Fast Unit-modulus Least Squares with Applications in Beamforming and Phase Retrieval

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

John Tranter

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Nicholas D. Sidiropoulos

May, 2016

Acknowledgements

I cannot express enough thanks to my research advisor, Dr. Nicholas Sidiropoulos, for his continued support and guidance. I am extremely grateful for the mentorship and learning opportunities I have received as a member of his group. Additionally, thank you to committee members Dr. Zhi-Quan Luo and Dr. Ankur Mani for their time and efforts. Finally, I am thankful for the expertise of Dr. Xiao Fu; his insights have been an enormous contribution to this thesis.

Dedication

To my parents William and Judith Tranter, for their love and encouragement

Abstract

Unit-modulus Least Squares (ULS) problems arise in many applications, including phase-only beamforming, phase retrieval and radar code design. These problems are NP-hard, so there exist problem instances that we cannot solve efficiently, and whether such solution exists remains elusive. ULS formulations can always be recast as Unit-modulus Quadratic Programs (UQPs), to which Semi-Definite Relaxation (SDR) can be applied, and is often the state-of-the-art approach (e.g., *PhaseCut*). SDR lifts the problem dimension and requires solving a much larger-scale convex problem, which makes it ill-suited for large-scale ULS/UQP. In this paper, we propose several first-order algorithms that meet or exceed SDR performance in terms of minimizing the cost function, and compare favorably to SDR in terms of runtime complexity. We establish convergence of the proposed first-order algorithms to a Karush-Kuhn-Tucker (KKT) point, and we demonstrate their superiority in two applications: beamforming using only phase-shifts, and phase retrieval.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Unit-modulus[1] Least Squares (ULS) optimization problems have many contemporary engineering applications, most notably in signal processing, wireless communications, and radar. Applications in phase-only receive and transmit beamforming readily admit a ULS representation. ULS problems can be transformed to Unit-modulus Quadratic Programs (UQPs), and vice-versa. Other problems, such as Unit-modulus radar code design and phase retrieval, have been cast as UQP, and can therefore be transformed to ULS as well. Although the general ULS/UQP problem is non-convex and NP-hard [1] due to the Unit-modulus constraints, it is also a special case of non-convex quadratically constrained quadratic programming (QCQP), to which a popular relaxation technique known as Semi-Definite (rank) Relaxation (SDR) [2] can be readily applied. When SDR returns a rank-one solution, this is optimal for the original problem as well. When SDR returns a higher-rank solution, however, one has to resort to randomization, and the global optimum may still be elusive, even after many randomization trials; see [2] and references therein. SDR is not well-suited for large-scale problems; due to lifting, its complexity is higher than sixth-order polynomial in the original problem size. Many modern applications of ULS/UQP require scalable optimization methods that can be used to tackle large-scale problem instances. For example, Multiple-Input-Multiple-Output (MIMO) communication systems have been of interest for over 15 years, due to the performance improvements that they enable. Arrays with multiple antennas enable

[1] Each (complex-valued) element of the sought LS solution has unit modulus. Note difference with *(total) unimodularity*, defined for real-valued integer matrices having ± 1 minors.

higher data rates, as well as longer reach and improved link reliability [3, 4]. At present, MIMO systems promise to play a major role in the evolution toward fifth-generation (5G) wireless technology. One of the emerging 5-G paradigms envisions equipping each base station with many more antennas than the number of active users in its service cell [5]. Phase retrieval and radar code design may likewise entail very high-dimensional optimization, especially in high-resolution scenarios. Because of the high-dimensional nature of these problems, many of the previously developed approximation methods become impractical.

1.1 Related Work

Phased-array beamforming picks the modulus and phase of each element of the complex beamforming vector to approximate a desired spatial beampattern. This corresponds to using a separate phase shifter and power amplifier for each antenna, which is costly. In many applications, especially those involving 'massive' antenna arrays and/or small form-factor / low power mobile devices, it is preferable to use a single power amplifier for all antennas, and rely on per-antenna phase shifters to steer the beam in the direction(s) of interest. This gives rise to a constant-modulus constraint on the beamforming vector. Several algorithms have been proposed for phase-only beamforming, such as phase perturbation methods [6]-[7]. Thompson [8] utilized a gradient-search method (with an explicit angle parametrization of the Unit-modulus constraint) for adaptively adjusting the phase shifters. Smith [9] proposed a combination of conjugate gradient and Newton algorithms for phase-only adaptive nulling, where optimization is performed over the N-dimensional Unit-modulus torus (with an angle paramaterization of the Unit-modulus constraint). The conjugate gradient algorithm is employed to first approach the solution, followed by multiple Newton refinements. The algorithm requires calculation and inversion of the Hessian matrix at every iteration, and thus is not ideal for large-scale problems. Choi and Sarkar [10] devised a direct data domain least squares algorithm, which adaptively adjusts the phase weights from snapshots of complex voltages at each antenna element via a conjugate gradient algorithm (also with an angle parametrization).

A common denominator of early approaches is that they employ an explicit angle parametrization of the Unit-modulus constraint that enables classic unconstrained optimization methods such as gradient descent to be applied to this non-convex and (NP-)hard problem. SDR is a much more advanced approach for this type of problem, and a closely related phase-only beamforming formulation employing a linearly constrained minimum variance (LCMV) criterion has been considered in Lu *et al.* [11], who proposed SDR for its approximation. Our experience is that methods based on explicit angle parametrization perform worse than SDR, so we will use SDR as the state-of-art for comparison in the sequel.

Phase retrieval is another application where Unit-modulus optimization methods can be used. Several authors [12]-[13] have proposed alternating optimization algorithms for the measurement model $\mathbf{y} = |\mathbf{A}\mathbf{x}|$ (+ noise), which are unfortunately prone to stalling at local minima. More recently, Waldspurger *et al.* [14] devised a relaxation (called *PhaseCut*) that formulates the phase retrieval problem as an SDP similar to the SDR approximation of the classical MaxCut problem. A block coordinate descent algorithm is employed that is favorable to general SDR in terms of computational complexity.

Motivated by applications in Unit-modulus (constant-envelope) radar code design, Soltanalian and Stoica [15] proposed a monotonically error-bound improving technique (MERIT), as well as a power method-like iteration for UQP optimization problems. MERIT is comparatively complex, but it provides a sub-optimality guarantee that is sometimes tighter than that provided by SDR. The power-like iteration in [15] can be used to improve any initial estimate at a relatively low (second-order) cost, but the ultimate result depends a lot on initialization. Reference [15] also showed that a host of other problems can be formulated as UQP; but it didn't make the connection to phase retrieval via PhaseCut and the associated fast SDR implementation, which is in fact applicable to any UQP.

1.2 Contributions

In this paper, we propose several low-complexity first-order algorithms for ULS/UQP. In particular, we demonstrate that *projected* (as opposed to *parametrized*) gradient descent methods perform comparably (and in some cases, favorably) to SDR in terms of least

squares cost, at lower complexity. First-order methods are naturally appealing for large-scale problems, but for non-convex and NP-hard problems such as ULS/UQP, one would normally have low expectations from simple first-order methods. If state-of-the-art high-complexity approximations like SDR can fail, how well can first-order methods work?

To the best of our knowledge, there is very little work on convergence of first-order (e.g., projected gradient) methods for non-convex problems, like the ones we consider here. After considerable experimentation and initial successes that prompted us to develop new theoretical insights, we found that relatively simple first-order methods can perform surprisingly well, provided one avoids explicit parametrization of the non-convex constraint and pays attention to initialization and proper selection of step-size. What is more, we were able to establish convergence of these simple algorithms to a Karush-Kuhn-Tucker (KKT) point of the original NP-hard problem. We exemplify the comparative advantages of these first-order methods (both in terms of cost minimization and runtime complexity) relative to existing popular approaches, such as SDR, using phased-array beamforming and phase retrieval as illustrative applications. Furthermore, we employ the Cramér-Rao Bound (CRB), which is normally used as a performance bound in estimation problems, as a benchmark for optimization-based NP-hard design problems, by associating them with corresponding estimation problems.

Chapter 2

Problem Statements

2.1 Phase-Only Beamforming

In receive beamforming, complex weights are applied to the signals coming from the different antennas before combining them, to create a desired spatial beampattern that enhances signals from specified directions of interest and attenuates those from other directions. In transmit beamforming, a common information-bearing signal is fed to multiple antennas, using a different complex weight (corresponding to separate power amplification and carrier phase shift) per antenna. Transmit beamforming can be *unicast* (pointing to a single receiver of interest) or *multicast* (pointing to multiple receivers, interested in a common information stream). In both cases, the objective is to steer power in the direction(s) of interest while mitigating interference to other users within transmission range. Although many classical beamforming scenarios apply both magnitude and phase weighting to the antenna elements, there are many advantages to phase-only beamforming [11, 16, 17], especially for massive MIMO systems where one is interested in using many antennas but few up/down conversion chains.

For simplicity, let us consider a Uniform Linear Array (ULA) comprising N antennas with equidistant spacing ($\lambda/2$, where λ is the wavelength of the carrier frequency) (our formulation can handle any beamforming scenario with a known array manifold). Let $\boldsymbol{\theta}$ denote an $M \times 1$ vector representing a discretization of the angle space, i.e.,

$$\boldsymbol{\theta} = \left[0, \frac{2\pi}{M}, \frac{4\pi}{M}, \ldots, \frac{2(M-1)\pi}{M}\right]. \tag{2.1}$$

In a ULA scenario, the $N \times 1$ steering vectors have Vandermonde structure $\mathbf{a}(\theta_i) = [1, e^{-j\theta_i}, e^{-j2\theta_i}, \ldots, e^{-j(N-1)\theta_i}]^T$. We can then construct our design matrix

$$\mathbf{A} = [\mathbf{a}(\theta_1), \ldots, \mathbf{a}(\theta_M)]^H, \tag{2.2}$$

where $(\cdot)^H$ denotes the Hermitian, or conjugate transpose, and formulate the following Unit-modulus Least Squares (ULS) optimization problem

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{C}^N} \quad & \|\mathbf{y} - \mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1, \ldots, N, \end{aligned} \tag{2.3}$$

where w_i is the i-th element of vector $\mathbf{w}$. Let $\mathcal{J}$ denote the set containing the indices i corresponding to the direction(s) of interest in $\boldsymbol{\theta}$; $\mathbf{w} \in \mathbb{C}^{N \times 1}$ represents the unit-modulus beamforming weight vector. We wish to construct a vector $\mathbf{y}$ that will encourage $\mathbf{w}$ to yield a desired power response for each $|\mathbf{a}^H(\theta_i)\mathbf{w}|^2$, where θ_i is a desired broadcast direction. First consider the following choice for $\mathbf{y}$:

$$y_i = \begin{cases} 1 & \text{if } i \in \mathcal{J}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

Note that

$$|\mathbf{a}^H(\theta_i)\mathbf{w}| = \left| \sum_{j=1}^{N} a_j^*(\theta_i) w_j \right| \leq \sum_{j=1}^{N} |a_j^*(\theta_i) w_j| = N, \tag{2.5}$$

and, from the Cauchy-Schwartz inequality, the maximum is achieved if and only if $\mathbf{w} = e^{j\phi}\mathbf{a}$. Therefore, the scaling of $\mathbf{y}$ matters in our ULS formulation, even though in beamforming we are really interested in the (relative) beampattern. In order to compensate for this, we can introduce an additional scaling variable $s \in \mathbb{C}$ to obtain

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{C}^N, s \in \mathbb{C}} \quad & \|\mathbf{y} - s\mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1, \ldots, N. \end{aligned} \tag{2.6}$$

Exploiting separability, we may compute and substitute the optimal s as a function of $\mathbf{w}$:

$$s_{opt} = \frac{\mathbf{w}^H \mathbf{A}^H \mathbf{y}}{\|\mathbf{A}\mathbf{w}\|^2}. \tag{2.7}$$

The resulting problem turns out to be a UQP, as we will see. Figures 2.1 illustrate an example multicast beampattern aimed towards two users in the directions $\{-20^o, +20^o\}$; note that the geometry of the ULA scenario will result in symmetry about the axis defined by the ULA.

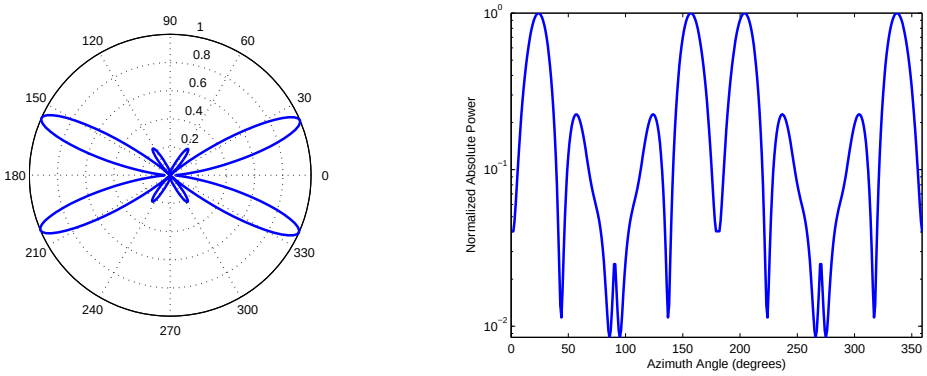

Figure 2.1: Polar beam pattern example (left) and $|\mathbf{a}^H(\theta_i)\mathbf{w}|^2$ (right), $M = 360$, $N = 16$ for angles $\{-20^o, +20^o\}$

2.2 Phase Retrieval

In phase retrieval, the goal is to recover a signal from the magnitude of linear measurements (signal $\mathbf{x}$ from measurements $|\mathbf{Ax}|$ in the noiseless case). A popular approach is to cast the problem as a non-convex quadratic program, to which convex relaxation can be applied. However, as in the case of phase-only beamforming, it is hoped that first-order methods can yield (at least) comparable results at greatly reduced complexity.

Consider the signal model

$$\mathbf{y} = |\mathbf{Ax}| + \mathbf{n}, \tag{2.8}$$

where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, i.e., zero-mean Gaussian with covariance $\sigma^2\mathbf{I}$. We desire to retrieve $\mathbf{x} \in \mathbb{C}^N$ from the noisy amplitude measurements $\mathbf{y} \in \mathbb{C}^M$. We can formulate this as maximum likelihood from incomplete data, as follows:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{C}^N, \mathbf{u}\in\mathbb{C}^M} \quad & \|\mathbf{Ax} - \mathrm{Diag}(\mathbf{y})\mathbf{u}\|_2^2 \\ \text{subject to} \quad & |u_i| = 1, i = 1, \ldots, M. \end{aligned} \tag{2.9}$$

Here Diag($\mathbf{y}$) is a diagonal matrix with $\mathbf{y}$ on its diagonal. Since $\mathbf{x}$ is unconstrained, we can explicitly substitute the least squares solution $\mathbf{x} = \mathbf{A}^{\dagger}\text{Diag}(\mathbf{y})\mathbf{u}$ into (2.9):

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{C}^M} \quad & \|\mathbf{A}\mathbf{A}^{\dagger}\text{Diag}(\mathbf{y})\mathbf{u} - \text{Diag}(\mathbf{y})\mathbf{u}\|_2^2 \\ \text{subject to} \quad & |u_i| = 1, i = 1, \ldots, M. \end{aligned} \tag{2.10}$$

Collecting terms, we have the UQP

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{C}^M} \quad & \mathbf{u}^H\mathbf{R}\mathbf{u} \\ \text{subject to} \quad & |u_i| = 1, i = 1, \ldots, M, \end{aligned} \tag{2.11}$$

with

$$\mathbf{R} := \text{Diag}(\mathbf{y})(\mathbf{A}\mathbf{A}^{\dagger} - \mathbf{I})^H(\mathbf{A}\mathbf{A}^{\dagger} - \mathbf{I})\text{Diag}(\mathbf{y}), \tag{2.12}$$

which is positive semi-definite. This approach to phase retrieval is known as *PhaseCut* [14].

Chapter 3

ULS and UQP Preliminaries

Although the beamformer design problem is expressed as ULS in (2.3), and the phase retrieval problem is expressed as a UQP in (2.11), we will show that any ULS can be recast as a UQP, and vice versa.

Consider the ULS optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{C}^N} \quad & \|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 \\ \text{subject to} \quad & |x_i| = 1, i = 1, \ldots, N. \end{aligned} \tag{3.1}$$

Expanding the squared norm, we obtain

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{C}^N} \quad & \mathbf{x}^H\mathbf{A}^H\mathbf{A}\mathbf{x} - \mathbf{y}^H\mathbf{A}\mathbf{x} - \mathbf{x}^H\mathbf{A}^H\mathbf{y} + \mathbf{y}^H\mathbf{y} \\ \text{subject to} \quad & |x_i| = 1, i = 1, \ldots, N \end{aligned} \tag{3.2}$$

which can be expressed as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{C}^N} \quad & \begin{bmatrix}\mathbf{x}^H & 1\end{bmatrix} \begin{bmatrix}\mathbf{A}^H\mathbf{A} & -\mathbf{A}^H\mathbf{y} \\ -\mathbf{y}^H\mathbf{A} & 0\end{bmatrix} \begin{bmatrix}\mathbf{x} \\ 1\end{bmatrix} \\ \text{subject to} \quad & |x_i| = 1, i = 1, \ldots, N, \end{aligned} \tag{3.3}$$

where the (constant) $\mathbf{y}^H\mathbf{y}$ term has been discarded. Upon defining

$$\mathbf{R} := \begin{bmatrix}\mathbf{A}^H\mathbf{A} & -\mathbf{A}^H\mathbf{y} \\ -\mathbf{y}^H\mathbf{A} & 0\end{bmatrix}, \tag{3.4}$$

we may equivalently rewrite (3.3) as

$$\begin{aligned} \min_{\tilde{\mathbf{x}} \in \mathbb{C}^{N+1}} \quad & \tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}} \\ \text{subject to} \quad & |\tilde{x}_i|^2 = 1, i = 1, \ldots, N;\ \tilde{x}_{N+1} = 1. \end{aligned} \tag{3.5}$$

We may in fact relax the last constraint from $\tilde{x}_{N+1} = 1$ to $|\tilde{x}_{N+1}|^2 = 1$. To see this, consider the UQP

$$\begin{aligned} \min_{\tilde{\mathbf{x}} \in \mathbb{C}^{N+1}} \quad & \tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}} \\ \text{subject to} \quad & |\tilde{x}_i|^2 = 1, i = 1, \ldots, N+1. \end{aligned} \tag{3.6}$$

Notice that $\tilde{\mathbf{x}}$ is feasible for (3.6) if and only if $\tilde{x}^*_{N+1}\tilde{\mathbf{x}}$ (where $\tilde{x}^*_{N+1}$ denotes the complex conjugate of $\tilde{x}_{N+1}$) is feasible for (3.5), at the same cost – since $\tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}}$ is invariant to a common phase shift of all elements of $\tilde{\mathbf{x}}$. Also note that although $\mathbf{R}$ as defined above may not be positive semi-definite, we can easily render the cost function convex via diagonal loading. Let λ_{min} be the minimum eigenvalue of $\mathbf{R}$; since

$$\begin{aligned} \tilde{\mathbf{x}}^H (\mathbf{R} - \lambda_{min}\mathbf{I}) \tilde{\mathbf{x}} &= \tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}} - \lambda_{min} \|\tilde{\mathbf{x}}\|^2 \\ &= \tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}} - \lambda_{min}(N+1), \end{aligned} \tag{3.7}$$

$\mathbf{R}$ can always be made positive semi-definite without changing the optimization problem.

Conversely, to express a UQP as ULS, consider the partitioning

$$\begin{aligned} \tilde{\mathbf{x}}^H \mathbf{R} \tilde{\mathbf{x}} &= \begin{bmatrix} \mathbf{x}^H e^{-j\theta} & e^{-j\theta} \end{bmatrix} \begin{bmatrix} \mathbf{A}^H \mathbf{A} & -\mathbf{A}^H \mathbf{y} \\ -\mathbf{y}^H \mathbf{A} & c \end{bmatrix} \begin{bmatrix} \mathbf{x} e^{j\theta} \\ e^{j\theta} \end{bmatrix} \\ &= \mathbf{x}^H \mathbf{A}^H \mathbf{A} \mathbf{x} - \mathbf{y}^H \mathbf{A} \mathbf{x} - \mathbf{x}^H \mathbf{A}^H \mathbf{y} + c. \end{aligned} \tag{3.8}$$

We can also change c to $\|\mathbf{y}\|^2$ to complete the square in (3.8), since adding a constant to the cost function does not change the solution of (3.6). This can then be recast as the ULS problem

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{C}^N} \quad & \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \\ \text{subject to} \quad & |x_i| = 1, i = 1, \ldots, N. \end{aligned} \tag{3.9}$$

We can therefore transform ULS to UQP and vice-versa.

Chapter 4

Algorithms for ULS and UQP: Prior Art

Consider the UQP problem

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{C}^N} \quad & \mathbf{w}^H\mathbf{R}\mathbf{w} \\ \text{subject to} \quad & |w_i|^2 = 1, i = 1, \ldots, N. \end{aligned} \tag{4.1}$$

As shown in (3.7), $\mathbf{R}$ can always be made positive semi-definite, which ensures that the cost function is convex. However, this problem is still non-convex (and NP-hard for general $\mathbf{R}$) due to the element-wise unit-modulus constraints on $\mathbf{w}$ [15]. Semi-Definite Relaxation (SDR) can be employed to isolate the non-convexity, as follows. Since $\mathbf{w}^H\mathbf{R}\mathbf{w} = \text{Trace}(\mathbf{w}^H\mathbf{R}\mathbf{w}) = \text{Trace}(\mathbf{R}\mathbf{w}\mathbf{w}^H)$, we can define a matrix $\mathbf{W} := \mathbf{w}\mathbf{w}^H$ and equivalently write (4.1) as

$$\begin{aligned} \min_{\mathbf{W}} \quad & \text{Trace}(\mathbf{R}\mathbf{W}) \\ \text{subject to} \quad & \mathbf{W}_{ii} = 1, i = 1, \ldots, N \\ & \mathbf{W} = \mathbf{w}\mathbf{w}^H. \end{aligned} \tag{4.2}$$

The constraint $\mathbf{W} = \mathbf{w}\mathbf{w}^H$ is equivalent to $\mathbf{W} \succeq \mathbf{0}$ (positive semi-definite) *and* $\text{rank}(\mathbf{W}) = 1$. The $\text{rank}(\mathbf{W}) = 1$ constraint is non-convex; dropping it yields a relaxation that is

equivalent to taking the Lagrange bi-dual of above problem, namely

$$\begin{aligned} \min_{\mathbf{W}} \quad & \text{Trace}(\mathbf{RW}) \\ \text{subject to} \quad & \mathbf{W}_{ii} = 1, i = 1, \ldots, N \\ & \mathbf{W} \succeq \mathbf{0}, \end{aligned} \tag{4.3}$$

which is convex. Due to rank relaxation, the optimal $\mathbf{W}_o$ of (4.3) will (in general) not be rank one. Therefore, a concern is how to extract a good feasible solution of (4.1) from $\mathbf{W}_o$. In addition to extracting the principal component of $\mathbf{W}_o$, we generate L candidate random vectors via

$$\boldsymbol{\zeta}_\ell = \mathbf{U}\boldsymbol{\Sigma}^{1/2}\mathbf{v}_\ell, \ell = 1, \ldots, L, \tag{4.4}$$

where each $\mathbf{v}_\ell$ is independently chosen from a circularly symmetric zero-mean complex Gaussian distribution of unit variance, and $\mathbf{W}_o = \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^H$ is the eigendecomposition of $\mathbf{W}_o$ [18]. Then each $\boldsymbol{\zeta}_\ell$ is projected onto the feasible set via

$$\hat{\mathbf{w}}_\ell = e^{j\angle(\boldsymbol{\zeta}_\ell)}, \ell = 1, \ldots, L, \tag{4.5}$$

where $\angle(\cdot)$ denotes the element-wise angle of the argument, and the cost of (4.1) is evaluated at this $\hat{\mathbf{w}}_\ell$. Finally, the $\hat{\mathbf{w}}_\ell$ that yields the smallest cost is chosen as the approximate solution. SDR can be effective under certain conditions and for relatively small problems. Because it lifts the problem dimension and solves a convex problem in $O(N^2)$ variables, it quickly becomes very hard to solve in practice as N grows. Fortunately, in this case we can exploit the special structure of the problem considered to reduce complexity and enable the use of SDR for higher N than would otherwise be possible. Faster implementation is made possible via a row-by-row block coordinate descent algorithm [19, 20]. Consider the following log-barrier implementation of the SDR problem in (4.2):

$$\begin{aligned} \min_{\mathbf{W} \in \mathbb{H}^N} \quad & \text{Trace}(\mathbf{RW}) - \lambda \log\det(\mathbf{W}) \\ \text{subject to} \quad & \mathbf{W}_{ii} = 1, i = 1, \ldots, N, \end{aligned} \tag{4.6}$$

where $\mathbb{H}^N$ is the set of $N \times N$ Hermitian matrices, $\lambda > 0$, and $\log\det(\mathbf{W})$ is defined as $-\infty$ when $\mathbf{W}$ is not positive definite. By letting $\overline{\mathbf{w}}_i$ denote the i-th row (or column) of

$\mathbf{W}$ with the i-th element removed (and analogously $\bar{\mathbf{r}}_i$ for matrix $\mathbf{R}$), and $\overline{\mathbf{W}}_{ii}$ denote the sub-matrix of $\mathbf{W}$ after removing the i-th row and column, it can be shown [19] that the minimization in (4.6) with respect to $\overline{\mathbf{w}}_i$ can be performed by solving

$$\min_{\overline{\mathbf{w}}_i \in \mathbb{C}^N} \quad \bar{\mathbf{r}}_i^H \overline{\mathbf{w}}_i - \lambda \log\left(1 - \overline{\mathbf{w}}_i^H \overline{\mathbf{W}}_{ii} \overline{\mathbf{w}}_i\right) \tag{4.7}$$

Note that the update for $\overline{\mathbf{w}}_i$ leaves $\mathbf{W}_{ii} = 1$ undisturbed, and that the log barrier term follows via the Schur complement of $\mathbf{W}$. The optimal $\overline{\mathbf{w}}_i$ admits the closed-form solution

$$\overline{\mathbf{w}}_i = \begin{cases} -\frac{1}{2\gamma}(\sqrt{\lambda^2 + 4\gamma})\overline{\mathbf{W}}_{ii}\overline{\mathbf{r}_i}, & \gamma > 0 \\ \mathbf{0}, & \text{otherwise,} \end{cases} \tag{4.8}$$

where $\gamma = \bar{\mathbf{r}}_i^H \overline{\mathbf{W}}_{ii} \bar{\mathbf{r}}_i$. Each row (or column) of $\mathbf{W}$ is then cyclically updated, until convergence. This row-by-row block coordinate descent algorithm is also used for phase retrieval using *PhaseCut* [14], a variant of the celebrated *MaxCut* approximation algorithm for two-way graph partitioning, where the elements of the optimization vector variable are constrained to lie in $\{-1, 1\}$.

Chapter 5

Proposed Algorithms

5.1 Baseline

For general $\mathbf{A}$, the NP-hard ULS optimization problem

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{C}^N} \quad & \|\mathbf{y}-\mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1, \ldots, N, \end{aligned} \tag{5.1}$$

can be approximated using SDR, or, with an explicit $w_i = e^{j\theta_i}$ parametrization of the unit modulus constraint, using unconstrained derivative-based methods, such as gradient descent. SDR is computationally expensive, even if one uses row-by-row block coordinate descent. Gradient descent using the explicit constraint parametrization does not seem to work well for this (nonconvex and NP-hard) problem. Different from earlier attempts, we propose keeping the unit modulus constraint and using *projected* gradient descent instead of unconstrained gradient descent. The details are provided in Algorithm 1.

Algorithm 1 Projected Gradient Descent

1: Initialization: Set $k = 0$, $\alpha = \frac{1}{\lambda_{max}(\mathbf{A}^H\mathbf{A})}$, $\mathbf{w}_0 = e^{j\angle(\mathbf{A}^\dagger\mathbf{y})}$
2: **Repeat**
3: $\quad\boldsymbol{\zeta}_{k+1} = \mathbf{w}_k + \alpha\mathbf{A}^H(\mathbf{y} - \mathbf{A}\mathbf{w}_k)$
4: $\quad\mathbf{w}_{k+1} = e^{j\angle(\boldsymbol{\zeta}_{k+1})}$
5: $\quad k = k + 1$
6: **until convergence**

For stopping criterion, we propose using[1]

$$\frac{\|\mathbf{w}_{k+1} - \mathbf{w}_k\|_2}{\sqrt{N}} < \epsilon. \tag{5.2}$$

At this point, the reader may rightfully question whether Algorithm 1 is guaranteed to converge, given that the constraint set that we project onto in step 4 is highly non-convex. The answer is affirmative, but non-trivial.

5.1.1 Convergence and choice of step size α

First consider the cost function

$$f(\mathbf{w}) = \|\mathbf{y} - \mathbf{A}\mathbf{w}\|_2^2, \tag{5.3}$$

where $\mathbf{w}$ is unconstrained. It is known that for plain gradient descent, with $\nabla f(\mathbf{w}) = -\mathbf{A}^H(\mathbf{y} - \mathbf{A}\mathbf{w})$, the iterations in the descent algorithm will converge to a Karush-Kuhn-Tucker (KKT) point if $0 < \alpha < \frac{1}{L}$, where $L = \lambda_{max}(\mathbf{A}^H\mathbf{A})$ is the Lipschitz constant for $\nabla f(\mathbf{w})$ [21]. From the definition

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2 \tag{5.4}$$

if follows that

$$\|\mathbf{A}^H\mathbf{A}(\mathbf{x} - \mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2. \tag{5.5}$$

Due to the submultiplicity property of the 2-norm,

$$\|\mathbf{A}^H\mathbf{A}(\mathbf{x} - \mathbf{y})\|_2 \leq \|\mathbf{A}^H\mathbf{A}\|_2\|\mathbf{x} - \mathbf{y}\|_2, \tag{5.6}$$

[1] For our numerical results, $\epsilon = 10^{-6}$ was used for beamforming simulations, and $\epsilon = 10^{-4}$ for simulations pertaining to phase retrieval.

and the property that the spectral norm of $\mathbf{A}^H\mathbf{A}$ is defined to be the maximum eigenvalue of $\mathbf{A}^H\mathbf{A}$, we conclude that $L = \lambda_{max}(\mathbf{A}^H\mathbf{A})$, and thus α is upper bounded by $1/\lambda_{max}(\mathbf{A}^H\mathbf{A})$.

We wish, however, to minimize such a cost function with a unit modulus constraint on $\mathbf{w}$, and Algorithm 1 enforces this constraint via projection onto the unit-modulus torus at each iteration. To the best of our knowledge, no convergence results have been reported to date for projected gradient descent with projections onto a nonconvex constraint set.

In Appendix A, it is proven that for $\alpha \leq 1/\lambda_{max}(\mathbf{A}^H\mathbf{A})$, Algorithm 1 converges to a KKT point of the nonconvex and NP-hard problem (5.1). This is a key contribution of this work (as submitted in [22]) – the first result of this kind, to the best of our knowledge.

5.2 Auto-scaling formulation

For the beamforming scenario, recall the formulation

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{C}^N, s\in\mathbb{C}} \quad & \|\mathbf{y} - s\mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1, \ldots, N. \end{aligned} \tag{5.7}$$

Using (2.7), we can tune s in each iteration, leading to Algorithm 2:

Algorithm 2 Projected Gradient Descent $(\mathbf{w}, s)$

1: Initialization: Set $k = 0$, $\mathbf{w}_0 = e^{j\angle(\mathbf{A}^\dagger\mathbf{y})}$
2: **Repeat**
3: $\quad s_{k+1} = \mathbf{w}_k^H\mathbf{A}^H\mathbf{y}/\|\mathbf{A}\mathbf{w}_k\|_2^2$
4: $\quad \alpha_{k+1} = 1/\lambda_{max}(|s_{k+1}|^2\mathbf{A}^H\mathbf{A})$
5: $\quad \boldsymbol{\zeta}_{k+1} = \mathbf{w}_k + \alpha_{k+1}s_{k+1}^*\mathbf{A}^H(\mathbf{y} - s_{k+1}\mathbf{A}\mathbf{w}_k)$
6: $\quad \mathbf{w}_{k+1} = e^{j\angle(\boldsymbol{\zeta}_{k+1})}$
7: $\quad k = k + 1$
8: **until convergence**

Remark 1 *It is also interesting to investigate convergence properties of Algorithm 2. This algorithm can be considered as an* inexact *alternating optimization approach; i.e., we alternate between solving subproblems with respect to s and $\mathbf{w}$, while fixing the other. During the updates, the subproblem with respect to s is optimally solved, while the subproblem with respect to $\mathbf{w}$ is not solved to optimality at each iteration – we only update $\mathbf{w}$ using a single iteration of gradient projection, for efficiency. Fortunately, this type of two-block inexact alternating optimization can be shown to converge to a KKT point, following the insights of [23] and [24]. The proof closely follows the recent work in [25], and thus the details are skipped for conciseness.*

5.3 Additional degrees of freedom in transmit beamforming

At this point it is worth highlighting an important difference between transmit and receive beamforming. In the receive beamforming scenario, we may be required to control the phase response as a function of $\boldsymbol{\theta}$, e.g., for phase coherence or constructive combining of specular multipath components. In transmit (including multicast) beamforming, however, it is sufficient to specify a desired magnitude for each direction or general channel vector of interest, as the receiver will have to perform any necessary phase estimation/correction anyway, due to local oscillator phase mismatch. We can mathematically model this situation by considering the modified optimization problem

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{C}^N,\boldsymbol{\theta}\in\mathbb{R}^M,s\in\mathbb{C}} \quad & \|\mathbf{y} \circledast e^{j\boldsymbol{\theta}} - s\mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1,\ldots,N, \end{aligned} \tag{5.8}$$

where $\circledast$ denotes the Hadamard (element-wise) product, and $\boldsymbol{\theta}$ represents the additional degrees of (phase response) freedom. Let $\mathbf{u} = e^{j\boldsymbol{\theta}}$; we can equivalently express (5.8) as

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{C}^N,\mathbf{u}\in\mathbb{C}^M,s\in\mathbb{C}} \quad & \|\mathbf{Y}\mathbf{u} - s\mathbf{A}\mathbf{w}\|_2^2 \\ \text{subject to} \quad & |w_i| = 1, i = 1,\ldots,N, \\ & |u_i| = 1, i = 1,\ldots,M, \end{aligned} \tag{5.9}$$

where $\mathbf{Y} = \text{Diag}(\mathbf{y})$. Performing alternating optimization on $\mathbf{w}$, $\mathbf{u}$ and s (projecting $\mathbf{w}$ and $\mathbf{u}$ onto the unit-modulus space after each iteration), we obtain the following algorithm:

Algorithm 3 Projected Gradient Descent ($\mathbf{w}$, $\mathbf{u}$, s)

1: Initialization: Set $k = 0$, obtain initial $\mathbf{w}_0$ from Algorithm 2, initialize $\mathbf{u}_0 = \boldsymbol{\xi}_0 = \mathbf{1}$, $\beta = 1/\lambda_{max}(\mathbf{Y}^H\mathbf{Y})$
2: Let $\mathcal{J} = \{i : y_i \neq 0\}$, $\mathbf{Y} = \text{Diag}(\mathbf{y})$, $\tilde{\mathbf{Y}} = \text{Diag}(\mathbf{y}(\mathcal{J}))$
3: **Repeat**
4: $s_{k+1} = \mathbf{w}_k^H \mathbf{A}^H \mathbf{Y}\mathbf{u}_k / \|\mathbf{A}\mathbf{w}_k\|_2^2$
5: $\alpha_{k+1} = 1/\lambda_{max}(|s_{k+1}|^2 \mathbf{A}^H\mathbf{A})$
6: $\boldsymbol{\zeta}_{k+1} = \mathbf{w}_k + \alpha_{k+1} s^* \mathbf{A}^H(\mathbf{Y}\mathbf{u}_k - s\mathbf{A}\mathbf{w}_k)$
7: $\mathbf{w}_{k+1} = e^{j\angle(\boldsymbol{\zeta}_{k+1})}$
8: $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k$
9: $\boldsymbol{\xi}_{k+1}(\mathcal{J}) = \mathbf{u}_k(\mathcal{J}) - \beta\tilde{\mathbf{Y}}^H(\tilde{\mathbf{Y}}\mathbf{u}_k(\mathcal{J}) - s\mathbf{A}(\mathcal{J},:)\mathbf{w}_{k+1})$
10: $\mathbf{u}_{k+1} = e^{j\angle(\boldsymbol{\xi}_{k+1})}$
11: $k = k + 1$
12: **until convergence**

Note that for the $\mathbf{u}$ update, only the elements corresponding to the non-zero values of $\mathbf{y}$ are of interest. As such, the set $\mathcal{J}$ has been defined as the set of indices where $y_i \neq 0$ and $\mathbf{A}(\mathcal{J},:)$ denotes a matrix comprised of the rows of $\mathbf{A}$ corresponding to the indices in $\mathcal{J}$. Thus, only the elements of $\mathbf{u}(\mathcal{J})$ are updated (steps 6 and 7 of the algorithm).

One may also consider the conceptually simpler formulation

$$\begin{aligned} \min_{\mathbf{z}\in\mathbb{C}^{M+N}} \quad & \|\mathbf{B}\mathbf{z}\|_2^2 \\ \text{subject to} \quad & |z_i| = 1, i = 1, \ldots, M+N, \end{aligned} \tag{5.10}$$

where

$$\mathbf{B} := \left[\begin{array}{c:c} \mathbf{Y} & -s\mathbf{A} \end{array}\right], \quad \mathbf{z} := \left[\begin{array}{c} \mathbf{u} \\ \hdashline \mathbf{w} \end{array}\right]. \tag{5.11}$$

Algorithm 1 can be directly applied to this formulation. However, the step size for the Projected Gradient Descent in (5.10) is a function of the largest eigenvalue of $\mathbf{B}^H\mathbf{B}$; depending on the scaling of $\mathbf{Y}$ and $\mathbf{A}$, the freedom that Algorithm 3 provides (with respect to employing different step sizes for the $\mathbf{w}$ and $\mathbf{u}$ updates) yields faster convergence, as demonstrated in Section 6.2.4.

5.4 Accelerated Projected Gradient Descent

(Projected) gradient descent has low per-iteration complexity, but there exist faster first-order methods (in terms of the number of iterations required until convergence) of comparable per-iteration complexity. Again, let us first consider the cost function in (5.3) with no constraint on $\mathbf{w}$. Applying Nesterov's Accelerated Gradient Descent algorithm [26] to (5.1), the resulting algorithm is defined by the following updates (with $\mathbf{x}_1 = \mathbf{z}_1$):

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{x}_k - \frac{1}{L}\nabla f(\mathbf{x}_k) \\ &= \mathbf{x}_k \ - \frac{1}{L}\mathbf{A}^H(\mathbf{y} - \mathbf{A}\mathbf{x}_k) \\ \mathbf{x}_{k+1} &= \gamma_k \mathbf{z}_{k+1} + (1-\gamma_k)\mathbf{z}_k \end{aligned} \tag{5.12}$$

with

$$\lambda_0 = 0, \quad \lambda_k = \frac{1+\sqrt{1+4\lambda_{k-1}^2}}{2}, \quad \gamma_k = \frac{\lambda_k - 1}{\lambda_{k+1}}. \tag{5.13}$$

The update for $\mathbf{z}_{k+1}$ is effectively identical to a simple gradient step; the $\mathbf{x}_{k+1}$ update then allows the momentum from the previous iteration to descend further in the direction of $\mathbf{z}_k$. The derivation of the sequences defined in (5.13) can be found in [27].

The algorithm is easily adaptable to our problems by the simple addition of a projection onto the unit-modulus constraint at each iteration. Note that as $\gamma_k \to 1$ as $k \to \infty$, the update for $\mathbf{x}_{k+1}$ effectively becomes a simple gradient update; as such, we can make similar (asymptotic) convergence claims as stated in Appendix A. The algorithm does indeed provide performance gains in terms of runtime complexity, as we will illustrate in Chapter 6.

Algorithm 4 Accelerated Projected Gradient Descent for (5.1)

Initialization: Set $k = 0$, $\lambda_0 = 0$, $\alpha = \frac{1}{\lambda_{max}(\mathbf{A}^H\mathbf{A})}$, $\mathbf{w}_0 = e^{j\angle(\mathbf{A}^\dagger\mathbf{y})}$
Repeat
  $\lambda_{k+1} = \frac{1+\sqrt{1+4\lambda_k^2}}{2}$
  $\gamma_k = \frac{\lambda_k - 1}{\lambda_{k+1}}$
  $\mathbf{z}_{k+1} = \mathbf{w}_k + \alpha\mathbf{A}^H(\mathbf{y} - \mathbf{A}\mathbf{w}_k)$
  $\boldsymbol{\zeta}_{k+1} = \gamma_k\mathbf{z}_{k+1} + (1-\gamma_k)\mathbf{z}_k$
  $\mathbf{w}_{k+1} = e^{j\angle(\boldsymbol{\zeta}_{k+1})}$
  $k = k + 1$
until convergence

Algorithm 5 Accelerated Projected Gradient Descent for (5.9)

Initialization: Set $k = 0$, $\lambda_0 = 0$, obtain initial $\mathbf{w}_0$ from Algorithm 2, initialize $\mathbf{u}_0 = \boldsymbol{\xi}_0 = \mathbf{1}$, $\alpha_1 = 1/\lambda_{max}(|s|^2\mathbf{A}^H\mathbf{A})$, $\beta = 1/\lambda_{max}(\mathbf{Y}^H\mathbf{Y})$
Let $\mathcal{J} = \{i : y_i \neq 0\}$, $\mathbf{Y} = \mathrm{Diag}(\mathbf{y})$, $\tilde{\mathbf{Y}} = \mathrm{Diag}(\mathbf{y}(\mathcal{J}))$
Repeat
  $s_{k+1} = \mathbf{w}_k^H\mathbf{A}^H\mathbf{Y}\mathbf{u}_k / \|\mathbf{A}\mathbf{w}_k\|_2^2$
  $\alpha_{k+1} = 1/\lambda_{max}(|s_{k+1}|^2\mathbf{A}^H\mathbf{A})$
  $\lambda_{k+1} = \frac{1+\sqrt{1+4\lambda_k^2}}{2}$
  $\gamma_k = \frac{\lambda_k - 1}{\lambda_{k+1}}$
  $s_{k+1} = \mathbf{w}_k^H\mathbf{A}^H\mathbf{Y}\mathbf{u}_k / \|\mathbf{A}\mathbf{w}_k\|^2$
  $\mathbf{z}_{k+1} = \mathbf{w}_k + \alpha_{k+1}s^*\mathbf{A}^H(\mathbf{Y}\mathbf{u}_k - s\mathbf{A}\mathbf{w}_k)$
  $\boldsymbol{\zeta}_{k+1} = \gamma_k\mathbf{z}_{k+1} + (1-\gamma_k)\mathbf{z}_k$
  $\mathbf{w}_{k+1} = e^{j\angle(\boldsymbol{\zeta}_{k+1})}$
  $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k$
  $\boldsymbol{\xi}_{k+1}(\mathcal{J}) = \mathbf{u}_k(\mathcal{J}) - \beta\tilde{\mathbf{Y}}^H(\tilde{\mathbf{Y}}\mathbf{u}_k(\mathcal{J}) - s\mathbf{A}(\mathcal{J},:)\mathbf{w}_{k+1})$
  $\mathbf{u}_{k+1} = e^{j\angle(\boldsymbol{\xi}_{k+1})}$
  $k = k + 1$
until convergence

Chapter 6

Simulations

6.1 Baseline ULS Scenario

We first consider the general ULS problem as shown in (2.3), comparing the mean squared error and runtime performance for both FastSDR (for the UQP form of the problem) and our proposed Projected Gradient Descent method as outlined in Algorithm (1). We will position this as an estimation problem, employing the signal model $\mathbf{y} = \mathbf{A}\mathbf{w} + \mathbf{n}$; $\mathbf{A} \in \mathbb{C}^{M\times N}$ and $\mathbf{n} \in \mathbb{C}^M$ are circularly symmetric Gaussian and unit variance, and $\mathbf{w} \in \mathbb{C}^N$ is drawn from $\mathcal{N}_{\mathbb{C}}(0, \mathbf{I})$, then projected onto the unit-modulus torus. We set $M = 144$ and SNR = 10dB, and observe the MSE of the estimated $\hat{\mathbf{w}}$ as our performance measure. 100 independent Monte Carlo trials are performed for each $N = 2, \ldots, 200$.

Figure 6.1 demonstrates that Algorithm 1 outperforms FastSDR with regards to MSE, and very nearly attains the Cramér-Rao Bound. Furthermore, PGD is more than 10 times faster than FastSDR for all N considered. Note that FastSDR is known to be a surprisingly fast algorithm for approximating UQP, and our results indicates an even more promising picture for PGD. Although this was performed on the simple baseline form of the problem, these simulations motivate additional exploration of the comparative merits of PGD.

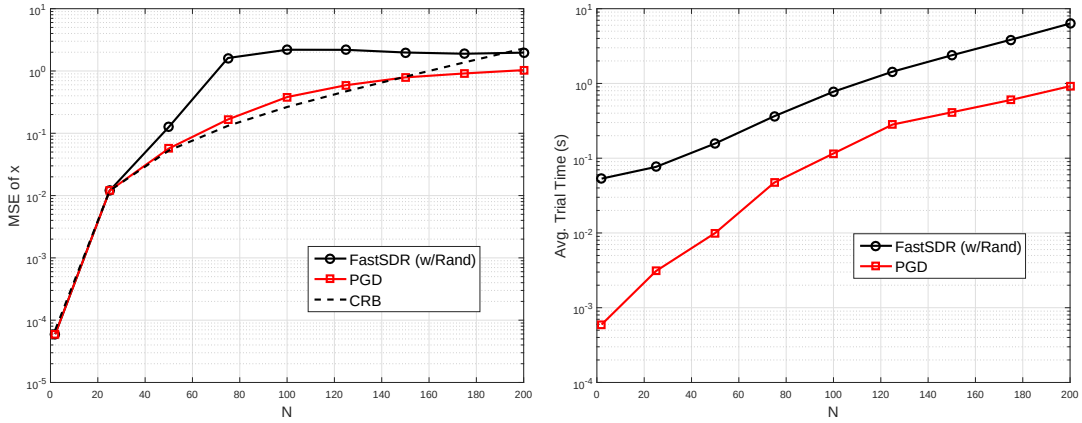

Figure 6.1: MSE (left) and runtime (right) comparison of Algorithm 1 and FastSDR for $N = 2, \ldots, 200$, SNR= 10dB.

6.2 Phase-Only Beamforming Scenario

The performance of the Row-by-Row Block Coordinate Descent SDR approximation algorithm with Gaussian randomization, as well as the Projected Gradient Descent algorithms in (5.7) and (5.9) (with and without acceleration) were evaluated for $N = 2, \ldots, 200$ with the angle space discretized into 36 and 144 regions (resulting in $\mathbf{A}$ of dimension $36 \times N$ and $144 \times N$, respectively). The ULA scenario is considered, with rows of $\mathbf{A}$ admitting the Vandermonde structure $[1, e^{j\theta_i}, e^{j2\theta_i}, \ldots, e^{j(N-1)\theta_i}]$ for each row i in $\mathbf{A}$. 100 independent Monte Carlo trials were performed for each N, and a $K = 2$ receiver multicast beamforming scenario was considered[1] . The two user angles were randomly drawn in each problem instance, with each $\mathbf{y}$ constructed in accordance with (2.4). Recall that any ULS can be expressed as a UQP, and vice versa; FastSDR was employed using a UQP reformulation of (5.7), and all Projected Gradient Descent algorithms were performed directly on their ULS representation. Below is the key for the simulated algorithms:

- FastSDR (w/Rand): Row-by-Row Block Coordinate Descent SDR approximation with Gaussian randomization (1000 randomization trials, each projected onto the

[1] Max-min multicast beamforming under a sum power constraint can in fact be optimally and efficiently solved in the Vandermonde case for any K [28], but note that the two problems differ in the cost function and in the constraints, as [28] employs a sum power as opposed to per-antenna constraints.

unit-modulus torus, principal component included).

- PGD: Projected Gradient Descent for Algorithm 2 (formulation in (5.7)).
- PGD (accel): Nesterov's Accelerated Projected Gradient Descent applied to PGD.
- PGD-Alt: Projected Gradient Descent for Algorithm 3 (formulation in (5.9)).
- PGD-Alt (accel): Nesterov's Accelerated Projected Gradient Descent applied to PGD-Alt.

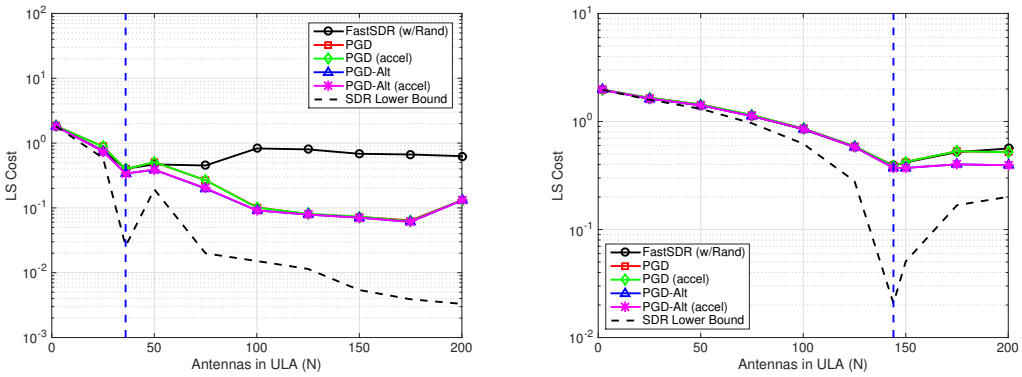

Figure 6.2: Least Squares cost for Vandermonde $\mathbf{A} \in \mathbb{C}^{36\times N}$ (left) and $\mathbf{A} \in \mathbb{C}^{144\times N}$ (right), $N = 2, \ldots, 200$.

As shown in Fig. 6.2, the proposed PGD algorithms perform on par or even outperform FastSDR in terms of least squares cost. PGD outperforms FastSDR for small M and large N. Note that as N increases, additional degrees of freedom are introduced. In Fig. 6.2, a dashed vertical line is drawn at $N = M$, where if one ignores the unit modulus constraints the relaxed linear system transitions from over- to under-determined.

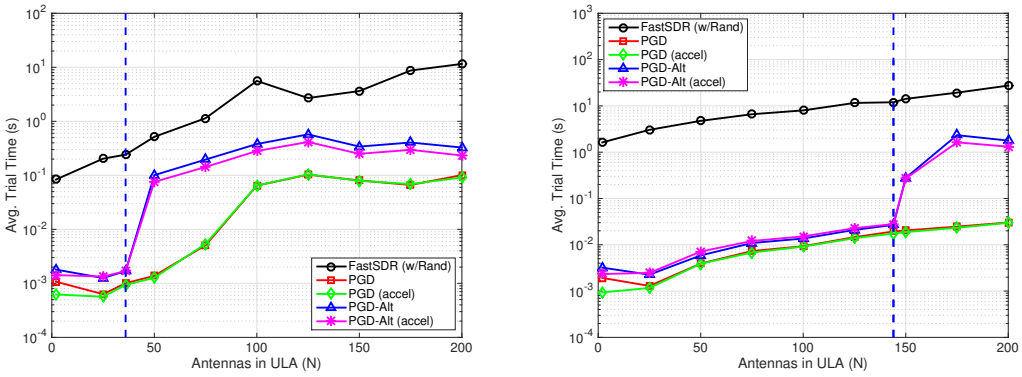

Figure 6.3: Trial runtime comparison for Vandermonde $\mathbf{A} \in \mathbb{C}^{36\times N}$ (left) and $\mathbf{A} \in \mathbb{C}^{144\times N}$ (right), $N = 2, \ldots, 200$

The companion plots illustrating runtime complexity are shown in Fig. 6.3; in terms of the least squares cost/complexity trade-off, all projected gradient algorithms clearly outperform FastSDR. Although FastSDR performs comparably (in terms of cost) to first-order methods when $M \geq N$, the performance gap widens as the number of antennas increases. Further, the accelerated Projected Gradient algorithms perform similarly to their unaccelerated counterparts in terms of cost, and we see modest gains in terms of runtime performance (1.1-1.5 times faster).

As expected, the PGD-Alt algorithms perform worse in terms of runtime (particularly as the number of antennas increases). However, even though we are only introducing $K = 2$ additional degrees of freedom (since the additional freedom to conveniently set the phase response can only be exercised at angles where the target magnitude response is nonzero), we observe a noticeable difference in least squares cost performance between the two Projected Gradient Descent algorithms. To test a case with more degrees of freedom, we next consider a *sector beamforming* scenario with many more non-zero entries in $\mathbf{y}$.

6.2.1 Sector Beamforming

Consider the scenario where rather than attempting to design a beamformer that confines the energy to selected discrete directions, a symmetric wedge spanning $[-40^o, 40^o]$ and its reflection is chosen as the desired magnitude response. For $M = 144$, this is

equivalent to $\mathcal{J} = \{1 \dots 18, 55 \dots 90, 127 \dots 144\}$, with $y_i = 1$ for all $i \in \mathcal{J}$.

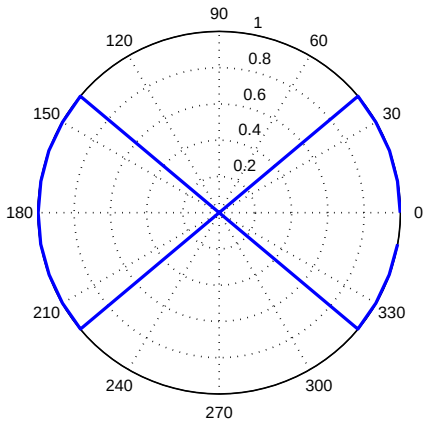

Figure 6.4: Sector beamforming illustration, $M = 144$, $\mathcal{J} = \{1 \dots 18, 55 \dots 90, 127 \dots 144\}$

Figure 6.4 is a polar plot of the resulting transmit beampattern. In this scenario, we again have $N+K$ degrees of freedom in (5.9), but now $K >> 2$; as such, it is reasonable to expect improved performance relative to (5.7) compared to the previously considered 'pencil beam' scenario. We observe a substantial improvement for PGD-Alt, as shown in Figure 6.5.

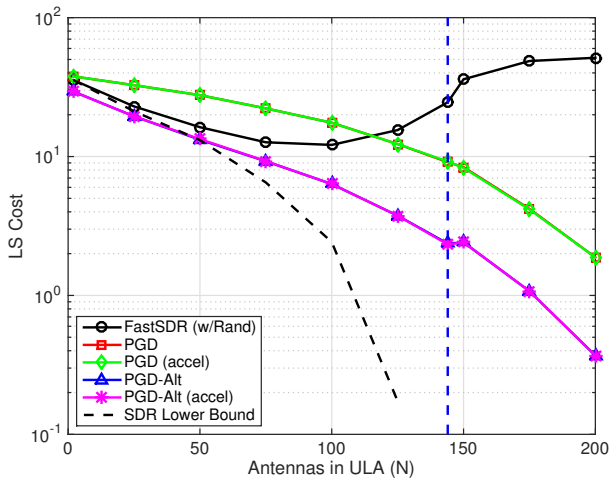

Figure 6.5: Least squares cost comparison for sector case, $M = 144$, $N = 2, \ldots, 200$

To summarize, the alternating optimization algorithm for the formulation in (5.9) provides performance gains (in terms of least squares cost), even with the introduction of very few additional degrees of freedom (K). Further, these improvements become increasingly pronounced as K increases (as in the sector beamforming scenario).

6.2.2 Gaussian case

Although the beamforming examples discussed so far employed a ULA (Vandermonde steering vectors), it is natural to consider the case where $\mathbf{A}$ is complex Gaussian, modeling a Rayleigh fading scenario. The simulations shown in Figure 6.2 were repeated for $\mathbf{A}$ which is circularly symmetric Gaussian and unit variance; as shown in Figure 6.6, all Projected Gradient Descent algorithms outperform FastSDR in terms of least squares cost. Again, accelerated methods demonstrate modestly improved average trial times with respect to their plain Projected Gradient Descent counterparts.

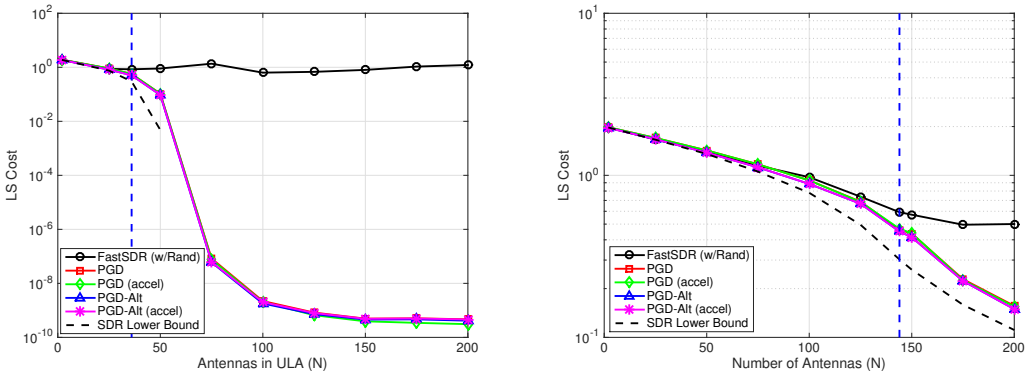

Figure 6.6: Least Squares cost comparison for Gaussian $\mathbf{A} \in \mathbb{C}^{36\times N}$ (left) and $\mathbf{A} \in \mathbb{C}^{144\times N}$ (right), $N = 2, \ldots, 200$

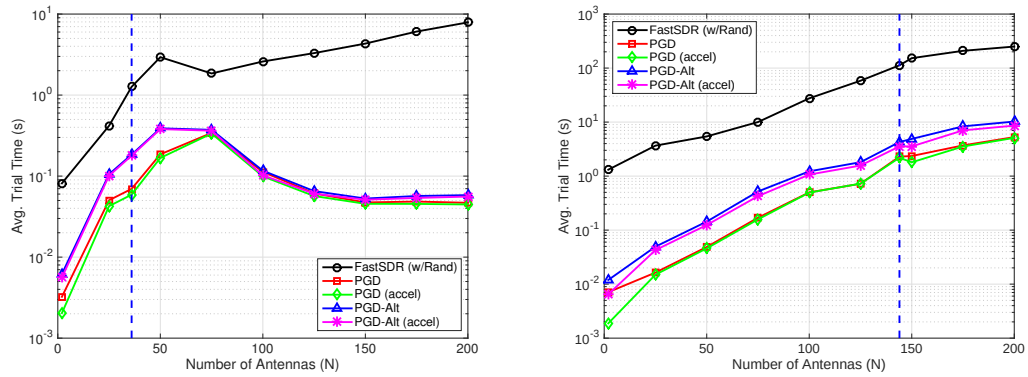

Figure 6.7: Trial runtime comparison for Gaussian $\mathbf{A} \in \mathbb{C}^{36\times N}$ (left) and $\mathbf{A} \in \mathbb{C}^{144\times N}$ (right), $N = 2, \ldots, 200$

6.2.3 Mean Squared Error Comparision to Cramér-Rao Bound

Although the beamforming design scenario is cast as an optimization problem, it is interesting to consider it as an estimation problem arising from a generative signal model. SDR provides a generally unattainable (optimistic) lower bound on the least squares cost for *each instance* of this NP-hard problem, and we can take the average of these lower bounds as a bound on the average least squares cost. This, however, tells us nothing about how far the design variables are from the optimal ones. An

alternative to using the SDR lower bound as a gauge is to *think* of our design problem as arising from maximum likelihood (ML) estimation for the generative signal model $\mathbf{y} = \mathbf{A}\mathbf{w}_o + \mathbf{n}$, where $\mathbf{A}$ is given, $\mathbf{w}_o$ is known to have unit-modulus elements but is otherwise unknown, and $\mathbf{n}$ is circularly symmetric i.i.d. Gaussian. ML estimation for this model boils down to ULS, and the associated CRB provides a lower bound on the variance of unbiased estimators of $\mathbf{w}_o$. Under certain conditions ($M >> N$, appropriate signal to noise ratio), ML approaches the CRB, which makes the latter predictive of ML performance. This way, the CRB can serve as a benchmark on the average attainable distance of the design variables from their optimal settings for our design problem. Of course, this bound will only be valid if we generate design problem instances from the given generative signal model, i.e., if we indeed draw desired response patterns from $\mathbf{y} = \mathbf{A}\mathbf{w}_o + \mathbf{n}$. Note that any $\mathbf{y}$ can be written in this way if we allow for low-enough signal to noise ratio in the generative model, but then the CRB will not be as predictive of the average attainable performance – albeit still a lower bound. This gives us an alternative way to explore algorithm performance for our NP-hard design problem.

For our generative model above, the CRB with respect to the angles in $\mathbf{w}$ can be shown to be (see Appendix B for derivation)

$$CRB = \frac{\sigma^2}{2}\left[\mathrm{Re}\left\{\mathrm{diag}(\mathbf{w})^H \mathbf{A}^H \mathbf{A}\mathrm{diag}(\mathbf{w})\right\}\right]^{-1}. \tag{6.1}$$

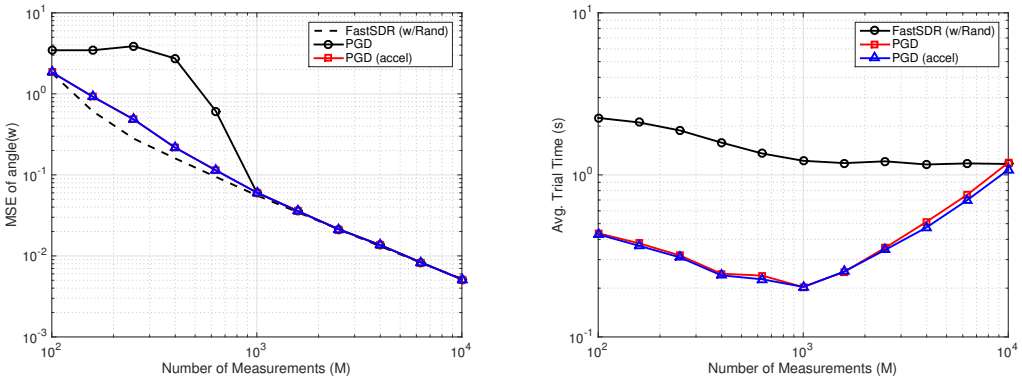

Figure 6.8: CRB/MSE (left) and runtime (right) comparison for $M = 10^2, \ldots, 10^4$, SNR = 10 dB

For each problem instance, since $\mathbf{A} \in \mathbb{C}^{M \times N}$ was drawn from an i.i.d. circularly-symmetric Gaussian distribution, with each $\mathbf{w}_o$ drawn from $\mathcal{N}_{\mathbb{C}}(0, \mathbf{I})$, then projected onto the unit-modulus torus. Each $\mathbf{n}$ was chosen from $\mathcal{N}(0, \sigma^2\mathbf{I})$; the noise variance was obtained for the desired SNR via the relationship

$$SNR = 10 \log_{10} \frac{\mathbb{E}\left[\text{trace}(\mathbf{A}^H \mathbf{A} \mathbf{w}_o \mathbf{w}_o^H)\right]}{M\sigma^2}. \tag{6.2}$$

Since both $\mathbf{A}$ and $\mathbf{w}_o$ are independently drawn, and functions of independent random variables are uncorrelated, we have

$$SNR = 10 \log_{10} \frac{\text{trace}(\mathbb{E}\left[\mathbf{A}^H \mathbf{A}\right] \mathbb{E}\left[\mathbf{w}_o \mathbf{w}_o^H\right])}{M\sigma^2}. \tag{6.3}$$

Note that $\mathbb{E}\left[\mathbf{w}_o \mathbf{w}_o^H\right] = \mathbf{I}$; we can then express (6.3) as

$$\begin{aligned} SNR &= 10 \log_{10} \frac{\mathbb{E}\left[\text{trace}(\mathbf{A}^H \mathbf{A})\right]}{M\sigma^2} \\ &= 10 \log_{10} \frac{\|\mathbf{A}\|_F^2}{M\sigma^2} = \frac{MN}{M\sigma^2} = \frac{N}{\sigma^2}. \end{aligned} \tag{6.4}$$

All simulations were averaged over 100 Monte Carlo trials, with SNR = 10 dB. The mean squared error (MSE) of PGD and FastSDR along with the CRB are plotted in Fig. 6.8; all algorithms effectively attain the CRB with sufficient number of measurements. Interestingly, the PGD algorithms perform much better than FastSDR for smaller M; this corroborates what was demonstrated in earlier simulations, as illustrated in Fig. 6.6 (note the divergence of the FastSDR cost as $\mathbf{A}$ becomes progressively under-determined). Further, Fig. 6.8 (right) demonstrates the comparative advantages of PGD vs. FastSDR in terms of runtime; first-order methods again perform much better, but the performance gap narrows as M increases after a certain point.

6.2.4 Two-block alternating vs. all-at-once projected gradient updates

In Chapter 4, the relative merits of a 'block' formulation (5.10) and the alternating formulation in (5.9) were considered. The argument was made that since we can selectively update the non-zero elements in $\mathbf{u}$ (illustrated in Algorithm 3), as well as considering the freedom provided via different step sizes for the $\mathbf{w}$ and $\mathbf{u}$ updates, we can expect the alternating formulation to yield faster convergence than its block counterpart. We verified that this is the case in simulations. Although both formulations

yield effectively identical least squares cost performance, the alternating formulation provides much faster convergence, as shown in Fig. 6.9.

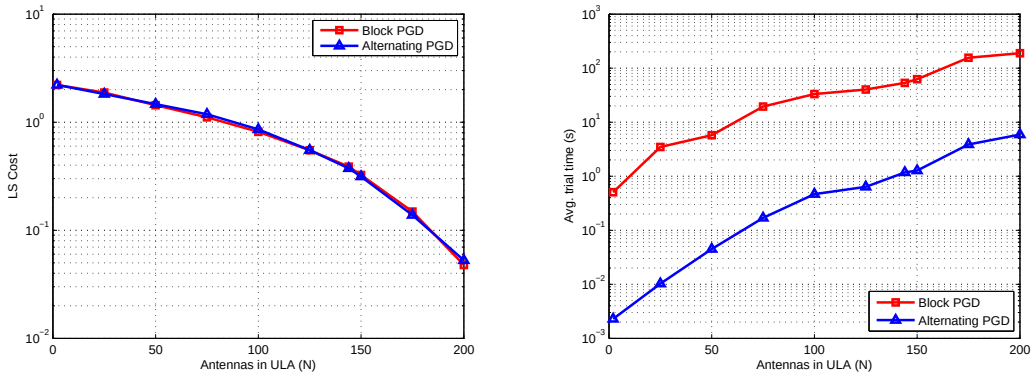

Figure 6.9: Least squares cost (left) and runtime (right) for Alternating (5.9) vs. Block (5.10) Projected Gradient Decent, $M = 144$, $N = 2, \ldots, 200$

6.2.5 Phase discretization

In certain cases discrete phase shifters are employed, which only provide a finite set of phase settings for each antenna. The phase error introduced by quantization can give rise to undesirable side lobes, but finer resolution may not be practical due to hardware availability or cost considerations.

The ULA scenario was again simulated with the Projected Gradient Descent algorithm for formulation (5.1). The obtained continuous result was then quantized for 1-6 bit resolution. Let us define the metric Least Squares Quantization Ratio (LSQR) as

$$\text{LSQR} := \frac{\|\mathbf{y} - \mathbf{A}\mathbf{w}_b\|^2}{\|\mathbf{y} - \mathbf{A}\mathbf{w}\|^2}, \tag{6.5}$$

where $\mathbf{w}$ signifies the beamforming vector obtained from Algorithm 1, and $\mathbf{w}_b$ denotes the b-bit quantization of $\mathbf{w}$. As Table 6.1 demonstrates, 5- and 6-bit quantization results in near-optimal (continuous-equivalent) performance.

Table 6.1: Least Squares Quantization Ratio for 1-6 bits

# of quantization bits (b)	LSQR
1	7.1898
2	3.6838
3	2.2179
4	1.4387
5	1.1470
6	1.0416

6.3 Phase Retrieval Scenario

PhaseCut (SDR), Projected Gradient Descent with fixed step size and Accelerated Projected Gradient Descent were compared for the phase retrieval problem in (2.11); all realizations of $\mathbf{y}$ were generated from signal model $\mathbf{y} = |\mathbf{A}\mathbf{x}| + \mathbf{v}$, where $\mathbf{v}$ is circularly symmetric white Gaussian noise (unit variance). For signal vector $\mathbf{x} \in \mathbb{C}^N$ and matrix $\mathbf{A} \in \mathbb{C}^{M \times N}$ (both randomly drawn from a circularly-symmetric Gaussian distribution, unit variance), $N = 50$ elements and $M = 400$ measurements were simulated at SNR $= \{-10, 0, \ldots, 40\}$ dB, with noise variance obtained via (6.4). 100 Monte Carlo trials were performed for each SNR.

Recall that the phase retrieval formulation in (2.9) can be recast as the UQP in (2.11), which is the native formulation for PhaseCut. Further, we have demonstrated that this can also be cast as a ULS, to which we can directly apply our Projected Gradient algorithms.

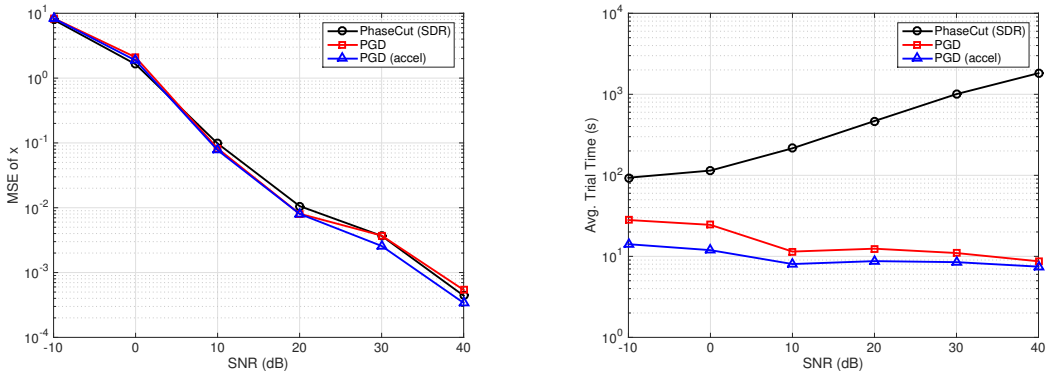

Figure 6.10: Mean squared error (left) and runtime (right) performance comparison for phase retrieval, $N = 50$, $M = 400$

As illustrated in Fig. 6.10, the two Projected Gradient Descent algorithms very nearly match PhaseCut with $M = 400$ measurements. Additionally, the first-order methods generally outperform PhaseCut in terms of runtime complexity, with Nesterov acceleration providing substantial performance gains (nearly two orders of magnitude at high SNR). A sample comparison of the convergence rate of both Projected Gradient Descent algorithms (fixed-step PGD and Nesterov's accelerated algorithm) is shown in Fig. 6.11.

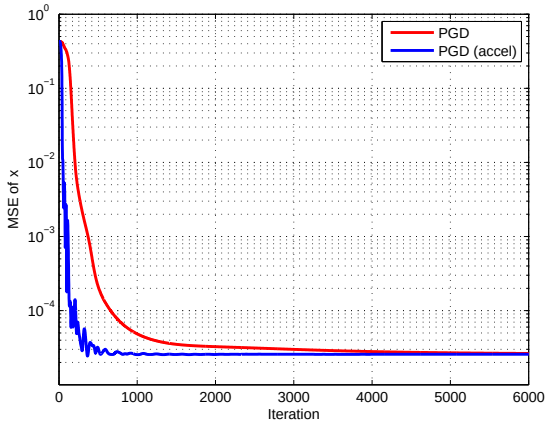

Figure 6.11: Convergence comparison for phase retrieval, $N = 50$, $M = 400$, SNR = 40 dB

Unlike the beamforming design scenario, phase retrieval is natively posed as an estimation problem. It can be shown [25] that the CRB for deterministic unknown $\mathbf{x}$ in the generative signal model $\mathbf{y} = |\mathbf{A}\mathbf{x}| + \mathbf{v}$, where $\mathbf{v}$ is i.i.d. real Gaussian, is given by

$$\begin{aligned}\mathbf{F} =& \frac{1}{\sigma^2}\begin{bmatrix}\mathrm{Re}\{\mathbf{A}^H\mathrm{Diag}(\mathbf{A}\mathbf{x})\}\\ \mathrm{Im}\{\mathbf{A}^H\mathrm{Diag}(\mathbf{A}\mathbf{x})\}\end{bmatrix}\mathrm{Diag}(|\mathbf{A}\mathbf{x}|^{-2})\\ &\times\begin{bmatrix}\mathrm{Re}\{\mathbf{A}^H\mathrm{Diag}(\mathbf{A}\mathbf{x})\}\\ \mathrm{Im}\{\mathbf{A}^H\mathrm{Diag}(\mathbf{A}\mathbf{x})\}\end{bmatrix}^T. \end{aligned} \tag{6.6}$$

Note that all algorithms converge to the same small gap above the CRB, as illustrated in Fig. 6.12. However, in terms of runtime, our proposed first-order methods perform substantially better than PhaseCut.

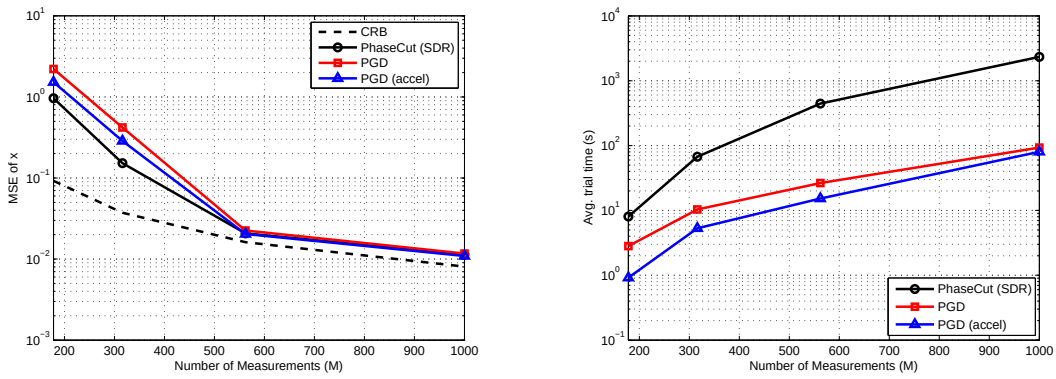

Figure 6.12: CRB/MSE (left) and runtime (right) comparison, $N = 50$, $M = 100, \ldots, 1000$, SNR $= 20$ dB

Chapter 7

Conclusion

In this paper we have proposed several first-order methods to tackle problems that admit a ULS / UQP representation, including phase-only beamforming and phase retrieval. Many other applications exist, such as in radar code design, see [15] and references therein. We have established convergence of these simple projected gradient-type algorithms to a KKT point of the original NP-hard problem. This is interesting because of the projections onto a *nonconvex* set, for which no analogous convergence result was previously available.

Selected problem generalizations and extensions were also considered, accounting for the need to auto-scale the response, or exploit additional degrees of freedom in the phase response, and corresponding algorithms were developed that combine first order methods with two-block alternating optimization.

The proposed algorithms were carefully compared against state-of-art methods based on SDR, and were found to perform at least as well in terms of least squares cost / mean squared error, and even better in several scenarios at significantly lower runtime complexity. Finally, we have observed that by employing the CRB, we can establish another benchmark for algorithm performance by associating NP-hard design problems with corresponding estimation problems.

References

[1] Shuzhong Zhang and Yongwei Huang. Complex quadratic optimization and semidefinite programming. *SIAM Journal on Optimization*, 16(3):871–890, 2006.

[2] Zhi-Quan Luo, Wing-Kin Ma, A.M.-C. So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *Signal Processing Magazine, IEEE*, 27(3):20–34, May 2010.

[3] I.E. Telatar. Capacity of multi-antenna Gaussian channels. *Eur. Trans. Telecommun.*, 10(6):585–596, Nov.-Dec. 1999.

[4] X. Zhang, A.F. Molisch, and S.-Y. Kung. Variable-phase-shift-based RF-baseband codesign for MIMO antenna selection. *Trans. on Signal Processing, IEEE*, 53(11):4091–4103, November 2005.

[5] J.G. Andrews, S. Buzzi, W. Choi, S.V. Hanly, A.L. Lozano, A.C.K. Soong, and J.C. Zhang. What will 5G be? *Journal on Selected Areas in Commun., IEEE*, 32(6):1065–1082, June 2014.

[6] C.A. Baird and G.G. Rassweiler. Adaptive sidelobe nulling using digitally controlled phase-shifters. *Trans. Antennas Propagat., IEEE*, AP-24:638–649, 1976.

[7] T.H. Ismail and M.M. Dawoud. Null steering in phased arrays by controlling the element positions. *Trans. Antennas Propagat., IEEE*, 39:1561–1566, 1991.

[8] P.A. Thompson. Adaptation by direct phase-shift adjustment in narrow-band adaptive antenna systems. *Trans. Antennas Propagat., IEEE*, 24(5):756–760, 1976.

[9] S. Smith. Optimum phase-only adaptive nulling. *Trans. on Signal Processing, IEEE*, 47(7):1835–1842, July 1999.

[10] W.S. Choi and T.K. Sarkar. Phase-only adaptive processing based on a direct data domain least squares approach using the conjugate gradient method. *Trans. Antennas Propagat., IEEE*, 10(6):585–596, Nov.-Dec. 2004.

[11] C.-J. Lu, W.-X. Sheng, Y.-B. Han, and X.-F. Ma. A novel adaptive phase-only beamforming algorithm based on semidefinite relaxation. *Phased Array Systems Technology, 2013 IEEE International Symposium on*, pages 617–621, October 2013.

[12] J.R. Fineup. Phase retrieval algorithms: a comparison. *Appl. Opt.*, 21(15):2758–2769, 1982.

[13] D. Griffin and J. Lim. Signal estimation from modified short-time fourier transform. *Trans. Acoust. Speech Signal Process., IEEE*, 32(2):236–243, 1984.

[14] I. Waldspurger, A. d'Aspremont, and S. Mallat. Phase recovery, MaxCut and complex semidefinite programming. *Mathematical Programming A*, 149:47–81, February 2015.

[15] M. Soltanalian and P. Stoica. Designing unimodular codes via quadratic optimization. *Trans. on Signal Processing, IEEE*, 62(5):1221–1234, March 2014.

[16] T.H. Ismail and Z.M. Hamici. Array pattern synthesis using digital phase control by quantized particle swarm optimization. *Trans. on Antennas and Propagation, IEEE*, 58(6):2142–2145, June 2010.

[17] T.K. Sinhamahapatra, A. Ahmed, G.K. Mahanti, N. Pathak, and A. Chakrabarty. Design of discrete phase-only dual-beam array antennas with minimum dynamic range ratio. *Applied Electromagnetics Conference, IEEE*, pages 1–4, December 2007.

[18] N.D. Sidiropoulos and Z.-Q. Luo. A semidefinite relaxation approach to MIMO detection for high-order QAM constellations. *Signal Processing Letters, IEEE*, 13(9):525–528, September 2006.

[19] Z. Wen, D. Goldfarb, S. Ma, and K. Scheinberg. Roy by row methods for semidefinite programming. *Dept. of IEOR, Columbia University, Tech. Rep.*, April 2009.

[20] X. Fu, F.K.W. Chan, W.-K. Ma, and H.-C. So. A complex-valued semidefinite relaxation approach for two-dimensional source localization using distance measurements and imperfect receiver positions. *Signal Processing (ICSP) Proceedings, IEEE*, 2:1491–1494, 2012.

[21] D.P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachusetts, 2008.

[22] J. Tranter, N.D. Sidiropoulos, X. Fu, and A. Swami. Fast unit-modulus least squares with applications in beamforming and phase retrieval. *submitted*, 2016.

[23] M. Razaviyayn, M. Hong, and Z.-Q. Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013.

[24] Bilian Chen, Simai He, Zhening Li, and Shuzhong Zhang. Maximum block improvement and polynomial optimization. *SIAM Journal on Optimization*, 22(1):87–107, 2012.

[25] C. Qian, X. Fu, N.D. Sidiropoulos, L. Huang, and J. Xie. Iterative reweighted least squares for phase retrieval in impulsive noise. *Trans. on Signal Processing, IEEE*, to be submitted for publication.

[26] Y.E. Nesterov. A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Dokl. Acad. Nauk SSSR*, pages 543–547, 1983.

[27] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.

[28] E. Karipidis, N.D. Sidiropoulos, and Zhi-Quan Luo. Far-field multicast beamforming for uniform linear antenna arrays. *Signal Processing, IEEE Transactions on*, 55(10):4916–4927, Oct 2007.

[29] S.M. Kay. *Fundamentals of Signal Processing: Estimation Theory.* Prentice Hall, Upper Saddle River, New Jersey, 1993.

Appendix A

Convergence Analysis for Projected Gradient Descent with Constant Modulus Constraint

Let us consider the following problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{C}^N} \quad & \|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 \\ \text{subject to} \quad & |x_i| = 1, i = 1, \ldots, N. \end{aligned} \tag{A.1}$$

Let $f(\mathbf{x})$ denote the convex and differentiable cost function in (A.1), which satisfies

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2^2 \leq L\|\mathbf{x}-\mathbf{y}\|_2^2. \tag{A.2}$$

To solve this problem, we apply the projected gradient algorithm, i.e.,

$$\begin{aligned} \mathbf{x}^{r+1} &= P(\mathbf{x}^r - \alpha(\nabla f(\mathbf{x}^r))) \\ &= P(\mathbf{x}^r + \alpha\mathbf{A}^H(\mathbf{y}-\mathbf{A}\mathbf{x})) \end{aligned} \tag{A.3}$$

where

$$P(\mathbf{y}) = \min_{|x_i|=1} \|\mathbf{x}-\mathbf{y}\|_2^2 = e^{j\angle(\mathbf{y})} \tag{A.4}$$

Claim 1 *Every limit point of the proposed projected gradient method is a KKT point of Problem* (A.1)*, provided that the step size* $\alpha \leq L$*. Furthermore, let* $\mathcal{X}^*$ *be the set of*

KKT points. Then, the whole sequence satisfies that

$$\lim_{r\to\infty} \text{dist}(\mathbf{x}^r, \mathcal{X}^*) = 0,$$

where $\text{dist}(\mathbf{x}^r, \mathcal{X}^*)$ *is defined as the distance between* $\mathbf{x}^r$ *and the nearest KKT point in* $\mathcal{X}^*$.

Proof: Let us rewrite Problem (A.1) in terms of the quadratic upper bound surrogate function

$$\min_{|x_i|=1} \min_{\mathbf{s}} \; g(\mathbf{x}, \mathbf{s}) = f(\mathbf{s}) + (\nabla f(\mathbf{s}))^T(\mathbf{x} - \mathbf{s}) + \frac{L}{2}\|\mathbf{x} - \mathbf{s}\|_2^2. \tag{A.5}$$

By (A.2), $g(\mathbf{x}, \mathbf{s}) \geq f(\mathbf{x})$ and the equality holds if and only if $\mathbf{s} = \mathbf{x}$. Now, let us derive an alternating optimization algorithm for problem (A.5). First, we optimize with respect to $\mathbf{s}$, so

$$\mathbf{s}^{r+1} = \arg\min_{\mathbf{s}} \; g(\mathbf{x}^r, \mathbf{s}),$$

which results in

$$\mathbf{s}^{r+1} = \mathbf{x}^r. \tag{A.6}$$

Then, we optimize with respect to $\mathbf{x}$, i.e.,

$$\begin{aligned}
\mathbf{x}^{r+1} =& \arg\min_{\mathbf{s}} \; g(\mathbf{x}, \mathbf{s}^{r+1}) \\
=& \arg\min_{|x_i|=1} \; f(\mathbf{s}^{r+1}) + (\nabla f(\mathbf{s}^{r+1}))^T(\mathbf{x} - \mathbf{s}^{r+1}) \\
&+ \frac{L}{2}\|\mathbf{x} - \mathbf{s}^{r+1}\|_2^2 \\
=& \arg\min_{|x_i|=1} \; f(\mathbf{x}^r) + (\nabla f(\mathbf{x}^r))^T(\mathbf{x} - \mathbf{x}^r) \\
&+ \frac{L}{2}\|\mathbf{x} - \mathbf{x}^r\|_2^2 \\
=& \arg\min_{|x_i|=1} \; \left\|\mathbf{x} - (\mathbf{x}^r - \frac{1}{L}\nabla f(\mathbf{x}^r))\right\|_2^2 \\
=& P(\mathbf{x}^r - \frac{1}{L}\nabla f(\mathbf{x}^r)).
\end{aligned} \tag{A.7}$$

Now, the claim is that every limit point of $(\mathbf{x}^r, \mathbf{s}^r)$ is a KKT point of Problem (A.5). In fact, since both of the updates with respect to $\mathbf{x}$ and $\mathbf{s}$ are conditionally optimal, it is easily seen that

$$g(\mathbf{x}^r, \mathbf{s}^r) \geq g(\mathbf{x}^{r+1}, \mathbf{s}^{r+1}). \tag{A.8}$$

Let $\{r_j\}$ be the convergent subsequence; we have

$$\lim_{j\to\infty}(\mathbf{x}^{r_j}, \mathbf{s}^{r_j}) = (\mathbf{x}^*, \mathbf{s}^*).$$

We see that

$$g(\mathbf{x}, \mathbf{s}^{r_j}) \geq g(\mathbf{x}^{r_j}, \mathbf{s}^{r_j}) \tag{A.9a}$$

$$\geq g(\mathbf{x}^{r_j+1}, \mathbf{s}^{r_j+1}) \tag{A.9b}$$

$$\geq g(\mathbf{x}^{r_{j+1}}, \mathbf{s}^{r_{j+1}}). \tag{A.9c}$$

Taking $j \to \infty$, we obtain

$$g(\mathbf{x}, \mathbf{s}^*) \geq g(\mathbf{x}^*, \mathbf{s}^*). \tag{A.10}$$

The inequality in (A.10) means that $\mathbf{x}^*$ is a block-wise minimum; and similarly for $\mathbf{s}^*$. Therefore, the partial KKT condition with respect to $\mathbf{x}$ is

$$\nabla g(\mathbf{x}^*, \mathbf{s}^*) + \lambda^*(2\mathbf{x}^* - 1) = 0. \tag{A.11}$$

In addition, it is easily shown that

$$\nabla f(\mathbf{x}^r) = \nabla g(\mathbf{x}^r, \mathbf{s}^r), \tag{A.12}$$

which leads to

$$\nabla f(\mathbf{x}^*) = \nabla g(\mathbf{x}^*, \mathbf{s}^*). \tag{A.13}$$

Combining, we see that

$$\nabla f(\mathbf{x}^*) + \lambda^*(2\mathbf{x}^* - 1) = 0, \tag{A.14}$$

meaning that $\mathbf{x}^*$ is a KKT point. Suppose that there is a subsequence x^{r_j} such that $\text{dist}(x^{r_j}, \mathcal{X}^\star) \geq \gamma$, where $\gamma > 0$. Since x^{r_j} lies in a compact set, it further has a subsequence that converges to a KKT point. This contradicts the assumption that $\text{dist}(x^{r_j}, \mathcal{X}^\star) \geq \gamma$, and thus $\mathbf{x}^*$ must lie in $\mathcal{X}^*$. ■

Appendix B

Cramér-Rao Bound Derivation of (6.1)

Consider the generative signal model $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{v}$, where $\mathbf{x}$ is constrained to the unit-modulus torus and $\mathbf{v} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$. We can explicitly parameterize in terms of the vector of angles $\boldsymbol{\theta}$ as $\mathbf{y} = \mathbf{A}e^{j\boldsymbol{\theta}} + \mathbf{v}$, hence $\mathbf{y} \sim \mathcal{N}(\mathbf{A}e^{j\boldsymbol{\theta}}, \sigma^2\mathbf{I})$. The Fisher Information Matrix for this model can be expressed as [29]

$$[\mathbf{F}]_{k,l} = \frac{2}{\sigma^2}\Re\left\{\left(\frac{\partial \mathbf{A}e^{j\boldsymbol{\theta}}}{\partial\theta_k}\right)^H\left(\frac{\partial \mathbf{A}e^{j\boldsymbol{\theta}}}{\partial\theta_l}\right)\right\}, \tag{B.1}$$

where parameter vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_N]^T$. We then obtain the derivative with respect to each θ_i as $\frac{\partial \mathbf{A}e^{j\boldsymbol{\theta}}}{\partial\theta_i} = je^{j\theta_i}\mathbf{a}_i$, where $\mathbf{a}_i$ denotes the i-th column of $\mathbf{A}$. It follows that

$$[\mathbf{F}]_{k,l} = \frac{2}{\sigma^2}\mathrm{Re}\left\{e^{-j\theta_k}\mathbf{a}_k^H\mathbf{a}_l e^{j\theta_l}\right\}, \tag{B.2}$$

from which we can construct the Fisher Information Matrix as

$$\mathbf{F} = \frac{2}{\sigma^2}\mathrm{Re}\left\{\mathrm{Diag}\left(e^{-j\boldsymbol{\theta}}\right)\mathbf{A}^H\mathbf{A}\mathrm{Diag}\left(e^{j\boldsymbol{\theta}}\right)\right\}. \tag{B.3}$$

Thus, the CRB on $\boldsymbol{\theta}$ can be compactly written as ($\mathbf{x} = e^{j\boldsymbol{\theta}}$)

$$CRB = \mathbf{F}^{-1} = \frac{\sigma^2}{2}\left[\mathrm{Re}\left\{\mathrm{Diag}(\mathbf{x})^H\mathbf{A}^H\mathbf{A}\mathrm{Diag}(\mathbf{x})\right\}\right]^{-1}. \tag{B.4}$$