Research Article

Zhisong Hou*, Hongwei Jiao*, Lei Cai, and Chunyang Bai

Branch-delete-bound algorithm for globally solving quadratically constrained quadratic programs

https://doi.org/10.1515/math-2017-0099
Received January 21, 2017; accepted July 25, 2017.

Abstract: This paper presents a branch-delete-bound algorithm for effectively solving the global minimum of quadratically constrained quadratic programs problem, which may be nonconvex. By utilizing the characteristics of quadratic function, we construct a new linearizing method, so that the quadratically constrained quadratic programs problem can be converted into a linear relaxed programs problem. Moreover, the established linear relaxed programs problem is embedded within a branch-and-bound framework without introducing any new variables and constrained functions, which can be easily solved by any effective linear programs algorithms. By subsequently solving a series of linear relaxed programs problems, the proposed algorithm can converge the global minimum of the initial quadratically constrained quadratic programs problem. Compared with the known methods, numerical results demonstrate that the proposed method has higher computational efficiency.

Keywords: Quadratically constrained quadratic programs, Global optimization; Linearizing method, Deleting technique, Branch-delete-bound algorithm

MSC: 90C20, 90C26, 65K05

1 Introduction

The quadratically constrained quadratic programs problem (QCQP) has attracted a huge attention of practitioners and researchers for many years. In part, this is because the quadratically constrained quadratic programs problem finds a wide range of applications in management science and engineering, product subassembly, production programs, portfolio decision optimization, chance problem, production design, finance and economy, etc. (see [1-7]). In particular, many practical problems (such as stochastic programs problem, packing problem, 0-1 programs problem, etc. [8,9]) can be transformed into the quadratically constrained quadratic programs problem. In addition, the problem (QCQP) possess multiple local optimum points which are not globally optimum, i.e., from a research point of view, this problem (QCQP) poses significant theoretical and computational complication.

In this paper, the mathematical modelling of the investigated quadratically constrained quadratic programs problem is given as follows:
The main operation in the proposed branch-delete-bound algorithm is computation of the lower bounds of the initial problem and its partitioned subproblems. The lower bounds for the initial problem and its partitioned subproblems can be computed by solving their corresponding linear relaxed programs problems, which are derived by the linearizing method. To this aim, by utilizing the characteristics of quadratic function, we first introduce a new linearizing method, then by using the linearizing method, we show that the initial problem (QCQP) and its sub-problems can be systematically converted into the corresponding linear relaxed programs problems, and the optimal solutions of these converted linear relaxed programs problems can infinitely approximate the global optimum of the associated quadratically constrained quadratic programs problem. Moreover, the established linear relaxed programs problems are embedded within a branch-and-bound framework without introducing any new variables and constrained functions, which can be easily solved by any effective linear programming algorithms, and which are easier to be solved than any convex relaxation programs problem. We then combine the established linear relaxed programs problem, the branching operation, deleting operation and bounding operation together, so an effective branch-delete-bound algorithm is presented for globally solving the QCQP. Finally, compared with some known algorithms, numerical experimental results show that our method has higher computational efficiency.

The goal of this article is to present a branch-delete-bound algorithm for globally solving the QCQP problem by developing new linearizing technique. To this aim, by utilizing the characteristics of quadratic function, we first introduce a new linearizing method, then by using the linearizing method, we show that the initial problem (QCQP) and its sub-problems can be systematically converted into the corresponding linear relaxed programs problems, and the optimal solutions of these converted linear relaxed programs problems can infinitely approximate the global optimum of the associated quadratically constrained quadratic programs problem. Moreover, the established linear relaxed programs problems are embedded within a branch-and-bound framework without introducing any new variables and constrained functions, which can be easily solved by any effective linear programming algorithms, and which are easier to be solved than any convex relaxation programs problem. We then combine the established linear relaxed programs problem, the branching operation, deleting operation and bounding operation together, so an effective branch-delete-bound algorithm is presented for globally solving the QCQP. Finally, compared with some known algorithms, numerical experimental results show that our method has higher computational efficiency.

The rest of this paper is organized as follows. Based on the characteristics of quadratic function, Section 2 formulates a new linearizing method, and the linear relaxed programs problems of the initial problem (QCQP) and its sub-problems are established. Based on the linear relaxed programs problem derived in Section 2, Section 3 presents a branch-delete-bound algorithm, and its global convergence is discussed and proved. In Section 4, compared with some known algorithms, numerical results demonstrate the computational efficiency of the proposed algorithm. Finally, some concluding remarks are drawn.

2 New linearizing method

The main operation in the proposed branch-delete-bound algorithm is computation of the lower bounds of the initial problem and its partitioned subproblems. The lower bounds for the initial problem and its partitioned subproblems can be computed by solving their corresponding linear relaxed programs problems, which are derived by the following new linearizing method.

Let \( Z = \{ z \in R^n | l \leq z \leq u \} \subseteq Z^0 \). For \( \forall z \in Z \), for any \( j, k \in \{ 1, 2, \ldots, n \} \), define

\[
\begin{align*}
    f_j(z) &= z_j^2, \\
    f_j^I(z) &= (l_j + u_j)z_j - \frac{(l_j + u_j)^2}{4}, \\
    f_j^R(z) &= (l_j + u_j)z_j - l_ju_j, \\
    \Delta_j(z) &= f_j(z) - f_j^I(z), \\
    \nabla_j(z) &= f_j^R(z) - f_j(z), \\
    f_{jk}(z) &= z_jz_k, \\
    f_{jk}^I(z) &= \frac{1}{2}(l_j + u_j)z_k + (l_k + u_k)z_j - \frac{(l_j + u_j)^2}{4} - \frac{(l_k + u_k)^2}{4} + (l_j - u_k)(u_j - l_k),
\end{align*}
\]
By (1), (2) and (3), we have
\[
\Delta_{jk}(z) = f_{jk}(z) - f'_{jk}(z),
\]
\[
\nabla_{jk}(z) = f''_{jk}(z) - f_{jk}(z),
\]
\[
\Delta(z_j - z_k) = (z_j - z_k)^2 - [(l_j - u_k + u_j - l_k)(z_j - z_k) - \frac{(l_j - u_k + u_j - l_k)^2}{4}] ,
\]
\[
\nabla(z_j - z_k) = [(l_j - u_k + u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k)] - (z_j - z_k)^2.
\]

**Theorem 2.1.** For any \( z \in Z = [l, u] \subseteq Z^0 \), for any \( j, k \in \{1, 2, \ldots, n\} \), we have the following conclusions:
(i) \( f_j^l(z) \leq f_j(z) \leq f_j^u(z) \);
(ii) \( f_j^l(z) \leq f_{jk}(z) \leq f_j^u(z) \), \( j \neq k \);
(iii) \( \Delta_j(z) \to 0, \nabla_j(z) \to 0, \Delta(z_j - z_k) \to 0, \nabla(z_j - z_k) \to 0 \) and \( \nabla_{jk}(z) \to 0 \) as \( \|u - l\| \to 0 \).

**Proof.** (i) By the convex characters of the quadratic function \( f_j(z) = z_j^2 \), we have
\[
f_j^l(z) = (l_j + u_j)z_j - \frac{(l_j + u_j)^2}{4} \leq z_j^2 \leq (l_j + u_j)z_j - l_j u_j = f_j^u(z),
\]
i.e.,
\[
f_j^l(z) \leq f_j(z) \leq f_j^u(z).
\]
(ii) Since \( (z_j - z_k)^2 \) is a convex function about \( (z_j - z_k) \) over the interval \([l_j - u_k, u_j - l_k]\), similarly by the conclusion (i), we have
\[
(l_j - u_k + u_j - l_k)(z_j - z_k) - \frac{(l_j - u_k + u_j - l_k)^2}{4} \leq (z_j - z_k)^2
\]
and
\[
(l_j - u_k + u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k) \geq (z_j - z_k)^2.
\]
By (1), (2) and (3), we have
\[
f_{jk}(z) = \frac{1}{2}[z_j^2 + z_k^2 - (z_j - z_k)^2]
\]
\[
= \frac{1}{2}[(l_j + u_j)z_j - \frac{(l_j + u_j)^2}{4} + (l_k + u_k)z_k - \frac{(l_k + u_k)^2}{4}]
\]- \frac{1}{2}[(l_j - u_k + u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k)]
\]
\[
= \frac{1}{2}[(l_j + u_j)z_k + (l_k + u_k)z_j - \frac{(l_j + u_j)^2}{4} - \frac{(l_k + u_k)^2}{4} + (l_j - u_k)(u_j - l_k)]
\]
\[
f_{jk}^l(z) = \frac{1}{2}[z_j^2 + z_k^2 - (z_j - z_k)^2]
\]
\[
\leq \frac{1}{2}[(l_j + u_j)z_j - \frac{(l_j + u_j)^2}{4} + (l_k + u_k)z_k - \frac{(l_k + u_k)^2}{4}]
\]- \frac{1}{2}[(l_j - u_k + u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k)]
\]
\[
= \frac{1}{2}[(l_j + u_j)z_k + (l_k + u_k)z_j - \frac{(l_j + u_j)^2}{4} - \frac{(l_k + u_k)^2}{4} + (l_j - u_k)(u_j - l_k)]
\]
\[
f_{jk}^u(z),
\]
i.e.,
\[
f_{jk}^l(z) \leq f_{jk}(z) \leq f_{jk}^u(z).
\]
(iii) Since
\[
\Delta_j(z) = f_j(z_j) - f_j^l(z_j) = z_j^2 - (l_j + u_j)z_j - \frac{(l_j + u_j)^2}{4}
\]
is a convex function about \( z_j \) over the interval \([l_j, u_j]\). Thus, \( \Delta_j(z) \) can obtain the maximum value at the point \( l_j \) or \( u_j \), i.e.,
\[
\max_{z_j \in [l_j, u_j]} \Delta_j(z) = \frac{(u_j - l_j)^2}{4}.
\]
Similarly, since
\[ \nabla_j(z) = f_j^2(z_j) - f_j(z_j) = (l_j + u_j)z_j - l_ju_j - z_j^2 \]
is a concave function about \( z_j \) over the interval \([l_j, u_j]\), therefore \( \nabla_j(z) \) can obtain maximum value at the point \( l_j + u_j \), i.e.,
\[ \max_{z_j \in [l_j, u_j]} \nabla_j(z) = \frac{(u_j - l_j)^2}{4}. \tag{5} \]
By (4) and (5), we have
\[ \max_{z_j \in [l_j, u_j]} \Delta_j(z) = \max_{z_j \in [l_j, u_j]} \nabla_j(z) \to 0, \text{ as } \|u - l\| \to 0. \tag{6} \]
Since
\[ \Delta(z_j - z_k) = (z_j - z_k)^2 - [(l_j - u_k + u_j - l_k)(z_j - z_k) - \frac{(l_j - u_k + u_j - l_k)^2}{4}] \]
is a convex function about \( z_j - z_k \) over the interval \([l_j - u_k, u_j - l_k]\), thus, \( \Delta(z_j - z_k) \) can obtain the maximum value at the point \( l_j - u_k \) or \( u_j - l_k \), i.e.,
\[ \max_{(z_j - z_k) \in [(l_j - u_k), (u_j - l_k)]} \Delta(z_j - z_k) = \frac{(u_j - l_k - l_j + u_k)^2}{4}. \tag{7} \]
Similarly, since
\[ \nabla(z_j - z_k) = [(l_j - u_k + u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k)] \]
is a concave function about \( z_j - z_k \) over the interval \([l_j - u_k, (u_j - l_k)]\), therefore \( \nabla(z_j - z_k) \) can obtain maximum value at the point \( \frac{l_j - u_k + u_j - l_k}{2} \), i.e.,
\[ \max_{(z_j - z_k) \in [(l_j - u_k), (u_j - l_k)]} \nabla(z_j - z_k) = \frac{(u_j - l_k - l_j + u_k)^2}{4}. \tag{8} \]
By (7) and (8), we have: as \( \|u - l\| \to 0, \)
\[ \max_{(z_j - z_k) \in [(l_j - u_k), (u_j - l_k)]} \Delta(z_j - z_k) = \max_{(z_j - z_k) \in [(l_j - u_k), (u_j - l_k)]} \nabla(z_j - z_k) \to 0. \tag{9} \]
Since
\[ \Delta_{jk}(z) = f_{jk}(z) - f_{jk}^u(z) \]
\[ = z_jz_k - \frac{1}{2}[(l_j + u_j + u_k)z_j + (l_j + u_k)z_k - \frac{(l_j + u_j)^2}{4} - \frac{(l_j + u_k)^2}{4} + (l_j - u_k)(u_j - l_k)] \]
\[ = \frac{1}{2}[z_j^2 + z_k^2 - (z_j - z_k)^2] - \frac{1}{2}[(l_j + u_j + u_k)z_j + (l_j + u_k)z_k - \frac{(l_j + u_j)^2}{4} + \frac{(l_j + u_k)^2}{4} + (l_j - u_k)(u_j - l_k)] \]
\[ = \frac{1}{2}[z_j^2 + z_k^2 - (z_j - z_k)^2] - \frac{1}{2}[(l_j + u_j + u_k)z_j - \frac{(l_j + u_j)^2}{4} + (l_j - u_k)(u_j - l_k)] \]
\[ + \frac{1}{2}[(l_j - u_k)(u_j - l_k)(z_j - z_k) - (l_j - u_k)(u_j - l_k)] \]
\[ = \frac{1}{2}\Delta_j(z) + \frac{1}{2}\Delta_k(z) + \frac{1}{2}\nabla(z_j - z_k) \]
\[ \leq \frac{1}{2}\max_{z_j \in [l_j, u_j]} \Delta_j(z) + \frac{1}{2}\max_{z_k \in [l_k, u_k]} \Delta_k(z) \]
\[ + \frac{1}{2}\max_{(z_j - z_k) \in [(l_j - u_k), (u_j - l_k)]} \nabla(z_j - z_k). \]
By (6) and (9), we have \( \Delta_{jk}(z) \to 0 \) as \( \|u - l\| \to 0. \)
Similarly, we can prove that \( \nabla_{jk}(z) \to 0 \) as \( \|u - l\| \to 0. \) Therefore, the proof is complete. \( \square \)

For convenience, without loss of generality, for any \( j \) and \( k \in \{1, \ldots, n\} \), we let
\[ g_{jk}^{i,j}(z) = \begin{cases} a_{jk}^l f_{jk}^l(z), & \text{if } a_{jk}^l \geq 0, \\ a_{jk}^u f_{jk}^u(z), & \text{if } a_{jk}^u < 0. \end{cases} \]
\[ g^i_{jk}(z) = \begin{cases} a^i_{jk} f^j_{jk}(z), & \text{if } a^i_{jk} \geq 0, \ j \neq k, \\ a^i_{jk} f^j_{jk}(z), & \text{if } a^i_{jk} < 0, \ j \neq k; \end{cases} \]

\[ g^i_{k}(z) = \begin{cases} a^i_{kk} f^j_{jk}(z), & \text{if } a^i_{kk} < 0, \\ a^i_{kk} f^j_{jk}(z), & \text{if } a^i_{kk} \geq 0; \end{cases} \]

\[ g^i_{jk}(z) = \begin{cases} a^i_{jk} f^j_{jk}(z), & \text{if } a^i_{jk} < 0, \ j \neq k. \\ a^i_{jk} f^j_{jk}(z), & \text{if } a^i_{jk} \geq 0; \end{cases} \]

Obviously, we have

\[ a^i_{kk} z_k^2 \geq g^i_{k}(z), \ a^i_{jk} z_j z_k \geq g^i_{jk}(z), \ a^i_{kk} z_k^2 \leq g^i_{k}(z), \ a^i_{jk} z_j z_k \leq g^i_{jk}(z). \tag{10} \]

And for each \( i = 1, \ldots, p, m = 1, \ldots, M, \) for any \( z \in Z, \) define

\[
G^L_i(z) = \sum_{k=1}^{n} (d^i_{k} z_k + g^i_{k}(z)) + \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} g^j_{jk}(z), \\
G^U_i(z) = \sum_{k=1}^{n} (d^i_{k} z_k + \overline{g}^i_{k}(z)) + \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} \overline{g}^j_{jk}(z).
\]

**Theorem 2.2.** For all \( z \in Z = [l, u] \subseteq \mathbb{Z}^0, \) for each \( i = 1, \ldots, p, m = 1, \ldots, M, \) we have the following conclusions:

(i) \( G^U_i(z) \geq G_i(z) \geq G^L_i(z); \)

(ii) \( G_i(z) - G^L_i(z) \to 0 \) and \( G^U_i(z) - G_i(z) \to 0, \) as \( \|u - l\| \to 0. \)

**Proof.** (i) Obviously, from (10) we can easily get that \( G^U_i(z) \geq G_i(z) \geq G^L_i(z) \) holds.

(ii) Considering the error \( G_i(z) - G^L_i(z), \) we have

\[
G_i(z) - G^L_i(z) = \sum_{k=1}^{n} d^i_{k} z_k + \sum_{k=1}^{n} a^i_{kk} z_k^2 + \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} a^i_{jk} z_j z_k \\
- \left[ \sum_{k=1}^{n} (d^i_{k} z_k + g^i_{k}(z)) + \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} g^j_{jk}(z) \right] \\
= \sum_{k=1}^{n} a^i_{kk} z_k^2 - g^i_{k}(z) + \sum_{j=1}^{n} \sum_{k=1, k \neq j}^{n} [a^i_{jk} z_j z_k - g^j_{jk}(z)] \\
= \sum_{k=1, a^i_{kk} > 0}^{n} a^i_{kk} [z_k^2 - f^j_k(z)] + \sum_{k=1, a^i_{kk} < 0}^{n} a^i_{kk} [z_k^2 - f^j_k(z)] \\
+ \sum_{j=1, k \neq j, a^i_{jk} > 0}^{n} a^i_{jk} [z_j z_k - f^j_k(z)] \\
+ \sum_{j=1, k \neq j, a^i_{jk} < 0}^{n} a^i_{jk} [z_j z_k - f^j_k(z)] \\
= \sum_{k=1, a^i_{kk} > 0}^{n} a^i_{kk} \Delta_k(z) - \sum_{k=1, a^i_{kk} < 0}^{n} a^i_{kk} \hat{\nabla}_k(z) \\
+ \sum_{j=1, k \neq j, a^i_{jk} > 0}^{n} a^i_{jk} \Delta_{jk}(z) - \sum_{j=1, k \neq j, a^i_{jk} < 0}^{n} a^i_{jk} \hat{\nabla}_{jk}(z) \\
\]

By the conclusions of Theorem 2.1, we have \( \Delta_j(z), \hat{\nabla}_j(z), \Delta_{jk}(z) \) and \( \hat{\nabla}_{jk}(z) \to 0, \) as \( \|u - l\| \to 0. \)

Therefore, \( G_i(z) - G^L_i(z) \to 0 \) as \( \|u - l\| \to 0. \)

Using the similar method as in the above proof, we can conclude that

\[ G^U_i(z) - G_i(z) \to 0 \] as \( \|u - l\| \to 0. \)

The proof of the conclusion (ii) is complete. \( \square \)
By Theorem 2.2, we can construct the corresponding approximation linear relaxed programs problem (LRPP) of the QCQP in $Z$ as follows.

$$\text{LRPP}(Z) : \begin{cases} \min G_L^T(z), \\
\text{s.t. } G_L^T(z) \leq b_i, \ i = 1, \ldots, m, \\
z \in Z = \{ z : l \leq z \leq u \} \subseteq Z^0, 
\end{cases}$$

where

$$G_L^T(z) = \sum_{k=1}^{n} (d_{ik} z_k + g_{ik}^T(z)) + \sum_{j=1}^{n} \sum_{k=1,k \neq j}^{n} g_{jk}^j(z).$$

From the constructing method of the above linear relaxed programs, for any $Z \subseteq Z^0$, every feasible point of the QCQP in sub-rectangle $Z$ is also feasible to the LRPP in sub-rectangle $Z$; and the optimum value of the LRPP in sub-rectangle $Z$ is less than or equal to that of the QCQP in sub-rectangle $Z$. Thus, the LRPP in sub-rectangle $Z$ provides a valid lower bound for the global minimum of the QCQP in sub-rectangle $Z$.

3 Branch-delete-bound algorithm

In this section, based on the linear relaxed programs problem derived by new linearizing method in Section 2, we will present an effective branch-delete-bound algorithm for globally solving the QCQP. In this algorithm, there are three fundamental operations: branching operation, deleting operation and bounding operation. We then introduce this three fundamental operations as follows.

3.1 Branching operation

Here, we select a standard branching operation, which is called as bisection method of rectangle maximum edge. The selected branching operation iteratively subdivides the investigated rectangle $Z$ into two sub-rectangles $Z_{k,1}$ and $Z_{k,2}$, it generates a more refined partition that cannot yet be excluded from further consideration in finding the global minimum of the QCQP in $Z^0$. This selected branching operation is enough to ensure the global convergence of the proposed algorithm since the interval of each variable is shrank into a singleton through infinite rectangle bisection. For any identified sub-rectangle $Z^k = [l^k, u^k] \subseteq Z^0$. This branching operation is formulated as follows.

(a) Let $\xi = \max \{ u_i - l_i : i = 1, \ldots, n \}$.

(b) Let

$$Z_{k,1} = \{ z \in \mathbb{R}^n | l_i \leq z_i \leq u_i, i \neq \xi; l_\xi \leq z_\xi \leq \frac{l_\xi + u_\xi}{2} \},$$

$$Z_{k,2} = \{ z \in \mathbb{R}^n | l_i \leq z_i \leq u_i, i \neq \xi; \frac{l_\xi + u_\xi}{2} \leq z_\xi \leq u_\xi \}.$$

So that the identified sub-rectangle $Z$ is divided into two sub-rectangles $Z_{k,1}$ and $Z_{k,2}$.

3.2 Deleting operation

Based upon the linear relaxed programs in section 2 and branch-and-bound structure, we will introduce a deleting operation to improve the convergent speed of the proposed algorithm, which is used to delete a part of the rectangle $Z$ or the whole rectangle $Z$ without rejecting any global optimal solution of the initial problem (QCQP) in $Z^0$. For convenience, for any $z \in Z = (Z_j)_{n \times 1}$ with $Z_j = [l_j, u_j]$ ($j = 1, \ldots, n$), without loss of generality, we rewrite the LRPP into the following linear programming problem in sub-rectangle $Z$:

$$\text{LP}(Z) : \begin{cases} \min \sum_{j=1}^{n} \gamma_{0j} z_j + \eta_0 \\
\text{s.t. } \sum_{j=1}^{n} \gamma_{ij} z_j + \eta_i \leq b_i, \ i = 1, \ldots, m, \\
z \in Z = \{ z : l \leq z \leq u \}. 
\end{cases}$$
Let \( UB_k \) be a currently known upper bound of the global optimal value for the QCQP in \( Z^0 \), which is obtained after \( k \) iterations, and set

\[
LB_j = \sum_{j=1}^{n} \min\{\gamma_{ij} l_j, \gamma_{ij} u_j\} + \eta_i, \ i = 0, 1, \ldots, m;
\]

\[
\beta_p = \frac{UB_k - LB_0}{Y_{ip}} + \min\{\gamma_{0p}, \gamma_{0p} u_p\}, \ p = 1, \ldots, n;
\]

\[
\lambda_i = \frac{b_i - LB_0 + \min\{\gamma_{ip}, \gamma_{ip} u_p\}}{Y_{ip}}, \ p = 1, \ldots, n, i = 1, \ldots, m;
\]

\[
\overline{Z}_j = \begin{cases} Z_j, j \neq p, j = 1, \ldots, n, \\ (\beta_p, u_p) \cap Z_p, j = p; \end{cases}
\]

\[
Z_j = \begin{cases} Z_j, j \neq p, j = 1, \ldots, n, \\ [l_p, \beta_p) \cap Z_p, j = p; \end{cases}
\]

\[
\overline{Z} = \begin{cases} Z_j, j \neq p, j = 1, \ldots, n, \\ (\lambda_{ip}, u_p) \cap Z_p, j = p; \end{cases}
\]

\[
\hat{Z}_j = \begin{cases} Z_j, j \neq p, j = 1, \ldots, n, \\ [l_p, \lambda_i) \cap Z_p, j = p. \end{cases}
\]

Similarly as in Theorem 3 in [23], for any sub-rectangle \( Z \subseteq Z^0 \), we can easily prove that the following conclusions hold:

(i) If \( LB_0 > UB_k \), then the sub-rectangle \( Z \) can be deleted; else if \( LB_0 \leq UB_k \), then: for each \( p \in \{1, 2, \ldots, n\} \), if \( \gamma_{0p} > 0 \), then the sub-rectangle \( \overline{Z} = (\overline{Z}_j)_{n \times 1} \) can be deleted; if \( \gamma_{0p} < 0 \), then the sub-rectangle \( Z = (Z_j)_{n \times 1} \) can be deleted.

(ii) If \( LB_j > b_i \) for some \( i \in \{1, \ldots, m\} \), then the sub-rectangle \( Z \) can be deleted; else if \( LB_j \leq b_i \) for some \( i \in \{1, \ldots, m\} \), then: for each \( p \in \{1, 2, \ldots, n\} \), if \( \gamma_{ip} > 0 \), then the sub-rectangle \( \overline{Z} = (\overline{Z}_j)_{n \times 1} \) can be deleted; if \( \gamma_{ip} < 0 \), then the sub-rectangle \( Z = (\hat{Z}_j)_{n \times 1} \) can be deleted.

By utilizing the deleting operation to delete a part of the investigated rectangle where the global optimal solution of the QCQP in \( Z^0 \) does not exist, we can improve the computational speed of the proposed branch-and-bound procedure, and accelerate the global convergence of the proposed branch-and-bound algorithm.

### 3.3 Bounding operation

The bounding operations are used to update the lower bounds and upper bounds of the global optimal value of the QCQP in \( Z^0 \). This main computations for updating lower bounds need to solve a sequence of linear relaxed programs problems, which can be easily solved by using simplex methods. In additions, the upper bounds can be updated by computing the objective function value of the QCQP, which is corresponding to the optimal solution of each linear relaxed programs problem or midpoint of the investigated rectangle \( Z^k \), respectively.

### 3.4 Branch-delete-bound algorithm

Let \( LB(Z^k) \) and \( z^k = z(Z^k) \) be the optimum value and optimum solution for the LRPP in the sub-rectangle \( Z^k \), respectively. Combining the former linear relaxed programs, the branching operation, deleting operation and bounding operation together, we can establish an effective branch-delete-bound algorithm for globally solving the problem (QCQP) as follows.

#### Branch-Delete-Bound Algorithm:

**Initializing Step.** Initializing the counter of iteration \( k := 0 \), the active node set \( \Lambda_0 = \{Z^0\} \), the feasible point set \( F = \emptyset \), the convergence judgement error \( \epsilon > 0 \), the initial upper bound \( UB_0 = +\infty \). Compute the LRPP(\( Z^0 \)), obtain \( LB_0 := LB(Z^0) \) and \( z^0 := z(Z^0) \). If \( G_i(z^0) \leq b_i \) holds for all \( i = 1, \ldots, m \), then we update the feasible point set \( F = \{z^0\} \) and the upper bound \( UB_0 = G_0(z^0) \). If \( UB_0 - LB_0 \leq \epsilon \) holds, then the algorithm stops with \( z^0 \) as the global \( \epsilon \)-optimal solution of the QCQP in \( Z^0 \); else go on the following Branching Step.
Branching Step. Select a rectangle $Z^k \in \Lambda_k$ to determine a branching variable $z_q$, and employ the selected branching operation to divide the selected rectangle $Z^k$ into two new sub-rectangles, and represent the new subdivided sub-rectangles set as $\hat{Z}^k$.

Deleting Step. For any sub-rectangle $Z \in \hat{Z}^k$, compute $\underline{L}B_i (i = 0, 1, \ldots, m), \beta_p (p = 1, \ldots, n), \lambda_{ip} (i = 1, \ldots, m, p = 1, \ldots, n)$.

For each $i \in \{1, \ldots, m\}$, if $\underline{L}B_i > b_i$, then delete the investigated sub-rectangle $Z$; else if $\gamma_{1p} > 0$ and $\lambda_{ip} < u_p$ for some $p \in \{1, \ldots, n\}$, then let $u_p = \lambda_{ip}$; else if $\gamma_{1p} < 0$ and $\lambda_{ip} > l_p$ for some $p \in \{1, \ldots, n\}$, then let $l_p = \lambda_{ip}$.

If $\underline{L}B_0 > UB_k$, then delete the investigated sub-rectangle $Z$; else if $\gamma_{0p} > 0$ and $\beta_p < u_p$ for some $p \in \{1, \ldots, n\}$, then let $u_p = \beta_p$; else if $\gamma_{0p} < 0$ and $\beta_p > l_p$ for some $p \in \{1, \ldots, n\}$, then let $l_p = \beta_p$.

Finally, still denote the remaining sub-rectangle by $Z$, and denote the remaining partition sub-rectangle set by $\hat{Z}^k$.

Bounding Step. Solve the LRPP($Z$) for each sub-rectangle $Z \in \hat{Z}^k$ to get $LB(Z)$ and $z(Z)$, if $LB(Z) > UB_k$, then set $\hat{Z}^k := \hat{Z}^k \backslash Z$; else if the midpoint $z_{mid}$ of $Z$ satisfies constrained condition for the QCP in $Z^0$, then set $F := F \cup \{z_{mid}\}$, and if $z(Z)$ satisfies constrained condition for the QCP in $Z^0$, then let $F := F \cup \{z(Z)\}$, at the same time, we update the upper bound $UB_k := \min_{z \in F} G_0(z)$. If $F \neq \emptyset$, denote $z^k := \arg \min_{z \in F} G_0(z)$ as the current best feasible point. Let $\Lambda_k := (\Lambda_k \backslash Z) \cup \hat{Z}^k$, we then update the lower bound $LB_k := \inf_{Z \in \Lambda_k} LB(Z)$.

Optimality Judgement Step. If $UB_k - LB_k \leq \epsilon$, then the algorithm stops, at the same time, we get that $UB_k$ and $z^k$ are the global $\epsilon$–optimal value and the global $\epsilon$–optimal solution for the initial problem (QCQP), respectively; else let $k := k + 1$, and select a new active node $Z^k$ satisfying $Z^k = \arg \min_{Z \in \Lambda_k} LB(Z)$, and return to Branching Step.

3.5 Global convergence analysis

The global convergence of the proposed branch-delete-bound algorithm is formulated as follows.

Theorem 3.1. If the proposed branch-delete-bound algorithm terminates after $k$ iterations, then $z^k$ is a local $\epsilon$–optimal solution for the (QCQP); else if the branch-delete-bound algorithm does not finitely terminates after $k$ iteration, then it must generate an infinite subsequence $(z^{k_i})$ of iterations, which satisfies that its any accumulation point must be the global optimum solution of the QCQP.

Proof. If the proposed branch-delete-bound algorithm finitely terminates after $k$ iterations, where $k \geq 0$, then by optimality judgement step, we have $UB_k - LB_k \leq \epsilon$. By the bounding operation for the upper bound, this implies that there must exist a feasible point $z^k$ satisfying $UB_k = G_0(z^k)$, thus we can follow that $G_0(z^k) - LB_k \leq \epsilon$, i.e. $G_0(z^k) - \epsilon \leq LB_k$. Denote $v^*$ as the optimal value of the QCQP, obviously, by the structure of branch-and-bound framework, it follows that $LB_k \leq v$. Since $z^k$ is feasible to the QCQP, therefore, it follows that $G_0(z^k) \geq v$, i.e. $G_0(z^k) - \epsilon \geq v - \epsilon$. Combining the above inequalities, it follows that $v \geq LB_k \geq G_0(z^k) - \epsilon \geq v - \epsilon$, i.e. $v \leq G_0(z^k) \leq v + \epsilon$. Therefore, $z^k$ is a global $\epsilon$–optimum solution for the QCQP.

If the proposed algorithm does not finitely terminates after $k$ iterations, a sufficient condition for the branch-delete-bound algorithm that is convergent to the global minimum is that the bounding operation must be consistent and the selection operation must satisfy that bound can be improved.

By the proposed branch-delete-bound algorithm, the employed branching operation is bisection, which satisfies the exhaustiveness, that is to say that any unfathomed partition can be further refined by the branching operation. Therefore, by Theorem 2.2 and the relationship between the QCP and its linear relaxed programs problem (LRPP), it is so easy to conclude that $\lim_{k \to \infty} (UB_k - LB_k) = 0$ holds, this implies that the employed bounding operation is consistent.
By the proposed branching operation, the selected sub-rectangle $Z^k$, which actually attained lower bound, is immediately selected for further partition in the later iteration. So that the selecting operation of the branch-delete-bound algorithm must satisfy that bound can be improved.

In general, it follows that the bounding operation is consistent and selection operation satisfy that bound can be improved. Finally, by Refs.[1,3], we can follow that the proposed branch-delete-bound algorithm converges to the global minimum of the initial problem (QCQP).

4 Numerical experiments

To compare the proposed branch-delete-bound algorithm with the known algorithms in computational speed and solution quality, some numerical examples in recent literature are solved on microcomputer. The solving procedure is coded in C++ software, and each linear relaxed programs problem in the solving procedure is solved by using simplex method. These test examples are listed as follows, and compared with the known methods. Numerical results are given in Tables 1-3. In Table 1 the number of algorithm iteration is denoted by “Iter.”.

Example 4.1 ([15,23]).

\[
\begin{align*}
\min & \quad -z_1^2 + z_1z_2 + z_2^2 + z_1 - 2z_2 \\
\text{s.t.} & \quad z_1 + z_2 \leq 6, \quad -2z_1^2 + z_2^2 + 2z_1 + z_2 \leq -4, \\
& \quad 1 \leq z_1 \leq 6, \quad 1 \leq z_2 \leq 6.
\end{align*}
\]

Example 4.2 ([17,19,23]).

\[
\begin{align*}
\min & \quad z_1^2 + z_2^2 \\
\text{s.t.} & \quad 0.3z_1z_2 \geq 1, \quad 2 \leq z_1, z_2 \leq 5.
\end{align*}
\]

Example 4.3 ([17,19,23,24]).

\[
\begin{align*}
\min & \quad z_1 \\
\text{s.t.} & \quad 4z_2 - 4z_1^2 \leq 1, \quad -z_1 - z_2 \leq -1, \\
& \quad 0.01 \leq z_1, z_2 \leq 15.
\end{align*}
\]

Example 4.4 ([22,23]).

\[
\begin{align*}
\min & \quad 6z_1^2 + 4z_2^2 + 5z_1z_2 \\
\text{s.t.} & \quad -6z_1z_2 \leq -48, \\
& \quad 0 \leq z_1, z_2 \leq 10.
\end{align*}
\]

Example 4.5 ([23,25]).

\[
\begin{align*}
\min & \quad -z_1 + z_1z_2^{0.5} - z_2 \\
\text{s.t.} & \quad -6z_1 + 8z_2 \leq 3, \\
& \quad 3z_1 - z_2 \leq 3, \\
& \quad 1 \leq z_1 \leq 1.5, \quad 1 \leq z_2 \leq 1.5.
\end{align*}
\]

Example 4.6 ([21,23]).

\[
\begin{align*}
\min & \quad z_1 \\
\text{s.t.} & \quad \frac{1}{7}z_1 + \frac{1}{7}z_2 - \frac{1}{10}z_1^2 - \frac{1}{10}z_2^2 \leq 1, \\
& \quad \frac{1}{14}z_1^2 + \frac{1}{14}z_2^2 - \frac{3}{7}z_1 - \frac{3}{7}z_2 \leq -1, \\
& \quad 1 \leq z_1 \leq 5.5, \quad 1 \leq z_2 \leq 5.5.
\end{align*}
\]
Example 4.7 ([21,23]).
\[
\begin{align*}
\min & \quad z_1 z_2 - 2z_1 + z_2 + 1 \\
\text{s.t.} & \quad 8z_2^2 - 6z_1 - 16z_2 \leq -11, \\
& \qquad -z_2^2 + 3z_1 + 2z_2 \leq 7, \\
& \quad 1 \leq z_1 \leq 2.5, \quad 1 \leq z_2 \leq 2.225.
\end{align*}
\]

Example 4.8 ([23,26]).
\[
\begin{align*}
\min & \quad -4z_2 + (z_1 - 1)^2 + z_2^2 - 10z_3^2 \\
\text{s.t.} & \quad z_1^2 + z_2^2 + z_3^2 \leq 2, \\
& \qquad (z_1 - 2)^2 + z_2^2 + z_3^2 \leq 2, \\
& \quad 2 - \sqrt{2} \leq z_1 \leq \sqrt{2}, \quad 0 \leq z_2, z_3 \leq \sqrt{2}.
\end{align*}
\]

Table 1. Numerical results for Examples 4.1-4.8

| Example | Refs. | Optimal solution | Optimal value | Iter. |
|---------|-------|-----------------|---------------|-------|
| 1       | This paper | (5.000000000, 1.000000000) | -16.000000000 | 1     |
|         | [15]   | (5.0, 1.0)      | -16.0         | 10    |
|         | [23]   | (5.000000000, 1.000000000) | -16.000000000 | 5     |
| 2       | This paper | (2.000000000, 1.666666667) | 6.777778336  | 3     |
|         | [17]   | (2.00003, 1.66665) | 6.7780        | 44    |
|         | [19]   | (2.000000000, 1.666666667) | 6.777782016  | 40    |
|         | [19]   | (2.000000000, 1.666666667) | 6.777781963  | 32    |
|         | [23]   | (2.000000000, 1.666666667) | 6.777777779  | 10    |
| 3       | This paper | (0.500000000, 0.500000000) | 0.500000361  | 21    |
|         | [17]   | (0.5, 0.5)      | 0.5           | 91    |
|         | [19]   | (0.5 0.5)       | 0.500004627   | 24    |
|         | [19]   | (0.5, 0.5)      | 0.5           | 29    |
|         | [23]   | (0.500000000, 0.500000000) | 0.500000442  | 37    |
|         | [24]   | (0.5, 0.5)      | 0.5           | 96    |
| 4       | This paper | (2.555730431, 3.130220581) | 118.383672506 | 44    |
|         | [22]   | (2.555779370, 3.130164639) | 118.383756475 | 210   |
|         | [23]   | (2.555745855, 3.130201668) | 118.383671904 | 59    |
| 5       | This paper | (1.500000000, 1.500000000) | -1.162882693  | 11    |
|         | [23]   | (1.500000000, 1.500000000) | -1.162882693  | 24    |
|         | [25]   | (1.5, 1.5)      | -1.16288      | 84    |
| 6       | This paper | (1.177124344, 2.177124344) | 1.177125181   | 19    |
|         | [21]   | (1.177124327, 2.177124353) | 1.177124327   | 432   |
|         | [23]   | (1.177124344, 2.177124344) | 1.177125051   | 22    |
| 7       | This paper | (2.000000000, 1.000000000) | -1.000000000  | 2     |
|         | [21]   | (2.000000, 1.000000)       | -1.0          | 24    |
|         | [23]   | (2.000000000, 1.000000000) | -0.999999410  | 21    |
| 8       | This paper | (1.0, 0.181815435, 0.983332674) | -11.363551588 | 148   |
|         | [23]   | (1.0, 0.181818470, 0.983332113) | -11.363636364 | 420   |
|         | [26]   | (0.998712, 0.196213, 0.979216) | -10.35        | 1648  |

Example 4.9 ([18,23,27]).
\[
\begin{align*}
\min & \quad -\sum_{i=1}^{n} z_i^2 \\
\text{s.t.} & \quad \sum_{i=1}^{n} z_j \leq j, \quad j \in \{1, 2, \ldots, n\}, \quad z_j \geq 0, \quad i \in \{1, 2, \ldots, n\}.
\end{align*}
\]
Table 2. Numerical results for Example 4.9

| Refs. | Dimension $n$ | Optimal value | Number of iteration | Time(s) |
|-------|---------------|---------------|---------------------|---------|
| This paper | 5 | -25.0 | 1 | 0.00231396 |
|          | 10 | -100.0 | 1 | 0.0153671 |
|          | 20 | -400.0 | 1 | 0.0993256 |
|          | 30 | -900.0 | 1 | 0.342342 |
|          | 40 | -1600.0 | 1 | 0.919515 |
|          | 50 | -2500.0 | 1 | 2.05164 |
|          | 60 | -3600.0 | 1 | 3.96127 |
|          | 70 | -4900.0 | 1 | 7.05643 |
|          | 80 | -6400.0 | 1 | 11.7382 |
|          | 90 | -8100.0 | 1 | 18.18 |
|          | 100 | -10000.0 | 1 | 27.2642 |
|          | 200 | -40000.0 | 1 | 403.016 |

Compared with the known algorithms ([18,23,27]), the numerical results for Examples 4.1-4.9 show that the proposed algorithm can be used to globally solve the quadratically constrained quadratic programs problem with higher computational efficiency.

Example 4.10 ([27]).

$$\min G_0(z) = \frac{1}{2} z^T A^0 z + (d^0)^T z$$

s.t. $$G_i(z) = \frac{1}{2} z^T A_i^j z + (d_i^j)^T z \leq b_i, \; i = 1, \ldots, m,$$

$$0 \leq z_j \leq 10, \; j = 1, 2, \ldots, n.$$ 

All elements of $A^0$ and $d^0$ are all randomly generated in $[0,1]$, all elements of $A_i^j$ and $d_i^j$ are all randomly generated in $[-1,0]$; all $b_i, i = 1, 2, \ldots, m$, are randomly generated in $[-300,-90]$, and $n = 5$.

The numerical comparisons of computational results for Example 4.10 are listed in the following Table 3, where $n$ denotes the number of variables, $m$ denotes the number of constraints. The numerical results show that our algorithm has higher computational efficiency than that of [27].

5 Concluding remarks

In this article, an effective branch-delete-bound algorithm is presented for globally solving the quadratically constrained quadratic programs problem. Based on the characteristics of quadratic function, we first introduce a new linearizing technique, by utilizing this technique the initial quadratically constrained quadratic programs problem can be converted into a linear relaxed programs problem. By utilizing the currently known upper bound and the characters of the linear relaxed programs problem of the QCQP, a deleting operation is introduced, which can be used to accelerate the convergent speed of the proposed algorithm. Next, combining the established linear relaxed
Table 3. Numerical comparisons with [27] for Example 4.10

| Parameter | Ref. [27] | This paper |
|-----------|-----------|------------|
| m | Iter | $L_{\text{max}}$ | Time(s) | Iter | $L_{\text{max}}$ | Time(s) |
| 5 | 1571 | 300 | 2.92998 | 481 | 199 | 0.9604 |
| 10 | 754 | 171 | 2.56231 | 567 | 202 | 1.2962 |
| 20 | 1490 | 462 | 14.4705 | 381 | 153 | 1.4200 |
| 30 | 1829 | 562 | 36.1397 | 394 | 159 | 2.8441 |
| 40 | 1700 | 467 | 60.9473 | 497 | 178 | 6.2495 |
| 50 | 2722 | 429 | 160.367 | 574 | 205 | 11.5478 |
| 60 | 3054 | 518 | 276.695 | 537 | 221 | 16.4134 |
| 70 | 1965 | 521 | 256.028 | 597 | 234 | 25.568 |
| 80 | 2299 | 540 | 434.802 | 506 | 179 | 29.9703 |
| 90 | 1961 | 623 | 485.656 | 526 | 199 | 42.3564 |

programs problem with branching operation and deleting operation in a branch-and-bound framework, we formulate a branch-delete-bound algorithm for effectively solving the QCQP. By subsequently dividing the initial rectangle and subsequently solving a series of linear relaxed programs problems, the presented algorithm is convergent to the global minimum of the initial problem (QCQP). Compared with the known methods, numerical results demonstrate that the proposed branch-delete-bound method has higher computational efficiency.

Competing interests
The authors declare that they have no competing interests.

Acknowledgement: This paper is supported by the National Natural Science Foundation of China under Grant (61304061), the Natural Science Foundation of Henan Province (152300410097), the Higher School Key Scientific Research Projects of Henan Province (18A110019, 17A110021, 16A110014), the Henan Provincial Youth Backbone Teachers Training Program (2016GGJS-107), the Major Scientific Research Projects of Henan Institute of Science and Technology (2015ZD07), the Basic and Advanced Technology Research Project of Henan Province (152300410097), the High-level Scientific Research Personnel Project for Henan Institute of Science and Technology(2015037).

References

[1] Jiao H., Liu S., An efficient algorithm for quadratic sum-of-ratios fractional programs problem, Numer. Func. Anal. Opt., (in press), DOI: 10.1080/01630563.2017.1328769
[2] Gao Y., Wei F., A new bound-and-reduce approach of nonconvex quadratic programming problems, Appl. Math. Comput., 2015, 250, 298-308
[3] Azab A., Quadratic assignment problem mathematical modelling for process planning, Int. J. Comput. Integ. M., 2016, 29, 561-580
[4] Kong X., Huang G., Fan Y., Li Y., A duality theorem-based algorithm for inexact quadratic programming problems: Application to waste management under uncertainty, Eng. Optimiz., 2016, 48, 562-581
[5] Jiao H., Chen Y., Yin J., Optimality condition and iterative thresholding algorithm for $l_p$-regularization problems, SpringerPlus, 2016, https://doi.org/10.1186/s40064-016-3516-3
[6] Jiao H., Wang Z., Chen Y., Global optimization algorithm for sum of generalized polynomial ratios problem, Appl. Math. Model., 2013, 37(1), 187-197
[7] Wang Z.-J., Li K.W., Group decision making with incomplete intuitionistic preference relations based on quadratic programming models, Comput. Ind. Eng., 2016, 93, 162-170
[8] Bose S., Gayme D.F., Chandy K.M., Low S.H., Quadratically constrained quadratic programs on acyclic graphs with application to power flow, IEEE Trans. Control Net. Syst., 2015, 2, 278-287
[9] Jiao H., Liu S., Yin J., Zhao Y., Outcome space range reduction method for global optimization of sum of affine ratios problem, Open Math., 2016, 14, 736-746
[10] Sherali H.D., Tuncbilek C.H., A reformulation-convexification approach for solving nonconvex quadratic programming problems, J. Glob. Optim., 1995, 7, 1-31
[11] Voorhis T.V., A global optimization algorithm using lagrangian underestimates and the interval newton method, J. Glob. Optim., 2002, 24, 349-370
[12] Linder J., A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs, Math. Program., 2005, 103, 251-282
[13] Burer S., Vandenbussche D., A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations, Math. Program., 2008, 113, 259-282
[14] Zheng X.J., Sun X.L., Li D., Convex relaxations for nonconvex quadratically constrained quadratic programming: matrix cone decomposition and polyhedral approximation, Math. Program., 2011, 129, 301-329
[15] Thoai N.V., Duality Bound Method for the General Quadratic Programming Problem with Quadratic Constraints, J. Optim. Theory Appl., 2000, 107, 331-354
[16] Shen P., Duan Y., Ma Y., A robust solution approach for nonconvex quadratic programs with additional multiplicative constraints, Appl. Math. Comput., 2008, 201, 514-526
[17] Gao Y., Shang Y., Zhang L., A branch and reduce approach for solving nonconvex quadratic programming problems with quadratic constraints, OR transactions, 2005, 9, 9-20
[18] Gao Y., Xue H., Shen P., A new rectangle branch-and-reduce approach for solving nonconvex quadratic programming problems, Appl. Math. Comput., 2005, 168, 1409-1418
[19] Shen P., Liu L., A global optimization approach for quadratic programs with nonconvex quadratic constraints, Chin. J. Eng. Math., 2008, 25, 923-926
[20] Qu S.-J., Zhang K.-C., Ji Y., A global optimization algorithm using parametric linearization relaxation, Appl. Math. Comput., 2007, 186, 763-771
[21] Shen P., Jiao H., A new rectangle branch-and-pruning approach for generalized geometric programming, Appl. Math. Comput., 2006, 183, 1027-1038
[22] Wang Y., Liang Z., A deterministic global optimization algorithm for generalized geometric programming, Appl. Math. Comput., 2005, 168, 722-737
[23] Jiao H., Chen Y., A global optimization algorithm for generalized quadratic programming, J. Appl. Math., 2013, http://dx.doi.org/10.1155/2013/215312
[24] Wang Y.J., Zhang K.C., Gao Y.L., Global optimization of generalized geometric programming, Comput. Math. Appl., 2004, 48, 1505-1516
[25] Shen P., Linearization method of global optimization for generalized geometric programming, Appl. Math. Comput., 2005, 162, 353-370
[26] Shen P., Li X., Branch-reduction-bound algorithm for generalized geometric programming, J. Glob. Optim., 2013, 56, 1123-1142
[27] Jiao H., Liu S., Lu N., A parametric linear relaxation algorithm for globally solving nonconvex quadratic programming, Appl. Math. Comput., 2015, 250, 973-985
[28] Jiao H., Liu S., Zhao Y., Effective algorithm for solving the generalized linear multiplicative problem with generalized polynomial constraints, Appl. Math. Model., 2015, 39, 7568-7582