RAIRO-Oper. Res. 55 (2021) S2241–S2258
https://doi.org/10.1051/ro/2020090

RAIRO Operations Research
www.rairo-ro.org

SDO AND LDO RELAXATION APPROACHES TO COMPLEX FRACTIONAL QUADRATIC OPTIMIZATION

Ali Ashrafi* and Arezu Zare

Abstract. This paper examines a complex fractional quadratic optimization problem subject to two quadratic constraints. The original problem is transformed into a parametric quadratic programming problem by the well-known classical Dinkelbach method. Then a semidefinite and Lagrangian dual optimization approaches are presented to solve the nonconvex parametric problem at each iteration of the bisection and generalized Newton algorithms. Finally, the numerical results demonstrate the effectiveness of the proposed approaches.

Mathematics Subject Classification. 90C32, 90C26, 90C22.

Received March 25, 2020. Accepted August 18, 2020.

1. Introduction

The following fractional optimization problem is considered in this paper:

$$\begin{aligned} \min_{x\in\mathbf{C}^n} \quad & \frac{f_1(x)}{f_2(x)} := \frac{x^H A_1 x - 2Re(a_1^H x) + \beta_1}{x^H A_2 x - 2Re(a_2^H x) + \beta_2} \\ \text{s.t.} \quad & g_i(x) := x^H B_i x - 2Re(b_i^H x) + \gamma_i \le 0, \quad i = 1, 2, \end{aligned} \tag{1.1}$$

where $A_i, B_i \in \mathbf{H}^{n\times n}$ are complex Hermitian matrices, $a_i, b_i \in \mathbf{C}^n$ are vectors, $\beta_i, \gamma_i \in \mathbf{R}$ are constants for $i = 1, 2$. The superscript "H" denotes the conjugate transpose. Furthermore, we require the denominator of the objective function to be positive in $\mathcal{S} := \{x \in \mathbf{C}^n|\ g_i(x) \le 0,\ i = 1, 2\}$, in which $\mathcal{S} \ne \emptyset$. In general, problem (1.1) is nonconvex.

Quadratic fractional optimization problems have attracted the attention of many researchers over the last decades, due to their application in many fields such as signal processing, economics, transportation science, engineering, and finance [2, 9, 10, 18, 22, 23]. The first work on nonlinear programming in the complex space appeared when Abrams *et al.* [1] studied duality for the complex nonlinear programming problem. Swarup *et al.* [29] have investigated linear fractional programming in the complex space. Bector *et al.* [4] considered a complex nonlinear fractional programming problem. They developed a Lagrangian dual optimization approach to duality for the problem over cones. Liu *et al.* [19] investigated the complex nondifferentiable fractional programming and established optimality conditions and a duality theorem for the complex nonlinear programming.

Keywords. Fractional quadratic optimization, nonconvex problem, semidefinite programming, Lagrangian dual optimization.

Department of Mathematics, Statistics and Computer Science, Semnan University, Semnan, Iran.
*Corresponding author: a_ashrafi@semnan.ac.ir

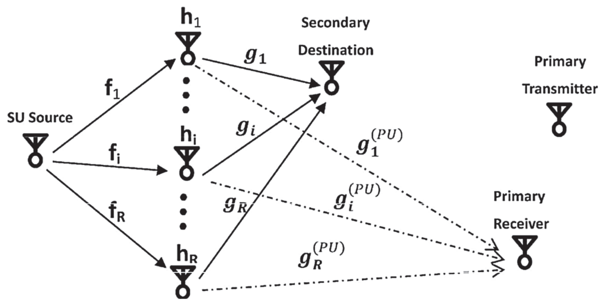

FIGURE 1. Filter-and-forward relay network signal model [21].

Recently, there have been several results considering complex fractional quadratic optimization problems where can be applied to signal processing applications such as robust adaptive beamforming, steering direction, and filter design [7, 11–13, 16, 25, 33–35]. In what follows, we briefly describe an application where a problem of the form (1.1) arises [21].

Consider a cognitive radio (CR) relay network in a frequency-selective wireless channel, as shown in Figure 1. It consists of R single-antenna CR relay nodes, a secondary user (SU) source to relay channels (fore-channels), relay to secondary destination channels (back-channels) and relay to primary user (PU) channels (interference channels) corresponding to the ith relay which

$$f_i = [f_i(0), \dots, f_i(L_f - 1)]^T ,$$
$$g_i = [g_i(0), \dots, g_i(L_g - 1)]^T ,$$
$$g_i^{(\mathrm{PU})} = \left[g_i^{(\mathrm{PU})}(0), \dots, g_i^{(\mathrm{PU})}(L_g^{(\mathrm{PU})} - 1)\right]^T ,$$

for the ith relay node, with the superscript "T" denoting transpose, L_f, L_g and $L_g^{(\mathrm{PU})}$ are the corresponding finite impulse response (FIR) filter length. It is assumed that the direct link between source and destination nodes does not exist and the transmission is divided into two phases. In the first phase, the SU source broadcasts a signal to all CR relay nodes, and in the second phase, each CR relay node filters the received signal and then re-transmit it to the secondary destination. Moreover, the instantaneous channel state information (CSI) of all transmission channels is perfectly known by the system and the computation of all relay coefficients could be performed at the secondary destination or some central node. The results can then be fed back to the relay nodes. The primary destination does not need the channel state information.

Accordingly, the signal received at the relay nodes can be modeled as an $R \times 1$ vector $r(n) = [r_1(n), \dots, r_R(n)]^T$, with $r_i(n)$ given by

$$r_i(n) = s(n) * f_i(n) + n_i(n), \tag{1.2}$$

where $s(n)$ is the information-bearing sequence of symbols transmitted by the SU source node with power of $P_s = E\{| \ s(n) \ |^2\}$, $E\{\cdot\}$ is the expectation operation, $*$ denotes the convolution sum, and $n_i(n)$ is the additive white Gaussian noise (AWGN), with power of $\sigma_n^2 = E\{| \ n_i(n) \ |^2\}$. Then the received signal $r_i(n)$ passes through the ith CR relay filter, with an impulse response $h_i = [h_i(0), \dots, h_i(L_h - 1)]^T$, where L_h represents the relay filter length. Note that the channel impulse responses are assumed to be independent quasi-static, which means that h_i remains static over a frame period. The signal received by the secondary destination node is given by

$$y(n) = \sum_{i=1}^{R} r_i(n) * h_i(n) * g_i(n) + v(n) = s(n) * h_{eqv}(n) + n_{\mathrm{pro}}(n) + v(n), \tag{1.3}$$

where $h_{eqv}(n) = \sum_{i=1}^{R} f_i(n) * h_i(n) * g_i(n)$ is the overall equivalent channel impulse response from the SU source to the secondary destination node, $n_{\rm pro}(n) = \sum_{i=1}^{R} n_i(n) * h_i(n) * g_i(n)$ is the propagation noise from the CR relay nodes and $v(n)$ is the AWGN with power $\sigma_v^2 = E\{| v_i(n) |^2\}$. The leakage signal introduced by CR relays at the primary receiver can be expressed as

$$\begin{aligned} y^{(\rm PU)}(n) &= \sum_{i=1}^{R} \delta_i^{(\rm PU)}(n) \\ &= \sum_{i=1}^{R} s_i(n) * f_i(n) * h_i(n) * g_i^{(\rm PU)}(n) + n_i(n) * g_i^{(\rm PU)}(n), \end{aligned} \tag{1.4}$$

where $\delta_i^{(\rm PU)}(n)$ denotes the part of leakage signal from the ith CR relay nodes. In the following, for better illustration and to facilitate the subsequent problem formulation, the signal model (1.3) will be rewritten in matrix form. To this end, the convolution sum of fore-channel and back-channel related to the ith CR relay node can be expressed as follows

$$c_i = f_i * g_i = \bar{F}_i \cdot g_i = [c_{i,1}, \ldots, c_{i,L_c}], \tag{1.5}$$

where $L_c = (L_f + L_g - 1)$, and $\bar{F}_i$ is a column-circulant matrix of size $L_c \times L_g$

$$\begin{aligned} \bar{F}_i &= [F_i(0), \ldots, F_i(L_g - 1)] \\ F_i(l) &= \Big[\overbrace{0 \ \ldots \ 0}^{l \text{ columns}} \ f_i \ \overbrace{0 \ \ldots \ 0}^{(L_g - l - 1) \text{ columns}}\Big]^T, \quad l = 0, \ldots, L_g - 1. \end{aligned} \tag{1.6}$$

Then the equivalent channel $h_{eqv}(n)$ can be rewritten in matrix form as

$$h_{eqv} = \sum_{i=1}^{R} c_i * h_i = \sum_{i=1}^{R} \bar{C}_i \cdot h_i = \Psi w \tag{1.7}$$

where $\Psi = [\bar{C}_1, \ldots, \bar{C}_R]$, $w = [h_1^T, \ldots, h_R^T]^T$, and $\bar{C}_i$ is a column-circulant matrix, with the size of $(L_f + L_g + L_h - 2) \times L_h$, defined by

$$\begin{aligned} \bar{C}_i &= [C_i(0), \ldots, C_i(L_h)] \\ C_i(l) &= \Big[\overbrace{0 \ \ldots \ 0}^{l \text{ columns}} \ c_i \ \overbrace{0 \ \ldots \ 0}^{(L_h - l - 1) \text{ columns}}\Big]^T, \quad l = 0, \ldots, L_h - 1. \end{aligned} \tag{1.8}$$

The propagation noise $n_{\rm pro}(n)$ can also be expressed in a matrix form

$$n_{\rm pro}(n) = \sum_{i=1}^{R} h_i^H \bar{G}_i^T n_i(n) \tag{1.9}$$

where $\bar{G}_i$ is a column-circulant matrix with a similar form as $\bar{F}_i$, given by

$$\begin{aligned} \bar{G}_i &= [G_i(0), \ldots, G_i(L_h - 1)] \\ G_i(l) &= \Big[\overbrace{0 \ \ldots \ 0}^{l \text{ columns}} \ g_i \ \overbrace{0 \ \ldots \ 0}^{(L_h - l - 1) \text{ columns}}\Big]^T, \quad l = 0, \ldots, L_h - 1 \end{aligned} \tag{1.10}$$

and $n_i(n)$ in (1.9) is the relay noise vector with

$$n_i(n) = [n_i(n), n_i(n-1), \ldots, n_i(n - L_g - L_h + 2)]^T.$$

Now from (1.7) and (1.9), we can rewrite the overall signal model (1.3) in matrix form as

$$y(n) = w^H \Psi s(n) + \sum_{i=1}^{R} h_i^H \bar{G}_i^T n_i(n) + v(n) \tag{1.11}$$

where $s(n) = [s(n), s(n-1)), \ldots, s(n-(L_f+L_g+L_h)-1)]^T$.

Let $\vec{\psi}$ and $\bar{\Psi}$ denote the first row and remaining part of Ψ, respectively and define $\bar{s}(n) = [s(n-1), s(n-2), s(n-L_g-L_h+2)]$. Then,

$$y(n) = \overbrace{w^H \vec{\psi}^T s(n)}^{\text{Desired signal}} + \overbrace{w^H \bar{\Psi}^T \bar{s}(n)}^{\text{Inter-symbol interference}} + \overbrace{\sum_{i=1}^{R} h_i^H \bar{G}_i^T n_i(n) + v(n)}^{\text{Propagation noise and receiver noise}}. \tag{1.12}$$

Equation (1.4) can be rewritten in a matrix form as

$$y^{(\mathrm{PU})}(n) = \sum_{i=1}^{R} \delta_i^{(\mathrm{PU})}(n) = \sum_{i=1}^{R} h_i^H \bar{C}_i^{(\mathrm{PU})^T} s_i(n) + h_i^H \bar{G}_i^{(\mathrm{PU})^T} n_i(n) \tag{1.13}$$

where $\bar{C}_i^{(\mathrm{PU})}$ and $\bar{G}_i^{(\mathrm{PU})}$ are two column-circulant matrices with a similar structure as $\bar{C}_i$ and $\bar{G}_i$, respectively. Accordingly, the power of the desired signal, inter-symbol interference and propagation noise at the secondary destination from (1.12) as follows.

$$E\{|\, y(n)\,|^2\} = w^H Q_s w + w^H Q_i w + w^H Q_n w + \sigma_v^2, \tag{1.14}$$

where

$$Q_s = P_s \cdot \vec{\psi}^T \vec{\psi}^*, \quad Q_i = P_s \cdot \bar{\Psi}^T \bar{\Psi}^*, \quad Q_n = \sigma_n^2 \cdot \mathrm{blkdiag}\left\{\bar{G}_1^T \bar{G}_1^*, \ldots, \bar{G}_R^T \bar{G}_R^*\right\}$$

“blkdiag” is an operation to build a block diagonal matrix from the input argument inside the bracket. The leakage signal power at the primary user can be derived from (1.13),

$$P_{\mathrm{leak}} = E\left\{|\, y^{(\mathrm{PU})}(n)\,|^2\right\} = w^H Q_{\mathrm{leak}} w, \tag{1.15}$$

where Q_{leak} is given by

$$\begin{aligned} Q_{\mathrm{leak}} &= Q_{\mathrm{leak},s} + Q_{\mathrm{leak},n} \\ Q_{\mathrm{leak},s} &= \begin{bmatrix} \bar{C}_1^{(\mathrm{PU})^T} \bar{C}_1^{(\mathrm{PU})^*} & \cdots & \bar{C}_1^{(\mathrm{PU})^T} \bar{C}_R^{(\mathrm{PU})^*} \\ \bar{C}_2^{(\mathrm{PU})^T} \bar{C}_1^{(\mathrm{PU})^*} & \cdots & \bar{C}_2^{(\mathrm{PU})^T} \bar{C}_R^{(\mathrm{PU})^*} \\ \vdots & \ddots & \vdots \\ \bar{C}_R^{(\mathrm{PU})^T} \bar{C}_1^{(\mathrm{PU})^*} & \cdots & \bar{C}_R^{(\mathrm{PU})^T} \bar{C}_R^{(\mathrm{PU})^*} \end{bmatrix} \\ Q_{\mathrm{leak},n} &= \sigma_n^2 \cdot \mathrm{blkdiag}\left\{\bar{G}_1^{(\mathrm{PU})^T} \bar{G}_1^{(\mathrm{PU})^*}, \ldots, \bar{G}_R^{(\mathrm{PU})^T} \bar{G}_R^{(\mathrm{PU})^*}\right\}. \end{aligned}$$

Now, the received signal to interference plus noise ratio (SINR) can be written as

$$\mathrm{SINR} = \frac{w^H Q_s w}{w^H Q_i w + w^H Q_n w + \sigma_v^2}. \tag{1.16}$$

Moreover, the transmitted signal vector from each relay node to destination node is given by

$$t_i(n) = r_i(n) * h_i(n) = s(n) * f_i(n) * h_i(n) + n_i(n) * h_i(n). \tag{1.17}$$

It can be further expressed in matrix form as:

$$t_i(n) = \left(\hat{F}_i h_i\right)^T \hat{s}(n) + h_i^T \hat{n}_i(n) \tag{1.18}$$

where

$$\hat{s}(n) = [s(n), s(n-1), \ldots, s(n - L_f - L_h + 2)]^T,$$
$$\hat{n}_i(n) = [n_i(n), n_i(n-1), \ldots, n_i(n - L_h + 1)]^T$$

are the received signal vector at relay node and the relay noise vector, respectively. $\hat{F}_i$ is a column-circulant matrix with a similar form as $\bar{F}_i$

$$\hat{F}_i = \left[\tilde{F}_i(0), \ldots, \tilde{F}_i(L_h - 1)\right]$$
$$\tilde{F}_i(l) = \left[\overbrace{0 \ \ldots \ 0,}^{l \text{ columns}} f_i, \overbrace{0 \ \ldots \ 0}^{(L_h - l - 1) \text{ columns}}\right], \qquad l = 0, \ldots, L_h - 1.$$

Therefore, the output power at relay node is

$$P_0 = \sum_{i=1}^{R} \{E \mid t_i(n) \mid^2\} = \sum_{i=1}^{R} h_i^T \left(P_s \cdot \hat{F}_i^T \hat{F}_i^* + \sigma^2 \cdot I_{Lh}\right) h_i^* = w^T D w$$

where

$$D = P_s \cdot \text{blkdiag}\{F_1^T F_1^*, \ldots, F_R^T F_R^*\} + \sigma_n^2 \cdot I_{RL_h}.$$

Accordingly, the problem formulation is given by

$$\begin{aligned} \max_{w} \quad & \text{SINR} \\ \text{s.t.} \quad & w^H Q_{\text{leak}} w \le P_N, \\ & w^H D w \le P_0, \end{aligned} \tag{1.19}$$

where $P_N = w^H Q_n w$.

Moreover, Cai *et al.* [6] have studied a nonconvex quadratically constrained quadratic fractional optimization problem in the complex space. They transformed the fractional problem into a nonfractional one by using a parametric approach. They have applied an algorithm for the problem and showed that an optimal solution to the nonfractional problem can be found by solving a single semidefinite optimization problem. Chen *et al.* [8] dealt with a nonconvex fractional optimization problem for solving a large-scale multiple-input-multiple-output (MIMO) system. They transformed the problem into a nonfractional one based on the Dinkelbach method. Then, they presented a Lagrangian dual approach to solve it. In a most recent work, Zare *et al.* [31] have proposed a method to solve a quadratic fractional optimization problem with two quadratic constraints in complex space. The method is based on S-procedure, parametric approach of Dinkelbach and the rank-one decomposition.

The purpose of this paper is to reduce (1.1) to a parametric problem, which itself involves a nonconvex quadratically constraint quadratic subproblem. We have utilized a semidefinite optimization and Lagrangian dual optimization relaxations into the step of bisection and generalized Newton algorithms to solve the parametric problem. The remainder of the paper is organized as follows: In Section 2, we use the classical Dinkelbach method and transform the original problem into a nonfractional one. The resulting nonfractional problem is solved by a bisection and generalized Newton algorithms. In Section 3, a semidefinite optimization relaxation (SDO) is proposed to solve the nonfractional problem at each iteration within both algorithms. In Section 4, a relaxation approach is introduced, which transforms the nonfractional problem into a Lagrangian dual optimization relaxation (LDO) problem. Some numerical results are given for two sets of examples in Section 5. Finally, the conclusions are presented in Section 6.

2. Parametric programming approach

The following proposition gives the relationship between fractional and parametric problems by Dinkelbach [14].

Proposition 2.1. *The following two statements are equivalent:*

$$\begin{aligned}&(1)\quad \min_{x\in\mathcal{S}} \ \frac{f_1(x)}{f_2(x)} = \alpha^*\\ &(2)\quad \mathcal{F}(\alpha^*) := \min_{x\in\mathcal{S}}\{f_1(x) - \alpha^* f_2(x)\} = 0.\end{aligned} \tag{2.1}$$

Utilizing this proposition, the root of $\mathcal{F}$ is the optimal value of (1.1) and then an optimal solution of (2.1) is also an optimal solution of (1.1). Therefore, we focus on the parametric optimization problem (2.1) instead of the original problem (1.1). Now, we give some properties of the univariate function $\mathcal{F}$.

Theorem 2.2 ([32])**.** *The following statements hold.*

(a) $\mathcal{F}$ *is concave over* $\mathbf{R}$.
(b) $\mathcal{F}$ *is continuous at any* $\alpha \in \mathbf{R}$.
(c) $\mathcal{F}$ *is strictly decreasing.*
(d) $\mathcal{F}(\alpha) = 0$ *has a unique solution.*

The function $\mathcal{F}$ is not differentiable. However, there exists an explicit expression of its subgradient.

Theorem 2.3 ([32])**.** *For any* $\alpha \in \mathbf{R}$, *let* $x_\alpha \in \arg\max_{\mathbf{x}\in\mathcal{S}}\{-f_1(x) + \alpha f_2(x)\}$. *Then, a subgradient of* $-\mathcal{F}$ *at* α *is given by* $f_2(x_\alpha)$, *i.e.,*

$$f_2(x_\alpha) \in \partial\mathcal{E}(\alpha), \tag{2.2}$$

where $\partial\mathcal{E}$ *denotes the clarke subdifferential of* $-\mathcal{F}$.

Now, we propose the following algorithms [32] to solve the nonconvex quadratically constrained quadratic minimization problem (2.1).

Algorithm 2.4. Bisection method.

Step 1. Choose l_0 and u_0 such that $l_0 \le \min_{x\in\mathcal{S}} \frac{f_1(x)}{f_2(x)} \le u_0$ holds. Set $k := 1$.

Step 2. Let $\alpha_k := \frac{l_{k-1}+u_{k-1}}{2}$. Then, calculate $\mathcal{F}(\alpha_k)$ by solving problem (2.1).

Step 3. If $\mid \mathcal{F}(\alpha_k) \mid\le \epsilon$, then terminate. Otherwise, update l_k and u_k as follows:

$$\begin{cases} l_k := l_{k-1} \\ u_k := \alpha_k \end{cases} \text{if } \mathcal{F}(\alpha_k) \le 0, \qquad \begin{cases} l_k := \alpha_k \\ u_k := u_{k-1} \end{cases} \text{if } \mathcal{F}(\alpha_k) > 0.$$

Step 4. Let $k := k + 1$ and return to Step 1.

Algorithm 2.5. Generalized Newton method.

Step 1. Choose starting point $\alpha_1 \in \mathbf{R}$. Set k := 1.
Step 2. Calculate $\mathcal{F}(\alpha_k)$ by solving the problem (2.1).
Step 3. If $| \mathcal{F}(\alpha_k) | \leq \epsilon$, then terminate. Otherwise, let:

$$\alpha_{k+1} := \alpha_k - \frac{\mathcal{F}(\alpha_k)}{-f_2(x_k)} = \frac{f_1(x_k)}{f_2(x_k)}.$$

Step 4. Let $k := k+1$ and return to Step 1.

Note that we need to solve the following problem in both Algorithms 2.4 and 2.5

$$\min_{x \in \mathcal{S}} \quad x^H A x - 2Re\left(a^H x\right) + \beta, \tag{2.3}$$

where $A = A_1 - \alpha A_2$, $a = a_1 - \alpha a_2$ and $\beta = \beta_1 - \alpha \beta_2$.
Furthermore, the following assumption is made throughout the paper.

Assumption 2.6. *There exists $\xi_1, \xi_2 \geq 0$ such that $A + \sum_{i=1}^{2} \xi_i B_i \succ 0$.*

3. SDO relaxation approach

In this section, we use an SDO relaxation approach to solve (2.3) globally. To this end, we consider the following problem which is equivalent to (2.3)

$$\begin{aligned} \min_{x \in \mathbf{C}^n} \quad & M_0 \bullet X \\ \text{s.t.} \quad & M_i \bullet X \leq 0, \qquad i = 1, 2, \\ & M_3 \bullet X = 1, \end{aligned} \tag{3.1}$$

where

$$M_0 = \begin{bmatrix} \beta & -a^H \\ -a & A \end{bmatrix}, \; M_1 = \begin{bmatrix} \gamma_1 & -b_1^H \\ -b_1 & B_1 \end{bmatrix}, \; M_2 = \begin{bmatrix} \gamma_2 & -b_2^H \\ -b_2 & B_2 \end{bmatrix},$$

$$M_3 = \begin{bmatrix} 1 & 0_{1\times n} \\ 0_{n\times 1} & 0_{n\times n} \end{bmatrix}, \; X = \begin{bmatrix} 1 & x_0^H \\ x_0 & x_0 x_0^H \end{bmatrix},$$

and $A \bullet B = Re(tr(A^H B))$, where $\mathrm{tr}(\cdot)$ denotes the trace of matrix.
Now, the semidefinite relaxation of (3.1) is given by [5]

$$\begin{aligned} \min_{x \in \mathbf{C}^n} \quad & M_0 \bullet X \\ \text{s.t.} \quad & M_i \bullet X \leq 0, \qquad i = 1, 2, \\ & M_3 \bullet X = 1, \\ & X \succeq 0. \end{aligned} \tag{3.2}$$

For an $n \times n$ matrix Q we let $Q \succeq 0$ ($\succ 0$) denote that Q is positive semidefinite (definite).
The dual of (3.2) is

$$\begin{aligned} \max \quad & y_3 \\ \text{s.t.} \quad & Z = M_0 + y_1 M_1 + y_2 M_2 - y_3 M_3 \succeq 0_{n+1\times n+1}, \\ & y_1, y_2 \geq 0. \end{aligned} \tag{3.3}$$

Next theorem shows that both problems (3.2) and (3.3) are solvable with zero duality gap.

Theorem 3.1. *Suppose that problem* (1.1) *has a strictly feasible solution* x_0 *and Assumption* 2.6 *holds. Then both problems* (3.2) *and* (3.3) *also satisfy the strict feasibility condition. Hence, both problems attain their optimal values and the duality gap is zero.*

Proof. Let X_0 be as follows:

$$X_0 = \begin{bmatrix} 1 & x_0^H \\ x_0 & x_0 x_0^H + Q \end{bmatrix},$$

where $Q = \mathrm{diag}(q_1, \ldots, q_n)$ with all $q_j > 0$ and sufficiently small. Obviously by the Schur complement theorem, X_0 is positive definite. Moreover,

$$\begin{aligned} M_i \bullet X_0 &= \begin{bmatrix} \gamma_i & -b_i^H \\ -b_i & B_i \end{bmatrix} \bullet \begin{bmatrix} 1 & x_0^H \\ x_0 & x_0 x_0^H + Q \end{bmatrix} \\ &= \gamma_i - 2Re(b_i^H x_0) + x_0^H B_i x_0 + \sum_{j=1}^{n} (B_i)_j q_j < 0, \quad i = 1, 2. \end{aligned}$$

Furthermore

$$M_3 \bullet X_0 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \bullet \begin{bmatrix} 1 & x_0^H \\ x_0 & x_0 x_0^H + Q \end{bmatrix} = 1.$$

Then, X_0 is a strictly feasible solution for problem (3.2). For the dual problem we have

$$\begin{aligned} Z &= M_0 + y_1 M_1 + y_2 M_2 - y_3 M_3 \\ &= \begin{bmatrix} \beta & -a^H \\ -a & A \end{bmatrix} + y_1 \begin{bmatrix} \gamma_1 & -b_1^H \\ -b_1 & B_1 \end{bmatrix} + y_2 \begin{bmatrix} \gamma_2 & -b_2^H \\ -b_2 & B_2 \end{bmatrix} \\ &\quad - y_3 \begin{bmatrix} 1 & 0_{1\times n} \\ 0_{n\times 1} & 0_{n\times n} \end{bmatrix}. \end{aligned}$$

Thus

$$Z = \begin{bmatrix} \beta + \sum_{i=1}^{2} \gamma_i y_i - y_3 & -a^H - \sum_{i=1}^{2} y_i b_i^H \\ -a - \sum_{i=1}^{2} y_i b_i & A + \sum_{i=1}^{2} y_i B_i \end{bmatrix}.$$

Now by Schur complement theorem

$$\begin{aligned} Z \succ 0 \Leftrightarrow \left(A + \sum_{i=1}^{2} y_i B_i \right) &- \frac{1}{\beta + \sum_{i=1}^{2} \gamma_i y_i - y_3} \\ &\times \left(a + \sum_{i=1}^{2} y_i b_i \right) \left(a + \sum_{i=1}^{2} y_i b_i \right)^H \succ 0. \end{aligned}$$

Since

$$\left(a + \sum_{i=1}^{2} y_i b_i \right) \left(a + \sum_{i=1}^{2} y_i b_i \right)^H \succeq 0$$

and $A + \sum_{i=1}^{2} \xi_i B_i \succ 0$ then by choosing y_3 sufficiently large and $y_1 = \xi_1, y_2 = \xi_2$, Z is positive definite which implies the strict feasibility condition of (3.2). □

According to Theorem 3.1, X^* with rank r and $(y_1^*, y_2^*, y_3^*, Z^*)$ are optimal solutions for problems (3.2) and (3.3), respectively if and only if the following conditions are satisfied

[I] $M_i \bullet X^* \le 0, \quad i = 1, 2,$
[II] $M_3 \bullet X^* = 1,$
[III] $X^* \succeq 0,$
[IV] $y_i \,(M_i \bullet X^*) = 0,\ i = 1, 2,$
[V] $Z^* \bullet X^* = 0,$
[VI] $Z^* \succeq 0,$
[VII] $y_i \ge 0,\ i = 1, 2.$

The following theorem shows that there exists a rank-one decomposition of the optimal solution (3.2).

Theorem 3.2 ([17]). *Suppose that $X \in \mathbf{H}^n$ is a complex Hermitian positive semidefinite matrix of rank r, and $M_1, M_2 \in \mathbf{H}^n$ are two given Hermitian matrices. Then, there is a rank-one decomposition of X,*

$$X = \sum_{k=1}^{r} x_k x_k^H,$$

such that

$$x_k^H M_1 x_k = \frac{M_1 \bullet X}{r}, \qquad x_k^H M_2 x_k = \frac{M_2 \bullet X}{r}, \qquad k = 1, \ldots, r.$$

Theorem 3.3. *Suppose strict feasibility and Assumption 2.6 hold. If X^* is the optimal solution of (3.2), then there exists $\tilde{x}_k^*$ from the rank-one decomposition of X^*, such that $\tilde{x}_k^*$ is the optimal solution of (2.3).*

Proof. From Theorem 3.2, there is a rank-one decomposition of X^*

$$X^* = \sum_{k=1}^{r} x_k^* {x_k^*}^H$$

such that

$$ {x_k^*}^H M_i x_k^* = M_i x_k^* {}^H x_k^* = \frac{M_i \bullet x_k^*}{r}, \quad i = 1, 2, \quad k = 1, \ldots, r,$$

where r is rank of X^*. Because

$$M_3 \bullet X^* = \sum_{k=1}^{r} M_3 \bullet x_k^* {x_k^*}^H = 1,$$

there must exist a k such that $M_3 \bullet x_k^* {x_k^*}^H > 0$. Let $\bar{x}_k = \frac{x_k^*}{\sqrt{{x_k^*}^H M_3 x_k^*}}$, then, we have

$$M_i \bullet \bar{x}_k \bar{x}_k^H \le 0, \quad i = 1, 2.$$

According to [IV], we have

$$y_i^* (M_i \bullet \bar{x}_k \bar{x}_k^H) = 0, \quad i = 1, 2.$$

Also, from [V], we have $Z^* \bullet X^* = \sum_{k=1}^{r} {x_k^*}^H Z^* x_k^* = 0$, then

$$\forall k, \quad {x_k^*}^H Z^* x_k^* = Z^* \bullet x_k^* {x_k^*}^H = 0 \Rightarrow Z^* \bullet \bar{x}_k \bar{x}_k^H = 0.$$

Hence, $\bar{x}_k \bar{x}_k^H$ is the optimal solution of (3.2). So, let $\bar{x}_k = \begin{bmatrix} 1 \\ \tilde{x}_k^* \end{bmatrix}$, then for $i = 1, 2$,

$$\begin{aligned} \tilde{x}_k^* B_i {\tilde{x}_k^*}^H - 2Re(b_i^H \tilde{x}_k^*) + \gamma_i &= \begin{bmatrix} \gamma_i & -b_i^H \\ -b_i & B_i \end{bmatrix} \bullet \begin{bmatrix} 1 & {\tilde{x}_k^*}^H \\ \tilde{x}_k^* & \tilde{x}_k^* {\tilde{x}_k^*}^H \end{bmatrix} \\ &= M_i \bullet \bar{x}_j \bar{x}_j^H \le 0. \end{aligned}$$

Furthermore, $\tilde{x}_k^*$ is the optimal solution of (2.3). □

Since (3.2) is a relaxation of (3.1), hence $\tilde{x}_k^*$ is optimal for (3.1) as well. Theorem 3.3 also implies that $\tilde{x}_k^*$ is optimal for (2.3). Indeed, in each iteration of Algorithms 2.4 and 2.5, we can obtain an optimal solution (X^*) of (3.2) by using the convex optimization toolbox CVX [15], and then we will find the optimal solution ($\tilde{x}_k^*$) of (2.3) based on Algorithm 3.4 where is a rank-one decomposition [17]. Due to Proposition 2.1, solving problem (1.1) is equivalent to determining the root of the equation $\mathcal{F}(\alpha^*) = 0$, namely, the optimal of (2.3) is optimal for problem (1.1).

Algorithm 3.4. Rank-one decomposition.
Input. $X, A, B \in \mathbf{H}^n$ and X is a complex Hermitian positive semidefinite matrix with $r = rank(X)$.
Output. $X = \sum_{j=1}^{r} x_j^* {x_j^*}^H$, a rank-one decomposition of X such that

$$ {x_j^*}^H A x_j^* = \frac{A \bullet X}{r}, \quad {x_j^*}^H B x_j^* = \frac{B \bullet X}{r}, \quad j = 1, \ldots, r. $$

Step 1. Apply Corollary 4 of [28] to obtain $X = \sum_{j=1}^{r} x_j x_j^H$ such that $x_j^H A x_j = \frac{A \bullet X}{r}$, $j = 1, \ldots, r$.

Step 2. If $x_j^H B x_j = \frac{B \bullet X}{r}$, $j = 1, \ldots, r$, then $x_j^* = x_j$ break and terminate. Otherwise, let j and k be two indices such that

$$ x_j^H B x_j > \frac{B \bullet X}{r}, \; x_k^H B x_k < \frac{B \bullet X}{r} \cdot $$

Step 3. Let

$$ \alpha_1 := \mathbf{Arg}(x_j^H A x_k), \qquad \gamma_1 := \mid x_j^H A x_k \mid, $$
$$ \alpha_2 := \mathbf{Arg}(x_j^H B x_k), \qquad \gamma_2 := \mid x_j^H B x_k \mid, $$

where "Arg" denotes the principal argument of a complex number (which means that $x_j^H A x_k = \gamma_1 e^{i\alpha_1}, x_j^H B x_k = \gamma_2 e^{i\alpha_2}$), then calculate the roots of the following equation

$$ \left(x_j^H B x_j - \frac{B \bullet X}{r} \right) y^2 + 2\gamma_2 \left(\sin(\alpha_2 - \alpha_1) \right) y + \left(x_k^H B x_k - \frac{B \bullet X}{r} \right) = 0. \tag{3.4} $$

Step 4. Let γ be the positive root of (3.4), $\alpha := \alpha_1 + \frac{\pi}{2}$ and $\omega := \gamma e^{i\alpha}$, then

$$ z_j := \frac{\omega x_j + x_k}{\sqrt{1+\gamma^2}}, \qquad z_k := \frac{-x_j + \bar{\omega} x_k}{\sqrt{1+\gamma^2}} \cdot $$

Step 5. Set $x_j := z_j$ and $x_k := z_k$ and return to Step 2.

4. LDO relaxation approach

The LDO relaxation of (2.3) is given by

$$ \max_{\nu \geq 0} \quad \psi(\nu) \tag{4.1} $$

where

$$ \psi(\nu) := \min_x \left[x^H \left(A + \sum_{i=1}^{2} \nu_i B_i \right) x - 2Re \left(a + \sum_{i=1}^{2} \nu_i b_i \right)^H x + \left(\beta + \sum_{i=1}^{2} \nu_i \gamma_i \right) \right]. $$

By [27], this is equivalent to

$$\begin{aligned} &\max_{\nu\geq 0,\tau} \quad \tau \\ &\text{s.t.} \quad x^H\left(A+\sum_{i=1}^{2}\nu_i B_i\right)x - 2Re\left(a+\sum_{i=1}^{2}\nu_i b_i\right)^H x + \left(\beta+\sum_{i=1}^{2}\nu_i\gamma_i\right) \geq \tau. \end{aligned} \tag{4.2}$$

Also, problem (4.2) can be rewritten as the semidefinite programming [3, 20]

$$\begin{aligned} &\max_{\nu\geq 0,\tau} \quad \tau \\ &\text{s.t.} \quad \begin{bmatrix} \beta+\sum_{i=1}^{2}\nu_i\gamma_i-\tau & -a^H-\sum_{i=1}^{2}\nu_i b_i^H \\ -a-\sum_{i=1}^{2}\nu_i b_i & A+\sum_{i=1}^{2}\nu_i B_i \end{bmatrix} \succeq 0, \end{aligned} \tag{4.3}$$

whose dual is equivalent to the Schur relaxation [26]

$$\begin{aligned} &\min_{x,X} \quad A\bullet X - 2Re\left(a^H x\right)+\beta \\ &\text{s.t.} \quad B_i\bullet X - 2Re\left(b_i^H x\right)+\gamma_i \leq 0, \quad i=1,2, \\ &\qquad X \succeq xx^H. \end{aligned} \tag{4.4}$$

We will consider the LDO relaxation approach for problem (2.3) and solve it in each iteration of Algorithms 2.4 and 2.5. First, we divide the maximization of the objective function of problem (4.1) into two levels, [30],

$$\max_{\mu\in\mathcal{G}} \ \max_{\eta\geq 0} \quad \psi(\mu\eta), \tag{4.5}$$

where $\mathcal{G}=\{\mu\geq 0 \mid \ e^T\mu=1, \ e=[1,1]^T\}$ is the standard simplex. Since we use a subgradient-based approach, we need the projection of a vector onto $\mathcal{G}\subset\mathbf{R}^2$ [24]. Now by taking a $\mu\in\mathcal{G}$, consider the inner problem of (4.5) as follows:

$$\varphi(\mu) := \max_{\eta\geq 0} \quad \psi(\mu\eta). \tag{4.6}$$

In fact, the problem (4.6) is the Lagrangian dual for the following problem

$$\begin{aligned} \varphi(\mu) = &\min_{x} \quad x^H A x - 2Re\left(a^H x\right)+\beta \\ &\text{s.t.} \quad x^H\left(\sum_{i=1}^{2}\mu_i B_i\right)x - 2Re\left(\sum_{i=1}^{2}\mu_i b_i^H x\right) + \sum_{i=1}^{2}\mu_i\gamma_i \leq 0. \end{aligned} \tag{4.7}$$

By Theorem 3.1, there is no duality gap between problems (4.6) and (4.7). Now we can consider (4.1)

$$\max_{\mu\in\mathcal{G}} \quad \varphi(\mu). \tag{4.8}$$

Now, we transform (4.7) into the form

$$\begin{aligned} &\min_{x\in\mathbf{C}^n, t\in\mathbf{R}} \quad t \\ &\text{s.t.} \quad x^H A x - 2Re\left(a^H x\right)+\beta \leq t, \\ &\qquad x^H\left(\sum_{i=1}^{2}\mu_i B_i\right)x - 2Re\left(\sum_{i=1}^{2}\mu_i b_i^H x\right) + \sum_{i=1}^{2}\mu_i\gamma_i \leq 0. \end{aligned} \tag{4.9}$$

Moreover, problem (4.9) satisfies Assumption 2.6 when either A or $\sum_{i=1}^{2} \mu_i B_i$ is positive definite. So, we need to define a convex set as follows:

$$\mathcal{W} := \left\{ \delta \geq 0 \mid \ A + \delta \sum_{i=1}^{2} \mu_i B_i \succeq 0 \right\}. \tag{4.10}$$

Set $\mathcal{W}$ is an interval by Assumption 2.6. Thus, we have

$$\delta_* = \min_{\delta \in \mathcal{W}} \delta, \quad \delta^* = \sup_{\delta \in \mathcal{W}} \delta. \tag{4.11}$$

Using δ_* and δ^*, we can consider the following relaxation of problem (4.9)

$$\begin{aligned} \min_{x \in \mathbf{C}^n, t \in \mathbf{R}} \quad & t \\ \text{s.t.} \quad & x^H E_* x - 2Re\left(e_*^H x\right) + \beta + \varepsilon_* \leq t, \\ & x^H E^* x - 2Re\left({e^*}^H x\right) + \beta + \varepsilon^* \leq t, \end{aligned} \tag{4.12}$$

where

$$\begin{aligned} E_* &= A + \delta_* \sum_{i=1}^{2} \mu_i B_i, \qquad e_* = a + \delta_* \sum_{i=1}^{2} \mu_i b_i, \qquad \varepsilon_* = \delta_* \sum_{i=1}^{2} \mu_i \gamma_i, \\ E^* &= A + \delta^* \sum_{i=1}^{2} \mu_i B_i, \qquad e^* = a + \delta^* \sum_{i=1}^{2} \mu_i b_i, \qquad \varepsilon^* = \delta^* \sum_{i=1}^{2} \mu_i \gamma_i. \end{aligned}$$

Theorem 4.1. *Suppose the strict feasibility condition for* (4.7) *and Assumption 2.6 hold. Then problem* (4.7) *is equivalent to* (4.12).

Proof. Let $\mathcal{O}$ denote the feasible set of (4.7), which is equivalent to (4.9). We also denote by $\mathcal{O}_1$ and $\mathcal{O}_2$ the feasible sets of problems (4.9) and (4.12), respectively. Namely

$$\begin{aligned} \mathcal{O}_1 &= \{(x_1, t_1) \in \mathbf{C}^n \times \mathbf{R}_+ \mid \ x_1 \in \mathcal{O}, \ r_1(x_1) \leq t_1\}, \\ \mathcal{O}_2 &= \{(x_2, t_2) \in \mathbf{C}^n \times \mathbf{R}_+ \mid \ x_2 \in \mathcal{O}, \ \hat{r}_1(x_2) \leq t_2, \ \hat{r}_2(x_2) \leq t_2\}, \end{aligned}$$

where $\mathbf{R}_+$ denote the set of positive real numbers,

$$\begin{aligned} r_1(x) &= x^H A x - 2Re\left(a^H x\right) + \beta, \\ \hat{r}_1(x) &= x^H E_* x - 2Re\left(e_*^H x\right) + \beta + \varepsilon_*, \\ \hat{r}_2(x) &= x^H E^* x - 2Re\left({e^*}^H x\right) + \beta + \varepsilon^*. \end{aligned}$$

From (4.10), (4.11) and conditions theorem, it is obvious that $\mathcal{O}_1 \subseteq \mathcal{O}_2$ holds. Hence, we must show that all elements in $(\mathcal{O}_2 \setminus \mathcal{O}_1)$ are suboptimal for (4.12). So for any $(x_2, t_2) \in \mathcal{O}_2$ we have

$$\hat{r}_1(x_2) = \vartheta_1, \quad \hat{r}_2(x_2) = \vartheta_2,$$

where $\vartheta_1, \vartheta_2 \in (0, 1)$ by choosing the suitable δ_* and δ^*. Therefore, we can consider the following element of $\mathcal{O}_1$

$$x_1 = \frac{x_2}{\vartheta^*}, \quad t_1 = \frac{t_2}{\vartheta^*},$$

where $\vartheta^* = \max\{\vartheta_1, \vartheta_2\}$. Thus for any $(x_2, t_2) \in \mathcal{O}_2$ there exists $(x_1, t_1) \in \mathcal{O}_1$ and ϑ^*, such that if $(x_2, t_2) \in \mathcal{O}_2 \setminus \mathcal{O}_1$ then $(x_2, t_2) = (\vartheta^* x_1, \vartheta^* t_1)$. Now, by objective functions of the problems (4.9) and (4.12) we find

$$t_2 = \vartheta^* t_1 \le t_1. \tag{4.13}$$

In addition, if $(x_2, t_2) \in \mathcal{O}_2 \setminus \mathcal{O}_1$ then the inequality (4.13) is strict. Therefore, all elements in $\mathcal{O}_2 \setminus \mathcal{O}_1$ are suboptimal for (4.12). □

According to Assumption 2.6, we need either A or $\sum_{i=1}^{2} \mu_i B_i$, which is a positive semidefinite, to solve problem (4.10). Therefore, the following cases from (4.10) are considered.

(A) When $\delta_* = 0$ then $A \succ 0$. Thus problem (4.12) is equivalent to the following relaxed problem

$$\begin{aligned} \min_{x\in\mathbf{C}^n, t\in\mathbf{R}} \quad & t \\ \text{s.t.} \quad & x^H A x - 2Re\left(a^H x\right) + \beta \le t, \\ & x^H E^{**} x - 2Re\left({e^{**}}^H x\right) + \varepsilon^{**} \le \delta^{**} t, \end{aligned} \tag{4.14}$$

where E^{**}, e^{**} and ε^{**} are calculated similar to E^*, e^* and ε^*, respectively, and

$$\delta^{**} = \left|\min\left\{\lambda_{\min}\left(A^{-\frac{1}{2}} B_1 A^{-\frac{1}{2}}\right), \lambda_{\min}\left(A^{-\frac{1}{2}} B_2 A^{-\frac{1}{2}}\right), 0\right\}\right|,$$

where $\lambda_{\min}(Q)$ denotes the minimum eigenvalue of Q.

(B) When $\delta^* = +\infty$ then $\sum_{i=1}^{2} \mu_i B_i \succ 0$. Thus by dividing the second constraint of (4.12) by δ^* and $\delta^* \to +\infty$, it is equivalent to

$$\begin{aligned} \min_{x\in\mathbf{C}^n, t\in\mathbf{R}} \quad & t \\ \text{s.t.} \quad & x^H E_{**} x - 2Re\left(e_{**}^H x\right) + \varepsilon_{**} \le t, \\ & x^H \left(\sum_{i=1}^{2} \mu_i B_i\right) x - 2Re\left(\sum_{i=1}^{2} \mu_i {b_i}^H x\right) + \sum_{i=1}^{2} \mu_i \gamma_i \le 0, \end{aligned} \tag{4.15}$$

where, E_{**}, e_{**} and ε_{**} are calculated similar to E_*, e_* and ε_*, respectively. In addition, δ_* can be obtained in a similar manner to that δ^{**}.

It should be noted that the well-known optimality conditions hold for problems (4.14) and (4.15). Indeed, x^* and $\bar{x}^*$ are optimal solutions of problems (4.14) and (4.15), respectively, if and only if there exists $\nu_1^*, \nu_2^* \ge 0$ and $\bar{\nu}_1^*, \bar{\nu}_2^* \ge 0$ such that

(a) $\nu_1^* \left(A x_1^* - 2a\right) + \nu_2^* \left(E^{**} x_1^* - 2e^{**}\right) = 0,$
(b) $\nu_1^* \left({x_1^*}^H A x_1^* - 2Re\left(a^H x_1^*\right) + \beta - t\right) = 0,$
(c) $\nu_2^* \left({x_1^*}^H E^{**} x_1^* - 2Re\left({e^{**}}^H x_1\right) + \varepsilon^{**} - \delta^{**} t\right) = 0,$
(d) $\nu_1^* A + \nu_2^* E^{**} \succeq 0,$

and

(e) $\bar{\nu}_1^* \left(E_{**} \bar{x}^* - 2e_{**}\right) + \bar{\nu}_2^* \left(\left(\sum_{i=1}^{2} \mu_i B_i\right) \bar{x}^* - 2\left(\sum_{i=1}^{2} \mu_i b_i\right)\right) = 0,$
(f) $\bar{\nu}_1^* \left({\bar{x}^*}^H E_{**} \bar{x}^* - 2Re\left(e_{**}^H \bar{x}^*\right) + \varepsilon_{**} - t\right) = 0,$

(g) $\bar{\nu}_2^* \left(\bar{x}^{*^H} \left(\sum_{i=1}^2 \mu_i B_i \right) \bar{x}^* - 2Re \left(\sum_{i=1}^2 \mu_i b_i^H \bar{x}^* \right) + \sum_{i=1}^2 \mu_i \gamma_i \right) = 0,$
(h) $\bar{\nu}_1^* E_{**} + \bar{\nu}_2^* \left(\sum_{i=1}^2 \mu_i B_i \right) \succeq 0.$

Furthermore, if (d) and (h) are replaced with

$$\nu_1^* A + \nu_2^* E^{**} \succ 0, \tag{4.16}$$

$$\bar{\nu}_1^* E_{**} + \bar{\nu}_2^* \left(\sum_{i=1}^2 \mu_i B_i \right) \succ 0, \tag{4.17}$$

respectively. Then the optimal solutions of (4.14) and (4.15) are unique. In what follows, we find an optimal solution of the LDO problem (4.12) based on Algorithm 4.2. As shown in Theorem 4.1, (4.12) is equivalent to (4.7), which is equivalent to (2.3). Thus, an optimal solution for (4.12) is optimal for (2.3) as well. Accordingly, from proposition 2.1, an optimal solution of (1.1) can be achieved by finding an optimal solution for (4.12).

Algorithm 4.2. LDO relaxation algorithm

Step 1. Choose an initial $\mu^{(0)}$. Set $h := 0$.
Step 2. Find an optimal solution and the optimal value (4.7) by solving problem (4.14) or (4.15).
Step 3. If $| \varphi \left(\mu^{(h)} \right) | \leq \epsilon$, then stop the algorithm. Otherwise, update $\mu^{(h)}$ by Algorithm 2 of [30].
Step 4. Let $h := h + 1$ and return to Step 2.

It is seen that the main computational costs of Algorithm 4.2 happen in Steps 2 and 3. In Step 2, solving problem (4.14) or (4.15) needs $\mathcal{O}(n^2)$ times. Algorithm 2 of [30] also is done in $\mathcal{O}(m)$ times in Step 3. Hence, we have the following result.

Proposition 4.3. *Algorithm 4.2 correctly solves problems (4.14) or (4.15) in $\mathcal{O}(n^2 m)$ time.*

Remark 4.4. In fact, Algorithm 4.2 can be used for problems with more two quadratic constraints.

5. Numerical results

In this section, two sets of examples for dimensions 100–4000 of different densities are used to test the performance of the SDO and LDO methods. For each dimension, we generate five test problems and report the average CPU time and roots. $\epsilon = 10^{-6}$ is chosen as the tolerance of the optimality. In addition, "–" means the algorithm cannot solve the problem, because of the shortage of memory. The numerical tests are coded in MATLAB 9.2 and run on a personal computer with Intel(R) Core Duo CPU 2.40 GHz and 8.00 GB of RAM.

Example 5.1. Consider the following problem

$$\begin{aligned} \min_{x \in \mathbf{C}^n} \quad & \frac{x^H A_1 x - 2Re(a_1^H x) + \beta_1}{\| x \|^2 + 1} \\ \text{s.t.} \quad & x^H B_j x - 2Re(b_j^H x) + \gamma_j \leq 0, \quad j = 1, 2, \end{aligned}$$

where $A_1, B_1, B_2 \in \mathbf{H}^{n \times n}$, $b_1, b_2, a_1, a_2 \in \mathbf{C}^n$ and $\gamma_1, \gamma_2, \beta_1 \in \mathbf{R}$. Matrices and vectors are generated using the following MATLAB code:

1. fprintf('Enter the size of the problem');
2. n = input(' ');
3. fprintf('Enter the density of the matrix ');
4. density = input(' ');
5. $H_1 = sprand\,(n, n, density) + i * sprand\,(n, n, density)\,;$

6. $A_1 = \left(\frac{H_1+H_1^{'}}{2}\right);$
7. $A_2 = 10 * eye\,(n,n)\,;$
8. $H_2 = sprand\,(n,n,density) + i * sprand\,(n,n,density)\,;$
9. $B_1 = \left(\frac{H_2+H_2^{'}}{2}\right)^{'} * \left(\frac{H_2+H_2^{'}}{2}\right) + 10 * eye\,(n,n)\,;$
10. $H_3 = sprand\,(n,n,density) + i * sprand\,(n,n,density)\,;$
11. $B_2 = \left(\frac{H_3+H_3^{'}}{2}\right)^{'} * \left(\frac{H_3+H_3^{'}}{2}\right) + 10 * eye\,(n,n)\,;$
12. $a_1 = complex\,(rand\,(n,1)\,, rand\,(n,1))\,;$
13. $a_2 = zeros(n,1);$
14. $b_1 = complex\,(rand\,(n,1)\,, rand\,(n,1))\,;$
15. $b_2 = complex\,(rand\,(n,1)\,, rand\,(n,1))\,;$
16. $\beta_1 = rand;$
17. $\beta_2 = 1;$
18. $\gamma_1 = rand;$
19. $\gamma_2 = rand;$

The numerical results of Example 5.1 are provided in Table 1. As we see, the LDO relaxation approach is able to solve all the problems for both algorithms, while the SDO relaxation approach fails for problems with dimensions of 500–4000. Among the problems which can be solved by both algorithms, the LDO relaxation-based algorithm is faster on most problems. Moreover, the results in Table 1 suggest that the generalized Newton method combined with the LDO relaxation approach is more efficient than other optimization techniques.

Example 5.2. Consider the following problem

$$\begin{aligned} \min_{x\in\mathbf{C}^n} \quad & \frac{x^H A_1 x - 2Re(a_1^H x) + \beta_1}{x^H A_2 x - 2Re(a_2^H x) + \beta_2} \\ \text{s.t.} \quad & x^H B_j x - 2Re(b_j^H x) + \gamma_j \le 0, \quad j = 1,2, \end{aligned}$$

where $A_1, A_2, B_1, B_2 \in \mathbf{H}^{n\times n}$, $b_1, b_2, a_1, a_2 \in \mathbf{C}^n$ and $\gamma_1, \gamma_2, \beta_1, \beta_2 \in \mathbf{R}$. Moreover, the test problems are generated using the following MATLAB code:

1. fprintf('Enter the size of the problem');
2. n =input(' ');
3. fprintf('Enter the density of the matrix');
4. density = input(' ');
5. $H_1 = sprandn\,(n,n,density) + i * sprandn\,(n,n,density)\,;$
6. $A_1 = \left(\frac{H_1+H_1^{'}}{2}\right);$
7. $H_2 = sprandn\,(n,n,density) + i * sprandn\,(n,n,density)\,;$
8. $A_2 = \left(\frac{H_2+H_2^{'}}{2}\right);$
9. $H_3 = sprandn\,(n,n,density) + i * sprandn\,(n,n,density)\,;$
10. $B_1 = \left(\frac{H_3+H_3^{'}}{2}\right)^{'} * \left(\frac{H_3+H_3^{'}}{2}\right) + 10 * eye\,(n,n)\,;$
11. $H_4 = sprandn\,(n,n,density) + i * sprandn\,(n,n,density)\,;$
12. $B_2 = \left(\frac{H_4+H_4^{'}}{2}\right)^{'} * \left(\frac{H_4+H_4^{'}}{2}\right) + 10 * eye\,(n,n)\,;$
13. $a_1 = complex\,(randn\,(n,1)\,, randn\,(n,1))\,;$

Table 1. Numerical results for Example 5.1.

		LDO-Newton method		SDO-Newton method		LDO – bisection method		SDO-bisection method	
n	Density	fvalue	Time(s)	fvalue	Time(s)	fvalue	Time(s)	fvalue	Time(s)
100	1	1.3887e-01	0.0979	1.3887e-01	0.9363	1.3887e-01	1.6928	1.3887e-01	16.3509
200	1	8.6614e-02	0.1215	8.6614e-02	4.9893	8.6614e-02	3.3776	8.6614e-02	99.2316
300	1	9.6206e-02	0.1857	9.6206e-02	15.2659	9.6206e-02	3.4078	9.6206e-02	115.4498
400	1	5.7425e-02	0.2512	5.7425e-02	72.0461	5.7425e-02	5.5552	5.7425e-02	238.9328
500	1	2.3291e-02	0.3487	–	–	2.3291e-02	8.4306	–	–
1000	1	1.1684e-01	1.6550	–	–	1.1684e-01	41.3568	–	–
2000	1	2.0114e-01	8.7360	–	–	2.0114e-01	93.5866	–	–
4000	1	7.6783e-02	76.1519	–	–	7.6783e-02	154.0249	–	–
100	0.5	1.6353e-01	0.0965	1.6353e-01	0.9305	1.6353e-01	1.4830	1.6353e-01	16.1519
200	0.5	1.2680e-01	0.1205	1.2680e-01	4.8691	1.2680e-01	3.3976	1.2680e-01	97.8057
300	0.5	7.5750e-02	0.1765	7.5750e-02	14.9651	7.5750e-02	3.2473	7.5750e-02	113.6291
400	0.5	4.3021e-02	0.3201	4.3021e-02	70.4382	4.3021e-02	5.3166	4.3021e-02	235.1078
500	0.5	2.0632e-02	0.4188	–	–	2.0632e-02	8.1944	–	–
1000	0.5	1.8694e-01	1.6629	–	–	1.8694e-01	36.2839	–	–
2000	0.5	1.1775e-01	8.1773	–	–	1.1775e-01	91.1818	–	–
4000	0.5	1.0931e-01	75.2445	–	–	1.0931e-01	149.7948	–	–
100	0.25	1.7817e-01	0.0952	1.7817e-01	0.9055	1.7817e-01	1.3386	1.7817e-01	12.9427
200	0.25	5.4638e-02	0.1192	5.4638e-02	4.8156	5.4638e-02	3.2749	5.4638e-02	94.5781
300	0.25	1.3814e-01	0.1742	1.3814e-01	14.2365	1.3814e-01	3.0561	1.3814e-01	109.7708
400	0.25	4.4435e-02	0.2957	4.4435e-02	69.3668	4.4435e-02	5.2091	4.4435e-02	234.5194
500	0.25	1.8493e-01	0.4102	–	–	1.8493e-01	8.1122	–	–
1000	0.25	8.4374e-02	1.6231	–	–	8.4374e-02	35.3248	–	–
2000	0.25	1.6446e-01	8.5305	–	–	1.6446e-01	88.5417	–	–
4000	0.25	1.2025e-01	74.7569	–	–	1.2025e-01	147.9403	–	–
100	0.1	1.5113e-01	0.0947	1.5113e-01	0.8954	1.5113e-01	1.1963	1.5113e-01	10.8885
200	0.1	1.7542e-01	0.1104	1.7542e-01	4.4880	1.7542e-01	3.1958	1.7542e-01	93.8011
300	0.1	2.4415e-01	0.1627	2.4415e-01	13.8721	2.4415e-01	2.9054	2.4415e-01	105.3246
400	0.1	1.3595e-01	0.2741	1.3595e-01	68.3084	1.3595e-01	5.0446	1.3595e-01	232.2136
500	0.1	3.2957e-02	0.4009	–	–	3.2957e-02	6.1595	–	–
1000	0.1	1.6079e-01	1.5630	–	–	1.6079e-01	34.1997	–	–
2000	0.1	7.4992e-02	8.0114	–	–	7.4992e-02	86.9996	–	–
4000	0.1	1.7430e-01	74.1659	–	–	1.7430e-01	143.5753	–	–
100	0.01	6.1254e-02	0.0916	6.1254e-02	0.8527	6.1254e-02	0.9878	6.1254e-02	9.2283
200	0.01	5.9301e-02	0.0935	5.9301e-02	4.1764	5.9301e-02	2.9807	5.9301e-02	91.5755
300	0.01	1.2588e-01	0.1311	1.2588e-01	13.1455	1.2588e-01	2.8371	1.2588e-01	104.2394
400	0.01	1.3655e-01	0.2543	1.3655e-01	65.0409	1.3655e-01	4.8319	1.3655e-01	228.7518
500	0.01	1.7155e-01	3.9996	–	–	1.7155e-01	5.9433	–	–
1000	0.01	1.3161e-01	1.4806	–	–	1.3161e-01	30.7722	–	–
2000	0.01	1.3374e-01	7.9278	–	–	1.3374e-01	83.5018	–	–
4000	0.01	2.6504e-01	72.6922	–	–	2.6504e-01	139.7429	–	–

14. $a_2 = complex\left(randn\left(n,1\right), randn\left(n,1\right)\right);$
15. $b_1 = complex\left(randn\left(n,1\right), randn\left(n,1\right)\right);$
16. $b_2 = complex\left(randn\left(n,1\right), randn\left(n,1\right)\right);$
17. $\beta_1 = randn;$
18. $\beta_2 = randn;$
19. $\gamma_1 = randn;$
20. $\gamma_2 = randn;$

In Table 2, we have reported the results for Example 5.2, which can be interpreted similarly to Table 1.

TABLE 2. Numerical results for Example 5.2.

		LDO-Newton method		SDO-Newton method		LDO-bisection method		SDO-bisection method	
n	Density	fvalue	Time(s)	fvalue	Time(s)	fvalue	Time(s)	fvalue	Time(s)
100	1	−1.8416e-01	0.0956	−1.8416e-01	0.9417	−1.8416e-01	1.7471	−1.8416e-01	16.4409
200	1	−3.6909e-01	0.1657	−3.6909e-01	4.7835	−3.6909e-01	2.4577	−3.6909e-01	103.8579
300	1	−2.5645e-01	0.2063	−2.5645e-01	16.8312	−2.5645e-01	4.0095	−2.5645e-01	199.1776
400	1	−3.7253e-02	0.3349	−3.7253e-02	68.9528	−3.7253e-02	6.2086	−3.7253e-02	239.7016
500	1	−6.5344e-01	0.4987	–	–	−6.5344e-01	9.1417	–	–
1000	1	−1.2212e-01	2.2984	–	–	−1.2212e-01	44.4041	–	–
2000	1	−5.2547e-02	12.9513	–	–	−5.2547e-02	136.9980	–	–
4000	1	−2.5679e-01	95.247	–	–	−2.5679e-01	198.2473	–	–
100	0.5	−4.5056e-02	0.0873	−4.5056e-02	0.8713	−4.5056e-02	1.6750	−4.5056e-02	15.2248
200	0.5	−1.6779e-01	0.1735	−1.6779e-01	5.0963	−1.6779e-01	2.3897	−1.6779e-01	100.1329
300	0.5	−1.3056e-01	0.1971	−1.3056e-01	16.3578	−1.3056e-01	3.7324	−1.3056e-01	183.5519
400	0.5	−1.2786e-01	0.3248	−1.2786e-01	68.2113	−1.2786e-01	5.0087	−1.2786e-01	237.5629
500	0.5	−9.6638e-02	0.4923	–	–	−9.6638e-02	8.7206	–	–
1000	0.5	−9.6798e-01	2.0099	–	–	−9.6798e-01	41.6869	–	–
2000	0.5	−6.5820e-02	11.9515	–	–	−6.5820e-02	132.1846	–	–
4000	0.5	−3.6375e-02	93.0018	–	–	−3.6375e-02	195.0349	–	–
100	0.25	−1.2645e-01	0.0812	−1.2645e-01	0.8526	−1.2645e-01	1.6125	−1.2645e-01	15.1064
200	0.25	−2.7449e-01	0.1632	−2.7449e-01	4.8847	−2.7449e-01	2.1693	−2.7449e-01	99.2217
300	0.25	−8.0128e-02	0.1928	−8.0128e-02	16.1504	−8.0128e-02	3.5986	−8.0128e-02	180.6669
400	0.25	−5.7902e-02	0.3117	−5.7902e-02	66.9423	−5.7902e-02	4.8311	−5.7902e-02	235.0087
500	0.25	−1.0169e-01	0.4445	–	–	−1.0169e-01	8.5235	–	–
1000	0.25	−1.5580e-01	1.8493	–	–	−1.5580e-01	40.0677	–	–
2000	0.25	−1.4799e-01	11.4262	–	–	−1.4799e-01	131.5967	–	–
4000	0.25	−2.0684e-01	92.7349	–	–	−2.0684e-01	192.9447	–	–
100	0.1	−1.8559e-01	0.0798	−1.8559e-01	0.8029	−1.8559e-01	1.4687	−1.8559e-01	13.6998
200	0.1	−4.7505e-01	0.1457	−4.7505e-01	4.7754	−4.7505e-01	1.9356	−4.7505e-01	97.5454
300	0.1	−6.1356e-02	0.1056	−6.1356e-02	14.9793	−6.1356e-02	3.3209	−6.1356e-02	179.2517
400	0.1	−2.8399e-01	0.2674	−2.8399e-01	63.8407	−2.8399e-01	4.7654	−2.8399e-01	234.1926
500	0.1	−4.5896e-02	0.4216	–	–	−4.5896e-02	8.5128	–	–
1000	0.1	−2.7744e-01	1.8228	–	–	−2.7744e-01	37.5518	–	–
2000	0.1	−1.4825e-01	10.4458	–	–	−1.4825e-01	130.2155	–	–
4000	0.1	−1.3308e-01	91.5199	–	–	−1.3308e-01	189.9861	–	–
100	0.01	−2.2614e-01	0.0734	−2.2614e-01	0.7903	−2.2614e-01	1.1956	−2.2614e-01	12.7485
200	0.01	−8.9024e-02	0.0986	−8.9024e-02	4.3208	−8.9024e-02	1.8849	−8.9024e-02	96.0037
300	0.01	−1.4961e-01	0.1184	−1.4961e-01	12.8060	−1.4961e-01	3.2157	−1.4961e-01	176.9925
400	0.01	−2.5668e-01	0.2584	−2.5668e-01	61.7438	−2.5668e-01	4.6150	−2.5668e-01	232.1345
500	0.01	−1.6662e-01	0.3896	–	–	−1.6662e-01	8.3967	–	–
1000	0.01	−2.0162e-01	1.3752	–	–	−2.0162e-01	35.2463	–	–
2000	0.01	−4.1786e-02	9.9815	–	–	−4.1786e-02	127.6128	–	–
4000	0.01	−7.3453e-01	88.1094	–	–	−7.3453e-01	186.1117	–	–

6. Conclusions

In this paper, we considered the quadratic fractional programming problems with two quadratic constraints in the complex space. We have presented two algorithms from literature and used an SDO and LDO relaxation approaches to solve the inner subproblems within both algorithms. Our computational results on randomly generated test problems with various dimensions and densities show that the Lagrangian dual relaxation approach within the generalized Newton method algorithm is much more efficient compared with other optimization techniques.

References

[1] R.A. Abrams and A. Ben-Israel, A duality theorem for complex quadratic programming. *J. Optim. Theory App.* **4** (1969) 244–252.

[2] Y. Almogy and O. Levin, A class of fractional programming problems. *Oper. Res.* **19** (1971) 57–67.

[3] L. Bai, J.E. Mitchell and J.-S. Pang, Using quadratic convex reformulation to tighten the convex relaxation of a quadratic program with complementarity constraints. *Optim. Lett.* **8** (2014) 811–822.

[4] C. Bector, S. Chandra and T. Gulati, A lagrangian approach to duality for complex nonlinear fractional programming over cones. *Optimization* **8** (1977) 17–25.

[5] A. Ben-Tal and A. Nemirovski, Lectures on Modern Convex Optimization. SIAM, Philadelphia, PA (2001).

[6] H. Cai, Y. Wang and T. Yi, An approach for minimizing a quadratically constrained fractional quadratic problem with application to the communications over wireless channels. *Optim. Methods Softw.* **29** (2014) 310–320.

[7] H. Chen, A.B. Gershman and S. Shahbazpanahi, Filter-and-forward distributed beamforming in relay networks with frequency selective fading. *IEEE Trans. Signal Process.* **58** (2009) 1251–1262.

[8] X. Chen, X. Wang and X. Chen, Energy-efficient optimization for wireless information and power transfer in large-scale mimo systems employing energy beamforming. *IEEE Wireless Commun. Lett.* **2** (2013) 667–670.

[9] C.S. Colantoni, R.P. Manes and A. Whinston, Programming, profit rates and pricing decisions. *Acc. Rev.* **44** (1969) 467–481.

[10] B.D. Craven, Fractional Programming. Heldermann Verlag, Weinheim **4** (1988).

[11] A. De Maio, S. De Nicola, Y. Huang, S. Zhang and A. Farina, Adaptive detection and estimation in the presence of useful signal and interference mismatches. *IEEE Trans. Signal Process.* **57** (2008) 436–450.

[12] A. De Maio and Y. Huang, New results on fractional QCQP with applications to radar steering direction estimation. *IEEE Signal Process. Lett.* **21** (2014) 895–898.

[13] A. De Maio, Y. Huang, D.P. Palomar, S. Zhang and A. Farina, Fractional QCQP with applications in ml steering direction estimation for radar detection. *IEEE Trans. Signal Process.* **59** (2010) 172–185.

[14] W. Dinkelbach, On nonlinear fractional programming. *Manage. Sci.* **13** (1967) 492–498.

[15] M. Grant and S. Boyd, Cvx: Matlab software for disciplined convex programming, version 2.1 (2014).

[16] A. Hassanien and S.A. Vorobyov, A robust adaptive dimension reduction technique with application to array processing. *IEEE Signal Process. Lett.* **16** (2008) 22–25.

[17] Y. Huang and S. Zhang, Complex matrix decomposition and quadratic programming. *Math. Oper. Res.* **32** (2007) 758–768.

[18] H.-C. Lai and J. Liu, Complex fractional programming involving generalized quasi/pseudo convex functions. *ZAMM-J. Appl. Math. Mech./Z. Angew. Math. Mech.* **82** (2002) 159–166.

[19] J.-C. Liu, C.-C. Lin and R.-L. Sheu, Optimality and duality for complex nondifferentiable fractional programming. *J. Math. Anal. App.* **210** (1997) 804–824.

[20] C. Lu, S.-C. Fang, Q. Jin, Z. Wang and W. Xing, KKT solution and conic relaxation for solving quadratically constrained quadratic programming problems. *SIAM J. Optim.* **21** (2011) 1475–1490.

[21] J. Ma, W. Liu and R. Langley, Filter-and-forward distributed relay beamforming for cognitive radio systems. In: 2015 IEEE International Conference on Communication Workshop (ICCW). IEEE (2015) 895–900.

[22] J. Ohlson and W. Ziemba, Optimal portfolio policies for an investor with a power utility function facing a log normal securities market. *J. Financ. Quant. Anal* **11** (1976) 1.

[23] N.T.H. Phuong and H. Tuy, A unified monotonic approach to generalized linear fractional programming. *J. Global Optim.* **26** (2003) 229–259.

[24] B.T. Polyak, A general method for solving extremal problems. In: Vol. 174 of *Doklady Akademii Nauk*. Russian Academy of Sciences, Doklady (1967) 33–36.

[25] S. Ramprashad, T.W. Parks and R. Shenoy, Signal modeling and detection using cone classes. *IEEE Trans. Signal Process.* **44** (1996) 329–338.

[26] N.Z. Shor, Dual quadratic estimates in polynomial and boolean programming. *Ann. Oper. Res.* **25** (1990) 163–168.

[27] R.J. Stern and H. Wolkowicz, Indefinite trust region subproblems and nonsymmetric eigenvalue perturbations. *SIAM J. Optim.* **5** (1995) 286–313.

[28] J.F. Sturm and S. Zhang, On cones of nonnegative quadratic functions. *Math. Oper. Res.* **28** (2003) 246–267.

[29] K. Swarup and J. Sharma, Programming with linear fractional functionals in complex spaces. *Cahiers du centre d'Etudes et de Recherche Operationelle* **12** (1970) 103–109.

[30] S. Yamada and A. Takeda, Successive lagrangian relaxation algorithm for nonconvex quadratic optimization. *J. Global Optim.* **71** (2018) 313–339.

[31] A. Zare, M. Keyanpour and M. Salahi, On fractional quadratic optimization problem with two quadratic constraints. *Numer. Algebra Control Optim.* **10** (2020) 301.

[32] A. Zhang and S. Hayashi, Celis-dennis-tapia based approach to quadratic fractional programming problems with two quadratic constraints. *Numer. Algebra Control Optim.* **1** (2011) 83–98.

[33] J. Zhang and M.C. Gursoy, Relay beamforming strategies for physical-layer security. In: 2010 44th Annual Conference on Information Sciences and Systems (CISS). IEEE (2010) 1–6.

[34] J. Zhang, L. Guo, T. Kang and P. Zhang, Cooperative beamforming in cognitive radio network with two-way relay. In: 2014 IEEE 79th Vehicular Technology Conference (VTC Spring). IEEE (2014) 1–5.

[35] G. Zheng, K.-K. Wong, A. Paulraj and B. Ottersten, Collaborative-relay beamforming with perfect CSI: Optimum and distributed implementation. *IEEE Signal Process. Lett.* **16** (2009) 257–260.