Optim Lett (2018) 12:985–996
https://doi.org/10.1007/s11590-017-1203-0

ORIGINAL PAPER

DC decomposition based branch-and-bound algorithms for box-constrained quadratic programs

Cheng Lu[1] · **Zhibin Deng**[2,3]

Received: 5 October 2016 / Accepted: 28 September 2017 / Published online: 7 October 2017

Abstract The difference-of-convex (DC) decomposition is an effective method for designing a branch-and-bound algorithm. In this paper, we design two new branch-and-bound algorithms based on DC decomposition, to find global solutions of nonconvex box-constrained quadratic programming problems, and compare the efficiency of the proposed algorithms with two previous state-of-the-art branch-and-bound algorithms. Numerical experiments are conducted to show the competitiveness of the proposed algorithms on 20–60 dimensional box-constrained quadratic programming problems.

Keywords Quadratic program · Difference-of-convex · Branch-and-bound algorithm · Global optimization · Convex relaxation

Lu's research has been supported by National Natural Science Foundation of China Grant No. 11701177, Fundamental Research Funds for the Central Universities Grant No. 2017MS058. Deng's research has been supported by National Natural Science Foundation of China Grant No. 11501543, and Research Foundation of UCAS Grants Nos. Y65201VY00 and Y65302V1G4.

✉ Zhibin Deng
zhibindeng@ucas.ac.cn

Cheng Lu
lucheng1983@163.com

1 School of Economics and Management, North China Electric Power University, Beijing 102206, China

2 School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China

3 Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, China

1 Introduction

Consider the following box-constrained quadratic programming problem:

$$\begin{aligned} v^* = \min \quad & F(x) = \frac{1}{2}x^T Qx + c^T x \\ \text{s.t.} \quad & x \in [0, 1]^n, \end{aligned} \tag{Box-QP}$$

where $Q \in \mathbb{R}^{n\times n}$ is a real symmetric matrix and $c \in \mathbb{R}^n$ is a real vector. Nonconvex Box-QP is NP-hard in general, even when Q has only one negative eigenvalue [14].

Many branch-and-bound algorithms have been proposed to find global solutions of Box-QP problems in the literature (see, e.g., [5–9,12,21,23]). One of the most important ingredients of a branch-and-bound algorithm is to find a good relaxation method (i.e., the lower bound method). Among different relaxations, the SDP+RLT relaxation [3], which combines the semidefinite constraints [22] and the well-known RLT constraints [18,19] together, is a very tight relaxation for Box-QP problems. However, the SDP+RLT relaxation is not a competitive choice for a branch-and-bound algorithm, due to its high computational cost. A looser but easily computed lower bound is desired for designing an overall efficient branch-and-bound algorithm.

The DC decomposition is an effective method for designing convex relaxations [1,2,8]. The main idea of DC decomposition is to decompose the objective function $F(x) = \frac{1}{2}x^T Qx + c^T x$ to the difference of two convex functions $G(x)$ and $H(x)$, i.e., $F(x) = G(x) - H(x)$, and then relax $-H(x)$ to a convex function $L(x)$ such that $L(x) \leqslant -H(x)$ for any feasible solution $x \in [0, 1]^n$. Various DC decompositions have been proposed in the literature. For example, An and Tao [1] decomposed $F(x) = [F(x) - \frac{1}{2}\lambda_{\min} x^T x] + \frac{1}{2}\lambda_{\min} x^T x$, where $\lambda_{\min}$ is the smallest eigenvalue of Q. Later they designed an ellipsoidal relaxation, and used the DC approximation algorithm to solve the ellipsoidal relaxation problem [2]. Cambini and Sodinia reformulated the objective function to $F(x) = \frac{1}{2}x^T Q_0 x + c^T x - \frac{1}{2}\sum_{i=1,\ldots,p} \lambda_i (v_i^T x)^2$, where Q_0 is positive semidefinite, $p \leq n$ is the number of negative eigenvalue of Q, λ_i is the negative eigenvalue and v_i is the corresponding eigenvector, $i = 1, \ldots, p$. The main feature of the above methods is that the quadratic programming relaxations can be solved much more efficiently than a semidefinite programming relaxation. However, these lower bounds are generally not as tight as that of a semidefinite programming relaxation. In recent years, many new relaxation methods of high quality have been proposed [11,16,24]. However, these papers only study the effectiveness of these lower bounds. The efficiency of these bounds in a branch-and-bound algorithm is not known yet. Different from their perspectives, this paper aims to design two complete branch-and-bound algorithms for globally solving Box-QP programs. Our main concern is the efficiency of the proposed branch-and-bound algorithms, rather than the performance of certain lower bound methods.

The main contribution of this paper is that it proposes two branch-and-bound algorithms with new DC decompositions. Different from previous DC decompositions designed in [1,2,8], we solve semidefinite relaxations and doubly nonnegative relaxations of Box-QP problems, and use the optimal solutions of these relaxation problems to construct tighter relaxations. The proposed algorithms are compared with the QUADPROGBB algorithm [9] and BARON [21], and numerical results show the competitiveness of the proposed algorithms.

The rest of this paper is arranged as follows. In Sect. 2, we propose the first DC decomposition based branch-and-bound algorithm, in which the convex quadratic programming relaxation is constructed from the solution of the conventional semidefinite relaxation problem. In Sect. 3, we design the second branch-and-bound algorithm with an improved DC decomposition, which is constructed from the solution of a doubly nonnegative relaxation problem. Numerical results are presented in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 A branch-and-bound algorithm based on DC decomposition

In this section, we design the first DC decomposition based branch-and-bound algorithm. For the nonconvex quadratic objective function $F(x) = \frac{1}{2}x^T Qx + c^T x$, we decompose it into the difference of two convex functions:

$$F(x) = x^T\left[\frac{1}{2}Q + diag(\lambda)\right]x + c^T x - \sum_{i=1}^{n}\lambda_i x_i^2,$$

where $\lambda := (\lambda_1, \ldots, \lambda_n)^T \geqslant 0$, $diag(\lambda)$ is the diagonal matrix with λ_i being its i-th diagonal entry, and $\frac{1}{2}Q + diag(\lambda)$ is positive semidefinite (denoted as $\frac{1}{2}Q + diag(\lambda) \succeq 0$). Assume that the variable x_i is bounded by $l_i \leqslant x_i \leqslant u_i$, then we have $-(l_i + u_i)x_i + l_i u_i \leqslant -x_i^2$ for all $x_i \in [l_i, u_i]$, and the function $F(x)$ can be relaxed to

$$P^{\lambda}_{[l,u]}(x) = x^T\left[\frac{1}{2}Q + diag(\lambda)\right]x + c^T x - \sum_{i=1}^{n}\lambda_i(l_i + u_i)x_i + \sum_{i=1}^{n}\lambda_i l_i u_i,$$

which is convex and satisfies $P^{\lambda}_{[l,u]}(x) \leqslant F(x)$ for all $x \in [l, u] := \prod_{i=1}^{n}[l_i, u_i]$. Thus, the problem $\min_{x\in[l,u]} P^{\lambda}_{[l,u]}(x)$, denoted as $\mathrm{DCR}_{[l,u]}$, is a convex quadratic programming problem that provides a lower bound of $F(x)$ over $x \in [l, u]$.

Based on the convex relaxation $\mathrm{DCR}_{[l,u]}$, the first branch-and-bound algorithm (named as DC-BB algorithm) is designed and presented in Fig. 1. In the t-th enumeration of DC-BB algorithm, the node that has the smallest lower bound is selected in Line 8. Based on this node selection rule, the lower bound L^t defined in Line 8 satisfies $L^t \leqslant v^*$, where v^* denotes the optimal value of problem Box-QP. Besides, an upper bound U^* is estimated by recording the best known solution in Lines 22 and 29. If $U^* - L^t \leqslant \varepsilon$, then the DC-BB algorithm terminates, and $U^* - v^* \leqslant \varepsilon$ is satisfied. The following results guarantee the convergence of the DC-BB algorithm.

Lemma 1 *Given a parameter λ such that $\lambda \geqslant 0$ and $\frac{1}{2}Q + diag(\lambda) \succeq 0$, let $\bar{x}$ be the optimal solution of $DCR_{[l,u]}$ and $i^* = \arg\max_{i\in\{1,\ldots,n\}} r_i$, where $r_i = \lambda_i[(l_i + u_i)\bar{x}_i - l_i u_i - \bar{x}_i^2]$. For any given $\varepsilon > 0$, let $\delta = (\frac{4\varepsilon}{n\lambda_{i^*}})^{\frac{1}{2}}$. If $u_{i^*} - l_{i^*} \leqslant \delta$, then $F(\bar{x}) - P^{\lambda}_{[l,u]}(\bar{x}) \leqslant \varepsilon$.*

```
Input: A Box-QP instance, and a given error tolerance $\varepsilon > 0$.
 1: Initialize $[l^0, u^0] = [0,1]^n$, Set $t = 0$.
 2: Compute a $\lambda$, such that $\frac{1}{2}Q + diag(\lambda) \succeq 0$, $\lambda \geqslant 0$.
 3: Solve $\mathrm{DCR}_{[l^0,u^0]}$ for its optimal solution $x^0$ and optimal value $L^0$.
 4: Set $U^* = F(x^0)$, $x^* = x^0$
 5: Construct $\mathcal{P}$ and insert $\{[l^0,u^0], x^0, L^0\}$ into $\mathcal{P}$.
 6: loop
 7:    Set $t \leftarrow t+1$.
 8:    Choose a problem from $\mathcal{P}$, denoted as $\{[l^t,u^t], x^t, L^t\}$, such that the bound $L^t$ is the
       smallest one in $\mathcal{P}$.
 9:    Delete the chosen subproblem from $\mathcal{P}$.
10:    if $U^* - L^t \leqslant \varepsilon$ then
11:       return $x^*$ and terminate.
12:    end if
13:    Compute $r_i^t = \lambda_i[(l_i^t + u_i^t)x_i^t - l_i^t u_i^t - (x_i^t)^2]$ for $i = 1, ..., n$
14:    Set $i^* = \mathrm{argmax}_{i \in \{1,\dots,n\}} r_i^t$, and $z_{i^*}^* = \frac{1}{2}(l_{i^*}^t + u_{i^*}^t)$.
15:    Construct box $[l^a, u^a]$ by setting $l^a = l^t$, $u_i^a = u_i^t$ for $i \neq i^*$ and $u_{i^*}^a = z_{i^*}$.
16:    Construct box $[l^b, u^b]$ by setting $u^b = u^t$, $l_i^b = l_i^t$ for $i \neq i^*$ and $l_{i^*}^b = z_{i^*}$.
17:    Solve $\mathrm{DCR}_{[l^a,u^a]}$ for its optimal solution $x^a$ and optimal value $L^a$.
18:    if $L^a \leqslant U^*$ then
19:       insert $\{[l^a,u^a], x^a, L^a\}$ into $\mathcal{P}$.
20:    end if
21:    if $U^* > F(x^a)$ then
22:       set $U^* = F(x^a)$, $x^* = x^a$.
23:    end if
24:    Solve $\mathrm{DCR}_{[l^b,u^b]}$ for its optimal solution $x^b$ and optimal value $L^b$.
25:    if $L^b \leqslant U^*$ then
26:       insert $\{[l^b,u^b], x^b, L^b\}$ into $\mathcal{P}$.
27:    end if
28:    if $U^* > F(x^b)$ then
29:       set $U^* = F(x^b)$, $x^* = x^b$.
30:    end if
31: end loop
```

Fig. 1 DC-BB algorithm for solving Box-QPs

Proof Let $\delta = (\frac{4\varepsilon}{n\lambda_{i^*}})^{\frac{1}{2}}$. Given the condition $u_{i^*} - l_{i^*} \leqslant (\frac{4\varepsilon}{n\lambda_{i^*}})^{\frac{1}{2}}$, we have $r_{i^*} = \lambda_{i^*}[(l_{i^*} + u_{i^*})\bar{x}_{i^*} - l_{i^*}u_{i^*} - \bar{x}_{i^*}^2] \leqslant \lambda_{i^*}\frac{(u_{i^*}-l_{i^*})^2}{4} \leqslant \frac{\varepsilon}{n}$. Then $F(\bar{x}) - P_{(l,u)}^{\lambda}(\bar{x}) = \sum_{i=1}^{n} r_i \leqslant \varepsilon$. □

Theorem 1 *For any given $\varepsilon > 0$, the DC-BB algorithm terminates in a finite number of iterations and returns a feasible solution x^* such that $F(x^*) \leqslant v^* + \varepsilon$.*

Proof Based on Lemma 1, we know that if i^* is selected as a branching direction in Line 14 of the DC-BB algorithm, then $u_{i^*}^t - l_{i^*}^t > (\frac{4\varepsilon}{n\lambda_{i^*}})^{\frac{1}{2}}$. Otherwise, we have $U^* - L^t \leqslant F(x^t) - L^t \leqslant \varepsilon$, which means that the algorithm should have already terminated in Line 11. Hence, the length of the interval being selected for branching is larger than $(\frac{4\varepsilon}{n\lambda_{i^*}})^{\frac{1}{2}}$. Since the initial set $[l^0, u^0] = [0,1]^n$ is bounded, the DC-BB algorithm must terminates in a finite number of iterations, and returns a solution x^* such that $F(x^*) - v^* \leqslant \varepsilon$. □

The tightness of $\mathrm{DCR}_{[l,u]}$ depends on the choice of the parameter λ. A different λ may lead to a different lower bound, which affects the efficiency of the DC-BB algorithm significantly. Motivated by the results in [24], we construct a high quality relaxation with the following method: find a $\lambda \in \mathbb{R}^n$, such that $\frac{1}{2}Q + diag(\lambda) \succeq 0$, $\lambda \geqslant 0$, and the optimal value of $\mathrm{DCR}_{[l^0,u^0]}$ with $[l^0, u^0] = [0, 1]^n$ is maximized. This problem can be formulated as

$$\max_{\lambda \geqslant 0, \frac{1}{2}Q+diag(\lambda)\succeq 0} \min_{x\in[0,1]} P^{\lambda}_{[0,1]}(x),$$

which is equivalent to the following problem (see Theorem 1 of [24]):

$$\begin{aligned} \max\ & \frac{1}{2}\sigma \\ \text{s.t.}\ & \begin{bmatrix} -\sigma & (c-\lambda)^T \\ c-\lambda & Q+2\mathrm{diag}(\lambda) \end{bmatrix} \succeq 0, \\ & \lambda \geqslant 0. \end{aligned} \tag{SDR}$$

The solution λ^* is computed once and for all at the beginning of the DC-BB algorithm and used throughout the algorithm. The above implementation is similar to the one of quadratic convex reformulation for binary quadratic programming problems in [4,13], in which a reformulation that provides the tightest relaxation bound is constructed before solving the problem with a general-purpose global optimization solver.

3 An enhanced convex relaxation problem

In this section, we derive an enhanced convex relaxation that is tighter than $\mathrm{DCR}_{[l,u]}$. Let $N \in \mathbb{R}^{n\times n}$ be a nonnegative symmetric matrix with zero diagonal entries. We rewrite the function $P^{\lambda}_{[l,u]}(x)$ as $P^{\lambda}_{[l,u]}(x) - x^T Nx + x^T Nx$. For any vector $x \in [l, u]$, we have $x_i x_j \geqslant l_i x_j + l_j x_i - l_i l_j$, $i, j \in \{1, \dots, n\}$. By choosing an N such that $\frac{1}{2}Q + diag(\lambda) - N \succeq 0$, we define the following convex function:

$$\bar{P}^{\lambda,N}_{[l,u]}(x) = P^{\lambda}_{[l,u]}(x) - x^T Nx + \sum_{i=1}^{n}\sum_{j=1}^{n} N_{ij}(l_i x_j + l_j x_i - l_i l_j).$$

Problem Box-QP is then relaxed to the problem $\min_{x\in[l,u]} \bar{P}^{\lambda,N}_{[l,u]}(x)$, which is denoted as Enhanced-$\mathrm{DCR}_{[l,u]}$.

Based on the enhanced convex relaxation, we propose an enhanced branch-and-bound algorithm by modifying the DC-BB algorithm in Fig. 1: the $\mathrm{DCR}_{[l,u]}$ relaxations in Lines 3, 17 and 24 are replaced by Enhanced-$\mathrm{DCR}_{[l,u]}$, and the definition of r_i^t in Line 13 is replaced by $r_i^t = \lambda_i[(l_i^t + u_i^t)x_i^t - l_i^t u_i^t - (x_i^t)^2] + \sum_{j=1}^n N_{ij}(x_i^t - l_i^t)(x_j^t - l_j^t)$. The modified algorithm will be called as EBB algorithm. To show the convergence of the EBB algorithm, we need the next lemma.

Lemma 2 *Given $\lambda \geqslant 0$ and $N \geqslant 0$ such that $\frac{1}{2}Q + diag(\lambda) - N \succeq 0$, let $\bar{x}$ be the optimal solution of Enhanced-$DCR_{[l,u]}$ and $i^* = \arg\max_{i\in\{1,\dots,n\}} r_i$, where $r_i =$*

$\lambda_i[(l_i+u_i)\bar{x}_i - l_iu_i - \bar{x}_i^2] + \sum_{j=1}^n N_{ij}(\bar{x}_i - l_i)(\bar{x}_j - l_j)$. *Then, for any given* $\varepsilon > 0$, *there exists a* $\delta > 0$, *such that if* $u_{i^*} - l_{i^*} \leqslant \delta$, *then* $F(\bar{x}) - \bar{P}^{\lambda,N}_{[l,u]}(\bar{x}) \leqslant \varepsilon$.

Proof Since $x_i - l_i \leq u_i - l_i \leqslant 1$ for all $i = 1,\ldots,n$, we have $r_i \leqslant \frac{(u_i-l_i)^2}{4} + \sum_{j=1}^n N_{ij}(u_i - l_i) \leqslant (\frac{1}{4} + \sum_{j=1}^n N_{ij})(u_i - l_i)$. Let $\rho = \frac{1}{4} + \sum_{j=1}^n N_{ij}$ and $\delta = \frac{\varepsilon}{n\rho} > 0$. If $u_{i^*} - l_{i^*} \leqslant \delta$, then $r_{i^*} \leqslant \frac{\varepsilon}{n}$ and $F(\bar{x}) - \bar{P}^{\lambda,N}_{[l,u]}(\bar{x}) = \sum_{i=1}^n r_i \leqslant \varepsilon$. □

Based on Lemma 2, we provide the convergence result in the next theorem, which can be proved in the same line as Theorem 1.

Theorem 2 *For any given* $\varepsilon > 0$, *the EBB algorithm terminates in a finite number of iterations and returns a feasible solution* x^* *such that* $F(x^*) - v^* \leqslant \varepsilon$.

Similar to the analysis in Sect. 2, the tightness of Enhanced-DCR$_{[l,u]}$ depends on the choice of λ and N. To obtain a high quality convex relaxation, we adopt the following parameter strategy: Find a pair of (λ, N) such that $\frac{1}{2}Q + diag(\lambda) - N \succeq 0$, $\lambda \geqslant 0$, $N \geqslant 0$, and the optimal value of Enhanced-DCR$_{[0,1]}$ is maximized. This strategy can formulated as

$$\begin{aligned} \max_{\lambda,N} \quad & \min_{x\in[0,1]^n} \ \bar{P}^{\lambda,N}_{[0,1]^n}(x) \\ \text{s.t.} \quad & \frac{1}{2}Q + diag(\lambda) - N \succeq 0, \\ & \lambda \geqslant 0,\ N \geqslant 0. \end{aligned} \tag{P1}$$

The next theorem shows that problem P1 is equivalent to a semidefinite programming problem.

Theorem 3 *Problem P1 is equivalent to the following problem:*

$$\begin{aligned} \max \quad & \frac{1}{2}\sigma \\ \text{s.t.} \quad & \begin{bmatrix} -\sigma & (c-\lambda)^T \\ c-\lambda & Q + 2diag(\lambda) - 2N \end{bmatrix} \succeq 0, \\ & \lambda \geqslant 0,\ N \geqslant 0. \end{aligned} \tag{P2}$$

We omit the proof of Theorem 3 since it can be derived similar to the one of Theorem 1 in [24]. Also note that the dual problem of P2 is

$$\begin{aligned} \min \quad & \frac{1}{2}Q \cdot X + c^T x \\ \text{s.t.} \quad & X_{ii} \leqslant x_i,\ i = 1,\ldots,n, \\ & X_{11} = 1,\ X \succeq 0,\ X \geqslant 0, \end{aligned} \tag{DNN}$$

which is a doubly nonnegative relaxation of Box-QP, and is indeed tighter than SDR. Thus, the Enhanced-DCR$_{[l,u]}$ relaxation is tighter than the DCR$_{[l,u]}$ relaxation at the root node.

For a problem with tens of variables, the DNN relaxation cannot be solved efficiently by using a typical interior-point-based algorithm, since there may involve $O(n^2)$ linear constraints in $X \geqslant 0$. To overcome this difficulty, we use the following iterative procedure for solving the DNN problem[1]: At first, an initial problem, which drops the nonnegativity constraint $X \geqslant 0$ in problem DNN, is solved to obtain a solution $\bar{X}$. Then, the relaxation is resolved after adding constraints of the form $X_{ij} \geqslant 0$ for i, j's such that $\bar{X}_{ij} < 0$. This procedure is repeated until $\bar{X} \geqslant 0$ is satisfied. This iterative procedure is generally much more efficient than a direct application of an interior-point algorithm for solving the original DNN problem.

Remark 1 It is worth to point out that in our Enhanced-DCR$_{[l,u]}$ relaxation, the non-diagonal entries of Q are perturbed. The idea of non-diagonal perturbation has also been considered in the generalized α-BB algorithm [20]. The main difference between our algorithm and the generalized α-BB algorithm is that our convex relaxation Enhanced-DCR$_{[l,u]}$ is defined as $\min_{x\in[l,u]} \bar{P}^{\lambda,N}_{[l,u]}(x)$, which is a convex Box-QP with n variables, while the convex relaxation in the generalized α-BB algorithm introduces more variables and more linear constraints (the number of new variables and new constraints may be $\mathcal{O}(n^2)$ in the worst case, see [20] for more details). Moreover, our method further exploits the special structure of Box-QP to derive the convex relaxation Enhanced-DCR$_{[l,u]}$, which is as tight as the DNN relaxation at the root node.

4 Numerical experiments

In this section, we compare the proposed algorithms with the QUADPROGBB algorithm,[2] a previous state-of-the-art algorithm proposed by Chen and Burer [9]. The main feature of the QUADPROGBB algorithm is that it implements an augmented Lagrangian type method for computing doubly nonnegative relaxation based lower bounds, and calls the CPlex interfaces to compute upper bounds.

Our proposed algorithms are implemented in Matlab R2014. We use "quadprog" function to solve convex quadratic programming relaxations, SeDuMi [17] to solve semidefinite programming problems, and the iterative method described in Sect. 3 to solve the DNN relaxation problems. We run all of the three algorithms on a PC with Intel Core i7-2600 (3.40 GHz) and 4 GB memory. Our numerical experiments have been carried out on three test sets:

- *Basic Set* The Basic Set was generated by Vandenbussche and Nemhauser [23], which contains 54 Box-QP instances with dimensions ranging from 20 to 60. The Basic Set is a standard test set that has been widely used in the literature (Refs. [7,9,23]).
- *Extended Set* The Extended Set was generated by Burer and Vandenbussche [7], which contains 36 Box-QP instances with dimensions ranging form 70 to 100. The Extended Set has been used in [7,9].

[1] The idea of using an iterative procedure is motivated by the works in [5,15].

[2] Available at https://github.com/sburer/QUADPROGBB.

Table 1 Number of instances being solved to 10^{-4} relative error bound within 2 h

Test sets information			# Solved		
Name	# Instances	Dimensions	DC-BB	EBB	QUADPROGBB
Basic	54	20–60	54	54	54
Extended	36	70–100	18	31	32
Random	70	20–80	66	69	70

- *Random Set* This set contains 70 randomly generated Box-QP instances with dimensions ranging from 20 to 80. The entries of matrix Q and vector c are uniformly sampled from the interval $[-100, 100]$.

The first two sets are publicly available.[3] There are 160 instances in total in these three sets. For all algorithms, we use a relative optimality tolerance 10^{-4} for termination, i.e., if $|U^* - L^t|/|U^*| < 10^{-4}$, then the algorithm terminates. A running time limit of 2 h is enforced for all the algorithms. We regard instances as trivial if they can be solved by one of the three algorithms within 1 s, and regard instances as hard if they cannot be solved by one of the three algorithms within 1000 s. The information of test sets and the number of problems having been successfully solved by these algorithms are summarized in Table 1. Detailed comparison results will be presented in the following two subsections.

4.1 Comparison between DC-BB and EBB

To compare the efficiency of the DC-BB algorithm and the EBB algorithm, we present their log–log plots of the CPU times in Fig. 2. We observe that the EBB algorithm performs better than the DC-BB algorithm in terms of computational time on most non-trivial instances. Especially, the EBB algorithm runs faster than the DC-BB algorithm on all instances that can not be solved within 10 s by either one of the two algorithms. The results on non-trivial instances indicate that the Enhanced-DCR$_{[l,u]}$ relaxation provides a tighter relaxation bound than the DCR$_{[l,u]}$ relaxation, and reduces the number of enumerations effectively.

The main difference between the two algorithms is that the DC-BB algorithm solves problem SDR in preprocess, while the EBB alorithm solves problem DNN. To see the effects of the preprocessing steps more clear, we list information of the two relaxation methods in Fig. 3, where the results are based on the numerical results on the 20–80 dimensional instances in Random Set. Define

$$\text{Gap Closed} := \frac{v(\text{DNN}) - v^*}{v(\text{SDR}) - v^*},$$

where $v(\text{SDR})$, $v(\text{DNN})$ and v^* denote the SDR bound, the DNN bound, and the global optimal value (computed by the proposed branch-and-bound algorithm), respectively.

[3] Available at http://sburer.github.io/files/Box-QP.tar.gz.

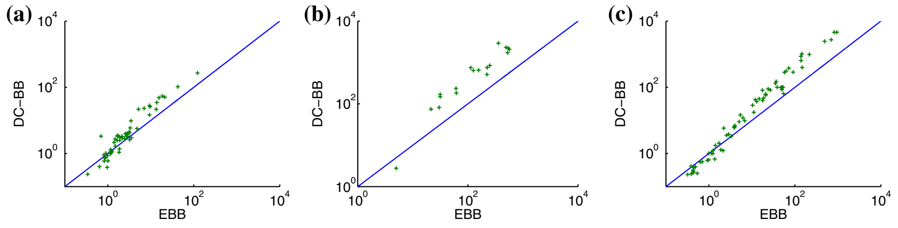

Fig. 2 Log–log plots of CPU times—DC-BB versus EBB. **a** Basic Set. **b** Extended Set. **c** Random Set

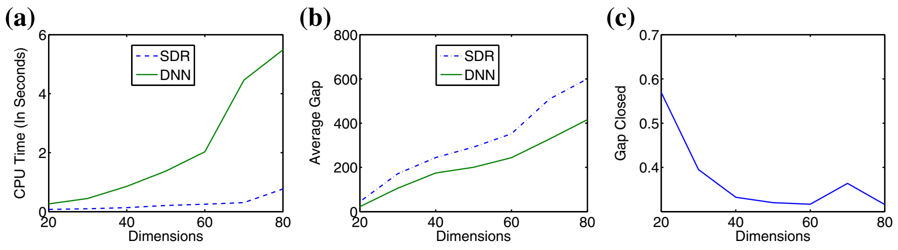

Fig. 3 Preprocessing time and average gap on root node—DC-BB versus EBB. **a** Average CPU time. **b** Average gap. **c** Gap closed

For each dimension n, the results are averaged over 10 instances. From Fig. 3, we discover that the DNN relaxation provides a tighter bound at the root node, but needs longer computational time. However, for all nontrivial test instances, the preprocessing time is relatively small in comparison with the total time being saved by the improved lower bound. Quite the reverse for the trivial instances, the CPU time for solving the DNN relaxation is even longer than the total time of DC-BB algorithm. Thus, EBB algorithm outperforms DC-BB algorithm on non-trivial instances, but underperforms on trivial instances.

Remark 2 The CPU time on solving the DNN relaxation is based on the iterative procedure described in Sect. 3. The computational time for solving the DNN relaxation by directly using SeDuMi is much longer. For example, it takes SeDuMi 32.26 s on average to solve a 60-dimensional DNN relaxation directly, while the iterative procedure only needs 2 s on average.

4.2 Comparison between EBB and QUADPROGBB

The log–log plots of the CPU times for the EBB algorithm versus the QUADPROGBB algorithm are presented in Fig. 4. The conclusions on the comparison results between the two algorithms are indefinite. For the Basic Set, the EBB algorithm performs better than the QUADPROGBB algorithm on most instances (53 out of 54 instances). For the Extended Set and Random Set, the QUADPROGBB algorithm outperforms the EBB algorithm on most hard instances. Meanwhile, the EBB algorithm performs faster than the QUADPROGBB algorithm on most non-hard instances. The above phenomenon can be explained as follows: the EBB algorithm solves a DNN relaxation at the root

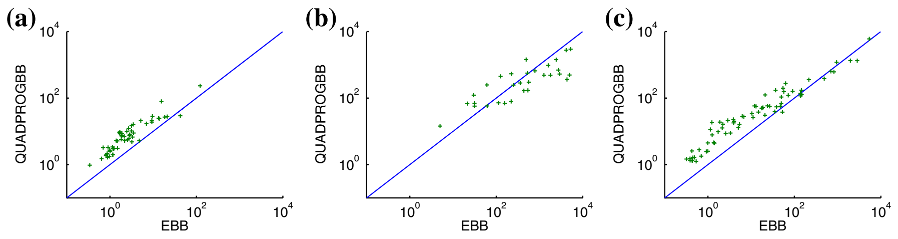

Fig. 4 Log–log plots of CPU times—QUADPROGBB versus EBB. **a** Basic Set. **b** Extended Set. **c** Random Set

node to compute the parameters λ^* and N^*, which are then fixed in later enumerations. Thus, the convex quadratic relaxation can be as tight as the DNN relaxation at the root node, and deteriorates as the enumeration goes into the deep layers of the branch-and-bound tree. Therefore, the EBB algorithm is expected to perform well for most non-hard instances, but not as well on hard instances.

4.3 Results on 20–60 dimensional instances

Based on the observations in Sects. 4.1 and 4.2, we discover that there is no single algorithm outperforms the others on all instances with different dimensions. However, the two proposed algorithms are competitive on 20–60 dimensional instances. This motivates us to implement a composed algorithm (named as CP-BB), which runs DC-BB for Box-QP instances with no more than 30 variables, and runs EBB for the instances with more than 30 variables. The CP-BB algorithm is tested against QUADPROGBB and BARON on all 20–60 dimensional instances in the three test sets. The total number of 20–60 dimensional instances is 104, including the 54 test instances in Basic Set. All of the 104 instances have been solved by the three algorithms to a 10^{-4} relative error within 1-h time limit. The log–log plots are presented in Fig. 5. We have two observations from this figure: (i) CP-BB performs better than QUADPROGBB on 101 out of 104 test instances; (ii) CP-BB performs more robust than BARON, in the sense that the CP-BB solves all the instances within 120 s, while BARON fails to solve 21 test instances within 120 instances and still fails to solve 10 instances within 1000 s.[4] These observations show that CP-BB is a competitive choice for solving 20–60 dimensional Box-QP problems.

5 Conclusions

In this paper, we propose two DC decomposition based branch-and-bound algorithms, DC-BB algorithm and EBB algorithm, to find global solutions of box-constrained quadratic programming problems. The DC-BB algorithm is developed from a DC

[4] The BARON solver is called via the interface provided on the NEOS Server [10]. However, we believe that the significant difference of the performances between CP-BB and BARON is mainly due to algorithmic design, rather than the hardware of the machines.

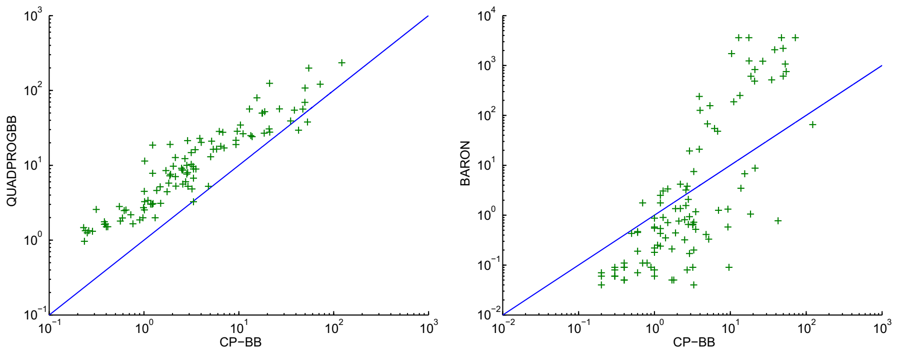

Fig. 5 Log–log plots of CPU times on 20–60 dimensional instances

decomposition based quadratic convex relaxation, whose parameters are obtained from the solution of problem SDR. The EBB algorithm is based on an enhanced convex relaxation derived from the dual solution of problem DNN. We compare the performances of the proposed algorithms with the state-of-the-art algorithms, including the QUADPROGBB algorithm proposed by Chen and Burer, and the commercial solver BARON. Numerical results show that the EBB algorithm outperforms QUADPROGBB algorithm on the Basic Set, and a composed algorithm CP-BB, which adaptively switches between the DC-BB algorithm and the EBB algorithm depending on the size of the problem, performs more efficient than the QUADPROGBB algorithm and more robust than BARON on 20–60 dimensional test instances.

An potential direction in our further research is to exploit more RLT inequalities, rather than only the nonnegativity inequalities $X \geqslant 0$, to derive convex quadratic programming relaxations that can be tighter than Enhanced-DCR$_{[l,u]}$. The remaining question is how to design efficient algorithms to solve the SDP+RLT relaxation. A specially designed augmented Lagrangian algorithm might be a suitable solution candidate.

References

1. An, L.T.H., Tao, P.D.: Solving a class of linearly constrained indefinite quadratic problems by D.C. algorithms. J. Global Optim. **11**, 253–285 (1997)
2. An, L.T.H., Tao, P.D.: A branch and bound method via d.c. optimization algorithms and ellipsoidal technique for box constrained nonconvex quadratic problems. J. Global Optim. **13**, 171–206 (1998)
3. Anstreicher, K.M.: Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. J. Global Optim. **43**, 471–484 (2009)
4. Billionnet, A., Elloumi, S.: Using a mixed integer quadratic programming solver for the unconstrained quadratic 0–1 Problem. Math. Program. **109**, 55–68 (2007)
5. Buchheim, C., Wiegele, A.: Semidefinite relaxations for non-convex quadratic mixed-integer programming. Math. Program. **141**, 435–452 (2013)
6. Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs. Math. Program. Comput. **2**, 1–19 (2010)
7. Burer, S., Vandenbussche, D.: A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. Math. Program. **113**, 259–282 (2008)

8. Cambini, R., Sodini, C.: Decomposition methods for solving nonconvex quadratic programs via branch and bound. J. Global Optim. **33**, 316–336 (2005)
9. Chen, J., Burer, S.: Globally solving nonconvex quadratic programming problems via completely positive programming. Math. Program. Comp. **4**, 33–52 (2012)
10. Czyzyk, J., Mesnier, M.P., Moré, J.J.: The NEOS server. IEEE J. Comput. Sci. Eng. **5**, 68–75 (1998)
11. Dong, H.: Relaxation nonconvex quadratic functions by multiple adaptive diagonal perturbations. SIAM J. Optim. **26**, 1962–1985 (2016)
12. Linderoth, J.: A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs. Math. Program. **103**, 251–282 (2005)
13. Lu, C., Guo, X.: Convex reformulation for binary quadratic programming problems via average objective value maximization. Optim. Lett. **9**, 523–535 (2015)
14. Pardalos, P.M., Vavasis, S.A.: Quadratic programming with one negative eigenvalue is NP-Hard. J. Global Optim. **1**, 15–22 (1991)
15. Qualizza, A., Belotti, P., Margot, F.: Linear programming relaxations of quadratically constrained quadratic programs. In: IMA Volume Series, pp. 407–426. Springer, Berlin (2012)
16. Saxena, A., Bonami, P., Lee, J.: Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. Math. Program. **130**, 359–413 (2010)
17. Sturm, J.F.: Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. Optim. Method Softw. **11**, 625–653 (1999)
18. Sherali, H.D., Tuncbilek, C.H.: A global optimization algorithm for polynomial programming problems using a reformulation-linearization technique. J. Global Optim. **2**, 101–112 (1992)
19. Sherali, H.D., Adams, W.P.: A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems. Kluwer, Dordrecht (1999)
20. Skjäl, A., Westerlund, T., Misener, R., Floudas, C.A.: A generalization of the classical α-BB convex underestimation via diagonal and nondiagonal quadratic terms. J. Optim. Theory Appl. **154**, 462–490 (2012)
21. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. (Ser. B) **103**, 225–249 (2005)
22. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Rev. **38**, 49–95 (1996)
23. Vandenbussche, D., Nemhauser, G.L.: A branch-and-cut algorithm for nonconvex quadratic programs with box constraints. Math. Program. **102**, 559–575 (2005)
24. Zheng, X., Sun, X., Li, D.: Nonconvex quadratically constrained quadratic programming: best D.C. decompositions and their SDP representations. J. Global Optim. **50**, 695–712 (2011)