On Convergence Rates of Quadratic Transform and WMMSE Methods

Kaiming Shen, Ziping Zhao, Yannan Chen, Zepeng Zhang, and Hei Victor Cheng

***Abstract*—Fractional programming (FP) plays an important role in information science because of the Cramér-Rao bound, the Fisher information, and the signal-to-interference-plus-noise ratio (SINR). A state-of-the-art method called the quadratic transform has been extensively used to address the FP problems. This work aims to accelerate the quadratic transform-based iterative optimization via gradient projection and extrapolation. The main contributions of this work are three-fold. First, we relate the quadratic transform to the gradient projection, thereby eliminating the matrix inverse operation from the iterative optimization; our result generalizes the weighted sum-of-rates (WSR) maximization algorithm in [1] to a wide range of FP problems. Second, based on this connection to gradient projection, we incorporate Nesterov's extrapolation strategy [2] into the quadratic transform so as to accelerate the convergence of the iterative optimization. Third, from a minorization-maximization (MM) point of view, we examine the convergence rates of the conventional quadratic transform methods—which include the weighted minimum mean square error (WMMSE) algorithm as a special case—and the proposed accelerated ones. Moreover, we illustrate the practical use of the accelerated quadratic transform in two popular application cases of future wireless networks: (i) integrated sensing and communication (ISAC) and (ii) massive multiple-input multiple-output (MIMO).**

***Index Terms*—Fractional programming (FP), quadratic transform, weighted minimum mean square error (WMMSE), convergence rate, acceleration, minorizatioin-maximization (MM).**

I. Overview

FRACTIONAL programming (FP) is the study of optimization aimed at the ratio terms. For the matrix coefficients $\boldsymbol{A},\boldsymbol{B}_1,\ldots,\boldsymbol{B}_n$ and the vector variables $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ with their sizes properly defined, this paper focuses on the following type of ratio term:

$$\big(\boldsymbol{A}\boldsymbol{x}_i\big)^{\mathrm{H}}\left(\sum_{j=1}^{n}\boldsymbol{B}_j\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_j^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}\boldsymbol{x}_i\big),$$

or its generalization with the matrix variables $\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n$ as

$$\big(\boldsymbol{A}\boldsymbol{X}_i\big)^{\mathrm{H}}\left(\sum_{j=1}^{n}\boldsymbol{B}_j\boldsymbol{X}_j\boldsymbol{X}_j^{\mathrm{H}}\boldsymbol{B}_j^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}\boldsymbol{X}_i\big),$$

where $(\cdot)^{\mathrm{H}}$ denotes the conjugate transpose,. The above ratio term is of significant research interest not only because it is a natural extension of the Rayleigh quotient, but also because several key metrics in the information science field are written in this form, e.g., the Cramér-Rao bound, the Fisher information, and the signal-to-interference-plus-noise ratio (SINR).

The quadratic transform [3], [4] is a state-of-the-art tool for FP. Its main idea is to decouple each ratio term and thereby reformulate the FP problem as a quadratic program that can be addressed efficiently (and often in closed form) in an iterative manner. As shown in [4], the quadratic transform has a connecting link to the minorization-maximization (MM) theory [5], [6], so it immediately follows that the quadratic transform method guarantees monotonic convergence to some stationary point provided that the original problem is differentiable. In particular, [7] shows that the quadratic transform method encompasses the well-known weighted minimum mean square error (WMMSE) algorithm [8], [9] as a special case; [7] further proposes a better way of applying the quadratic transform than WMMSE when dealing with discrete variables. Despite the extensive studies on the quadratic transform, its convergence rate remains a complete mystery (even for the WMMSE algorithm case), with the following open problems:

i. *How fast does the quadratic transform converge?*
ii. *How is it compared to the gradient method?*
iii. *Can we further accelerate the quadratic transform?*

Roughly speaking, the answers given in this paper are: when the starting point is sufficiently close to a strict local optimum, the quadratic transform yields an objective-value error bound of $O(1/k)$, where k is the iteration index; it can be faster than the gradient method in iterations, but slower in time; the error bound can be further reduced to $O(1/k^2)$ by incorporating Nesterov's extrapolation strategy [2].

The classic study on fractional programming concerns the ratio between the scalar-valued numerator and denominator [10]. The simplest case is the single-ratio problem. Under the concave-convex condition [3], even though the single-ratio problem is still nonconvex, its optimal solution can be efficiently obtained by the Charnes-Cooper algorithm [11], [12] or Dinkelbach's algorithm [13]. In contrast, the multi-ratio problem is much more challenging. Except for the max-min-ratio problem that can be efficiently solved by a generalized Dinkelbach's algorithm [14], most multi-ratio problems can only be handled by the branch-and-bound algorithms [15]–[24]. Actually, [15] shows that the multi-ratio problems are NP-complete. Differing from the traditional literature that

Manuscript submitted for possible publication December 12, 2023.

Kaiming Shen and Yannan Chen are with the School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen), 518172 Shenzhen, China (e-mail: shenkaiming@cuhk.edu.cn; yannanchen@link.cuhk.edu.cn).

Ziping Zhao and Zepeng Zhang are with the School of Information Science and Technology, ShanghaiTech University, 201210 Shanghai, China (e-mail: zipingzhao@shanghaitech.edu.cn; zhangzp1@shanghaitech.edu.cn.)

Hei Victor Cheng is with the Electrical and Computer Engineering Department, Aarhus Universrity, 8200 Aarhus, Denmark (e-mail: hvc@ieee.org).

aims at the global optimum of the multi-ratio problems, [3] seeks a stationary point—which can be readily obtained after every ratio term is decoupled by the quadratic transform. Furthermore, a line of works [3], [4], [7], [25], [26] extend the quadratic transform to the various FP cases.

As a special case of the quadratic transform, the WMMSE algorithm [8], [9] has been extensively considered in the literature for its own sake because of the weighted sum-of-rates (WSR) maximization problem in wireless networks. The computational complexity is however a major bottleneck of the WMMSE algorithm. The algorithm incurs frequent computation of the matrix inverse—which is costly in modern wireless networks because the matrix size is proportional to the number of antennas. Assuming that the channel matrices are all full row-rank, the recent work [27] takes advantage of the WSR problem structure to facilitate the matrix inverse computation. The more recent work [1] goes further: it does not require any channel assumptions and yet can get rid of the matrix inverse operation completely. Most importantly, [1] shows that an improved WMMSE algorithm can be interpreted as a gradient projection. One main contribution of the present work is to extend the results in [1] to a broad range of FP problems (not limited to the WSR problem). Moreover, another recent work [28] suggests combining Nesterov's extrapolation and WMMSE in a heuristic way, but its proposed algorithm still involves the matrix inverse operation and cannot provide any performance guarantee.

The main results of this paper are summarized below:

- *Accelerated Quadratic Transform:* We establish a connection between the quadratic transform and the gradient projection. As a result, the WSR maximization algorithm proposed in [1] can then be extended to a wide range of FP problems. Furthermore, in light of this connection, Nesterov's extrapolation strategy [2] can be incorporated into the quadratic transform to accelerate the convergence of the corresponding iterative optimization.
- *Convergence Rate Analysis:* We examine the local convergence behaviors of the various quadratic transform methods. We show that the conventional quadratic transform (including the WMMSE algorithm [8], [9] as a special case) yields faster convergence than the proposed nonhomogeneous quadratic transform in iterations, but slower in time, both of which guarantee an objective-value error bound of $O(1/k)$ given the iteration number k. The proposed extrapolated quadratic transform can further reduce the error bound to $O(1/k^2)$.
- *Application Cases:* We illustrate the use of the proposed accelerated quadratic transform in two popular application cases of the 6G network, i.e., the integrated sensing and communications (ISAC) and the massive multiple-input multiple-output (MIMO) transmission. Notice that the ISAC problem contains the Fisher information and SINRs, while the massive MIMO problem contains the SINRs which are nested in logarithms.

The rest of the paper is organized as follows. Section II states the sum-of-weighted-ratios FP problem. Section III reviews the conventional quadratic transform in [3], and then shows how the quadratic transform can be accelerated by using the gradient projection and Nesterov's extrapolation strategy, thus obtaining the nonhomogeneous quadratic transform and the extrapolated quadratic transform. Section V gives the convergence rate analysis for the different quadratic transform methods. Section VI discusses other FP problems. Section VII discusses the extension to the matrix ratio case. Two application cases are presented in Section VIII. Finally, Section IX concludes the paper.

Here and throughout, bold lower-case letters represent vectors while bold upper-case letters represent matrices. For a vector $\boldsymbol{a}$, $\boldsymbol{a}^c$ is its complex conjugate, $\boldsymbol{a}^{\mathrm{H}}$ is its conjugate transpose, and $\|\boldsymbol{a}\|_2$ is its ℓ_2 norm. For a matrix $\boldsymbol{A}$, $\boldsymbol{A}^c$ is its complex conjugate, $\boldsymbol{A}^\top$ is its transpose, $\boldsymbol{A}^{\mathrm{H}}$ is its conjugate transpose, $\lambda_{\max}(\boldsymbol{A})$ is its largest eigenvalue, and $\|\boldsymbol{A}\|_F$ is its Frobenius norm. For a square matrix $\boldsymbol{A}$, $\mathrm{tr}(A)$ is its trace. For a positive semi-definite matrix $\boldsymbol{A}$, $\boldsymbol{A}^{\frac{1}{2}}$ is its square root. Denote by $\boldsymbol{I}$ the identity matrix, $\mathbb{C}^\ell$ the set of $\ell \times 1$ vectors, $\mathbb{C}^{d\times m}$ the set of $d \times m$ matrices, and $\mathbb{S}^{d\times d}_{++}$ the set of $d \times d$ positive definite matrices. For a complex number $a \in \mathbb{C}$, $\Re\{a\}$ is its real part. The underlined letters represent the collections of the associated vectors or matrices, e.g., for $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \in \mathbb{C}^d$ we write $\underline{\boldsymbol{a}} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n]^\top \in \mathbb{C}^{nd}$.

II. Problem Statement

Consider a total of n ratio terms, each written as

$$M_i(\underline{\boldsymbol{x}}) = \big(\boldsymbol{A}_i\boldsymbol{x}_i\big)^{\mathrm{H}}\left(\sum_{j=1}^{n}\boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}_i\boldsymbol{x}_i\big), \tag{1}$$

where $\boldsymbol{A}_i \in \mathbb{C}^{\ell\times d}, \boldsymbol{B}_{ij} \in \mathbb{C}^{\ell\times d}, \boldsymbol{x}_j \in \mathbb{C}^d$ for all $i, j = 1, \ldots, n$. Assume that each $M_i(\underline{\boldsymbol{x}})$ is differentiable. Denote by $f_o(\underline{\boldsymbol{x}})$ the sum-of-weighted-ratios objective function:

$$f_o(\underline{\boldsymbol{x}}) = \sum_{i=1}^{n}\omega_i M_i(\underline{\boldsymbol{x}}), \tag{2}$$

where each weight $\omega_i > 0$. We consider the constrained sum-of-weighted-ratios FP problem:

$$\underset{\underline{\boldsymbol{x}}}{\text{maximize}} \quad f_o(\underline{\boldsymbol{x}}) \tag{3a}$$

$$\text{subject to} \quad \boldsymbol{x}_i \in \mathcal{X}_i, \ \text{for } i = 1, \ldots, n, \tag{3b}$$

where $\mathcal{X}_i$ is a nonempty convex set for $\boldsymbol{x}_i$. Let the Cartesian product $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n$ be the corresponding constraint set for $\underline{\boldsymbol{x}}$.

It is worth pointing out that we can include constant terms in the numerators and denominators by introducing some dummy variables $\boldsymbol{x}_j = [1, \ldots, 1]^\top$. Furthermore, with the variables being matrices $\boldsymbol{X}_i \in \mathbb{C}^{d\times m}$, each ratio term becomes

$$\boldsymbol{M}_i(\underline{\boldsymbol{X}}) = \big(\boldsymbol{A}_i\boldsymbol{X}_i\big)^{\mathrm{H}}\left(\sum_{j=1}^{n}\boldsymbol{B}_{ij}\boldsymbol{X}_j\boldsymbol{X}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}_i\boldsymbol{X}_i\big). \tag{4}$$

The resulting matrix-FP case is discussed in Section VII.

III. Quadratic Transform

We start by reviewing a state-of-the-art FP method called the quadratic transform [3], [7], and then establish its connection to the gradient projection method, based on which the accelerated quadratic transform is developed.

A. Preliminary

The previous work [3] proposes using the quadratic transform to decouple every ratio term as follows:

Proposition 1 (Theorem 2 in [3]): For a nonempty constraint set $\mathcal{X}$ as well as a sequence of function $\boldsymbol{s}_i : \mathcal{X} \to \mathbb{C}^\ell$ and function $\boldsymbol{G}_i : \mathcal{X} \to \mathbb{S}^{\ell\times\ell}_{++}$, $i = 1, \dots, n$, the FP problem

$$\begin{aligned} \underset{\underline{\boldsymbol{x}}}{\text{maximize}} \quad & \sum_{i=1}^{n} \boldsymbol{s}_i^{\mathrm{H}}(\underline{\boldsymbol{x}})\boldsymbol{G}_i^{-1}(\underline{\boldsymbol{x}})\boldsymbol{s}_i(\underline{\boldsymbol{x}}) \\ \text{subject to} \quad & \underline{\boldsymbol{x}} \in \mathcal{X} \end{aligned}$$

is equivalent to

$$\begin{aligned} \underset{\underline{\boldsymbol{x}},\,\underline{\boldsymbol{y}}}{\text{maximize}} \quad & \sum_{i=1}^{n} \Big[2\Re\{\boldsymbol{s}_i^{\mathrm{H}}(\underline{\boldsymbol{x}})\boldsymbol{y}_i\} - \boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{G}_i(\underline{\boldsymbol{x}})\boldsymbol{y}_i\Big] \\ \text{subject to} \quad & \underline{\boldsymbol{x}} \in \mathcal{X},\ \boldsymbol{y}_i \in \mathbb{C}^\ell,\ \text{for } i = 1, \dots, n \end{aligned}$$

in the sense that $\underline{\boldsymbol{x}}^\star$ is a solution to the original problem if and only if $(\underline{\boldsymbol{x}}^\star, \underline{\boldsymbol{y}}^\star)$ is a solution to the new problem with an optimal $\underline{\boldsymbol{y}}^\star$.

The above quadratic transform can be readily extended to the sum-of-weighted-ratios case in (2); the original objective function $f_o(\underline{\boldsymbol{x}})$ is then converted to a new objective function:

$$f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}) = \sum_{i=1}^{n} \omega_i \left[2\Re\{\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i\} - \sum_{j=1}^{n} \boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\boldsymbol{y}_i\right]. \tag{5}$$

The benefit of adopting this new objective is that the primal variable $\underline{\boldsymbol{x}}$ and the auxiliary variable $\underline{\boldsymbol{y}}$ can be alternatingly optimized in closed form. By completing the square for each $\boldsymbol{y}_i$ in (5), the optimal $\boldsymbol{y}_i$ can be obtained as

$$\boldsymbol{y}_i^\star = \left(\sum_{j=1}^{n} \boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1} \big(\boldsymbol{A}_i\boldsymbol{x}_i\big). \tag{6}$$

To solve for $\underline{\boldsymbol{x}}$ with $\underline{\boldsymbol{y}}$ held fixed, we rewrite $f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}})$ as

$$f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}) = \sum_{i=1}^{n} \Big[2\Re\{\omega_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i\} - \boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{D}_i\boldsymbol{x}_i\Big], \tag{7}$$

where

$$\boldsymbol{D}_i = \sum_{j=1}^{n} \omega_j\boldsymbol{B}_{ji}^{\mathrm{H}}\boldsymbol{y}_j\boldsymbol{y}_j^{\mathrm{H}}\boldsymbol{B}_{ji}, \tag{8}$$

and then optimally determine $\boldsymbol{x}_i$ as

$$\boldsymbol{x}_i^\star = \arg\min_{\boldsymbol{x}_i \in \mathcal{X}_i} \big\|\boldsymbol{D}_i^{\frac{1}{2}}\big(\boldsymbol{x}_i - \omega_i\boldsymbol{D}_i^{-1}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i\big)\big\|_2. \tag{9}$$

In particular, if $\omega_i\boldsymbol{D}_i^{-1}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i \in \mathcal{X}_i$, then the optimal update of $\boldsymbol{x}_i$ is given by

$$\boldsymbol{x}_i^\star = \omega_i\boldsymbol{D}_i^{-1}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i.$$

Algorithm 1 summarizes the above steps of the conventional quadratic transform method. Moreover, as shown in [4], Algorithm 1 yields a monotonic convergence to a stationary point of the original FP problem so long as $f_o(\underline{\boldsymbol{x}})$ is differentiable.

Algorithm 1 Conventional Quadratic Transform [3]

```
Initialize $\underline{\boldsymbol{x}}$ to a feasible value.
repeat
    Update each $\boldsymbol{y}_i$ according to (6).
    Update each $\boldsymbol{x}_i$ according to (9).
until the value of $f_o(\underline{\boldsymbol{x}})$ converges
```

The computation in (9) can be quite costly when $\boldsymbol{D}_i$ is a large matrix. In Section III-B, we show that (9) can be recognized as the projection on an ellipsoid, and suggest converting it to the projection on a sphere—which is computationally much easier to carry out.

B. Connection with Gradient Projection Method

The following result is inspired by the WSR maximization algorithm recently proposed in [1]. We first introduce two lemmas.

Lemma 1: After $\underline{\boldsymbol{y}}$ has been updated as in (6) for the current $\underline{\boldsymbol{x}}$, the partial derivative of each fractional function $M_i(\underline{\boldsymbol{x}})$ with respect to the complex conjugate[1] of $\boldsymbol{x}_j$ is given by

$$\frac{\partial M_i(\underline{\boldsymbol{x}})}{\partial \boldsymbol{x}_j^{\mathrm{c}}} = \begin{cases} 2\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i - 2\boldsymbol{B}_{ii}^{\mathrm{H}}\boldsymbol{y}_i\boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{B}_{ii}\boldsymbol{x}_i & \text{if } j = i; \\ -2\boldsymbol{B}_{ji}^{\mathrm{H}}\boldsymbol{y}_j\boldsymbol{y}_j^{\mathrm{H}}\boldsymbol{B}_{ji}\boldsymbol{x}_i & \text{if } j \neq i. \end{cases}$$

Lemma 2 (Nonhomogeneous Bound [6]): Suppose that the two Hermitian matrices $\boldsymbol{L}, \boldsymbol{K} \in \mathbb{C}^{d\times d}$ satisfy the condition $\boldsymbol{L} \preceq \boldsymbol{K}$. Then for any two vectors $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{C}^d$, one has

$$\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}\boldsymbol{x} \leq \boldsymbol{x}^{\mathrm{H}}\boldsymbol{K}\boldsymbol{x} + 2\Re\{\boldsymbol{x}^{\mathrm{H}}(\boldsymbol{L}-\boldsymbol{K})\boldsymbol{z}\} + \boldsymbol{z}^{\mathrm{H}}(\boldsymbol{K}-\boldsymbol{L})\boldsymbol{z}, \tag{10}$$

where the equality holds if $\boldsymbol{z} = \boldsymbol{x}$. The above bound is called nonhomogeneous due to the linear term $2\Re\{\boldsymbol{x}^{\mathrm{H}}(\boldsymbol{L}-\boldsymbol{K})\boldsymbol{z}\}$.

Treating $\boldsymbol{D}_i$ as $\boldsymbol{L}$ in (10), we let

$$\boldsymbol{K} = \lambda_i\boldsymbol{I} \quad \text{where } \lambda_i \geq \lambda_{\max}(\boldsymbol{D}_i) \tag{11}$$

so as to have $\boldsymbol{L} \preceq \boldsymbol{K}$; one possible choice is $\lambda_i = \|\boldsymbol{D}_i\|_F$. Thus, by virtue of Lemma 2, we can bound $f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}})$ in (7) from below as

$$f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}) \geq f_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}) \tag{12}$$

for any $(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}})$, where

$$\begin{aligned} f_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}) = \sum_{i=1}^{n} \Big[& 2\Re\{\omega_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i + \boldsymbol{x}_i^{\mathrm{H}}(\lambda_i\boldsymbol{I} - \boldsymbol{D}_i)\boldsymbol{z}_i\} \\ & + \boldsymbol{z}_i^{\mathrm{H}}(\boldsymbol{D}_i - \lambda_i\boldsymbol{I})\boldsymbol{z}_i - \lambda_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{x}_i\Big]. \end{aligned} \tag{13}$$

In particular, the equality in (12) holds if $\boldsymbol{z}_i = \boldsymbol{x}_i$ for all i.

We take $f_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}})$ as a lower-bound approximation of $f_q(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}})$ and optimize the three variables $(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}})$ iteratively. Observe that each iterate can be performed in closed form. When $\underline{\boldsymbol{y}}$ and $\underline{\boldsymbol{x}}$ are both held fixed, the optimal update of $\underline{\boldsymbol{z}}$ follows by the equality condition in Lemma 2 as

$$\boldsymbol{z}_i^\star = \boldsymbol{x}_i, \quad \text{for } i = 1, \dots, n. \tag{14}$$

[1] The motivation of considering $\partial M_i(\underline{\boldsymbol{x}})/\partial \boldsymbol{x}_j^{\mathrm{c}}$ rather than $\partial M_i(\underline{\boldsymbol{x}})/\partial \boldsymbol{x}_j$ is that the resulting differential is simpler. According to Theorem 2 in [29], the two types of partial derivatives are both feasible for computing the stationary point. In the rest of the paper, we shall always use the former type.

After $\underline{\boldsymbol{z}}$ has been updated to $\underline{\boldsymbol{x}}$, for $\underline{\boldsymbol{z}}$ and $\underline{\boldsymbol{x}}$ both fixed, the optimal $\boldsymbol{y}_i$ in (13) is still determined as in (6). Next, when $\underline{\boldsymbol{y}}$ and $\underline{\boldsymbol{z}}$ are both held fixed, the optimal $\boldsymbol{x}_i$ in (13) is given by

$$\begin{aligned}\boldsymbol{x}_i^\star &= \arg\min_{\boldsymbol{x}_i\in\mathcal{X}_i} \big\|\lambda_i\boldsymbol{x}_i - \omega_i\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i - \big(\lambda_i\boldsymbol{I}-\boldsymbol{D}_i\big)\boldsymbol{z}_i\big\|_2\\ &= \mathcal{P}_{\mathcal{X}_i}\bigg(\boldsymbol{z}_i + \frac{1}{\lambda_i}\Big(\omega_i\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i - \boldsymbol{D}_i\boldsymbol{z}_i\Big)\bigg), \qquad (15)\end{aligned}$$

where $\mathcal{P}_{\mathcal{X}_i}(\cdot)$ is the projection on $\mathcal{X}_i$ in the Euclidean distance.

We are now ready to interpret the above iterative optimization as a gradient projection method. We use the superscript $k=1,2,\ldots$ to index the iteration, and assume that the three variables $(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}})$ are cyclically updated as

$$\underline{\boldsymbol{x}}^0 \to \cdots \to \underline{\boldsymbol{x}}^{k-1} \to \underline{\boldsymbol{z}}^k \to \underline{\boldsymbol{y}}^k \to \underline{\boldsymbol{x}}^k \to \underline{\boldsymbol{z}}^{k+1} \to \cdots.$$

With the optimal $\underline{\boldsymbol{y}}$ in (6) and the optimal $\underline{\boldsymbol{z}}$ in (14) substituted into (15), the optimal update of $\boldsymbol{x}_i$ in iteration k boils down to a gradient projection:

$$\begin{aligned}\boldsymbol{x}_i^k &= \mathcal{P}_{\mathcal{X}_i}\bigg(\boldsymbol{z}_i^k + \frac{1}{\lambda_i^k}\Big(\omega_i\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i^k - \boldsymbol{D}_i^k\boldsymbol{z}_i^k\Big)\bigg)\\ &\overset{(a)}{=} \mathcal{P}_{\mathcal{X}_i}\bigg(\boldsymbol{x}_i^{k-1} + \frac{1}{\lambda_i^k}\Big(\omega_i\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i^k - \boldsymbol{D}_i^k\boldsymbol{x}_i^{k-1}\Big)\bigg)\\ &\overset{(b)}{=} \mathcal{P}_{\mathcal{X}_i}\Bigg(\boldsymbol{x}_i^{k-1} + \frac{1}{2\lambda_i^k}\sum_{j=1}^n\bigg[\omega_j\cdot\frac{\partial M_j(\underline{\boldsymbol{x}}^{k-1})}{\partial\boldsymbol{x}_i^c}\bigg]\Bigg)\\ &= \mathcal{P}_{\mathcal{X}_i}\bigg(\boldsymbol{x}_i^{k-1} + \frac{1}{2\lambda_i^k}\cdot\frac{\partial f_o(\underline{\boldsymbol{x}}^{k-1})}{\partial\boldsymbol{x}_i^c}\bigg),\end{aligned}$$

in which $\boldsymbol{D}_i$ is assigned the iteration index k because it has been updated by (8) for $\underline{\boldsymbol{y}}^k$, and λ_i is assigned the iteration index k because it has been updated by (11) for $\boldsymbol{D}_i^k$. Here, step (a) follows by (14), and step (b) follows by Lemma 1.

Remark 1: The gradient projection interpretation was first proposed in [1] for its WSR maximization algorithm, wherein each $\boldsymbol{x}_i$ is a transmit beamforming vector and $\mathcal{X}_i$ is the power constraint set. This is discussed in more detail in Section VI.

Remark 2: From a geometric perspective, Algorithm 1 requires the projection on an ellipsoid as in (9), while Algorithm 2 requires the projection on a sphere as in (15); the latter task is computationally much easier.

IV. Accelerated Quadratic Transform

When connecting the quadratic transform to the gradient projection in Section III-B, we already implicitly devise an iterative algorithm for the FP problem, as summarized in Algorithm 2 and referred to as the nonhomogeneous quadratic transform method because of the use of Lemma 2. But is Algorithm 2 more efficient than Algorithm 1?

In terms of the per-iteration complexity, it is evident that Algorithm 2 is more efficient since it does not[2] require computing matrix inverse for the iterative update of $\underline{\boldsymbol{x}}$.

Algorithm 2 Nonhomogeneous Quadratic Transform

```
Initialize $\underline{\boldsymbol{x}}$ to a feasible value.
repeat
    Update each $\boldsymbol{z}_i$ according to (14).
    Update each $\boldsymbol{y}_i$ according to (6).
    Update each $\boldsymbol{x}_i$ according to (15).
until the value of $f_o(\underline{\boldsymbol{x}})$ converges
```

The overall complexity is however much more difficult to examine because it also depends on how many iterations the algorithm entails to reach the convergence. Algorithm 1 uses $f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}})$ to approximate the original objective $f_o(\underline{\boldsymbol{x}})$ from below, while Algorithm 2 further uses $f_t(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}})$ to approximate $f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}})$ from below, i.e.,

$$f_o(\underline{\boldsymbol{x}}) \ge f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}}) \ge f_t(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}}).$$

Intuitively speaking, Algorithm 1 should converge faster in iterations because its approximation of $f_o(\underline{\boldsymbol{x}})$ is tighter. A formal analysis of their convergence rates is provided in Section V.

To sum up, Algorithm 2 is more efficient per iteration, but in the meanwhile requires more iterations to attain convergence. One can find a balance between Algorithm 1 and Algorithm 2 via timesharing; the convergence to a stationary point is still guaranteed by the MM theory as shown in Section V.

Nevertheless, our idea is to reduce the number of iterations for Algorithm 2 by means of extrapolation. In principle, since the nonhomogeneous quadratic transform in essence utilizes the gradients, its convergence can be accelerated by momentum or heavy-ball method. Specifically, following Nesterov's extrapolation strategy [2], we propose to extrapolate each $\boldsymbol{x}_i$ along the direction of the difference between the preceding two iterates before the gradient projection, i.e.,

$$\boldsymbol{\nu}_i^{k-1} = \boldsymbol{x}_i^{k-1} + \eta_{k-1}(\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_i^{k-2}), \qquad (16)$$

$$\boldsymbol{x}_i^k = \mathcal{P}_{\mathcal{X}_i}\bigg(\boldsymbol{\nu}_i^{k-1} + \frac{1}{2\lambda_i^k}\cdot\frac{\partial f_o(\underline{\boldsymbol{\nu}}^{k-1})}{\partial\boldsymbol{x}_i^c}\bigg), \qquad (17)$$

where the extrapolation step η_k is chosen as

$$\eta_k = \max\left\{\frac{k-2}{k+1}, 0\right\}, \quad \text{for } k=1,2,\ldots,$$

and the starting point is $\boldsymbol{x}^{-1}=\boldsymbol{x}^0$. The above gradient projection with extrapolation can be implemented with the assistance of the auxiliary variables $(\underline{\boldsymbol{y}},\underline{\boldsymbol{z}})$ as shown in Algorithm 3, which is referred to as the extrapolated quadratic transform.

Fig. 1 compares the above three algorithms numerically. Aside from Algorithms 1 to 3, we consider the gradient method with a diminishing stepsize $1/k$ where k is the iteration index, and also a variant of Algorithm 3 by using Polyak's extrapolation strategy [30] in place of Nesterov's extrapolation strategy (i.e., the projection step is now performed prior to the extrapolation step). Observe that Algorithm 1 converges faster than Algorithm 3 in iterations according to Fig. 1(a), but this is no longer the case when the convergence is considered in terms of time as shown in Fig. 1(b).

[2] Notice that Algorithm 2 still requires computing the $\ell\times\ell$ matrix inverse when updating $\underline{\boldsymbol{y}}$. Actually, this matrix inverse can be also eliminated by applying Lemma 2 one more time, and consequently it would introduce a new group of auxiliary variables. We do not consider this straightforward extension in this paper because ℓ is quite small in our application cases and thus it is unnecessary to eliminate the matrix inverse in (6).

Algorithm 3 Extrapolated Quadratic Transform

```
Initialize $\underline{\boldsymbol{x}}$ to a feasible value.
repeat
    Update each $\boldsymbol{\nu}_i$ according to (16) and set $\boldsymbol{x}_i = \boldsymbol{\nu}_i$.
    Update each $\boldsymbol{z}_i$ according to (14).
    Update each $\boldsymbol{y}_i$ according to (6).
    Update each $\boldsymbol{x}_i$ according to (15).
until the value of $f_o(\underline{\boldsymbol{x}})$ converges
```

V. Convergence Analysis

In this section, we first show that the various quadratic transform methods all guarantee convergence to a stationary point of the FP problem in (3), and then analyze their rates of convergence.

The proof of the stationary-point convergence is based on the MM theory. Write the optimal update of $\underline{\boldsymbol{y}}$ in (6) as a function of $\underline{\boldsymbol{x}}$:

$$\mathcal{Y}(\underline{\boldsymbol{x}}) = \underline{\boldsymbol{y}} \text{ with each } \boldsymbol{y}_i = \left(\sum_{i=1}^{n} \boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}(\boldsymbol{A}_i\boldsymbol{x}_i).$$

By Algorithm 1, after $\underline{\boldsymbol{y}}^k$ is optimally updated for the previous $\underline{\boldsymbol{x}}^{k-1}$, the current new objective function $f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}})$ can be rewritten as a function $r_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1})$ of $\underline{\boldsymbol{x}}$ conditioned on $\underline{\boldsymbol{x}}^{k-1}$:

$$r_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1}) = f_q(\underline{\boldsymbol{x}},\mathcal{Y}(\underline{\boldsymbol{x}}^{k-1})), \tag{18}$$

and accordingly the update of $\underline{\boldsymbol{x}}$ in (9) can be rewritten as

$$\underline{\boldsymbol{x}}^k = \arg\max_{\underline{\boldsymbol{x}}\in\mathcal{X}} r_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1}). \tag{19}$$

Importantly, it always holds that

$$r_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1}) \le f_o(\underline{\boldsymbol{x}}) \text{ and } r_q(\underline{\boldsymbol{x}}^{k-1}|\underline{\boldsymbol{x}}^{k-1}) = f_o(\underline{\boldsymbol{x}}^{k-1}),$$

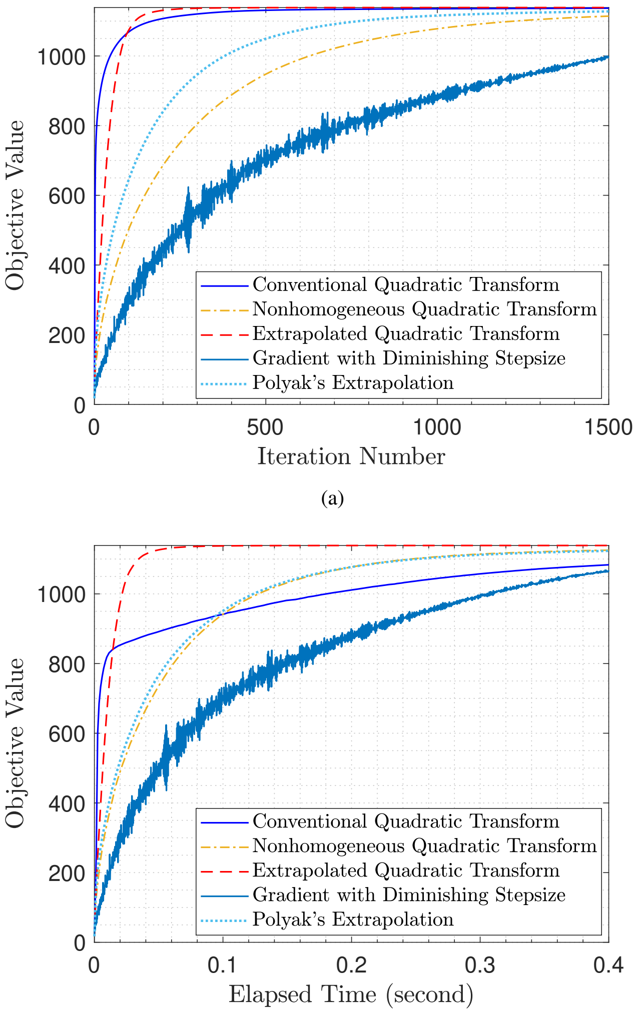
so updating $\underline{\boldsymbol{y}}$ for $\underline{\boldsymbol{x}}^{k-1}$ is equivalent to constructing a surrogate function $r_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1})$ for $f_o(\underline{\boldsymbol{x}})$ at $\underline{\boldsymbol{x}}^{k-1}$, namely the *minorization* step. Moreover, (19) can be recognized as the *maximization* step. As such, Algorithm 1 turns out to be an MM method, and hence it guarantees convergence to a stationary point of problem (3). By a similar argument, we can also interpret Algorithm 2 as an MM method, with the surrogate function

$$r_t(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1}) = f_t(\underline{\boldsymbol{x}},\mathcal{Y}(\underline{\boldsymbol{x}}^{k-1}),\underline{\boldsymbol{x}}^{k-1}). \tag{20}$$

Besides, the tradeoff between Algorithm 1 and Algorithm 2 via timesharing constitutes an MM algorithm as well and hence preserves the stationary-point convergence. Furthermore, recall that Algorithm 2 can also be interpreted as a gradient projection method; since it has provable convergence to a stationary point, so does its accelerated version Algorithm 3. The following proposition summarizes the above results.

Proposition 2: Algorithms 1 and 2 are both the MM methods. Algorithms 1, 2, and 3 all guarantee convergence to some stationary point of the FP problem 3.

We then analyze the rate of convergence for the various quadratic transform methods. Due to the nonconvexity of the FP problem, the global analysis (assuming that the starting point is far from any stationary point) is intractable. We would like to give a local analysis by restricting the constraint set to a small neighborhood of a strict local optimum (so that the starting point is not far away), i.e.,



(a)

(b)

Fig. 1. Solving the FP problem (3) in which $n = 5$, $d = 9$, $\ell = 4$ each $\omega_i = 1$, each $\mathcal{X}_i = \{\boldsymbol{X} \in \mathbb{C}^{d\times\ell} : \mathrm{tr}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{H}}) \le 10\}$, and each entry of $\boldsymbol{A}_i$ and $\boldsymbol{B}_{ij}$ drawn i.i.d. according to the complex Gaussian distribution $\mathcal{CN}(0,1)$; further, we add $\boldsymbol{I}$ to each matrix denominator to ensure positive definiteness. We plot the average convergence of the different algorithms over 100 random realizations. Figure (a) displays the convergence in iterations, while figure (b) displays the convergence in time.

$$\mathcal{X} = \{\underline{\boldsymbol{x}} : \|\underline{\boldsymbol{x}} - \underline{\boldsymbol{x}}^*\|_2 \le R\}, \tag{21}$$

where $\underline{\boldsymbol{x}}^*$ is a strict local optimum of (3) satisfying

$$\nabla^2 f_o(\underline{\boldsymbol{x}}^*) \preceq -\xi\boldsymbol{I} \prec \boldsymbol{0}$$

for some strictly positive constant $\xi > 0$, and the radius $R > 0$ is sufficiently small so that $f_o(\underline{\boldsymbol{x}})$ is concave on $\mathcal{X}$. Assume that the Hessian of $f_o(\underline{\boldsymbol{x}})$ is L-Lipschitz continuous on $\mathcal{X}$, i.e.,

$$\|\nabla^2 f_o(\underline{\boldsymbol{x}}) - \nabla^2 f_o(\underline{\boldsymbol{x}}')\|_2 \le L\|\underline{\boldsymbol{x}} - \underline{\boldsymbol{x}}'\|_2$$

for any $\underline{\boldsymbol{x}},\underline{\boldsymbol{x}}' \in \mathcal{X}$. By Corollary 1.2.2 of [2], we have

$$\nabla^2 f_o(\underline{\boldsymbol{x}}) \preceq \nabla^2 f_o(\underline{\boldsymbol{x}}^*) + L\|\underline{\boldsymbol{x}} - \underline{\boldsymbol{x}}^*\|_2\boldsymbol{I},$$

so it suffices to require $R \leq \xi/L$ in order to ensure that $f_o(\underline{\boldsymbol{x}})$ is concave on $\mathcal{X}$.

The following analysis uses the MM interpretation in Proposition 2. Conditioned on $\underline{\boldsymbol{x}}' \in \mathcal{X}$, define the gaps between $f_o(\underline{\boldsymbol{x}})$ and the two surrogate functions to be two functions of $\underline{\boldsymbol{x}} \in \mathcal{X}$ as

$$\begin{aligned} \delta_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}') &= f_o(\underline{\boldsymbol{x}}) - f_q(\underline{\boldsymbol{x}}, \mathcal{Y}(\underline{\boldsymbol{x}}')),\\ \delta_t(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}') &= f_o(\underline{\boldsymbol{x}}) - f_t(\underline{\boldsymbol{x}}, \mathcal{Y}(\underline{\boldsymbol{x}}'), \underline{\boldsymbol{x}}'). \end{aligned}$$

It can be readily shown that

$$\delta_q(\underline{\boldsymbol{x}}^k|\underline{\boldsymbol{x}}^k) = \delta_t(\underline{\boldsymbol{x}}^k|\underline{\boldsymbol{x}}^k) = 0, \tag{22a}$$

$$\nabla\delta_q(\underline{\boldsymbol{x}}^k|\underline{\boldsymbol{x}}^k) = \nabla\delta_t(\underline{\boldsymbol{x}}^k|\underline{\boldsymbol{x}}^k) = \mathbf{0}. \tag{22b}$$

Moreover, define the two quantities:

$$\begin{aligned} \Lambda_q &= \max_{\underline{\boldsymbol{x}}\in\mathcal{X}} \lambda_1\big(\nabla^2\delta_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}})\big),\\ \Lambda_t &= \max_{\underline{\boldsymbol{x}}\in\mathcal{X}} \lambda_1\big(\nabla^2\delta_t(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}})\big). \end{aligned}$$

Recall each ratio $M_i(\underline{\boldsymbol{x}})$ is finite with nonsingular denominator matrix $\sum_{i=1}^n \boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}$, so each entry of $\underline{\boldsymbol{y}}$ is finite and hence each $\lambda_1(\boldsymbol{D}_i) < \infty$ according to (8). As a result, $\lambda_1(\nabla^2_{\underline{\boldsymbol{x}}} f_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}) = -\max_i 2\lambda_1(\boldsymbol{D}_i) > -\infty$. Further, $\Lambda_t \leq \lambda_1(\nabla^2_{\underline{\boldsymbol{x}}} f_o(\underline{\boldsymbol{x}})) - \lambda_1(\nabla^2_{\underline{\boldsymbol{x}}} f_t(\underline{\boldsymbol{x}}, \mathcal{Y}(\underline{\boldsymbol{x}}'), \underline{\boldsymbol{x}}')) < \infty$. Moreover, because $\nabla^2_{\underline{\boldsymbol{x}}} f_t(\underline{\boldsymbol{x}}, \mathcal{Y}(\underline{\boldsymbol{x}}'), \underline{\boldsymbol{x}}') \preceq \nabla^2_{\underline{\boldsymbol{x}}} f_o(\underline{\boldsymbol{x}}, \mathcal{Y}(\underline{\boldsymbol{x}}'))$, we must have $\Lambda_q \leq \Lambda_t < \infty$. We are now ready to show the (local) convergence rates of Algorithm 1 and Algorithm 2.

Proposition 3 (Convergence Rates of Algorithm 1 and Algorithm 2): For the FP problem (3), the local convergence rate of Algorithm 1 or Algorithm 2 is

$$f_o(\underline{\boldsymbol{x}}^*) - f_o(\underline{\boldsymbol{x}}^1) \leq \frac{\Lambda R^2}{2} + \frac{LR^3}{6}, \tag{23}$$

$$f_o(\underline{\boldsymbol{x}}^*) - f_o(\underline{\boldsymbol{x}}^k) \leq \frac{2\Lambda R^2 + 2LR^3/3}{k+3}, \text{ for } k \geq 2, \tag{24}$$

where

$$\Lambda = \begin{cases} \Lambda_q & \text{for Algorithm 1;}\\ \Lambda_t & \text{for Algorithm 2.} \end{cases} \tag{25}$$

Proof: See Appendix A. ∎

Because $0 \leq \Lambda_q \leq \Lambda_t$, Algorithm 1 converges faster than Algorithm 2 in iterations according to Proposition 3. Notice that Λ_q and Λ_t in essence characterize how well their corresponding surrogate functions approximate the second-order profile of $f_o(\underline{\boldsymbol{x}})$. In the ideal case, the surrogate function and $f_o(\underline{\boldsymbol{x}})$ have exactly the same second-order profile so that $\Lambda = 0$, then the objective-value error bound in Proposition 3 becomes

$$f_o(\underline{\boldsymbol{x}}) - f_o(\underline{\boldsymbol{x}}^k) \leq \frac{L}{6}\|\underline{\boldsymbol{x}} - \underline{\boldsymbol{x}}^{k-1}\|_2^3, \tag{26}$$

which also holds for the *cubically regularized Newton's method* due to Nesterov as shown in [2]. Equipped with the error bound (26), it immediately follows from Theorem 4.1.4 in [2] that

$$f_o(\underline{\boldsymbol{x}}^*) - f_o(\underline{\boldsymbol{x}}^1) \leq \frac{LR^3}{6}, \tag{27}$$

$$f_o(\underline{\boldsymbol{x}}^*) - f_o(\underline{\boldsymbol{x}}^k) \leq \frac{LR^3}{2(1+k/3)^2}, \text{ for } k \geq 2. \tag{28}$$

We now show that the extrapolated quadratic transform method in Algorithm 3 can achieve fairly close to the ideal case stated in (27) and (28). Since Algorithm 2 is a gradient projection method and Algorithm 3 accelerates it by Nesterov's extrapolation, we immediately obtain the following convergence rate from Proposition 6.2.1 of [31].

Proposition 4 (Convergence Rate of Algorithm 3): Suppose that the gradient of $f_o(\underline{\boldsymbol{x}})$ is C-Lipschitz continuous and let $\lambda_i^k = 1/(2C)$. Then Algorithm 3 yields

$$f(\underline{\boldsymbol{x}}^*) - f(\underline{\boldsymbol{x}}) \leq \frac{2C\cdot[f(\underline{\boldsymbol{x}}^*) - f(\underline{\boldsymbol{x}}^0)]}{(k+1)^2}, \text{ for } k \geq 1. \tag{29}$$

In summary, as compared to Algorithm 1 and Algorithm 2 that both yield an objective-value error bound of $O(1/k)$, Algorithm 3 yields a smaller error bound of $O(1/k^2)$.

VI. Other FP Problems

Our discussion thus far is limited to the sum-of-weighted-ratios FP problem in (3). The goal of this section is to extend the above results to the other FP problems. We now consider a general FP objective function

$$g_o(\underline{\boldsymbol{x}}) = \mathcal{G}\big(M_1(\underline{\boldsymbol{x}}), \ldots, M_n(\underline{\boldsymbol{x}})\big) \tag{30}$$

in place of $f_o(\underline{\boldsymbol{x}})$ in problem (3), where $\mathcal{G}: \mathbb{R}^n \to \mathbb{R}$ is a differentiable function with n ratio arguments.

Assume that $g_o(\underline{\boldsymbol{x}})$ can be bounded from below as

$$h(\underline{\boldsymbol{x}}, \boldsymbol{t}) = \sum_{i=1}^n \Big[\alpha_i(\boldsymbol{t})\cdot\hat{M}_i(\underline{\boldsymbol{x}})\Big] + \beta(\boldsymbol{t}), \tag{31}$$

where $\boldsymbol{t}$ is an auxiliary variable, $\alpha_i(\boldsymbol{t}) \geq 0$, $i = 1, \ldots, n$, and $\beta(\boldsymbol{t})$ are all differentiable scalar-valued functions of $\boldsymbol{t}$, and

$$\hat{M}_i(\underline{\boldsymbol{x}}) = \big(\hat{\boldsymbol{A}}_i\boldsymbol{x}_i\big)^{\mathrm{H}}\left(\sum_{j=1}^n \hat{\boldsymbol{B}}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\hat{\boldsymbol{B}}_{ij}^{\mathrm{H}}\right)^{-1}\big(\hat{\boldsymbol{A}}_i\boldsymbol{x}_i\big). \tag{32}$$

Notice that $\hat{M}_i(\underline{\boldsymbol{x}}) \neq M_i(\underline{\boldsymbol{x}})$ in general. Assume also that the lower bound is tight, i.e.,

$$g_o(\underline{\boldsymbol{x}}) = \max_{\boldsymbol{t}} h(\underline{\boldsymbol{x}}, \boldsymbol{t}). \tag{33}$$

The optimal value of the auxiliary variable $\boldsymbol{t}$ depends on what the current $\underline{\boldsymbol{x}}$ is, so it can be written as a function of $\underline{\boldsymbol{x}}$, i.e.,

$$\boldsymbol{t}^\star = \arg\max_{\boldsymbol{t}} h(\underline{\boldsymbol{x}}, \boldsymbol{t}) \triangleq \mathcal{T}(\underline{\boldsymbol{x}}). \tag{34}$$

A natural idea is to optimize $\underline{\boldsymbol{x}}$ and $\boldsymbol{t}$ alternatingly in the lower bound $h(\underline{\boldsymbol{x}}, \boldsymbol{t})$. In particular, since the optimization of $\underline{\boldsymbol{x}}$ in $h(\underline{\boldsymbol{x}}, \boldsymbol{t})$ under fixed $\boldsymbol{t}$ is a sum-of-weighed-ratios FP problem, the various quadratic transform methods can be readily applied.

The key observation is that updating $\boldsymbol{t}$ in (34) amounts to constructing a surrogate function for $g_o(\underline{\boldsymbol{x}})$; recall that the quadratic transform is also to construct a surrogate function.

Algorithm 4 Generalized Quadratic Transform

```
1: Initialize $\underline{\boldsymbol{x}}$ to a feasible value.
2: repeat
3:     Update $\boldsymbol{t}$ according to (34).
4:     Update $\underline{\boldsymbol{x}}$ by applying one iteration of Algorithm 1, or
       Algorithm 2, or Algorithm 3 to (31) for fixed $\boldsymbol{t}$.
5: until the value of $f_o(\underline{\boldsymbol{x}})$ converges
```

Thus, updating t followed by one iteration of the quadratic transform method can be recognized as constructing a surrogate function of $g_o(\underline{\boldsymbol{x}})$ for the current $\underline{\boldsymbol{x}}$. In other words, after updating $\boldsymbol{t}$, we do not need to run the quadratic transform method (Algorithm 1, or Algorithm 2, or Algorithm 3) with the convergence reached in full; rather, we can just run one iteration of the quadratic transform method, and its convergence is guaranteed by the MM theory. Algorithm 4 summarizes the above method.

An interesting fact is that the connection with the gradient projection carries over to Algorithm 4, as stated in the following proposition.

Proposition 5: If Algorithm 2 is used in step 4 of Algorithm 4, then Algorithm 4 is equivalent to a gradient projection, i.e.,

$$\boldsymbol{x}_i^k = \mathcal{P}_{\mathcal{X}_i}\left(\boldsymbol{x}_i^{k-1} + c_i^k \cdot \frac{\partial g_o(\underline{\boldsymbol{x}}^{k-1})}{\partial \boldsymbol{x}_i^c}\right), \tag{35}$$

where $k = 1, 2, \ldots$ is the iteration index for the loop in Algorithm 4, and $c_i^k > 0$ is the stepsize.

Proof: See Appendix B. ■

We now illustrate the result of Proposition 5 through a concrete example which was first proposed in [1] for the WSR problem. Consider the logarithmic FP problem

$$\underset{\underline{\boldsymbol{x}}}{\text{maximize}} \quad \sum_{i=1}^n \mu_i \log\big(1 + M_i(\underline{\boldsymbol{x}})\big) \tag{36a}$$

$$\text{subject to} \quad \|\boldsymbol{x}_i\|_2^2 \le \rho, \text{ for } i = 1, \ldots, n, \tag{36b}$$

where each weight $\mu_i > 0$. The above type of problem is extensively considered in the communication field.

By the Lagrangian dual transform [7], we can construct the surrogate function $h(\underline{\boldsymbol{x}}, \boldsymbol{t})$ as

$$h(\underline{\boldsymbol{x}}, \boldsymbol{t}) = \sum_{i=1}^n \Big[\mu_i(1+t_i) \cdot \hat{M}_i(\underline{\boldsymbol{x}})\Big] + \sum_{i=1}^n \Big[\mu_i \log(1+t_i) - \mu_i t_i\Big], \tag{37}$$

where

$$\hat{M}_i(\underline{\boldsymbol{x}}) = \big(\boldsymbol{A}_i\boldsymbol{x}_i\big)^{\mathrm{H}}\left(\boldsymbol{A}_i\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}} + \sum_{j=1}^n \boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}_i\boldsymbol{x}_i\big). \tag{38}$$

The corresponding new objective function $g_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t})$ by the nonhomogeneous quadratic transform is

$$\begin{aligned} g_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t}) = \sum_{i=1}^n \Big[2\Re\big\{\mu_i(1+t_i)\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i \\ + \boldsymbol{x}_i^{\mathrm{H}}(\hat{\lambda}_i\boldsymbol{I} - \hat{\boldsymbol{D}}_i)\boldsymbol{z}_i\big\} + \boldsymbol{z}_i^{\mathrm{H}}(\hat{\boldsymbol{D}}_i - \hat{\lambda}_i\boldsymbol{I})\boldsymbol{z}_i - \hat{\lambda}_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{x}_i\Big] \\ + \sum_{i=1}^n \Big[\mu_i \log(1+t_i) - \mu_i t_i\Big], \end{aligned} \tag{39}$$

where

$$\hat{\boldsymbol{D}}_i = \mu_i(1+t_i)\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i\boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{A}_i + \sum_{j=1}^n \mu_j(1+t_j)\boldsymbol{B}_{ji}^{\mathrm{H}}\boldsymbol{y}_j\boldsymbol{y}_j^{\mathrm{H}}\boldsymbol{B}_{ji}.$$

The variables of $g_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t})$ in (39) are optimized iteratively. First, by Lemma 2, each $\boldsymbol{z}_i$ is optimally determined as

$$\boldsymbol{z}_i^\star = \boldsymbol{x}_i. \tag{40}$$

By completing the square for each $\boldsymbol{y}_i$ in (39), we obtain the optimal $\boldsymbol{y}_i$ as

$$\boldsymbol{y}_i^\star = \left(\boldsymbol{A}_i\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}} + \sum_{i=1}^n \boldsymbol{B}_{ij}\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}\boldsymbol{A}_i\boldsymbol{x}_i. \tag{41}$$

With the above $\boldsymbol{z}_i^\star$ and $\boldsymbol{y}_i^\star$ substituted in (39), each optimal t_i can be obtained as

$$t_i^\star = M_i(\underline{\boldsymbol{x}}). \tag{42}$$

Moreover, when $(\underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t})$ are all held fixed, we complete the square for each $\boldsymbol{x}_i$ in $g_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t})$ and solve for $\boldsymbol{x}_i$ as

$$\boldsymbol{x}_i^\star = \begin{cases} \hat{\boldsymbol{x}}_i & \text{if } \|\hat{\boldsymbol{x}}_i\|_2^2 \le \rho; \\ \big(\sqrt{\rho}/\|\hat{\boldsymbol{x}}_i\|_2\big)\hat{\boldsymbol{x}}_i & \text{otherwise,} \end{cases} \tag{43}$$

where

$$\hat{\boldsymbol{x}}_i = \boldsymbol{z}_i + \frac{1}{\hat{\lambda}_i}\Big(\mu_i(1+t_i)\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{y}_i - \hat{\boldsymbol{D}}_i\boldsymbol{z}_i\Big). \tag{44}$$

Because $h(\underline{\boldsymbol{x}}, \boldsymbol{t})$ in (37) meets the condition in Proposition 5, we have the following claim:

Corollary 1: For the logarithmic FP problem (36), updating $\underline{\boldsymbol{z}}$, $\underline{\boldsymbol{y}}$, $\boldsymbol{t}$, and $\underline{\boldsymbol{x}}$ respectively by (40), (41), (42), and (43) in an iterative fashion is equivalent to the gradient projection:

$$\boldsymbol{x}_i^k = \mathcal{P}_{\|\boldsymbol{x}_i\|_2^2 \le \rho}\left(\boldsymbol{x}_i^{k-1} + \frac{1}{\hat{\lambda}_i} \cdot \frac{\partial g_o(\underline{\boldsymbol{x}}^{k-1})}{\partial \boldsymbol{x}_i^c}\right),$$

where $g_o(\underline{\boldsymbol{x}}) = \sum_{i=1}^n \mu_i \log\big(1 + M_i(\underline{\boldsymbol{x}})\big)$.

Remark 3: In the above example, we treat each $\mu_j(1+t_j)$ in (37) as the weight of the ratio $\hat{M}_i(\underline{\boldsymbol{x}})$, and then applying the conventional quadratic transform gives rise to the WMMSE algorithm [8], [9]. Alternatively, we could have let $\hat{M}_i(\underline{\boldsymbol{x}})$ absorb $\mu_i(1+t_i)$ and then treat $\mu_i(1+t_i)\hat{M}_i(\underline{\boldsymbol{x}})$ as the ratio term with weight one; [7] shows the above type of ratio term is more suited for the discrete FP solving. In fact, there are infinitely many ways of deciding which part is the ratio term and which part is the weight. The resulting quadratic transform method can be accelerated as in Algorithm 3, regardless.

VII. Matrix Ratio Case

This section extends the preceding results to the generalized matrix ratios with the matrix variables $\boldsymbol{X}_i \in \mathbb{C}^{d\times m}$ as in (4). The sum-of-weighted-ratios FP problem in (3) now becomes

$$\begin{aligned} &\underset{\underline{\boldsymbol{X}}}{\text{maximize}} && \sum_{i=1}^{n}\Big[\omega_i \mathrm{tr}\big(\boldsymbol{M}_i(\underline{\boldsymbol{X}})\big)\Big] && \text{(45a)}\\ &\text{subject to} && \boldsymbol{X}_i \in \mathcal{X}_i,\ \text{for } i=1,\ldots,n. && \text{(45b)}\end{aligned}$$

The new objective function $f_q(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}})$ by the quadratic transform is shown in (46), where an auxiliary variable $\boldsymbol{Y}_i \in \mathbb{C}^{d\times M}$ is introduced for each matrix ratio $\boldsymbol{M}_i(\underline{\boldsymbol{X}})$.

Optimizing $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Y}}$ alternatingly in $f_q(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}})$ leads us to the matrix-ratio version of Algorithm 1, wherein

$$\boldsymbol{Y}_i^\star = \left(\sum_{i=1}^{n} \boldsymbol{B}_{ij}\boldsymbol{X}_j\boldsymbol{X}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)^{-1}\big(\boldsymbol{A}_i\boldsymbol{X}_i\big) \tag{47}$$

and

$$\boldsymbol{X}_i^\star = \arg\min_{\boldsymbol{X}_i\in\mathcal{X}_i} \big\|\boldsymbol{D}_i^{\frac{1}{2}}\big(\boldsymbol{X}_i - \omega_i\boldsymbol{D}_i^{-1}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{Y}_i\big)\big\|_F \tag{48}$$

with

$$\boldsymbol{D}_i = \sum_{j=1}^{n}\omega_j\boldsymbol{B}_{ji}^{\mathrm{H}}\boldsymbol{Y}_j\boldsymbol{Y}_j^{\mathrm{H}}\boldsymbol{B}_{ji}. \tag{49}$$

We further extend the new objective function $f_t(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}},\underline{\boldsymbol{Z}})$ of the nonhomogeneous quadratic transform as shown in (50), with an auxiliary variable $\boldsymbol{Z}_i \in \mathbb{C}^{d\times m}$ introduced for each $\boldsymbol{D}_i$. Again, we optimize the variables of $f_t(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}},\underline{\boldsymbol{Z}})$ in an iterative fashion: $\underline{\boldsymbol{Z}}$ is optimally updated to $\underline{\boldsymbol{X}}$, $\underline{\boldsymbol{Y}}$ is optimally updated as in (47), and $\underline{\boldsymbol{X}}$ is optimally updated as

$$\boldsymbol{X}_i^\star = \mathcal{P}_{\mathcal{X}_i}\left(\boldsymbol{Z}_i + \frac{1}{\lambda_i}\Big(\omega_i\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{Y}_i - \boldsymbol{D}_i\boldsymbol{Z}_i\Big)\right).$$

Combining the above steps gives the matrix-ratio version of Algorithm 2. Its connection with the gradient projection continues to hold:

$$\boldsymbol{X}^k = \mathcal{P}_{\mathcal{X}_i}\left(\boldsymbol{X}_i^{k-1} + \frac{1}{2\lambda_i^k}\cdot\frac{\partial f_o(\underline{\boldsymbol{X}}^{k-1})}{\partial\boldsymbol{X}_i^c}\right). \tag{51}$$

Equipped with (51), Algorithm 3 can be immediately extended to the matrix ratio case as well.

VIII. Two Application Cases

A. ISAC

Consider two base-stations (BSs) as depicted in Fig. 2. BS 1 performs ISAC while BS 2 only performs downlink transmission. The two BSs have M transmit antennas each, the two downlink users have N antennas each, and BS 1 has N_r radar receive antennas. Denote by $\boldsymbol{H}_{ij} \in \mathbb{C}^{N\times M}$ the channel from BS j to downlink user i, where $i,j\in\{1,2\}$, $\boldsymbol{G}\in\mathbb{C}^{N_r\times M}$ the channel from BS 2 to the radar receiver at BS 1, σ_i^2 the background noise power at downlink user i, and σ_r^2 the background noise power at BS 1. Let $\boldsymbol{v}_i \in \mathbb{C}^M$ be the transmit precoder at BS i subject to the power constraint $P_{\max}$, i.e., $\|\boldsymbol{v}_i\|_2^2 \le P_{\max}$.

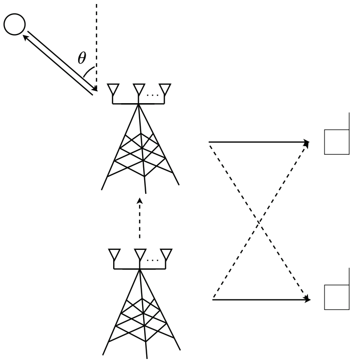


Fig. 2. Two BSs serve one downlink user each. One BS performs ISAC while the other BS only performs transmission; the aim of sensing is to recover the angle θ. The dashed arrows represent the interference.

Moreover, for BS 1, consider the transmit steering vector $\boldsymbol{a}_t(\theta)\in\mathbb{C}^M$ and the receive steering vector $\boldsymbol{a}_r(\theta)\in\mathbb{C}^{N_r}$, both dependent on the target angle θ as shown in Fig. 2:

$$\begin{aligned}\boldsymbol{a}_t(\theta) &= \big[1, e^{-j\pi\sin\theta},\ldots,e^{-j\pi(M-1)\sin\theta}\big]^\top,\\ \boldsymbol{a}_r(\theta) &= \big[1, e^{-j\pi\sin\theta},\ldots,e^{-j\pi(N_r-1)\sin\theta}\big]^\top.\end{aligned}$$

Thus, for the complex Gaussian symbol $\boldsymbol{s}_i \sim \mathcal{CN}(\boldsymbol{0},\boldsymbol{I})$ from BS $i\in\{1,2\}$, the received echo signal at BS 1 is given by

$$r = \xi\boldsymbol{a}_r(\theta)\boldsymbol{a}_t(\theta)^\top\boldsymbol{v}_1\boldsymbol{s}_1 + \boldsymbol{G}\boldsymbol{v}_2\boldsymbol{s}_2 + \boldsymbol{z}, \tag{52}$$

where $\xi\in\mathbb{C}$ is the reflection coefficient, and $\boldsymbol{z}\sim\mathcal{CN}(\boldsymbol{0},\sigma_r^2\boldsymbol{I})$ is the background noise at BS 1. Let $\hat{\boldsymbol{F}} = \boldsymbol{G}\boldsymbol{v}_2\boldsymbol{s}_2 + \boldsymbol{z}$ and define the interference-plus-noise covariance matrix to be

$$\boldsymbol{Q} = \mathbb{E}[\hat{\boldsymbol{F}}\hat{\boldsymbol{F}}^{\mathrm{H}}] = \sigma_r^2\boldsymbol{I} + \boldsymbol{G}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{G}^{\mathrm{H}}. \tag{53}$$

The Fisher information about the target angle θ in Fig. 2 is then computed as

$$J_\theta = \alpha\boldsymbol{v}_1^{\mathrm{H}}\dot{\boldsymbol{A}}^{\mathrm{H}}\boldsymbol{Q}^{-1}\dot{\boldsymbol{A}}\boldsymbol{v}_1, \tag{54}$$

where $\boldsymbol{A} = \boldsymbol{a}_r(\theta)\boldsymbol{a}_t(\theta)^\top$, $\dot{\boldsymbol{A}} = \partial\boldsymbol{A}/\partial\theta$, and $\alpha = 2|\xi|^2$. The

$$f_q(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}}) = \sum_{i=1}^{n}\left[\omega_i\cdot\mathrm{tr}\left(\boldsymbol{X}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{Y}_i + \boldsymbol{Y}_i^{\mathrm{H}}\boldsymbol{A}_i\boldsymbol{X}_i - \boldsymbol{Y}_i^{\mathrm{H}}\left(\sum_{j=1}^{n}\boldsymbol{B}_{ij}\boldsymbol{X}_j\boldsymbol{X}_j^{\mathrm{H}}\boldsymbol{B}_{ij}^{\mathrm{H}}\right)\boldsymbol{Y}_i\right)\right] \tag{46}$$

$$f_t(\underline{\boldsymbol{X}},\underline{\boldsymbol{Y}},\underline{\boldsymbol{Z}}) = \sum_{i=1}^{n}\left[\mathrm{tr}\Big(\omega_i\boldsymbol{X}_i^{\mathrm{H}}\boldsymbol{A}_i^{\mathrm{H}}\boldsymbol{Y}_i + \omega_i\boldsymbol{Y}_i^{\mathrm{H}}\boldsymbol{A}_i\boldsymbol{X}_i + 2\boldsymbol{X}_i^{\mathrm{H}}(\lambda_i\boldsymbol{I}-\boldsymbol{D}_i)\boldsymbol{Z}_i + \boldsymbol{Z}_i^{\mathrm{H}}(\boldsymbol{D}_i-\lambda_i\boldsymbol{I})\boldsymbol{Z}_i - \lambda_i\boldsymbol{X}_i^{\mathrm{H}}\boldsymbol{X}_i\Big)\right] \tag{50}$$

SINRs of the two downlink users are given by

$$\mathrm{SINR}_1 = \boldsymbol{v}_1^{\mathrm{H}}\boldsymbol{H}_{11}^{\mathrm{H}}\left(\sigma_1^2\boldsymbol{I}+\boldsymbol{H}_{12}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{H}_{12}^{\mathrm{H}}\right)^{-1}\boldsymbol{H}_{11}\boldsymbol{v}_1, \tag{55}$$

$$\mathrm{SINR}_2 = \boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{H}_{22}^{\mathrm{H}}\left(\sigma_2^2\boldsymbol{I}+\boldsymbol{H}_{21}\boldsymbol{v}_1\boldsymbol{v}_1^{\mathrm{H}}\boldsymbol{H}_{21}^{\mathrm{H}}\right)^{-1}\boldsymbol{H}_{22}\boldsymbol{v}_2. \tag{56}$$

We seek the optimal precoders $\underline{\boldsymbol{v}}=\{\boldsymbol{v}_1,\boldsymbol{v}_2\}$ to maximize a linear combination of the Fisher information (for the sensing purpose) and the two SINRs (for the communication purpose):

$$\underset{\underline{\boldsymbol{v}}}{\text{maximize}} \quad J_\theta+\omega_1\mathrm{SINR}_1+\omega_2\mathrm{SINR}_2 \tag{57a}$$

$$\text{subject to} \quad \|\boldsymbol{v}_i\|_2^2\le P_{\max},\;\; i\in\{1,2\}, \tag{57b}$$

where $\omega_i>0$ reflects the priority of SINR_i as compared to J_θ in the ISAC task.

By the conventional quadratic transform, the original objective function can be recast to

$$\begin{aligned} f_q(\underline{\boldsymbol{v}},\underline{\boldsymbol{y}}) &= 2\Re\{\alpha\boldsymbol{v}_1^{\mathrm{H}}\dot{\boldsymbol{A}}^{\mathrm{H}}\boldsymbol{y}_r\}-\alpha\boldsymbol{y}_r^{\mathrm{H}}\left(\sigma_r^2\boldsymbol{I}+\boldsymbol{G}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{G}^{\mathrm{H}}\right)\boldsymbol{y}_r\\ &+2\Re\{\omega_1\boldsymbol{v}_1^{\mathrm{H}}\boldsymbol{H}_{11}^{\mathrm{H}}\boldsymbol{y}_1\}-\omega_1\boldsymbol{y}_1^{\mathrm{H}}\left(\sigma_1^2\boldsymbol{I}+\boldsymbol{H}_{12}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{H}_{12}^{\mathrm{H}}\right)\boldsymbol{y}_1\\ &+2\Re\{\omega_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{H}_{22}^{\mathrm{H}}\boldsymbol{y}_2\}-\omega_2\boldsymbol{y}_2^{\mathrm{H}}\left(\sigma_2^2\boldsymbol{I}+\boldsymbol{H}_{21}\boldsymbol{v}_1\boldsymbol{v}_1^{\mathrm{H}}\boldsymbol{H}_{21}^{\mathrm{H}}\right)\boldsymbol{y}_2, \end{aligned} \tag{58}$$

where the auxiliary variables $\boldsymbol{y}_r\in\mathbb{C}^{N_r}$, $\boldsymbol{y}_1\in\mathbb{C}^{N}$, and $\boldsymbol{y}_2\in\mathbb{C}^{N}$ are introduced for J_θ, SINR_1, and SINR_2, respectively. We optimize the precoders $\underline{\boldsymbol{v}}$ and the auxiliary variables $\underline{\boldsymbol{y}}=(\boldsymbol{y}_r,\boldsymbol{y}_1,\boldsymbol{y}_2)$ alternatingly as

$$\underline{\boldsymbol{v}}^0\to\cdots\to\underline{\boldsymbol{v}}^{k-1}\to\underline{\boldsymbol{y}}^k\to\underline{\boldsymbol{v}}^k\to\cdots.$$

When $\underline{\boldsymbol{v}}$ is held fixed, all the auxiliary variables can be optimally determined for $f_q(\underline{\boldsymbol{v}},\underline{\boldsymbol{y}})$ in closed form as

$$\boldsymbol{y}_r^\star=\left(\sigma_r^2\boldsymbol{I}+\boldsymbol{G}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{G}^{\mathrm{H}}\right)^{-1}\dot{\boldsymbol{A}}\boldsymbol{v}_1, \tag{59a}$$

$$\boldsymbol{y}_1^\star=\left(\sigma_1^2\boldsymbol{I}+\boldsymbol{H}_{12}\boldsymbol{v}_2\boldsymbol{v}_2^{\mathrm{H}}\boldsymbol{H}_{12}^{\mathrm{H}}\right)^{-1}\boldsymbol{H}_{11}\boldsymbol{v}_1, \tag{59b}$$

$$\boldsymbol{y}_2^\star=\left(\sigma_2^2\boldsymbol{I}+\boldsymbol{H}_{21}\boldsymbol{v}_1\boldsymbol{v}_1^{\mathrm{H}}\boldsymbol{H}_{21}^{\mathrm{H}}\right)^{-1}\boldsymbol{H}_{22}\boldsymbol{v}_2. \tag{59c}$$

After the update of the auxiliary variables, we find the optimal $\underline{\boldsymbol{v}}$ in closed form as

$$\boldsymbol{v}_1^\star=\left(\eta_1\boldsymbol{I}+\boldsymbol{D}_1\right)^{-1}\left(\alpha\dot{\boldsymbol{A}}^H\boldsymbol{y}_r+\omega_1\boldsymbol{H}_{11}^{\mathrm{H}}\boldsymbol{y}_1\right), \tag{60a}$$

$$\boldsymbol{v}_2^\star=\left(\eta_2\boldsymbol{I}+\boldsymbol{D}_2\right)^{-1}\omega_2\boldsymbol{H}_{22}^{\mathrm{H}}\boldsymbol{y}_2, \tag{60b}$$

where

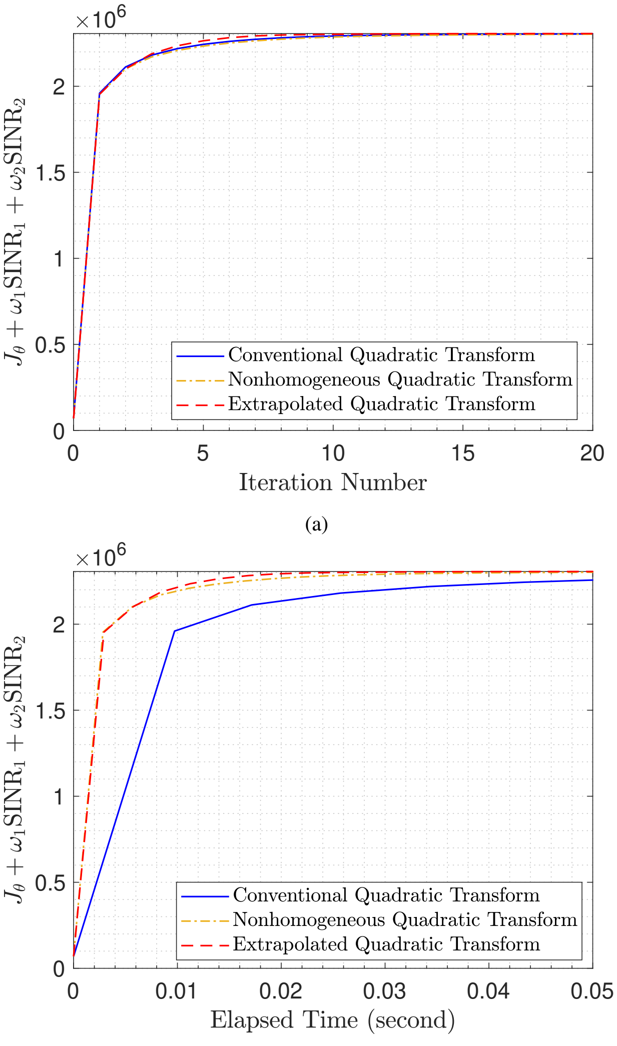
$$\boldsymbol{D}_1=\omega_2\boldsymbol{H}_{21}^{\mathrm{H}}\boldsymbol{y}_2\boldsymbol{y}_2^{\mathrm{H}}\boldsymbol{H}_{21}, \tag{61a}$$

$$\boldsymbol{D}_2=\alpha\boldsymbol{G}^{\mathrm{H}}\boldsymbol{y}_r\boldsymbol{y}_r^{\mathrm{H}}\boldsymbol{G}+\omega_1\boldsymbol{H}_{12}^{\mathrm{H}}\boldsymbol{y}_1\boldsymbol{y}_1^{\mathrm{H}}\boldsymbol{H}_{12}, \tag{61b}$$

and the Lagrange multipliers (η_1,η_2) for the power constraint are optimally determined as

$$\eta_i^\star=\min\left\{\eta_i\ge0:\|\boldsymbol{v}_i\|_2^2\le P_{\max}\right\}. \tag{62}$$

To implement (62) in practice, we may first try out $\eta_i^\star=0$ to see if $\|\boldsymbol{v}_i\|_2^2\le P_{\max}$; if not, then we further tune $\eta^\star$ via bisection search to render $\|\boldsymbol{v}_i\|_2^2=P_{\max}$.



(a)

(b)

Fig. 3. Maximizing a weighted sum of Fisher information and SINRs in the ISAC problem. Figure (a) shows the convergence in iterations, while figure (b) shows the convergence in time.

Differing from the above conventional quadratic transform, the nonhomogeneous quadratic transform recasts the original objective function (57a) to $f_t(\underline{\boldsymbol{v}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}})$ as shown in (63). Again, we optimize the variables in $f_t(\underline{\boldsymbol{v}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}})$ iteratively as

$$\underline{\boldsymbol{v}}^0\to\cdots\to\underline{\boldsymbol{v}}^{k-1}\to\underline{\boldsymbol{z}}^k\to\underline{\boldsymbol{y}}^k\to\underline{\boldsymbol{v}}^k\to\underline{\boldsymbol{z}}^{k+1}\to\cdots.$$

When $\underline{\boldsymbol{v}}$ and $\underline{\boldsymbol{y}}$ are both held fixed, $\underline{\boldsymbol{z}}$ is optimally updated

$$\begin{aligned} f_t(\underline{\boldsymbol{v}},\underline{\boldsymbol{y}},\underline{\boldsymbol{z}}) = 2\Re\Big\{&\boldsymbol{v}_1^{\mathrm{H}}\Big[\alpha\dot{\boldsymbol{A}}^{\mathrm{H}}\boldsymbol{y}_r+\omega_1\boldsymbol{H}_{11}^{\mathrm{H}}\boldsymbol{y}_1+(\lambda_1\boldsymbol{I}-\boldsymbol{D}_1)\boldsymbol{z}_1\Big]+\boldsymbol{v}_2^{\mathrm{H}}\Big[\omega_2\boldsymbol{H}_{22}^{\mathrm{H}}\boldsymbol{y}_2+(\lambda_2\boldsymbol{I}-\boldsymbol{D}_2)\boldsymbol{z}_2\Big]\Big\}\\ &\boldsymbol{z}_1^{\mathrm{H}}(\boldsymbol{D}_1-\lambda_1\boldsymbol{I})\boldsymbol{z}_1+\boldsymbol{z}_2^{\mathrm{H}}(\boldsymbol{D}_2-\lambda_2\boldsymbol{I})\boldsymbol{z}_2-\lambda_1\|\boldsymbol{v}_1\|_2^2-\lambda_2\|\boldsymbol{v}_2\|_2^2-\alpha\sigma_r^2\|\boldsymbol{y}_r\|_2^2-\omega_1\sigma_1^2\|\boldsymbol{y}_1\|_2^2-\omega_2\sigma_2^2\|\boldsymbol{y}_2\|_2^2 \end{aligned} \tag{63}$$

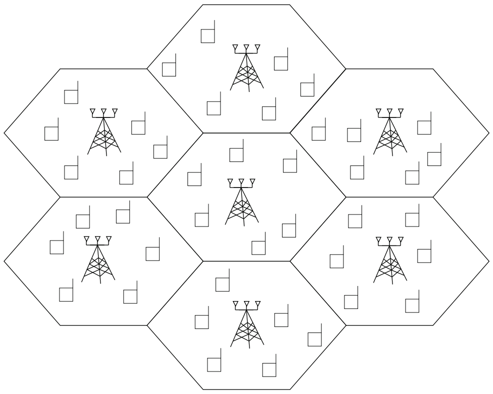


Fig. 4. A 7-cell wrapped-around massive MIMO network.

as $\boldsymbol{z}_1^\star = \boldsymbol{v}_1$, and $\boldsymbol{z}_2^\star = \boldsymbol{v}_2$. The optimal update of $\underline{\boldsymbol{y}}$ is the same as in (59a), (59b), and (59c). When $\underline{\boldsymbol{y}}$ and $\underline{\boldsymbol{z}}$ are both held fixed, we first compute

$$\hat{\boldsymbol{v}}_1 = \boldsymbol{z}_1 + \frac{1}{\lambda_1}\Big(\alpha \dot{\boldsymbol{A}}^{\mathrm{H}}\boldsymbol{y}_r + \omega_1 \boldsymbol{H}_{11}^{\mathrm{H}}\boldsymbol{y}_1 - \boldsymbol{D}_1\boldsymbol{z}_1\Big) \tag{64}$$

and

$$\hat{\boldsymbol{v}}_2 = \boldsymbol{z}_2 + \frac{1}{\lambda_2}\Big(\omega_2 \boldsymbol{H}_{22}^{\mathrm{H}}\boldsymbol{y}_2 - \boldsymbol{D}_2\boldsymbol{z}_2\Big), \tag{65}$$

and then update $\underline{\boldsymbol{v}}$ optimally as

$$\boldsymbol{v}_i^\star = \begin{cases} \hat{\boldsymbol{v}}_i & \text{if } \|\hat{\boldsymbol{v}}_i\|_2^2 \le P_{\max}; \\ \big(\sqrt{P_{\max}}/\|\hat{\boldsymbol{v}}_i\|_2\big)\hat{\boldsymbol{v}}_i & \text{otherwise.} \end{cases} \tag{66}$$

We validate the performance of the various quadratic transform methods in Fig. 3. In our simulation case, $M = 64$, $N = 2$, $N_r = 72$, $\omega_1 = \omega_2 = 10^5$, $\sigma_1^2 = \sigma_2^2 = -80$ dBm, $\sigma_r^2 = -80$ dBm, and $P_{\max} = 20$ dBm. The path loss (in dB) is computed as $32.6 + 36.7\log_{10} d$, where d is the distance in meters; the position coordinates of BS 1, BS 2, user 1, user 2, and the sensed object are $(0,0)$, $(250,0)$, $(-10,100)$, $(350,100)$, and $(200,200)$, respectively, all in meters. The Rayleigh fading model is adopted. Algorithm 1, Algorithm 2, and Algorithm 3 are tested. As shown in Fig. 2(a), if the convergence is considered in terms of iterations, then all these algorithms yield almost the same convergence rate. The objective value is monotonically increasing with the iteration number by all these algorithms. If we instead evaluate convergence in terms of the elapsed time as displayed in Fig. 2(b), then the proposed two accelerated methods, Algorithm 2 and Algorithm 3, become much faster than Algorithm 1; the former two algorithms attain convergence after 0.02 seconds, whereas the latter algorithm still does not converge after 0.05 seconds. Algorithm 3 outperforms Algorithm 2, but their gap is marginal.

B. Massive MIMO

The application case of massive MIMO is closely related to the example stated in Section VI. Consider a downlink multi-cell network with L cells as depicted in Fig. 4. In each cell, one BS with M antennas sends independent messages towards Q downlink user terminals simultaneously by spatial multiplexing; it shall be well understood that $Q \le M$. Assume also that each user terminal has N receive antennas. In particular, $M \gg N$ under the massive MIMO setting.

Moreover, we use $\ell, i = 1,\ldots,L$ to index the cells and the corresponding BSs, and use $q, j = 1,\ldots,Q$ to index the users in each cell. Denote by $\boldsymbol{H}_{\ell q,i} \in \mathbb{C}^{N\times M}$ the channel from BS i to the qth user in cell ℓ, denote by $\boldsymbol{v}_{\ell q} \in \mathbb{C}^M$ the transmit precoder of BS ℓ for its qth associated user, and denote by σ^2 the background noise power. The SINR of the qth user in cell ℓ, denoted by $\mathrm{SINR}_{\ell q}$, is computed[3] as

$$\mathrm{SINR}_{\ell q} = \boldsymbol{v}_{\ell q}^{\mathrm{H}}\boldsymbol{H}_{\ell q,\ell}^{\mathrm{H}}\Bigg(\sigma^2\boldsymbol{I} + \sum_{(i,j)\neq(\ell,q)} \boldsymbol{H}_{\ell q,i}\boldsymbol{v}_{ij}\boldsymbol{v}_{ij}^{\mathrm{H}}\boldsymbol{H}_{\ell q,i}^{\mathrm{H}}\Bigg) \boldsymbol{H}_{\ell q,\ell}\boldsymbol{v}_{\ell q}. \tag{67}$$

Assigning a positive weight $\mu_{\ell q} > 0$ for each user q in cell ℓ, we seek the optimal set of precoding vectors $\underline{\boldsymbol{v}} = \{\boldsymbol{v}_{\ell q}\}$ to maximize the weighted sum-of-rates throughout the network:

$$\underset{\underline{\boldsymbol{v}}}{\text{maximize}} \quad \sum_{\ell=1}^{L}\sum_{q=1}^{Q}\mu_{\ell q}\log\big(1+\mathrm{SINR}_{\ell q}\big) \tag{68a}$$

$$\text{subject to} \quad \sum_{q=1}^{Q}\|\boldsymbol{v}_{\ell q}\|_2^2 \le P_{\max},\ \text{for } \ell = 1,\ldots,L, \tag{68b}$$

where the constraint (68b) states that the total transmit power at each BS cannot exceed the power budget $P_{\max}$.

The traditional WMMSE method [8], [9] addresses the above problem by performing the following iterative updates:

$$\underline{\boldsymbol{v}}^0 \to \cdots \to \underline{\boldsymbol{v}}^{k-1} \to \underline{\boldsymbol{y}}^k \to \boldsymbol{t}^k \to \underline{\boldsymbol{v}}^k \to \cdots,$$

where the auxiliary variable $\boldsymbol{t}$ is updated as

$$t_{\ell q}^\star = \mathrm{SINR}_{\ell q} \tag{69}$$

for the current $\underline{\boldsymbol{v}}$, and the auxiliary variable $\underline{\boldsymbol{y}}$ is updated as

$$\boldsymbol{y}_{\ell q}^\star = \Bigg(\sigma^2\boldsymbol{I} + \sum_{i=1}^{L}\sum_{j=1}^{Q}\boldsymbol{H}_{\ell q,i}\boldsymbol{v}_{ij}\boldsymbol{v}_{ij}^{\mathrm{H}}\boldsymbol{H}_{\ell q,i}^{\mathrm{H}}\Bigg)^{-1}\boldsymbol{H}_{\ell q,\ell}\boldsymbol{v}_{\ell q}. \tag{70}$$

With the auxiliary variables held fixed, the precoding vectors are optimally updated as

$$\boldsymbol{v}_{\ell q}(\eta_\ell) = \Bigg(\eta_\ell\boldsymbol{I} + \sum_{i=1}^{L}\sum_{j=1}^{Q}\mu_{ij}(1+t_{ij})\boldsymbol{H}_{ij,\ell}^{\mathrm{H}}\boldsymbol{y}_{ij}\boldsymbol{y}_{ij}^{\mathrm{H}}\boldsymbol{H}_{ij,\ell}\Bigg)^{-1} \mu_{\ell q}(1+t_{\ell q})\boldsymbol{H}_{\ell q,\ell}^{\mathrm{H}}\boldsymbol{y}_{\ell q}, \tag{71}$$

where the Lagrange multiplier η_ℓ accounts for the power constraint at BS ℓ and is optimally determined as

$$\eta_\ell^\star = \min\Bigg\{\eta \ge 0 : \sum_{q=1}^{Q}\|\boldsymbol{v}_{\ell q}(\eta)\|_2^2 \le P_{\max}\Bigg\}. \tag{72}$$

In the practical implementation, the above η_ℓ can be obtained via bisection search; an $M \times M$ matrix inverse needs to be

[3] The SINR in (67) can be achieved by using the MMSE receive beamformer in practice. Moreover, it is worth noticing that the optimal update of $\boldsymbol{y}_{\ell q}$ in (70) turns out to be a scaled MMSE receive beamformer at user q in cell ℓ.

computed in (71) for each bisection search iterate, which can be quite costly because $M \gg N$.

In contrast, the nonhomogeneous quadratic transform reformulates the objective function as

$$g_t(\underline{\boldsymbol{v}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t}) = \sum_{\ell=1}^{L}\sum_{q=1}^{Q}\Big[2\Re\{\mu_{\ell q}(1+t_{\ell q})\boldsymbol{v}_{\ell q}^{\mathrm{H}}\boldsymbol{H}_{\ell q,\ell}^{\mathrm{H}}\boldsymbol{y}_{\ell q} + \boldsymbol{v}_{\ell q}^{\mathrm{H}}(\hat{\lambda}_\ell \boldsymbol{I} - \hat{\boldsymbol{D}}_\ell)\boldsymbol{z}_{\ell q}\} + \boldsymbol{z}_{\ell q}^{\mathrm{H}}(\hat{\boldsymbol{D}}_\ell - \hat{\lambda}_\ell \boldsymbol{I})\boldsymbol{z}_{\ell q} - \hat{\lambda}_\ell \boldsymbol{v}_{\ell q}^{\mathrm{H}}\boldsymbol{v}_{\ell q} - \mu_{\ell q}(1+t_{\ell q})\sigma^2\boldsymbol{y}_{\ell q}^{\mathrm{H}}\boldsymbol{y}_{\ell q} + \mu_{\ell q}\log(1+t_{\ell q}) - \mu_{\ell q}t_{\ell q}\Big], \quad (73)$$

for which the iterative updates are carried out as

$$\underline{\boldsymbol{v}}^0 \to \cdots \to \underline{\boldsymbol{v}}^{k-1} \to \underline{\boldsymbol{z}}^k \to \underline{\boldsymbol{y}}^k \to \boldsymbol{t}^k \to \underline{\boldsymbol{v}}^k \to \cdots,$$

where the auxiliary variable $\underline{\boldsymbol{z}}$ is updated as $\boldsymbol{z}_{iq} = \boldsymbol{v}_{iq}$, and the other two auxiliary variables $\underline{\boldsymbol{y}}$ and $\boldsymbol{t}$ are updated as in (70) and (69), respectively. To update $\underline{\boldsymbol{v}}$, we first compute

$$\hat{\boldsymbol{v}}_{\ell q} = \boldsymbol{z}_{\ell q} + \frac{1}{\hat{\lambda}_\ell}\Big(\mu_{\ell q}(1+t_{\ell q})\boldsymbol{H}_{\ell q,\ell}^{\mathrm{H}}\boldsymbol{y}_{\ell q} - \hat{\boldsymbol{D}}_\ell \boldsymbol{z}_{\ell q}\Big), \quad (74)$$

where

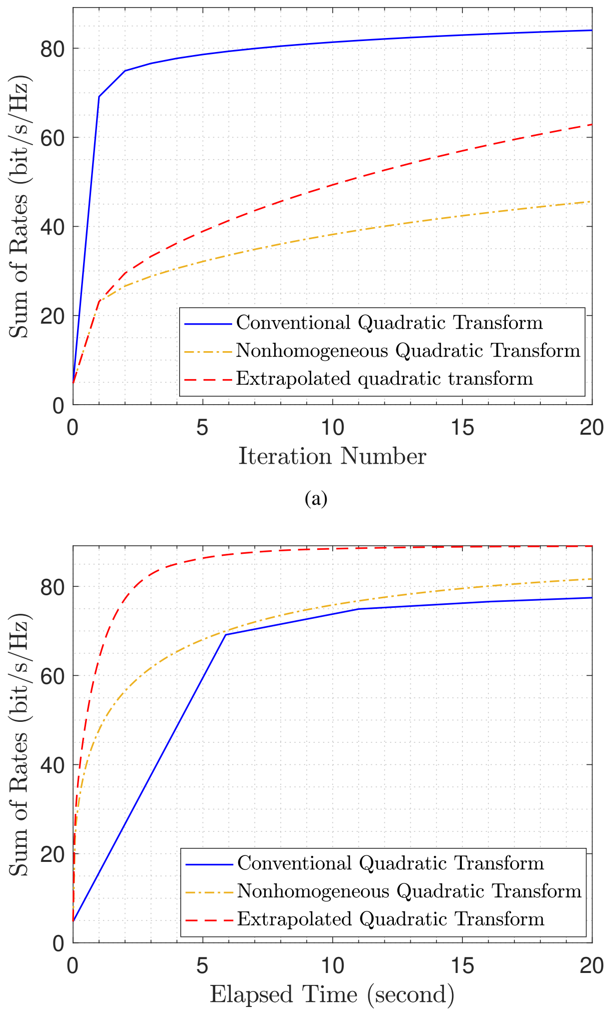
$$\hat{\boldsymbol{D}}_\ell = \sum_{i=1}^{L}\sum_{j=1}^{Q}\mu_{ij}(1+t_{ij})\boldsymbol{H}_{ij,\ell}^{\mathrm{H}}\boldsymbol{y}_{ij}\boldsymbol{y}_{ij}^{\mathrm{H}}\boldsymbol{H}_{ij,\ell}, \quad (75)$$

and then incorporate the power constraint as

$$\boldsymbol{v}_{\ell q}^{\star} = \begin{cases} \hat{\boldsymbol{v}}_{\ell q} & \text{if } \sum_{j=1}^{Q}\|\hat{\boldsymbol{v}}_{\ell j}\|_2^2 \le P_{\max}; \\ \sqrt{\frac{P_{\max}}{\sum_{j=1}^{Q}\|\hat{\boldsymbol{v}}_{\ell j}\|_2^2}}\hat{\boldsymbol{v}}_{\ell q} & \text{otherwise.} \end{cases}$$

As opposed to the updating formula (71) of the WMMSE algorithm, the update of $\underline{\boldsymbol{v}}$ in Algorithm 2 no longer incurs any matrix inverse. Even though Algorithm 2 still requires computing matrix inverse for updating the auxiliary variable $\underline{\boldsymbol{y}}$ as in (70), the matrix size is just $N \times N$ with $N \ll M$ and thus can be neglected. Moreover, the above beamforming method for massive MIMO can be accelerated via extrapolation as in Algorithm 3.

We now test the various quadratic transform methods for massive MIMO in a simulated 7-hexagonal-cell wrapped-around network as considered in [3]. Within each cell, the BS is located at the center and the 6 downlink users are randomly placed. Each BS has 128 antennas and each user has 4 antennas. The BS-to-BS distance is set to be 0.8 km. The maximum transmit power level at the BS side is set to be 20 dBm, and the AWGN power level is set to be -90 dBm. The downlink distance-dependent path-loss is simulated by $128.1 + 37.6\log_{10}(d) + \tau$ (in dB), where d represents the BS-to-user distance in km, and τ is a zero-mean Gaussian random variable with 8 dB standard deviation for the shadowing effect. We consider sum rate maximization by setting all the weights to 1. Again, Algorithm 1, Algorithm 2, and Algorithm 3 are the competitors. As shown in Fig. 5(a), Algorithm 1 converges faster than the other two methods in terms of iterations; this result agrees with the former discussion below Proposition 3. When it comes to the convergence evaluated by time, as shown in Fig. 5(b), the two accelerated quadratic transform methods are much more efficient than the conventional method in Algorithm 1. In particular, observe that Algorithm 3 is also much faster than Algorithm 2, as opposed to the ISAC case in Fig. 2. There are two reasons. First, there are more matrix ratio terms in the massive MIMO problem case; second, the FP of massive MIMO has a more complicated structure (with ratios nested in logarithms). When MPF contains more ratios or has a more complicated structure, the surrogate function approximation by the nonhomogeneous quadratic transform tends to be loose, in which case Nesterov's extrapolation becomes more effective.



(a)

(b)

Fig. 5. Maximizing the sum of rates in a multi-cell downlink massive MIMO network. Figure (a) shows the convergence in iterations, while figure (b) shows the convergence in time.

IX. Conclusion

This work considerably develops the existing theory and algorithm of FP, focusing on their applications in wireless networks. The quadratic transform is a state-of-the-art tool in the FP area. As a starting point, we establish a connection between the quadratic transform and the gradient projection; this connection turns out to be fairly useful in that it enables the iterative optimization to get rid of matrix inverses. We then propose further accelerating the quadratic transform via extrapolation. Of fundamental importance is the convergence rate analysis that follows. To the best of our knowledge, this is the very first work that examines how fast the quadratic transform (including its special case the WMMSE algorithm) converges and also how to render it even faster. Moreover, we demonstrate the practical usefulness of the accelerated quadratic transform through two application cases, ISAC and massive MIMO, both of which are envisioned to be the key components of the next-generation wireless networks.

Appendix A
Proof of Proposition 3

We focus on the convergence rate of Algorithm 1; the convergence rate of Algorithm 2 can be established similarly. Lemma 1.2.4 in [2] states that for any twice-differentiable function $f(\boldsymbol{x})$ with L-Lipschitz continuous gradient, we have

$$\begin{aligned}\Big| f(\boldsymbol{x}') - f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})^{\mathrm{H}}(\boldsymbol{x}'-\boldsymbol{x})&\\ -\frac{1}{2}(\boldsymbol{x}'-\boldsymbol{x})^{\mathrm{H}}\nabla^2 f(\boldsymbol{x})(\boldsymbol{x}'-\boldsymbol{x})\Big| &\le \frac{L}{6}\|\boldsymbol{x}'-\boldsymbol{x}\|_2^3\end{aligned}$$

given any two feasible $\boldsymbol{x}$ and $\boldsymbol{x}'$. Applying the above lemma to the function $\delta_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1})$ and using the results in (22) give

$$\begin{aligned}&\frac{L}{6}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^3\\ &\ge \delta_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1})\\ &\quad -\frac{1}{2}(\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1})^{\mathrm{H}}\nabla^2\delta_q(\underline{\boldsymbol{x}}^{k-1}|\underline{\boldsymbol{x}}^{k-1})(\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1})\\ &\ge \delta_q(\underline{\boldsymbol{x}}|\underline{\boldsymbol{x}}^{k-1}) - \frac{\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2\\ &= f_o(\underline{\boldsymbol{x}}) - f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}}^k) - \frac{\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2\\ &\overset{(a)}{\ge} f_o(\underline{\boldsymbol{x}}) - f_q(\underline{\boldsymbol{x}}^k,\underline{\boldsymbol{y}}^k) - \frac{\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2\\ &\overset{(b)}{\ge} f_o(\underline{\boldsymbol{x}}) - f_q(\underline{\boldsymbol{x}}^k,\underline{\boldsymbol{y}}^{k+1}) - \frac{\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2\\ &\overset{(c)}{=} f_o(\underline{\boldsymbol{x}}) - f_o(\underline{\boldsymbol{x}}^k) - \frac{\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2,\end{aligned}\tag{76}$$

where step (a) follows since $\underline{\boldsymbol{x}}^k$ maximizes $f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}})$ for the current $\underline{\boldsymbol{y}}=\underline{\boldsymbol{y}}^k$, step (b) follows since $\underline{\boldsymbol{y}}^{k+1}$ maximizes $f_q(\underline{\boldsymbol{x}},\underline{\boldsymbol{y}})$ for the current $\underline{\boldsymbol{x}}=\underline{\boldsymbol{x}}^k$, and step (c) follows by the property of the surrogate function. Following Nesterov's proof technique in [2], we let

$$\underline{\boldsymbol{x}} = \pi\underline{\boldsymbol{x}}^* + (1-\pi)\underline{\boldsymbol{x}}^{k-1},\tag{77}$$

where the parameter $\pi\in[0,1]$. Then the concavity of $f_o(\underline{\boldsymbol{x}})$ on $\mathcal{X}$ gives

$$f_o(\underline{\boldsymbol{x}}) \le \pi f_o(\underline{\boldsymbol{x}}^*) + (1-\pi)f_o(\underline{\boldsymbol{x}}^{k-1}).\tag{78}$$

Denote the gap in the objective value as

$$v_k = f_o(\underline{\boldsymbol{x}}^*) - f_o(\underline{\boldsymbol{x}}^k).\tag{79}$$

Substituting (77) and (78) into (76) gives rise to

$$\begin{aligned}v_k &\le (1-\pi)v_{k-1} + \frac{\pi^2\Lambda_q}{2}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^2\\ &\quad + \frac{\pi^3 L}{6}\|\underline{\boldsymbol{x}}-\underline{\boldsymbol{x}}^{k-1}\|_2^3\\ &\le (1-\pi)v_{k-1} + \pi^2\left(\frac{\Lambda_q R^2}{2}+\frac{LR^3}{6}\right),\end{aligned}\tag{80}$$

where the second inequality follows by (21) and $0\le\pi\le1$. The choice of π depends on k.

When $k=1$, we let $\pi=1$ in (80) and obtain

$$v_1 \le \frac{\Lambda_q R^2}{2} + \frac{LR^3}{6}.\tag{81}$$

When $k\ge2$, we let

$$\pi = \frac{v_{k-1}}{\Lambda_q R^2 + LR^3/3}.\tag{82}$$

It can be shown by induction that the above π is always feasible (i.e., $0\le\pi\le1$) for all $k\ge2$. Plugging (82) in (80) yields

$$v_k \le v_{k-1}\left(1-\frac{v_{k-1}}{2\Lambda_q R^2 + 2LR^3/3}\right),\tag{83}$$

which can be further rewritten as

$$\begin{aligned}\frac{1}{v_k} &\ge \frac{1}{v_{k-1}}\cdot\left(1-\frac{v_{k-1}}{2\Lambda_q R^2+2LR^3/3}\right)^{-1}\\ &\ge \frac{1}{v_{k-1}}\cdot\left(1+\frac{v_{k-1}}{2\Lambda_q R^2+2LR^3/3}\right)\\ &= \frac{1}{v_{k-1}} + \frac{1}{2\Lambda_q R^2+2LR^3/3},\end{aligned}\tag{84}$$

where the second inequality follows since $(1-a)^{-1}>1+a$ for any $0\le a\le1$. The result of (84) immediately gives

$$\begin{aligned}\frac{1}{v_k} &\ge \frac{1}{v_1} + \frac{k-1}{2\Lambda_q R^2+2LR^3/3}\\ &\ge \frac{k+3}{2\Lambda_q R^2+2LR^3/3},\end{aligned}\tag{85}$$

where the second inequality is due to (81). The proof is then completed for Algorithm 1. The case of Algorithm 2 can be verified similarly.

Appendix B
Proof of Proposition 5

Because optimizing $\boldsymbol{t}$ in (34) for fixed $\underline{\boldsymbol{x}}$ is an unconstrained differentiable problem, the optimal $\boldsymbol{t}^\star$ must satisfy the first-order condition

$$\frac{d\beta(\boldsymbol{t}^\star)}{d\boldsymbol{t}} + \sum_{j=1}^{n}\left[\frac{d\alpha_j(\boldsymbol{t}^\star)}{d\boldsymbol{t}}\cdot\hat{M}_j(\underline{\boldsymbol{x}})\right] = 0,\tag{86}$$

in light of which the partial derivative of $g_o(\underline{\boldsymbol{x}})$ can be considerably simplified as

$$\begin{aligned}\frac{\partial g_o(\underline{\boldsymbol{x}})}{\partial\boldsymbol{x}_i^c} &= \frac{\partial h(\underline{\boldsymbol{x}},\mathcal{T}(\underline{\boldsymbol{x}}))}{\partial\boldsymbol{x}_i^c}\\ &= \sum_{j=1}^{n}\left[\frac{d\alpha_j(\boldsymbol{t}^\star)}{d\boldsymbol{t}}\frac{\partial\mathcal{T}(\underline{\boldsymbol{x}})}{\partial\boldsymbol{x}_i^c}\hat{M}_j(\underline{\boldsymbol{x}}) + \alpha_j(\boldsymbol{t}^\star)\frac{\partial\hat{M}_j(\underline{\boldsymbol{x}})}{\partial\boldsymbol{x}_i^c}\right]\\ &\quad + \frac{d\beta(\boldsymbol{t}^\star)}{d\boldsymbol{t}}\frac{\partial\mathcal{T}(\underline{\boldsymbol{x}})}{\partial\boldsymbol{x}_i^c}\end{aligned}$$

$$= \sum_{j=1}^{n} \left[\alpha_j(\boldsymbol{t}^{\star}) \frac{\partial \hat{M}_j(\underline{\boldsymbol{x}})}{\partial \boldsymbol{x}_i^c} \right]. \tag{87}$$

We now apply to $h(\underline{\boldsymbol{x}}, \boldsymbol{t})$ the nonhomogeneous quadratic transform in Section III-B, and thus obtain the new objective function

$$g_t(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t}) = \sum_{i=1}^{n} \Big[2\Re\big\{ \alpha_i(\boldsymbol{t}) \boldsymbol{x}_i^{\mathrm{H}} \boldsymbol{A}_i^{\mathrm{H}} \boldsymbol{y}_i + \boldsymbol{x}_i^{\mathrm{H}} (\hat{\lambda}_i \boldsymbol{I} - \hat{\boldsymbol{D}}_i) \boldsymbol{z}_i \big\} \\ + \boldsymbol{z}_i^{\mathrm{H}} (\hat{\boldsymbol{D}}_i - \hat{\lambda}_i \boldsymbol{I}) \boldsymbol{z}_i - \hat{\lambda}_i \boldsymbol{x}_i^{\mathrm{H}} \boldsymbol{x}_i \Big] + \beta(\boldsymbol{t}), \tag{88}$$

where

$$\hat{\boldsymbol{D}}_i = \sum_{j=1}^{n} \alpha_j(\boldsymbol{t}) \hat{\boldsymbol{B}}_{ji}^{\mathrm{H}} \boldsymbol{y}_j \boldsymbol{y}_j^{\mathrm{H}} \hat{\boldsymbol{B}}_{ji} \tag{89}$$

and

$$\hat{\lambda}_i \geq \lambda_{\max}(\hat{\boldsymbol{D}}_i). \tag{90}$$

We optimize the variables $(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}, \underline{\boldsymbol{z}}, \boldsymbol{t})$ iteratively as

$$\underline{\boldsymbol{x}}^0 \to \cdots \to \underline{\boldsymbol{x}}^{k-1} \to \underline{\boldsymbol{z}}^k \to \underline{\boldsymbol{y}}^k \to \boldsymbol{t}^k \to \underline{\boldsymbol{x}}^k \to \cdots .$$

The optimal update of $\boldsymbol{x}_i^k$ is

$$\boldsymbol{x}_i^k \overset{(a)}{=} \mathcal{P}_{\mathcal{X}_i} \left(\boldsymbol{x}_i^{k-1} + \frac{1}{\hat{\lambda}_i^k} \Big(\alpha_i(\boldsymbol{t}^k) \hat{\boldsymbol{A}}_i^{\mathrm{H}} \boldsymbol{y}_i^k - \hat{\boldsymbol{D}}_i^k \boldsymbol{x}_i^{k-1} \Big) \right) \\ \overset{(b)}{=} \mathcal{P}_{\mathcal{X}_i} \left(\boldsymbol{x}_i^{k-1} + \frac{1}{2\hat{\lambda}_i^k} \sum_{j=1}^{n} \left[\alpha_j(\boldsymbol{t}^k) \frac{\partial \hat{M}_j(\underline{\boldsymbol{x}}^{k-1})}{\partial \boldsymbol{x}_i^c} \right] \right), \tag{91}$$

where step (a) follows since each $\boldsymbol{z}_i^k$ has been updated to $\boldsymbol{x}_i^{k-1}$, and step (b) follows by Lemma 1. Substituting (87) in the above equation completes the proof.

References

[1] Z. Zhang, Z. Zhao, K. Shen, D. P. Palomar, and W. Yu, "Discerning and enhancing the weighted sum-rate maximization algorithms in communications," Nov. 2023, [Online]. Available: https://arxiv.org/pdf/2311.04546.

[2] Y. Nesterov, "Lectures on convex optimization (second edition)." Springer, 2018.

[3] K. Shen and W. Yu, "Fractional programming for communication systems—Part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, Mar. 2018.

[4] K. Shen, W. Yu, L. Zhao, and D. P. Palomar, "Optimization of MIMO device-to-device networks via matrix fractional programming: A minorization–maximization approach," *IEEE/ACM Trans. Netw.*, vol. 27, no. 5, pp. 2164–2177, Oct. 2019.

[5] 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.

[6] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 794–816, Aug. 2016.

[7] K. Shen and W. Yu, "Fractional programming for communication systems—Part II: Uplink scheduling via matching," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2631–2644, Mar. 2018.

[8] S. S. Christensen, R. Agarwal, E. D. Carvalho, and J. M. Cioffi, "Weighted sum-rate maximization using weighted MMSE for MIMO-BC beamforming design," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4792–4799, Dec. 2008.

[9] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, Apr. 2011.

[10] I. M. Stancu-Minasian, *Fractional programming: Theory, methods and applications*. Norwell, MA, USA: Kluwer, 2012.

[11] A. Charnes and W. W. Cooper, "Programming with linear fractional functionals," *Nav. Res. Logist.*, vol. 9, no. 3, pp. 181–186, Dec. 1962.

[12] S. Schaible, "Parameter-free convex equivalent and dual programs of fractional programming problems," *Zeitschrift für Oper. Res.*, vol. 18, no. 5, pp. 187–196, Oct. 1974.

[13] W. Dinkelbach, "On nonlinear fractional programming," *Manage. Sci.*, vol. 13, no. 7, pp. 492–498, Mar. 1967.

[14] J. P. Crouzeix, J. A. Ferland, and S. Schaible, "An algorithm for generalized fractional programs," *J. Optim. Theory Appl.*, vol. 47, no. 1, pp. 35–49, Sep. 1985.

[15] R. W. Freund and F. Jarre, "Solving the sum-of-ratios problem by an interior-point method," *J. Global Optim.*, vol. 19, no. 1, pp. 83–102, Jan. 2001.

[16] N. T. H. Phuong and H. Tuy, "A unified monotonic approach to generalized linear fractional programming," *J. Global Optim.*, vol. 26, pp. 229–259, July 2003.

[17] H. Konno and K. Fukaishi, "A branch and bound algorithm for solving low rank linear multiplicative and fractional programming problems," *J. Global Optim.*, vol. 18, pp. 283–299, Nov. 2000.

[18] H. P. Benson, "Global optimization of nonlinear sums of ratios," *J. Math. Anal. Appl.*, vol. 263, no. 1, pp. 301–315, Nov. 2001.

[19] S. Qu, K. Zhang, and J. Zhao, "An efficient algorithm for globally minimizing sum of quadratic ratios problem with nonconvex quadratic constraints," *Appl. Math. Comput.*, vol. 189, no. 2, pp. 1624–1636, June 2007.

[20] T. Kuno, "A branch-and-bound algorithm for maximizing the sum of several linear ratios," *J. Global Optim.*, vol. 22, pp. 155–174, Jan. 2002.

[21] X. Liu, Y. Gao, B. Zhang, and F. Tian, "A new global optimization algorithm for a class of linear fractional programming," *MDPI Mathematics*, vol. 7, no. 9, p. 867, Sep. 2019.

[22] H. P. Benson, "Solving sum of ratios fractional programs via concave minimization," *J. Optim. Theory Appl.*, vol. 135, no. 1, pp. 1–17, June 2007.

[23] ——, "Global optimization algorithm for the nonlinear sum of ratios problem," *J. Optim. Theory Appl.*, vol. 112, pp. 1–29, Jan. 2002.

[24] ——, "Using concave envelopes to globally solve the nonlinear sum of ratios problem," *J. Global Optim.*, vol. 22, pp. 343–364, Jan. 2002.

[25] K. Shen, H. V. Cheng, X. Chen, Y. C. Eldar, and W. Yu, "Enhanced channel estimation in massive MIMO via coordinated pilot design," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 6872–6885, Nov. 2020.

[26] Y. Chen, L. Zhao, and K. Shen, "Mixed max-and-min fractional programming for wireless networks," May 2023, [Online]. Available: https://arxiv.org/pdf/2305.02704.

[27] X. Zhao, S. Lu, Q. Shi, and Z.-Q. Luo, "Rethinking WMMSE: Can its complexity scale linearly with the number of bs antennas?" *IEEE Trans. Signal Process.*, vol. 71, pp. 433–446, Feb. 2023.

[28] K. Zhou, Z. Chen, G. Liu, and Z. Chen, "A novel extrapolation technique to accelerate WMMSE," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, June 2023.

[29] A. Hjørungnes and D. Gesbert, "Complex-valued matrix differentiation: Techniques and key results," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2740–2746, May 2007.

[30] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[31] D. P. Bertsekas, "Convex optimization algorithms." Athena Scientific, 2015.