problem and is known to be NP–hard. We focus on the case where a feasible solution for (4.1) exists, and on developing polynomial-time algorithms for obtaining one. In order to establish the (in)feasibility of a given instance of (4.1), one may consider the following optimization problem.

$$\underset{\substack{\mathbf{x}\in\mathcal{X},\ \mathbf{s}_{\mathcal{I}}\in\mathbb{R}^{M_I},\\ \mathbf{s}_{\mathcal{E}}\in\mathbb{R}^{M_E}}}{\text{min.}} \quad \sum_{m=1}^{M_I} s_{\mathcal{I}}(m) + \sum_{m=1}^{M_E} s_{\mathcal{E}}(m) \tag{4.2a}$$

$$\text{s.t.} \quad \mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m \leq s_{\mathcal{I}}(m),\ s_{\mathcal{I}}(m) \geq 0,\ \forall\ m \in [M_I] \tag{4.2b}$$

$$-s_{\mathcal{E}}(m) \leq \mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m \leq s_{\mathcal{E}}(m),\ \forall\ m \in [M_E] \tag{4.2c}$$

where we have defined $\mathbf{s}_{\mathcal{I}} := [s_{\mathcal{I}}(1), \cdots, s_{\mathcal{I}}(M_I)]^T$ and $\mathbf{s}_{\mathcal{E}} := [s_{\mathcal{E}}(1), \cdots, s_{\mathcal{E}}(M_E)]^T$ as vectors of slack variables corresponding to the inequality and equality constraints respectively, with one slack variable being added to each constraint in order to ensure the feasibility of the overall problem. Note that the value of each slack variable corresponds to the degree of violation of the constraint with which it is associated. We impose an ℓ_1-penalty on the slack variables in order to promote sparsity of the constraint violations. If an optimal solution $(\mathbf{x}^*, \mathbf{s}^*_{\mathcal{I}}, \mathbf{s}^*_{\mathcal{E}})$ of (4.2) can be obtained for which $\mathbf{s}^*_{\mathcal{I}} = \mathbf{0}, \mathbf{s}^*_{\mathcal{E}} = \mathbf{0}$, then $\mathbf{x}^*$ is feasible for (4.1). Otherwise, (4.2) is infeasible and from the sparsity pattern of $\mathbf{s}^*_{\mathcal{I}}$ and $\mathbf{s}^*_{\mathcal{E}}$, we can determine the constraints which cause infeasibility. Nonetheless, computing an optimal solution of (4.2) remains a challenging proposition since it is non-convex and NP–hard in general. In [40], the SCA technique was used to approximate (4.2) via a sequence of convex subproblems. Starting from a random initialization point $\mathbf{x}^{(0)} \in \mathbb{R}^N$, at each SCA iteration $k \in \mathbb{N}$, a convex subproblem is obtained by constructing a convex *restriction* of the non-convex constraint set about the current iterate $\mathbf{x}^{(k)}$. This is accomplished by expressing each non-convex quadratic term as a difference of convex functions via eigen-decomposition of its associated matrix, followed by linearization of the non-convex term about $\mathbf{x}^{(k)}$. The resulting convex set can be expressed as

$$\mathcal{P}_r^{(k)} := \begin{cases} \mathbf{x}^T \mathbf{A}_m^{(+)} \mathbf{x} + 2\mathbf{x}^{(k)T} \mathbf{A}_m^{(-)} \mathbf{x} - \mathbf{x}^{(k)T} \mathbf{A}_m^{(-)} \mathbf{x}^{(k)} - b_m \leq s_{\mathcal{I}}(m), \forall\ m \in [M_I] \\ \mathbf{x}^T \mathbf{C}_m^{(+)} \mathbf{x} + 2\mathbf{x}^{(k)T} \mathbf{C}_m^{(-)} \mathbf{x} - \mathbf{x}^{(k)T} \mathbf{C}_m^{(-)} \mathbf{x}^{(k)} - d_m \leq s_{\mathcal{E}}(m), \forall\ m \in [M_E] \\ \mathbf{x}^T \mathbf{C}_m^{(-)} \mathbf{x} + 2\mathbf{x}^{(k)T} \mathbf{C}_m^{(+)} \mathbf{x} - \mathbf{x}^{(k)T} \mathbf{C}_m^{(+)} \mathbf{x}^{(k)} - d_m \geq -s_{\mathcal{E}}(m), \forall\ m \in [M_E] \end{cases} \tag{4.3}$$

where $\mathbf{A}_m := \mathbf{A}_m^{(+)} + \mathbf{A}_m^{(-)}$, $\mathbf{A}_m^{(+)} \succeq \mathbf{0}$ and $\mathbf{A}_m^{(-)} \prec \mathbf{0}, \forall\ m \in [M_I]$ and similarly, $\mathbf{C}_m := \mathbf{C}_m^{(+)} + \mathbf{C}_m^{(-)}$, $\mathbf{C}_m^{(+)} \succeq \mathbf{0}$ and $\mathbf{C}_m^{(-)} \prec \mathbf{0}, \forall\ m \in [M_E]$. Thus, at SCA iteration k, we obtain a convex optimization subproblem of the form

$$\mathbf{x}^{(k+1)} \in \underset{\substack{\mathbf{x} \in \mathcal{X} \cap \mathcal{P}_r^{(k)}, \\ \mathbf{s}_{\mathcal{I}} \in \mathbb{R}_+^{M_I},\ \mathbf{s}_{\mathcal{E}} \in \mathbb{R}^{M_E}}}{\arg\min} \sum_{m=1}^{M_I} s_{\mathcal{I}}(m) + \sum_{m=1}^{M_E} s_{\mathcal{E}}(m) \tag{4.4}$$

with the solution of the resulting problem being used for linearization in the next iteration. The overall algorithm has been termed as *Feasible Point Pursuit (FPP)-SCA*. Utilizing the theoretical results developed in [100] [1] , we can provide the following characterization of the sequence of generated iterates.

1. The iterate sequence has non-increasing cost.

2. Assuming there exists a convergent subsequence, then provided that Slater's Condition [100, Section 2.1.2] is satisfied at the limit of this subsequence, the limit point satisfies the KKT conditions of (4.2).

[1] To be precise, we verify that the constraint approximation functions defined in (4.3) satisfy the conditions laid out in [100, Assumption 1], followed by invoking [100, Theorem 1].

We point out that these theoretical results do not imply that FPP-SCA is guaranteed to converge to a feasible point of (4.1); only convergence to a KKT point of the feasibility problem (4.2) can be established. Nonetheless, FPP-SCA was empirically demonstrated to be highly successful in converging to a zero-slack solution (i.e., attain feasibility) in a finite number of iterations for various instances of (4.1). However, this comes at the expense of overall problem complexity. Each SCA subproblem (4.4) can be recast in Second-order Cone Programming (SoCP) form and solved via general purpose conic programming solvers (which use IPMs) at a worst case complexity of $O(N+M_I+M_E)^{3.5}$ [68], which is dependent on both N (the number of variables) and $M := M_I + M_E$ (the total number of constraints). It is evident that iteratively solving a sequence of SCA subproblems via such means can prove to be very computationally expensive for large N and/or M. Furthermore, it is required to compute the eigen-decomposition of the matrices $\{\mathbf{A}_m\}_{m=1}^{M_I}$ and $\{\mathbf{C}_m\}_{m=1}^{M_E}$ in order to separate each matrix into its positive and negative definite components, which are then stored in memory. Thus, the overhead in terms of memory can also prove to be quite substantial for large-scale problems.

With the aim of improving scalability and alleviating the aforementioned issues with FPP-SCA, we consider the possibility of using FOMs for feasible point pursuit. As a first step in this direction, we consider the following reformulation of the feasibility problem (4.2)

$$\min_{\mathbf{x}\in\mathcal{X}} \left\{ F(\mathbf{x}) := \sum_{m=1}^{M_I} (\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)^+ + \sum_{m=1}^{M_E} |\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m| \right\} \tag{4.5}$$

where $(x)^+ := \max\{x, 0\}$ and $|x|$ denotes the absolute value of x. Note formulations and (4.5) are equivalent at the globally optimal solutions of these two problems. The reformulation results in a problem where all the non-convex constraints of have been incorporated into the cost function which is composed of the sum of M non-convex, non-smooth functions; each of which measures the degree of violation of its corresponding constraint via a loss function (quadratic hinge-loss for the inequality constraints and absolute value for the equality constraints). In the literature, such a formulation is also known as an *exact penalty formulation* [101]. We note that (4.5) remains non-convex and is NP–hard in general. At this point, we can choose to proceed in one of the following two ways.

1. We can apply SCA on the exact penalty formulation (4.5) to obtain a convex optimization subproblem at each iteration, and then utilize FOMs to solve each subproblem at a reduced computational complexity relative to interior-point methods.

2. Alternatively, we can go one step further by eliminating the SCA procedure from consideration altogether and focus on tackling (4.5) *directly* via FOMs.

In subsequent sections, we describe in detail how to implement these approaches, followed by judicious experiments to demonstrate the viability of using FOMs as a competitive alternative

to pre-existing approaches [34, 40, 42] for determining feasible solutions of general non-convex QCQPs. In several of our experiments, we advocate using empirically chosen steps-sizes for our FOMs, which currently do not feature guaranteed convergence to the set of stationary solutions of our proposed optimization formulation. Such a choice is motivated by the following considerations: i) in several setups, the cost function we employ does not satisfy standard assumptions made in the analysis of FOMs, thereby precluding us from invoking pre-existing convergence results; ii) in other cases where these assumptions are satisfied, the requisite step-sizes are dependent on unknown constants which typically have to be estimated via crude means and ultimately result in very conservative step-sizes that exhibit poor practical performance; and most importantly, iii) a stationary point of our criterion is not guaranteed to be a feasible solution. In order to ensure convergence to a feasible solution (when one exists) using our framework, one has to establish convergence to the globally optimal solution of the non-convex cost function we utilize, which is NP–hard in general [102]. This implies that the standard metric of ensuring convergence to a stationary point using FOMs is not sufficient to guarantee recovery of a feasible solution in this case. While this may lead one to question the merit of adopting such a FOM based approach, we point out that this should not *a priori* be construed as being a glaring drawback. Indeed, these hardness results ultimately stem from the fact that establishing (in)feasibility of an arbitrary instance of QCQP is NP–hard in general, which implies that *all* possible polynomial-time approximation schemes are doomed to fail on certain instances of the feasibility problem under consideration.

These technical issues notwithstanding, one may also doubt the potency of our direct FOM based approach on the grounds that it is too simplistic for a problem which is non-convex and NP-hard in its general form. Hence, *a priori*, it may seem a foregone conclusion that the proposed approach is destined to perform poorly compared to sophisticated polynomial-time schemes [34, 40, 42] developed for this problem.

Given these ostensible drawbacks of our approach, the outcome of our experiments comes as a great surprise, as it reveals something entirely unanticipated: on synthetically generated feasible instances of large-scale non-convex QCQP, we provide compelling empirical evidence to demonstrate that the direct FOM based approach works remarkably well and outperforms pre-existing alternatives across all baselines. Additionally, we tested our FOMs on the problem of power system state estimation (PSSE) [12, 103], which is a real world problem arising in power systems engineering that entails solving a system of *non-random* quadratic equations, and is NP–hard in its general form [104]. Our numerical tests on standard power networks demonstrate that the FOMs can achieve very favorable performance (in terms of estimation error) at far lower complexity relative to competing alternatives.

4.2 Overview of First Order Methods

In this section, we provide a brief overview of the various FOMs which we propose to use for both direct (non-convex) and SCA (convex) approaches.

4.2.1 Direct Approach

Consider the following optimization problem of minimizing averages of finite sums

$$\min_{\mathbf{x}\in\mathcal{X}} \left\{ F(\mathbf{x}) := \frac{1}{M}\sum_{m=1}^{M} f_m(\mathbf{x}) \right\} \tag{4.6}$$

where $\mathcal{X} \subset \mathbb{R}^N$ is a convex, compact set and each $f_m : \mathbb{R}^N \to \mathbb{R}$ is a twice differentiable, non-convex function with $L-$Lipschitz continuous gradients [2] ; i.e, $\exists\, L \in \mathbb{R}_{++}$ for which

$$\|\nabla f_m(\mathbf{x}) - \nabla f_m(\mathbf{y})\|_2 \leq L\|\mathbf{x}-\mathbf{y}\|_2, \forall\, \mathbf{x},\mathbf{y} \in \mathcal{X} \tag{4.7}$$

When F is bounded below over $\mathcal{X}$, we can attempt to determine an approximate solution for (4.6) using the classical *gradient descent* (GD) algorithm which has the following update rule.

$$\mathbf{y}^{(k)} = \mathbf{x}^{(k-1)} - \frac{\alpha_k}{M}\sum_{m=1}^{M} \nabla f_m(\mathbf{x}^{(k-1)}) \tag{4.8a}$$

$$\mathbf{x}^{(k)} = \Pi_{\mathcal{X}}(\mathbf{y}^{(k)}), \forall\, k \in \mathbb{N} \tag{4.8b}$$

where $\Pi_{\mathcal{X}}(.)$ denotes the Euclidean projection operator onto $\mathcal{X}$ and $\alpha_k \in \mathbb{R}_{++}$ is the step-size in the k^{th} iteration.

Note that each step requires the computation of M gradients, and hence can be fairly expensive for large M. As a low complexity alternative, we can consider using *stochastic gradient descent* (SGD). The algorithm is iterative in nature, where at each iteration k we randomly draw an index m_k from a uniform distribution defined on the set $[M]$ and then apply the following update rule

$$\mathbf{y}^{(k)} = \mathbf{x}^{(k-1)} - \alpha_k \nabla f_{m_k}(\mathbf{x}^{(k-1)}) \tag{4.9a}$$

$$\mathbf{x}^{(k)} = \Pi_{\mathcal{X}}(\mathbf{y}^{(k)}), \forall\, k \in \mathbb{N} \tag{4.9b}$$

Note that the expectation $\mathbb{E}(\mathbf{y}^{(k)}|\,\mathbf{x}^{(k-1)})$ equals (4.8a) (where the expectation is taken with respect to the random variable m_k). Hence, the SGD updates (4.9) are equivalent to standard GD updates in expectation. The advantage of SGD is that the updates are $O(M)$ cheaper compared to GD since at each iteration, we only need to compute the gradient of a single component function.

[2] This in turn implies $F(\mathbf{x})$ is $L-$Lipschitz smooth since smoothness is preserved under convex combinations.

A third alternative, which has emerged recently, is *Stochastic Variance Reduced Gradient* (SVRG) [105,106]. The SVRG algorithm can be viewed as a hybrid between SGD and GD, and proceeds in multiple stages. In each stage s, SVRG defines a "centering" variable $\mathbf{y}_s$ from the output of the previous stage and computes its full gradient $\nabla F(\mathbf{y}_s)$. Next, a fixed number (say K) of modified inner SGD iterations are executed, where in each iteration $k \in \{1, \cdots, K\}$, an index m_k is drawn uniformly at random from $\mathcal{M}$ and the following update rule is used

$$\mathbf{x}_s^{(0)} = \mathbf{y}_s \tag{4.10a}$$

$$\mathbf{v}_s^{(k)} = \mathbf{x}_s^{(k-1)} - \alpha_s^{(k)}(\nabla f_{m_k}(\mathbf{x}_s^{(k-1)}) - \nabla f_{m_k}(\mathbf{y}_s) + \nabla F(\mathbf{y}_s)) \tag{4.10b}$$

$$\mathbf{x}_s^{(k)} = \Pi_{\mathcal{X}}(\mathbf{v}_s^{(k)}), \forall\, k \in [K] \tag{4.10c}$$

where the superscript k denotes the inner SGD iteration counter for stage s. Again, the expectation $\mathbb{E}(\mathbf{v}_s^{(k)} | \, \mathbf{x}_s^{(k-1)})$ equals (4.8a). Hence, in expectation, the SVRG updates are also the same as the GD updates. However, compared to SGD, SVRG uses a different unbiased gradient estimator which corrects the currently sampled gradient $\nabla f_{m_k}(\mathbf{x}_s^{(k-1)})$ by subtracting a bias term. The overall algorithm is given by

---

Algorithm 6 : SVRG

Initialization: Select number of stages S, update frequency K and step-size sequence. Randomly generate a starting point $\mathbf{z}_0 \in \mathcal{X}$.

Iterate: for $s = 1, 2, \cdots, S$

- Set $\mathbf{y}_s = \mathbf{z}_{s-1}$
- Compute $\mathbf{g}_s := \nabla F(\mathbf{y}_s)$
- Set $\mathbf{x}_s^{(0)} = \mathbf{y}_s$
- **Iterate:** for $k = 1, \cdots, K$
  Randomly pick $m_k \in [M]$ and update
  $\mathbf{v}_s^{(k)} = \mathbf{x}_s^{(k-1)} - \alpha_k^{(s)}(\nabla f_{m_k}(\mathbf{x}_s^{(k-1)}) - \nabla f_{m_k}(\mathbf{y}_s) + \mathbf{g}_s)$,
  $\mathbf{x}_s^{(k)} = \Pi_{\mathcal{X}}(\mathbf{v}_s^{(k)})$
- **End**
- Set $\mathbf{z}_s = \mathbf{x}_s^{(K)}$

End

Return: $\mathbf{z}_S$

The convergence behavior of these algorithms for non-convex problems is dictated by the

choice of step-size sequences. Regarding the convergence of GD for non-convex problems of the form (4.6), we first consider the case $\mathcal{X} = \mathbb{R}^N$. We define a point $\bar{\mathbf{x}} \in \mathbb{R}^N$ to be $\epsilon-$approximate stationary if $\|\nabla F(\bar{\mathbf{x}})\|_2^2 \leq \epsilon$. Assuming $F(\mathbf{x})$ is L-Lipschitz smooth on $\mathcal{X}$, then, GD with a $1/L$ step-size requires at most $O(\frac{L}{\epsilon})$ iterations to attain $\epsilon-$approximate stationarity [107, p. 29]. In the more general case of $\mathcal{X} \subset \mathbb{R}^N$, convergence rate guarantees can be established in terms of the *generalized projected gradient*, which is defined as

$$P_{\mathcal{X}}(\mathbf{x}, \alpha) := \frac{1}{\alpha}[\mathbf{x} - \mathbf{x}^+] \tag{4.11}$$

where $\mathbf{x}^+ := \arg\min_{\mathbf{u} \in \mathcal{X}} \nabla F(\mathbf{x})^T \mathbf{u} + \frac{1}{2\alpha}\|\mathbf{u} - \mathbf{x}\|_2^2$ for a given point $\mathbf{x} \in \mathbb{R}^N$ and step-size α. In [108, Lemma 3], it is shown that as $\|P_{\mathcal{X}}(\mathbf{x}, \alpha)\|_2$ diminishes, $\mathbf{x}^+$ approaches a stationary point of (4.6). For a constant $1/L$ step-size, it has been established [109, Corollary 1] that the number of iterations required for $\|P_{\mathcal{X}}(\mathbf{x}, \alpha)\|_2^2 \leq \epsilon$ is $O(\frac{L}{\epsilon})$ in the worst case. Since $F(\mathbf{x})$ is defined to be the average of M component functions, this translates into an iteration complexity bound of $O(\frac{ML}{\epsilon})$ for attaining ϵ-stationarity. While it is NP–hard in general to establish convergence to a local minimizer of a non-convex cost function [102], in the special case where $F(\mathbf{x})$ possesses the *strict saddle property* [3] and $\mathcal{X} = \mathbb{R}^N$, it has been shown [110] that GD with a constant step-size $< 1/L$ converges almost surely to a local minimizer.

Next, we consider the convergence behavior of SGD. In [111], it is shown that the $\alpha_k = \frac{1}{kL}$ step-size rule can be used to prove almost-sure (asymptotic) convergence of the SGD iterates to a stationary point. When $\mathcal{X} = \mathbb{R}^N$, an explicit iteration complexity upper bound for establishing convergence of SGD to an $\epsilon-$approximate stationary point ($\mathbb{E}[\|\nabla F(\mathbf{x})\|_2^2] \leq \epsilon$)[4] can also be derived, if, in addition to $L-$Lipschitz smoothness, the following assumption is made

$$\max_{m \in [M]} \{\|\nabla f_m(\mathbf{x})\|_2\} \leq \sigma, \forall\, \mathbf{x} \in \mathbb{R}^N \tag{4.12}$$

i.e., all component functions possess uniformly bounded gradients, which is also equivalent to each $f_m(\mathbf{x})$ being σ-Lipschitz continuous. Then, for a specifically chosen constant step-size, SGD requires $O(\frac{L\sigma^2}{\epsilon^2})$ iterations to obtain an $\epsilon-$approximate stationary point [112], [113, Theorem 1]. This choice of step-size requires knowing the total number of iterations beforehand, which may not be practical in all cases. Note that while this bound is independent of M, it depends on the variance of the stochastic gradients. Additionally, if $F(\mathbf{x})$ possesses the strict saddle property, then under certain conditions, convergence of SGD to a local minimizer can be established in (large) polynomial-time [114]. For the case $\mathcal{X} \subset \mathbb{R}^N$, an SGD algorithm with a randomized stopping criterion is described in [109] which achieves $\epsilon-$stationarity (i.e., $\mathbb{E}[\|\nabla P(\mathbf{x}, \alpha)\|_2^2] \leq \epsilon$) in at most $O(\frac{L\sigma^2}{\epsilon^2})$ iterations with a constant $1/2L$ step-size.

[3] i.e., the Hessian at every local minimizer is positive definite and at all other stationary points possesses *at least* one strictly negative eigenvalue.

[4] For stochastic iterative algorithms which make use of unbiased gradient estimators, the expectation is taken with respect to the stochasticity of the algorithm.

Finally, we focus on the convergence of SVRG for non-convex problems. In the unconstrained setting (i.e., $\mathcal{X} = \mathbb{R}^N$), assuming $F(\mathbf{x})$ is $L-$Lipschitz smooth, the references [113,115] have established that by properly tuning the algorithm parameters, the SVRG iterates require $O(\frac{M^{2/3}L}{\epsilon})$ iterations to converge to an ϵ-approximate stationary point of (4.6) (in expectation). The reference [116] considers the constrained case, and shows that SVRG requires $O(\frac{ML}{\epsilon})$ iterations to attain ϵ-approximate stationarity (with respect to $\mathbb{E}(\|P_{\mathcal{X}}(\mathbf{x}, \alpha)\|_2^2)$). With minibatching, this rate can be improved to $O(M + \frac{(M^{2/3})L}{\epsilon})$. However, the proof requires $F(\mathbf{x})$ to be *globally* L-smooth rather than being locally smooth over $\mathcal{X}$. We note that in spite of using randomly sampled gradients, the complexity bounds for SVRG are independent of the variance of the stochastic gradients due to the explicit variance reduction technique employed by SVRG.

4.2.2 SCA Approach

Consider the following optimization problem

$$\min_{\mathbf{x}\in\mathcal{X}} H(\mathbf{x}) \tag{4.13}$$

where $\mathcal{X} \subset \mathbb{R}^N$ is a compact, convex set and $H : \mathbb{R}^N \to \mathbb{R}$ is a non-smooth, Lipschitz continuous convex function. A standard method for solving such problems is *subgradient descent* (SD), which has the following update rule.

$$\mathbf{x}^{(k)} = \Pi_{\mathcal{X}}(\mathbf{x}^{(k-1)} - \alpha_k \mathbf{g}^{(k-1)}), \forall\, k \in \mathbb{N} \tag{4.14}$$

where $\mathbf{g}^{(k-1)} \in \partial H(\mathbf{x}^{(k-1)})$ is a subgradient drawn from the subdifferential set of the function H at the point $\mathbf{x}^{(k-1)}$.

In the special case $H(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} h_m(\mathbf{x})$, where each $h_m(\mathbf{x})$ is a non-smooth, Lipschitz continuous function, then we can alternatively use the technique of *stochastic subgradient descent* (SSD), which proceeds according to the following update rule

$$\mathbf{x}^{(k)} = \Pi_{\mathcal{X}}(\mathbf{x}^{(k-1)} - \alpha_k \mathbf{g}_{m_k}^{(k-1)}), \forall\, k \in \mathbb{N} \tag{4.15}$$

where the index m_k is drawn uniformly at random from the set $\mathcal{M}$ and $\mathbf{g}_{m_k}^{(k-1)}$ is a subgradient drawn from the subdifferential set of the function f_{m_k} at the point $\mathbf{x}^{(k-1)}$. Then, we have $\mathbb{E}(\mathbf{g}_{m_k}^{(k-1)}|\mathbf{x}^{(k-1)}) \in \partial F(\mathbf{x}^{(k-1)})$, which implies that the SSD iterations are equivalent to SD in expectation.

The convergence rates of these algorithms for convex problems are well established. For SD, selecting a step-size sequence $\alpha_k = O(\frac{1}{\sqrt{k}})$ guarantees a convergence rate of $O(\frac{1}{\sqrt{k}})$ in terms of the cost function, which is minimax optimal for this class of problems [71]. Using a similar step-size as SD together with iterate averaging, SSGD is also able to attain the same $O(\frac{1}{\sqrt{k}})$ convergence rate in expectation. However, the advantage of SSD lies in the fact that its individual

iterations are $O(M)$ faster since at every iteration we only need to compute a single subgradient.

This concludes our overview on FOMs. In the following section, we describe how to apply these methods to (4.5).

4.3 Proposed Algorithms

4.3.1 Direct Approach

We refrain from using SD/SSD on (4.5), since the sub-differential set of a non-smooth, non-convex function is not guaranteed to be non-empty at all points in its domain. This leaves us with GD, SGD and SVRG at our disposal. However, these methods are applicable to differentiable cost functions, whereas each component function of (4.5) is non-differentiable. Consequently, we propose to make the following modifications to (4.5).

First, consider the hinge-loss functions corresponding to the quadratic inequality constraints. Define $f_m(\mathbf{x}) := (\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)^+, \forall\ m \in [M_I]$. We now describe a procedure for constructing a smooth surrogate for each $f_m(\mathbf{x})$. Note that each $f_m(\mathbf{x})$ can be equivalently expressed as

$$f_m(\mathbf{x}) = \max_{0\leq y\leq 1} \{y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)\}, \forall\ m \in [M_I] \tag{4.16}$$

In order to construct a smooth surrogate of $f_m(\mathbf{x})$, consider the following modified version of (4.16)

$$f_m^{(\mu)}(\mathbf{x}) = \max_{0\leq y\leq 1} \{y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \mu\frac{y^2}{2}\}, \forall\ m \in [M_I] \tag{4.17}$$

where $\mu \in \mathbb{R}_{++}$ is a smoothing parameter. The maximization problem (4.17) can be solved in closed form to obtain the following equivalent smooth representation

$$f_m^{(\mu)}(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m \\ \frac{(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)^2}{2\mu}, & \text{if } b_m < \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m + \mu \\ \mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m - \frac{\mu}{2}, & \text{if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} > b_m + \mu \end{cases} \tag{4.18}$$

The derivation is relegated to Appendix A.4. Note that each $f_m^{(\mu)}(\mathbf{x})$ has continuous derivatives given by

$$\nabla f_m^{(\mu)}(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m \\ \frac{2(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)}{\mu}\mathbf{A}_m\mathbf{x}, & \text{if } b_m < \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m + \mu \\ 2\mathbf{A}_m\mathbf{x}, & \text{if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} > b_m + \mu \end{cases} \tag{4.19}$$

Hence, $f_m^{(\mu)}(\mathbf{x})$ is a smooth surrogate of $f_m(\mathbf{x}), \forall\, m \in \mathcal{M}_{\mathcal{I}}$. Furthermore, it can be shown that the following approximation bounds hold (see Appendix A.4).

$$f_m^{(\mu)}(\mathbf{x}) \leq f_m(\mathbf{x}) \leq f_m^{(\mu)}(\mathbf{x}) + \frac{\mu}{2}, \forall\, \mathbf{x} \in \mathbb{R}^N, \forall\, m \in [M_I] \tag{4.20}$$

The smoothing technique employed in (4.17) can be viewed as an extension of *Nesterov smoothing* [62] to the non-convex case setting; note that the representation of $f_m^{(\mu)}(\mathbf{x})$ in (4.17) does not correspond to the Fenchel conjugate of a strongly-convex function. As for the absolute value penalty functions $g_m(\mathbf{x}) := |\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m|, \forall\, m \in [M_E]$ in (4.5) corresponding to the equality constraints, we propose to replace them with quadratic penalty functions of the form

$$g_m^{(q)}(\mathbf{x}) := (\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m)^2, \forall\, m \in [M_E] \tag{4.21}$$

Following these steps, we obtain a non-convex, differentiable penalty formulation given by

$$\min_{\mathbf{x}\in\mathcal{X}} \left\{ F^{(s)}(\mathbf{x}) := \frac{1}{M}\left(\sum_{m=1}^{M_I} f_m^{(\mu)}(\mathbf{x}) + \sum_{m=1}^{M_E} g_m^{(q)}(\mathbf{x}) \right) \right\} \tag{4.22}$$

which is now in a form suitable for application of GD, SGD and SVRG. The convergence behavior of these algorithms is determined by the choice of the step-size sequence (and the additional parameters in the case of SVRG). As mentioned before, several theoretical results have appeared recently which establish non-asymptotic rates of convergence of these algorithms to a stationary point of non-convex problems of the form (4.6) for appropriate choices of step-size sequences and other parameters. However, applying these results to the problem under consideration is hampered by the following technical issues.

First, we point out that the cost function of (4.22) is a quartic polynomial which is neither globally Lipschitz continuous, nor does it possess globally Lipschitz continuous derivatives. Hence, in the unconstrained case (i.e., $\mathcal{X} = \mathbb{R}^N$), the existing non-asymptotic convergence results for GD, SGD and SVRG cannot be applied. When these assumptions do not hold, even establishing meaningful asymptotic convergence guarantees is a challenging proposition in general. In Appendix A.5, we show that when $M_I = 0$ in (4.22) (i.e., solving a general system of quadratic equations), it is indeed possible to establish such a meaningful, asymptotic convergence result for GD with backtracking line search.

Next, we consider the case $\mathcal{X} \subset \mathbb{R}^N$. If we make an additional assumption that $\mathcal{X}$ is compact, then $F^{(s)}(\mathbf{x})$ and its gradients are (locally) Lipschitz continuous on $\mathcal{X}$. It can be shown that the Lipschitz constant of $\nabla F^{(s)}(\mathbf{x})$ exhibits a $O(\frac{1}{\mu})$ dependence on μ, which is due to the fact that μ is a parameter which controls the level of smoothing applied to the non-differentiable function $F(\mathbf{x})$; i.e., a smaller value of μ allows a tighter degree of approximation, but results in $F^{(s)}(\mathbf{x})$ being less smooth. Typically, we would prefer to choose a small value for μ in order to ensure

tight approximation to $F(\mathbf{x})$. Meanwhile, the constants appearing in the $O(\frac{1}{\mu})$ expression (which depend on the spectral characteristics of the matrices in the constraints) are typically unknown and have to be estimated via some means. In general, such procedures generate overestimates of the constants, which adversely affect convergence speed. Taken together, this ultimately results in the step-size dictated by theory for convergence of GD and SGD being too conservative for the iterates to make any reasonable progress over a prescribed number of iterations.

Finally, we point out that we are ultimately interested in determining a feasible point (i.e., a point $\mathbf{x} \in \mathcal{X}$ for which the globally optimal cost $F^{(s)}(\mathbf{x}) = 0$ is attained) using our FOMs. Hence, it is evident that convergence to a stationary point (which only satisfies the necessary conditions for optimality) is not sufficient to guarantee feasibility. In this case, ensuring recovery of a feasible solution (when one exists) requires establishing convergence to the globally optimal cost 0, which, given the fact that (4.22) is NP–hard in its general form, is considerably more difficult to establish relative to showing convergence to a stationary point. Hence, in several of our experiments, we resorted to empirical step-size selection strategies for our FOMs. Although we cannot make any theoretical convergence claims for such step-sizes, our experiments indicate that these methods can still perform very favorably with these choices.

4.3.2 SCA Approach

The SCA approach is based on approximating (4.5) via a sequence of convex problems. Since the non-convexity in (4.5) is restricted to the cost function, this entails approximating the cost function via a sequence of convex majorization functions. We now describe the procedure for constructing such a majorization function at each iteration.

First, consider the hinge-loss functions $f_m(\mathbf{x}), \forall\, m \in [M_I]$. Again, we utilize eigen-decomposition to decompose each matrix $\mathbf{A}_m$ into its constituent positive and negative semidefinite components and then express the associated quadratic term as a difference of quadratic convex functions. After linearizing the concave term $\mathbf{x}^T\mathbf{A}_m^{(-)}\mathbf{x}$ about the current iterate $\mathbf{x} = \mathbf{x}^{(n)}$, we obtain the following function

$$u_m(\mathbf{x}, \mathbf{x}^{(n)}) := (\mathbf{x}^T\mathbf{A}_m^{(+)}\mathbf{x} + (2\mathbf{A}_m^{(-)}\mathbf{x}^{(n)})^T\mathbf{x} - \mathbf{x}^{(n)T}\mathbf{A}_m^{(-)}\mathbf{x}^{(n)} \quad - b_m)^+, \forall\, m \in [M_I] \tag{4.23}$$

It can be readily verified that $\forall\, m \in [M_I]$, $u_m(\mathbf{x}, \mathbf{x}^{(k)})$ is a convex, non-differentiable majorizer of $f_m(\mathbf{x})$ which is tight at $\mathbf{x} = \mathbf{x}^{(k)}$. Next, we equivalently express each absolute penalty function $g_m(\mathbf{x})$ as

$$g_m(\mathbf{x}) = |\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m| = \max\{\mathbf{x}^T\mathbf{C}_m\mathbf{x} - d_m, -\mathbf{x}^T\mathbf{C}_m\mathbf{x} + d_m\}, \forall\, m \in [M_E] \tag{4.24}$$

In order to majorize each such $g_m(\mathbf{x})$, we resort to the eigen-decomposition technique to express each of the quadratic terms inside the point-wise maximization operator as the difference of

convex quadratics. By linearizing the appropriate non-convex term about $\mathbf{x} = \mathbf{x}^{(n)}$, we obtain the pair of convex functions

$$v_m^{(+)}(\mathbf{x}, \mathbf{x}^{(n)}) := \mathbf{x}^T \mathbf{C}_m^{(+)} \mathbf{x} + (2\mathbf{C}_m^{(-)} \mathbf{x}^{(n)})^T \mathbf{x} - \mathbf{x}^{(n)T} \mathbf{C}_m^{(-)} \mathbf{x}^{(n)} - d_m, \forall\, m \in [M_E] \tag{4.25a}$$

$$v_m^{(-)}(\mathbf{x}, \mathbf{x}^{(n)}) := -\mathbf{x}^T \mathbf{C}_m^{(-)} \mathbf{x} - (2\mathbf{C}_m^{(+)} \mathbf{x}^{(n)})^T \mathbf{x} + \mathbf{x}^{(n)T} \mathbf{C}_m^{(+)} \mathbf{x}^{(n)} + d_m, \forall\, m \in [M_E] \tag{4.25b}$$

On defining

$$\omega_m(\mathbf{x}, \mathbf{x}^{(n)}) := \max\{v_m^{(+)}(\mathbf{x}, \mathbf{x}^{(n)}), v_m^{(-)}(\mathbf{x}, \mathbf{x}^{(n)})\}, \forall\, m \in [M_E] \tag{4.26}$$

we obtain a convex majorization function for each $g_m(\mathbf{x}), \forall\, m \in [M_E]$. Hence, at each SCA iteration, we obtain a non-smooth, convex optimization problem of the following form

$$\mathbf{x}^{(n+1)} \in \arg\min_{\mathbf{x} \in \mathcal{X}} \sum_{m=1}^{M_I} u_m(\mathbf{x}, \mathbf{x}^{(n)}) + \sum_{m=1}^{M_E} \omega_m(\mathbf{x}, \mathbf{x}^{(n)}) \tag{4.27}$$

We now point out that problem (4.27) is actually equivalent to (4.4), since (4.4) can be obtained via the epigraph transformation of (4.27). Hence, the resulting SCA algorithm inherits the same convergence properties as FPP-SCA [5] . From a computational standpoint, formulation (4.27) possesses the advantage of being in a form suitable for the application of low-complexity subgradient methods. While subgradient descent can be applied to solve each SCA subproblem of the form (4.27) at a rate independent of the problem dimension N, there is still an implicit dependence on the total number of constraints M. In order to remove the dependence on M, we can solve (4.27) using the SSGD algorithm, which has the benefit of possessing a convergence rate independent of N and M. However, the drawback of using SSGD is that it only converges in expectation, thus implying that the SCA iterates obtained via this method are not even guaranteed to exhibit monotonic decrease of the cost function in this case. Nevertheless, it offers a substantially low-complexity alternative for decreasing the cost function of (4.27) initially, with possible "last mile" refinement at a later stage via a more sophisticated algorithm.

For a given SCA subproblem (4.27), at each iteration of SSGD, we are only required to sample an index m from the set $[M]$ uniformly at random and then compute a subgradient for the associated function indexed by m in order to compute the update (4.15). If the indexed function is of the form $f(\mathbf{x}) = (\mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c)^+$, (where $\mathbf{A} \succeq \mathbf{0}$) we can compute a subgradient $\mathbf{g} \in \partial f(\mathbf{x})$ at the point $\mathbf{x}$ according to the following equation

$$\mathbf{g} = \begin{cases} 2\mathbf{A}\mathbf{x} + \mathbf{b}, & \text{if } \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c > 0 \\ \mathbf{0}, & \text{otherwise} \end{cases} \tag{4.28}$$

whereas for $f(\mathbf{x}) = \max\{h_1(\mathbf{x}), h_2(\mathbf{x})\}$, where $h_i(\mathbf{x}) = \mathbf{x}^T \mathbf{A}_i \mathbf{x} + \mathbf{b}_i^T \mathbf{x} + c_i, \forall\, i = \{1, 2\}$ are convex quadratics, a subgradient $\mathbf{g} \in \partial f(\mathbf{x})$ at the point $\mathbf{x}$ can be obtained by simply selecting any

[5] i.e., convergence to a KKT point of the *smooth* feasibility problem .

one of the two functions $h_1(\mathbf{x}), h_2(\mathbf{x})$ which attains the maximum and then taking its gradient. Additionally, in each SCA iteration, we warm start the SSGD algorithm from the current iterate $\mathbf{x}^{(k)}$ in order to obtain further savings in computation.

4.4 Experiments on Synthetic Data

In this section, we evaluate and compare the performance of our methods on synthetically generated experiments as well as on real engineering problems. First, we provide a few details regarding the implementation of the methods.

All our methods were implemented in MATLAB on a Linux desktop with 4 Intel i7 cores and 16 GB of RAM. We tested the performance of the direct methods (i.e., GD, SGD, and SVRG) with the following parameter settings.

1. The smoothing parameter for inequality constraints was set to $\mu = 10^{-4}$.

2. For selecting the step-size, the following rules were used

   (a) *Diminishing:* $\alpha_k = \frac{c_1}{k^\gamma}$

   (b) *Polynomial:* $\alpha_k = \frac{c_2}{(1+c_3 k/M)^\gamma}$

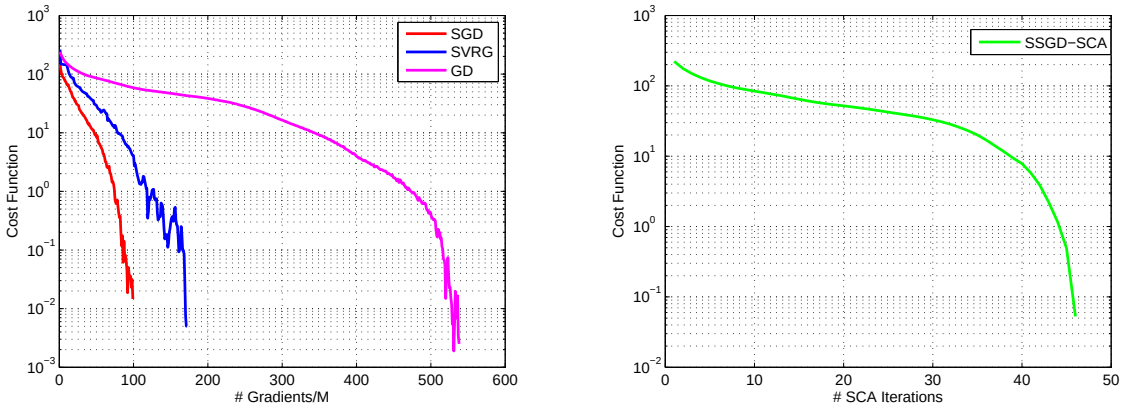
   (c) *Norm regularized:* $\alpha_k = c_4/\|\mathbf{x}^{(k)}\|_2^2$

   where $k \in \mathbb{N}$ denotes the iteration index, $c_1, c_2, c_3, c_4 \in \mathbb{R}_{++}$ and $\gamma \in (0, 1]$. The polynomial step-size rule can be viewed as a generalization of the popular inverse-t schedule (corresponding to $\gamma = 1$) while the norm regularized step-size rule can be motivated via arguments made in [44, Proposition 1] regarding worst-case convergence results of FOMs applied to minimize quartic functions. The parameters were empirically tuned to yield the best performance.

3. For SVRG, the length of each stage was set to $S = 4M$.

4. Since each method requires a different number of gradient evaluations per iteration, for fair comparison, we allocated a fixed number of total gradient evaluations to each method and evaluated the cost function after every M gradient evaluations. Of course, this implies that the maximum number of iterations for each method is different, depending on the number of gradients evaluated per iteration.

Regarding SSGD-SCA, we used a maximum of 50 SCA iterations while each inner convex subproblem was solved using 50×10^3 SSGD iterations with a step-size of $O(\frac{1}{\sqrt{k}})$ and iterate averaging.



(A) (B)

Figure 4.1: Single instance of feasibility problem with $N = 200$ variables and $M = 1000$ non-convex quadratic inequalities: (A) Evolution of penalty function for direct FOMs (B) Evolution of penalty function for SSGD-SCA.

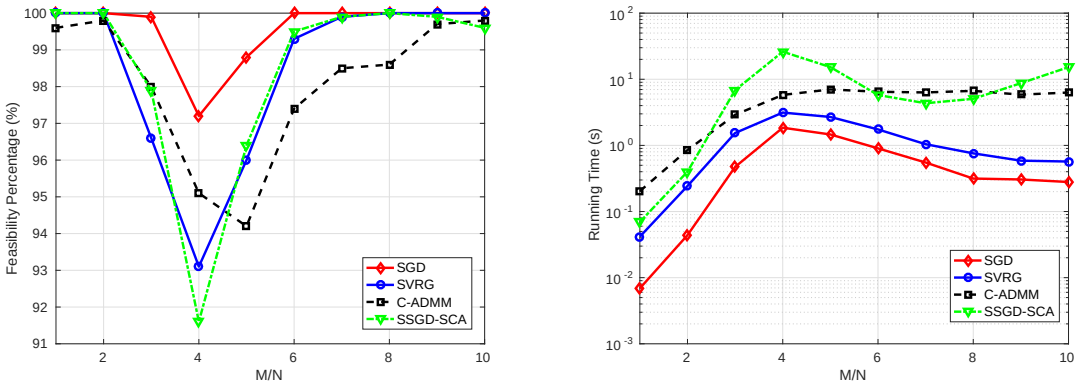
First, we present an illustrative experiment on a synthetically generated instance of a non-convex QCQP problem with $N = 200$ variables and $M = 1000$ constraints. Here, we set $M_E = 0$ (i.e., no equalities) and randomly generated the inequality constraint matrices $\{\mathbf{A}_m\}_{m=1}^M$ from a zero mean, i.i.d. Gaussian distribution with unit variance (followed by symmetrization). In order to ensure that the problem is feasible, we randomly generated a unit norm vector $\mathbf{p}$ and drew each of the right-hand sides $\{b_m\}_{m=1}^M$ from a Gaussian distribution $b_m \sim \mathcal{N}(\mathbf{p}^T\mathbf{A}_m\mathbf{p}, 1)$. In the event $\mathbf{p}^T\mathbf{A}_m\mathbf{p} > b_m$, we multiplied both sides of the inequality by -1 to get $\leq$ inequalities. In short, we randomly generated a quadratic feasibility problem with indefinite matrices which possesses a unit-norm feasible solution. We exploit this prior knowledge in our setup by setting $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^N | \|\mathbf{x}\|_2 \leq 1\}$. A randomly generated unit-norm vector was used to initialize GD, SGD, SVRG and SSGD-SCA. We used a maximum budget of $1000M$ gradient evaluations for each of the direct methods. For SGD, we use the diminishing step-size rule with $c_1 = 0.1$ and $\gamma = 0.5$, while for SVRG and GD, we used the polynomial averaging step-size rule with $c_3 = 1$ and $c_2 = 0.1, \gamma = 1$ for GD and $c_2 = 0.01, \gamma = 0.5$ for SVRG. We declare success in finding a feasible point if the value of the cost function in the exact penalty formulation (4.5) is smaller than a tolerance value of 10^{-6}. The results are depicted in Figure 4.1, where we plot the evolution of the constraint violation (as measured by the quadratic hinge-loss function in (4.5)) for the various methods.

In this case, all methods were successful in achieving feasibility; i.e., attaining the globally

optimal cost 0. Since the evolution of the penalty function in Figure 4.1 is represented on a logarithmic scale, it depicts the cost of the iterates before the value of 0 was attained. For the direct FOMs, the potential benefits accrued in opting for an aggressive, empirically chosen step-size strategy is clear. Meanwhile, it is obvious that, given the scale of the problem, we should refrain from using IPMs to solve each SCA sub-problem. As an alternative, SSGD-SCA performs admirably, and even exhibits monotonic decrease of the cost function in this case. Regarding timing, SGD performed the best by attaining feasibility in 17 secs, while SVRG, GD and SSGD-SCA required $27, 85$ and 233 secs respectively. Note that although we adopt a FOM based approach to SCA here, it still incurs substantially more complexity compared to the direct FOM approach. Overall, these preliminary results indicate that the direct FOMs have a distinct advantage over SSGD-SCA. We now seek to corroborate these findings via the following set of exhaustive simulations.

In our setup, we fixed the number of variables N and randomly generated instances of quadratic feasibility problems with varying number of inequality constraints M via the procedure describe in the previously. For each value of M, we generated 1000 such instances. We also added the C-ADMM algorithm proposed in [42] for comparison against our FOMs. We set the maximum iteration counter of C-ADMM to 5000 iterations. Meanwhile, for the direct FOMs, we again use a budget of $1000M$ gradient evaluations and a maximum of 50 SCA iterations for SSGD-SSCA. We also remove GD from contention here since our experiments indicate that it is always outperformed by SGD and SVRG at lower complexity. Furthermore, we allow a maximum of 2 restarts for SGD and SVRG in the event that feasibility is not attained within the prescribed number of iterations. In each instance, we initialize all the methods from a randomly generated unit-norm vector. The step-size rules for the direct FOMs and SSGD-SCA are also unchanged from the previous experiment. An alternative approach could be to tune the step-size parameters to achieve the best performance for each M, at the cost of more effort. As we demonstrate, our chosen parameters work well across a wide range of N and M, thereby considerably alleviating the burden of tuning parameters while simultaneously providing further empirical validation of our heuristic step-size sequences. We also used the same termination criterion used in the previous experiment for declaring convergence to a feasible point for all methods.

We plot the feasibility percentages averaged out over 1000 instances as a function of M/N along with the timing results, averaged out over the subset of instances where feasibility was successfully attained by the respective algorithms, in Figures 4.2,4.3 and 4.4. From these figures, it can be observed that for all N considered, SGD and SVRG demonstrate the best performance in terms of feasibility and timing (including restarts). For variable dimensions $N = 100$ and larger, the running time of SSGD-SCA becomes too expensive to merit comparison. Similarly,



(A) (B)

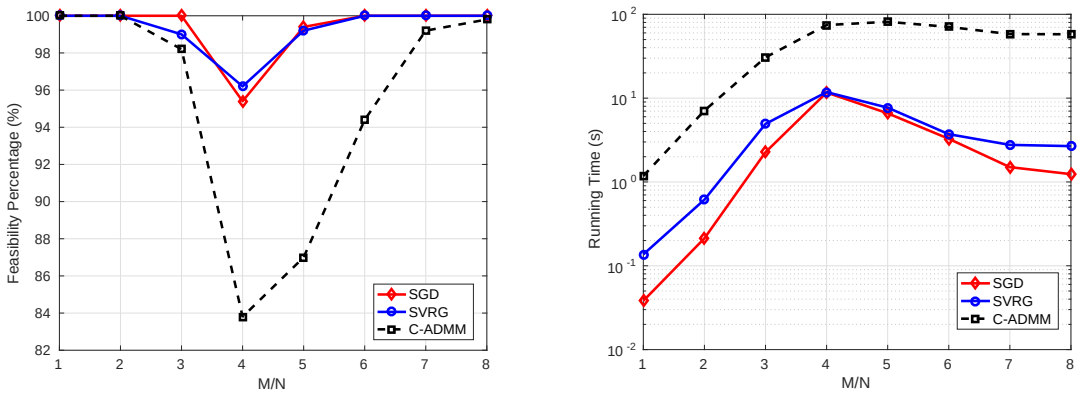
Figure 4.2: 1000 instances with $N = 50$: (A) Average feasibility percentage vs M/N (B) Average timing (secs) vs M/N.

for $N = 200$, we remove C-ADMM as well, since the average running time of C-ADMM is approximately 20 minutes in this case. In contrast, even for $N = 200$, the worst average running-time of SGD and SVRG, with restarts taken into account, is slightly in excess of one minute. It is evident that SVRG and SGD are significantly more scalable compared to the existing state-of-art, while, remarkably, exhibiting near-optimal performance with regard to recovering feasible solutions in all cases. Hence, SGD and SVRG emerge as the algorithms of choice in this case.

Finally, although the algorithms we have applied on the feasibility problem are quite different from each other, we point out that they exhibit a slight phase transition in terms of feasibility percentages as M/N varies. Note that this effect is least pronounced overall in the case of SGD and SVRG. The presented results (showing enhanced feasibility with more constraints) hinge upon the method adopted for generating instances of (4.1), in particular that a unit-norm solution exists. While we do not have a complete explanation of this phenomenon at present, a similar observation has been made and theoretically explained in the special case of solving random systems of quadratic equations (see [48, 117] for details). We conjecture that a similar line of reasoning can also be applied here; however, formally establishing this argument is beyond the scope of this thesis.

4.5 Application: Power System State Estimation

Power system state estimation (PSSE) is a fundamental problem in power systems engineering where the objective is to estimate the complex voltages across the constituent buses of an



(A) (B)

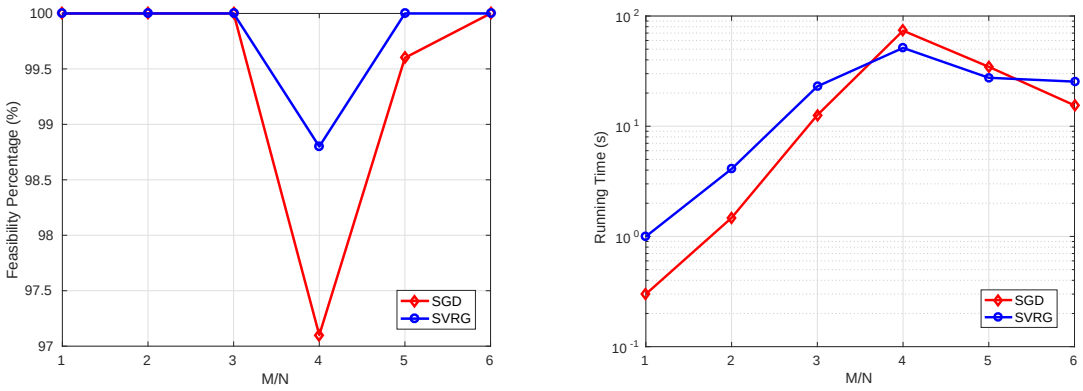
Figure 4.3: 1000 instances with $N = 100$: (A) Average feasibility percentage vs M/N (B) Average timing (secs) vs M/N.

electrical power grid from a set of measurements pertaining to the observable quantities [12]. In this section, we investigate the efficacy of using FOMs on the PSSE problem, which entails solving a system of indefinite quadratic equations, thus rendering it NP–hard in its general form [104].

4.5.1 Problem Formulation

Given an electrical power transmission network, we model it as an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} := \{1, \cdots, N\}$ denotes the set of buses (nodes) and $\mathcal{E}$ represents the set of transmission lines (edges), with each transmission line $(m, n) \in \mathcal{E}$ corresponding to an unordered pair of distinct buses. At each bus $n \in \mathcal{N}$, we define the following complex nodal quantities: voltage $V_n := |V_n|e^{j\theta_n}$, current injection $I_n := |I_n|e^{j\phi_n}$, and power injection $S_n := P_n + jQ_n$ (here P_n and Q_n denote the active and reactive power injections respectively). Associated with each transmission line $(m, n) \in \mathcal{E}$ are the following line quantities: I_{mn} is the complex current flowing from bus m to n, while $S_{mn} := P_{mn} + jQ_{mn}$ is the apparent power flow from bus m to n (here P_{mn} and Q_{mn} denote the active and reactive power flow respectively), as seen at the sending end. For notational simplicity, we represent the nodal quantities $\{V_n\}_{n\in\mathcal{N}}, \{I_n\}_{n\in\mathcal{N}}, \{P_n\}_{n\in\mathcal{N}}$ and $\{Q_n\}_{n\in\mathcal{N}}$ in the form of vectors $\mathbf{v} := [V_1^*, \cdots, V_N^*]^H \in \mathbb{C}^N$, $\mathbf{i} := [I_1^*, \cdots, I_N^*]^H \in \mathbb{C}^N, \mathbf{p} := [p_1, \cdots, p_N]^T \in \mathbb{R}^N$, and $\mathbf{q} := [q_1, \cdots, q_N]^T \in \mathbb{R}^N$ respectively.

Bus $n \in \mathcal{N}$ has access to a set $\mathcal{L}_n$ of (possibly noisy) SCADA measurements $\{z_l\}_{l\in\mathcal{L}_n}$ corresponding to the voltage magnitude $|V_n|$, active and reactive power injections P_n and Q_n,



(A) (B)

Figure 4.4: 1000 instances with $N = 200$: (A) Average feasibility percentage vs M/N (B) Average timing (secs) vs M/N.

and possibly the active and reactive power flows P_{nm} and Q_{nm}, $\forall\, m \in \{m|(n,m) \in \mathcal{E}\}$ (i.e., if bus n corresponds to the sending end).

In an AC power flow model, current and voltage laws mandate that the state variables $\mathbf{v}$ are quadratically related to the SCADA measurements $\{z_l\}_{l\in\mathcal{L}_n}, \forall\, n \in \mathcal{N}$. This holds trivially for the square of the voltage magnitude measurements, since $|V_n|^2 = V_n V_n^*, \forall\, n \in \mathcal{N}$. In order to see that such a relationship exists between $\mathbf{v}$ and the other power measurements, let $\mathbf{Y} \in \mathbb{C}^{N\times N}$ denote the bus-admittance matrix whose entries are given by

$$Y_{mn} := \begin{cases} -y_{mn}, & (m,n) \in \mathcal{E} \\ \bar{y}_{nn} + \sum_{k\in\mathcal{N}_n} y_{nk}, & m = n \\ 0, & \text{otherwise} \end{cases} \tag{4.29}$$

where y_{mn} is the admittance of line $(m,n) \in \mathcal{E}$, $\bar{y}_{nn}$ is the admittance to ground at bus $n \in \mathcal{N}$, and $\mathcal{N}_n := \{k|(n,k) \in \mathcal{E}\}$ denotes the immediate neighborhood of bus n. We point out that $\mathbf{Y}$ is symmetric but non-Hermitian, and is also sparse. Combining Kirchoff's current law and the multivariate form of Ohm's law, the relationship between the nodal voltages and currents can be expressed as

$$\mathbf{i} = \mathbf{Y}\mathbf{v} \tag{4.30}$$

For the power injections, under the AC power flow model, it holds that $P_n + jQ_n = V_n I_n^*, \forall\, n \in \mathcal{N}$. Utilizing (4.30), we obtain the following matrix-vector relationship

$$\mathbf{p} + j\mathbf{q} = \text{diag}(\mathbf{v})\mathbf{i}^* = \text{diag}(\mathbf{v})\mathbf{Y}^*\mathbf{v}^* \tag{4.31}$$

Meanwhile, by appealing to Ohm's and Kirchoff's laws, the line current I_{mn} can be expressed as

$$I_{mn} = \bar{y}_{mn} V_m + y_{mn}(V_m - V_n), \forall\, (m,n) \in \mathcal{E} \tag{4.32}$$

where $\bar{y}_{mn}$ denotes the shunt admittance at bus m corresponding to line (m,n). The reverse direction current I_{nm} can be obtained similarly by symmetry. Note that $I_{mn} \neq -I_{nm}$ as $\bar{y}_{mn} \neq 0$. The sending-end active and reactive power flow from bus m to n can now be expressed as

$$\begin{aligned} P_{mn} + jQ_{mn} &= V_m I_{mn}^* \\ &= (\bar{y}_{mn}^* + y_{mn}^*)|V_m|^2 - y_{mn}^* V_m V_n^*, \forall\, (m,n) \in \mathcal{E} \end{aligned} \tag{4.33}$$

where in the second step we have made use of (4.32). From (4.31) and (4.33), it can be observed that the power measurements are quadratically related to $\mathbf{v}$.

We now make explicit the relationship between the measurements $\{z_l\}_{l \in \mathcal{L}_n}$ and $\mathbf{v}$. At each bus $n \in \mathcal{N}$, the available measurements can be expressed in the quadratic form

$$z_l = \mathbf{v}^H \mathbf{H}_l \mathbf{v} + n_l, \forall\, l \in \mathcal{L}_n \tag{4.34}$$

where $\mathbf{H}_l \in \mathbb{C}^{N \times N}$ (to be specified shortly) and $n_l \sim \mathcal{N}(0, \sigma_l^2)$ represents additive Gaussian noise (assumed independent across all meters). Considering the measurement of the voltage magnitude squared at bus n, we have $|V_n|^2 = V_n V_n^* = \mathbf{v}^H \mathbf{e}_n \mathbf{e}_n^T \mathbf{v}$ (where $\mathbf{e}_n \in \mathbb{R}^N$ represents the n^{th} canonical basis vector of $\mathbb{R}^N$). Hence, in this case, we have $\mathbf{H}_{V_n} = \mathbf{e}_n \mathbf{e}_n^T$. In order to define $\{\mathbf{H}_l\}_{l \in \mathcal{L}_n \setminus V_n}$ for the active and reactive power injection and flow measurements, we first define the admittance-related matrices

$$\mathbf{Y}_n := \mathbf{e}_n \mathbf{e}_n^T \mathbf{Y}, \forall\, n \in \mathcal{N} \tag{4.35a}$$

$$\mathbf{Y}_{mn} := (\bar{y}_{mn} + y_{mn}) \mathbf{e}_m \mathbf{e}_m^T - y_{mn} \mathbf{e}_m \mathbf{e}_n^T, \forall\, (m,n) \in \mathcal{E} \tag{4.35b}$$

By equating the real and imaginary parts of (4.31) and (4.33), we obtain the matrices

$$\begin{aligned} \mathbf{H}_{P_n} &:= \frac{1}{2}(\mathbf{Y}_n + \mathbf{Y}_n^H), \qquad \mathbf{H}_{Q_n} := \frac{j}{2}(\mathbf{Y}_n - \mathbf{Y}_n^H) \\ \mathbf{H}_{P_{mn}} &:= \frac{1}{2}(\mathbf{Y}_{mn} + \mathbf{Y}_{mn}^H), \; \mathbf{H}_{Q_{mn}} := \frac{j}{2}(\mathbf{Y}_{mn} - \mathbf{Y}_{mn}^H) \end{aligned} \tag{4.36}$$

We point out that for each bus $n \in \mathcal{N}$, the measurement matrices $\{\mathbf{H}_l\}_{l \in \mathcal{L}_n \setminus V_n}$ are rank-2, indefinite Hermitian matrices, while $\mathbf{H}_{V_n}$ is rank-1, positive semidefinite.

Assuming the availability of the measurements $\{z_l\}_{l \in \mathcal{L}_n}$ corrupted by Gaussian noise across all buses $n \in \mathcal{N}$, adopting a maximum-likelihood estimation approach results in the following weighted least squares (WLS) optimization problem

$$\min_{\mathbf{v} \in \mathbb{C}^N} \left\{ F(\mathbf{v}) := \sum_{n=1}^{N} f_n(\mathbf{v}) := \sum_{n=1}^{N} \sum_{l \in \mathcal{L}_n} \frac{(\mathbf{v}^H \mathbf{H}_l \mathbf{v} - z_l)^2}{\sigma_l^2} \right\} \tag{4.37}$$

We will work with the rectangular coordinate representation of the state variables $\mathbf{v}$. For this purpose, we define $\bar{\mathbf{v}} := [\mathbf{v}_r^T, \mathbf{v}_i^T]^T \in \mathbb{R}^{2N}$, where $\mathbf{v}_r = \mathrm{Re}(\mathbf{v})$ and $\mathbf{v}_i = \mathrm{Im}(\mathbf{v})$. We also define a set of symmetric measurement matrices $\bar{\mathbf{H}}_l \in \mathbb{R}^{2N \times 2N}$ as

$$\bar{\mathbf{H}}_l := \begin{bmatrix} \mathrm{Re}\{\mathbf{H}_l\} & -\mathrm{Im}\{\mathbf{H}_l\} \\ \mathrm{Im}\{\mathbf{H}_l\} & \mathrm{Re}\{\mathbf{H}_l\} \end{bmatrix}, \forall\, l \in \mathcal{L}_n, \forall\, n \in \mathcal{N} \tag{4.38}$$

Note that we have

$$\mathbf{v}^H \mathbf{H}_l \mathbf{v} = \bar{\mathbf{v}}^T \bar{\mathbf{H}}_l \bar{\mathbf{v}},\ \forall\, l \in \mathcal{L}_n, \forall\, n \in \mathcal{N} \tag{4.39}$$

Hence, in terms of rectangular coordinates, (4.37) can be equivalently represented as

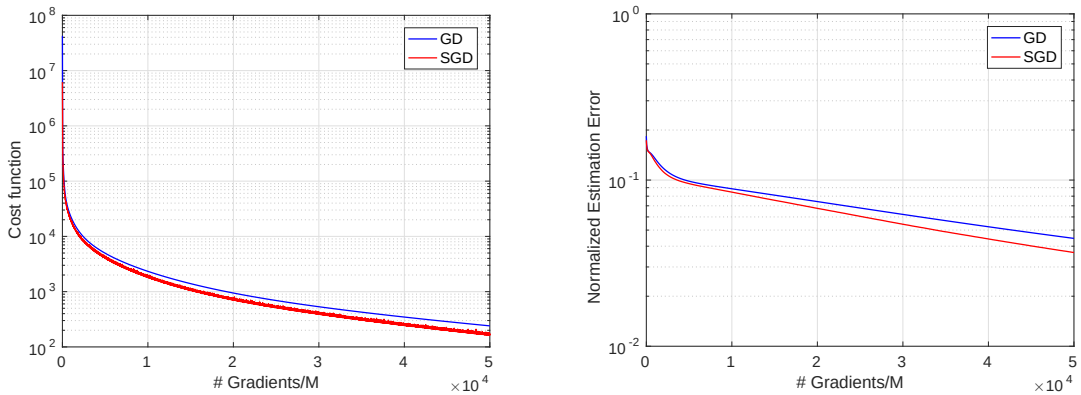
$$\min_{\bar{\mathbf{v}} \in \mathbb{R}^{2N}} \left\{ F(\bar{\mathbf{v}}) := \sum_{n=1}^{N} f_n(\bar{\mathbf{v}}) := \sum_{n=1}^{N} \sum_{l \in \mathcal{L}_n} \frac{(\bar{\mathbf{v}}^T \bar{\mathbf{H}}_l \bar{\mathbf{v}} - z_l)^2}{\sigma_l^2} \right\} \tag{4.40}$$

4.5.2 Numerical Results

The standard workhorse algorithm for this problem is the Gauss-Newton (GN) method, which is well suited for application on non-linear least squares problems. When initialized close to a local minimum, convergence of GN can be established. However, determining such an initialization is non-trivial in general, and the performance of GN is known to be sensitive to the choice of initialization. Here, we implemented a modified version of the GN algorithm described in [118, p. 61], which uses a backtracking line-search procedure for improved performance in practice. Our experiments indicate that this modified GN algorithm exhibits superior performance over standard GN, and should be used instead as the de-facto performance benchmark for this problem. Regarding our FOMs, theoretical convergence of GD with backtracking line-search can be established for this problem (see Appendix D), while as pointed out earlier, establishing convergence for the stochastic gradient methods is still an open problem. We also point out that a modified version of FPP-SCA for the WLS formulation has been recently proposed in [119], which essentially uses the weighted ℓ_2-norm square of the slack variables corresponding to the equalities in each SCA subproblem of the form . Thus, GD and FPP-SCA are the only algorithms under consideration here with convergence guarantees for PSSE. We also point out that due to data sparsity, computing gradients for the direct FOMs in this case requires sparse matrix-vector multiplications, which can be accomplished very efficiently. In contrast, exploiting data sparsity in general purpose conic programming solvers (utilized by FPP-SCA) is a more challenging proposition.

In our experiments, we evaluate estimation performance according to the normalized mean square error criterion, defined as NMSE $:= \frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_2}{\|\mathbf{x}\|_2}$, where $\hat{\mathbf{x}}$ is the estimated voltage profile and $\mathbf{x}$ is the true voltage profile. We used MATPOWER [120] for generating the voltage profiles and SCADA measurements. The voltage magnitude at each bus was generated from a uniform

distribution over the interval $[0.9, 1.1]$ while the voltage angle was uniformly distributed over $[-0.1\pi, 0.1\pi]$. The phase of the reference bus was set to 0 in order to resolve the phase ambiguity in all experiments. We also added independent, zero-mean Gaussian noise with variances of 10 and 13 dBm to the measurements corresponding to the voltage magnitudes and power flow/injections respectively. Regarding the algorithms, we run (modified) GN for a maximum of 100 iterations, since we observed that GN either always converges or ceases to improve the cost function within this iteration limit. We added a small regularization term to the inexact Hessian at each step in order to ensure that it is well-conditioned and set the line-search parameter $\alpha = 0.1$. For implementing FPP-SCA, we used the modeling language YALMIP [97] along with the general convex programming solver MOSEK [96]. As for the direct FOMs, we fix a maximum gradient budget to ensure a fair comparison between GD and SGD. For GD, we use a crude choice of line-search parameters (see [14, p. 466]) while we implemented SGD with minibatch stochastic gradients of size $\lfloor \frac{M}{10} \rfloor$ and the norm-regularized step-size rule. GN, GD and SGD were always initialized from the flat-voltage profile (i.e., the all-ones vector). The test buses used in our experiments were obtained from the NESTA archive [121].



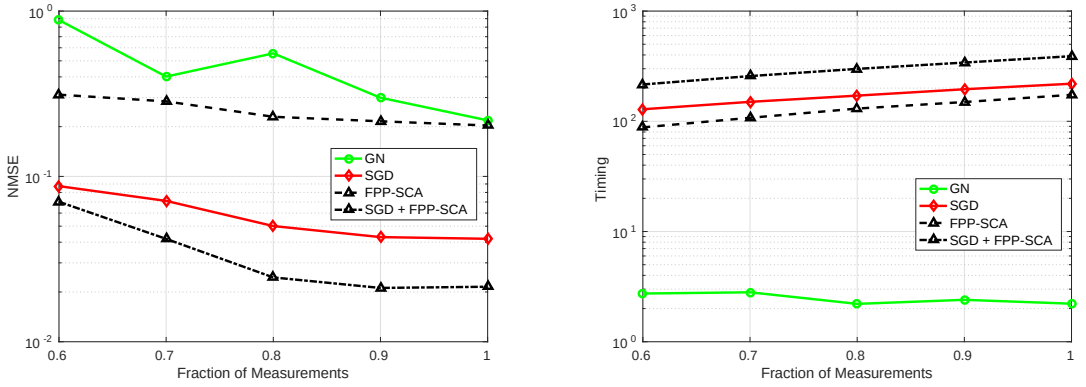
(A) (B)

Figure 4.5: Performance comparison on PEGASE 89 bus system with full set of SCADA measurements: (A) Evolution of cost function (B) Evolution of NMSE.

In Figure 4.5, we present a preliminary simulation result demonstrating the performance of SGD and GD (with backtracking line-search) for a single realization of the voltage profile on the PEGASE-89 bus system with the full set of noisy SCADA measurements (corresponding to $N = 178, M = |\mathcal{E}| = 687$). Here, we set $c_4 = 4 \times 10^{-5}$ for the norm-regularized step size rule of SGD. The evolution of the WLS cost function shows that SGD attains a solution with lower cost compared to GD within the prescribed gradient budget, which was set to be $5 \times 10^4 M$.

Furthermore, this also translates into better estimation performance for SGD compared to GD. In terms of timing, SGD was roughly 6 times faster compared to GD on our machine. Hence, on this real world problem, SGD is also capable of exhibiting very favorable performance, even when pitted against a provably convergent FOM.

Next, we devised an experiment where we evaluated the estimation performance of SGD, GN, and FPP-SCA (all initialized from the flat-voltage profile) with varying number of noisy measurements. GD is omitted here since it exhibits worse estimation performance compared to SGD at higher complexity on this network. We only run 2 iterations of FPP-SCA due to its high complexity. Additionally, we also refine the final solution returned by SGD using 2 iterations of FPP-SCA. Given a sampling fraction $\gamma \in (0, 1]$, we sample a fraction γ of the total measurements uniformly at random from each measurement type. This implies that for the active power injections, for example, out of a total of $N = 89$ available measurements, we subsampled $\lfloor \gamma N \rfloor$ measurements uniformly at random, and likewise for the other measurement types. All our results were averaged over 200 Monte-Carlo trials, and are depicted in Figure 4.6.



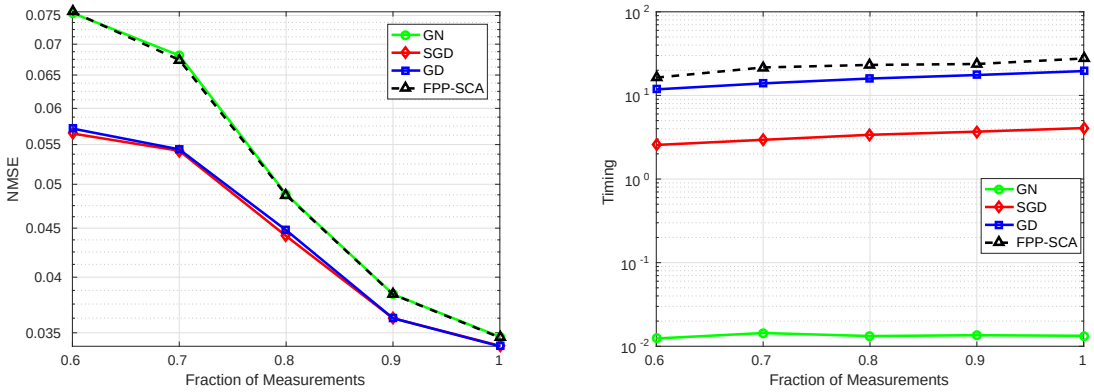
(A) (B)

Figure 4.6: Performance Results for PEGASE 89 bus system: (A) Avg. NMSE vs γ (B) Avg. Wall Time vs γ.

It is evident that on this network, GN, while being the fastest method, is also the one which exhibits the worst estimation performance. In contrast, SGD performs significantly better, albeit at (moderately) higher complexity. Owing to its high complexity, running FPP-SCA from the flat-start for 2 iterations incurs approximately the same running-time as SGD while being significantly worse-off in terms of NMSE. This highlights the ability of SGD to attain a very favorable performance-complexity trade-off compared to other alternatives. We additionally note that the solution determined by SGD can serve as a good initialization for FPP-SCA,

which obviates the need to initialize FPP-SCA from the flat voltage profile and thereby incur significantly higher complexity. Clearly, combining SGD with 2 iterations of FPP achieves the lowest NMSE. However, this combined heuristic still exhibits the highest complexity (even with 2 SCA iterations), which further underscores the benefit of SGD initialization.

We also carried out similar experiments on a series of test bus systems. Figure 4.7 depicts the results for the IEEE 30 bus network. In this case, we used a gradient budget of $5000M$ for the FOMs and set $c_4 = 0.02$ for SGD. As this is a fairly small network, we initialized FPP-SCA from the flat start and ran it until it attains convergence in the cost function, or a maximum of 20 iterations are executed. It is evident that both FOMs outperform GN and FPP-SCA in terms of estimation error this case, which is remarkable given the fact that they are considerably less sophisticated compared to the aforementioned methods. Furthermore, while it may be tempting to conjecture from the figures that GN attains the best performance-complexity trade-off in this case, this is not so: in our experiments, GN never improves upon its estimates beyond 100 iterations, and thus one cannot obtain better estimation performance for GN by using more iterations. Hence, it is SGD which is overall the best in this case from the perspective of performance-complexity trade-off, which makes it all the more remarkable given that it is the *least* sophisticated technique amongst all the methods under consideration.

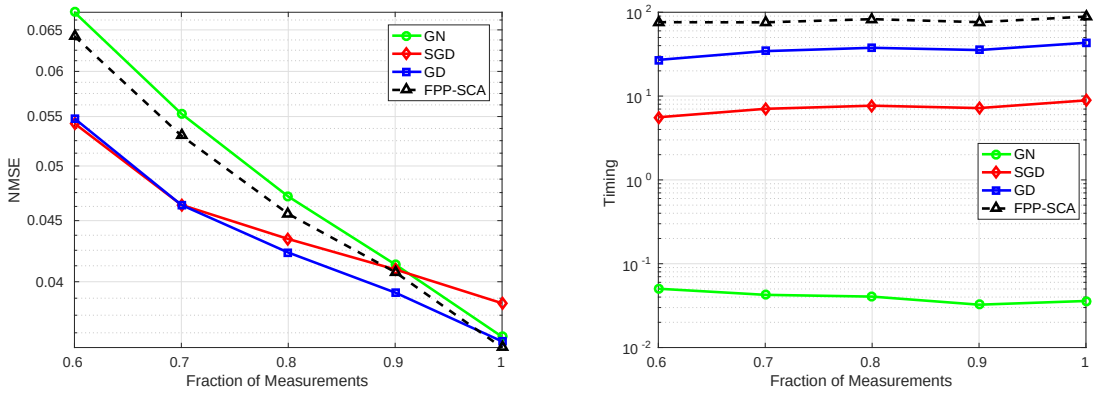


(A) (B)

Figure 4.7: Performance Results for IEEE 30 bus system: (A) Avg. NMSE vs γ (B) Avg. Wall Time vs γ.

In Figure 4.8, we show the results obtained on the IEEE 57 bus network, for which we used a maximum gradient budget of $5000M$ and set $c_4 = 0.05$. In this case as well, we initialized FPP-SCA from the flat start until convergence or a maximum of 20 iterations are reached. From the figures, it can be seen that both FOMs perform very admirably in terms of estimation

error compared to both GN and FPP-SCA: GN demonstrates class-leading performance only for $\gamma \geq 0.9$ while running FPP-SCA always results in higher complexity relative to the other methods. It is clear that when one has access to a partial set of measurements (i.e., $\gamma \leq 0.8$), the FOMs possess the upper hand in terms of performance.



(A) (B)

Figure 4.8: Performance Results for IEEE 57 bus system: (A) Avg. NMSE vs γ (B) Avg. Wall Time vs γ.

4.6 Conclusions

We presented a simple heuristic framework for determining feasible points of general non-convex QCQPs using memory efficient and computationally lightweight FOMs, which makes our approach very scalable. While a general theory of provable performance guarantees is elusive at present, we provided a selection of empirical step-sizes for which the FOMs surprisingly exhibit very favorable performance in terms of feasibility attained on synthetic experiments and estimation error for PSSE using real power network data compared to more established and sophisticated alternatives. Given the startling empirical performance of FOMs for this class of hard optimization problems, we can reasonably claim that our direct FOM based approach constitutes a significant advancement in the state-of-art for computing feasible solutions of large-scale non-convex QCQP. In particular, the stochastic gradient methods emerge as the algorithms of choice in our experiments.

Chapter 5

Summary and Future Directions

This dissertation introduced a framework for obtaining approximate solutions to non-convex QCQPs in an efficient manner. Chapter 2 considered the special case of homogeneous non-convex QCQP with Toeplitz-Hermitian quadratic forms and established that SDR followed by spectral factorization can used to optimally solve the problem in polynomial-time. Since the problem possesses hidden convexity, no approximation is required in this case. Furthermore, if the matrices additionally possess circulant structure, then the QCQP problem can be equivalently reformulated as a LP problem, and again optimally solved in polynomial-time at significantly lower complexity compared to SDP.

The subsequent chapters considered the general case of the non-convex QCQP problem, and proposed FOM based optimization approaches for efficiently computing feasible points. Chapter 3 considered the special case of non-convex quadratic feasibility problems where the feasible set is defined by the intersection of a convex, compact set and a system fo quadratic inequalities with negative semidefinite matrices. A feasibility criterion which minimizes the maximum violation of the inequality constraints was adopted and an SCA algorithm was developed for the purpose of achieving feasibility. The global convergence of the iterates generated by the SCA algorithm to the set of d-stationary solutions of the feasibility problem was established. In order to reduce the computational complexity associated with solving each SCA subproblem, the piece-wise linear cost function of each subproblem was equivalently reformulated as maximizing a bilinear function over a convex set. This special structure was exploited using specialized FOMs which are capable of efficiently computing solutions for each subproblem. The problem of single-group multicast beamforming was considered as an example, where the objective was to design a max-min fair beamformer subject to transmit power constraints. Simulations in various multicasting scenarios demonstrated that the proposed FOM based SCA algorithms attain a very favorable performance-complexity trade-off relative to the existing state-of-art.

Chapter 4 considered the most general case of the quadratic feasibility problem defined by an arbitrary system of inequalities and equalities. A non-convex, non-smooth exact penalty formulation was proposed for the purpose of computing a feasible solution. Appealing to Nesterov's smoothing technique, a smooth formulation of the exact penalty formulation was derived, which is well suited for the direct application of FOMs. The difficulty in adapting the existing convergence results of FOMs for non-convex problems was outlined, and the use of empirically chosen step-sizes was advocated. While this approach is extremely scalable owing to its simplicity, *apriori* it would appear that this comes at the considerable expense of lacking technical sophistication compared to prevailing approaches (namely, SDR, FPP-SCA and C-ADMM), and hence, perhaps does not even merit consideration for a problem which is non-convex and NP–hard in general. However, using judiciously designed experiments, compelling empirical evidence was provided to the contrary. On synthetic experiments, the stochastic gradient based algorithms demonstrated startling success in attaining feasible solutions at significantly lower complexity compared to the pre-existing alternatives. Additionally, when applied on the problem of power system state estimation, we demonstrated that SGD can attain very favorable estimation performance using a smaller subset of available measurements compared to FPP-SCA and GN. While the author himself was initially skeptical about the prowess of FOMs for feasibility pursuit, given their remarkable empirical success, he believes that the work reported herein is a significant advancement in the state-of-art for obtaining feasible solutions of general non-convex QCQP. For the better part of almost two decades, one has principally been restricted to using SDR for doing so. It is therefore our hope that this work will enable many more large-scale applications as well as research in non-convex QCQP.

Based on this dissertation, our ongoing research is concerned with the following extensions and applications:

- **Boolean Quadratic Optimization Problems:** While we considered QCQP subject to continuous constraints in this dissertation, several problems arising in network science and data analytics can be formulated as maximizing/minimizing a quadratic function subject to boolean constraints. In such cases, computing a feasible solution is trivial; instead, the challenge is in computing high-quality approximate solutions. However, such problems are not only NP-hard in their general form, but also inapproximable via polynomial-time algorithms. Nevertheless, motivated by the success of SCA for continuous problems, we seek to develop a discrete version of SCA and test its effectiveness for boolean QCQP.

- **Power Systems Grid Optimization:** QCQPs find widespread application in power systems engineering as the constraints imposed by the physical laws of the electrical network and the limits of power generation are defined by a system of quadratic equalities

and inequalities. The task of modernizing the electrical grid infrastructure to meet the energy demands of the 21st century has recently received significant research attention, which necessitates solving the basic non-convex QCQP problem (1.1) in various settings with appropriate modifications. Specific examples include, but are not limited to: i) performing PSSE in a real-time, decentralized fashion, due to the sheer scale of the grid interconnections. For this purpose, judicious online/distributed algorithms are required for computing effective solutions; ii) the ever increasing integration of renewable energy resources (RESs) demands that system operators take into account the uncertainty introduced in system operation owing to their intermittent nature. A reasonable approach to such problems would be to consider stochastic programming based or worst-case minimax formulations of (1.1); and finally, iii) the unit-commitment problem in economic dispatch (i.e., the task of satisfying power demand while respecting network constraints and minimizing the cost of generation) introduces boolean variables in (1.1) for solving a generator scheduling problem over a finite time horizon (on the order of hours or days). This requires developing approximation algorithms for tackling the mixed-integer constraints. We point out that all these problems are atleast as hard as the basic formulation (1.1), which makes computing effective solutions in reasonable time a daunting task. Prior approaches have relied upon using a simple linear system model for what is a nonlinear programming problem, thereby resulting in significant performance degradation, or employing computationally expensive SDR based approaches. Motivated by our successes in directly tackling the general case of non-convex QCQP in this dissertation, we will apply suitable modifications to the toolbox of algorithms proposed herein for the purpose of obtaining high-quality approximate solutions for these very challenging, yet timely, engineering problems.

References

[1] W.-K. Ma, T. N. Davidson, K. M. Wong, Z.-Q. Luo, and P.-C. Ching, "Quasi-maximum-likelihood multiuser detection using semidefinite relaxation with applications to synchronous CDMA", *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 912–922, Aug. 2002.

[2] N. Sidiropoulos, T. Davidson, and Z.-Q. Luo, "Transmit beamforming for physical-layer multicasting," *IEEE Trans. Signal Process.*, vol. 54, no. 6, pp. 2239–2251, June 2006.

[3] E. Karipidis, N. D. Sidiropoulos, and Z.-Q. Luo, "Quality of Service and Max-min-fair Transmit Beamforming to Multiple Co-channel Multicast Group," *IEEE Trans. Signal Process.*, vol. 56, no. 3, pp. 1268–1279, Mar. 2008.

[4] A. B. Gershman, N. D. Sidiropoulos, S. Shahbazpanahi, M. Bengtsson, and B. Ottersten, "Convex optimization based beamforming," *IEEE Signal Process. Mag.,* vol. 27, no. 3, pp. 62–75, May 2010.

[5] M. X. Goemans, and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problem using semi-definite programming," *Journal of ACM*, vol. 42, no. 6, pp. 1115–1145 , Nov. 1995.

[6] O. Besson, "Adaptive detection with bounded steering vectors mismatch angle," *IEEE Trans. Signal Process.*, vol. 55, no. 4, pp. 1560–1564, Apr. 2007.

[7] A. De Maio, S. De Nicola, Y. Huang, S. Zhang, and A. Farina, "Code design to optimize radar detection performance under accuracy and similarity constraints," *IEEE Trans. Signal Process.*, vol. 56, no. 11, pp. 5618–5629, Nov. 2008.

[8] A. Konar, N. Sidiropoulos, and O. Mehanna, "Parametric frugal sensing of power spectra for moving average models," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1073–1085, March 2015.

[9] J. R. Fienup, "Reconstruction of an object from the modulus of its fourier transform," *Optics Letters*, vol. 3, no. 1, pp. 27–29, 1978.

[10] M. Soltanalian, and P. Stoica, "Designing unimodular codes via quadratic optimization", *IEEE Trans. Signal Process.*, vol. 62, no. 5, pp. 1221–1234, March 2014.

[11] J. Carpentier, "Contribution to the economic dispatch problem", *Bulletin de la Societe Francoise des Electriciens*, vol. 3, no. 8, pp. 431–447, 1962.

[12] F. C. Schweppe, J. Wildes, and D. Rom, "Power system static state estimation: Parts I, II, and III," *IEEE Trans. Power App. Syst.*, vol. 89, pp. 120–135, Jan. 1970.

[13] M. Torun, A. Akansu, and M. Avellaneda, "Portfolio risk in multiple frequencies", *IEEE Signal Process. Mag.*, vol. 28, no. 5, pp. 61–71, Sept 2011.

[14] S. Boyd, and L. Vandenberghe, "*Convex Optimization*," Cambridge, U.K.: Cambridge Univ. Press, 2004.

[15] A. d'Aspremont, and S. Boyd, "Relaxations and Randomized Methods for Nonconvex QCQPs," *EE392 Lecture Notes, Stanford University*, 2003.

[16] Y. Huang and D. P. Palomar, "Randomized Algorithms for Optimal Solutions of Double-Sided QCQP With Applications in Signal Processing," *IEEE Trans. Signal Process.*, vol. 62, no. 5, pp 1093–1108, Mar. 2014.

[17] A. Ben-Tal and M. Teboulle, "Hidden convexity in some nonconvex quadratically constrained quadratic programming," *Math. Program.*, vol. 72, pp. 51–63, 1996.

[18] G. Pataki, "On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues," *Math. Operations Res.*, vol. 23, no. 2, pp. 339–358, 1998.

[19] Y. Ye and S. Zhang, "New results on quadratic minimization," *SIAM J. Optimiz.*, vol. 14, no. 1, pp. 245–267, 2003.

[20] A. Beck and Y. Eldar, "Strong duality in nonconvex quadratic optimization with two quadratic constraints," *SIAM J. Optimiz.*, vol. 17, no. 3, pp. 844–860, 2006.

[21] A. Beck and Y. Eldar, "Doubly constrained robust Capon beamformer with ellipsoidal uncertainty sets," *IEEE Trans. Signal Process.*, vol. 55, no. 2, pp. 753–758, Feb. 2007.

[22] Y. Huang and S. Zhang, "Complex matrix decomposition and quadratic programming," *Math. Operations Res.*, vol. 32, no. 3, pp. 758–768, 2007.

[23] W. Ai and S. Zhang, "Strong duality for the CDT subproblem: A necessary and sufficient condition," *SIAM J. Optimiz.*, vol. 19, no. 4, pp. 1735–1756, 2009.

[24] W. Ai, Y. Huang, and S. Zhang, "New results on Hermitian matrix rank-one decomposition," *Math. Program.: Series A*, vol. 128, no. 1-2, pp. 253–283, June 2011.

[25] Y. Huang, A. De Maio, and S. Zhang, "Semidefinite programming, matrix decomposition, and radar code design," Ch. 6 in *Convex Optimization in Signal Processing and Communications*, D. P. Palomar and Y. Eldar, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[26] Z. Y. Wu, D. Li, S. L. Zhang, and X. M. Wang, "Peeling off a nonconvex cover of an actual convex problem: hidden convexity," *SIAM J. Optimiz.*, vol. 18, no. 2, pp. 507–536, 2007.

[27] E. Karipidis, N. D. Sidiropoulos, and Z.-Q. Luo, "Far-field multicast beamforming for uniform linear antenna arrays," *IEEE Trans. Signal Process.*, vol. 55, no. 10, pp. 4916–4927, Oct. 2007.

[28] A. De Maio, S. De Nicola, Y. Huang, S. Zhang and A Farina, "Adaptive detection and estimation in the presence of useful signal and interference mismatches," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 436–450, Feb. 2009.

[29] Y. Huang and D. P. Palomar, "Rank-constrained separable semidefinite programming with applications to optimal beamforming," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp 664–678, Feb. 2010.

[30] Y. Huang and D. P. Palomar, "A dual perspective on separable semidefinite programming with applications to optimal downlink beamforming," *IEEE Trans. Signal Process.*, vol. 58, no. 8, pp. 4254–4271, Aug. 2010.

[31] A. De Maio, Y. Huang, D. P. Palomar, S. Zhang, and A. Farina, "Fractional QCQP with applications in ML steering direction estimation for radar detection," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 172–185, Jan. 2011.

[32] S. Bose, D. F. Gayme, K. M. Chandy, and S. H. Low, "Quadratically constrained quadratic programs on acyclic graphs with application to power flow", *arXiv preprint arXiv: 1203.5599*, 2012.

[33] H. Wolkowicz, "Relaxations of Q2P," in *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications*, H. Wolkowicz, R. Saigal, and L. Vandenberghe, Eds. Norwell, MA: Kluwer, 2000, ch. 13.4.

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

[35] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Rev.*, vol. 38, pp. 49–95, 1996.

[36] B. Marks and G. Wright, "A general inner approximation algorithm for nonconvex mathematical programs," *Oper. Res.*, vol. 26, no. 4, pp. 681–683, 1978.

[37] A. Beck, A. Ben-Tal, and L. Tetruashvili, "A sequential parametric convex approximation method with applications to non-convex truss topology design problems," *J. Global Optim.*, vol. 47, no. 1, pp. 29-51, 2010.

[38] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1126–1153, 2013.

[39] G. Scutari, F. Facchinei, L. Lampariello, and P. Song, "Parallel and distributed methods for nonconvex optimization-part I: Theory," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 1929-1944, Apr. 2017.

[40] O. Mehanna, K. Huang, B. Gopalakrishnan, A. Konar, and N. D. Sidiropoulos, "Feasible point pursuit and successive approximation of non-convex QCQPs", *IEEE Signal Process. Lett.*, vol. 22, no. 7, pp. 804–808, July 2015.

[41] C. I. Kanatsoulis, and N. D. Sidiropoulos, "Max-min feasible point pursuit for non-convex QCQP", in *Asilomar Conf. on Signals, Syst., and Comp.*, Pacific Grove, CA, Nov. 2015.

[42] K. Huang, and N. D. Sidiropoulos, "Consensus-ADMM for general quadratically constrained quadratic programming," *IEEE Trans. Signal Process.*, vol. 64, no. 20, pp. 5297–5310, Oct. 2016.

[43] S. Tu, R. Boczar, M. Soltanolkotabi, and B. Recht, "Low-rank solutions of linear matrix equations via procrustes flow," *arXiv preprint arXiv:1507.03566*, 2015.

[44] C. De Sa, K. Olukotun, and C. Re, "Global convergence of stochastic gradient descent for some non-convex matrix problems," *arXiv preprint arXiv:1411.1134v3*, 2015.

[45] R. Sun, and Z.-Q. Luo, "Guaranteed matrix completion via nonconvex factorization," *Proc. IEEE FOCS*, Oct. 17-20, 2015, Berkley, CA.

[46] Y. Chen, and M. J. Wainwright, "Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees", *arXiv preprint arXiv:1509.03025*, 2015.

[47] S. Bhojanapalli, A. Kyrillidis, and S. Sanghavi, "Dropping convexity for faster semi-definite optimization", *arXiv preprint arXiv:1509.03917*, 2015.

[48] S. Bhojanapalli, B. Neyshabur, and N. Srebro, "Global optimality of local search for low rank matrix recovery", *arXiv preprint arXiv:1605.07221v2*, 2016.

[49] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval via Wirtinger flow: Theory and algorithms," *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 1985–2007, 2015.

[50] Y. Chen, and E. J. Candes, "Solving random quadratic systems of equations is nearly as easy as solving linear systems," *Adv. Neural Info. Process Syst.*, pp. 739–747, 2015.

[51] J. Sun, Q. Qu, and J. Wright, "A geometric analysis of phase retrieval," *arXiv preprint arXiv:1602.06664*, 2016.

[52] A. Konar, and N. D. Sidiropoulos, "Hidden convexity in QCQP with Toeplitz-Hermitian quadratics," *IEEE Signal Process. Lett.*, vol. 22, no. 10, pp. 1623-1627, Oct. 2015.

[53] A. Konar, and N.D. Sidiropoulos, "A fast approximation algorithm for single-group multicast beamforming with large antenna arrays," in *Proc. 17th IEEE Int. Workshop on Signal Processing Advances in Wireless Communications* (SPAWC), Edinburgh, UK, July 2016.

[54] A. Konar, and N. D. Sidiropoulos, "Fast approximation algorithms for a class of nonconvex QCQP problems using first-order methods", *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3494–3509, July, 2017.

[55] A. Konar, and N.D. Sidiropoulos, "First-Order Methods for Fast Feasibility Pursuit of Non-convex QCQPs," in *Proc. IEEE ICASSP 2017*, New Orleans, USA, March 2017.

[56] A. Konar, and N.D. Sidiropoulos, "Fast feasibility pursuit for non-convex QCQPs via first-order methods," *IEEE Trans. Signal Process., to appear*, 2017.

[57] B. Dumitrescu, "*Positive trigonometric polynomials and signal processing applications*", Springer Science and Business Media, 2007.

[58] G. Dartmann and G. Ascheid, "Equivalent Quasi-Convex form of the multicast max-min beamforming problem," *IEEE Trans. Veh. Tech.,* vol. 62, no. 9, pp. 4643–4648, Nov. 2013.

[59] P. Stoica and R. Moses, "*Spectral Analysis of Signals*," Upper Saddle River, NJ: Pearson/Prentice Hall, 2005.

[60] B. Alkire and L. Vandenberghe, "Convex optimization problems involving finite autocorrelation sequences," *Math. Prog.*, vol. 93, no. 3, pp. 331–359, Dec. 2002.

[61] S.-P. Wu, S. Boyd, and L. Vandenberghe, "FIR filter design via semidefinite programming and spectral factorisation," *Proc. of IEEE Conf. Dec. and Control*, vol. 1, pp. 271–276, IEEE, 1996.

[62] Y. Nesterov, "Smooth minimization of non-smooth functions," *Math. Program.*, vol. 103, no. 1, pp 127-152, May 2005.

[63] A. Nemirovski, "Prox-method with rate of convergence O(1/T) for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems," *SIAM J. Optimiz.*, vol. 15, no. 1, pp. 229–251, 2004.

[64] R. Glowinski, and A. Marrocco, "Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualite, d'une classe de problems de Dirichlet non lineares," *Revue Francaise d'Automatique, Informatique, et Recherche Operationelle*, vol. 9, pp. 41-76, 1975.

[65] D. Gabay, and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximations," *Comp. Math. Appl.*, vol. 2, pp. 17–40, 1976.

[66] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1-122, 2011.

[67] R. T. Rockafellar, and R. J. B. Wets, *Variational Analysis*, Springer Science and Business Media, vol. 317, 2009.

[68] Y. Nesterov, and A. Nemirovski, *Interior Point Polynomial methods in Convex Programming: Theory and Applications*, SIAM, Philadelphia, 1994.

[69] B. T. Polyak, "*Introduction to Optimization*," Translations Series in Mathematics and Engineering, Optimization Software Inc. Publications Division, New York, 1987.

[70] N. Z. Shor, "*Minimization methods for non-differentiable functions*," Springer Series in Computational Mathematics, vol. 3, Springer-Verlag, Berlin, 1985.

[71] A. Nemirovski, D. Yudin, "*Problem complexity and method efficiency in Optimization*," John Wiley & Sons, 1983.

[72] Y. Nesterov, "A method for solving a convex programming problem with rate of convergence o(1/k^2)," *Soviet Math. Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

[73] L. Vandenberghe, "Smoothing", *EE236C Lecture Notes, UCLA*, 2010. Available `http://www.seas.ucla.edu/~vandenbe/236C/lectures/smoothing.pdf`.

[74] M. Sion, "On general minimax theorems," *Pacific J. Math.*, vol. 8, no. 1, pp. 171–176, 1958.

[75] F. Facchinei, and J. S. Pang, "*Finite-dimensional variational inequalities and complementarity problems*," Springer Series in Operations Research, Springer-Verlag, New York, 2003.

[76] E. K. Ryu, and S. Boyd, "Primer on monotone operator methods," *Appl. Comput. Math.*, vol. 15, no. 1, pp. 3–43, Jan. 2016.

[77] G. M. Korpelevich, "The extragradient method for finding saddle points and other problems," *Ekon. Mat. Metody*, vol. 12, pp. 747–756, 1976.

[78] A. Nemirovski, and D. Yudin, "On Cesaro's convergence of the gradient descent method for finding saddle points of convex-concave functions," *Doklady Akademii Nauk SSSR*, vol. 239, no. 4, 1978.

[79] A. Beck and M. Teboulle, "Mirror Descent and nonlinear projected subgradient methods for convex optimization," *Operations Res. Letters*, vol. 31, no. 3, pp. 167–175, May 2003.

[80] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, vol. 8, no. 3-4, pp. 231-357, 2015.

[81] N. Parikh, and S. Boyd, "Proximal Algorithms", *Found. Trends Optim.*, vol. 1, no. 3, pp. 123–231, 2013.

[82] L. Condat, "A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *J. Optim. Theory Appl.*, vol. 158, no. 2, pp. 460–479, 2013.

[83] B. He, and X. Yuan, "On the O(1/n) convergence of the Douglas-Rachford alternating direction method", *SIAM J. Numer. Anal.*, vol. 50, no. 2, pp. 700–709, April 2012.

[84] O. T. Demir and T. E. Tuncer, "Alternating maximization algorithm for the broadcast beamforming," in *Proc. EUSIPCO,* Lisbon, Portugal, Sep. 15, 2014.

[85] N. Bornhorst, and M. Pesavento, "An iterative convex approximation approach for transmit beamforming in multi-group multicasting," in *Proc. IEEE Int. Workshop on Signal Processing Advances in Wireless Communications (SPAWC),* pp. 411–415, San Francisco, CA, USA, June 2011.

[86] L. N. Tran, M. F. Hanif, and M. Juntti, "A conic quadratic programming approach to physical layer multicasting for large-scale antenna arrays," *IEEE Signal Process. Lett.*, vol. 21, no. 1, pp. 114-117, Jan. 2014.

[87] A. Lozano, "Long-term transmit beamforming for wireless multicasting," in *Proc. IEEE ICASSP*, Honolulu, HI, USA, Apr. 1520, 2007, pp. 417-420.

[88] E. Matskani, N. D. Sidiropoulos, Z. Q. Luo, and L. Tassiulas, "Efficient batch and adaptive approximation algorithms for joint multicast beamforming and admission control," *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4882-4894, Dec. 2009.

[89] A. Abdelkader, A. B. Gershman, and N. D. Sidiropoulos, "Multiple-antenna multicasting using channel orthogonalization and local refinement," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3922-3927, July 2010.

[90] B. Gopalakrishnan, and N. D. Sidiropoulos, "High performance adaptive algorithms for single-group multicast beamforming", *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4373-4384, Aug. 2015.

[91] F. Rusek, D. Persson, B. K. Lau, E. G. Larsson, T. L. Marzetta, O. Edfors, and F. Tufvesson, "Scaling up MIMO: Opportunities and challenges with very large scale arrays," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 40–60, Jan. 2013.

[92] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An overview of massive MIMO: Benefits and Challenges," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 742–758, Oct. 2014.

[93] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?," *IEEE J. Sel. Areas Comm.*, vol. 32, no. 6, pp. 1065–1082, June 2014.

[94] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Comm. Mag.*, vol. 52, no. 2, pp. 74–80, Feb. 2014.

[95] D. Christopoulos, S. Chatzinotas and B. Ottersten, "Multicast multigroup beamforming for per-antenna power constrained large-scale arrays," in *Proc. IEEE 16th Int. Workshop on Signal Processing Advances in Wireless Communications* (SPAWC), June 28 - July 1, 2015, Stockholm, pp. 271-275.

[96] APS Mosek, "The MOSEK optimization software", *Online at http://www. mosek. com*

[97] J. Lofberg, "Yalmip: A toolbox for modeling and optimization in MATLAB," in *Proc. CACSD*, Taipei, Sep. 4, 2004.

[98] S. R. Becker, E. J. Candes, and M. C. Grant, "Templates for convex cone problems with applications to sparse signal recovery," *Math. Prog. Comp.*, vol. 3, pp. 165-218, 2011.

[99] M. J. Lopez, "Multiplexing, scheduling, and multicasting strategies for antenna arrays in wireless networks," Ph.D. dissertation, Elect. Eng. and Comp. Sci. Dept., MIT, Cambridge, MA, 2002.

[100] M. Razaviyayn, "Successive convex approximation: Analysis and applications", Ph.D. dissertation, Dept. of Elect. and Comp. Eng., University of Minnesota, Minneapolis, MN, 2014.

[101] J. Nocedal and S. J. Wright, "*Numerical Optimization*," New York, NY, USA: Springer-Verlag, 2006.

[102] K. G. Murty and S. N. Kabadi, "Some NP-complete problems in quadratic and nonlinear programming," *Math. Prog.*, vol. 39, no. 2, pp. 117-129, 1987.

[103] G. B. Giannakis, V. Kekatos, N. Gatsis, S.-J. Kim, H. Zhu, and B. Wollenberg, "Monitoring and optimization for power grids: A signal processing perspective," *IEEE Signal Process. Mag.,* vol. 30, no. 5, pp. 107–128, Sept. 2013.

[104] K. Lehmann, A. Grastien, and P. Van Hentenryck, "AC-feasibility on tree networks in NP–Hard", *IEEE Trans. Power Syst.*, vol. 31, no. 1, pp. 798-801, Jan. 2016.

[105] R. Johnson, and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Adv. Neural Info. Process Syst.*, pp. 315–323, 2013.

[106] L. Xiao, and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM J. Optim.*, vol. 24, no. 4, pp. 2057-2075, 2014.

[107] Y. Nesterov, *Introductory Lectures on Convex Optimization*, vol. 87, Springer Science and Business Media, 2004.

[108] S. Ghadimi, and G. Lan, "Accelerated gradient methods for nonconvex nonlinear and stochastic programming", *Math. Prog.*, vol. 156, no. 1-2, pp. 59-99, 2016.

[109] S. Ghadimi, G. Lan, and H. Zhang, "Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization", *Math. Prog.*, vol. 155, no. 1-2, pp. 267-305, 2016.

[110] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht, "Gradient descent only converges to minimizers," *Proc. COLT*, pp. 1246–1257, 2016.

[111] M. Razaviyayn, M. Sanjabi, and Z.-Q. Luo, "A stochastic successive minimization method for nonsmooth nonconvex optimization with applications to transceiver design in wireless communication networks, " *Math. Prog.*, vol. 157, no. 2, pp. 515-545, June 2016.

[112] S. Ghadimi, and G. Lan, "Stochastic first and zeroth-order methods for nonconvex stochastic programming", *SIAM J. Optim.*, vol. 23, no. 4, pp. 2341-2368, 2013.

[113] S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. Smola, "Stochastic variance reduction for nonconvex optimization," *arXiv preprint arXiv:1603.06160v2*, 2016.

[114] R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle points–online stochastic gradient for tensor decomposition," *J. Mach. Learn. Res.*, vol. 40, pp. 1–46, June 2015.

[115] Z.-A. Zhu, and E. Hazan, "Variance reduction for faster non-convex optimization," *arXiv preprint arXiv:1603.05643*, 2016.

[116] S. J. Reddi, S. Sra, B. Poczos, and A. Smola, "Fast stochastic methods for nonsmooth nonconvex optimization," *arXiv preprint arXiv:1605.06900v1*, 2016.

[117] Y. Ye, "Data randomness makes optimization problems easier to solve?", *Technical Report*, available `http://web.stanford.edu/~yyye/NLPwithrandomdata.pdf`, Dec. 2016.

[118] L. Vandenberghe, "EE 133A Lecture Notes," *EE133A Lecture Notes, UCLA*, 2016. Available `http://www.seas.ucla.edu/~vandenbe/133A/133A-notes.pdf`.

[119] G. Wang, A. S. Zamzam, G. B. Giannakis, and N. D. Sidiropoulos, "Power system state estimation via feasible point pursuit", in *Proc. IEEE GlobalSIP*, Washington D.C., USA, Dec. 2016.

[120] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning and analysis tools for power systems research and education," {IEEE Trans. Power Syst.}, vol. 26, no. 1, pp. 12–19, Feb. 2011.

[121] C. Coffrin, D. Gordon, and P. Scott, "Nesta, the NICTA energy system test case archive", *arXiv preprint arXiv:1411.0359*, 2014.

[122] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed., Belmont, MA, USA: Athena Scientific, 1999.

[123] G. C. Calafiore and L. El Ghaoui, "*Optimization Models*," Cambridge, U.K.: Cambridge Univ. Press, 2014.

Appendix A

Proofs and Technical Claims

A.1 Proof of Proposition 3

In this section, we discuss the convergence properties of Algorithm 1. Our first goal is to establish that every limit point $\mathbf{x}_l$ of the iterates $\{\mathbf{x}^{(n)}\}_{n\in\mathbb{N}}$ generated by Algorithm 1 is a d-stationary point of (3.2); i.e., $f'(\mathbf{x}_l, \mathbf{d}) \geq 0$ for all $\mathbf{d}$ such that $\mathbf{x} + \mathbf{d} \in \mathcal{X}$. In order to do so, we will resort to [38, Theorem 1]. However, we first have to verify that the non-convex cost function $f(.)$ and its convex surrogate $v(.,.)$ satisfy the four conditions laid out in [38, Assumption 1]. By virtue of properties (A2-A4), simple inspection reveals that all but one of these conditions are apparently satisfied; the condition in question being [38, Assumption A3], which requires that the directional derivatives of $f(.)$ and $v(.,.)$ are equal at the point of approximation. This condition is hard to check in general, and a sufficient condition is proposed in [38, Proposition 1] under which it is automatically satisfied. Unfortunately, this sufficient condition does not hold in our case, which complicates matters. Nevertheless, by relying upon a different set of results borrowed from variational analysis [67], it is indeed possible to verify that [38, Assumption A3] is satisfied in our case, as we now show.

First, for ease of notation, we first introduce the definition $l_m(\mathbf{x}, \mathbf{x}^{(n)}) := \mathbf{c}_m^{(n)T}\mathbf{x} + d_m^{(n)}, \forall\, m \in [M]$. Now, consider the directional derivative of $v(\mathbf{x}, \mathbf{x}^{(n)}) = \max\limits_{m\in\mathcal{M}} l_m(\mathbf{x}, \mathbf{x}^{(n)})$, which, being a convex function, admits the following representation

$$v'(\mathbf{x}, \mathbf{x}^{(n)}; \mathbf{d}) = \max_{\mathbf{w}\in\partial v(\mathbf{x}, \mathbf{x}^{(n)})} \mathbf{w}^T\mathbf{d} \tag{A.1}$$

where $\partial v(\mathbf{x}, \mathbf{x}^{(n)}) = \text{conv}(\nabla l_i(\mathbf{x}, \mathbf{x}^{(n)}) |\, i \in \mathcal{M}(\mathbf{x}))$ and $\mathcal{M}(\mathbf{x}) := \{i \,|\, l_i(\mathbf{x}, \mathbf{x}^{(n)}) = v(\mathbf{x}, \mathbf{x}^{(n)})\} \subseteq [M]$. We then have that

$$v'(\mathbf{x}, \mathbf{x}^{(n)}; \mathbf{d})\Big|_{\mathbf{x}=\mathbf{x}^{(n)}} = \max_{\mathbf{w}\in\partial v(\mathbf{x}^{(n)}, \mathbf{x}^{(n)})} \mathbf{w}^T\mathbf{d} \tag{A.2}$$

where by construction of the surrogate function $v(.,.)$, we now have $\partial v(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) = \text{conv}(\nabla u_i(\mathbf{x}^{(n)}) | \, i \in \mathcal{M}(\mathbf{x}))$ and $\mathcal{M}(\mathbf{x}) := \{i \,|\, u_i(\mathbf{x}^{(n)}) = f(\mathbf{x}^{(n)})\}$. At this stage, it is fairly obvious that if we can establish a similar relationship for the directional derivative of $f(.)$ at $\mathbf{x} = \mathbf{x}^{(n)}$, the proof is complete. However, for non-convex functions, the representation (A.1) is not valid in general, which prevents us from establishing the desired result via the aforementioned arguments. Instead, under additional assumptions on f (which will be shown to be implicitly satisfied), and by exploiting the fact that f is the point-wise maximum of a finite number of smooth functions, we will utilize a different line of reasoning to derive an expression for $f'(\mathbf{x}; \mathbf{d})$, which, interestingly, will turn out to be the same as (A.2) at the point $\mathbf{x} = \mathbf{x}^{(n)}$.

Before we describe our approach in detail, we will require the following definitions. Adopting the exposition of [67], the *difference quotient function* associated with f at a point $\mathbf{x}$ (where $f(\mathbf{x})$ is finite) and a direction $\mathbf{d}$, is defined as

$$\Delta_\tau f(\mathbf{x})(\mathbf{d}) := \frac{f(\mathbf{x} + \tau\mathbf{d}) - f(\mathbf{x})}{\tau} \tag{A.3}$$

Clearly, we have

$$f'(\mathbf{x}; \mathbf{d}) := \lim_{\tau \downarrow 0} \Delta_\tau f(\mathbf{x})(\mathbf{d}) \tag{A.4}$$

The above definition can be generalized to define a *semiderivative* of f at $\mathbf{x}$ for $\mathbf{d}$ [67, Definition 7.20], which is given by

$$f_s(\mathbf{x}; \mathbf{d}) := \lim_{\substack{\tau \downarrow 0, \\ \mathbf{d}' \to \mathbf{d}}} \Delta_\tau f(\mathbf{x})(\mathbf{d}') \tag{A.5}$$

If the above limit exists, f is said to be semidifferentiable at $\mathbf{x}$ for $\mathbf{d}$. If it holds for all $\mathbf{d}$, then f is said to be semidifferentiable at $\mathbf{x}$. While $f'(\mathbf{x}; \mathbf{d})$ is only concerned with the limiting behavior of $\Delta_\tau f(\mathbf{x})(\mathbf{d})$ along the ray $\{\mathbf{x} + \tau\mathbf{d} | \tau \in \mathbb{R}_+\}$, the semiderivative, loosely speaking, tests the behavior of $\Delta_\tau f(\mathbf{x})(\mathbf{d})$ along all curves from $\mathbf{x}$ in the direction of $\mathbf{d}$. Clearly, if $f_s(\mathbf{x}; \mathbf{d})$ exists and is finite, then $f'(\mathbf{x}; \mathbf{d}) = f_s(\mathbf{x}; \mathbf{d})$. However, the converse is not true in general. When the existence of $f_s(\mathbf{x}; \mathbf{d})$ is not guaranteed, it is useful to work with *subderivatives* of $f(\mathbf{x})$ [67, Definition 8.1], which always exist and are defined as

$$\mathrm{d}f(\mathbf{x})(\mathbf{d}) = \liminf_{\substack{\tau \downarrow 0, \\ \mathbf{d}' \to \mathbf{d}}} \Delta_\tau f(\mathbf{x})(\mathbf{d}') \tag{A.6}$$

It is again evident that when $f_s(\mathbf{x}; \mathbf{d})$ exists, we must have $f_s(\mathbf{x}; \mathbf{d}) = \mathrm{d}f(\mathbf{x})(\mathbf{d})$ (since lim and lim inf coincide in this case). In addition, if $f_s(\mathbf{x}; \mathbf{d})$ is also finite, we obtain the following series of equalities

$$f'(\mathbf{x}; \mathbf{d}) = f_s(\mathbf{x}; \mathbf{d}) = \mathrm{d}f(\mathbf{x})(\mathbf{d}) \tag{A.7}$$

From (A.7), it can be inferred that it in order to obtain an expression for $f'(\mathbf{x}; \mathbf{d})$, it suffices to show that $f_s(\mathbf{x}; \mathbf{d})$ exists, compute $f_s(\mathbf{x}; \mathbf{d})$ (or possibly $\mathrm{d}f(\mathbf{x})(\mathbf{d})$) and verify that it is finite valued. This is precisely what we now set out to establish via the following claims.

Lemma 1. *The non-convex function* $f(\mathbf{x}) = \max_{m\in[M]} u_m(\mathbf{x})$ *is semidifferentiable for all* $\mathbf{x} \in \mathbb{R}^N$.

Proof. Follows directly from [67, Exercise 10.27(c)] □

Hence, although $f(\mathbf{x})$ is non-differentiable, the fact that it is the point-wise maximum of a finite number of smooth functions $\{u_m(\mathbf{x})\}_{m\in[M]}$ ensures that it is semidifferentiable. While this result establishes the existence of semiderivatives of $f(\mathbf{x})$, since we are interested in minimizing f over a convex, compact set $\mathcal{X}$, we require certain regularity assumptions on f [1] and $\mathcal{X}$ [2] being satisfied in order to proceed towards deriving an expression for the semiderivatives of f. The following result establishes that these regularity conditions are automatically satisfied in our case.

Lemma 2. *The set* $\mathcal{X}$ *is Clarke regular while* f *is subdifferentially regular for all* $\mathbf{x} \in \mathcal{X}$.

Proof. The first part follows directly from the fact that $\mathcal{X}$ is convex and then invoking [67, Theorem 6.4], while the second part holds due to the point-wise max structure of f which enables us to appeal to [67, Example 7.28]. □

Thanks to the above result, our overall convergence claims only depend upon the aforementioned regularity conditions *implicitly*; i.e., one does not have to check to see if they are verified; they are automatically guaranteed to hold by Claim 2. With these results in hand, we are now ready to state the main claim.

Lemma 3. *The subderivative of* f *for all* $\mathbf{x} \in \mathcal{X}$ *can be expressed as*

$$\mathrm{d}f(\mathbf{x})(\mathbf{d}) = \max_{i\in\mathcal{M}(\mathbf{x})} \nabla u_i(\mathbf{x})^T\mathbf{d} \tag{A.8}$$

where $\mathcal{M}(\mathbf{x}) := \{i \,|\, u_i(\mathbf{x}) = f(\mathbf{x})\}$. *Furthermore,* $\mathrm{d}f(\mathbf{x})(\mathbf{d}) < \infty$, $\forall\, \mathbf{x} \in \mathcal{X}$ *and* $\mathbf{d}$ *s.t.* $\mathbf{x}+\mathbf{d} \in \mathcal{X}$.

Proof. The first part of the claim follows from the regularity of $f(\mathbf{x})$, $\forall\, \mathbf{x} \in \mathcal{X}$, and then invoking [67, Exercise 8.31]. In order to show the second part, by our assumption that $\mathcal{X}$ is compact, we have that $\mathrm{diam}(\mathcal{X}) := \sup_{\mathbf{x},\mathbf{y}\in\mathcal{X}} \|\mathbf{x}-\mathbf{y}\|_2 = D < \infty$ for some $D \in \mathbb{R}_+$. This enables us to write

$$\begin{aligned} \nabla u_i(\mathbf{x})^T\mathbf{d} &\leq \|\nabla u_i(\mathbf{x})\|_2\|\mathbf{d}\|_2 = \|\bar{\mathbf{A}}_i\mathbf{x}\|_2\|\mathbf{d}\|_2 \\ &\leq \|\bar{\mathbf{A}}_i\|_2\|\mathbf{x}\|_2\|\mathbf{d}\|_2 \\ &\leq \|\bar{\mathbf{A}}_i\|_2 D^2, \forall\, i \in \mathcal{M}(\mathbf{x}) \\ \implies \max_{i\in\mathcal{M}(\mathbf{x})} \nabla u_i(\mathbf{x})^T\mathbf{d} &\leq D^2 \max_{i\in\mathcal{M}(\mathbf{x})} \|\bar{\mathbf{A}}_i\|_2 < \infty \end{aligned} \tag{A.9}$$

□

[1] Here, by regularity of f, we mean that f satisfies the notion of *subdifferential regularity* as defined in [67, Definition 7.25].

[2] By regularity of the set $\mathcal{X}$, we mean that $\mathcal{X}$ satisfies *Clarke regularity* as defined in [67, Definition 6.4].

Taken together, our claims guarantee that $f_s(\mathbf{x};\mathbf{d})$ always exists (Lemma 1), establish that all requisite regularity conditions are automatically satisfied by f and $\mathcal{X}$ (Lemma 2), and also provide a characterization of $\mathrm{d}f(\mathbf{x})(\mathbf{d})$, which is finite-valued over $\mathcal{X}$ (Lemma 3). Thus, it follows that the chain of equalities (A.7) hold in our case, which allows us to directly write

$$f(\mathbf{x};\mathbf{d}) = \max_{i\in\mathcal{M}(\mathbf{x})} \nabla u_i(\mathbf{x})^T\mathbf{d} = \max_{\substack{\mathbf{w}\in\mathrm{conv}(\nabla u_i(\mathbf{x})|\\ i\in\mathcal{M}(\mathbf{x}))}} \mathbf{w}^T\mathbf{d} \tag{A.10}$$

Comparing the above expression with (A.2), it directly follows that we have

$$f'(\mathbf{x};\mathbf{d})\Big|_{\mathbf{x}=\mathbf{x}^{(n)}} = v'(\mathbf{x},\mathbf{x}^{(n)};\mathbf{d})\Big|_{\mathbf{x}=\mathbf{x}^{(n)}} \tag{A.11}$$

which is the condition that we set out to verify. Note that, by the feasibility of the iterates $\{\mathbf{x}^{(n)}\}_{n\in\mathbb{N}}$ of Algorithm 1, (A.10) always holds for *every* SCA iteration n.

Now that we have verified all four conditions listed in [38, Assumption 1], it only remains to invoke [38, Theorem 1] to claim that every limit point $\mathbf{x}_l$ of $\{\mathbf{x}^{(n)}\}_{n\in\mathbb{N}}$ is a d-stationary point of (3.2). Of course, this is a weaker claim compared to what we stated in Proposition 3, as it only guarantees convergence along a subsequence of the iterates (provided that a convergent subsequence exists in the first place). While the compactness of $\mathcal{X}$ guarantees the existence of such a convergent subsequence, it also allows us to strengthen our result to achieve the desired outcome via the following lemma.

Lemma 4. *If $\mathcal{X}$ is a compact set, then, under [38, Assumption 1], the sequence of iterates $\{\mathbf{x}^{(n)}\}_{n\in\mathbb{N}}$ satisfy*

$$\lim_{n\to\infty} d(\mathbf{x}^{(n)},\mathcal{X}^*) = 0,$$

where $\mathcal{X}^*$ is the set of d-stationary solutions of (3.2).

Proof. Follows directly from [38, Corollary 1]. □

This concludes the proof of Proposition 3.

A.2 Proximal Operator of $\omega(.)$

The results of this appendix are derived in a manner similar to that in [81, Section 6.4]. Determining the proximal operator $\mathrm{prox}_{\frac{\omega}{\rho}}(\mathbf{x})$ of the function $\omega(\mathbf{y}) = \max_{m\in[M]}\{y_m + b_m\}$ requires one to solve the following convex optimization problem

$$\min_{\mathbf{y}} \omega(\mathbf{y}) + \frac{\rho}{2}\|\mathbf{y}-\mathbf{x}\|_2^2 \tag{A.12}$$

which can be represented in its epigraph form as the following smooth optimization problem

$$\min_{\mathbf{y},t} \quad t + \frac{\rho}{2}\|\mathbf{y}-\mathbf{x}\|_2^2 \tag{A.13a}$$

$$\text{s.t.} \quad y_m + b_m \leq t, \forall\ m \in [M] \tag{A.13b}$$

The KKT optimality conditions for (A.13) are given by

$$y_m^* + b_m \leq t^*, \tag{A.14a}$$

$$\eta_m^* \geq 0, \tag{A.14b}$$

$$\eta_m^*(y_m^* + b_m - t^*) = 0, \tag{A.14c}$$

$$\rho(y_m^* - x_m) + \eta_m^* = 0, \tag{A.14d}$$

$$\sum_{m=1}^{M} \eta_m^* = 1 \tag{A.14e}$$

where $m \in [M]$ and $\boldsymbol{\eta} = [\eta_1, \cdots, \eta_M]^T$ denotes the vector of dual variables. If $y_m^* + b_m < t^*$, then from the third condition, we have $\eta_m^* = 0$. Otherwise, if $y_m^* + b_m = t^*$, then from the fourth condition we obtain $\eta_m^* = \rho(x_m + b_m - t^*)$. Since $\eta_m^* \geq 0$, we must have that

$$\eta_m^* = \rho \max\{x_m + b_m - t^*, 0\} \tag{A.15}$$

Substituting for η_m^* in the final KKT condition, we obtain the equation

$$\rho \sum_{m=1}^{M} \max\{x_m + b_m - t^*, 0\} = 1 \tag{A.16}$$

which can be solved for t^* via bisection using the initial interval $[\min_m\{x_m+b_m\}-(1/\rho M), \max_m\{x_m+b_m\}]$. Once t^* is determined, we can solve for $\mathbf{y}^* = [y_1^*, \cdots, y_M^*]^T$ as

$$\begin{aligned} y_m^* &= x_m - \max\{x_m + b_m - t^*, 0\} \\ &= \min\{t^* - b_m, x_m\}, \forall\ m \in [M] \end{aligned} \tag{A.17}$$

The proximal operator of $\omega(.)$ is then given by

$$\text{prox}_{\frac{\omega}{\rho}}(\mathbf{x}) = \mathbf{y}^* \tag{A.18}$$

A.3 MU Algorithm for massive MIMO Multicasting

In this appendix, we derive a variant of the MU algorithm described in [90] for handling PAPCs in Massive MIMO multicasting. As a surrogate for the max-min fair problem (3.52), consider the following proportionally-fair formulation

$$\max_{\tilde{\mathbf{w}} \in \tilde{\mathcal{W}}} \sum_{m=1}^{M} \log\left(\tilde{\mathbf{w}}^T \tilde{\mathbf{R}}_m \tilde{\mathbf{w}} + \delta\right) \tag{A.19}$$

where $\delta \in \mathbb{R}_{++}$. After expressing the set $\tilde{\mathcal{W}}$ in terms of PAPCs, we obtain the following non-convex problem

$$\max_{\tilde{\mathbf{w}} \in \mathbb{R}^{2N}} \sum_{m=1}^{M} \log\left(\tilde{\mathbf{w}}^T \tilde{\mathbf{R}}_m \tilde{\mathbf{w}} + \delta\right) \tag{A.20a}$$

$$\text{s.t.} \quad \tilde{w}^2(i) + \tilde{w}^2(i+N) \leq P_i, \forall\, i \in [N] \tag{A.20b}$$

The MU algorithm proposes to solve (A.20) in the following iterative manner. Starting from an initial feasible point $\tilde{\mathbf{w}}^{(0)}$, at each iteration $n \geq 0$, we construct the following first order surrogate of $h(\tilde{\mathbf{w}}) := \sum_{m=1}^{M} \log\left(\tilde{\mathbf{w}}^T \tilde{\mathbf{R}}_m \tilde{\mathbf{w}} + \delta\right)$ about the current iterate $\tilde{\mathbf{w}} = \tilde{\mathbf{w}}^{(n)}$

$$h(\tilde{\mathbf{w}}) \approx h(\tilde{\mathbf{w}}) + \nabla h(\tilde{\mathbf{w}}^{(n)})^T (\tilde{\mathbf{w}} - \tilde{\mathbf{w}}^{(n)}) \tag{A.21}$$

by determining the first-order Taylor series expansion of $h(\tilde{\mathbf{w}})$ about $\tilde{\mathbf{w}}^{(n)}$. The gradient of $h(\tilde{\mathbf{w}})$ is given by

$$\nabla h(\tilde{\mathbf{w}}) := \sum_{m=1}^{M} \frac{2\tilde{\mathbf{R}}_m \tilde{\mathbf{w}}}{\tilde{\mathbf{w}}^T \tilde{\mathbf{R}}_m \tilde{\mathbf{w}} + \delta} \tag{A.22}$$

which corresponds to taking an inversely weighted combination of the gradients of the functions $u_m(\tilde{\mathbf{w}}) = \tilde{\mathbf{w}}^T \bar{\mathbf{R}}_m \tilde{\mathbf{w}}, \forall\, m \in [M]$, with more emphasis placed on the gradients of those functions $u_m(\tilde{\mathbf{w}})$ which are small. This intuitively suggests that $\nabla h(\tilde{\mathbf{w}})$ corresponds to a good search direction for attaining max-min fairness. The update rule of our algorithm at each iteration n is then given by

$$\tilde{\mathbf{w}}^{(n+1)} = \arg\max_{\tilde{\mathbf{w}} \in \tilde{\mathcal{W}}} \nabla h(\tilde{\mathbf{w}}^{(n)})^T (\tilde{\mathbf{w}} - \tilde{\mathbf{w}}^{(n)}) \tag{A.23a}$$

$$= \arg\max_{\substack{\tilde{w}^2(i) + \tilde{w}^2(i+N) \leq P_i, \\ \forall\, i \in [N]}} \nabla h(\tilde{\mathbf{w}}^{(n)})^T \tilde{\mathbf{w}} \tag{A.23b}$$

Define $\tilde{\mathbf{g}}^{(n)} := \nabla h(\tilde{\mathbf{w}}^{(n)})$. Then, the objective function of (A.23b) can be expressed as

$$\tilde{\mathbf{g}}^{(n)T} \tilde{\mathbf{w}} = \sum_{i=1}^{2N} \tilde{g}^{(n)}(i) \tilde{w}(i) = \sum_{i=1}^{N} \bar{\mathbf{g}}_i^{(n)T} \bar{\mathbf{w}}_i \tag{A.24}$$

where $\bar{\mathbf{g}}_i^{(n)} := [\tilde{g}^{(n)}(i), \tilde{g}^{(n)}(i+N)]^T, \bar{\mathbf{w}}_i := [\tilde{w}(i), \tilde{w}(i+N)]^T, \forall\, i \in [N]$. With the objective function represented in this form, it is obvious that (A.23b) decomposes into N parallel problems of the form

$$\bar{\mathbf{w}}_i^{(n+1)} = \arg\max_{\|\bar{\mathbf{w}}_i\|_2^2 \leq P_i} \bar{\mathbf{g}}_i^{(n)T} \bar{\mathbf{w}}_i = \sqrt{P_i} \frac{\bar{\mathbf{g}}_i}{\|\bar{\mathbf{g}}_i\|_2}, \forall\, i \in [N] \tag{A.25}$$

From the vectors $\{\bar{\mathbf{w}}_i^{(n+1)}\}_{i=1}^{N}$, the update vector $\tilde{\mathbf{w}}_i^{(n+1)}$ can be easily synthesized.

A.4 Derivation of smoothed-hinge function

Consider the following maximization problem

$$f_m^{(\mu)}(\mathbf{x}) = \max_{0\leq y\leq 1}\{y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \mu\frac{y^2}{2}\}, \forall\, m \in [M_I] \tag{A.26}$$

Note that for every $\mathbf{x} \in \mathbb{R}^N$, the corresponding maximization problem is strongly concave in y, and hence the maximum is always uniquely attained. Define the function $g(y) := y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \mu\frac{y^2}{2}$. In order to obtain a closed form solution to (A.26), we consider the following three cases.

1. $\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m \leq 0$: In this case, it can be readily seen that the choice of y which maximizes $g(y)$ over the interval $[0, 1]$ is $y = 0$. Thus, we obtain
$$f_m^{(\mu)}(\mathbf{x}) = 0, \text{ if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m \tag{A.27}$$

2. $0 < \mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m \leq \mu$: The function $g(y)$ attains its maximum at $y = \frac{\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m}{\mu}$, which in this case, lies in the interval $(0, 1]$. Substituting this value in (A.26) yields
$$f_m^{(\mu)}(\mathbf{x}) = \frac{(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)^2}{2\mu}, \text{if } b_m < \mathbf{x}^T\mathbf{A}_m\mathbf{x} \leq b_m + \mu \tag{A.28}$$

3. $\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m > \mu$: In this case, the function $g(y)$ attains its maximum at a point $y > 1$ which lies outside the interval $[0, 1]$. As the function is monotonically increasing, we choose the value $y = 1$ which maximizes $g(y)$ over $[0, 1]$ to obtain
$$f_m^{(\mu)}(\mathbf{x}) = \mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m - \frac{\mu}{2}, \text{if } \mathbf{x}^T\mathbf{A}_m\mathbf{x} > b_m + \mu \tag{A.29}$$

Meanwhile, the approximation bounds (4.20) can be derived using the same arguments first used in [62] for establishing approximation bounds for non-smooth, *convex* functions using Nesterov smoothing. Here, we show that these results can also be extended to our non-convex setting.

In order to derive the desired bounds (4.20), we will equivalently show that the following result holds.

$$f_m(\mathbf{x}) - \frac{\mu}{2} \leq f_m^{(\mu)}(\mathbf{x}) \leq f_m(\mathbf{x}), \forall\, \mathbf{x} \in \mathbb{R}^N, \forall\, m \in [M_I] \tag{A.30}$$

It is evident that the upper bound holds by inspection of the definitions of $f_m(\mathbf{x})$ (4.16) and $f_m^{(\mu)}(\mathbf{x})$ (4.17). To show that the lower bound holds, we first note that $\max_{0\leq y\leq 1}\frac{y^2}{2} = \frac{1}{2}$. Hence, it follows that for any $\mathbf{x} \in \mathbb{R}^N, \mu \in \mathbb{R}_{++}$ and $m \in [M_I]$, we have

$$\begin{aligned} & y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \mu\frac{y^2}{2} \geq y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \frac{\mu}{2}, \forall\, y \in [0, 1] \\ \implies & \max_{0\leq y\leq 1}\{y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m) - \mu\frac{y^2}{2}\} \geq \max_{0\leq y\leq 1}\{y(\mathbf{x}^T\mathbf{A}_m\mathbf{x} - b_m)\} - \frac{\mu}{2}, \\ \implies & f_m^{(\mu)}(\mathbf{x}) \geq f_m(\mathbf{x}) - \frac{\mu}{2} \end{aligned} \tag{A.31}$$

This concludes the proof.

A.5 Convergence of Gradient Descent without Lipschitz smoothness

In this section, we establish that GD with backtracking line search globally converges to the set of stationary points of the quadratic penalty formulation for solving general systems of quadratic equations (i.e., (4.22) with $M_I = 0$), without any Lipschitz smoothness assumptions. We emphasize that this is not a new result, but rather a refinement of the following pre-existing result.

Lemma 5. *[122, Proposition 1.2.1] Let $\{\mathbf{x}^{(k)}\}$ be a sequence of iterates generated by GD with backtracking line search for step-size selection. Then, every limit point of $\{\mathbf{x}^{(k)}\}$ is a stationary point.*

This result, while not requiring any Lipschitz smoothness assumptions, is still contingent on a general condition for gradient based methods being satisfied, which requires that the descent direction does not become asymptotically orthogonal to the gradient, unless the gradient vanishes (see [122, p. 35]). We point out that GD naturally satisfies this condition, thus implying that the convergence claim stated above holds. However, this result, on its own, is rather weak. It states that *if* the sequence $\{\mathbf{x}^{(k)}\}$ possesses a convergent subsequence, then the limit of the *subsequence* is a stationary point. In order to further improve upon this claim, we proceed as follows

1. First, we show that irrespective of initialization, the sequence of iterates $\{\mathbf{x}^{(k)}\}$ generated by GD with backtracking line search always possesses a convergent subsequence.

2. Using the above result in conjunction with Lemma 1, we will show that the *entire sequence* $\{\mathbf{x}^{(k)}\}$ converges to a stationary point.

The first condition can be established by exploiting the fact the cost function is a quartic polynomial which is *coercive*[3] . A useful attribute of coercive functions is that they satisfy the following property.

Lemma 6. *[123, Lemma 8.3] Let f be a continuous, coercive function. Then, every sub-level set $\mathcal{X}_\gamma := \{\mathbf{x} \in \mathbb{R}^N \mid f(\mathbf{x}) \leq \gamma\}$ of f is compact.*

[3] A function $f : \mathbb{R}^N \to \mathbb{R}$ is said to be coercive if for every sequence $\{\mathbf{x}^{(k)}\}$ for which $\|\mathbf{x}^{(k)}\| \to \infty$, we have $\lim_{k\to\infty} f(\mathbf{x}^{(k)}) = \infty$ [122, Definition A.4].

This implies that for any point $\mathbf{x}^{(0)} \in \mathbb{R}^N$ used to initialize GD, the initial sub-level set $\mathcal{X}_{f(\mathbf{x}^{(0)})} := \{\mathbf{x} \in \mathbb{R} \mid f(\mathbf{x}) \leq f(\mathbf{x}^{(0)})\}$ is always compact. Since backtracking line search is used to ensure descent of GD at each iteration, it follows that the entire sequence of iterates $\{\mathbf{x}^{(k)}\}$ generated by GD lie in $\mathcal{X}_{f(\mathbf{x}^{(0)})}$, and hence, are bounded. By appealing to the Weierstrass theorem [122, Proprosition A.8], it then follows that $\{\mathbf{x}^{(k)}\}$ possesses a convergent subsequence.

In order to complete the proof, we simply invoke [38, Corollary 1] once again. Hence, it follows that the entire sequence $\{\mathbf{x}^{(k)}\}$ globally converges to the set of stationary points, in an asymptotic sense.

Appendix B

Acronyms

This appendix contains a table of acronyms and their meaning.

Table B.1: Acronyms

Acronym	Meaning
ADMM	Alternating Direction Method of Multipliers
C-ADMM	Consensus-ADMM
DFT	Discrete Fourier Transform
eMBMS	Evolved Multimedia Broadcast Multicast Service
FAS	Finite Autocorrelation Sequence
FOM	First-order Method
FPP	Feasible Point Pursuit
GD	Gradient Descent
GN	Gauss-Newton
IPM	Interior-Point method
LP	Linear Programming
LMI	Linear Matrix Inequality
LTE	Long-Term Evolution
MIMO	Multiple-Input Multiple-Output
NMSE	Normalized Mean Squared Error
PSSE	Power System State Estimation
QCQP	Quadratically Constrained Quadratic Programming
SCA	Successive Convex Approximation

Continued on next page

Table B.1 – continued from previous page

Acronym	Meaning
SCADA	Supervisory Control and Data Acquisition
SD	Subgradient Descent
SDP	Semidefinite Programming
SDR	Semidefinite Relaxation
SGD	Stochastic Gradient Descent
SoCP	Second-order Cone Programming
SSGD	Stochastic Subgradient Decent
SVRG	Stochastic Variance Reduced Gradient