

The Boosted DC Algorithm for linearly constrained DC programming

F. J. Aragón Artacho* R. Campoy† P. T. Vuong‡

August 3, 2022

Dedicated to Professor Miguel A. Goberna on the occasion of his 70th birthday

Abstract

The Boosted Difference of Convex functions Algorithm (BDCA) has been recently introduced to accelerate the performance of the classical Difference of Convex functions Algorithm (DCA). This acceleration is achieved thanks to an extrapolation step from the point computed by DCA via a line search procedure. In this work, we propose an extension of BDCA that can be applied to difference of convex functions programs with linear constraints, and prove that every cluster point of the sequence generated by this algorithm is a Karush–Kuhn–Tucker point of the problem if the feasible set has a Slater point. When the objective function is quadratic, we prove that any sequence generated by the algorithm is bounded and R-linearly (geometrically) convergent. Finally, we present some numerical experiments where we compare the performance of DCA and BDCA on some challenging problems: to test the copositivity of a given matrix, to solve one-norm and infinity-norm trust-region subproblems, and to solve piecewise quadratic problems with box constraints. Our numerical results demonstrate that this new extension of BDCA outperforms DCA.

Keywords: Difference of convex functions; boosted difference of convex functions algorithm; global convergence; constrained DC programs; copositivity problem; trust region subproblem.

1 Introduction

In this paper, we are interested in solving the following DC (difference of convex functions) optimization problem:

$$\begin{cases} \min_{x \in \mathbb{R}^n} & \phi(x) := g(x) - h(x) \\ \text{s.t.} & \langle a_i, x \rangle \leq b_i, \quad i = 1, \dots, p, \end{cases} \quad (\mathcal{P})$$

*Department of Mathematics, University of Alicante, Alicante, Spain.

Email: francisco.aragon@ua.es

†Department of Statistics and Operational Research, Universitat de València, Valencia, Spain.

Email: ruben.campoy@uv.es

‡Mathematical Sciences School, University of Southampton, UK

Email: t.v.phan@soton.ac.uk

where $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$, $h : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ are proper, closed, and convex functions with g being smooth, $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$ for $i = 1, \dots, p$, and $\langle \cdot, \cdot \rangle$ denotes an inner product. We use the conventions:

$$\begin{aligned} (+\infty) - (+\infty) &= +\infty, \\ (+\infty) - \lambda &= +\infty \quad \text{and} \quad \lambda - (+\infty) = -\infty, \quad \forall \lambda \in]-\infty, +\infty[. \end{aligned}$$

Observe that we can rewrite problem (\mathcal{P}) as an unconstrained nonsmooth DC optimization problem, whose objective function is $g + \iota_{\mathcal{F}} - h$, where $\iota_{\mathcal{F}}$ denotes the indicator function of the feasible set

$$\mathcal{F} := \{x \in \mathbb{R}^n \mid \langle a_i, x \rangle \leq b_i, i = 1, \dots, p\}.$$

For solving this problem, one can apply the classical DC Algorithm (DCA) in [28, 15]. DC programming and the DCA have been developed and studied for more than 30 years [15]. The DCA has been successfully applied in different fields such as machine learning, financial optimization, supply chain management, and telecommunication, see, e.g. [17, 11, 24]. Nowadays, DCA has become a useful method to solve nonconvex problems.

To accelerate the convergence of DCA, which can be slow for some problems, a new method called *Boosted DC Algorithm* (BDCA) has been recently proposed in [1, 3]. The key idea of BDCA is to perform an extrapolation step via a line search procedure at the point computed by DCA at each iteration. This step allows the algorithm to take longer steps than the classical DCA, achieving in this way a larger reduction of the objective value per iteration. In addition to accelerating its convergence, BDCA may have better chances to escape from bad local optima thanks to the line search procedure, see [3, Example 3.3]. Therefore, BDCA is not only faster than DCA but also can provide better solutions. Extensive numerical experiments in diverse applications such as biochemistry [1], machine learning [36, 22] and data science [3] have been performed where BDCA clearly outperforms DCA. Note that all these applications are modeled as DC problems with g smooth. However, it is important to emphasize that, for unconstrained DC programs, the BDCA proposed in [1, 3] is not applicable when the function g in (\mathcal{P}) is nonsmooth (see [3, Example 3.4]).

The aim of this paper is to show that BDCA can still be applied if the nonsmooth function g is the sum of a smooth convex function and the indicator function of a polyhedral set. More precisely, we will show that it is possible to use BDCA for solving DC programs with linear constraints of the form (\mathcal{P}) . To compare the performance of DCA and BDCA, we provide numerical experiments on various challenging problems: to test the copositivity of a given matrix, to solve ℓ_1 and ℓ_∞ trust-region subproblems, and to find the minimum of a piecewise quadratic function with box constraints. The three first problems have a quadratic objective function and are known to be NP-hard [23] (the first was already heuristically investigated in [8] using DCA), while the last problem has a nonsmooth objective function. Our results confirm that BDCA significantly outperforms DCA in these applications.

The rest of this paper is organized as follows. Section 2 recalls some preliminary results. In Section 3, we propose a new variant of BDCA for solving (\mathcal{P}) and prove that the objective value of the sequence generated by the algorithm is monotonically decreasing, and that any limit point of the sequence is a KKT point of the problem. The R-linear convergence of any sequence generated by BDCA in the special case of quadratic objective functions is derived in Section 4. In Section 5, we provide some numerical experiments for testing

the copositivity of a given matrix, for solving ℓ_1 and ℓ_∞ trust-region subproblems and for solving piecewise quadratic problems with box constraints, where we compare BDCA and DCA. Finally, some conclusions and future research are briefly discussed in Section 6.

2 Preliminaries

In this section, we state our assumptions imposed on (\mathcal{P}) . We also recall some preliminary and basic results which will be used in the sequel.

Let $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be a proper extended real-valued convex function. The set $\text{dom } f := \{x \in \mathbb{R}^n \mid f(x) < +\infty\}$ denotes its (effective) *domain*, and

$$\partial f(x) := \{w \in \mathbb{R}^n \mid f(y) \geq f(x) + \langle w, y - x \rangle, \forall y \in \mathbb{R}^n\}$$

denotes the *subdifferential* of f at x . If f is differentiable at x , then $\partial f(x) = \{\nabla f(x)\}$, where $\nabla f(x)$ stands for the *gradient* of f at x . The one-side *directional derivative* of f at x with respect to the direction $d \in \mathbb{R}^n$ is

$$f'(x; d) := \lim_{t \searrow 0} \frac{f(x + td) - f(x)}{t}.$$

Recall that f is said to be *strongly convex* with *strong convexity parameter* $\rho > 0$ if the function $f - \frac{\rho}{2} \|\cdot\|^2$ is convex. The *conjugate* of f is the function $f^* : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ defined at $u \in \mathbb{R}^n$ as

$$f^*(u) = \sup_{x \in \mathbb{R}^n} \{\langle x, u \rangle - f(x)\}.$$

It holds that $f^{**} := (f^*)^* = f$. Moreover, for all $x, u \in \mathbb{R}^n$,

$$u \in \partial f(x) \Leftrightarrow f(x) + f^*(u) = \langle x, u \rangle \Leftrightarrow x \in \partial f^*(u).$$

Given a set $C \subseteq \mathbb{R}^n$, its *interior*, its *relative interior* and the *convex cone generated by* C are denoted by $\text{int } C$, $\text{ri } C$ and $\text{cone } C$, respectively. The function

$$\iota_C(x) := \begin{cases} 0, & \text{if } x \in C, \\ +\infty, & \text{otherwise;} \end{cases}$$

denotes the *indicator function* of C , and C is convex if and only if ι_C is so. Moreover, its subdifferential $\partial \iota_C$ is the *normal cone* to C , which is given by

$$N_C(x) := \begin{cases} \{u \in \mathbb{R}^n : \langle u, c - x \rangle \leq 0, \forall c \in C\}, & \text{if } x \in C, \\ \emptyset, & \text{otherwise.} \end{cases}$$

For proving our convergence results, we will make use of the following assumptions.

Assumption 1. Both g and h are strongly convex on their domain with the same strong convexity parameter $\rho > 0$.

Assumption 2. The function h is subdifferentiable at every point in $\text{dom } h$; i.e., $\partial h(x) \neq \emptyset$ for all $x \in \text{dom } h$. The function g is continuously differentiable on an open set containing $\text{dom } h$ and

$$\inf_{x \in \mathcal{F}} \phi(x) > -\infty. \quad (1)$$

Assumption 3. The feasible set \mathcal{F} has a Slater point, that is, there exists $\hat{x} \in \mathbb{R}^n$ such that $\langle a_i, \hat{x} \rangle < b_i$, for all $i = 1, \dots, p$.

Remark 2.1. Assumption 1 is not restrictive, in the sense that any DC decomposition of ϕ as $\phi = g - h$, can be expressed as $\phi = (g + \frac{\rho}{2} \|\cdot\|^2) - (h + \frac{\rho}{2} \|\cdot\|^2)$ for any $\rho > 0$. Observe that $\partial h(x) \neq \emptyset$ holds for all $x \in \text{ri dom } h$ (by [34, Theorem 23.4]), so the first part of Assumption 2 is clearly satisfied if $\text{dom } h = \mathbb{R}^n$. A key point of our method is the smoothness of g in Assumption 2, which cannot be in general omitted (see [3, Example 3.4]).

Associated with problem (\mathcal{P}) , we can construct its *dual* as

$$\inf_{u \in \mathbb{R}^n} h^*(u) - (g + \iota_{\mathcal{F}})^*(u), \quad (\mathcal{D})$$

which is also a DC program with the same optimal value. Indeed,

$$\begin{aligned} \inf_{u \in \mathbb{R}^n} \{h^*(u) - (g + \iota_{\mathcal{F}})^*(u)\} &= \inf_{u \in \mathbb{R}^n} \{h^*(u) - \sup_{x \in \mathbb{R}^n} \{\langle u, x \rangle - g(x) - \iota_{\mathcal{F}}(x)\}\} \\ &= \inf_{x \in \mathbb{R}^n} \{g(x) + \iota_{\mathcal{F}}(x) - \sup_{u \in \mathbb{R}^n} \{\langle u, x \rangle - h^*(u)\}\} \\ &= \inf_{x \in \mathbb{R}^n} \{g(x) + \iota_{\mathcal{F}}(x) - h(x)\} \\ &= \inf_{x \in \mathcal{F}} \{g(x) - h(x)\} = \inf_{x \in \mathcal{F}} \phi(x). \end{aligned}$$

We say that $\bar{x} \in \mathbb{R}^n$ (resp. $\bar{u} \in \mathbb{R}^n$) is a *critical point* of (\mathcal{P}) (resp. (\mathcal{D})) if $\nabla g(\bar{x}) \in \partial(h + \iota_{\mathcal{F}})(\bar{x})$ (resp. $\partial h^*(\bar{u}) \cap \partial(g + \iota_{\mathcal{F}})^*(\bar{u}) \neq \emptyset$). Recall from [21, Theorem 5.19] that \bar{x} is called a *KKT point* of (\mathcal{P}) if there exist $\mu_1, \mu_2, \dots, \mu_p \in \mathbb{R}$ such that

$$\begin{cases} 0 \in \nabla g(\bar{x}) - \partial h(\bar{x}) + \sum_{i=1}^p \mu_i a_i, \\ 0 = \mu_i (\langle a_i, \bar{x} \rangle - b_i), \quad i = 1, \dots, p, \\ \mu_i \geq 0, \quad \langle a_i, \bar{x} \rangle \leq b_i, \quad i = 1, \dots, p. \end{cases} \quad (2)$$

The DC algorithm is a primal-dual method in the sense that it is aimed to find solutions for both (\mathcal{P}) and (\mathcal{D}) , simultaneously. Our goal then is to design a BDCA variant that allow us to find KKT points of (\mathcal{P}) and critical points of (\mathcal{D}) .

To finish this section, we need to introduce the following notions regarding the geometry of the feasible set \mathcal{F} . The *cone of feasible directions* at $\bar{x} \in \mathcal{F}$ is denoted by

$$D(\bar{x}) := \{d \in \mathbb{R}^n \mid \exists \varepsilon > 0 \text{ such that } \bar{x} + td \in \mathcal{F}, \forall t \in [0, \varepsilon]\}.$$

Due to the polyhedral structure of \mathcal{F} , its normal cone coincides with the *active cone*; i.e.,

$$N_{\mathcal{F}}(\bar{x}) = A(\bar{x}) := \text{cone} \{a_i, \quad i \in I(\bar{x})\}, \quad \text{for } \bar{x} \in \mathcal{F},$$

where $I(\bar{x})$ stands for the set of *active constraints* at \bar{x} , i.e.,

$$I(\bar{x}) = \{i \in \{1, \dots, p\} \mid \langle a_i, \bar{x} \rangle = b_i\}.$$

Since we deal with affine constraints, we have (see e.g. [2, Proposition 4.14])

$$D(\bar{x}) = \{d \in \mathbb{R}^n \mid \langle a_i, d \rangle \leq 0, \quad i \in I(\bar{x})\}. \quad (3)$$

3 The Boosted DC Algorithm and its convergence

For solving (\mathcal{P}), we propose the following method, Algorithm 1, which can tackle more general problems than the Boosted DC Algorithm proposed in [3].

Algorithm 1 BDCA (Boosted DC Algorithm) for solving (\mathcal{P})

Input: An initial point $x_0 \in \mathcal{F}$ and two parameters $\alpha > 0$ and $\beta \in]0, 1[$;

- 1: $k \leftarrow 0$;
- 2: Select $u_k \in \partial h(x_k)$ and compute the unique solution y_k of

$$\begin{cases} \min_{x \in \mathbb{R}^n} & \phi_k(x) := g(x) - \langle u_k, x \rangle \\ \text{s.t.} & \langle a_i, x \rangle \leq b_i, \quad i = 1, \dots, p. \end{cases} \quad (\mathcal{P}_k)$$

- 3: $d_k \leftarrow y_k - x_k$;
 - 4: **if** $d_k = 0$ **then**
 - 5: **stop** and **return** x_k ;
 - 6: **end if**
 - 7: **if** $I(y_k) \subseteq I(x_k)$ **then**
 - 8: Choose any $\bar{\lambda}_k \geq 0$, set $\lambda_k \leftarrow \bar{\lambda}_k$, and reduce λ_k until $y_k + \lambda_k d_k \in \mathcal{F}$;
 - 9: **while** $\phi(y_k + \lambda_k d_k) > \phi(y_k) - \alpha \lambda_k^2 \|d_k\|^2$ **do**
 - 10: $\lambda_k \leftarrow \beta \lambda_k$;
 - 11: **end while**
 - 12: **else**
 - 13: $\lambda_k \leftarrow 0$;
 - 14: **end if**
 - 15: $x_{k+1} \leftarrow y_k + \lambda_k d_k$;
 - 16: $k \leftarrow k + 1$ and **go to** Line 2;
-

Let us make some comments on Algorithm 1.

- (i) Lines 1 to 6 of Algorithm 1 correspond to the classical DCA for solving (\mathcal{P}). Thus, when $\bar{\lambda}_k = 0$ for all k , Algorithm 1 coincides with DCA.
- (ii) Lines 7 to 15 present the boosting step. It first checks if d_k is a feasible direction at $y_k \in \mathcal{F}$. If so, it then performs a line search step along the direction d_k which maintains feasibility to improve the objective value ϕ . Otherwise, the boosting step is skipped and we simply use the DCA point y_k .
- (iii) In terms of per-iteration complexity, the boosting step requires to check the feasibility of direction d_k , which can be done by comparing the sets of active constraints at x_k and y_k (see Lemma 3.1). It also requires evaluating the objective function and checking the feasibility of the trial step $y_k + \lambda_k d_k$. The computational effort of this task will depend on the particular structure of ϕ and \mathcal{F} . In the case of box constraints, the largest step-size that makes $y_k + \lambda_k d_k$ feasible can be efficiently computed in Line 8,

see Remark 3.1 below. In Section 5 we show some computational results for the trust-region subproblems with ℓ_1 norm constraints (where feasibility needs to be checked) and ℓ_∞ norm constraints (where the largest step-size can be readily computed).

- (iv) When h is differentiable, the algorithms introduced by Fukushima and Mine in [10, 20] can be applied in our setting. On the one hand, the algorithm in [10] performs a line search which is similar to the one in Lines 9-11 of Algorithm 1, but with the significant difference that it is performed at the point x_k , instead of doing it at the DCA point y_k ; that is, it searches for the smallest non-negative integer l such that

$$\phi(x_k + \beta^l d_k) \leq \phi(x_k) - \alpha \beta^l \|d_k\|^2,$$

where $0 < \beta < 1$. Thus, the largest step-size allowed by their algorithm is 1, which corresponds with the point determined by the DCA, since $x_k + d_k = y_k$. On the other hand, the algorithm defined in [20] performs an exact line search in the direction d_k , which may be unaffordable in many practical applications.

Remark 3.1. An explicit formula to compute the largest step-size that makes $y_k + \lambda_k d_k$ feasible in Line 8 is provided in [9]. More precisely, feasibility of the trial step size is guaranteed if $\bar{\lambda}_k$ is chosen so that

$$\bar{\lambda}_k \leq \hat{\lambda}_k := \min \left\{ \frac{b_i - \langle a_i, y_k \rangle}{|\langle a_i, d_k \rangle|} : i \notin I(x_k) \text{ with } \langle a_i, d_k \rangle \neq 0 \right\}; \quad (4)$$

see [9, Lemma 5.4(ii)]. In principle, this permits to avoid the extra time consumed for checking feasibility in Line 8. However, the computation of $\hat{\lambda}_k$ in (4) may be even more expensive and inefficient than checking feasibility of λ_k in some applications. This is the case, for instance, of the ℓ_1 -norm trust region subproblem, whose reformulation as a linearly constrained DC program requires an exponential number of constraints (see Section 5.2).

The next auxiliary lemma shows the equivalence between Line 7 of Algorithm 1 and checking the feasibility of the direction generated by DCA.

Lemma 3.1. If x_k and y_k are generated by Algorithm 1, then

$$I(y_k) \subseteq I(x_k) \quad \Leftrightarrow \quad d_k := y_k - x_k \in D(y_k) \quad \Leftrightarrow \quad d_k \perp a_i, \forall i \in I(y_k).$$

Proof. Observe that, for any $i \in I(y_k)$, it holds that

$$\langle a_i, d_k \rangle = \langle a_i, y_k \rangle - \langle a_i, x_k \rangle = b_i - \langle a_i, x_k \rangle \geq 0.$$

Hence, the result easily follows by taking into account (3). \square

In the following proposition, we collect some key inequalities which are useful in the sequel for the convergence analysis of Algorithm 1.

Proposition 3.1. Under Assumptions 1 and 2, for all $k \in \mathbb{N}$, the next statements hold:

- (i) $\phi(y_k) \leq \phi(x_k) - \rho \|d_k\|^2$;
- (ii) $\phi'(y_k; d_k) \leq -\rho \|d_k\|^2$;

(iii) if the condition at Line 7 of Algorithm 1 holds, then there exists some $\delta_k > 0$ such that $y_k + \lambda_k d_k \in \mathcal{F}$ and

$$\phi(y_k + \lambda d_k) \leq \phi(y_k) - \alpha \lambda^2 \|d_k\|^2, \quad \text{for all } \lambda \in [0, \delta_k].$$

Consequently, the backtracking step at Lines 9–11 of Algorithm 1 terminates after a finite number of iterations.

Proof. The proof of (i) is similar to the one of [1, Proposition 3] and is therefore omitted. To prove (ii), pick any $v \in \partial h(y_k)$. Note that the one-sided directional derivative $\phi'(y_k; d_k)$ is given by

$$\begin{aligned} \phi'(y_k; d_k) &= \lim_{t \downarrow 0} \frac{\phi(y_k + t d_k) - \phi(y_k)}{t} \\ &= \lim_{t \downarrow 0} \frac{g(y_k + t d_k) - g(y_k)}{t} - \lim_{t \downarrow 0} \frac{h(y_k + t d_k) - h(y_k)}{t} \\ &\leq \langle \nabla g(y_k), d_k \rangle - \langle v, d_k \rangle, \end{aligned} \quad (5)$$

by convexity of h . Since y_k is the unique solution of the strongly convex problem (\mathcal{P}_k) , we can write down the KKT conditions (see, e.g., [2, Theorem 4.20]) of this problem as

$$\begin{cases} \nabla g(y_k) + \sum_{i=1}^p \mu_{k,i} a_i = u_k \in \partial h(x_k), \\ \mu_{k,i} (\langle a_i, y_k \rangle - b_i) = 0, \quad \mu_{k,i} \geq 0, \quad \langle a_i, y_k \rangle \leq b_i, \quad i = 1, \dots, p. \end{cases} \quad (6)$$

The fact that h is strongly convex with a parameter ρ implies, by [35, Exercise 12.59], that ∂h is strongly monotone with constant ρ . Therefore, since $v \in \partial h(y_k)$ and $u_k \in \partial h(x_k)$, we have

$$\langle u_k - v, x_k - y_k \rangle \geq \rho \|x_k - y_k\|^2.$$

Hence, combining these expressions, together with the fact that $x_k \in \mathcal{F}$, we can derive

$$\begin{aligned} \langle \nabla g(y_k) - v, d_k \rangle &= \left\langle u_k - \sum_{i=1}^p \mu_{k,i} a_i - v, y_k - x_k \right\rangle \\ &\leq -\rho \|d_k\|^2 - \sum_{i=1}^p \mu_{k,i} \langle a_i, y_k - x_k \rangle \\ &= -\rho \|d_k\|^2 + \sum_{i=1}^p \mu_{k,i} (\langle a_i, x_k \rangle - b_i) + \sum_{i=1}^p \mu_{k,i} (b_i - \langle a_i, y_k \rangle) \\ &\leq -\rho \|d_k\|^2, \end{aligned}$$

and the result follows by combining the last inequality with (5).

Having in mind the condition at Line 7 of Algorithm 1 and Lemma 3.1, we observe that the proof of (iii) is similar to the one of [3, Proposition 3.1], so we omit it for brevity. \square

Remark 3.2 (General convex constraints). Consider a generalized version of (\mathcal{P}) where the feasible set is formed by arbitrary convex constraints, i.e.,

$$\begin{cases} \min_{x \in \mathbb{R}^n} & \phi(x) := g(x) - h(x) \\ \text{s.t.} & c_i(x) \leq 0, \quad i = 1, \dots, p, \end{cases} \quad (\mathcal{P}')$$

where g and h satisfy Assumptions 1 and 2 and $c_i : \mathbb{R}^n \rightarrow \mathbb{R}$ are smooth, proper, closed and convex functions, for $i = 1, \dots, p$. Note that problem (\mathcal{P}) is a particular instance of (\mathcal{P}') with $c_i(x) := \langle a_i, x \rangle - b_i$, for $i = 1, \dots, p$. The assertion in Proposition 3.1(ii) still holds true for the more general problem (\mathcal{P}') ; that is, the direction generated by DCA remains a descent direction provided that x_k is feasible for (\mathcal{P}') . To confirm this, one can easily check that the proof can be rewritten by replacing the linearity of the gradients by the inequality

$$c_i(x_k) \geq c_i(y_k) - \langle \nabla c_i(y_k), y_k - x_k \rangle. \quad (7)$$

However, Line 7 of Algorithm 1 is no longer useful to verify if d_k is a feasible direction, as the equality in (3) only holds for affine constraints. For general convex constraints, we have the inclusion

$$\{d \in \mathbb{R}^n \mid \langle \nabla c_i(\bar{x}), d \rangle < 0, i \in I(\bar{x})\} \subset D(\bar{x}).$$

Therefore, one possibility would be to run the boosting step whenever $\langle \nabla c_i(y_k), d_k \rangle < 0$ for all $i \in I(y_k)$. Nevertheless, this will never be the case because x_k is feasible for (\mathcal{P}') . Indeed, from (7), we obtain that

$$\langle \nabla c_i(y_k), d_k \rangle \geq -c_i(x_k) \geq 0, \quad \text{for all } i \in I(y_k).$$

In fact, it can be proved that if $y_k + \lambda d_k \in \mathcal{F}$ for some particular $\lambda > 0$, then the points in the segment $[x_k, y_k + \lambda d_k]$ must be active for all $i \in I(y_k)$.

We are now in the position to establish the main convergence result of Algorithm 1.

Theorem 3.1. *Under Assumptions 1, 2 and 3, for any $x_0 \in \mathcal{F}$, either BDCA returns a KKT point of (\mathcal{P}) or it generates an infinite sequence such that the following statements hold.*

- (i) $\phi(x_k)$ is monotonically decreasing and hence convergent to some $\bar{\phi}$.
- (ii) Suppose that $\{x_k\}$ and $\{u_k\}$ are bounded. Then any limit point of $\{x_k\}$ is a KKT point of (\mathcal{P}) and any limit point of $\{u_k\}$ is a critical point of (\mathcal{D}) .
- (iii) We have $\sum_{k=0}^{+\infty} \|d_k\|^2 < +\infty$. Moreover, if there is some $\bar{\lambda}$ such that $\lambda_k \leq \bar{\lambda}$ for all k , then $\sum_{k=0}^{+\infty} \|x_{k+1} - x_k\|^2 < +\infty$.

Proof. If Algorithm 1 is terminated at Line 5 and returns x_k , then $x_k = y_k$. From (2) and (6), it is clear that x_k is a KKT point of (\mathcal{P}) . Otherwise, by Proposition 3.1 and Line 15 of Algorithm 1, we have

$$\phi(x_{k+1}) \leq \phi(y_k) - \alpha \lambda_k^2 \|d_k\|^2 \leq \phi(x_k) - (\alpha \lambda_k^2 + \rho) \|d_k\|^2, \quad (8)$$

where $\lambda_k \geq 0$. Therefore, the sequence $\{\phi(x_k)\}$ converges to some $\bar{\phi}$, since it is monotonically decreasing and bounded from below, by (1). As a consequence, we obtain

$$\phi(x_{k+1}) - \phi(x_k) \rightarrow 0, \quad \text{as } k \rightarrow \infty,$$

which implies $\|d_k\|^2 = \|y_k - x_k\|^2 \rightarrow 0$, by (8).

Now, if \bar{x} is a limit point of $\{x_k\}$, then there exists a subsequence $\{x_{k_j}\}$ converging to \bar{x} . Then, as $\|y_{k_j} - x_{k_j}\| \rightarrow 0$, we have $y_{k_j} \rightarrow \bar{x}$. From (6), we obtain

$$\begin{cases} \nabla g(y_{k_j}) + \sum_{i=1}^p \mu_{k_j, i} a_i = u_{k_j} \in \partial h(x_{k_j}), \\ \mu_{k_j, i} (\langle a_i, y_{k_j} \rangle - b_i) = 0, \quad \mu_{k_j, i} \geq 0, \quad \langle a_i, y_{k_j} \rangle \leq b_i, \quad i = 1, \dots, p. \end{cases} \quad (9)$$

Since the sequence $\{u_k\}$ is bounded by assumption, without loss of generality we may assume that $u_{k_j} \rightarrow \bar{u}$. It also holds that $\nabla g(y_{k_j}) \rightarrow \nabla g(\bar{x})$ by continuity of ∇g . The sequence of Lagrange multipliers $\{\mu_{k_j}\} \in \mathbb{R}^p$ must also be bounded. Indeed, suppose to the contrary that $\|\mu_{k_j}\| \rightarrow \infty$ and assume without loss of generality that $\lim_{j \rightarrow \infty} \frac{\mu_{k_j}}{\|\mu_{k_j}\|} = \mu^*$, with $\mu^* \in \mathbb{R}_+^p$ (the non-negative orthant) satisfying $\|\mu^*\| = 1$. Then, dividing the first equality in (9) by $\|\mu_{k_j}\|$ and letting $j \rightarrow \infty$, we obtain

$$\sum_{i=1}^p \mu_i^* a_i = 0_n.$$

Performing the same procedure in the second equality in (9) gives

$$\mu_i^* (\langle a_i, \bar{x} \rangle - b_i) = 0,$$

so we deduce that $\mu_i^* = 0$ for all $i \notin I(\bar{x})$. Thus,

$$\sum_{i \in I(\bar{x})} \mu_i^* a_i = 0_n.$$

Thanks to the Slater Assumption 3, we know that there exists $\hat{x} \in \mathbb{R}^n$ such that $\langle a_i, \hat{x} \rangle < b_i$ for all $i \in I(\bar{x})$. Hence,

$$0 = \sum_{i \in I(\bar{x})} \mu_i^* \langle a_i, \hat{x} - \bar{x} \rangle = \sum_{i \in I(\bar{x})} \mu_i^* (\langle a_i, \hat{x} \rangle - \langle a_i, \bar{x} \rangle) = \sum_{i \in I(\bar{x})} \mu_i^* (\langle a_i, \hat{x} \rangle - b_i),$$

which implies $\mu_i^* = 0$ for all $i \in I(\bar{x})$, since $\mu^* \in \mathbb{R}_+^p$. This implies that $\mu^* = 0_p$, a contradiction with the fact that $\|\mu^*\| = 1$.

Therefore, by extracting subsequences if necessary, we can assume that

$$\lim_{j \rightarrow \infty} \mu_{k_j, i} = \mu_i \geq 0, \quad \text{for all } i = 1, 2, \dots, p. \quad (10)$$

Taking the limit as $j \rightarrow \infty$ in (9), thanks to the closedness of the graph of ∂h (see [34, Theorem 24.4]), we obtain

$$\begin{cases} \nabla g(\bar{x}) + \sum_{i=1}^p \mu_i a_i = \bar{u} \in \partial h(\bar{x}), \\ \mu_i (\langle a_i, \bar{x} \rangle - b_i) = 0, \quad \mu_i \geq 0, \quad \langle a_i, \bar{x} \rangle \leq b_i, \quad i = 1, \dots, p, \end{cases} \quad (11)$$

which means that \bar{x} is a KKT point of (\mathcal{P}) . From (11) we derive that $\bar{x} \in \partial h^*(\bar{u})$ and also that

$$\bar{u} = \nabla g(\bar{x}) + \sum_{i \in I(\bar{x})} \mu_i a_i \in \nabla g(\bar{x}) + A(\bar{x}) = \partial(g + \iota_{\mathcal{F}})(\bar{x}),$$

which is equivalent to $\bar{x} \in \partial(g + \iota_{\mathcal{F}})^*(\bar{u})$ and, hence, \bar{u} is a critical point of (\mathcal{D}) . The proof of (iii) is similar to that of [1, Proposition 5(iii)] and is thus omitted. \square

Remark 3.3. The assertion in Theorem 3.1(ii) remains valid for any limit point of $\{x_k\}$ in the interior of $\text{dom } h$ without requiring the dual sequence $\{u_k\}$ to be bounded. Indeed, let $\bar{x} \in \text{int dom}(h)$ be a cluster point of $\{x_k\}$ and let $\{x_{k_j}\}$ be a subsequence converging to \bar{x} . We

can apply [34, Corollary 24.5.1] to obtain that for any $\varepsilon > 0$, there exists a positive integer $j_0 \geq 1$ such that

$$u_{k_j} \in \partial h(x_{k_j}) \subset \partial h(\bar{x}) + \varepsilon \mathbb{B}, \quad \forall j \geq j_0;$$

where \mathbb{B} denotes the Euclidean unit ball of \mathbb{R}^n . Hence, since $\partial h(\bar{x})$ is a bounded set, the dual subsequence $\{u_{k_j}\}$ is also bounded and the proof of Theorem 3.1(ii) stands.

Remark 3.4. Similar to [1, Theorem 1] and [3, Theorem 4.3 and Theorem 4.9], if we further assume that the function ϕ satisfies the Kurdyka–Łojasiewicz property, then it can be proved that the sequence $\{x_k\}$ converges to a KKT point of (\mathcal{P}) . Moreover, convergence rates can also be deduced depending on the Łojasiewicz exponent. Especially, when the objective function ϕ is quadratic (e.g., in some of our numerical experiments), it was proved [18, Theorem 4.2] that the function $\phi + \iota_{\mathcal{F}}$ satisfies the Kurdyka–Łojasiewicz property with exponent $\frac{1}{2}$. Combining this with the technique in [1, Theorem 1], it is a routine task to derive the linear convergence of the sequence $\{x_k\}$ when the sequence has a cluster point. The purpose of the next section is to prove, using a similar technique to [32, Theorem 2.1], that the latter condition is not needed: for quadratic functions, BDCA always generates a sequence which is linearly convergent to a KKT point of the problem without requiring the existence of a cluster point.

4 Linear convergence for quadratic objective functions

In this section we prove the R-linear convergence of BDCA when the objective function ϕ of (\mathcal{P}) is quadratic, that is, for problems of the form

$$\begin{cases} \min_{x \in \mathbb{R}^n} & \phi(x) := \frac{1}{2} \langle Qx, x \rangle + \langle q, x \rangle \\ \text{s.t.} & \langle a_i, x \rangle \leq b_i, \quad i = 1, \dots, p, \end{cases} \quad (\mathcal{P}_Q)$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix, $q \in \mathbb{R}^n$, and $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$, for $i = 1, \dots, p$. In this setting, \bar{x} is a KKT point of (\mathcal{P}_Q) if there exist multipliers $\mu_1, \mu_2, \dots, \mu_p \in \mathbb{R}$ such that

$$\begin{cases} 0 = Q\bar{x} + q + \sum_{i=1}^p \mu_i a_i, \\ 0 = \mu_i (\langle a_i, \bar{x} \rangle - b_i), \quad i = 1, \dots, p, \\ \mu_i \geq 0, \quad \langle a_i, \bar{x} \rangle \leq b_i, \quad i = 1, \dots, p. \end{cases}$$

We denote by $\overline{\mathcal{F}}$ the set of all KKT points of (\mathcal{P}) .

As Q is not required to be positive semidefinite, (\mathcal{P}_Q) is a nonconvex problem. However, the matrix Q can be easily decomposed as $Q = Q_1 - Q_2$, with Q_1 and Q_2 positive definite. Indeed, one can take

$$Q_1 := \sigma I \quad \text{and} \quad Q_2 := \sigma I - Q,$$

for $\sigma > \max\{0, \lambda_{\max}(Q)\}$, where $\lambda_{\max}(Q)$ is the largest eigenvalue of Q and I denotes the identity matrix. Thus, (\mathcal{P}_Q) can be equivalently written in the form of (\mathcal{P}) as

$$\begin{cases} \min_{x \in \mathbb{R}^n} & g(x) - h(x) = \phi(x) \\ \text{s.t.} & \langle a_i, x \rangle \leq b_i, \quad i = 1, \dots, p, \end{cases} \quad (12)$$

with

$$g(x) := \frac{\sigma}{2} \|x\|^2 + \langle q, x \rangle \quad \text{and} \quad h(x) := \frac{1}{2} \langle (\sigma I - Q)x, x \rangle. \quad (13)$$

Observe that both functions g and h are strongly convex with parameters σ and $\sigma - \lambda_{\max}(Q)$, respectively. Thus, Assumption 1 holds for

$$\rho := \min\{\sigma, \sigma - \lambda_{\max}(Q)\}, \quad (14)$$

while Assumption 2 trivially holds.

By using the indicator function $\iota_{\mathcal{F}}$ of the feasible set \mathcal{F} , we can rewrite problem (\mathcal{P}_Q) as an unconstrained nonsmooth DC optimization problem, which can be tackled by the DCA. In this case, the DCA becomes the projected gradient method from convex programming [30]

$$x_{k+1} = P_{\mathcal{F}} \left(x_k - \frac{1}{\sigma} (Qx_k + q) \right), \quad (15)$$

where $P_{\mathcal{F}}$ denotes the projection mapping onto the feasible set \mathcal{F} .

To derive the R-linear convergence of the iterative sequence generated by the BDCA, we will use the following lemmas. The first one is a classical result regarding the connected components of the KKT set $\overline{\mathcal{F}}$ and can be found in [19, Lemma 3.1].

Lemma 4.1. Let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_r$ be the connected components of the KKT set $\overline{\mathcal{F}}$. Then we have

$$\overline{\mathcal{F}} = \bigcup_{i=1}^r \mathcal{F}_i,$$

and the following properties hold:

- (i) each \mathcal{F}_i is the union of finitely many polyhedral convex sets;
- (ii) the sets $\mathcal{F}_i, i = 1, 2, \dots, r$, are properly separated from each others, that is

$$\inf \{d(x, \mathcal{F}_i), x \in \mathcal{F}_j\} > 0, \quad \text{for all } i \neq j;$$

- (iii) ϕ is constant on each \mathcal{F}_i .

The second one is a local error bound result originally stated in [19, Theorem 2.3] and later extended in [32, Lemma 2.1].

Lemma 4.2. There exist scalars $\varepsilon > 0$ and $\tau > 0$ such that

$$d(x, \overline{\mathcal{F}}) \leq \tau \left\| x - P_{\mathcal{F}} \left(x - \frac{1}{\sigma} (Qx + q) \right) \right\|, \quad (16)$$

for all $x \in \mathcal{F}$ with

$$\left\| x - P_{\mathcal{F}} \left(x - \frac{1}{\sigma} (Qx + q) \right) \right\| \leq \varepsilon. \quad (17)$$

We are now in a position to establish the R-linear (aka geometric) convergence of the iterative sequence $\{x_k\}$ generated by BDCA. Geometric convergence is a special type of R-linear convergence, which is implied by Q-linear convergence (see, e.g. [2, Section 5.2.1]). The next result extends [32, Theorem 2.1] to the BDCA. Its proof employs similar techniques, which are originally based on [19].

Theorem 4.1. *If (\mathcal{P}_Q) has a solution and Assumption 3 holds, then the sequence $\{x_k\}$ generated by BDCA converges geometrically to a KKT point \bar{x} of (\mathcal{P}_Q) , that is, there exist some constants $C > 0$ and $\eta \in]0, 1[$ such that*

$$\|x_k - \bar{x}\| \leq C\eta^k, \quad \text{for all large } k. \quad (18)$$

Proof. First, observe that by (8) we have

$$(\rho + \alpha\lambda_k^2)\|d_k\|^2 \leq \phi(x_k) - \phi(x_{k+1}).$$

By Theorem 3.1(i), we know that the right-hand side of this inequality converges to zero as $k \rightarrow \infty$. Thus,

$$\lim_{k \rightarrow \infty} \|y_k - x_k\| = \lim_{k \rightarrow \infty} \|d_k\| = 0 = \lim_{k \rightarrow \infty} \lambda_k \|d_k\|,$$

which implies

$$\lim_{k \rightarrow \infty} \|x_{k+1} - x_k\| = \lim_{k \rightarrow \infty} (1 + \lambda_k) \|d_k\| = 0. \quad (19)$$

Let $\varepsilon > 0$ and $\tau > 0$ be such that (16) and (17) hold. Since $y_k = P_{\mathcal{F}}\left(x_k - \frac{1}{\sigma}(Qx_k + q)\right)$, there exists some $k_0 \in \mathbb{N}$ such that

$$\left\| x_k - P_{\mathcal{F}}\left(x_k - \frac{1}{\sigma}(Qx_k + q)\right) \right\| = \|y_k - x_k\| \leq \varepsilon, \quad \forall k \geq k_0.$$

Hence, we obtain

$$d(x_k, \overline{\mathcal{F}}) \leq \tau \left\| x_k - P_{\mathcal{F}}\left(x_k - \frac{1}{\sigma}(Qx_k + q)\right) \right\| = \tau \|x_k - y_k\| = \tau \|d_k\|, \quad \forall k \geq k_0.$$

Due to the fact that $\overline{\mathcal{F}}$ is nonempty and closed, there exists $z_k \in \overline{\mathcal{F}}$, for each $k \in \mathbb{N}$, such that $d(x_k, \overline{\mathcal{F}}) = \|x_k - z_k\|$. Thus

$$\|x_k - z_k\| \leq \tau \|d_k\|, \quad \forall k \geq k_0, \quad (20)$$

which implies

$$\lim_{k \rightarrow \infty} \|x_k - z_k\| = 0. \quad (21)$$

Since

$$\|z_{k+1} - z_k\| \leq \|z_{k+1} - x_{k+1}\| + \|x_{k+1} - x_k\| + \|x_k - z_k\|,$$

it follows from (19) and (21) that

$$\lim_{k \rightarrow \infty} \|z_{k+1} - z_k\| = 0. \quad (22)$$

Now, let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_r$ be the connected components of $\overline{\mathcal{F}}$. By Lemma 4.1(ii) and (22) there exists $\mathcal{F}_0 \in \{\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_r\}$ and $k_1 \geq k_0$ such that $z_k \in \mathcal{F}_0$ for all $k \geq k_1$. The last assertion of Lemma 4.1 implies that

$$\phi(z_k) = c, \quad \forall k \geq k_1. \quad (23)$$

Note that, since z_k is a KKT point of (\mathcal{P}_Q) , we have $\langle Qz_k + q, x_k - z_k \rangle \geq 0$. Then,

$$\begin{aligned}\phi(z_k) - \phi(x_k) &= \frac{1}{2} \langle Qz_k, z_k \rangle - \frac{1}{2} \langle Qx_k, x_k \rangle + \langle q, z_k - x_k \rangle \\ &\leq \frac{1}{2} \langle Qz_k, z_k \rangle - \frac{1}{2} \langle Qx_k, x_k \rangle + \langle Qz_k, x_k - z_k \rangle \\ &= \frac{1}{2} \langle Q(x_k - z_k), z_k - x_k \rangle \\ &\leq \frac{1}{2} \|Q\| \|x_k - z_k\|^2.\end{aligned}$$

From Theorem 3.1(i) we know that $\lim_{k \rightarrow \infty} \phi(x_k) = \bar{\phi}$. Hence, for all $k \geq k_1$,

$$c = \phi(z_k) \leq \phi(x_k) + \frac{1}{2} \|Q\| \|z_k - x_k\|^2 \rightarrow \bar{\phi}, \text{ as } k \rightarrow \infty. \quad (24)$$

We prove now that $\bar{\phi} \leq c$. Indeed, on the one hand, from (8) and (23), we have for all $k \geq k_1$ that

$$\begin{aligned}\phi(x_{k+1}) - c &\leq \phi(y_k) - c = \phi(y_k) - \phi(z_k) \\ &= \frac{1}{2} \langle Qy_k, y_k \rangle + \langle q, y_k \rangle - \frac{1}{2} \langle Qz_k, z_k \rangle - \langle q, z_k \rangle \\ &= \langle Qz_k + q, y_k - z_k \rangle + \frac{1}{2} \langle Q(y_k - z_k), y_k - z_k \rangle \\ &\leq \langle Qz_k + q, y_k - z_k \rangle + \frac{1}{2} \|Q\| \|y_k - z_k\|^2.\end{aligned} \quad (25)$$

On the other hand, since $y_k = P_{\mathcal{F}}(x_k - \frac{1}{\sigma}(Qx_k + q))$ and $z_k \in \mathcal{F}$, we deduce (see, e.g., [4, Theorem 3.16]) that

$$\left\langle x_k - \frac{1}{\sigma}(Qx_k + q) - y_k, y_k - z_k \right\rangle \geq 0.$$

Therefore,

$$\begin{aligned}\langle Qz_k + q, y_k - z_k \rangle &= \langle Qx_k + q, y_k - z_k \rangle + \langle Q(z_k - x_k), y_k - z_k \rangle \\ &\leq \sigma \langle x_k - y_k, y_k - z_k \rangle + \|Q\| \|x_k - z_k\| \|y_k - z_k\| \\ &\leq (\sigma \|x_k - y_k\| + \|Q\| \|x_k - z_k\|) \|y_k - z_k\|.\end{aligned} \quad (26)$$

Combining (25) and (26), we obtain for all $k \geq k_1$ that

$$\phi(x_{k+1}) - c \leq \left(\sigma \|x_k - y_k\| + \|Q\| \|z_k - x_k\| + \frac{1}{2} \|Q\| \|y_k - z_k\| \right) \|y_k - z_k\|. \quad (27)$$

From (20), we have

$$\|y_k - z_k\| \leq \|y_k - x_k\| + \|x_k - z_k\| \leq (1 + \tau) \|d_k\|, \quad \forall k \geq k_1.$$

Hence, we deduce from (27) that

$$\phi(x_{k+1}) - c \leq \beta \|d_k\|^2, \quad \forall k \geq k_1, \quad (28)$$

where $\beta := (1 + \tau) (\sigma + \|Q\|\tau + \frac{1+\tau}{2}\|Q\|)$. Passing (28) to the limit as $k \rightarrow \infty$, we get

$$\bar{\phi} = \lim_{k \rightarrow \infty} \phi(x_{k+1}) \leq c.$$

The latter together with (24) imply that $\bar{\phi} = c$. Therefore, it follows from (28) and (8) that

$$\phi(x_{k+1}) - \bar{\phi} \leq \beta \|d_k\|^2 \leq \frac{\beta}{\rho} (\phi(x_k) - \phi(x_{k+1})), \quad \forall k \geq k_1,$$

or, equivalently,

$$\left(1 + \frac{\beta}{\rho}\right) (\phi(x_{k+1}) - \bar{\phi}) \leq \frac{\beta}{\rho} (\phi(x_k) - \bar{\phi}), \quad \forall k \geq k_1.$$

Since $\{\phi(x_k)\}$ is monotonically decreasing to $\bar{\phi}$, we deduce from the last inequality that

$$\begin{aligned} \phi(x_k) - \phi(x_{k+1}) &= (\phi(x_k) - \bar{\phi}) + (\bar{\phi} - \phi(x_{k+1})) \\ &\leq \phi(x_k) - \bar{\phi} \leq (\phi(x_{k_1}) - \bar{\phi}) \left(\frac{\beta}{\rho + \beta}\right)^{k-k_1} \\ &= M_0 \eta^{2k}, \quad \forall k \geq k_1, \end{aligned} \tag{29}$$

where $M_0 := (\phi(x_{k_1}) - \bar{\phi}) (\rho + \beta)^{k_1} \beta^{-k_1}$ and $\eta := \sqrt{\frac{\beta}{\rho + \beta}} \in]0, 1[$. Consider now

$$M_1 := \max_{\lambda \geq 0} \frac{(1 + \lambda)^2}{\alpha \lambda^2 + \rho} = \frac{\alpha + \rho}{\alpha \rho}.$$

Hence, from (8) and (29), we obtain

$$\begin{aligned} \|x_{k+1} - x_k\|^2 &= (1 + \lambda_k)^2 \|d_k\|^2 \\ &\leq \frac{(1 + \lambda_k)^2}{\alpha \lambda_k^2 + \rho} (\phi(x_k) - \phi(x_{k+1})) \\ &\leq M_1 M_0 \eta^{2k}, \quad \forall k \geq k_1, \end{aligned}$$

which implies

$$\|x_{k+1} - x_k\| \leq M \eta^k, \quad \forall k \geq k_1,$$

with $M := \sqrt{M_1 M_0}$. Then, for all $k \geq k_1$ and $m \geq 1$, we have

$$\begin{aligned} \|x_{k+m} - x_k\| &\leq \|x_{k+m} - x_{k+m-1}\| + \dots + \|x_{k+1} - x_k\| \\ &\leq (\eta^{m-1} + \dots + \eta + 1) M \eta^k \leq \frac{M}{1 - \eta} \eta^k. \end{aligned} \tag{30}$$

This implies that $\{x_k\}$ is a Cauchy sequence and hence converges to a point $\bar{x} \in \mathcal{F}$. According to Theorem 3.1(ii) and Remark 3.3, combined with the fact that $\text{dom } h = \mathbb{R}^n$ is an open set, we must have that \bar{x} is a KKT point of (\mathcal{P}_Q) . Moreover, passing the inequality (30) to the limit as $m \rightarrow \infty$, we obtain

$$\|\bar{x} - x_k\| \leq \frac{M}{1 - \eta} \eta^k, \quad \forall k \geq k_1,$$

which concludes the proof. \square

5 Numerical experiments

The purpose of this section is to compare the performance of BDCA (Algorithm 1) against the classical DCA. We refrain from comparing the algorithms against other methods, as our only aim here is to show that it is more advantageous to apply BDCA than DCA whenever this is possible, not to show that these methods are the most efficient for the problems that we consider. Consequently, we tested both algorithms in the three different settings that can occur, depending on whether the feasible set is unbounded, bounded with general constraints, and bounded with box constraints (so the feasibility in the line search step can be ensured). We applied both algorithms for testing copositivity [8] (unbounded) and for solving the ℓ_1 and ℓ_∞ trust-region subproblem [7, 12] (bounded, with box constraints in the case of ℓ_∞). In our last experiment we test the performance on piecewise quadratic problems with box constraints, whose objective function is thus nonsmooth. All these problems clearly satisfy Assumptions 1-3 and admit a DC decomposition with g being a quadratic function. Thus, subproblem (\mathcal{P}_k) can be solved by computing a projection onto the feasible set \mathcal{F} , analogous to that stated in equation (15). The projector onto the feasible set in our applications can be directly computed.

All the codes were written in Python 3.7 and the tests were run on an Intel Core i7-4770 CPU 3.40GHz with 32GB RAM, under Windows 10 (64-bit).

5.1 Testing copositivity

Recall that a given $n \times n$ matrix A is said to be *copositive* if

$$\langle Ax, x \rangle \geq 0, \quad \text{for all } x \in \mathbb{R}_+^n,$$

where \mathbb{R}_+^n stands for the non-negative orthant. Copositivity has recently attracted considerable attention in mathematical optimization, see e.g. [5, 6, 8, 25]. The problem of determining whether a given matrix is not copositive is known to be NP-complete [23] and it can be recast as the following non-convex optimization problem

$$\min_{x \in \mathbb{R}_+^n} \phi(x) := \langle Ax, x \rangle. \quad (31)$$

The copositivity of A is now equivalent to $\min_{x \in \mathbb{R}_+^n} \phi(x) = 0$. In [8], the authors reformulated (31) as a DC problem according to the decomposition in (12)-(13), and applied DCA as a heuristic for testing whether a matrix is not copositive. To be more specific, given $\sigma > \max\{\lambda_{\max}(A), 0\}$, the reformulation of problem (31) as a DC problem becomes

$$\begin{cases} \min_{x \in \mathbb{R}^n} & g(x) - h(x) = \phi(x) \\ \text{s.t.} & x_i \geq 0, \quad i = 1, \dots, n, \end{cases}$$

with

$$g(x) := \frac{\sigma}{2} \|x\|^2 \quad \text{and} \quad h(x) := \frac{1}{2} \langle (\sigma I - A)x, x \rangle.$$

Under this decomposition, DCA is applied as a heuristic to determine the copositivity of a given matrix as follows: if at some iterate $\phi(x_k) < 0$, then the matrix is non-copositive; otherwise, if a critical point is reached, the instance is undecidable.

Copositive matrices play an important role in graph theory. The size of the largest complete subgraph contained in a given graph G , denoted by $\gamma(G)$, is known as the *clique number* of G . If A and E are the adjacency matrix of G and the matrix of all ones, respectively, it can be shown (see [14, Corollary 2.4]) that

$$\gamma(G) = \min \{ \mu : \mu(E - A) - E \text{ is copositive} \}.$$

Therefore, the matrix $\mu(E - A) - E$ will be copositive if $\mu \geq \gamma(G)$ and non-copositive otherwise. Furthermore, in the latter case, the matrix will be closer to the copositive cone as μ approaches $\gamma(G)$ from the left.

In our tests we considered matrices constructed as follows. Let G be the cycle graph of n nodes whose adjacency matrix, $A_{\text{cycle}} = (a_{ij}) \in \mathbb{R}^{n \times n}$, is given component-wise by

$$a_{ij} := \begin{cases} 1, & \text{if } |i - j| \in \{1, n - 1\}, \\ 0, & \text{otherwise.} \end{cases}$$

Its clique number is clearly $\gamma(G) = 2$. Hence, the matrix

$$Q_n^\mu := \mu(E - A_{\text{cycle}}) - E \in \mathbb{R}^{n \times n} \quad (32)$$

is copositive for all $\mu \geq 2$ and non-copositive for $\mu < 2$. In fact, when $\mu = 2$ it coincides with the so-called Horn matrix H_n (see, e.g., [13, Section 4]). For instance, the Horn matrix H_5 takes the form

$$H_5 := Q_5^2 = \begin{pmatrix} 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{pmatrix}.$$

Experiments In our numerical tests we used the parameter setting as

$$\alpha := 0.01 \quad \text{and} \quad \beta := 0.1.$$

The trial step-size $\bar{\lambda}_k$ in the boosting step of BDCA (Line 8 of Algorithm 1) was chosen to be self-adaptive as in [3]. This technique proceeds as follows. At the first iteration, choose any $\bar{\lambda}_0 > 0$. Then, for $k \geq 1$, if the line search has never been used, we take $\bar{\lambda}_k = \bar{\lambda}_0$. Otherwise, if the two previous trial step-sizes have been directly accepted (without being reduced by the backtracking step), then the last accepted positive λ is scaled by a factor of $\gamma > 1$ and used as the current trial step-size. If that is not the case, the trial step-size is set as the last positive value of λ accepted in previous iterations. In our tests we used

$$\bar{\lambda}_0 := 1 \quad \text{and} \quad \gamma := 2.$$

In our first numerical experiment, we considered Horn matrices of different sizes, H_n , for $n \in \{1000, 1250, \dots, 5000\}$. For each size, DCA and BDCA were run from the same 100 starting points randomly generated in the intersection of the non-negative orthant with the unit ball. We stopped the algorithms when $\|d_k\| \leq 10^{-9}$ for the first time. The results are shown in Figure 1, where we can observe that, on average, BDCA was more than 15 times faster than DCA for all sizes. As expected, since Horn matrices are copositive, both

algorithms converged to critical points with a positive objective value very close to 0. It is worth to mention that the objective function at the points found by BDCA was usually smaller than at the ones found by DCA. We also show in Figure 2 the percentage of iterations at which the boosting step was activated, that is, when condition in line 7 of Algorithm 1 was fulfilled. We observed that, for all sizes, the linesearch was performed in around the 45% of the iterations. In Figure 3 we show the behavior of both algorithms in a particular instance for testing the copositivity of H_{1000} .

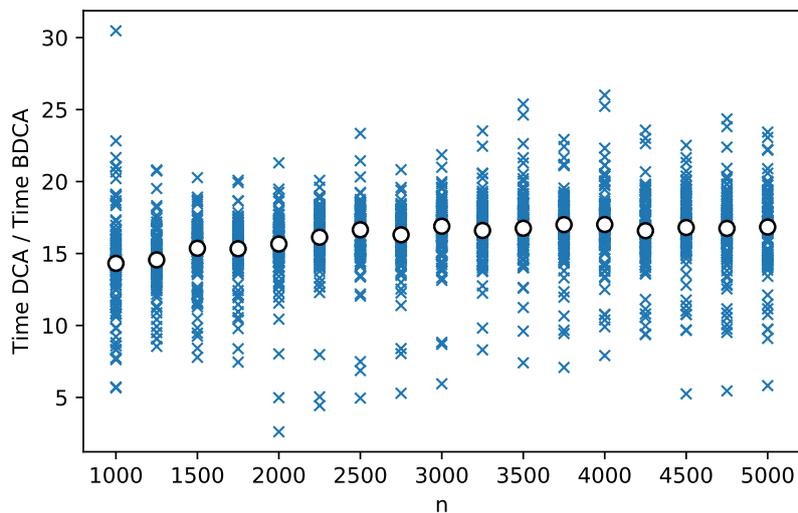


Figure 1: Comparison between DCA and BDCA for checking the copositivity of Horn matrices of order $n \in \{1000, 1250, \dots, 5000\}$. For each size, we represent the ratios of the running time between DCA and BDCA for 100 random starting points (blue crosses) and the median ratio among all of them (white circle).

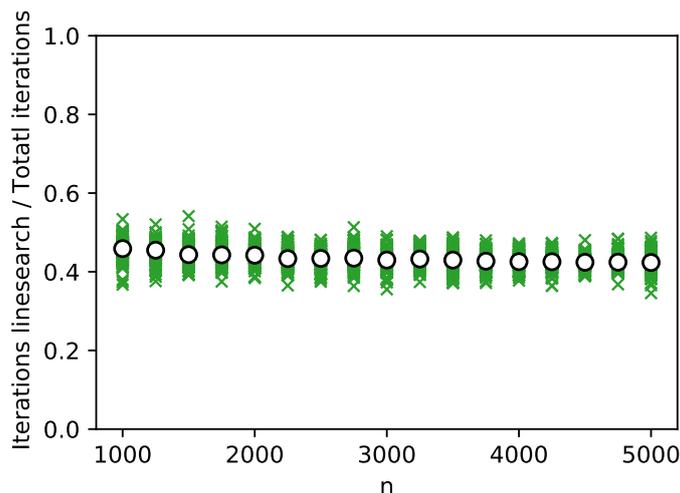


Figure 2: Ratio of the number of iterations at which the boosting step of BDCA was activated with respect the number of iterations performed for testing the copositivity of Horn matrices of order $n \in \{1000, 1250, \dots, 5000\}$. For each size, we represent this ratio for 100 random starting points (green crosses) and the median ratio among all of them (white circle).

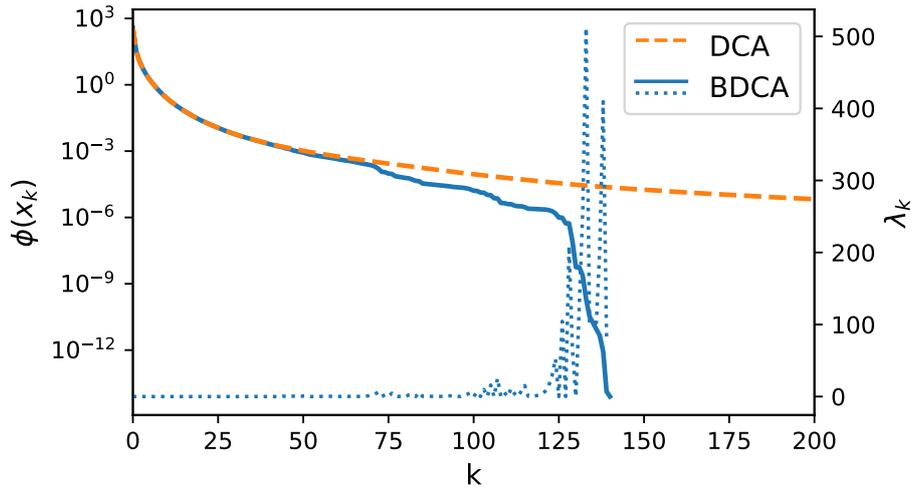


Figure 3: Value of the objective function of DCA and BDCA (using logarithmic scale in the left axis) as well as the step-size used in BDCA (right axis, dotted blue line), with respect to the iteration, for checking the copositivity of the Horn matrix of order $n = 1000$ from the same random starting point.

In our second experiment we considered matrices of the form Q_n^μ as defined in (32). In order to generate hard instances (those which are close to be copositive) we took $\mu := 1.9$. For each size $n \in \{1000, 1250, \dots, 5000\}$, DCA and BDCA were run from the same 100 random starting points generated as in our previous experiment. In this case, we let the algorithms run until they find a negative objective value (which exists because of the non-copositivity of the matrices). We used two stopping criteria, whose results are depicted in Figure 4: on the left, the algorithms were stopped when any negative objective value was found; on the right, the objective value was required to be smaller than -10^{-4} . We do not show any results on the second criterion for n greater than 2000 because DCA becomes extremely slow (for $n = 2000$, the instances solved by DCA required more than 5 minutes on average). This time the advantage of BDCA with respect to DCA increased with the size n , and was significantly greater than in the previous experiment when the second criterion was used.

Remark 5.1. Clearly, BDCA iterations will be more time consuming than those of DCA due to the need of checking condition in Line 7 and the linesearch procedure. Note that the linesearch requires of function evaluations of f , which may be expensive at some applications. For this reason, in our experiments we compare the total CPU running time of the algorithm, as it includes the possibly wasted extra time of the boosting step.

5.2 Solving the ℓ_1 and ℓ_∞ trust-region subproblem

The trust-region subproblem (TRSP) arises in trust-region methods, which are optimization algorithms that consist in replacing the original objective function by a model which is a good approximation of the original function at the current iterate. The models are usually defined to be a quadratic function, in which case the task consists in solving nonconvex

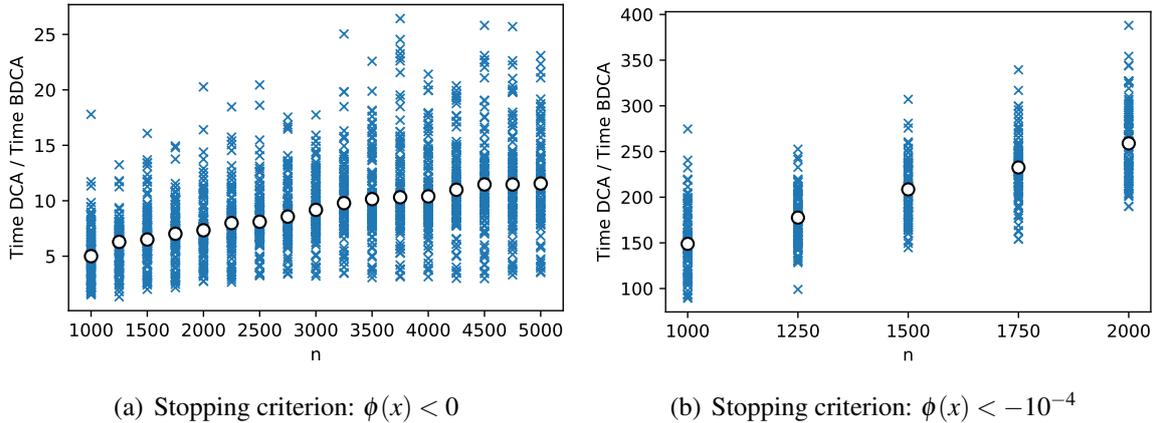


Figure 4: Comparison between DCA and BDCA for detecting the non-copositivity of matrices of various orders n . For each size, we represent the ratios of the running time between DCA and BDCA for 100 random starting points (blue crosses) and the median ratio among all of them (white circle).

quadratic optimization problems of the type

$$\min_{\|x\| \leq r} \frac{1}{2} \langle Ax, x \rangle + \langle b, x \rangle,$$

where A is an $n \times n$ real symmetric matrix, $b \in \mathbb{R}^n$, r is a positive number and $\|\cdot\|$ is any given norm. Of course, the choice of the norm has a serious impact on the difficulty for solving the subproblems.

The application of the DCA for solving the Euclidean trust-region subproblem (the most common choice for the norm) was first proposed in [29] and its convergence was further studied in [16, 31, 33]. Thanks to the structure of the trust-region subproblem, the implementation of the DCA is very simple and efficient, as the iterations are given by (15) and only require matrix-vector products. On the other hand, as observed in [7, Section 7.8] and [12], the choice of the Euclidean norm for the definition of the trust-region has several drawbacks, especially when the problem at hand is box-constrained, in which case the intersection of the Euclidean ball of the trust region with the feasible set has a complicated structure. The choice of the ℓ_∞ norm $\|\cdot\|_\infty$ for the trust-region when the problem involves bounds of the type $l \leq x \leq u$ permits to ensure feasibility by simply requiring similar bounds on the trust-region subproblem. Unfortunately, the ℓ_1 and ℓ_∞ trust-region subproblem are NP-hard [23], a class of problems for which it is commonly believed that there are no polynomial time algorithms.

In this subsection we compare the performance of DCA and BDCA on the challenging ℓ_1 and ℓ_∞ trust-region subproblems. To this aim, we consider the DC decomposition (12)-(13) of the (possibly) nonconvex part of the objective function. Thus, given $\sigma > \max\{\lambda_{\max}(A), 0\}$, the subproblems for the ℓ_1 norm take the form

$$\begin{cases} \min_{x \in \mathbb{R}^n} & g(x) - h(x) = \phi(x) \\ \text{s.t.} & \sum_{i=1}^n |x_i| \leq r, \end{cases} \quad (\mathcal{P}_1)$$

with

$$g(x) := \frac{\sigma}{2} \|x\|^2 + \langle b, x \rangle \quad \text{and} \quad h(x) := \frac{1}{2} \langle (\sigma I - A)x, x \rangle,$$

while for the ℓ_∞ norm these problems are

$$\begin{cases} \min_{x \in \mathbb{R}^n} & g(x) - h(x) = \phi(x) \\ \text{s.t.} & -r \leq x_i \leq r, i = 1, \dots, n. \end{cases} \quad (\mathcal{P}_\infty)$$

Observe that the feasible set of (\mathcal{P}_1) can be equivalently written in terms of 2^n linear constraints of the type $\pm x_1 \pm x_2 \pm \dots \pm x_n \leq r$, so the problem is a particular instance of (\mathcal{P}) . In principle, as mentioned in Remark 3.1, the largest step-size derived in [9] ensuring feasibility could be computed. However, as the feasible set has 2^n linear constraints, computing (4) would be very time-consuming. Nonetheless, given a point y_k and a feasible direction d_k , an upper bound of the maximum value of the step-size $\bar{\lambda}_k$ for (\mathcal{P}_1) is given by the Euclidean norm, since

$$\|y_k + \lambda d_k\|_1 \geq \|y_k + \lambda d_k\| > r, \text{ for all } \lambda > \hat{\lambda}_k,$$

where

$$\hat{\lambda}_k := \frac{\langle y_k, d_k \rangle + \sqrt{\langle y_k, d_k \rangle^2 - \|d_k\|^2(\|y_k\|^2 - r^2)}}{\|y_k\|^2},$$

so one must always choose $\bar{\lambda}_k \leq \hat{\lambda}_k$ to avoid extra time in checking feasibility. On the other hand, for the ℓ_∞ norm, the situation is more favorable, as feasibility can be guaranteed whenever

$$0 \leq \lambda \leq \min_{i \notin I(x_k)} \left\{ \frac{r}{|d_{k,i}|} - \frac{y_{k,i}}{d_{k,i}} \mid d_{k,i} \neq 0 \right\},$$

which coincides with (4), so no time will be wasted in Step 8 of Algorithm 1 if one takes $\bar{\lambda}_k$ satisfying the latter inequalities.

Experiments We used the same parameter setting as in the previous section for Algorithm 1, except for the value of γ , which was increased to 20. The matrix A and the vector b were generated with coordinates randomly and uniformly chosen in $] -1, 1[$, while the value of r was randomly and uniformly chosen in the range $]0, \sqrt{n}/4[$ for (\mathcal{P}_1) and $]0, 1/4[$ for (\mathcal{P}_∞) . For each size $n \in \{1000, 1500, \dots, 5000\}$, DCA and BDCA were run from the same 100 starting points randomly picked inside the trust-region. We stopped the algorithms when $\|d_k\|/\|x_k\| \leq 10^{-8}$ for the first time. The time comparison for both norms are summarized in Figure 5. BDCA was consistently faster than DCA. On average, it was 3.8 times faster for solving (\mathcal{P}_1) and 3.65 times faster for (\mathcal{P}_∞) .

We also show in Figure 6 the percentage of iterations at which the boosting step was activated. We observe that, for all sizes, the linesearch was performed on average in around 80% of the iterations for the case of the ℓ_1 norm, and 40% for the ℓ_∞ norm.

When applied to (\mathcal{P}_1) , both algorithms obtained the same value in 875 instances, BDCA reached a smaller objective value in 18 instances, while DCA did so in 7. When applied to (\mathcal{P}_∞) , BDCA obtained a smaller objective value in 461 instances, while DCA did so in 417 (the value was the same in 22 instances). In Figure 7 we show the behavior of both algorithms for a particular instance with $n = 1000$, which was chosen so that both algorithms attained the same objective value.

We conclude this section with some comments about the rate of convergence of DCA and BDCA. According to Theorem 4.1, both methods are R-linearly convergent, so by (18),

$$\lim_{k \rightarrow \infty} \|x_k - \bar{x}\|^{\frac{1}{k}} \leq \eta,$$

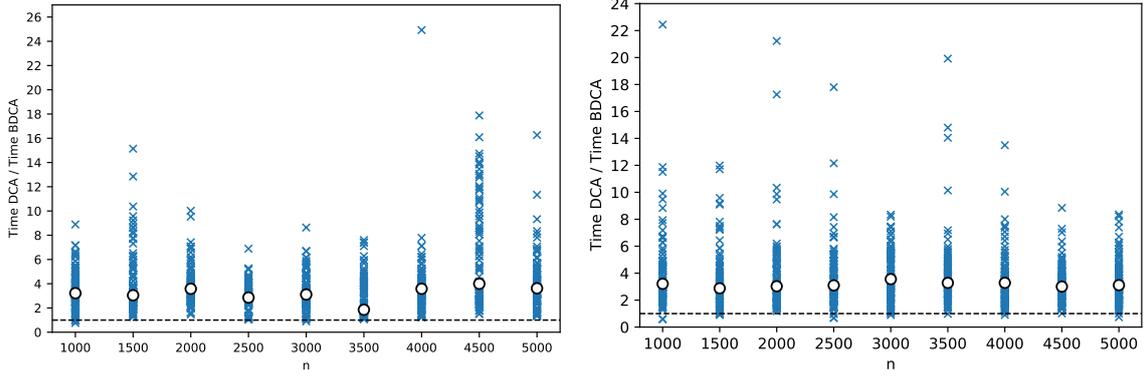


Figure 5: Comparison between DCA and BDCA for solving randomly generated ℓ_1 (left) and ℓ_∞ (right) trust-region subproblems in \mathbb{R}^n , with $n \in \{1000, 1250, \dots, 5000\}$. For each size, we represent the ratios of the running time between DCA and BDCA for 100 random starting points (blue crosses) and the median ratio among all of them (white circle).

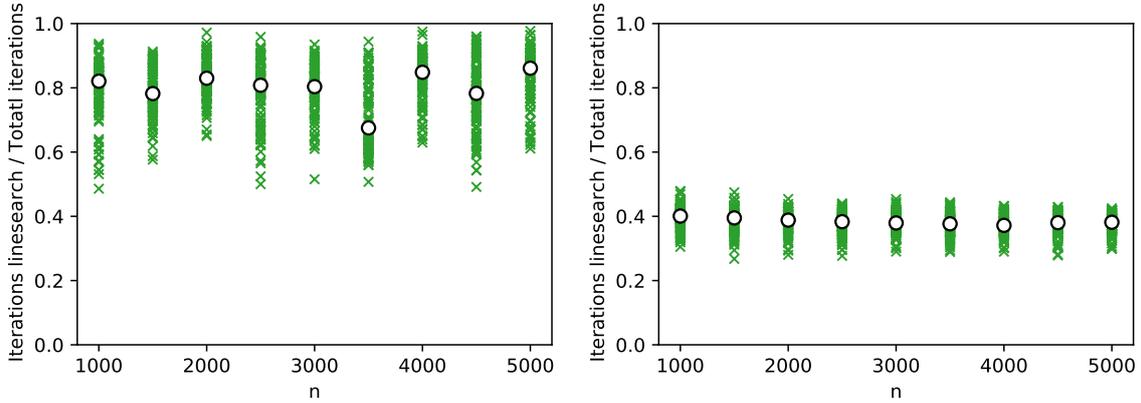


Figure 6: Ratio of the number of iterations at which the boosting step of BDCA was activated with respect to the number of iterations performed for solving randomly generated ℓ_1 (left) and ℓ_∞ (right) trust-region subproblems in \mathbb{R}^n , with $n \in \{1000, 1250, \dots, 5000\}$. For each size, we represent this ratio for 100 random starting points (green crosses) and the median ratio among all of them (white circle).

for some $\eta \in]0, 1[$. In Figure 8, for a particular random instance, we have represented the sequence $\{\|x_k - \bar{x}\|^{1/k}\}$, which was computed for both algorithms after obtaining the limit point \bar{x} with a higher precision. We observe that the value of η is very close to 1 for DCA, while line-searches clearly helped improving this slow rate. A possible explanation about the bad R-linear convergence rate of DCA can be deduced from the proof of Theorem 4.1. The upper bound obtained there is $\frac{\beta}{\rho + \beta}$, with $\beta > \sigma$. When $\lambda_{\max}(Q) > 0$, Assumption 1 holds for ρ as in (14), which would then be equal to $\sigma - \lambda_{\max}(Q)$. Therefore,

$$\frac{\beta}{\rho + \beta} = \frac{\beta}{\sigma - \lambda_{\max}(Q) + \beta} > \frac{\sigma}{2\sigma - \lambda_{\max}(Q)},$$

where the last inequality holds since the left term is an increasing function with respect to β . In our numerical tests we took $\sigma - \lambda_{\max}(Q) = 0.01$ (we numerically observed how larger

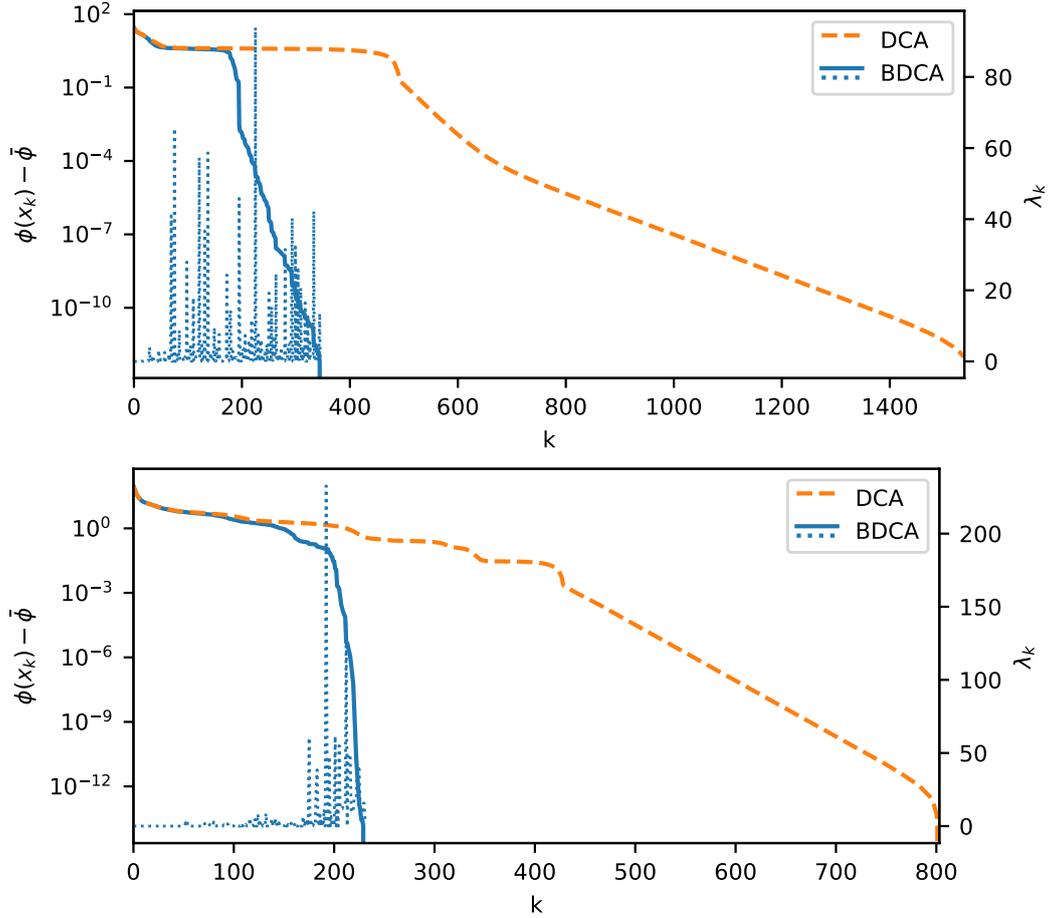


Figure 7: Value of the objective function of DCA and BDCA (using logarithmic scale in the left axis) as well as the step-size used in BDCA (right axis, dotted blue line), with respect to the iteration, for solving a randomly generated ℓ_1 (top) and ℓ_∞ (bottom) trust-region subproblem in \mathbb{R}^{1000} from the same random starting point. Both algorithms attained the same objective value $\bar{\phi}$.

values slowed down both DCA and BDCA). In the random instances shown in Figure 8 $\lambda_{\max}(Q)$ was equal to 25.77 for the ℓ_1 norm and 25.89 for the ℓ_∞ norm, which would give upper bounds on the R-linear convergence rate of 0.9996 for both problems. According to the figures, these bounds seem to be tight, particularly for the ℓ_∞ norm.

5.3 Piecewise quadratic problems with box constraints

Finally, we test BDCA on optimization problems with nonsmooth objective function. To this aim, we consider piecewise quadratic problems with linear constraints of the form

$$\begin{cases} \min_{x \in \mathbb{R}^n} & \phi(x) := \min_{j \in \{1, \dots, m\}} \left\{ \frac{1}{2} \|x - c^j\|^2 \right\} \\ \text{s.t.} & l_i \leq x_i \leq u_i, \quad i = 1, \dots, n; \end{cases} \quad (33)$$

for given $l = (l_1, \dots, l_n), u = (u_1, \dots, u_n) \in \mathbb{R}^n$ and $c^1, \dots, c^m \in \mathbb{R}^n$.

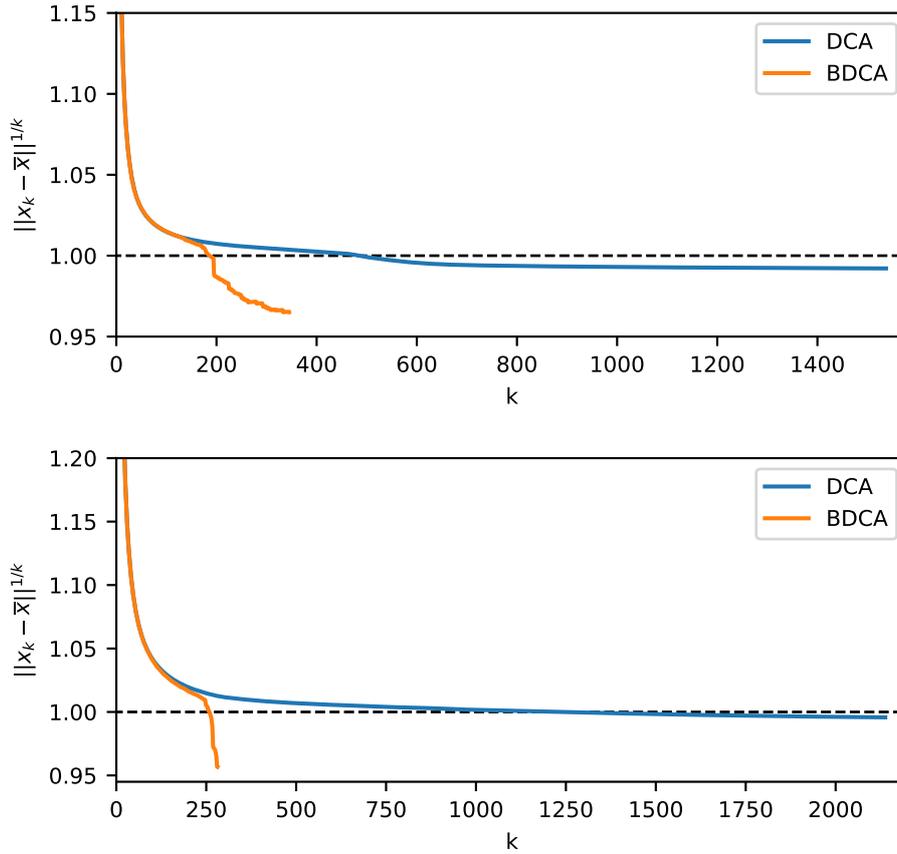


Figure 8: Comparison on the rate of R-linear convergence of DCA and BDCA for solving the same randomly generated ℓ_1 (top) and ℓ_∞ (bottom) trust-region subproblems in \mathbb{R}^{1000} from Figure 7.

As discussed in [26, § 6.3], the objective function of this problem admits the DC decomposition $\phi(x) = g(x) - h(x)$ with

$$g(x) := \frac{1}{2} \sum_{j=1}^m \|x - c^j\|^2 \quad \text{and} \quad h(x) := \max_{l \in \{1, \dots, m\}} \left\{ \frac{1}{2} \sum_{\substack{j=1 \\ j \neq l}}^m \|x - c^j\|^2 \right\}.$$

Experiments For our numerical tests, we generated problems of the form (33) as follows. First, the vector $l \in \mathbb{R}^n$ was generated with coordinates randomly chosen in $] -5, 5[$. Then, we randomly generated a vector $e \in \mathbb{R}^n$ with coordinates in $]0, 5[$ and we set $u := l + e$. Finally, the points $c_1, \dots, c_m \in \mathbb{R}^n$ were generated so that they became unfeasible for all constraints as follows: each component $i \in \{1, \dots, n\}$ of each of these vectors was randomly picked in one of the intervals $]l_i - 10, l_i[$ or $]u_i, u_i + 10[$. In our experiment, for each $n \in \{100, 150, \dots, 1000\}$ and each $m \in \{100, 150, \dots, 1000\}$, DCA and BDCA were run from the same 100 starting points randomly picked inside the feasible set. We used the same parameter setting and stopping criterion for the algorithms as in the previous TRSP experiments. The time comparison is shown in Figure 9, where we represent the median ration between DCA and BDCA among the 100 instances. Detailed results for all 100 runs

are shown in Figure 10(a) for fixed $n = 500$ and Figure 10(b) for fixed $m = 500$. From the results of this comparison we infer that the superiority of BDCA increases as m does, while it stabilizes as n increases. As in previous experiments, we show in Figure 11 the percentage of iterations at which the boosting step was activated. The objective value attained by DCA and BDCA were the same in all instances.

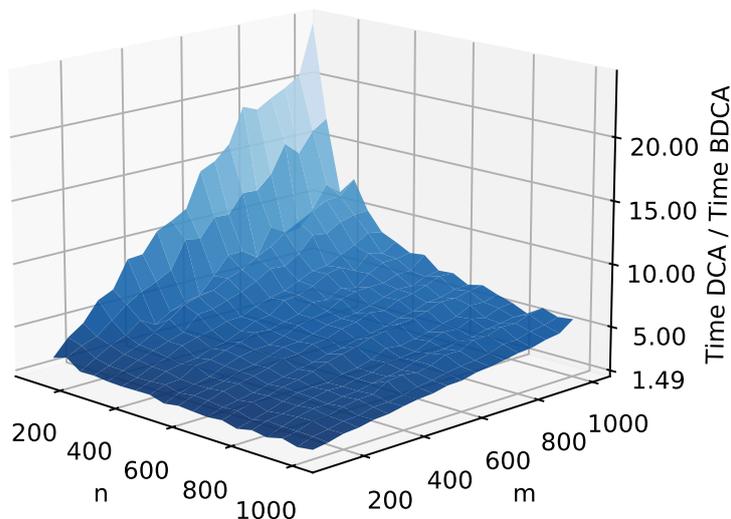


Figure 9: Comparison between DCA and BDCA for solving randomly generated piecewise quadratic problems with m pieces in \mathbb{R}^n , for $n \in \{100, 200, \dots, 1000\}$ and $m \in \{100, 200, \dots, 1000\}$. For each pair (n, m) , we represent the median of the ratios of the running time between DCA and BDCA for 100 random starting points.

6 Concluding remarks

We have extended the Boosted DC Algorithm for solving linearly constrained DC programming. The algorithm is proved to provide KKT points of the constrained problem. In addition, we have shown why this approach cannot be extended to more general convex constraints. The theoretical results were confirmed by some numerical experiments for testing the copositivity of matrices and for solving ℓ_1 and ℓ_∞ trust-region subproblems. For copositive matrices, BDCA was on average fifteen times faster than DCA; for non-copositive ones, this advantage was much more superior; and for trust-region subproblems, BDCA was more than three times faster than DCA. We also considered piecewise quadratic problems with box constraints, which thus have nonsmooth objective functions. We observed again that BDCA was faster than DCA and, further, that the advantage was more noticeable as the number of pieces of the objective function increased. Future research includes investigation of alternative approaches to derive a Boosted DCA that permits to address any type of constrained DC programs. It would also be interesting to combine BDCA with the inertial technique employed in [27].

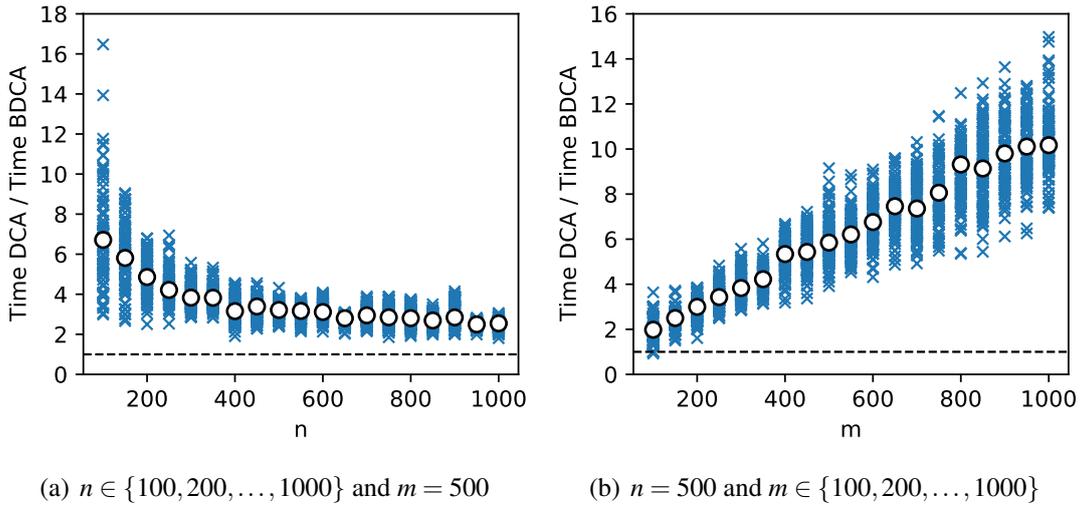


Figure 10: Comparison between DCA and BDCA for solving randomly generated piecewise quadratic problems with m pieces in \mathbb{R}^n , for $n \in \{100, 200, \dots, 1000\}$ and $m = 500$ (left), and for $n = 500$ and $m \in \{100, 200, \dots, 1000\}$ (right). For each problem, we represent the ratios of the running time between DCA and BDCA for 100 random starting points (blue crosses) and the median ratio among all of them (white circle).

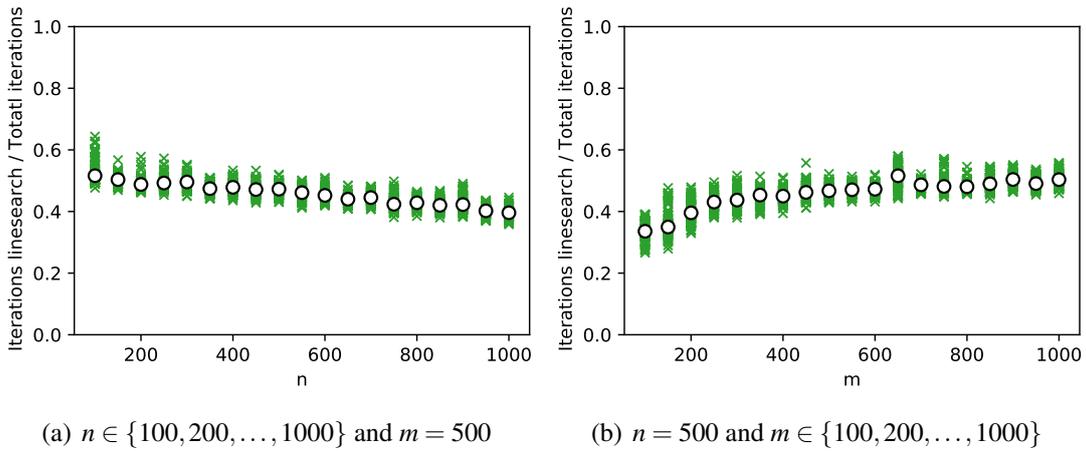


Figure 11: Ratio of the number of iterations at which the boosting step of BDCA was activated with respect to the number of iterations performed for solving randomly generated piecewise quadratic problems with m pieces in \mathbb{R}^n , for $n \in \{100, 200, \dots, 1000\}$ and $m = 500$ (left), and for $n = 500$ and $m \in \{100, 200, \dots, 1000\}$ (right). For each size, we represent this ratio for 100 random starting points (green crosses) and the median ratio among all of them (white circle).

Acknowledgments FJAA and RC were partially supported by the Ministry of Science, Innovation and Universities of Spain and the European Regional Development Fund (ERDF) of the European Commission (PGC2018-097960-B-C22), and by the Generalitat Valenciana (AICO/2021/165). PTV was supported by Vietnam Ministry of Education and Training Project hosting by the University of Technology and Education, Ho Chi Minh City Vietnam (2023-2024).

References

- [1] Aragón Artacho, F.J., Fleming, R., Vuong, P.T.: Accelerating the DC algorithm for smooth functions. *Math. Program.* 169(1), 95–118 (2018)
- [2] Aragón, F.J., Goberna, M.A., López, M.A., Rodríguez, M.M.L.: *Nonlinear Optimization*. Springer Undergraduate Texts in Mathematics and Technology (2019)
- [3] Aragón Artacho, F.J., Vuong, P.T.: The boosted difference of convex functions algorithm for nonsmooth functions. *SIAM J. Optim.* 30(1), 980–1006 (2020)
- [4] Bauschke, H.H., Combettes, P.L.: *Convex analysis and monotone operator theory in Hilbert spaces, 2nd ed.* Springer, Berlin (2017)
- [5] Bomze, I.M.: Copositive optimization-recent developments and applications. *European J. Oper. Res.* 216(3), 509–520 (2012)
- [6] Burer, S.: On the copositive representation of binary and continuous nonconvex quadratic programs. *Math. Program.* 120(2), 479–495 (2009)
- [7] Conn, A.R., Gould, N.I.M., Toint, P.L.: *Trust-region methods*. MPS/SIAM Series on Optimization (2000)
- [8] Dür, M., Hiriart-Urruty, J.-B.: Testing copositivity with the help of difference-of-convex optimization. *Math. Program.* 140(1), 31–43 (2013)
- [9] Ferreira, O.P., Santos, E.M., Souza, J.C.O.: Boosted scaled subgradient method for DC programming. ArXiv: <https://arxiv.org/abs/2103.10757> (2021)
- [10] Fukushima, M., Mine, H.: A generalized proximal point algorithm for certain non-convex minimization problems. *Int. J. Syst. Sci.* 12(8), 989–1000 (1981)
- [11] Geremew, W., Nam, N.M., Semenov, A., Boginski, V., Pasiliao, E.: A DC programming approach for solving multicast network design problems via the Nesterov smoothing technique. *J. Glob. Optim.* 72(4), 705–729 (2018)
- [12] Geremew, S., Mouffe, M., Toint, P.L., Weber-Mendonça, M.: A recursive ℓ_∞ -trust-region method for bound-constrained nonlinear optimization. *IMA J. Numer. Anal.* 28(4), 827–861 (2008)
- [13] Johnson, C.R., Reams, R.: Constructing copositive matrices from interior matrices. *Electron. J. Linear Al.*, 17, 9–20 (2008)
- [14] de Klerk, E., Pasechnik, D.V.: Approximation of the stability number of a graph via copositive programming. *SIAM J. Optim.* 12(4), 875–892 (2002)
- [15] Le Thi, H.A., Pham Dinh, T.: DC Programming and DCA: Thirty Years of Developments. *Math. Program.* 169(1), 5–68 (2018)
- [16] Le Thi, H.A., Pham Dinh, T., Yen, N.D.: Behavior of DCA sequences for solving the trust-region subproblem. *J. Global Optim.* 53(2), 317–329 (2012)

- [17] Le Thi, H.A., Pham Dinh, T.: The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Ann. Oper. Res.* 133(1-4), 23–46 (2005)
- [18] Le Thi, H.A., Huynh V.N., Pham Dinh, T.: Convergence analysis of Difference-of-Convex Algorithm with subanalytic data. *J. Optim. Theory Appl.* 179(1), 103–126 (2018)
- [19] Luo, Z.Q., Tseng, P.: Error bound and convergence analysis of matrix splitting algorithms for the affine variational inequality problem. *SIAM J. Optim.* 2(1), 43–54 (1992)
- [20] Mine, H., Fukushima, M.: A minimization method for the sum of a convex function and a continuously differentiable function. *J. Optim. Theory Appl.* 33(1), 9–23 (1981)
- [21] Mordukhovich, B.S.: *Variational Analysis and Generalized Differentiation, vol. II.* Springer-Verlag Berlin Heidelberg (2006)
- [22] Moosaei, H., Bazikar, F., Ketabchi, S., Hladík, M.: Universum parametric-margin ν -support vector machine for classification using the difference of convex functions algorithm. *Appl. Intell.* 52(3), 2634–2654 (2022)
- [23] Murty, K.G., Kabadi, S.N.: Some NP-complete problems in quadratic and nonlinear programming. *Math. Program.* 39(2), 117–129 (1987)
- [24] Nam, N.M., Geremew, W., Reynolds, R., Tran, T.: Nesterov’s smoothing technique and minimizing differences of convex functions for hierarchical clustering. *Optim. Lett.* 12(3), 455–473 (2018)
- [25] Nie, J., Yang, Z., Zhang, X.: A complete semidefinite algorithm for detecting copositive matrices and tensors. *SIAM J. Optim.* 28(4), 2902–2921 (2018)
- [26] de Oliveira, W.: Proximal bundle methods for nonsmooth DC programming. *J. Global Optim.* 75(2), 523–563 (2019)
- [27] de Oliveira, W., Tcheou, M.P.: An inertial algorithm for DC programming. *Set-Valued Var. Anal.* 27(4), 895–919 (2019)
- [28] Pham Dinh, T., Le Thi, H.A.: Convex analysis approach to DC programming: theory, algorithms and applications. *Acta Math. Vietnam.* 22(1), 289–355 (1997)
- [29] Pham Dinh, T., Le Thi, H.A.: A D.C. optimization algorithm for solving the trust-region subproblem. *SIAM J. Optim.* 8(2), 476–505 (1998)
- [30] Pham Dinh, T., Le Thi, H.A., Akoa, F. : Combining DCA (DC Algorithms) and interior point techniques for large-scale nonconvex quadratic programming. *Optim. Methods Softw.* 23(4), 609–629 (2008)
- [31] Tuan H.N.: Convergence rate of the Pham Dinh-Le Thi algorithm for the trust-region subproblem. *J. Optim. Theory Appl.* 154(3), 904–915 (2012)
- [32] Tuan H.N.: Linear convergence of a type of iterative sequences in nonconvex quadratic programming. *J. Math. Anal. Appl.* 423(2), 1311–1319 (2015)

- [33] Tuan H.N., Yen, N.D.: Convergence of the Pham Dinh-Le Thi's algorithm for the trust-region subproblem. *J. Glob. Optim.* 55(2), 337-347 (2013)
- [34] Rockafellar, R.T.: *Convex Analysis*. Princeton University Press (1972)
- [35] Rockafellar, R.T., Wets, R.J.-B.: *Variational Analysis*, Grundlehren Math. Wiss. 317, Springer, New York (1998)
- [36] Xu, H.M., Xue, H., Chen, X.H., Wang, Y.Y.: Solving indefinite kernel support vector machine with difference of convex functions programming. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (2017)