Improved DC Programming Approaches for Solving the Quadratic Eigenvalue Complementarity Problem

Yi-Shuai Niu[a,*], Joaquim Júdice[b], Hoai An Le Thi[c], Dinh Tao Pham[d]

[a]*School of Mathematical Sciences & SJTU-Paristech Elite Institute of Technology, Shanghai Jiao Tong University, Shanghai 200240, China*
[b]*Instituto de Telecomunicações, Coimbra, Portugal*
[c]*University of Lorraine, Metz, France*
[d]*National Institute of Applied Sciences, Rouen, France*

Abstract

In this paper, we discuss the solution of a Quadratic Eigenvalue Complementarity Problem (QEiCP) by using Difference of Convex (DC) programming approaches. We first show that QEiCP can be represented as dc programming problem. Then we investigate different dc programming formulations of QEiCP and discuss their dc algorithms based on a well-known method – DCA. A new local dc decomposition is proposed which aims at constructing a better dc decomposition regarding to the specific feature of the target problem in some neighborhoods of the iterates. This new procedure yields faster convergence and better precision of the computed solution. Numerical results illustrate the efficiency of the new dc algorithms in practice.

Keywords: Global Optimization, Eigenvalue Problem, Complementarity Problem, DC Programming
2010 MSC: 90C26, 90C30, 90C33

1. Introduction

Let $\mathcal{M}_n$ be the set of all $n \times n$ real matrices with $n \in \mathbb{N}^*$, where $\mathbb{N}^*$ denotes the set of all nonzero natural numbers. Given three matrices A, B, C in $\mathcal{M}_n$, the *Quadratic Eigenvalue Complementarity Problem* (QEiCP) consists of finding a scalar $\lambda \in \mathbb{R}$ and a vector $x \in \mathbb{R}^n \setminus \{0\}$ such that

$$\begin{cases} w = \lambda^2 Ax + \lambda Bx + Cx, \\ x^\top w = 0, \\ x \geq 0, w \geq 0. \end{cases} \tag{1}$$

This problem has been introduced in [1] where some applications were highlighted. We denote the problem (1) as QEiCP(A, B, C). In any solution (λ, x) of QEiCP, the λ-component is a *quadratic complementary eigenvalue* and the x-component is a *quadratic complementary eigenvector* of (A, B, C) associated to λ.

QEiCP is an extension of the well-known Eigenvalue Complementarity Problem (EiCP). This last problem has been introduced in [2] and consists of finding a complementary eigenvalue $\lambda \in \mathbb{R}$ and an associated complementary eigenvector $x \in \mathbb{R}^n \setminus \{0\}$ such that

$$\begin{cases} w = \lambda Bx - Cx, \\ x \geq 0, w \geq 0, \\ x^\top w = 0. \end{cases} \tag{2}$$

where $B, C \in \mathcal{M}_n$. Therefore, EiCP is a special case of QEiCP when the matrix A is null (i.e., EiCP(B, C) = QEiCP$(0, B, -C)$).

[*]Corresponding author
Email address: `niuyishuai@sjtu.edu.cn` (Yi-Shuai Niu)

EiCP has a solution if the matrix B of the leading λ-term is positive definite (PD) [2, 3], that is,

$$x^\top Bx > 0, \forall x \in \mathbb{R}^n \setminus \{0\}. \tag{3}$$

During the past several years, many theoretical results [2, 3, 7, 10], applications [9, 15, 20] and extensions [4, 8, 11, 14, 21, 22] of EiCP have been discussed and a number of efficient algorithms have been proposed for the solution of this problem [5, 6, 12, 13, 16, 17, 18, 19].

Contrary to the EiCP, QEiCP may have no solution even when the matrix A of the leading λ-term is PD [23]. The existence of a solution of QEiCP depends on the properties of the given matrices A, B and C. If the matrix A is PD, then QEiCP has a solution if one of the two following conditions holds:

(i) $C \notin S_0$ [23], where S_0 is the class of matrices defined in [24] by

$$S_0 = \{C \in \mathcal{M}_n : \exists x \in \mathbb{R}^n_+ \setminus \{0\}, Cx \geq 0\}. \tag{4}$$

(i) *co-hyperbolic* [1]

$$(x^\top Bx)^2 \geq 4(x^\top Ax)(x^\top Cx), \forall x \geq 0. \tag{5}$$

As discussed in [23], none of these conditions implies the other. Furthermore, investigating whether $C \notin S_0$ reduces to solving a special linear program [23]. On the other hand, it is very hard to prove that the co-hyperbolic condition holds in practice. However, there are some sufficient conditions which imply (5). For instance, this occurs if A and $-C$ are both PD matrices.

A number of algorithms have been proposed for the solution of QEiCP and its extensions to other cases when $A \in$ PD and one of the conditions $C \notin S_0$ or co-hyperbolicity holds [5, 25, 23, 26, 13, 27, 28, 1]. Two nonlinear programming (NLP) formulations of QEiCP have been introduced in [23, 13] such that (λ, x) is a solution of QEiCP if and only if (λ, x) is a global minimum of NLP with an optimal value equal to zero. In this paper, we introduce dc programming formulations for QEiCP which are based on these nonlinear programs.

As for the NLPs mentioned before, a stationary point of the dc program with null objective function value gives a solution of QEiCP. In order to find such a stationary point, the classical DC algorithm (DCA) [29, 30] may be used. In our previous work [31], we showed that DCA has a very good performance for solving QEiCP with fast convergence to a complementary eigenvalue for most of the test problems. However, it is also observed that for some instances, DCA converges quite slowly with a very small reduction of the objective function value from one iteration to the next. Improving the performance of DCA for QEiCP is the biggest motivation for this paper. Next, we summarize the major contributions of this paper.

(i) Finding an appropriate dc decomposition for any dc function is an open question in dc programming [30]. We propose a tool to *qualify an appropriate dc decomposition* (Proposition 1) and Definition 1) which relies on the convexity of the dc components. Since any dc function has infinitely many dc decompositions, this tool is important for comparing the quality of two dc decompositions.

(ii) We propose a new dc programming formulation for QEiCP based on the *Difference of Convex Sums-of-Squares (DCSOS) decompositions* defined in [32]. This new decomposition is compared with the classical universal dc decomposition proposed in [31]. Numerical results reported in section 9 and illustrated in Tables 3 and 4 indicate a good performance of our new approach.

(iii) We introduce a new formulation to *compute a tighter bound* for complementary eigenvalues given in Theorem 8. Numerical simulations illustrate that this bound is tighter than those given in [13, 31]. A tighter bound is still required for the universal dc decomposition, which will provide a better dc decomposition for DCA.

(iv) The use of DCA for solving DCSOS decomposition requires solving a convex polynomial optimization in each iteration that is observed inefficient particularly for large-scale instances using existing convex optimization solvers such as IPOPT [33]. In order to improve the efficiency of DCA, we propose an *equivalent second order conic programming (SOCP) formulation* for this convex polynomial optimization problem. The SOCP formulation can be solved more effectively in practice by existing SOCP solvers such as CPLEX [34] and GUROBI [35].

(v) Finding a good initial point is an open question in dc programming [30]. We propose a heuristic procedure to *estimate a good initial point* for DCA when applied to QEiCP. This estimation only requires solving a strictly convex quadratic optimization problem over a standard simplex.

(vi) We propose a *local dc decomposition algorithm* which updates in each iteration a better dc decomposition and yields a faster convergence rate for DCA to get a computed solution with a better precision.

The organization of the paper is as follows. In section 2, we discuss how to qualify the dc decompositions which provide an useful tool for analyzing the quality of different dc decompositions. Section 3 contains two different nonlinear programming formulations (NLPs) for QEiCP and their dc programming formulations are established in sections 4, 5 and 6. Some new results concerning tighter bounds estimation for complementary eigenvalues are discussed in section 7. Some algorithms based on the classical DCA for solving the dc programming formulations are presented in section 8. In particular, the SOCP formulation for polynomial convex subproblem is introduced in subsection 8.3, a heuristic for a good initial point estimation is discussed in subsection 8.4, and a new local dc decomposition method is proposed in subsection 8.5. Computational experiments with these algorithms are reported in section 9. Finally, some conclusions are presented in the last section of the paper.

2. Quality of dc decompositions

Let D be a non-empty closed convex set of $\mathbb{R}^n$, let $f : D \to \mathbb{R}$ be a dc function with a dc decomposition as $f = g - h$ where g and h are real valued convex functions defined on D. A dc program is defined by

$$\min\{g(x) - h(x) : x \in D\} \tag{6}$$

An efficient DC algorithm, called DCA, was introduced by D.T. Pham in 1985 and extensively developed by H.A. Le Thi and D.T. Pham since 1994 (the reader can refer to `http://www.lita.univ-lorraine.fr/~lethi/index.php` and [29, 36, 30]).

DCA consists of constructing two sequences $\{x^k\}$ and $\{y^k\}$ via the scheme:

$$\begin{gathered}\text{Given } x^0 \in \mathbb{R}^n,\\ x^k \to y^k \in \partial h(x^k)\\ \swarrow\\ x^{k+1} \in \partial g^*(y^k) = \operatorname{argmin}\{g(x) - \langle x, y^k\rangle : x \in D\}.\end{gathered}$$

The symbol ∂h stands for the sub-differential of the convex function h, and g^* is the conjugate function of g. These definitions are fundamental to understand the algorithm and can be found in any textbook of convex analysis (see e.g., [37]).

There exist infinitely many dc decompositions for any dc function f. In fact, if f has a dc decomposition $g - h$, then $f = (g + \phi) - (h + \phi)$ is also a dc decomposition for any convex function $\phi : D \to \mathbb{R}$. An open question in the field of dc programming is what is a good dc decomposition for DCA and how to find it.

Firstly, we should define some rules to qualify a dc decomposition. Regarding to the geometrical interpretation of DCA given in [38, 39], the quality of a dc decomposition depends on the specific feature of the dc function f at a given point in the convex set D. More specifically, DCA requires constructing in the k-th ($k \in \mathbb{N}^*$) iteration a convex overestimation of f at point x^k, denoted by f^k as

$$f^k(x) = g(x) - (h(x^k) + \langle x - x^k, y^k\rangle) \tag{7}$$

where $y^k \in \partial h(x^k)$. So $f^k(x)$ is obtained by replacing $h(x)$ by its affine approximation at x^k. Hence

$$f^k(x) \geq f(x), \forall x \in D, \forall k \in \mathbb{N}^* . \tag{8}$$

Therefore, the smaller the gap between f^k and f on D is, the better f^k fits f, and the better minimizers of f^k approach minimizers of f.

In order to find out what is the most important key point to reduce the gap between the convex function f^k and the dc function f, we recall that for two given convex functions g and g', we say that g is more (resp. less) convex than g' if $g - g'$ (resp. $g' - g$) is still a convex function. Then, we have the following result:

Proposition 1. *Let $g-h$ and $g'-h'$ be two dc decompositions for a dc function f over a closed convex set D.*

(i) The function g is less (resp. more) convex than g' if and only if h is less (resp. more) convex than h'.

(ii) If g is less convex than g', then the convex overestimations f^k_g and $f^k_{g'}$ given by

$$f^k_g(x) = g(x) - (h(x^k) + \langle x - x^k, y^k\rangle), \text{ with } y^k \in \partial h(x^k).$$

$$f^k_{g'}(x) = g'(x) - (h'(x^k) + \langle x - x^k, y'^k\rangle), \text{ with } y'^k \in \partial h'(x^k).$$

satisfy

$$f^k_g(x) \le f^k_{g'}(x), \forall x \in D.$$

Proof. (i) If g is less (resp. more) convex than g', then $g'-g$ (resp. $g-g'$) is a convex function, and $h'-h = g'-g$ (resp. $h-h' = g-g'$) is also convex, which yields the desired result.
(ii) Let us denote $d = g'-g$, which is a convex function on D if g is less convex than g'. Then $g = g'-d$ and $h = h'-d$. Thus we have

$$\begin{aligned} f^k_g(x) &= g(x) - (h(x^k) + \langle x - x^k, y^k\rangle) \\ &= g'(x) - d(x) - (h'(x^k) - d(x^k) + \langle x - x^k, y^k - y'^k + y'^k\rangle) \\ &= g'(x) - (h'(x^k) + \langle x - x^k, y'^k\rangle) - (d(x) - (d(x^k) + \langle x - x^k, y'^k - y^k\rangle)) \\ &= f^k_{g'}(x) - (d(x) - (d(x^k) + \langle x - x^k, y'^k - y^k\rangle)) \end{aligned}$$

Since d is convex and $y'^k - y^k \in \partial(h'-h)(x^k) = \partial d(x^k)$, then we get

$$d(x) - (d(x^k) + \langle x - x^k, y'^k - y^k\rangle) \ge 0, \forall x \in D.$$

Hence,

$$f^k_g(x) \le f^k_{g'}(x), \forall x \in D.$$

According to Proposition 1 and the inequality (8), if g is less convex than g' then

$$f(x) \le f^k_g(x) \le f^k_{g'}(x), \forall x \in D,$$

which means that f^k_g should fit better f than $f^k_{g'}$ on D. This result reveals that the most important point for choosing a good dc decomposition is to use functions g and h that are less convex as possible. Now, we can give a definition that states when dc decomposition is better than another one.

Definition 1. Let $g-h$ and $g'-h'$ be two dc decomposition for a dc function f. We say that the dc decomposition with g and h is better than the one with g' and h' if g is less convex than g' or h is less convex than h'.

3. Nonlinear programming formulations of QEiCP

The QEiCP defined in (1) is obviously equivalent to the problem:

$$\begin{cases} w = \lambda^2 Ax + \lambda Bx + Cx, \\ x^\top w = 0, \\ e^\top x = 1, \\ x \ge 0, w \ge 0. \end{cases} \tag{9}$$

by introducing the additional constraint $e^\top x = 1$. As previously discussed in [13, 31], the problem (9) is equivalent to :

$$\begin{cases} w = Az + By + Cx, \\ y = \lambda x, \\ z = \lambda y, \\ x^\top w = 0, \\ e^\top x = 1, \\ e^\top y = \lambda, \\ x \geq 0, w \geq 0. \end{cases} \tag{10}$$

which can be rewritten as a nonlinear polynomial optimization problem described as follows:

$$(P) \qquad \min\{f(x, y, z, w, \lambda) : (x, y, z, w, \lambda) \in \mathcal{C}\}$$

where

$$f(x, y, z, w, \lambda) = \|y - \lambda x\|^2 + \|z - \lambda y\|^2 + x^\top w \tag{11}$$

and

$$\mathcal{C} = \{(x, y, z, w, \lambda) : w = Az + By + Cx, e^\top x = 1, e^\top y = \lambda, (x, w, z) \geq 0\}. \tag{12}$$

In the whole paper, we denote $\|x\|$ for the euclidean norm (2-norm) of vector $x \in \mathbb{R}^n$.

Remark 1. In problem (10), the linear constraint $e^\top y = \lambda$ is redundant since it is always satisfied when $y = \lambda x$ and $e^\top x = 1$. However, it is useful to be presented in (P) when we remove $y = \lambda x$ and add $\|y - \lambda x\|$ into objective function, since this linear constraint reveals a relationship between y and λ, which could reduce the feasible set of (P) without loosing any optimal solution.

The following theorem establishes the equivalence between QEiCP and (P) which is a simple consequence of the definition of QEiCP and (P).

Theorem 2.

(i) If (P) is infeasible, then QEiCP(A, B, C) is infeasible.

(ii) If (P) has nonzero global optimal value, then QEiCP(A, B, C) has no solution.

(iii) If (P) has a global optimal solution $(x^*, y^*, z^*, w^*, \lambda^*)$ such that $f(x^*, y^*, z^*, w^*, \lambda^*) = 0$, then (λ^*, x^*) is a solution of QEiCP(A, B, C).

(iv) Conversely, if (λ^*, x^*) is a solution of QEiCP(A, B, C), then $(\bar{x}, \bar{y}, \bar{z}, \bar{w}, \bar{\lambda})$ defined by

$$\bar{\lambda} = \lambda^*; \bar{x} = \frac{x^*}{e^\top x^*}; \bar{y} = \bar{\lambda}\bar{x}; \bar{z} = \bar{\lambda}\bar{y}; \bar{w} = A\bar{z} + B\bar{y} + C\bar{x}$$

is a global optimal solution of (P) with zero global optimal value.

Observe that under the assumption $x \geq 0, w \geq 0$, the complementarity constraint $w^\top x = 0$ is equivalent to $\sum_{i=1}^n \min\{x_i, w_i\} = 0$. Hence the bilinear term $w^\top x$ in f can be replaced by a concave polyhedral function $\sum_{i=1}^n \min\{x_i, w_i\}$ which yields the following equivalent formulation of (P)

$$(P') \qquad \min\{f'(x, y, z, w, \lambda) : (x, y, z, w, \lambda) \in \mathcal{C}\}$$

where

$$f'(x, y, z, w, \lambda) = \|y - \lambda x\|^2 + \|z - \lambda y\|^2 + \sum_{i=1}^n \min\{x_i, w_i\} \tag{13}$$

and $\mathcal{C}$ is defined in (12). The difference between (P) and (P') is only related to their objective functions. The objective function of the problem (P) is a $\mathcal{C}^\infty$ polynomial function, while the objective function in (P') consists of a polyhedral function which is not differentiable at some points. Nevertheless the non-smoothness of problem (P'), it is worthwhile to introduce this formulation since this polyhedral function can be easily presented as a DC function and DCA possesses finite convergence for optimizing a polyhedral DC function on a convex set defined by linear constraints [40]. In the following sections, we will investigate how to represent the problems (P) and (P') into dc programming formulations.

4. A first dc programming formulation

Problem (P) is a nonconvex polynomial optimization problem in which a nonconvex polynomial function f is minimized on a polyhedral convex set $\mathcal{C}$. As in [31], we reformulate (P) into a dc programming problem by writing the function f given by (11) as follows:

$$\begin{aligned} f(x,y,z,w,\lambda) &= \|y\|^2 + \|z\|^2 - 2\lambda y^\top (x+z) + \lambda^2(\|x\|^2 + \|y\|^2) + x^\top w \\ &= f_0(y,z) + f_1(x,y,z,\lambda) + f_2(x,y,\lambda) + f_3(x,w) \end{aligned} \tag{14}$$

with

$$\begin{cases} f_0(y,z) = \|y\|^2 + \|z\|^2, \\ f_1(x,y,z,\lambda) = -2\lambda y^\top (x+z), \\ f_2(x,y,\lambda) = \lambda^2(\|x\|^2 + \|y\|^2), \\ f_3(x,w) = x^\top w. \end{cases} \tag{15}$$

The function f_0 is a convex quadratic function, while f_1, f_2 and f_3 are all nonconvex polynomial functions. Next, we explain how to get a dc decomposition for f_1, f_2 and f_3.

1. As in [31], a dc decomposition for bilinear function f_3 is given by

$$f_3(x,w) = \frac{\|x+w\|^2}{4} - \frac{\|x-w\|^2}{4} \tag{16}$$

where $\frac{\|x+w\|^2}{4}$ and $\frac{\|x-w\|^2}{4}$ are both convex quadratic functions.

2. A dc decomposition for f_2 is given by

$$f_2(x,y,\lambda) = \frac{(\lambda^2 + \|x\|^2)^2 + (\lambda^2 + \|y\|^2)^2}{2} - \frac{2\lambda^4 + \|x\|^4 + \|y\|^4}{2} \tag{17}$$

where $\frac{(\lambda^2+\|x\|^2)^2+(\lambda^2+\|y\|^2)^2}{2}$ and $\frac{2\lambda^4+\|x\|^4+\|y\|^4}{2}$ are both convex functions. Note that this dc decomposition is different from the one presented in [31].

3. In order to get a dc decomposition for f_1, we can rewrite λ as the dc decomposition

$$\lambda = \frac{(\lambda+1)^2}{2} - \frac{\lambda^2+1}{2} \tag{18}$$

and $y^\top(x+z)$ as the dc decomposition

$$y^\top(x+z) = \left(\frac{\|y+x\|^2}{4} + \frac{\|y+z\|^2}{4}\right) - \left(\frac{\|y-x\|^2}{4} + \frac{\|y-z\|^2}{4}\right). \tag{19}$$

Finally, f_1 is the product of two dc functions -2λ and $y^\top(x+z)$ with positive dc components. The following proposition gives an explicit dc decomposition of the product function $f_1 f_2$.

Proposition 3 ([41]). *If $\phi_1 = g_1 - h_1$ and $\phi_2 = g_2 - h_2$ whose dc components g_1, h_1, g_2, h_2 are all positive convex functions, then $\phi_1\phi_2$ has a dc decomposition*

$$\phi_1\phi_2 = \frac{(g_1+g_2)^2 + (h_1+h_2)^2}{2} - \frac{(g_1+h_2)^2 + (g_2+h_1)^2}{2}.$$

Based on Proposition 3, we obtain a dc decomposition of f_1 as:

$$\begin{aligned} f_1(x,y,z,\lambda) = \\ \frac{(4\lambda^2 + 4 + \|y+x\|^2 + \|y+z\|^2)^2 + (4(\lambda+1)^2 + \|y-x\|^2 + \|y-z\|^2)^2}{32} \\ - \frac{(4\lambda^2 + 4 + \|y-x\|^2 + \|y-z\|^2)^2 + (4(\lambda+1)^2 + \|y+x\|^2 + \|y+z\|^2)^2}{32}. \end{aligned} \tag{20}$$

From (14)–(17) and (20), a dc decomposition for the polynomial objective function f is given by:

$$f(x,y,z,w,\lambda) = g(x,y,z,w,\lambda) - h(x,y,z,w,\lambda) \tag{21}$$

where

$$\begin{cases} g(x,y,z,w,\lambda) = & \|y\|^2 + \|z\|^2 + \frac{\|x+w\|^2}{4} + \frac{(\lambda^2+\|x\|^2)^2+(\lambda^2+\|y\|^2)^2}{2} \\ & + \frac{(4\lambda^2+4+\|y+x\|^2+\|y+z\|^2)^2+(4(\lambda+1)^2+\|y-x\|^2+\|y-z\|^2)^2}{32}, \\ h(x,y,z,w,\lambda) = & \frac{\|x-w\|^2}{4} + \frac{2\lambda^4+\|x\|^4+\|y\|^4}{2} \\ & + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)^2+(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)^2}{32}. \end{cases} \tag{22}$$

Hence, a dc programming formulation of (P) is finally stated as follows:

DC Programming Formulation 1: (P_{DC})

$$\min\{g(x,y,z,w,\lambda) - h(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \mathcal{C}\} \tag{23}$$

where g and h are defined in (22) and $\mathcal{C}$ is given by (12).

Note that the dc components g and h defined by (22) are written in form of sums-of-squares (SOS). This decomposition is called *Difference of Convex SOS* (DCSOS) decomposition, which is an extension of the Difference of SOS (DSOS) decomposition introduced by Niu in [32]. It is proved that any polynomial can be rewritten as DCSOS whose decomposition can be constructed in polynomial time. Another dc decomposition for polynomial functions is the Difference of SOS-convex decomposition proposed by Ahmadi and Hall [42] whose dc components are not supposed to be positive. The reader can refer to [32, 42] for more discussions about DCSOS and Difference of SOS-convex decompositions for polynomials.

5. A second dc programming formulation

This dc decomposition is based on an *universal dc decomposition technique* for the functions f_1 and f_2. This approach exploits the fact that the spectral radius of the Hessian matrices of f_1 and f_2 are bounded on the convex set $\mathcal{C}$. In fact, if $\phi : D \to \mathbb{R}$ is a convex function of class $\mathcal{C}^2$ defined on a compact convex set D of $\mathbb{R}^n$, then the spectral radius of the Hessian matrix $\nabla^2\phi(x)$, denoted by $\rho(\nabla^2\phi(x))$, is bounded on D. We get a dc decomposition for ϕ as:

$$\phi(x) = \left(\frac{\rho^*}{2}\|x\|^2\right) - \left(\frac{\rho^*}{2}\|x\|^2 - \phi(x)\right) \tag{24}$$

where ρ^* is a constant verifying $\rho^* \geq \max_{x\in D} \rho(\nabla^2\phi(x))$. Obviously, $\frac{\rho^*}{2}\|x\|^2$ and $\frac{\rho^*}{2}\|x\|^2 - \phi(x)$ are both convex functions on D.

As stated in [31, 43], in order to obtain a dc decomposition for the nonconvex homogeneous polynomial functions f_1 and f_2, we first compute their Hessian matrices as follows:

1. Gradient and Hessian of f_1:

$$\nabla f_1(x,y,z,\lambda) = \begin{bmatrix} \nabla_x f_1(x,y,z,\lambda) \\ \nabla_y f_1(x,y,z,\lambda) \\ \nabla_z f_1(x,y,z,\lambda) \\ \nabla_\lambda f_1(x,y,z,\lambda) \end{bmatrix} = \begin{bmatrix} -2\lambda y \\ -2\lambda(x+z) \\ -2\lambda y \\ -2y^\top(x+z) \end{bmatrix},$$

$$\nabla^2 f_1(x,y,z,\lambda) = \begin{bmatrix} 0 & -2\lambda I & 0 & -2y \\ -2\lambda I & 0 & -2\lambda I & -2(x+z) \\ 0 & -2\lambda I & 0 & -2y \\ -2y^\top & -2(x+z)^\top & -2y^\top & 0 \end{bmatrix}.$$

2. Gradient and Hessian of f_2:

$$\nabla f_2(x,y,\lambda) = \begin{bmatrix} \nabla_x f_2(x,y,\lambda) \\ \nabla_y f_2(x,y,\lambda) \\ \nabla_\lambda f_2(x,y,\lambda) \end{bmatrix} = \begin{bmatrix} 2\lambda^2 x \\ 2\lambda^2 y \\ 2\lambda(\|x\|^2 + \|y\|^2) \end{bmatrix},$$

$$\nabla^2 f_2(x,y,\lambda) = \begin{bmatrix} 2\lambda^2 I & 0 & 4\lambda x \\ 0 & 2\lambda^2 I & 4\lambda y \\ 4\lambda x^\top & 4\lambda y^\top & 2(\|x\|^2 + \|y\|^2) \end{bmatrix}.$$

Let us denote $\|x\|_1$ and $\|x\|_\infty$ the 1-norm and infinity norm of the vector $x \in \mathbb{R}^n$ respectively. Let $\rho(\nabla^2 f_1) : \mathbb{R}^{3n+1} \to \mathbb{R}$ and $\rho(\nabla^2 f_2) : \mathbb{R}^{2n+1} \to \mathbb{R}$ be functions defined by

$$\rho(\nabla^2 f_1)(x,y,z,\lambda) = \rho(\nabla^2 f_1(x,y,z,\lambda)),$$
$$\rho(\nabla^2 f_2)(x,y,\lambda) = \rho(\nabla^2 f_2(x,y,\lambda)).$$

The bounds of $\rho(\nabla^2 f_1)$ and $\rho(\nabla^2 f_2)$ on $\mathcal{C}$ are given by the following theorem:

Theorem 4. *$\rho(\nabla^2 f_1)$ and $\rho(\nabla^2 f_2)$ are bounded on $\mathcal{C}$ by*

$$\rho(\nabla^2 f_1) \le 2\max\{|\lambda| + \|y\|_\infty, \|x+z\|_\infty + 2|\lambda|, 2\|y\|_1 + \|x+z\|_1\},$$
$$\rho(\nabla^2 f_2) \le 2\max\{\lambda^2 + 2\|\lambda x\|_\infty, \lambda^2 + 2\|\lambda y\|_\infty, \|(x,y)\|^2 + 2(\|\lambda x\|_1 + \|\lambda y\|_1)\}.$$

Proof. These inequalities are due to the fact that $\rho(A) \le \|A\|_1$ where $\|A\|_1$ stands for the induced 1-norm of the matrix A defined by $\|A\|_1 = \max_{j\in[\![1,n]\!]} \sum_{i=1}^n |A_{ij}|$ (in which $[\![1,n]\!]$ stands for the set $\{1,\dots,n\}$). Thus it is sufficient to compute $\|\nabla^2 f_1\|_1$ and $\|\nabla^2 f_2\|_1$ with $(x,y,z,\lambda) \in \mathcal{C}$, which yield the required inequalities.

It follows from Theorem 4 that the functions $\rho(\nabla^2 f_1)$ and $\rho(\nabla^2 f_2)$ are bounded on a bounded set $\mathcal{C}$. It is known [13] that the number of complementary eigenvalues of QEiCP(A,B,C) is finite. So, if QEiCP(A,B,C) has a solution, then there exist l and u in $\mathbb{R}$ such that

$$l \le \lambda \le u$$

for all complementary eigenvalues λ.

Theorem 5. *If QEiCP(A,B,C) has a solution (λ, x) with $\lambda \in [l,u]$, then any optimal solution of (P) satisfies:*

$$x \in [0,1]^n, y \in [\min\{0,l\}, \max\{0,u\}]^n, z \in [0,p^2]^n, e^\top z \le p^2, w \in [0,\bar{w}]$$

with $p = \max\{|l|,|u|\}$ and $\bar{w} = [\bar{w}_i]_{i\in[\![1,n]\!]}$ is given by

$$\bar{w}_i = p^2\sqrt{\sum_{j=1}^n A_{ij}^2} + p\sqrt{\sum_{j=1}^n B_{ij}^2} + \sqrt{\sum_{j=1}^n C_{ij}^2}, \forall i \in [\![1,n]\!].$$

Proof. The proofs for the bounds of x,y,z are similar to the proposition 1 in [31]. So we only have to prove that $e^\top z \le p^2$ and the new upper bound $\bar{w}$ for w.

(i) Based on $z = \lambda^2 x, e^\top x = 1$ and $\lambda \in [l,u]$, it follows that $e^\top z = \lambda^2 \le p^2$.

(ii) The upper bound of w can be obtained from the definition of w as $Az + By + Cx$. Since x,y,z are all bounded, then $w_i, \forall i \in [\![1,n]\!]$ is also bounded. In fact,

$$w_i \le |w_i| = \left|\sum_{j=1}^n A_{ij}z_j + \sum_{j=1}^n B_{ij}y_j + \sum_{j=1}^n C_{ij}x_j\right|$$
$$\le \left|\sum_{j=1}^n A_{ij}z_j\right| + \left|\sum_{j=1}^n B_{ij}y_j\right| + \left|\sum_{j=1}^n C_{ij}x_j\right|.$$

According to the Cauchy-Schwartz inequality, we get respectively that

$$\left|\sum_{j=1}^{n} A_{ij} z_j\right| \leq \|z\| \sqrt{\sum_{j=1}^{n} A_{ij}^2} = \lambda^2 \|x\| \sqrt{\sum_{j=1}^{n} A_{ij}^2} \leq \max\{l^2, u^2\} \sqrt{\sum_{j=1}^{n} A_{ij}^2},$$

$$\left|\sum_{j=1}^{n} B_{ij} y_j\right| \leq \|y\| \sqrt{\sum_{j=1}^{n} B_{ij}^2} = \|\lambda\| \|x\| \sqrt{\sum_{j=1}^{n} B_{ij}^2} \leq \max\{|l|, |u|\} \sqrt{\sum_{j=1}^{n} B_{ij}^2},$$

and

$$\left|\sum_{j=1}^{n} C_{ij} x_j\right| \leq \|x\| \sqrt{\sum_{j=1}^{n} C_{ij}^2} \leq \sqrt{\sum_{j=1}^{n} C_{ij}^2}.$$

Thus, we have $\forall i \in [\![1, n]\!]$,

$$\begin{aligned} w_i &\leq \max\{l^2, u^2\} \sqrt{\sum_{j=1}^{n} A_{ij}^2} + \max\{|l|, |u|\} \sqrt{\sum_{j=1}^{n} B_{ij}^2} + \sqrt{\sum_{j=1}^{n} C_{ij}^2} \\ &= p^2 \sqrt{\sum_{j=1}^{n} A_{ij}^2} + p \sqrt{\sum_{j=1}^{n} B_{ij}^2} + \sqrt{\sum_{j=1}^{n} C_{ij}^2} = \bar{w}_i. \end{aligned}$$

Remark 2. The vector $\bar{w}$ provides a tighter bound for w than the one given in [31], and its computation requires the knowledge of the bounds l and u for λ. Formulas for computing these two bounds l and u are introduced in section 7.

Since the solution set of (P) is bounded, if we add the bound obtained in Theorem 5 into the problem (P), the resulting problem should be equivalent to (P). Let us consider the compact convex polyhedral set

$$\begin{aligned} \hat{\mathcal{C}} = \{(x, y, z, w, \lambda) : w = Az + By + Cx, e^\top x = 1, e^\top y = \lambda, e^\top z \leq p^2, \lambda \in [l, u], \\ x \in [0, 1]^n, w \in [0, \bar{w}], y \in [\min\{0, l\}, \max\{0, u\}]^n, z \in [0, p^2]^n\}. \end{aligned} \tag{25}$$

So the following problem

$$(\hat{P}) \qquad \min\{f(x, y, z, w, \lambda) : (x, y, z, w, \lambda) \in \hat{\mathcal{C}}.\}$$

has the same optimal solution set as (P) when $\rho(\nabla^2 f_1)$ and $\rho(\nabla^2 f_2)$ are bounded.

If we assume that $C \notin S_0$, then $\lambda = 0$ cannot be a solution of QEiCP and this problem has at least a posyive and a negative complementarity eigenvalues [28]. So $\hat{C}$ can be reduced into two subsets regarding to λ positive or negative. If $\lambda \geq 0$, then $y = \lambda x \geq 0$, and (x, y, z, w, λ) in $\hat{C}$ should be nonnegative. Otherwise, $\lambda \leq 0$, then $y \leq 0$ and (x, z, w) is still nonnegative. Let us define the two subsets

$$\begin{aligned} \hat{\mathcal{C}_1} = \{(x, y, z, w, \lambda) : w = Az + By + Cx, e^\top x = 1, e^\top y = \lambda, \\ e^\top z \leq p^2, \lambda \in [0, u], x \in [0, 1]^n, w \in [0, \bar{w}], y \in [0, u]^n, z \in [0, p^2]^n\}. \end{aligned} \tag{26}$$

$$\begin{aligned} \hat{\mathcal{C}_2} = \{(x, y, z, w, \lambda) : w = Az + By + Cx, e^\top x = 1, e^\top y = \lambda, \\ e^\top z \leq p^2, \lambda \in [l, 0], x \in [0, 1]^n, w \in [0, \bar{w}], y \in [l, 0]^n, z \in [0, p^2]^n\}. \end{aligned} \tag{27}$$

Obviously, problem $(\hat{P})$ is equivalent to problem:

$$\min\{f(x, y, z, w, \lambda) : (x, y, z, w, \lambda) \in \hat{\mathcal{C}_1} \cup \hat{\mathcal{C}_2}\} \tag{28}$$

which means that, for seeking a positive (resp. negative) complementary eigenvalue, we just need to solve the problem $(\hat{P})$ on $\hat{\mathcal{C}_1}$ (resp. $\hat{\mathcal{C}_2}$).

The next theorem provides upper bounds for $\rho(\nabla^2 f_1)$ and $\rho(\nabla^2 f_2)$ on $\hat{\mathcal{C}_1}$ and $\hat{\mathcal{C}_2}$.

Theorem 6. *On $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$, we have*

$$\rho(\nabla^2 f_1) \leq \rho_1, \rho(\nabla^2 f_2) \leq \rho_2$$

with $\rho_1 = 2(p+1)^2$, $\rho_2 = 6p^2 + 4p + 2$ and $p = \max\{|l|, |u|\}$.

Proof. Since $|\lambda| \leq \max\{|l|, |u|\} = p$ both on $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$, it follows from Theorem 4 that $\rho(\nabla^2 f_1) \leq 2\max\{|\lambda| + \|y\|_\infty, \|x+z\|_\infty + 2|\lambda|, 2\|y\|_1 + \|x+z\|_1\}$. According to Theorem 5, we get

$$|\lambda| \leq p, \|x\|_1 = 1, \|x\| \leq 1, \|x\|_\infty \leq 1.$$

$$\|y\|_1 = \sum_{i=1}^{n} |y_i| = \left(\begin{cases} e^\top y & , \text{on } \hat{\mathcal{C}}_1 \\ -e^\top y & , \text{on } \hat{\mathcal{C}}_2 \end{cases}\right) \leq |\lambda| \leq p.$$

$$\|y\| \leq p, \|y\|_\infty \leq p, \|z\|_1 \leq p^2, \|z\|_\infty \leq p^2.$$

$$\|x+z\|_\infty \leq \|x\|_\infty + \|z\|_\infty \leq 1 + p^2 \text{ and } \|x+z\|_1 \leq \|x\|_1 + \|z\|_1 \leq 1 + p^2.$$

Hence,

$$\rho(\nabla^2 f_1) \leq 2\max\{2p, 1+p^2+2p, 1+p^2+2p\} = 2(p+1)^2 = \rho_1.$$

Similarly,

$$\rho(\nabla^2 f_2) \leq 2\max\{\lambda^2 + 2\|\lambda x\|_\infty, \lambda^2 + 2\|\lambda y\|_\infty, \|x\|^2 + \|y\|^2 + 2(\|\lambda x\|_1 + \|\lambda y\|_1)\}$$

$$\leq 2\max\{p^2 + 2p, 3p^2, 3p^2 + 2p + 1\} = 6p^2 + 4p + 2 = \rho_2.$$

Remark 3. These upper bounds ρ_1 and ρ_2 for spectral radius are tighter than those given in our previous work [31].

Finally, we can obtain a similar dc decomposition for f_1 and f_2 as in [31] but with smaller values for ρ_1 and ρ_2 as:

$$f_1(x,y,z,\lambda) = \frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2 - \left(\frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2 - f_1(x,y,z,\lambda)\right). \tag{29}$$

$$f_2(x,y,\lambda) = \frac{\rho_2}{2}\|(x,y,\lambda)\|^2 - \left(\frac{\rho_2}{2}\|(x,y,\lambda)\|^2 - f_2(x,y,\lambda)\right). \tag{30}$$

150 where $\frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2 - f_1(x,y,z,\lambda)$ and $\frac{\rho_2}{2}\|(x,y,\lambda)\|^2 - f_2(x,y,\lambda)$ are both locally convex functions on $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$, and $\frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2$ and $\frac{\rho_2}{2}\|(x,y,\lambda)\|^2$ are quadratic convex functions.

Hence, we get from (14)–(16), (29) and (30) a dc decomposition of f as:

$$f(x,y,z,w,\lambda) = \hat{g}(x,y,z,w,\lambda) - \hat{h}(x,y,z,w,\lambda)$$

with

$$\begin{cases} \hat{g}(x,y,z,w,\lambda) = \frac{\|x+w\|^2}{4} + \|(y,z)\|^2 + \frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2 + \frac{\rho_2}{2}\|(x,y,\lambda)\|^2, \\ \hat{h}(x,y,z,w,\lambda) = \hat{g}(x,y,z,w,\lambda) - f(x,y,z,w,\lambda), \end{cases} \tag{31}$$

which yields the following dc programming formulation of $(\hat{P})$:

DC Programming Formulation 2: $(\hat{P}_{DC})$

$$\min\{\hat{g}(x,y,z,w,\lambda) - \hat{h}(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \hat{\mathcal{C}}_1(\text{or } \hat{\mathcal{C}}_2)\} \tag{32}$$

with $\hat{g}$ and $\hat{h}$ defined in (31), and $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$ are given by (26) and (27).

6. DC programming formulations of (P')

In a similar fashion to the two previous sections, we can obtain two different dc programming formulations for (P'). The only difference between (P) and (P') is related with the last term of the objective function where the bilinear term $f_3(x,w) = w^\top x$ is replaced by the concave polyhedral function $\tilde{f}_3(x,w) = \sum_{i=1}^n \min\{x_i, w_i\}$. This concave function $\tilde{f}_3$ has an explicit dc decomposition as:

$$\tilde{f}_3(x,w) = \sum_{i=1}^n \min\{x_i, w_i\} = (0) - (-\sum_{i=1}^n \min\{x_i, w_i\}) \tag{33}$$

where $-\sum_{i=1}^n \min\{x_i, w_i\}$ is a convex polyhedral function.

Thus we can do exactly in the same way as in (P) and $(\hat{P})$ by only replacing f_3 by $\tilde{f}_3$ to get the nonlinear programming formulations (P') and $(\hat{P}')$ as well as their corresponding dc programming formulations.

For problem

$$(P') \qquad \min\{f'(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \mathcal{C}\},$$

we can generate a dc decomposition for f' as

$$f'(x,y,z,w,\lambda) = g'(x,y,z,w,\lambda) - h'(x,y,z,w,\lambda) \tag{34}$$

where

$$\begin{cases} g'(x,y,z,w,\lambda) = & \|y\|^2 + \|z\|^2 + \frac{(\lambda^2+\|x\|^2)^2+(\lambda^2+\|y\|^2)^2}{2} \\ & + \frac{(4\lambda^2+4+\|y+x\|^2+\|y+z\|^2)^2+(4(\lambda+1)^2+\|y-x\|^2+\|y-z\|^2)^2}{32}, \\ h'(x,y,z,w,\lambda) = & -\sum_{i=1}^n \min\{x_i, w_i\} + \frac{2\lambda^4+\|x\|^4+\|y\|^4}{2} \\ & + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)^2+(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)^2}{32}. \end{cases} \tag{35}$$

Hence, a dc programming formulation of (P') is given by:

DC Programming Formulation 3: (P'_{DC})

$$\min\{g'(x,y,z,w,\lambda) - h'(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \mathcal{C}\} \tag{36}$$

where $\mathcal{C}$ is defined in (12) and g' and h' are given by (35).

We can also get an equivalent problem $(\hat{P}')$ similar to $(\hat{P})$ as:

$$(\hat{P}') \qquad \min\{f'(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \hat{\mathcal{C}}\}.$$

By the universal dc decomposition, a dc decomposition for f' is given by:

$$f'(x,y,z,w,\lambda) = \hat{g}'(x,y,z,w,\lambda) - \hat{h}'(x,y,z,w,\lambda)$$

with

$$\begin{cases} \hat{g}'(x,y,z,w,\lambda) = \|(y,z)\|^2 + \frac{\rho_1}{2}\|(x,y,z,\lambda)\|^2 + \frac{\rho_2}{2}\|(x,y,\lambda)\|^2, \\ \hat{h}'(x,y,z,w,\lambda) = \hat{g}'(x,y,z,w,\lambda) - f'(x,y,z,w,\lambda), \end{cases} \tag{37}$$

which yields the following dc programming formulation of $(\hat{P}')$:

DC Programming Formulation 4: $(\hat{P}'_{DC})$

$$\min\{\hat{g}'(x,y,z,w,\lambda) - \hat{h}'(x,y,z,w,\lambda) : (x,y,z,w,\lambda) \in \hat{\mathcal{C}}_1 (\text{or } \hat{\mathcal{C}}_2)\} \tag{38}$$

with $\hat{g}'$ and $\hat{h}'$ defined in (37), and $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$ are given by (26) and (27).

7. Lower and upper bounds for λ

The dc formulations described in the previous sections require the existence of lower and upper bounds for the variable λ. In this section, we discuss the computation of such bounds for QEiCP with a matrix $A \in$ PD and satisfying the co-hyperbolic condition or $C \notin S_0$.

7.1. Lower and upper bounds of λ when $A \in PD$ and co-hyperbolicity

The bounds are given in the following theorems:

Theorem 7 (see [31]). *If $A \in PD$ and the co-hyperbolic condition holds, the λ-component of any solution of QEiCP satisfies*

$$l = \beta - \sqrt{\alpha} \leq \lambda \leq \gamma + \sqrt{\alpha} = u$$

with

$$\alpha = \max\{\gamma^2, \beta^2\} + \max_{i,j}\{-C_{ij}\}/s,$$

$$\beta = \begin{cases} \min\{-B_{ij}\}/(2\max\{A_{ij}\}), & \textit{if } \min\{-B_{ij}\} > 0; \\ \min\{-B_{ij}\}/(2s), & \textit{if } \min\{-B_{ij}\} \leq 0, \end{cases}$$

$$\gamma = \begin{cases} \max\{-B_{ij}\}/(2s), & \textit{if } \max\{-B_{ij}\} > 0; \\ \max\{-B_{ij}\}/(2\max\{A_{ij}\}), & \textit{if } \max\{-B_{ij}\} \leq 0, \end{cases}$$

in which $s = \min\{x^\top Ax : e^\top x = 1, x \geq 0\}$.

Proof. The reader is referred to the proof given in [31].

Based on a similar idea used in Theorem 7, we get a new bound for λ in the next result:

Theorem 8. *Suppose that $A \in PD$ and QEiCP satisfies the co-hyperbolic condition. Let $\lambda_{\max}(M)$ (resp. $\lambda_{\min}(M)$) be the largest (resp. the smallest) real eigenvalue of the matrix M. The λ-component of any solution of QEiCP satisfies*

$$l = \beta - \sqrt{\alpha} \leq \lambda \leq \gamma + \sqrt{\alpha} = u$$

with $a = \lambda_{\min}(A + A^\top)/2$, $\bar{a} = \lambda_{\max}(A + A^\top)/2$, $b = \lambda_{\min}(-B - B^\top)/2$, $\bar{b} = \lambda_{\max}(-B - B^\top)/2$, $\bar{c} = \lambda_{\max}(-C - C^\top)/2$, $\alpha = \max\{\gamma^2, \beta^2\} + \bar{c}/a$,

$$\beta = \begin{cases} b/(2\bar{a}) & , \textit{if } b > 0; \\ b/(2a) & , \textit{if } b \leq 0, \end{cases} \quad \gamma = \begin{cases} \bar{b}/(2a) & , \textit{if } \bar{b} > 0; \\ \bar{b}/(2\bar{a}) & , \textit{if } \bar{b} \leq 0. \end{cases}$$

Proof. The proof is almost the same as in Theorem 7. The major difference is to use the set $V = \{x \in \mathbb{R}^n : \|x\| = 1, x \geq 0\}$ instead of the set U used in the proof of this theorem [31]. The reason for introducing the constraint $e^\top x = 1$ is to avoid a null vector to be a solution of QEiCP. Obviously, $\forall M \in \mathcal{M}_n$ and $x \in V$, we have

$$x^\top Mx = x^\top \frac{M + M^\top}{2} x.$$

Since $(M + M^\top)/2$ is a symmetric matrix, then it is diagonalizable and the quadratic form defined on V is bounded by

$$\lambda_{\min}\left(\frac{M + M^\top}{2}\right) \leq x^\top \frac{M + M^\top}{2} x \leq \lambda_{\max}\left(\frac{M + M^\top}{2}\right), \forall x \in V.$$

Therefore, we can use the above inequality to get our new bounds of λ following the same procedure used in the proof of Theorem 7.

Remark 4. Note that another bound of λ has been provided in [13], and it seems interesting to compare these two bounds. Theorem 5 shows that the bounds of (x, y, z, w) depend on the bounds of λ, and the tighter bound of λ is, the tighter the bounds of (w, y, z, w) are.

7.2. Lower and upper bounds of λ for $A \in PD$ and $C \notin S_0$

In this case, it is proved that QEiCP(A, B, C) has both positive and negative complementary eigenvalues [23]. We just need to focus on the bounds for a positive complementary eigenvalue, since the negative one can be done in a similar way.

Theorem 9 ([28]). *Let $A \in PD$, $C \notin S_0$ and define the following programs*

$$(LP) \qquad \min\{e^\top v + e^\top y : Av + By + Cx \geq 0, e^\top y + e^\top x = 1, (x, y, v) \geq 0\}$$

and

$$(UP) \qquad \max\{\frac{p^\top y}{y^\top Ay + x^\top x} : e^\top y + e^\top x = 1, (x, y) \geq 0\}$$

where p is a vector with components $p_i = 1 + \sum_{j=1}^{n}(\max\{0, B_{ij}\} + \max\{0, C_{ij}\}), \forall i \in [\![1, n]\!]$. Then

(i) The linear program (LP) has an optimal solution with positive optimal value which is a lower bound of λ.

(ii) Any stationary point of (UP) is a global maximum of (UP) and its optimal value is an upper bound of λ.

8. DC algorithms for solving dc programming formulations

In this section, we investigate how to use DCA for solving the dc programming formulations (P_{DC}), $(\hat{P}_{DC})$, (P'_{DC}) and $(\hat{P}'_{DC})$.

8.1. DCA for solving (P_{DC})

Since the function h defined in (22) is differentiable, then $\partial h(x, y, z, w, \lambda)$ is reduced to a singleton $\{\nabla h(x, y, z, w, \lambda)\}$. The following fixed-point scheme describes the major computations in dc algorithm for (P_{DC})

$$\begin{array}{l}(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) \in \operatorname{argmin}\{g(x, y, z, w, \lambda) \\ -\langle (x, y, z, w, \lambda), \nabla h(x^k, y^k, z^k, w^k, \lambda^k)\rangle : (x, y, z, w, \lambda) \in \mathcal{C}\}\end{array} \tag{39}$$

where g is defined in (22), $\mathcal{C}$ is defined in (12), and $\langle u, v\rangle$ represents inner product of two vectors u and v. Since this problem is a convex polynomial optimization, then any KKT solution is a global optimal solution.

The gradient of h can be computed as follows:

$$\nabla h(x, y, z, w, \lambda) = \begin{bmatrix} \nabla_x h(x, y, z, w, \lambda) \\ \nabla_y h(x, y, z, w, \lambda) \\ \nabla_z h(x, y, z, w, \lambda) \\ \nabla_w h(x, y, z, w, \lambda) \\ \nabla_\lambda h(x, y, z, w, \lambda) \end{bmatrix} \tag{40}$$

where

$$\left\{\begin{array}{ll} \nabla_x h(x, y, z, w, \lambda) = & \frac{x-w}{2} + 2\|x\|^2 x + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(x-y)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(x+y)}{8}. \\ \nabla_y h(x, y, z, w, \lambda) = & 2\|y\|^2 y + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(2y-x-z)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(2y+x+z)}{8}. \\ \nabla_z h(x, y, z, w, \lambda) = & \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(z-y)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(z+y)}{8}. \\ \nabla_w h(x, y, z, w, \lambda) = & \frac{w-x}{2}. \\ \nabla_\lambda h(x, y, z, w, \lambda) = & 4\lambda^3 + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)\lambda}{2} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(\lambda+1)}{2}. \end{array}\right. \tag{41}$$

The steps of DCA for solving (P_{DC}) are described in Algorithm 1:

Algorithm 1 DCA for solving (P_{DC})

```
Inputs: Initial point $(x^0, y^0, z^0, w^0, \lambda^0)$, the tolerance for optimal value $\epsilon_1 > 0$, the tolerance for optimal solution $\epsilon_2 > 0$, and the tolerance for globality $\epsilon_3 > 0$.
Outputs: Optimal solution $(x^*, y^*, z^*, w^*, \lambda^*)$ and optimal value $f^*$.

Set $k = 0$, $f^* = +\infty$, $\Delta f = +\infty$ and $\Delta X = +\infty$.
while ($\Delta f > \epsilon_1$ and $\Delta X > \epsilon_2$ and $f^* > \epsilon_3$) do
Step 1:   Compute $\nabla h(x^k, y^k, z^k, w^k, \lambda^k)$ by formula (41).
Step 2:   Compute $(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1})$ by solving the convex optimization problem (39).
Step 3:   Compute
              $\Delta f \leftarrow |f(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) - f(x^k, y^k, z^k, w^k, \lambda^k)|.$
              $\Delta X \leftarrow \|(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) - (x^k, y^k, z^k, w^k, \lambda^k)\|.$
              $(x^*, y^*, z^*, w^*, \lambda^*) \leftarrow (x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}).$
              $f^* \leftarrow f(x^*, y^*, z^*, w^*, \lambda^*).$
          Update $k \leftarrow k + 1$.
end while
return $(x^*, y^*, z^*, w^*, \lambda^*)$ and $f^*$.
```

This algorithm should terminate if one of the following stopping criteria is satisfied:

1. The sequence $\{f(x^k, y^k, z^k, w^k, \lambda^k)\}$ converges, i.e., $\Delta f \leq \epsilon_1$.
2. The sequence $\{(x^k, y^k, z^k, w^k, \lambda^k)\}$ converges, i.e., $\Delta X \leq \epsilon_2$.
3. The *sufficient global optimality condition* holds, i.e., $f^* \leq \epsilon_3$.

We have the following convergence theorem for DCA:

Theorem 10 (Convergence theorem for DCA). *DCA for solving (P_{DC}) generates convergent sequences $\{(x^k, y^k, z^k, w^k, \lambda^k)\}$ and $\{f(x^k, y^k, z^k, w^k, \lambda^k)\}$ such that*

(i) The sequence $\{f(x^k, y^k, z^k, w^k, \lambda^k)\}$ is decreasing and bounded below.

(ii) The sequence $\{(x^k, y^k, z^k, w^k, \lambda^k)\}$ converges either to an approximate solution of QEiCP satisfying the sufficient global optimality condition, or to an approximate solution of general KKT point of (P_{DC}).

Proof. This theorem is a consequence of the general convergence theorem for DCA [29, 36]. The sufficient global optimality condition is based on Theorem 2, which states that the optimal value of (P_{DC}) is zero for any solution of QEiCP.

Remark 5. If the sequence $f(x^k, y^k, z^k, w^k, \lambda^k)$ does not converge to zero, it may be caused by one of the following reasons:

(i) QEiCP is infeasible.

(ii) The computed solution found by DCA is a KKT point of (P_{DC}) which is not a global minimum.

Note that it is very easy to show whether QEiCP is infeasible since this verification reduces to the solution of a linear program [24]. If QEiCP is feasible and DCA terminates in the second case, then we should combine it with a global optimization solver in order to find a solution of QEiCP. The hybrid enumerative approach method described in [28] can be modified in order to incorporate with DCA. This should be a topic for future research.

8.2. *DCA for solving* $(\hat{P}_{DC})$, (P'_{DC}) *and* $(\hat{P}'_{DC})$

This method is Algorithm 1 with slightly differences in steps 1 and 2, which are discussed below.

Step 1 for $(\hat{P}_{DC})$. Since $\hat{h}$ defined in (31) is differentiable, $\partial\hat{h}(x,y,z,w,\lambda)$ is reduced to a singleton $\{\nabla\hat{h}(x,y,z,w,\lambda)\}$, which is given by:

$$\begin{aligned}\nabla\hat{h}(x,y,z,w,\lambda) &= \nabla(\hat{g}-f)(x,y,z,w,\lambda)\\ &= \begin{bmatrix} \frac{x+w}{2} + (\rho_1+\rho_2)x + 2\lambda y - 2\lambda^2 x - w \\ (\rho_1+\rho_2-2\lambda^2)y + 2\lambda(x+z) \\ \rho_1 z + 2\lambda y \\ \frac{w-x}{2} \\ (\rho_1+\rho_2-2(\|x\|^2+\|y\|^2))\lambda + 2y^\top(x+z) \end{bmatrix}.\end{aligned} \tag{42}$$

Step 1 for (P'_{DC}). In this case, h' defined in (35) is non-differentiable. Hence $\partial h'(x,y,z,w,\lambda)$ is an nonempty convex set, which is computed by:

$$\partial h'(x,y,z,w,\lambda) = \begin{bmatrix} \partial_x h'(x,y,z,w,\lambda) \\ \partial_y h'(x,y,z,w,\lambda) \\ \partial_z h'(x,y,z,w,\lambda) \\ \partial_w h'(x,y,z,w,\lambda) \\ \partial_\lambda h'(x,y,z,w,\lambda) \end{bmatrix} \tag{43}$$

where

$$\begin{cases} \partial_x h'(x,y,z,w,\lambda) = & -u + 2\|x\|^2 x + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(x-y)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(x+y)}{8}. \\ \partial_y h'(x,y,z,w,\lambda) = & 2\|y\|^2 y + \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(2y-x-z)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(2y+x+z)}{8}. \\ \partial_z h'(x,y,z,w,\lambda) = & \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)(z-y)}{8} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(z+y)}{8}. \\ \partial_w h'(x,y,z,w,\lambda) = & -v. \\ \partial_\lambda h'(x,y,z,w,\lambda) = & \frac{(4\lambda^2+4+\|y-x\|^2+\|y-z\|^2)\lambda}{2} \\ & + \frac{(4(\lambda+1)^2+\|y+x\|^2+\|y+z\|^2)(\lambda+1)}{2}. \end{cases} \tag{44}$$

and

$$\begin{cases} u = [u_i]_{i\in[\![1,n]\!]}, \text{ with } u_i \in \begin{cases} \{1\}, & x_i < w_i; \\ \{0,1\}, & x_i = w_i; \\ \{0\}, & x_i > w_i. \end{cases} \\ v = [v_i]_{i\in[\![1,n]\!]}, \text{ with } v_i \in \begin{cases} \{0\}, & x_i < w_i; \\ \{0,1\}, & x_i = w_i; \\ \{1\}, & x_i > w_i. \end{cases} \end{cases} \tag{45}$$

Step 1 for $(\hat{P}'_{DC})$. The function $\hat{h}'$ defined in (37) is also non-differentiable. So $\partial\hat{h}'(x,y,z,w,\lambda)$ is an non-empty convex set which could be computed by:

$$\begin{aligned}\partial\hat{h}'(x,y,z,w,\lambda) &= \partial(\hat{g}'-f')(x,y,z,w,\lambda)\\ &= \begin{bmatrix} (\rho_1+\rho_2)x + 2\lambda y - 2\lambda^2 x - u \\ (\rho_1+\rho_2-2\lambda^2)y + 2\lambda(x+z) \\ \rho_1 z + 2\lambda y \\ -v \\ (\rho_1+\rho_2-2(\|x\|^2+\|y\|^2))\lambda + 2y^\top(x+z) \end{bmatrix}.\end{aligned} \tag{46}$$

with u and v defined in (45).

Step 2 for $(\hat{P}_{DC})$***,*** (P'_{DC}) ***and*** $(\hat{P}'_{DC})$. As stated before, step 2 of Algorithm 1 requires solving a convex optimization problem of the form (39). Let us write this problem as follows:

$$(P^k) \qquad u^{k+1} \in \operatorname{argmin}\{g(u) - \langle u, v^k\rangle : u \in D\}$$

where u, u^k, v^k and D are defined in (39). The definitions of these vectors and set D for applying Algorithm 1 to $(\hat{P}_{DC})$, (P'_{DC}) $(\hat{P}'_{DC})$ are given in Table 1 below:

Table 1: Step 2 of DCA for $(\hat{P}_{DC})$, (P'_{DC}) and $(\hat{P}'_{DC})$

(P^k)	$(\hat{P}_{DC})$	(P'_{DC})	$(\hat{P}'_{DC})$
g	$\hat{g}$	g'	$\hat{g}'$
u	(x,y,z,w,λ)	(x,y,z,w,λ)	(x,y,z,w,λ)
u^{k+1}	$(x^{k+1},y^{k+1},z^{k+1},w^{k+1},\lambda^{k+1})$	$(x^{k+1},y^{k+1},z^{k+1},w^{k+1},\lambda^{k+1})$	$(x^{k+1},y^{k+1},z^{k+1},w^{k+1},\lambda^{k+1})$
v^k	$\nabla\hat{h}(x^k,y^k,z^k,w^k,\lambda^k)$	$\partial h'(x^k,y^k,z^k,w^k,\lambda^k)$	$\partial\hat{h}'(x^k,y^k,z^k,w^k,\lambda^k)$
D	$\hat{\mathcal{C}}$	$\mathcal{C}$	$\hat{\mathcal{C}}$

8.3. Solution of convex optimization problem in Step 2 of DCA

The problem (P^k) to be solved in step 2 is a convex polynomial optimization problem for (P_{DC}) and $(\hat{P}_{DC})$, and a convex quadratic program for (P'_{DC}) and $(\hat{P}'_{DC})$. Since KKT conditions are necessary and sufficient global optimality conditions for convex optimization, thus these problems can be solved efficiently via polynomial time algorithms such as Interior-Point Methods (IPM) [44]. In practice, we suggest to use GUROBI [35], CPLEX [34] and IPOPT [33] for solving the convex quadratic programs required for DCA when applied to (P'_{DC}) and $(\hat{P}'_{DC})$. For the convex polynomial cases in (P_{DC}) and $(\hat{P}_{DC})$, we propose to reformulate them as a convex quadratic program with convex quadratic constraints. In fact, our experience has shown that solving a high-order convex polynomial optimization with IPM (such as using IPOPT [33] or MATLAB `fmincon` [45]) is much more difficult in practice than solving a convex quadratic program with convex quadratic constraints. Next, we show how to make such a reformulation.

Since the function g in (P_{DC}) has the form

$$\begin{aligned} g(x,y,z,w,\lambda) = \; & \|y\|^2 + \|z\|^2 + \tfrac{\|x+w\|^2}{4} + \tfrac{(\lambda^2+\|x\|^2)^2+(\lambda^2+\|y\|^2)^2}{2} \\ & + \tfrac{(4\lambda^2+4+\|y+x\|^2+\|y+z\|^2)^2+(4(\lambda+1)^2+\|y-x\|^2+\|y-z\|^2)^2}{32} \end{aligned}$$

then by introducing an additional vector $t \in \mathbb{R}^8$ and the convex quadratic constraints:

$$\begin{cases} \|x\|^2 \le t_1, \\ \|y\|^2 \le t_2, \\ \|x+y\|^2 \le t_3, \\ \|y+z\|^2 \le t_4, \\ \|y-x\|^2 \le t_5, \\ \|y-z\|^2 \le t_6, \\ \lambda^2 \le t_7, \\ (\lambda+1)^2 \le t_8, \end{cases} \tag{47}$$

we obtain the following equivalent convex quadratic program with convex quadratic constraints

$$\begin{aligned} &(x^{k+1},y^{k+1},z^{k+1},w^{k+1},\lambda^{k+1}) \in \operatorname{argmin}\{\bar{g}(x,y,z,w,\lambda,t) \\ &-\langle (x,y,z,w,\lambda), \nabla h(x^k,y^k,z^k,w^k,\lambda^k)\rangle : (x,y,z,w,\lambda) \in \mathcal{C}, (47)\}, \end{aligned} \tag{48}$$

where

$$\begin{aligned} \bar{g}(x,y,z,w,\lambda,t) = \; & t_2 + \|z\|^2 + \|x+w\|^2/4 + (t_7+t_1)^2/2 + (t_7+t_2)^2/2 + \\ & (4t_7+4+t_3+t_4)^2/32 + (4t_8+t_5+t_6)^2/32. \end{aligned} \tag{49}$$

This optimization problem can be solved efficiently by an IPM code such as CPLEX, GUROBI or IPOPT.

8.4. Initial point estimation

Finding a good initial point for DCA is an open question which is highly depending on the specific structure of the dc program. For the case where $A \in$ PD and QEiCP satisfies the co-hyperbolic condition, we propose the following procedure to compute a potentially good initial point:

$$\begin{cases} x^0 \in \operatorname{argmin}\{x^\top Ax : e^\top x = 1, x \geq 0\}, \\ \lambda^0 = (-{x^0}^\top Bx^0 \pm \sqrt{({x^0}^\top Bx^0)^2 - 4({x^0}^\top Ax^0)({x^0}^\top Cx^0)})/(2{x^0}^\top Ax^0), \\ y^0 = \lambda^0 x^0, \\ z^0 = \lambda^0 y^0, \\ w^0 = Az^0 + By^0 + Cx^0. \end{cases} \tag{50}$$

Since $A \in$ PD, then x^0 is computed by solving a strictly convex quadratic program. This problem can be efficiently solved by an IPM [44] or a simple extension of the Block Principal Pivoting algorithm discussed in [46].

After computing x^0 and due to co-hyperbolic condition, then λ^0 can be found by

$$\lambda^0 = \left(-{x^0}^\top Bx^0 \pm \sqrt{({x^0}^\top Bx^0)^2 - 4({x^0}^\top Ax^0)({x^0}^\top Cx^0)}\right)/(2{x^0}^\top Ax^0). \tag{51}$$

Then, by the definitions of y, z and w, we compute $y^0 = \lambda^0 x^0, z^0 = \lambda^0 y^0$ and $w^0 = Az^0 + By^0 + Cx^0$.

Note that if $w^0 \geq 0$, then this initial point is also a solution of QEiCP, and DCA starting with this point does not perform any iteration. If the co-hyperbolic condition is not satisfied, then it maybe impossible to compute λ^0 by the formula (51). If in this case $C \notin S_0$, then we should choose λ^0 as an arbitrary number belonging to the interval $[l, u]$ discussed in subsection 7.2.

8.5. Local dc decomposition

The major differences among dc programming formulations (P_{DC}), (P'_{DC}), $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ is that $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ are based on two parameters ρ_1 and ρ_2 given in Theorem 6, while (P_{DC}) and (P'_{DC}) are based on DCSOS decompositions without a parameter. In virtue of Definition 1, it is easy to conclude that for problems $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$, the smaller ρ_1 and ρ_2 are, the better dc decomposition is. Moreover, since ρ_1 and ρ_2 defined in Theorem 6 depend on the bounds of λ, thus we can further conclude that the tighter bounds of λ are, the better dc decomposition is. This explains why we were so interested in seeking a tighter bound for λ in section 7.

In order to find a better dc decomposition than (31) and (37), we propose a *local dc decomposition* strategy that is explained below.

Suppose that we have found $\lambda^k \in [l, u]$ at k-th iteration of DCA. Then for the $(k+1)$-th iteration of DCA, we can restricted λ in a smaller interval such as

$$\lambda \in [\lambda^k - a, \lambda^k + a] \subset [l, u]$$

for some suitable parameter $a \geq 0$. For example, we can take $a = \min\{1, (\lambda^k - l)/2, (u - \lambda^k)/2\}$. Now consider the set

$$\begin{aligned} \hat{\mathcal{C}}^k = \quad & \{(x, y, z, w, \lambda) \in \hat{\mathcal{C}} : \lambda \in [\lambda^k - a, \lambda^k + a], z \in [0, p_k]^n, e^\top z \leq (p_k)^2, \\ & y \in [\min\{0, \lambda^k - a\}, \max\{0, \lambda^k + a\}]^n\}. \end{aligned} \tag{52}$$

where $p_k = \max\{|\lambda^k - a|, |\lambda^k + a|\}$. We can update ρ_1 and ρ_2 on $\hat{\mathcal{C}}^k$ as follows:

$$\begin{cases} \rho_1^k = 2(p_k + 1)^2, \\ \rho_2^k = 6p_k^2 + 4p_k + 2. \end{cases} \tag{53}$$

Clearly, $0 \leq p^k \leq p$ implies $\rho_1^k \leq \rho_1$ and $\rho_2^k \leq \rho_2$. Therefore, the dc decompositions (31) and (37) defined with updated parameters ρ_1^k and ρ_2^k on $\hat{\mathcal{C}}^k$ should be better than the ones defined with ρ_1 and ρ_2 on $\hat{\mathcal{C}}^k$. Using this idea, we propose a new DCA Algorithm 2 for solving $(\hat{P}_{DC})$.

Algorithm 2 DCA with a local dc decomposition strategy for solving ($\hat{P}_{DC}$)

```
**Inputs**: Given $(x^0, y^0, z^0, w^0, \lambda^0)$ with $\lambda^0 \in [l, u]$, and $\epsilon_1 > 0$, $\epsilon_2 > 0$, $\epsilon_3 > 0$ are the tolerances mentioned in Algorithm 1.
**Outputs**: Optimal solution $(x^*, y^*, z^*, w^*, \lambda^*)$ and optimal value $f^*$.

Set $k = 0$, $f^* = +\infty$, $\Delta f = +\infty$ and $\Delta X = +\infty$.
while ($\Delta f > \epsilon_1$ and $\Delta X > \epsilon_2$ and $f^* > \epsilon_3$) do
  Step 1:  Compute $a = \min\{1, (\lambda^k - l)/2, (u - \lambda^k)/2\}$.
  Step 2:  Compute $\hat{C}^k$ via formula (52).
  Step 3:  Compute $\hat{h}(x^k, y^k, z^k, w^k, \lambda^k)$ via formula (42).
  Step 4:  Solve convex optimization problem
             $(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) \in \operatorname{argmin}\{\hat{g}(x, y, z, w, \lambda) - \langle (x, y, z, w, \lambda), \nabla \hat{h}(x^k, y^k, z^k, w^k, \lambda^k) \rangle : (x, y, z, w, \lambda) \in \hat{C}^k\}$.
  Step 5:  Compute
             $\Delta f \leftarrow |f(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) - f(x^k, y^k, z^k, w^k, \lambda^k)|$.
             $\Delta X \leftarrow \|(x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1}) - (x^k, y^k, z^k, w^k, \lambda^k)\|$.
             $(x^*, y^*, z^*, w^*, \lambda^*) \leftarrow (x^{k+1}, y^{k+1}, z^{k+1}, w^{k+1}, \lambda^{k+1})$.
             $f^* \leftarrow f(x^*, y^*, z^*, w^*, \lambda^*)$.
           Update $k \leftarrow k + 1$.
end while
return $(x^*, y^*, z^*, w^*, \lambda^*)$ and $f^*$.
```

Accordingly, for ($\hat{P}'_{DC}$), we can replace $\hat{g}$ by $\hat{g}'$, $\nabla \hat{g}$ by $\partial \hat{g}'$, and f by f' in Algorithm 2 to get a DCA with local dc decomposition.

Remark 6. Algorithm 2 has the same convergence properties of Algorithm 1 stated in Theorem 10. Moreover, since Algorithm 2 has a better dc decomposition than Algorithm 1 for ($\hat{P}_{DC}$) and ($\hat{P}'_{DC}$). So, it is expected Algorithm 2 to have a better numerical performance, which will be confirmed by the numerical results to be shown in the next section. Note that it is still hard to say whether Algorithm 1 for (P_{DC}) and (P'_{DC}) is better than Algorithm 2 for ($\hat{P}_{DC}$) and ($\hat{P}'_{DC}$). The performances of all these algorithms will be reported in the same section.

9. Experimental results

In this section, we report some numerical results of our algorithms implemented and tested on MATLAB 2016b [45]. We use Yalmip [47] to build our optimization models. Yalmip provides us a convenient interface to call many solvers such as GUROBI, CPLEX and IPOPT in MATLAB for solving SOCPs.

Our tests are performed on Dell Workstation equipped with i7-6820HQ 2.70GHz CPU, 32GB RAM, 64 bits Windows 10. We have used QEiCP test problems defined in [5, 23, 13, 28]. The first problem denoted by SeegerAdlyQ(3) has been taken from [5]. In all the remaining problems, the matrix A is the identity matrix and the matrices B and C are randomly generated with elements uniformly distributed in the intervals $[0, 1]$, $[0, 10]$, and $[0, 100]$.

9.1. Computing bounds for λ

Firstly, we made some experiments for computing the three bounds estimations of λ given in Theorems 7 and 8 and in [13] in order to find the tightest bound in practice. The numerical results of these experiments are reported in Table 2.

In this table and in the remaining ones, we use the following notations:

- n: order of the matrices A, B and C.
- Bounds 1,2 and 3: bounds of λ in Theorems 7 and 8 and in [13].
- l: lower bound for λ.
- u: upper bound for λ.
- IT: number of iterations of DCA.
- CPU: cpu time (in seconds).

Table 2: Lower and upper bounds of λ computed by three different procedures

PROB	n	Bounds 1		Bounds 2		Bounds 3	
		l	u	l	u	l	u
SeegerAdlyQ(3)	3	−15.197	7.697	−5.313	1.813	−10.875	5.469
Rand(0,1,05)	5	−5.705	3.243	−3.540	2.513	−4.944	2.669
Rand(0,1,10)	10	−10.695	5.807	−5.652	3.746	−9.345	4.903
Rand(0,1,20)	20	−20.953	10.951	−10.911	6.630	−19.596	10.042
Rand(0,1,30)	30	−30.959	15.963	−16.238	9.549	−29.585	15.037
Rand(0,1,40)	40	−40.971	20.972	−21.271	12.270	−39.555	20.022
Rand(0,1,50)	50	−50.961	25.971	−26.103	14.918	−49.273	24.886
Rand(0,10,05)	5	−47.863	24.205	−22.840	15.606	−42.789	21.607
Rand(0,10,10)	10	−99.508	50.230	−56.698	34.105	−95.230	47.858
Rand(0,10,20)	20	−199.347	100.162	−96.851	56.714	−188.383	94.447
Rand(0,10,30)	30	−300.109	150.553	−150.278	86.672	−289.527	145.014
Rand(0,10,40)	40	−400.669	200.829	−198.833	111.831	−389.041	194.772
Rand(0,10,50)	50	−500.785	250.891	−253.829	140.857	−489.202	244.850
Rand(0,100,05)	5	−485.358	242.858	−242.185	149.519	−439.463	219.978
Rand(0,100,10)	10	−997.611	497.982	−513.415	305.980	−930.600	465.548
Rand(0,100,20)	20	−1994.165	997.308	−962.099	570.225	−1863.996	932.255
Rand(0,100,30)	30	−2999.717	1500.358	−1502.535	854.526	−2906.336	1453.417
Rand(0,100,40)	40	−3999.291	2000.068	−2038.123	1155.180	−3893.380	1946.940
Rand(0,100,50)	50	−4999.169	2500.071	−2492.850	1385.791	−4896.833	2448.667

We observe in Table 2 that the Bounds 2 computed via Theorem 8 are the tightest bounds for λ and are much better than Bounds 1 and 3. Thus, we decide to use these Bounds 2 to perform the remaining simulations.

9.2. Testing DCA without local dc decomposition

We have tested the performance (CPU time and number of iterations) of DCA for solving (P_{DC}), (P'_{DC}), $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ without local dc decomposition. The convex quadratic subproblems were solved by GUROBI. The numerical results are summarized in Table 3 with all $\epsilon_{i\in[\![1,3]\!]} = \epsilon = 10^{-3}$ and in Table 4 with all $\epsilon_{i\in[\![1,3]\!]} = \epsilon = 10^{-3}$. The starting points are computed by the heuristic described in subsection 8.4. The values in AVG and STD stand for the average and the standard deviation of the corresponding column.

Based on the set of problems that we have tested, we observe in Table 3 that all these algorithms got very similar computed eigenvalues, but their performances are quite different. The AVG in iterations and CPU time for DCSOS formulations ((P_{DC}) and (P'_{DC})) are smaller than those for universal dc formulations ($(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$), and this difference becomes particularly evident in Table 4 when ϵ is decreased. The STD in iterations and CPU time for DCSOS formulations are less sensitive than for the universal dc decompositions with respect to both the increase of problem size and the decrease of tolerance ϵ. For instance, DCA for $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ perform very slowly with many iterations, but in some other cases such as Rand$(0, 10, 30)$

Table 3: Numerical results of DCA for (P_{DC}), (P'_{DC}), $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ with $\epsilon = 10^{-3}$

PROB	DCA for (P_{DC})			DCA for $(\hat{P}_{DC})$			DCA for (P'_{DC})			DCA for $(\hat{P}'_{DC})$		
	λ	IT	CPU	λ	IT	CPU	λ	IT	CPU	λ	IT	CPU
SeegerAdlyQ(3)	−0.717	6	0.67	−0.713	5	0.34	−0.718	5	0.59	−0.713	3	0.23
Rand(0,1,05)	0.703	11	1.34	0.680	8	0.64	0.706	10	1.25	0.717	39	3.13
Rand(0,1,10)	0.752	7	1.03	0.777	10	0.83	0.753	7	1.05	0.759	37	3.26
Rand(0,1,20)	0.937	7	1.57	0.960	2	0.18	0.935	6	1.37	0.954	75	7.12
Rand(0,1,30)	0.915	6	2.49	0.937	2	0.20	0.927	6	2.42	0.932	137	14.96
Rand(0,1,40)	0.954	7	4.41	0.971	2	0.23	0.970	7	4.47	0.979	185	23.76
Rand(0,1,50)	0.951	6	6.50	0.972	2	0.30	0.976	8	7.96	0.977	209	33.41
Rand(0,10,05)	0.876	10	2.00	0.918	229	30.93	0.880	10	2.05	0.999	2	0.27
Rand(0,10,10)	0.961	17	3.92	0.911	2	0.28	0.929	17	3.81	0.910	14	1.96
Rand(0,10,20)	1.145	18	5.37	1.180	2	0.29	1.171	16	5.00	1.180	9	1.76
Rand(0,10,30)	1.074	24	12.12	1.074	2	0.32	1.104	19	9.46	1.074	2	0.34
Rand(0,10,40)	1.047	16	12.74	1.087	2	0.42	1.097	12	9.93	1.087	2	0.45
Rand(0,10,50)	1.035	18	22.67	1.049	2	0.51	1.078	13	15.89	1.049	14	3.65
Rand(0,100,05)	1.297	39	14.37	1.117	378	88.90	1.203	31	11.56	1.127	2	0.48
Rand(0,100,10)	0.848	10	3.91	1.015	33	8.04	0.949	8	3.20	1.016	9	2.17
Rand(0,100,20)	1.097	12	5.84	1.146	2	0.50	1.105	17	8.58	1.146	2	0.52
Rand(0,100,30)	1.043	9	6.32	1.047	2	0.55	1.061	12	8.48	1.047	2	0.56
Rand(0,100,40)	0.972	10	9.40	1.007	2	0.59	0.990	9	8.86	1.007	2	0.63
Rand(0,100,50)	1.071	10	14.03	1.049	2	0.67	1.045	8	11.40	1.049	2	0.72
AVG		12.8	6.88		36.3	7.09		11.6	6.18		39.3	5.23
STD		7.89	5.74		95.05	20.48		6.11	4.30		63.42	8.83

Table 4: Numerical results of DCA for (P_{DC}), (P'_{DC}), $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ with $\epsilon = 10^{-4}$

PROB	DCA for (P_{DC})			DCA for $(\hat{P}_{DC})$			DCA for (P'_{DC})			DCA for $(\hat{P}'_{DC})$		
	λ	IT	CPU	λ	IT	CPU	λ	IT	CPU	λ	IT	CPU
SeegerAdlyQ(3)	−0.722	13	1.38	−0.713	5	0.38	−0.723	11	1.27	−0.713	3	0.25
Rand(0,1,05)	0.712	32	3.94	0.720	176	14.66	0.711	32	4.03	0.726	91	7.71
Rand(0,1,10)	0.746	21	2.97	0.757	195	17.15	0.747	20	3.03	0.759	129	11.84
Rand(0,1,20)	0.940	26	5.71	0.951	181	17.15	0.934	18	4.00	0.956	130	13.17
Rand(0,1,30)	0.928	40	16.04	0.937	12	1.45	0.941	20	7.94	0.935	197	23.04
Rand(0,1,40)	0.959	22	13.82	0.971	2	0.24	0.976	16	9.63	0.982	254	33.99
Rand(0,1,50)	0.966	27	25.86	0.972	2	0.31	0.997	33	30.02	0.984	388	64.56
Rand(0,10,05)	0.857	36	7.28	0.882	633	88.54	0.854	22	4.56	0.966	1808	260.74
Rand(0,10,10)	0.984	71	15.88	0.894	2437	340.51	0.957	42	9.15	0.910	14	1.96
Rand(0,10,20)	1.186	65	18.75	1.167	1023	150.53	1.175	40	11.80	1.180	10	1.49
Rand(0,10,30)	1.161	112	50.37	1.067	927	151.06	1.183	72	32.96	1.074	2	0.36
Rand(0,10,40)	1.038	46	33.48	1.087	2	0.46	1.115	32	23.52	1.087	2	0.46
Rand(0,10,50)	1.047	63	69.56	1.049	2	0.51	1.139	51	54.25	1.049	15	4.02
Rand(0,100,05)	1.323	71	24.17	1.112	6289	1473.80	1.228	95	32.51	1.127	2	0.48
Rand(0,100,10)	0.866	21	7.70	0.962	9610	2297.14	0.955	31	11.36	1.016	10	2.43
Rand(0,100,20)	1.100	28	12.40	1.134	10000	2640.69	1.142	49	23.22	1.146	2	0.53
Rand(0,100,30)	1.046	19	12.60	1.042	4565	1311.74	1.076	36	22.80	1.047	2	0.58
Rand(0,100,40)	0.967	24	21.49	1.007	2	0.62	0.992	25	22.33	1.007	2	0.64
Rand(0,100,50)	1.073	20	25.52	1.049	2	0.71	1.054	23	28.11	1.049	2	0.74
AVG		39.8	19.42		1898.2	447.77		35.2	17.71		161.2	22.58
STD		24.87	16.60		3184.12	811.22		20.03	13.57		402.06	58.29

and Rand$(0, 100, 50)$, DCA for $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ has a very fast convergence. This unstable issue for the universal dc decomposition seems much more visible for most of the test problems when the tolerance ϵ is decreased (see Table 4 for ϵ equal to 10^{-4}). This sensitivity issue is probably caused by the impact of the parameters ρ_1 and ρ_2, since a bigger ρ leads to a worse dc decomposition and yields a bad quality in DCA. Thus, their performances are hopefully to be improved by using local dc decompositions since $\rho_i^k \leq \rho, i = 1, 2$.

9.3. Testing DCA with local dc decompositions

Table 5 reports the numerical results with local dc decompositions for $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ with $\epsilon = 10^{-4}$. We use GUROBI for solving the required convex quadratic programs with convex quadratic constraints. For the test problem SeegerAdlyQ(3), we use the set $\hat{\mathcal{C}}_2$ in the formulations $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$. The remaining test problems were solved by employing $\hat{\mathcal{C}}_1$.

Table 5: Numerical results of DCA for $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ with local dc decomposition and $\epsilon = 10^{-4}$

PROB	DCA for $(\hat{P}_{DC})$			DCA for $(\hat{P}'_{DC})$		
	λ	IT	CPU	λ	IT	CPU
SeegerAdlyQ(3)	−0.718	39	3.86	−0.718	39	4.37
Rand(0,1,05)	0.722	74	8.24	0.722	70	8.25
Rand(0,1,10)	0.750	57	7.54	0.754	50	7.04
Rand(0,1,20)	0.949	48	6.87	0.953	45	6.76
Rand(0,1,30)	0.923	46	7.06	0.944	43	7.62
Rand(0,1,40)	0.971	62	10.65	0.995	44	9.03
Rand(0,1,50)	0.962	46	8.88	1.000	46	10.85
Rand(0,10,05)	0.834	83	10.71	0.864	85	11.27
Rand(0,10,10)	0.985	144	19.82	0.966	143	20.75
Rand(0,10,20)	1.141	73	11.12	1.210	109	18.72
Rand(0,10,30)	1.164	114	19.83	1.209	135	26.77
Rand(0,10,40)	1.066	50	9.66	1.102	62	14.43
Rand(0,10,50)	1.069	64	13.48	1.126	116	30.19
Rand(0,100,05)	1.290	276	41.56	1.251	97	14.76
Rand(0,100,10)	0.826	85	13.41	0.954	73	11.96
Rand(0,100,20)	1.195	111	19.46	1.225	211	40.44
Rand(0,100,30)	1.047	185	35.67	1.115	104	23.05
Rand(0,100,40)	0.972	64	13.65	1.042	73	18.25
Rand(0,100,50)	1.070	69	16.33	1.070	65	18.23
AVG		88.9	14.62		84.7	15.93
STD		56.96	9.40		43.00	9.04

By comparing these results with those reported in Table 4, we observed that, in most of tested cases, the AVG for iterations and CPU time of DCA with local dc decomposition are much smaller than those for the algorithms without local dc decomposition. The STD values for local dc decomposition are also better than those obtained without local dc decomposition. Therefore, we can conclude that when local decomposition is used, the performances of DCA for universal dc decompositions $(\hat{P}_{DC})$ and $(\hat{P}'_{DC})$ can be improved. Moreover, the fastest and most stable algorithms with respect to AVG and STD values appear to be still the DCA for DCSOS formulations. This good performance for QEiCP indicates that the DCSOS decompositions for polynomial function should be considered as a promising technique to improve the performance of DCA for solving other polynomial optimization problems.

10. Conclusions

In this paper, we have proposed several dc programming formulations for Quadratic Eigenvalue Complementarity Problem. A DCSOS decomposition for polynomial function, and a local dc decomposition technique should be considered as the most important contributions. The corresponding DCAs have been tested and the numerical results show a good performance for our methods. Moreover, we discuss a new insight in recognizing the most important key point for characterizing the quality of a dc decomposition, and propose a potentially good initial point for QEiCP. We believe that these contributions will be important tools for helping to answer these two most common open questions that arise in dc programming.

Acknowledgments

The research of Yi-Shuai Niu is funded by the Natural Science Foundation of China (Grant No: 11601327) and by the Key Construction National "985" Program of China (Grant No: WF220426001). The research of Joaquim Júdice was partially supported in the scope of R&D Unit UID/EEA/5008/2013, financed by the applicable financial framework (FCT/MEC) through national funds and when applicable co-funded by FEDER-PT2002 Partnership Agreement.

[1] A. Seeger, Quadratic eigenvalue problems under conic constraints, SIAM Journal on Matrix Analysis and Applications 32 (2011) 700–721.

[2] A. Seeger, Eigenvalue analysis of equilibrium processes defined by linear complementaritv conditions, Linear Algebra and Its Applications 294 (1999) 1–14.

[3] J. J. Júdice, H. D. Sherali, I. Ribeiro, S. Rosa, On the asymmetric eigenvalue complementarity problem, Optimization Methods and Software 24 (2009) 549–586.

[4] S. Adly, H. Rammal, A new method for solving second-order cone eigenvalue complementarity problems, Journal of Optimization Theory and Applications 165 (2) (2015) 563–585.

[5] S. Adly, A. Seeger, A non-smooth algorithm for cone constrained eigenvalue problems, Computational Optimization and Applications 49 (2) (2011) 299–318.

[6] C. P. Brás, A. Fischer, J. J. Júdice, K. Schönefeld, S. Seifert, A block active set algorithm with spectral choice line search for the symmetric eigenvalue complementarity problem, Applied Mathematics & Computation 294 (2017) 36–48.

[7] C. P. Brás, M. Fukushima, J. J. Júdice, S. Rosa, Variational inequality formulation for the asymmetric eigenvalue complementarity problem and its solution by means of a gap function, Pacific Journal of Optimization 8 (2012) 197–215.

[8] Z. M. Chen, L. Q. Qi, A semismooth newton method for tensor eigenvalue complementarity problem, Computational Optimization & Applications 65 (1) (2016) 109–126.

[9] A. P. D. Costa, J. A. C. Martins, I. N. Figueiredo, J. J. Júdice, The directional instability problem in systems with frictional contacts, Computer Methods in Applied Mechanics and Engineering 193 (2004) 357–384.

[10] A. P. D. Costa, A. Seeger, Cone constrained eigenvalue problems, theory and algorithms, Computational Optimization and Applications 45 (2010) 25–57.

[11] A. P. D. Costa, A. Seeger, F. M. F. Simões, Complementarity eigenvalue problems for nonlinear matrix pencils, Applied Mathematics & Computation 312 (2017) 134–148.

[12] F. Facchinei, J. S. Pang, Finite-dimensional Variational Inequalities and Complementarity Problems, Springer Science & Business Media, 2007.

[13] L. M. Fernandes, J. J. Júdice, H. D. Sherali, M. Fukushima, On the computation of all eigenvalues for the eigenvalue complementarity problem, Journal of Global Optimization 59 (2014) 307–326.

[14] L. M. Fernandes, M. Fukushima, J. J. Júdice, H. D. Sherali, The second-order cone eigenvalue complementarity problem, Optimization Methods and Software 31 (2016) 24–52.

[15] R. Fernandes, J. J. Júdice, V. Trevisan, Complementary eigenvalues of graphs, Linear Algebra & Its Applications 527 (2017) 216–231.

[16] A. N. Iusem, J. J. Júdice, V. Sessa, P. Sarabando, Splitting methods for the eigenvalue complementarity problem, to appear in Optimization Methods and Software.

[17] J. J. Júdice, M. Raydan, S. Rosa, S. Santos, On the solution of the symmetric complementarity problem by the spectral projected gradient method, Numerical Algorithms 37 (2008) 391–407.

[18] H. A. LeThi, M. Moeini, D. T. Pham, J. J. Júdice, A dc programming approach for solving the symmetric eigenvalue complementarity problem, Computational Optimization and Applications 51 (2012) 1097–1117.

[19] Y. S. Niu, H. A. LeThi, D. T. Pham, J. J. Júdice, Efficient dc programming approaches for the asymmetric eigenvalue complementarity problem, Optimization Methods and Software 28 (2013) 812–829.

[20] A. Patrascu, I. Necoara, Efficient random coordinate descent algorithms for large-scale structured nonconvex optimization, Journal of Global Optimization 61 (1) (2015) 19–46.

[21] L. Zhang, C. Shen, M. Yang, J. J. Júdice, A lanczos method for large-scale extreme lorentz eigenvalue problems, SIAM Journal on Matrix Analysis and Applications 39 (2) (2018) 611–631.

[22] Y. H. Zhou, M. S. Gowda, On the finiteness of the cone spectrum of certain linear transformations on euclidean jordan algebras, Linear Algebra and Its Applications 431 (5) (2009) 772–782.

[23] C. P. Brás, A. N. Iusem, J. J. Júdice, On the quadratic eigenvalue complementarity problem, Journal of Global Optimization 66 (2) (2016) 153–171.

[24] R. Cottle, J. S. Pang, R. E. Stone, The Linear Complementarity Problem, Academic Press, New York, 1992.

[25] C. P. Brás, M. Fukushima, A. N. Iusem, J. J. Júdice, On the quadratic eigenvalue complementarity problem over a general convex cone, Applied Mathematics & Computation 271 (2015) 594–608.

[26] L. M. Fernandes, J. J. Júdice, M. Fukushima, A. Iusem, On the symmetric quadratic eigenvalue complementarity problem, Optimization Methods and Software 29 (2014) 751–770.

[27] A. N. Iusem, J. J. Júdice, V. Sessa, H. Sherali, The second-order cone quadratic eigenvalue complementarity problem, Pacific Journal of Optimization 13 (2017) 475–500.

[28] A. N. Iusem, J. J. Júdice, V. Sessa, H. D. Sherali, On the numerical solution of the quadratic eigenvalue complementarity problem, Numerical Algorithms 72 (2016) 721–747.

[29] D. T. Pham, H. A. LeThi, Dc optimization algorithms for solving the trust region subproblem, SIAM Journal on Optimization 8 (1998) 476–507.

[30] H. A. LeThi, D. T. Pham, Dc programming and dca: thirty years of developments, Math. Program., Ser. B (2018) 1–64.

[31] Y. S. Niu, J. J. Júdice, H. A. LeThi, D. T. Pham, Solving the quadratic eigenvalue complementarity problem by dc programming, Modelling, Computation and Optimization in Information Systems and Management Sciences, Advances in Intelligent Systems and Computing 359 (2015) 203–214.

[32] Y. S. Niu, On difference of sos decompositions and difference of sos convex decompositions for polynomials.
URL `arXiv:1803.09900`

[33] A. Wächter, L. T. Biegler, On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming, Mathematical Programming 106 (1) (2006) 25–57.

[34] IBM, Ibm ilog cplex optimization studio 12.7.
URL `https://www.ibm.com/us-en/marketplace/ibm-ilog-cplex`

[35] G. Optimization, Gurobi 7.0.
URL `http://www.gurobi.com/`

[36] D. T. Pham, H. A. LeThi, The dc programming and dca revisited with dc models of real world nonconvex optimization problems, Annals of Operations Research 133 (2005) 23–46.

[37] R. T. Rockafellar, Convex Analysis, Princeton University Press, Princeton, 1970.

[38] Y. S. Niu, Programmation DC & DCA en Optimisation Combinatoire et Optimisation Polynomiale via les Techniques de SDP, Ministère de l'enseignement supérieur et de la recherche, Institut National Des Sciences Appliquées de Rouen, France, 2010.

[39] Y. S. Niu, D. T. Pham, Dc programming approaches for bmi and qmi feasibility problems, Advanced Computational Methods for Knowledge Engineering: Proceedings of the 2nd International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA 2014) 282 (2014) 37–63.

[40] D. T. Pham, H. A. LeThi, Dc programming. theory, algorithms, applications: The state of the art, First International Whorkshop on Global Constrained Optimization and Constraint Satisfaction, Nice.

[41] R. Horst, P. M. Pardalos, V. T. Nguyen, Introduction to Global Optimization, 2nd Edition, Springer US, 2000.

[42] A. A. Ahmadi, G. Hall, Dc decomposition of nonconvex polynomials with algebraic techniques, Math. Program., Ser.B (2017) 1–26.

[43] D. T. Pham, Y. S. Niu, An efficient dc programming approach for portfolio decision with higher moments, Computational Optimization and Applications 50 (3) (2011) 525–554.

[44] S. Boyd, L. Vandenberghe, Convex Optimization, 1st Edition, Cambridge University Press, 2004.

[45] MathWorks, Matlab documentation.
URL `http://www.mathworks.com/help/matlab/`

[46] J. J. Júdice, F. M. Pires, A block principal pivoting algorithm for large-scale strictly monotone linear complementarity problems, Computers & Operations Research 21 (1994) 587–596.

[47] J. Löfberg, Yalmip: A toolbox for modeling and optimization in matlab, Proceedings of the CACSD Conference, Taipei, Taiwan (2004).
URL `https://yalmip.github.io/`