Jeff Linderoth

A Simplicial Branch-and-Bound Algorithm for Solving Quadratically Constrained Quadratic Programs

Abstract. We propose a branch-and-bound algorithm for solving nonconvex quadratically-constrained quadratic programs. The algorithm is novel in that branching is done by partitioning the feasible region into the Cartesian product of two-dimensional triangles and rectangles. Explicit formulae for the convex and concave envelopes of bilinear functions over triangles and rectangles are derived and shown to be second-order cone representable. The usefulness of these new relaxations is demonstrated both theoretically and computationally.

1. Introduction

In this paper, we discuss branch-and-bound methods for solving the quadratically constrained quadratic program (QCQP) which can be written as

$$\min_{x \in \mathbb{R}^n} \{ q_0(x) \mid q_k(x) \geq b_k \ \forall k \in M, l_i \leq x_i \leq u_i \ \forall i \in I \},$$

where $q_k = c_k^T x + x^T Q_k x \ \forall k \in M \cup \{0\}$, and $I = \{1, 2, \ldots n\}$. We assume that explicit lower bounds and upper bounds on x_i are known, so that the feasible region is compact. We do not assume any convexity properties of the $q_k(x)$, so the objective function of the problem may be convex, concave, or indefinite, and the set of feasible solutions need not be convex or connected.

QCQP generalizes many well-known, difficult optimization problems. Linear mixed 0-1 programming, fractional programming, bilinear programming, polynomial programming, and bilevel programming problems can all be written as instances of QCQP, so QCQP is NP-Hard. From a practical standpoint, QCQP is one of the most challenging optimization problems—the current size of instances that can be solved to provable optimality remains very small in comparison to other NP-Hard problem classes such as mixed integer programming.

In this work, we describe a branch-and-bound algorithm for solving QCQP that is based on subdividing the feasible region into the Cartesian product of triangles and rectangles. It can be viewed as an extension of the work of Al-Khayyal and Falk [2], who derive a formula for the convex envelope of a product of variables over a rectangle and give a branch-and-bound algorithm based on the formula. Al-Khayyal [1] extends the formula for the concave envelope of the product of variables, and Al-Khayyal, Larsen, and Van Voorhis develop a branch-and-bound algorithm based on these relaxations [3].

Jeff Linderoth: Industrial and Systems Engineering Department, Lehigh University, 200 West Packer Avenue, Bethlehem, PA 18015; jtl3@lehigh.edu

Raber [24, 25] also gives a simplicial-subdivision based algorithm for QCQP. In Raber's work, the feasible region is enclosed in a high-dimensional simplex, and this simplex is subdivided in the spirit suggested by Horst [15]. Our work is different in that the feasible region is enclosed in an initial hyper-rectangle, and that hyper-rectangle is subdivided into the Cartesian product of rectangles and triangles (low-dimensional simplices). Sherali and Alameddine [30] use the Reformulation-Linearization Technique (RLT) to solve bilinear programming problems, and Audet *et al.* [5] extend the use of RLT in solving QCQP by including different classes of linearizations. Kim and Kojima [19] extend the lift-and-project idea of RLT to create a second-order cone programming relaxation for QCQP. DC (Difference of Convex) programming techniques were used by Phong, Tao and Hoai An to solve QCQP, and DC programming techniques form the basis of the general global optimization software αBB [4]. BARON [28, 33] is a mature, sophisticated software package that can solve QCQP using convex/concave envelopes (in the spirit of [2]), but augmented with features such as sophisticated range reduction and branching techniques [27, 34]. BARON also can be accessed through a link to the commercial modeling language GAMS.

The paper is organized as follows. In Section 2, expressions for the convex and concave envelope of the bilinear function $f(x, y) = xy$ over rectangular and triangular regions are derived. In Section 3 a simple triangle-based branching scheme is introduced, and based upon such a scheme a nonlinear programming relaxation for QCQP is given. Section 4 demonstrates that the nonlinear constraints in the relaxation are representable as second-order cone constraints. A polyhedral outerapproximation to the second-order cone is given, resulting in a new linear programming relaxation to QCQP. Section 5 introduces two measures of the tightness of approximations to the convex and concave envelopes and computes these measures for the envelope expressions derived in Section 2. In Section 6, computational results are given to show the usefulness of the triangle-based branch-and-bound method.

2. Relaxations for QCQP

Tractable relaxations of the nonconvex problem QCQP can be obtained using the notions of *convex envelopes* and *concave envelopes*. For a function $f : \Omega \rightarrow \mathbb{R}$, the *convex envelope* of f over Ω, denoted vex$_\Omega(f)$, is the pointwise supremum of convex underestimators of f over Ω. Likewise, the concave envelope of f over Ω, denoted cav$_\Omega(f)$ is the pointwise infimum of concave overestimators of f over Ω. This paper refers extensively to the convex and concave envelope expressions for the bilinear function $f(x, y) = xy$. To simplify the notation, the following definitions are made:

$$\text{vexxy}_\Omega \stackrel{\text{def}}{=} \text{vex}_\Omega(xy) \text{ and}$$
$$\text{cavxy}_\Omega \stackrel{\text{def}}{=} \text{cav}_\Omega(xy).$$

2.1. Envelopes Over Rectangles

McCormick [22] gave a linear relaxation for the product of variables xy over a rectangle $R = \{(x, y) \in \mathbb{R}^2 \mid l_x \leq x \leq u_x, l_y \leq y \leq u_y\}$, and Al-Khayyal and Falk [2] and Al-Khayyal [1] subsequently showed that the linear relaxation defined the convex and concave envelopes.

Theorem 1 (McCormick [22], Al-Khayyal and Falk [2], Al-Khayyal [1]).

The convex and concave envelopes of the bilinear function xy over a rectangular region

$$R \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{R}^2 \mid l_x \leq x \leq u_x,\ l_y \leq y \leq u_y\}$$

are given by the expressions

$$vexxy_R(x,y) = \max\{l_y x + l_x y - l_x l_y,\ u_y x + u_x y - u_x u_y\} \tag{1}$$

$$cavxy_R(x,y) = \min\{u_y x + l_x y - l_x u_y,\ l_y x + u_x y - u_x l_y\}. \tag{2}$$

Using Theorem 1 and introducing auxiliary variables z_{ij} to act as an approximation of $x_i x_j$, a linear programming relaxation LPR_1 of QCQP can be written. For ease of notation, the z_{ij} variables are arranged into a matrix Z, and we write the inner product of two matrices $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times n}$ as $A \bullet B = \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{ij}$.

$$v_{LPR_1} = \min_{x,z}\ c^T x + Q_0 \bullet Z \tag{LPR_1}$$

subject to

$$c_k^T x + Q_k \bullet Z \geq b_k \qquad \forall k \in M,$$
$$z_{ij} - l_i x_j - l_j x_i + l_i l_j \geq 0 \qquad \forall i \in I, j \in I,$$
$$z_{ij} - u_i x_j - u_j x_i + u_i u_j \geq 0 \qquad \forall i \in I, j \in I,$$
$$z_{ij} - l_i x_j - u_j x_i + l_i u_j \leq 0 \qquad \forall i \in I, j \in I,$$
$$z_{ij} - u_i x_j - l_j x_i + u_i l_j \leq 0 \qquad \forall i \in I, j \in I,$$
$$x_i \in [l_i, u_i] \qquad \forall i \in I.$$

In practice, the auxiliary variables z_{ij} need only be introduced for the nonzero elements of the Q matrices, and the expressions for the convex or concave envelopes are included to bound the values of z_{ij} depending on the sign of the matrix elements q_{ij} and right hand side elements b_k.

Figure 1 shows a plot of the function xy and of its convex and concave envelopes over a rectangle. The envelopes $vexxy_R$ and $cavxy_R$ define a simplex in $\mathbb{R}^3$, and using this geometric observation, an equivalent linear programming relaxation can be written using a dual notion of the convex and concave envelope. The relaxation is obtained by noticing that over R, the values of xy are contained in the convex hull of the points obtained by evaluating xy at the extreme points of R. Because xy is a linear function if either x or y is fixed, the approximation is exact along the boundary. Sherali and Alameddine derive a similar expression using arguments from LP duality [29]. We use Δ^k to denote the unit simplex in $\mathbb{R}^k$.

Theorem 2 (Sherali and Alameddine [29]).
The convex and concave envelopes of the bilinear function xy over a rectangular region

$$R \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{R}^2 \mid l_x \leq x \leq u_x,\ l_y \leq y \leq u_y\}$$

are given by the expressions

$$vexxy_R(x,y) = \min_{\lambda \in \Delta^4} \{l_x l_y \lambda_1 + u_x l_y \lambda_2 + u_x u_y \lambda_3 + l_x u_y \lambda_4 \mid$$
$$x = l_x(\lambda_1 + \lambda_4) + u_x(\lambda_2 + \lambda_3),\ y = l_y(\lambda_1 + \lambda_2) + u_y(\lambda_3 + \lambda_4)\}$$

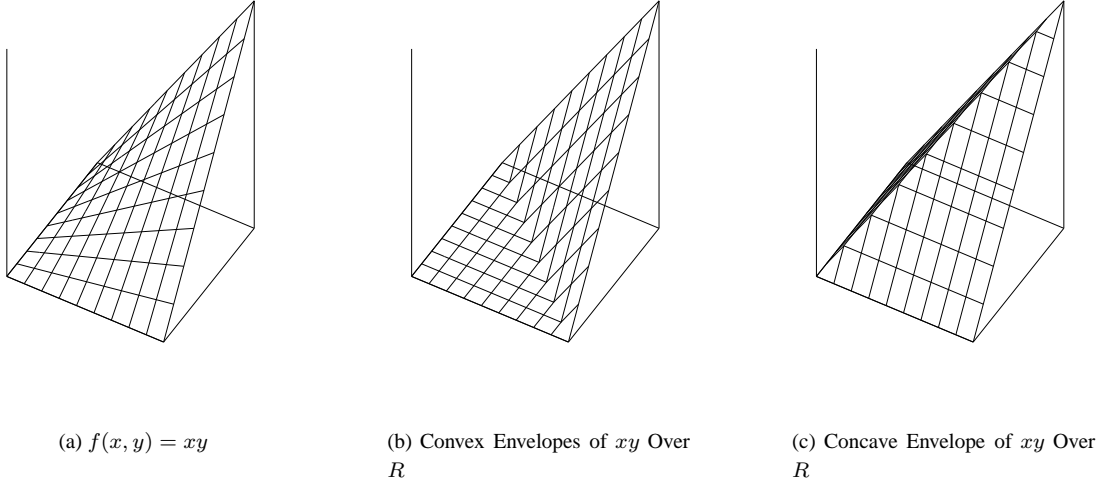(a) $f(x, y) = xy$ (b) Convex Envelopes of xy Over R (c) Concave Envelope of xy Over R

Fig. 1. xy and its Convex and Concave Envelopes

and

$$cavxy_R(x, y) = \max_{\lambda \in \Delta^4} \{l_x l_y \lambda_1 + u_x l_y \lambda_2 + u_x u_y \lambda_3 + l_x u_y \lambda_4 \mid$$

$$x = l_x(\lambda_1 + \lambda_4) + u_x(\lambda_2 + \lambda_3), \ y = l_y(\lambda_1 + \lambda_2) + u_y(\lambda_3 + \lambda_4)\}.$$

By Theorem 2, a linear programming relaxation LPR_2 of QCQP can be written as follows:

$$v_{LPR_2} = \min \quad c^T x + Q_0 \bullet Z \qquad (LPR_2)$$

subject to

$$c_k^T x + Q_k \bullet Z \geq b_k \qquad \forall k \in M,$$
$$(\lambda_{ij1} + \lambda_{ij4})l_i + (\lambda_{ij2} + \lambda_{ij3})u_i = x_i \qquad \forall i \in I,$$
$$(\lambda_{ij1} + \lambda_{ij2})l_j + (\lambda_{ij3} + \lambda_{ij4})u_j = x_j \qquad \forall j \in I,$$
$$\lambda_{ij1}l_i l_j + \lambda_{ij2}u_i l_j + \lambda_{ij3}u_i u_j + \lambda_{ij4}l_i u_j = z_{ij} \qquad \forall i \in I, j \in I,$$
$$\sum_{l=1}^{4} \lambda_{ijl} = 1 \qquad \forall i \in I, \forall j \in I,$$
$$x_i \in [l_i, u_i] \qquad \forall i \in I,$$
$$\lambda_{ijl} \geq 0 \qquad \forall i \in I, \forall j \in I, \forall l \in \{1, 2, 3, 4\}.$$

It follows directly from Theorems 1 and 2 that $v_{LPR_1} = v_{LPR_2}$.

2.2. Envelopes Over Triangles

The branch-and-bound method described subsequently in Section 3 subdivides the initial hyper-rectangle $\Omega = \times_{i \in N}[l_i, u_i]$ not into finer and finer hyper-rectangles, but into the Cartesian product of triangles and rectangles. To develop tight and tractable relaxations for the method, it is useful to complement Theorem 1 by deriving expressions for the convex and concave envelopes of the function xy over various triangular shapes. Two of the expressions (Theorems 3 and 4) also appear in an implicit form in the work of Sherali and Alameddine [29], but this is the first work to explicitly give an algebraic description of the envelopes over triangular regions. For the remainder of the paper, we will refer to the following triangular regions.

$$R_{x,y} \overset{\text{def}}{=} \{(x,y) \in \mathbb{R}^2 \mid l_x \leq x \leq u_x, \; l_y \leq y \leq u_y\} \tag{3}$$

$$SE_{x,y} \overset{\text{def}}{=} R_{x,y} \cap \{(x,y) \in \mathbb{R}^2 \mid (y - l_y)(u_x - l_x) \leq (u_y - l_y)(x - l_x)\} \tag{4}$$

$$NW_{x,y} \overset{\text{def}}{=} R_{x,y} \cap \{(x,y) \in \mathbb{R}^2 \mid (y - l_y)(u_x - l_x) \geq (u_y - l_y)(x - l_x)\} \tag{5}$$

$$SW_{x,y} \overset{\text{def}}{=} R_{x,y} \cap \{(x,y) \in \mathbb{R}^2 \mid (y - u_y)(u_x - l_x) \leq (l_y - u_y)(x - l_x)\} \tag{6}$$

$$NE_{x,y} \overset{\text{def}}{=} R_{x,y} \cap \{(x,y) \in \mathbb{R}^2 \mid (y - u_y)(u_x - l_x) \geq (l_y - u_y)(x - l_x)\} \tag{7}$$

$$N_{x,y} \overset{\text{def}}{=} NE_{x,y} \cap NW_{x,y}, \tag{8}$$

$$S_{x,y} \overset{\text{def}}{=} SE_{x,y} \cap SW_{x,y}, \tag{9}$$

$$E_{x,y} \overset{\text{def}}{=} NE_{x,y} \cap SE_{x,y}, \text{ and} \tag{10}$$

$$W_{x,y} \overset{\text{def}}{=} NW_{x,y} \cap SW_{x,y}. \tag{11}$$

Figure 2 depicts the various triangular regions. The N, S, E, and W notation is meant to mimic the "North", "South", "East", and "West" compass directions.

(a) Triangles N, S, E, W (b) Triangles SE, NW (c) Triangles SW, NE
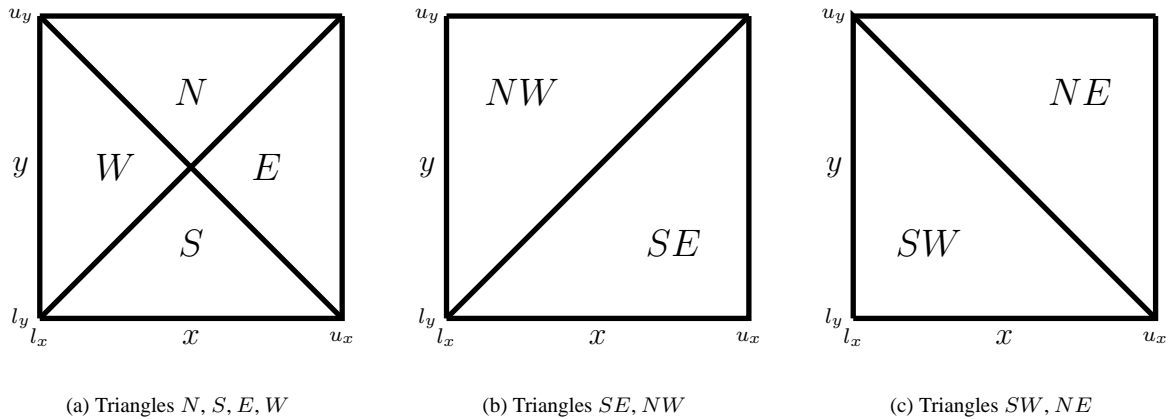
Fig. 2. Pictorial Description of Triangular Regions Studied

Theorem 3 (Sherali and Alameddine [29]).
The convex envelope of the function xy over the triangular region $SE_{x,y}$ is given as

$$vexxy_{SE_{x,y}}(x,y) = \begin{cases} u_x l_y & \text{if } x = u_x \text{ and } y = l_y, \\ q_{SE}(x,y)/l_{SE}(x,y) & \text{otherwise,} \end{cases} \tag{12}$$

where

$$q_{SE}(x,y) \overset{\text{def}}{=} (l_y^2 - l_y u_y)x^2 + (u_x^2 - l_x u_x)y^2$$
$$+ (l_y u_x - l_x u_y)xy + (l_x l_y u_y + l_y u_x u_y - 2l_y^2 u_x)x$$
$$+ (l_x l_y u_x + l_x u_x u_y - 2u_x^2 l_y)y + (l_y^2 u_x^2 - l_x l_y u_x u_y),$$

and

$$l_{SE}(x,y) \overset{\text{def}}{=} -2l_y u_x + l_x l_y + u_x u_y + (u_x - l_x)y + (l_y - u_y)x.$$

The convex envelopes of the function xy over the triangular regions $SE_{x,y}, S_{x,y}, E_{x,y}$ are the same:

$$vexxy_{SE_{x,y}}(x,y) = vexxy_{S_{x,y}}(x,y) = vexxy_{E_{x,y}}(x,y). \tag{13}$$
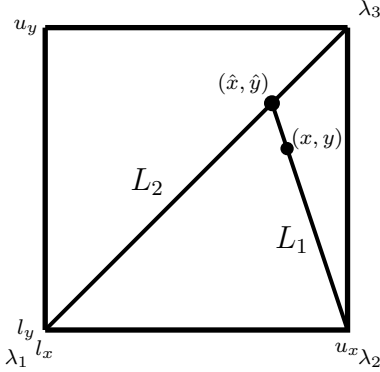

Fig. 3. Depiction of Entities in Proof of Theorem 3

Proof. In Figure 3, the line segment $L_1 = \{(x,y) \in \mathbb{R}^2 \mid y - l_y = m(x - u_x)\}$, where $m = (l_y - \hat{y})/(u_x - \hat{x})$. The function xy is strictly concave on the domain L_1, since on this domain, $xy = mx^2 + (l_y - mu_x)x$, which is a strictly concave function, since $m < 0$. Our aim is to define the equation of line segment connecting $(\hat{x}, \hat{y}, \hat{x}\hat{y})$ and $(u_x, l_y, u_x l_y) \in \mathbb{R}^3$ as a function of x and y. To that end, define (convex) multipliers $\lambda_1, \lambda_2, \lambda_3$ associated with the points (l_x, l_y), (u_x, l_y), and (u_x, u_y), respectively. The following five equations all define valid relationships between $x, y, \hat{x}, \hat{y}, \lambda_1, \lambda_2,$ and λ_3:

$$(\hat{y} - l_y)(u_x - l_x) = (u_y - l_y)(\hat{x} - l_x),$$
$$(\hat{y} - l_y)(u_x - x) = (l_y - y)(\hat{x} - u_x),$$
$$\lambda_1 l_x + \lambda_2 u_x + \lambda_3 u_x = x,$$
$$\lambda_1 l_y + \lambda_2 l_y + \lambda_3 u_y = y,$$
$$\lambda_1 + \lambda_2 + \lambda_3 = 1.$$

The solution of these five equations in the five unknowns $(\hat{x}, \hat{y}, \lambda_1, \lambda_2, \lambda_3)$ yields the unique solution

$$\hat{x} = \frac{(u_x^2 - l_x u_x)y + (l_x l_y - l_x u_y)x - u_x^2 l_y + l_x u_x u_y}{(u_x - l_x)y + (l_y - u_y)x + l_x l_y + u_x u_y - 2 l_y u_x} \tag{14}$$

$$\hat{y} = \frac{(u_x u_y - l_x u_y)y + (l_y^2 - l_y u_y)x - l_y^2 u_x + l_x l_y u_y}{(u_x - l_x)y + (l_y - u_y)x + l_x l_y + u_x u_y - 2 l_y u_x} \tag{15}$$

$$\lambda_2 = \frac{(u_y - l_y)x - (u_x - l_x)y + l_y u_x - u_y l_x}{u_x u_y - l_y u_x + l_x l_y - u_y l_x}. \tag{16}$$

The line segment connecting $(\hat{x}, \hat{y}, \hat{x}\hat{y})$ and $(u_x, l_y, u_x l_y)$ can be written as

$$F_{SE}(x, y) \stackrel{\text{def}}{=} \lambda_2 u_x l_y + (1 - \lambda_2)\hat{x}\hat{y}. \tag{17}$$

Substituting equations (14)—(16) into (17) yields

$$F_{SE}(x, y) = q_{SE}(x, y)/l_{SE}(x, y).$$

The remainder of the proof argues that $F_{SE}(x, y)$ is indeed the convex envelope. It is clear that no convex function is larger, since $F_{SE}(x, y)$ is the functional form of the line segment connecting the two endpoints of a strictly concave function. The convexity of $F_{SE}(x, y)$ can be established by examining its Hessian. The function $F_{SE}(x, y)$ is not defined at (u_x, l_y), but it follows from construction that $\text{vexxy}_{SE}(u_x, l_y)) = u_x l_y$. (See also Corollary 2.5 of Tawarmalani and Sahinidis [33]). The exact same proof also shows that $\text{vexxy}_{SE_{x,y}}$ is the convex envelope over the regions $S_{x,y}$ and $E_{x,y}$. □

Example 1. Let $SE' = \{(x, y) \in \mathbb{R}^2 \mid 0 \le x \le 1, 0 \le y \le 1, x \ge y\}$, then

$$\text{vexxy}_{SE'}(x, y) = \frac{y^2}{1 + y - x}.$$

Figure 4 is a plot of $\text{vexxy}_{SE'}(x, y)$. (Note that the function is only the convex envelope in the lower triangular region of the figure). The convexity of the function can be seen by computing the Hessian matrix for $\text{vexxy}_{SE'}(x, y)$:

$$\nabla^2(\text{vexxy}_{SE'}(x, y)) = \frac{1}{(1 - x + y)^3} \begin{bmatrix} 2y^2 & 2y(1 - x) \\ 2y(1 - x) & 2(1 - 2x + x^2) \end{bmatrix}.$$

For $(x, y) \in SE'$, the relations $(1 - x + y)^3 \ge 0, 2y^2 \ge 0, 2(1 - 2x + x^2) \ge 0$, and $4y^2(1 - 2x + x^2) - 4y^2(1 - x)^2 = 0$ all hold, so $\nabla^2 \text{vexxy}_{SE'}(x, y)$ is positive semidefinite, and $\text{vexxy}_{SE'}(x, y)$ is indeed convex on SE'.

Theorem 4. *The convex envelope of the function xy over the triangular region $NW_{x,y}$ is given as*

$$\text{vexxy}_{NW_{x,y}}(x, y) = \begin{cases} l_x u_y & \text{if } x = l_x \text{ and } y = u_y, \\ q_{NW}(x, y)/l_{NW}(x, y) & \text{Otherwise,} \end{cases} \tag{18}$$

where

$$q_{NW}(x, y) = (u_y^2 - l_y u_y)x^2 + (u_y l_x - l_y u_x)xy$$
$$+ (-l_x u_x + l_x^2)y^2 + (u_y l_y u_x - 2u_y^2 l_x + u_y l_y l_x)x$$
$$+ (l_x l_y u_x - 2l_x^2 u_y + l_x u_y u_x)y - l_x u_y l_y u_x + u_y^2 l_x^2$$
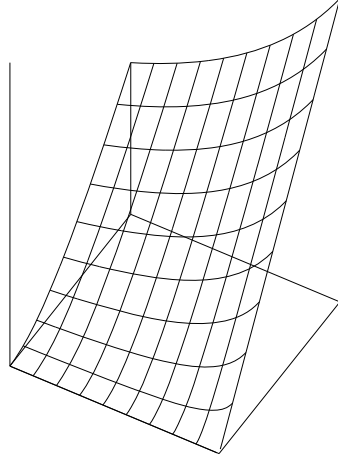
Fig. 4. $\text{vexxy}_{SE'}(x, y)$

and

$$l_{NW}(x, y) = (u_y - l_y)x + (l_x - u_x)y + u_yu_x - 2u_yl_x + l_yl_x.$$

Further, the convex envelopes of the function xy over the triangular regions $NW_{x,y}, N_{x,y}, W_{x,y}$ are equivalent:

$$vexxy_{NW_{x,y}}(x, y) = vexxy_{N_{x,y}}(x, y) = vexxy_{W_{x,y}}(x, y). \tag{19}$$

Proof. The proof proceeds in a similar fashion to the proof of Theorem 3. Namely, it can be shown that $q_{NW}(x, y)/l_{NW}(x, y)$ is precisely the expression for the line segment connecting the point (l_x, u_y, l_xu_y) and $(\hat{x}, \hat{y}, \hat{x}\hat{y})$ where $(\hat{x}, \hat{y})$ is on the line segment L_2 in Figure 3, and the same proof suffices to show that the the convex envelopes over the triangles $NW_{x,y}, N_{x,y}$ and $W_{x,y}$ all share the same functional form. □

Concave envelopes for triangles $SW_{x,y}$ and $NE_{x,y}$ are derived from similar arguments.

Theorem 5. *The concave envelope of the function xy over the region $SW_{x,y}$ is given as*

$$cavxy_{SW_{x,y}}(x, y) = \begin{cases} l_xl_y & \text{if } x = l_x \text{ and } y = l_y, \\ q_{SW}(x, y)/l_{SW}(x, y) & \text{otherwise.} \end{cases} \tag{20}$$

where

$$q_{SW}(x, y) = (l_yu_y - l_y^2)x^2 + (u_xu_y - l_xl_y)xy$$
$$+ (l_xu_x - l_x^2)y^2 + (-l_xl_yu_y - l_yu_xu_y + 2l_y^2l_x)x$$
$$+ (-l_xl_yu_x - l_xu_xu_y + 2l_x^2l_y)y + (-l_x^2l_y^2 + l_xl_yu_xu_y)$$

and

$$l_{SW}(x, y) = (u_y - l_y)x + (u_x - l_x)y - l_x u_y - u_x l_y + 2l_x l_y.$$

The concave envelopes of the function xy over the triangular regions $SW_{x,y}, S_{x,y}, W_{x,y}$ are equivalent:

$$cavxy_{SW_{x,y}}(x, y) = cavxy_{S_{x,y}}(x, y) = cavxy_{W_{x,y}}(x, y).$$

Proof. In Figure 5, the line segment $L_1 = \{(x, y) \in \mathbb{R}^2 \mid y - l_y = m(x - l_x)\}$, where $m = (\hat{y} - l_y)/(u_x - \hat{x})$, and the function xy is strictly convex on L_1, since on this domain $xy = mx^2 + (l_y - ml_x)x$, with $m > 0$. Similar to the proof of Theorem 3, it can be shown that $q_{SW}(x, y)/l_{SW}(x, y)$ is the functional form of the line segment in $\mathbb{R}^3$ connecting $(l_x, l_y, l_x l_y)$ to $(\hat{x}, \hat{y}, \hat{x}\hat{y})$, and that the infinite collection of these line segment forms the concave envelope. The same proof suffices to establish the formulae for the concave envelopes over $S_{x,y}$ and $W_{x,y}$. □
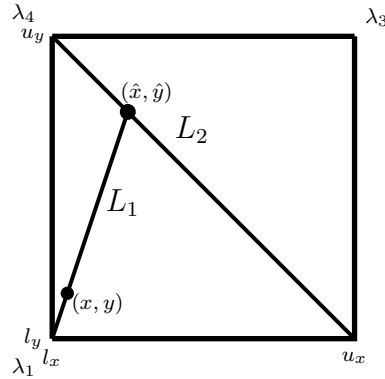

Fig. 5. Depiction of Entities in Proof of Theorem 5

Example 2. Let $SW' = \{(x, y) \in \mathbb{R}^2 \mid 0 \le x \le 1, 0 \le y \le 1, x + y \le 1\}$, then

$$\text{cavxy}_{SW'}(x, y) = \frac{xy}{x + y}.$$

Figure 6 shows a graph of cavxy $_{SW_{x,y}}$.

Theorem 6. *The concave envelope of the function xy over the region $NE_{x,y}$ is given as*

$$cavxy_{NE}(x, y) = \begin{cases} u_x u_y & \text{if } x = u_x \text{ and } y = u_y, \\ q_{NE}(x, y)/l_{NE}(x, y) & \text{otherwise.} \end{cases} \tag{21}$$

where

$$q_{NE}(x, y) = (u_y^2 - l_y u_y)x^2 + (u_x u_y - l_x l_y)xy +$$
$$(u_x^2 - l_x u_x)y^2 + (-2u_y^2 u_x + u_x l_y u_y + l_x l_y u_y)x +$$
$$(-2u_x^2 u_y + l_x u_x u_y + l_x l_y u_x)y + u_x^2 u_y^2 - l_x u_x l_y u_y$$
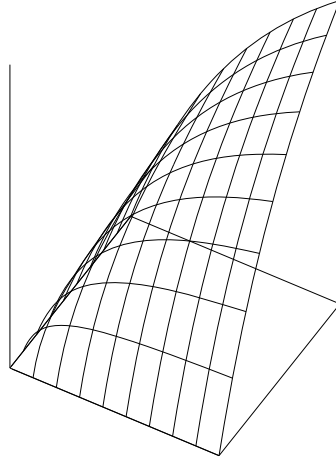
Fig. 6. cavxy$_{SW_{x,y}}(x, y)$

and

$$l_{NE}(x, y) = (u_y - l_y)x + (u_x - l_x)y - 2u_x u_y + l_x u_y + l_y u_x.$$

The concave envelopes of the function xy over the triangular regions $NE_{x,y}$, $N_{x,y}$, $E_{x,y}$ are the same:

$$cavxy_{NE_{x,y}}(x, y) = cavxy_{N_{x,y}}(x, y) = cavxy_{E_{x,y}}(x, y). \tag{22}$$

Proof. The proof is to show that $q_{NE}(x, y)/l_{NE}(x, y)$ is the expression for the line segment in $\mathbb{R}^3$ connecting $(u_x, u_y, u_x u_y)$ to $(\hat{x}, \hat{y}, \hat{x}\hat{y})$, where $(\hat{x}, \hat{y})$ is on the line segment L_2 in Figure 5. Once again, the same proof shows that $NE_{x,y}$, $N_{x,y}$, and $E_{x,y}$ all share the same concave envelope. $\square$

To augment Theorems 3-6, we give the complementary convex and concave expressions over the four triangular regions of the theorems. The envelopes are the equations of appropriate planes in the convex and concave envelope expressions of xy over a rectangle R from Theorem 1:

$$\text{cavxy}_{SE_{x,y}}(x, y) = l_y x + u_x y - u_x l_y, \tag{23}$$

$$\text{cavxy}_{NW_{x,y}}(x, y) = u_y x + l_x y - l_x u_y, \tag{24}$$

$$\text{vexxy}_{SW_{x,y}}(x, y) = l_y x + l_x y - l_x l_y, \tag{25}$$

$$\text{vexxy}_{NE_{x,y}}(x, y) = u_y x + u_x y - u_x u_y. \tag{26}$$

3. Triangle-Based Branch-and-Bound

This section demonstrates how to exploit the formulae for convex and concave envelopes derived in Section 2 in a triangle-based branch and bound scheme.
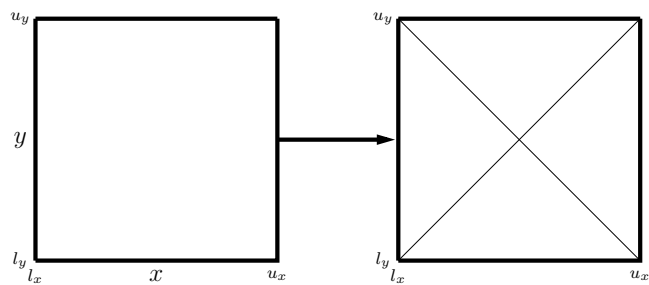
3.1. Partitioning Schemes

Starting with each pair of variables (x_i, x_j) being constrained to lie in the rectangle R_{x_i,x_j}, the branch-and-bound procedure partitions the feasible region by subdividing the regions R_{x_i,x_j}. The partitioning tightens the convex and concave envelope approximations of the bilinear term $x_i x_j$ by tightening the lower and upper bounds on the variables x_i and x_j, which in turn improves the bounds on the variable z_{ij}. Typically, in a rectangle-based branch-and-bound scheme, the original rectangle is subdivided into either two or four rectangles [2,3]. When triangles are allowed as elements of the partition, the variety of ways in which the feasible region might be divided increases. For example, a rectangular region R_{x_i,x_j} might be divided into four triangular regions of the type $N_{x_i,x_j}, S_{x_i,x_j}, E_{x_i,x_j}, W_{x_i,x_j}$, as in Figure 7(a). A rectangular region R_{x_i,x_j} might be divided into two regions, one of the type NW_{x_i,x_j} and the other of the type SE_{x_i,x_j}, like Figure 7(b). A region of type NW_{x_i,x_j} might be divided into a region of type R_{x_i,x_j} and two regions of type NW_{x_i,x_j} in a manner suggested by Figure 7(c). A complete description of a partitioning rule is given by specifying the manner to partition each shape that might occur in a rule. An example of such a rule is depicted by Figure 8. In this partitioning rule, each rectangular region is divided into two triangular regions, of types $\{NW, SE\}$ or $\{SW, NE\}$), depending on whether the value of variable z_{ij} is smaller or larger than the product of values $x_i x_j$. Triangular regions are partitioned into the union of one rectangle and two smaller triangular regions of the same type.

In practice, partitioning schemes often ensure that the solution to the relaxed problem intersects with the boundary of feasible region in child subproblems. In order to ensure global convergence, the branching scheme should partition the region so that nested sequences of partitions converge to a singleton point. Then, since the convex and concave envelope approximations are exact at such a point, the bounding procedure is *consistent* and global convergence can be assured as long as the node selection operation periodically evaluates the node with the smallest lower bound (c.f. [16]).

### 3.2. Nonlinear Programming Relaxations

In any of the partitioning schemes of Section 3.1, each pair of variables (x_i, x_j) is constrained to be in a rectangular or triangular region of a type given by the sets (3)—(11) in which the lower and upper bounds on the variables x_i and x_j have been adjusted according to the partitioning. In what follows, we let the bounds (l_{x_i}, u_{x_i}) on a variable x_i refer to the lower and upper bounds of a variable at a particular node of the branch-and-bound tree (i.e. within a particular partitioning of the feasible region), not the original lower and upper bounds on variable x_i.

(a) Four-Triangle-Partition
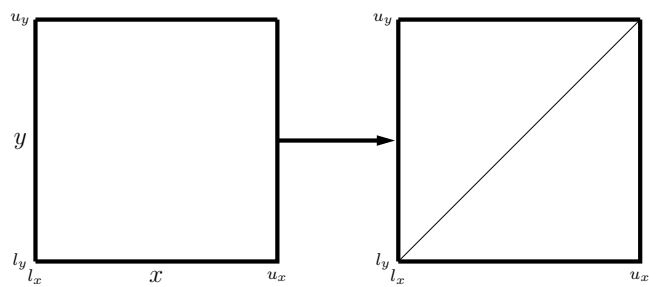
(b) Two-Triangle-Partition
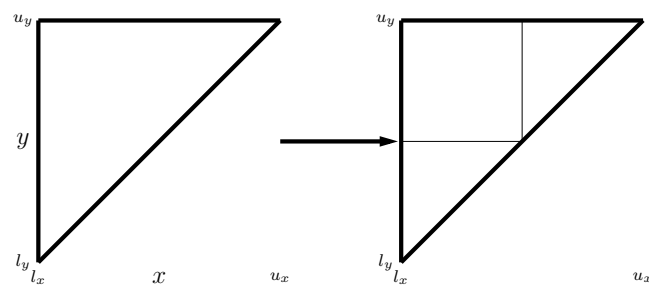
(c) Two-Triangle-Rectangle Partition

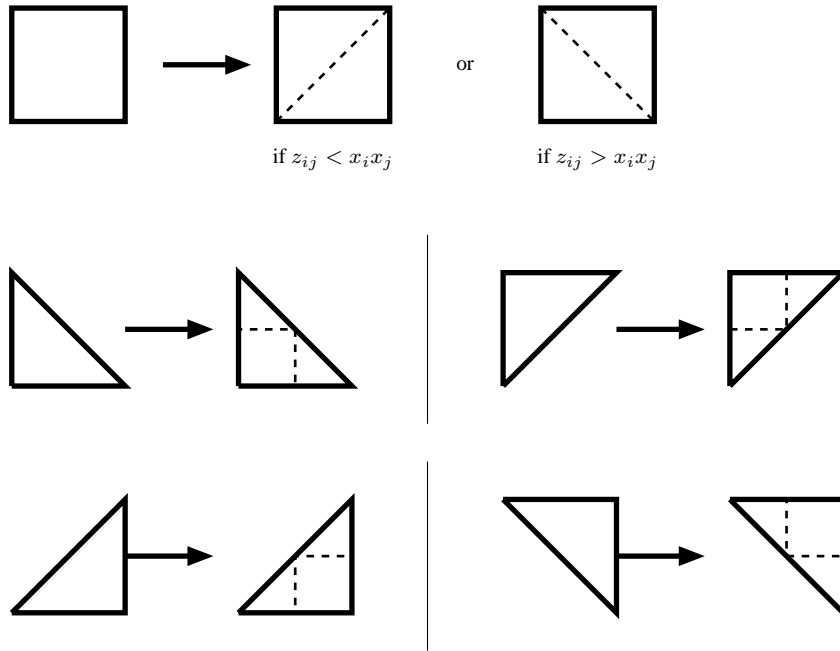**Fig. 7.** Sample Partitioning Schemes

Fig. 8. Example of Partitioning Scheme

In order to concisely write down a valid relaxation for an arbitrary direct product of partitions, let the sets

$$\mathcal{R} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in R_{x_i, x_j}\},$$

$$\mathcal{SE} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in SE_{x_i, x_j}\},$$

$$\mathcal{NW} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in NW_{x_i, x_j}\},$$

$$\mathcal{SW} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in SW_{x_i, x_j}\},$$

$$\mathcal{NE} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in NE_{x_i, x_j}\},$$

$$\mathcal{N} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in N_{x_i, x_j}\},$$

$$\mathcal{S} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in S_{x_i, x_j}\},$$

$$\mathcal{E} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in E_{x_i, x_j}\},$$

$$\mathcal{W} \stackrel{\text{def}}{=} \{(i,j) \in I \times I \mid (x_i, x_j) \in W_{x_i, x_j}\}$$

be the index sets of pairs of variables that are constrained to be in a region of each shape. Thus, the polyhedron

$$\mathcal{F} = \{x \in \mathbb{R}^n \mid (x_i, x_j) \in R_{x_i, x_j} \ \forall (i,j) \in \mathcal{R}, (x_i, x_j) \in SE_{x_i, x_j} \ \forall (i,j) \in \mathcal{SE},$$
$$(x_i, x_j) \in NW_{x_i, x_j} \ \forall (i,j) \in \mathcal{NW}, (x_i, x_j) \in SW_{x_i, x_j} \ \forall (i,j) \in \mathcal{SW},$$
$$(x_i, x_j) \in N_{x_i, x_j} \ \forall (i,j) \in \mathcal{N}, (x_i, x_j) \in S_{x_i, x_j} \ \forall (i,j) \in \mathcal{S},$$
$$(x_i, x_j) \in E_{x_i, x_j} \ \forall (i,j) \in \mathcal{E}, (x_i, x_j) \in W_{x_i, x_j} \ \forall (i,j) \in \mathcal{W}\}$$

is the set of points satisfying the inequalities defining a particular element of the partition (a node of the branch-and-bound tree).

A nonlinear programming relaxation $NLPR$ of QCQP where the pairs of terms (x_i, x_j) are constrained to lie in the region $\mathcal{F}$ is

$$v_{NLPR} = \min_{x,z} c^T x + Q_0 \bullet Z \qquad\qquad (NLPR)$$

subject to

$$
\begin{aligned}
c_k^T x + Q_k \bullet Z &\geq b_k & \forall k \in M, \\
z_{ij} - l_i x_j - l_j x_i + l_i l_j &\geq 0 & \forall (i,j) \in \mathcal{R}, \\
z_{ij} - u_i x_j - u_j x_i + u_i u_j &\geq 0 & \forall (i,j) \in \mathcal{R}, \\
z_{ij} - l_i x_j - u_j x_i + l_i u_j &\leq 0 & \forall (i,j) \in \mathcal{R}, \\
z_{ij} - u_i x_j - l_j x_i + u_i l_j &\leq 0 & \forall (i,j) \in \mathcal{R}, \\
z_{ij} - \text{vexxy}_{NW_{x_i,x_j}}(x_i, x_j) &\geq 0 & \forall (i,j) \in \mathcal{N} \cup \mathcal{W} \cup \mathcal{NW}, \\
z_{ij} - \text{cavxy}_{NE_{x_i,x_j}}(x_i, x_j) &\leq 0 & \forall (i,j) \in \mathcal{N} \cup \mathcal{E} \cup \mathcal{NE}, \\
z_{ij} - \text{vexxy}_{SE_{x_i,x_j}}(x_i, x_j) &\geq 0 & \forall (i,j) \in \mathcal{S} \cup \mathcal{E} \cup \mathcal{SE}, \\
z_{ij} - \text{cavxy}_{SW_{x_i,x_j}}(x_i, x_j) &\leq 0 & \forall (i,j) \in \mathcal{S} \cup \mathcal{W} \cup \mathcal{SW}, \\
x &\in \mathcal{F}. &
\end{aligned}
$$

4. Second-Order Cone Representations

In this section, we demonstrate that the nonlinear constraints in the relaxation $NLPR$ are representable as second-order cone constraints. A second-order cone constraint is a constraint of the form

$$\|Ax + b\|_2 \leq c^T x + d.$$

The second-order (or Lorentz) cone of order n is the set of points

$$\mathcal{K}_q^n = \left\{ x \in \Re^n \mid x_1 \geq \sqrt{\sum_{i=2}^n x_i^2} \right\}.$$

The set of points that satisfy a second-order cone constraint is a convex set (the set is the inverse image of the unit second-order cone under an affine mapping), and hence efficient and robust algorithms exist for the solution of problems containing second-order cone constraints [31,23]. Kim and Kojima [19] have also considered relaxations of QCQP consisting of second-order cone and semidefinite cone constraints. These relaxations are based on the idea of lift-and-project and are of a different flavor than those presented herein. The second-order cone relaxations presented here are more in the flavor of Tawarmalani and Sahinidis [32], who describe the convex envelope of a simple function of two variables over a rectangle as being characterized by the set of points obeying a semidefinite programming constraint.

The ability to represent inequalities (27)—(27) as second-order cone constraints more directly stems from the fact that they can be shown to be what is known as *hyperbolic constraints* of the form $w^2 \leq uv$, with $u \geq 0, v \geq 0$. Hyperbolic constraints are second-order cone constraints, which is evident from the relations

$$w^2 \leq uv, u \geq 0, v \geq 0 \Leftrightarrow \left\| \begin{bmatrix} 2w \\ u-v \end{bmatrix} \right\|_2 \leq u + v \Leftrightarrow \begin{bmatrix} u+v \\ 2w \\ u-v \end{bmatrix} \in \mathcal{K}_q^3 \qquad (27)$$

Theorems (7)–(10) show that each of the nonlinear constraints present in our nonlinear programming relaxation to QCQP can be represented as a second-order cone constraint.

Theorem 7. *Let* $(x, y) \in SE_{x,y}$ *and define*

$$w_{SE} = \frac{y - l_y}{u_y - l_y},$$

$$u_{SE} = \frac{z - l_y x - l_x y + l_x l_y}{(u_x - l_x)(u_y - l_y)}, and$$

$$v_{SE} = 1 - \frac{x - l_x}{u_x - l_x} + \frac{y - l_y}{u_y - l_y}.$$

Then

$$z \geq vex_{SE_{x,y}}(x, y) \Leftrightarrow \begin{bmatrix} u_{SE} + v_{SE} \\ 2w_{SE} \\ u_{SE} - v_{SE} \end{bmatrix} \in \mathcal{K}_q^3.$$

Proof. The proof is a tedious exercise in algebra to show that

$$z \geq \frac{q_{SE}(x, y)}{l_{SE}(x, y)}$$

if and only if

$$\left(\frac{y - l_y}{u_y - l_y} \right)^2 \leq \left(\frac{z - l_y x - l_x y + l_x l_y}{(u_x - l_x)(u_y - l_y)} \right) \left(1 - \frac{x - l_x}{u_x - l_x} + \frac{y - l_y}{u_y - l_y} \right). \tag{28}$$

The proof is completed by recognizing the terms w_{SE}, u_{SE}, and v_{SE} in (28), noting that $(x, y) \in SE, z \leq xy \Rightarrow u_{SE} \geq 0, v_{SE} \geq 0$, and using the relations (27). $\qquad\square$

Theorem 8. *Let* $(x, y) \in NW_{x,y}$ *and define*

$$w_{NW} = \frac{x - l_x}{u_x - l_x},$$

$$u_{NW} = \frac{z - l_y x - l_x y + l_x l_y}{(u_x - l_x)(u_y - l_y)}, and$$

$$v_{NW} = 1 - \frac{y - l_y}{u_y - l_y} + \frac{x - l_x}{u_x - l_x}.$$

Then

$$z \geq vex_{NW}(xy) \Leftrightarrow \begin{bmatrix} u_{NW} + v_{NW} \\ 2w_{NW} \\ u_{NW} - v_{NW} \end{bmatrix} \in \mathcal{K}_q^3.$$

Proof. The proof is a nearly identical exercise in algebra to that found in the proof of Theorem 7. $\qquad\square$

Theorem 9. *Let* $(x, y) \in SW_{x,y}$ *and define the variables*

$$w_{SW} = \frac{x - l_x}{u_x - l_x}, \tag{29}$$

$$u_{SW} = \frac{u_y x + l_x y - l_x u_y - z}{(u_x - l_x)(u_y - l_y)}, \tag{30}$$

$$v_{SW} = \frac{x - l_x}{u_x - l_x} + \frac{y - l_y}{u_y - l_y}. \tag{31}$$

Then

$$z \leq cav_{SW}(xy) \Leftrightarrow \begin{bmatrix} u_{SW} + v_{SW} \\ 2w_{SW} \\ u_{SW} - v_{SW} \end{bmatrix} \in \mathcal{K}_q^3.$$

Proof. The proof begins by showing that

$$z \leq \frac{q_{SW}(x,y)}{l_{SW}(x,y)}$$

if and only if

$$\left(\frac{x - l_x}{u_x - l_x} \right)^2 \leq \left(\frac{u_y x + l_x y - l_x u_y - z}{(u_x - l_x)(u_y - l_y)} \right) \left(\frac{x - l_x}{u_x - l_x} + \frac{y - l_y}{u_y - l_y} \right) \tag{32}$$

Substituting (29), (30), and (31) into (32), noting that $(x,y) \in SW, z \geq xy \Rightarrow u_{SW} \geq 0$ and $v_{SW} \geq 0$, and using the relations (27) completes the proof. □

Theorem 10. *Let $(x,y) \in NE_{x,y}$ and define the variables*

$$w_{NE} \stackrel{\text{def}}{=} 1 - \frac{y - l_y}{u_y - l_y},$$

$$u_{NE} \stackrel{\text{def}}{=} \frac{u_y x + l_x y - l_x u_y - z}{(u_x - l_x)(u_y - l_y)},$$

$$v_{NE} \stackrel{\text{def}}{=} 2 - \frac{x - l_x}{u_x - l_x} - \frac{y - l_y}{u_y - l_y}.$$

Then

$$z \leq cav_{NE}(xy) \Leftrightarrow \begin{bmatrix} u_{NE} + v_{NE} \\ 2w_{NE} \\ u_{NE} - v_{NE} \end{bmatrix} \in \mathcal{K}_q^3.$$

Proof. The proof is a nearly identical exercise in algebra to that of Theorem 9. □

4.1. A Linear Relaxation

The convex and concave envelope inequalities over the triangular regions in the relaxation $NLPR$ to QCQP are nonlinear. Nonlinear programs (even if posed as second-order cone programs) are more difficult to solve and in general pose more numerical difficulties that linear programs. In branch-and-bound, portions of the search space are fathomed based on bounds provided by the solution of relaxed problems, so relaxations that are numerically unstable pose considerable difficulties. In general, when bounding procedures are based on calculations using floating point arithmetic, mathematically rigorous guarantee of global optimality can only be ensured using interval methods [18,14], but the problem is only exacerbated if the solvers for the relaxations are not numerically robust or reliable. Due to the huge commercial success of linear programming, powerful, numerically robust tools can be used for its solution, so it is useful to have an outerapproximation to the convex and concave envelope constraints of $NLPR$ that can be solved using linear programming. The relaxation given here is in the same spirit of Ben-Tal and Nemirovski [6] who give a polyhedral approximation to the second-order cone and in the spirit of the BARON (v6.0) package [33,34] whose default behavior is to use polyhedral outer-approximations of nonlinear relaxations through a variant of the sandwich algorithm [26].

Take for example the triangular region $SE_{x,y}$, (as development for the other regions is similar). In Theorem 3, it was shown that $\text{vexxy}_{SE_{x,y}}(x, y)$ was the line segment through the points (u_x, l_y) and (x, y). This line segment is labeled L_1 in Figure 3 and intersects the line segment L_2 defining the boundary $SE_{x,y}$. Therefore, appealing to the dual (or convex hull) representation of the convex envelope and to the geometry of the bilinear form xy, we can write $\text{vexxy}_{SE_{x,y}}(x, y)$ as the solution to the following semi-infinite optimization problem:

$$\text{vexxy}_{SE_{x,y}}(x, y) = \min_{\xi, \lambda} \sum_{(\hat{x}_i, \hat{y}_i \in L_2)} \hat{x}_i \hat{y}_i \xi_i + u_x l_y \lambda$$

subject to

$$\sum_{(\hat{x}_i, \hat{y}_i) \in L_2} \hat{x}_i \xi_i + u_x \lambda = x,$$

$$\sum_{(\hat{x}_i, \hat{y}_i) \in L_2} \hat{y}_i \xi_i + l_y \lambda = y,$$

$$\sum_{(\hat{x}_i, \hat{y}_i) \in L_2} \xi_i + \lambda = 1,$$

$$\lambda \geq 0,$$

$$\xi_i \geq 0 \qquad \forall i \mid (\hat{x}_i, \hat{y}_i) \in L_2.$$

We outerapproximate this semi-infinite program through a discretization procedure, underestimating the objective function coefficient for ξ_i at certain points on L_2 using the subgradient inequality. For simplicity, assume that all points are equally spaced. Then a polyhedral outerapproximation to $\text{vexxy}_{SE_{x,y}}$ can be written as follows.

Let

$$\hat{x}_k = l_x + \frac{k(u_x - l_x)}{2K + 2} \qquad k = 0, 1, \ldots 2K + 2, \tag{33}$$

$$\hat{y}_k = l_y + \frac{k(u_y - l_y)}{2K + 2} \qquad k = 0, 1, \ldots 2K + 2, \tag{34}$$

$$\hat{z}_k = \begin{cases} \hat{x}_k \hat{y}_k & \text{if } k \text{ is even} \\ \hat{x}_{k-1} \hat{y}_{k-1} + \frac{1}{2K+2}(\hat{y}_{k-1}(u_x - l_x) + \hat{x}_{k-1}(u_y - l_y)) & \text{if } k \text{ is odd.} \end{cases} \tag{35}$$

The outerapproximation is accomplished by associating with each of the points $k = 0, 1, \ldots, 2K + 2$ a convex multiplier variable ξ_k, and writing (x, y, z) as appropriate combinations of these multipliers:

$$x = u_x \lambda_2 + \sum_{k=0}^{2K+2} \hat{x}_k \xi_k, \tag{36}$$

$$y = l_y \lambda_2 + \sum_{k=0}^{2K+2} \hat{y}_k \xi_k, \tag{37}$$

$$z \geq u_x l_y \lambda_2 + \sum_{k=0}^{2K+2} \hat{z}_k \xi_k. \tag{38}$$

With these definitions, a polyhedral set that contains the epigraph of vexxy $_{SE_{x,y}}$ can be written as

$$
\begin{aligned}
PSE_{x,y}(K) = \{(x,y,z) \in \mathbb{R}^3 \mid & (x,y) \in R_{x,y}, \\
& (\hat{x}, \hat{y}, \hat{z}) \text{ obey } (33),(34),(35), \\
& (x,y,z) \text{ obey } (36),(37),(38), \\
& (\lambda_2, \xi) \in \Delta^{2K+4} \}.
\end{aligned}
$$

A similar outerapproximating polyhedra can be defined for $NW_{x,y}$ triangular regions by writing (x,y,z) in a similar fashion:

$$
x = l_x \lambda_4 + \sum_{k=0}^{2K+2} \hat{x}_k \xi_k, \tag{39}
$$

$$
y = u_y \lambda_4 + \sum_{k=0}^{2K+2} \hat{y}_k \xi_k, \tag{40}
$$

$$
z \geq l_x u_y \lambda_4 + \sum_{k=0}^{2K+2} \hat{z}_k \xi_k. \tag{41}
$$

$$
\begin{aligned}
PNW_{x,y}(K) = \{(x,y,z) \in \mathbb{R}^3 \mid & (x,y) \in R_{x,y}, \\
& (\hat{x}, \hat{y}, \hat{z}) \text{ obey } (33),(34),(35), \\
& (x,y,z) \text{ obey } (39),(40),(41), \\
& (\lambda_4, \xi) \in \Delta^{2K+4} \}.
\end{aligned}
$$

For the regions $SW_{x,y}$ and $NE_{x,y}$ a similar discretization procedure can be applied. In these cases, the discretization points are

$$
\hat{x}_k = l_x + \frac{k(u_x - l_x)}{2K+2} \qquad k = 0,1,\ldots 2K+2, \tag{42}
$$

$$
\hat{y}_k = u_y + \frac{k(l_y - u_y)}{2K+2} \qquad k = 0,1,\ldots 2K+2, \tag{43}
$$

$$
\hat{z}_k = \begin{cases} \hat{x}_k \hat{y}_k & \text{if } k \text{ is even} \\ \hat{x}_{k-1}\hat{y}_{k-1} + \frac{1}{2K+2}(\hat{y}_{k-1}(u_x - l_x) + \hat{x}_{k-1}(l_y - u_y)) & \text{if } k \text{ is odd.} \end{cases} \tag{44}
$$

As above, associate with each of the points $k = 0,1,\ldots,2K+2$ a convex multiplier variable μ_k, defining points and function values to obey

$$
x = l_x \lambda_1 + \sum_{k=0}^{2K+2} \hat{x}_k \mu_k, \tag{45}
$$

$$
y = l_y \lambda_1 + \sum_{k=0}^{2K+2} \hat{y}_k \mu_k, \tag{46}
$$

$$
z \leq l_x l_y \lambda_1 + \sum_{k=0}^{2K+2} \hat{z}_k \mu_k. \tag{47}
$$

$$PSW_{x,y}(K) = \{(x,y,z) \in \mathbb{R}^3 \mid (x,y) \in R_{x,y},$$
$$(\hat{x}, \hat{y}, \hat{z}) \text{ obey } (42), (43), (44),$$
$$(x,y,z) \text{ obey } (45), (46), (47),$$
$$(\lambda_1, \mu) \in \Delta^{2K+4}\}.$$

Construction of a polyhedral set containing the hypograph of cavxy $_{NE_{x,y}}$ is similar.

$$x = u_x \lambda_3 + \sum_{k=0}^{2K+2} \hat{x}_k \mu_k, \tag{48}$$

$$y = u_y \lambda_3 + \sum_{k=0}^{2K+2} \hat{y}_k \mu_k, \tag{49}$$

$$z \leq u_x u_y \lambda_3 + \sum_{k=0}^{2K+2} \hat{z}_k \mu_k. \tag{50}$$

$$PNE_{x,y}(K) = \{(x,y,z) \in \mathbb{R}^3 \mid (x,y) \in R_{x,y},$$
$$(\hat{x}, \hat{y}, \hat{z}) \text{ obey } (42), (43), (44),$$
$$(x,y,z) \text{ obey } (48), (49), (50),$$
$$(\lambda_3, \mu) \in \Delta^{2K+4}\}.$$

Make one final definition of $PR_{x,y}$ as

$$PR_{x,y} = \{(x,y,z) \in \mathbb{R}^3 \mid x = (\lambda_1 + \lambda_4) l_x + (\lambda_2 + \lambda_3) u_x$$
$$y = (\lambda_1 + \lambda_2) l_y + (\lambda_3 + \lambda_4) u_y$$
$$z = l_x l_y \lambda_1 + u_x l_y \lambda_2 + u_x u_y \lambda_3 + l_x u_y \lambda_4$$
$$\lambda \in \Delta^4\}.$$

Using the linear outerapproximations of all nonlinear constraints, a linear programming relaxation LPR_3 to QCQP can be written:

$$\min \quad c^T x + Q_0 \bullet Z \tag{LPR_3}$$

subject to

$$c_k^T x + Q_k \bullet Z \geq b_k \qquad \forall k \in M,$$
$$x_i \in [l_i, u_i] \qquad \forall i \in I,$$
$$(x_i, x_j, z_{ij}) \in PR_{x_i, x_j} \qquad \forall (i,j) \in \mathcal{R},$$
$$(x_i, x_j, z_{ij}) \in PSE_{x_i, x_j} \qquad \forall (i,j) \in \mathcal{SE} \cup \mathcal{S} \cup \mathcal{E},$$
$$(x_i, x_j, z_{ij}) \in PNW_{x_i, x_j} \qquad \forall (i,j) \in \mathcal{NW} \cup \mathcal{N} \cup \mathcal{W},$$
$$(x_i, x_j, z_{ij}) \in PSW_{x_i, x_j} \qquad \forall (i,j) \in \mathcal{SW} \cup \mathcal{S} \cup \mathcal{W},$$
$$(x_i, x_j, z_{ij}) \in PNE_{x_i, x_j} \qquad \forall (i,j) \in \mathcal{NE} \cup \mathcal{N} \cup \mathcal{E}.$$

5. Strength of Envelope Expressions

In this section we quantify the strength of convex and concave envelope expressions for the bilinear expression xy. In some sense, one cannot hope to obtain a tighter (tractable) relaxation for xy than by using the convex and concave envelopes. Still, it may be that certain regions have tighter relaxations than others, and the strength of the relaxation can be taken into account when designing partitioning rules for an algorithm, a few examples of which are presented in Section 3. To quantify the strength of the convex and concave envelopes expressions, there are a number of useful measures that can be defined. For example, over a region $\Gamma \subset \mathbb{R}^2$, the total lower error (η_L), total upper error (η_U), and total error (η) of a convex/concave envelope approximation to xy can be defined as

$$\eta_L(\Gamma) \stackrel{\text{def}}{=} \int_\Gamma (xy - \text{vexxy}_\Gamma(x,y)) dx\, dy,$$

$$\eta_U(\Gamma) \stackrel{\text{def}}{=} \int_\Gamma (\text{cavxy}_\Gamma(x,y) - xy) dx\, dy,$$

$$\eta(\Gamma) \stackrel{\text{def}}{=} \int_\Gamma (\text{cavxy}_\Gamma(x,y) - \text{vexxy}_\Gamma(x,y)) dx\, dy.$$
$$= \eta_L(\Gamma) + \eta_U(\Gamma).$$

Using the functional expressions for the convex and concave envelopes (1) and (2), it is an exercise in integration to compute the η error measures for the rectangular region $R_{x,y}$. To ease notation, we define the symbolic constants

$$\alpha \stackrel{\text{def}}{=} u_x^2 u_y^2 + u_x^2 l_y^2 + u_y^2 l_x^2 + l_x^2 l_y^2,$$

$$\beta \stackrel{\text{def}}{=} u_x^2 u_y l_y + u_x u_y^2 l_y + u_x l_x l_y^2 + u_y l_x^2 l_y, \text{ and}$$

$$\gamma \stackrel{\text{def}}{=} u_x u_y l_x l_y.$$

The formula for error measure for $R_{x,y}$ is

$$\eta(R_{x,y}) = \alpha/6 - \beta/3 + 2\gamma/3.$$

By symmetry, it is not surprising that

$$\eta_L(R_{x,y}) = \eta_U(R_{x,y}) = \eta(R_{x,y})/2.$$

In a similar fashion, the error integrals for the triangular regions $SE_{x,y}$, $NW_{x,y}$, $SW_{x,y}$, and $NE_{x,y}$ can be computed to be

$$\eta_L(SW_{x,y}) = \alpha/24 - \beta/12 + \gamma/6,$$

$$\eta_L(NE_{x,y}) = \eta_L(SW_{x,y}),$$

$$\eta_L(NW_{x,y}) = \alpha/72 - \beta/36 + \gamma/18,$$

$$\eta_L(SE_{x,y}) = \eta_L(NW_{x,y}),$$

$$\eta_U(NW_{x,y}) = \alpha/24 - \beta/12 + \gamma/6,$$

$$\eta_U(SE_{x,y}) = \eta_U(NW_{x,y}),$$

$$\eta_U(SW_{x,y}) = \alpha/72 - \beta/36 + \gamma/18,$$

$$\eta_U(NE_{x,y}) = \eta_U(SW_{x,y}).$$

Note that for the triangular regions $NW_{x,y}$ and $SE_{x,y}$, the lower envelope approximation is better than the upper envelope approximation. For the triangular regions $SW_{x,y}$ and $NE_{x,y}$, the upper envelope approximation is better than the lower envelope. This asymmetry is not present for the triangular regions $N_{x,y}$, $S_{x,y}$, $E_{x,y}$, and $W_{x,y}$, as evidenced by the computed formulae for the error integrals:

$$\eta(N_{x,y}) = \alpha/72 - \beta/36 + \gamma/18,$$
$$\eta(S_{x,y}) = \eta(E_{x,y}) = \eta(W_{x,y}).$$

Measuring the maximum approximation error is also useful, and similar error expressions for the maximum lower error (ϕ_L), maximum upper error (ϕ_U), and maximum total error (ϕ) can be defined:

$$\phi_L(\Gamma) \overset{\text{def}}{=} \max_{\Gamma}(xy - \text{vexxy}_\Gamma(x, y)),$$
$$\phi_U(\Gamma) \overset{\text{def}}{=} \max_{\Gamma}(\text{cavxy}_\Gamma(x, y) - xy),$$
$$\phi(\Gamma) \overset{\text{def}}{=} \text{cavxy}_\Gamma(x, y) - \text{vexxy}_\Gamma(x, y).$$

Note that it is *not* in general true that $\phi_L(\Gamma) + \phi_U(\Gamma) = \phi(\Gamma)$ as is the case for the η measure of error. It may also be of interest to know the point at which the maximum error is achieved, and to that end define

$$(x_L^\phi(\Gamma), y_L^\phi(\Gamma)) \overset{\text{def}}{=} \arg\max_{\Gamma}(xy - \text{vexxy}_\Gamma(x, y)),$$
$$(x_U^\phi(\Gamma), y_U^\phi(\Gamma)) \overset{\text{def}}{=} \arg\max_{\Gamma}(\text{cavxy}_\Gamma(x, y) - xy),$$
$$(x^\phi(\Gamma), y^\phi(\Gamma)) \overset{\text{def}}{=} \arg\max_{\Gamma}(\text{cavxy}_\Gamma(x, y) - \text{vexxy}_\Gamma(x, y)).$$

In the rectangular case, the point with the largest error is in the middle of the region $R_{x,y}$:

$$(x_L^\phi(\Gamma), y_L^\phi(\Gamma)) = (x_U^\phi(\Gamma), y_U^\phi(\Gamma)) = (x^\phi(\Gamma), y^\phi(\Gamma)) = \left(\frac{1}{2}(l_x + u_x), \frac{1}{2}(l_y + u_y)\right).$$

So the max error measures for the convex and concave envelope over $R_{x,y}$ are

$$\phi_L(R_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y)$$
$$\phi_U(R_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y)$$
$$\phi(R_{x,y}) = \frac{1}{2}(u_x u_y - u_x l_y - l_x u_y + l_x l_y)$$

Computing ϕ for the triangular regions is slightly more complicated, but can be accomplished by examining the optimality conditions for the optimization problem present in the definition of ϕ. Performing the analysis

yields

$$\left(x_L^{\phi}(SE_{x,y}), y_L^{\phi}(SE_{x,y})\right) = \left(\frac{3}{4}u_x + \frac{1}{4}l_x, \frac{3}{4}l_y + \frac{1}{4}u_y\right),$$

$$\left(x_U^{\phi}(SE_{x,y}), y_U^{\phi}(SE_{x,y})\right) = \left(\frac{1}{2}(l_x + u_x), \frac{1}{2}(l_y + u_y)\right),$$

$$\left(x^{\phi}(SE_{x,y}), y^{\phi}(SE_{x,y})\right) = \left(\frac{1}{2}(l_x + u_x), \frac{1}{2}(l_y + u_y)\right),$$

$$\phi_L(SE_{x,y}) = \frac{1}{16}(u_x u_y - u_x l_y - l_x u_y + l_x l_y),$$

$$\phi_U(SE_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y),$$

$$\phi(SE_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y).$$

Not surprisingly (by symmetry), the error measures for similar shaped triangles are the same. They are included in the Appendix.

For the triangles $S_{x,y}$, $N_{x,y}$, $E_{x,y}$, and $W_{x,y}$ a similar analysis can be performed to show that

$$\left(x_L^{\phi}(S_{x,y}), y_L^{\phi}(S_{x,y})\right) = \left(x_U^{\phi}(S_{x,y}), y_U^{\phi}(S_{x,y})\right) = \left(x^{\phi}(S_{x,y}), y^{\phi}(S_{x,y})\right) =$$

$$\left(\frac{1}{2}(l_x + u_x), \frac{1}{2}l_y + \frac{\sqrt{2}-1}{2}u_y\right),$$

$$\phi(S_{x,y}) = \left(\frac{3}{2} - \sqrt{2}\right)(u_x u_y + l_x l_y - l_x u_y - u_x l_y),$$

and

$$\phi(S_{x,y}) = \phi(N_{x,y}) = \phi(E_{x,y}) = \phi(W_{x,y}).$$

(The points at which the maximum error are obtained for regions besides $S_{x,y}$ are listed in the Appendix.)

Example 3. Consider a region $\hat{R} = \{(x, y) \in [0, 2] \times [0, 2]\}$ that we divide into four subrectangles:

$$\widehat{SW} = \{(x, y) \in [0, 1] \times [0, 1]\},$$
$$\widehat{NW} = \{(x, y) \in [0, 1] \times [1, 2]\},$$
$$\widehat{NE} = \{(x, y) \in [1, 2] \times [1, 2]\},$$
$$\widehat{SE} = \{(x, y) \in [1, 2] \times [0, 1]\}.$$

The difference between concave and convex envelopes over these regions are

$$\eta(\widehat{SW}) = \eta(\widehat{NW}) = \eta(\widehat{NE}) = \eta(\widehat{SE}) = \frac{1}{6}.$$

If the same region $\hat{R}$ is subdivided into the four triangles

$$\hat{N} = \hat{R} \cap \{(x, y) \mid x + y \geq 2, y \geq x\},$$
$$\hat{S} = \hat{R} \cap \{(x, y) \mid x + y \leq 2, y \leq x\},$$
$$\hat{E} = \hat{R} \cap \{(x, y) \mid x + y \leq 2, y \geq x\},$$
$$\hat{W} = \hat{R} \cap \{(x, y) \mid x + y \geq 2, y \leq x\},$$

The total error in the resulting relaxations of xy are

$$\eta(\hat{N}) = \eta(\hat{S}) = \eta(\hat{E}) = \eta(\hat{W}) = \frac{2}{9}.$$

Thus we are led to the somewhat surprising conclusion that for this particular branching scheme, measure by the total error measure η, it is better to subdivide the region into rectangles. The story is different if the maximum error measure ϕ is used. In this case,

$$\phi(\widehat{SW}) = \phi(\widehat{NW}) = \phi(\widehat{NE}) = \phi(\widehat{SE}) = \frac{1}{2},$$

and

$$\phi(\hat{N}) = \phi(\hat{S}) = \phi(\hat{E}) = \phi(\hat{W}) = 6 - 4\sqrt{2} \approx 0.343.$$

So in terms of the maximum error, it is better to partition the region into triangles.

Example 4. Suppose we wish to decide whether to partition the region $\hat{R} = \{(x, y) \in [0, 2] \times [0, 2]\}$ into two regions

$$\hat{R}_1 = \{(x, y) \in [0, 1] \times [0, 2]\}$$
$$\hat{R}_2 = \{(x, y) \in [1, 2] \times [0, 2]\}$$

or into the two regions

$$\widehat{SE} = \hat{R} \cap \{(x, y) \mid y \leq x\}$$
$$\widehat{NW} = \hat{R} \cap \{(x, y) \mid y \geq x\}.$$

The total error measures are

$$\eta_L(\hat{R}_1) = \eta_L(\hat{R}_2) = \eta_U(\hat{R}_1) = \eta_U(\hat{R}_2) = \frac{1}{3},$$
$$\eta(\hat{R}_1) = \eta(\hat{R}_2) = \frac{2}{3},$$
$$\eta_L(\widehat{SE}) = \eta_L(\widehat{NW}) = \frac{2}{9},$$
$$\eta_U(\widehat{SE}) = \eta_U(\widehat{NW}) = \frac{2}{3},$$
$$\eta(\widehat{SE}) = \eta(\widehat{NW}) = \frac{8}{9}.$$

The maximum error measures are

$$\phi_L(\hat{R}_1) = \phi_L(\hat{R}_2) = \phi_U(\hat{R}_1) = \phi_U(\hat{R}_2) = \frac{1}{2},$$
$$\phi(\hat{R}_1) = \phi(\hat{R}_2) = 1,$$
$$\phi_L(\widehat{SE}) = \phi_L(\widehat{NW}) = \frac{1}{4},$$
$$\phi_U(\widehat{SE}) = \phi_U(\widehat{NW}) = 1,$$
$$\phi(\widehat{SE}) = \phi(\widehat{NW}) = 1.$$

Since $\eta_L(\widehat{SE}) = \eta_L(\widehat{NW}) < \eta_L(\hat{R}_1) = \eta_L(\hat{R}_2)$, what example 4 demonstrates is that the triangle-based branching scheme improves the lower approximation more than a rectangle-based scheme. In general, by branching into triangles, one can specifically improve one of the lower or over approximations, depending on where the relaxed solution z_{ij}^* lies in relation to the desired value $x_i^* x_j^*$. This can be used to advantage in designing a branch-and-bound algorithm. For example, this is precisely the manner in which we partition the rectangular regions into triangular regions for the computational results presented in Section 6.

6. Computational Results

The purpose of this section is to demonstrate that branch-and-bound algorithms for solving QCQP can benefit from a triangular partitioning scheme, and thus the convex and concave envelope expressions derived in this paper. The utility of a triangular branching scheme is demonstrated by comparing the solution times and number of branch-and-bound nodes for two and four rectangle partitioning schemes with a simple triangle-based scheme on a fixed set of QCQP test instances. Most of the test instances come from the GAMS GlobalLib [12]. Versions of these instance in the AMPL modeling language are available from the COCONUT Benchmark [7]. The test suite has been augmented with problems 4 and 7 from Audet et al. [5], and one instance (ex_7_3_3) has been altered by added artificial upper and lower bounds on all of the variables. The instances were chosen to contain a significant number of bilinear terms (in comparison to squared terms), since a triangular partitioning only has any real impact in the case of quadratics with bilinear terms. Table 1 shows some characteristics of our test instances, where n is the number of variables, m is the number of constraints, s_o is the number of squared terms in the objective function, b_o is the number of bilinear terms in the objective function, s_c is the number of squared terms in the constraints, and b_c is the number of bilinear terms in the constraints.

Table 1. Characteristics of Test Instances

name	n	m	s_o	b_o	s_c	b_c
audet4	6	4	0	3	4	3
audet7	16	23	0	5	5	37
ex2_1_9	10	1	0	22	0	0
ex3_1_1	8	6	0	0	0	5
ex3_1_2	5	6	1	1	2	16
ex3_1_4	3	3	0	0	3	3
ex5_2_2_case1	9	6	0	0	0	4
ex5_2_2_case2	9	6	0	0	0	4
ex5_2_2_case3	9	6	0	0	0	4
ex5_3_2	22	16	0	0	0	12
ex5_4_2	8	6	0	0	0	5
ex5_4_3	16	13	0	0	0	8
ex5_4_4	27	19	0	0	0	15
ex7_3_3-bounded	5	8	0	0	1	3
himmel11	9	3	1	1	1	8
himmel16	15	21	0	0	45	23
prolog-bounded	20	22	0	4	0	4

The relaxations and partitioning schemes have been implemented in a code called QPBB consisting of roughly 10,000 lines of C++. Linear programming relaxations in QPBB are created and solved through the COIN-OR `OsiSolverInterface`, and as such can use a number of different linear programming solvers [8]. We use the `Clp` linear programming toolkit (also from COIN-OR) to produce the computational results in this paper. Nonlinear relaxations can be modeled using the NLPAPI available in COIN-OR, which has interfaces to both the nonlinear programming solvers Lancelot [9] or IPOPT [35]. However, in this study, the nonlinear programming software is not used, instead we use the linear programming outerapproximation to the nonlinear constraints developed in Section 4.1. QPBB has an interface to the AMPL modeling language. The QPBB branch-and-bound code is implemented as an instantiation of the virtual class library MW [13], and therefore QPBB is capable of running in parallel on a loose federation of processors known as a

computational grid [11]. In this study QPBB has been compiled with the Gnu g++ compiler and configured to run (using the MW Independent mode) on a single Intel Pentium(R) CPU with a clock speed of 2.40GHz. QPBB solves instances to approximate global optimality, where approximate is defined using the following two definitions.

Definition 1. *A solution $(\bar{x}, \bar{z})$ to a relaxation of QCQP is said to be ϵ_f-feasible for QCQP if and only if*

- $l_i \leq \bar{x}_i \leq u_i$, $\forall i \in I$,
- $c_k^T \bar{x} + Q_k \bullet \bar{Z} \geq b_k$, $\forall k \in M$, *and*
- $|\bar{z}_{ij} - \bar{x}_i \bar{x}_j| \leq \epsilon_f$ $\forall i \in I$ $\forall j \in I$.

Definition 2. *A solution $(\bar{x}, \bar{z})$, with relaxation objective value v^* is said to be $(\epsilon_f - \epsilon_v)$ optimal if and only if $(\bar{x}, \bar{z})$ is ϵ_f-feasible and there does not exist another ϵ_f-feasible solution whose relaxation objective value $\hat{v}$ satisfies $|v^* - \hat{v}| \geq \epsilon_v$.*

In our experiments, we attempt to solve all instances to the precision $\epsilon_f = \epsilon_v = 10^{-6}$. We assume that the relaxations are solved exactly, though this assumption could be relaxed by obtaining rigorous lower bounds in a manner suggested by Jansson [17]. Since the purpose of this experiment is solely to test the usefulness of a triangular partitioning scheme, no advanced branching or range reduction techniques are employed by QPBB. Given a solution to the relaxation $(\hat{x}, \hat{z})$, we branch on the pair of variables x_{i^*}, x_{j^*} satisfying

$$(i^*, j^*) = \arg \max_{i,j \in I} \{|\hat{z}_{ij} - \hat{x}_i \hat{x}_j|\}.$$

For this experiment, QPBB had been configured to evaluate the node of the branch and bound tree whose parent has the smallest relaxation value (best bound node selection).

The triangular branching scheme tested was the scheme depicted pictorially in Figure 8. Namely,

- Rectangular regions R_{x_i, x_j} were partitioned into two regions SE_{x_i, x_j} and NW_{x_i, x_j} (like in Figure 7(b)) if the lower envelope approximation needed to be improved more than the upper. Conversely a rectangular region R_{x_i, x_j} was partitioned into two regions SW_{x_i, x_j} and NE_{x_i, x_j} if the upper envelope approximation needed to be improved more than the lower (measured in terms of the absolute error of the approximation).
- Triangular regions were partitioned into two triangular regions and a rectangular region in a manner depicted by Figure 7(c).

This triangular partitioning scheme was compared against a two-rectangle partitioning scheme, where the rectangle was divided into two subrectangles by bisecting the longest edge and a four-rectangle scheme obtained by bisecting the original rectangle along both edges. The results of the computational experiment are given in Table 2. Figure 9 shows a performance profile (see [10] for an explanation of performance profiles) of the number of nodes of the branch-and-bound tree, and Figure 10 shows a profile of the CPU time used by each of the three partitioning methods. The experimental results show that the four rectangle-based scheme and the triangle-based scheme exhibit fairly similar performance, and both of these methods dominate the two triangle-based scheme. For the instances requiring less than 3 CPU seconds, the rectangular schemes are clearly better, but as the difficulty of the problem instance increases, the triangule-based scheme seems to outperform both the two and four rectangle-based schemes. Future work will aim on testing different triangle-based partitioning schemes and on combining the schemes with more advanced techniques for branch selection, node selection, and problem preprocessing.

Table 2. Results of Branch-and-Bound for Various Partitioning Schemes

	Two Rectangles		Four Rectangles		Triangles and Rectangles	
Name	Nodes	Time(s)	Nodes	Time(s)	Nodes	Time(s)
audet4	410	0.88	372	0.79	286	0.75
audet7	14655	112.3700	4119	30.6200	1886	20.6700
ex2_1_9	3527	12.84	3574	12.92	821	6.15
ex3_1_1	27370	48.41	50236	87.96	8810	28.92
ex3_1_2	57	0.10	53	0.11	55	0.14
ex3_1_4	43	0.06	43	0.06	47	0.08
ex5_2_2_case1	225	0.25	125	0.15	138	0.18
ex5_2_2_case2	420	0.47	246	0.28	299	0.38
ex5_2_2_case3	299	0.32	141	0.16	147	0.18
ex5_3_2	2099	5.45	497	1.04	465	1.47
ex5_4_2	3817	6.13	1844	2.80	1884	3.59
ex5_4_3	22	0.03	6	0.01	13	0.03
ex5_4_4	100	0.28	59	0.17	247	1.19
ex7_3_3-bounded	86	0.11	83	0.10	209	0.30
himmel11	57	0.10	54	0.10	56	0.14
himmel16	228	1.11	230	1.15	327	2.05
prolog-bounded	5515	11.68	1194	2.47	962	2.62
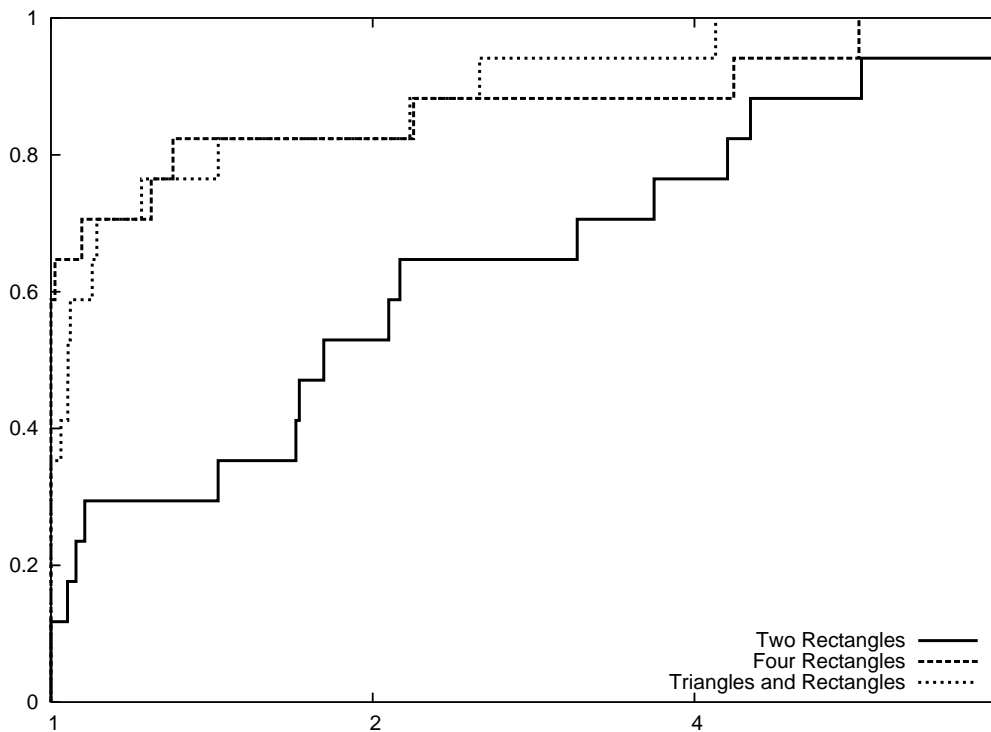
Fig. 9. Performance Profile Comparing Number of Branch-and-Bound Nodes for Three Partitioning Methods
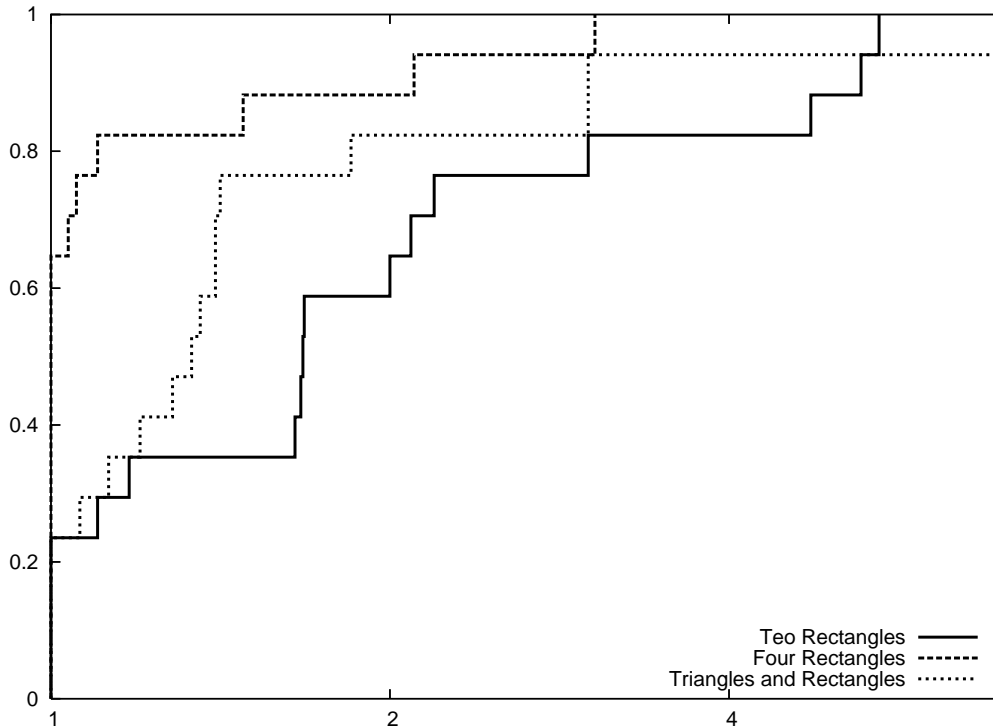
Fig. 10. Performance Profile Comparing CPU Time for Three Partitioning Methods

7. Conclusions and Future Directions

In this paper, nonlinear expressions for the convex and concave envelopes of a bilinear function $f(x, y) = xy$ over various triangular shapes were derived. These expressions were used to create a second-order cone programming relaxation of the quadratically constrained quadratic program (QCQP), and a polyhedral outerapproximation of the second-order cones was described, yielding a new linear programming relaxation to QCQP. Two measures of the tightness of convex and concave envelope approximations were introduced, and we showed that the envelopes over triangular regions are often demonstrably tighter than their rectangular counterparts under these measures. The formulae were embedded into a branch-and-bound algorithm for solving QPBB and shown to often reduce the computational effort required to solve instances.

The ultimate goal of this research is to create a solver capable of solving QCQP instances of larger magnitude than currently possible. There are a number of areas for further investigation required to make this goal possible. First, investigation of more sophisticated partitioning schemes that adaptively use the error measure formulae to decide on a proper partitioning is required. Partitioning rules that subdivide the region so that the solution to the relaxation is on the boundary of the new feasible regions will be explored. The QPBB solver has already been augmented with features to choose a better branching entity by solving auxiliary linear programs (akin to strong branching in mixed integer programming) and to tighten variable bounds at nodes of the branch-and-bound tree by solving auxiliary linear programs (strong preprocessing) [20, 21]. Since the QPBB code uses the MW software framework, ultimately it has been designed to run on a high-powered computational grid computing platform. The resulting code will be used to solve large-scale QCQP instances.

Appendix—Error Measure Formulae

The maximum errors for the triangular regions $NW_{x,y}$, $SW_{x,y}$, and $NE_{x,y}$ are the following:

$$\phi_L(NW_{x,y}) = \frac{1}{16}(u_x u_y + -u_x l_y - l_x u_y + l_x l_y),$$

$$\phi_U(NW_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y),$$

$$\phi(NW_{x,y}) = \frac{1}{4}(u_x u_y - u_x l_y - l_x u_y + l_x l_y),$$

$$\phi_U(SW_{x,y}) = \frac{1}{16}(u_x u_y + -u_x l_y - l_x u_y + l_x l_y)$$

$$\phi_L(SW_{x,y}) = \frac{1}{4}(u_x u_y + -u_x l_y - l_x u_y + l_x l_y)$$

$$\phi(SW_{x,y}) = \phi_L(SW_{x,y})$$

$$\phi_U(NE_{x,y}) = \phi_U(SW_{x,y})$$

$$\phi_L(NE_{x,y}) = \phi_U(SW_{x,y})$$

$$\phi(NE_{x,y}) = \phi(SW_{x,y}).$$

The points at which the maximum errors are achieved in the triangular regions $N_{x,y}$, $W_{x,y}$, $E_{x,y}$ are the following:

$$(x_L^\phi(N_{x,y}), y_L^\phi(N_{x,y})) = (x_U^\phi(N_{x,y}), y_U^\phi(N_{x,y})) = (x^\phi(N_{x,y}), y^\phi(N_{x,y})) =$$
$$\left(\frac{1}{2}(l_x + u_x), \frac{1}{2}l_y + \frac{3 - \sqrt{2}}{2}u_y \right),$$

$$(x_L^\phi(W_{x,y}), y_L^\phi(W_{x,y})) = (x_U^\phi(W_{x,y}), y_U^\phi(W_{x,y})) = (x^\phi(W_{x,y}), y^\phi(W_{x,y})) =$$
$$\left(\frac{1}{2}l_x + \frac{\sqrt{2} - 1}{2}u_x, \frac{1}{2}(l_y + u_y) \right),$$

$$(x_L^\phi(E_{x,y}), y_L^\phi(E_{x,y})) = (x_U^\phi(E_{x,y}), y_U^\phi(E_{x,y})) = (x^\phi(E_{x,y}), y^\phi(E_{x,y})) =$$
$$\left(\frac{1}{2}l_x + \frac{3 - \sqrt{2}}{2}u_x, \frac{1}{2}(l_y + u_y) \right).$$

Acknowledgement

References

1. F. A. Al-Khayyal. Generalized bilinear programming, part I: Models, applications, and linear programming relaxation. *European Journal on Operations Research*, 60:306–314, 1992.

2. F. A. Al-Khayyal and J. E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8:273–286, 1983.

3. F. A. Al-Khayyal, C. Larsen, and T. Van Voorhis. A relaxation method for nonconvex quadratically constrained programs. *Journal of Global Optimization*, 6:215–230, 1995.

4. I. P. Androulakis, C. D. Maranas, and C. A. Floudas. aBB : A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

5. C. Audet, P. Hansen, B. Jaumard, and G. Savard. A branch and cut algorithm for nonconvex quadratically constrained quadratic programs. *Mathematical Programming*, 87:131–152, 2000.

6. A. Ben-Tal and A. Nemirovski. On polyhedral approximations of the second-order cone. *Mathematics of Operations Research*, 26:193–205, 2001.

7. The COCONUT benchmark: A benchmark for global optimization and constraint satisfaction, 2004. `http://www.mat.univie.ac.at/~neum/glopt/coconut/benchmark.html`.

8. COIN-OR: Computational Infrastructure for Operations Research, 2004. `http://www.coin-or.org`.

9. A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization (Release A)*. Springer–Verlag, 1992.

10. Elizabeth Dolan and Jorge Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

11. I. Foster and C. Kesselman. Computational grids. In I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1999. Chapter 2.

12. Globallib, 2004. `http://www.gamsworld.org/global/globallib.htm`.

13. J.-P. Goux, S. Kulkarni, J. T. Linderoth, and M. Yoder. Master-Worker : An enabling framework for master-worker applications on the computational grid. *Cluster Computing*, 4:63–70, 2001.

14. P. Van Hentenryck, L. Michel, and Y. Deville. *Numerica. A Modeling Language for Global Optimization*. MIT Press, Cambridge, MA, 1997.

15. R. Horst. An algorithm for nonconvex programming problems. *Mathematical Programming*, 10:312–321, 1976.

16. Reiner Horst and Hoang Tuy. *Global Optimization*. Springer-Verlag, New York, 1993.

17. C. Jansson. Rigorous lower and upper bounds in linear programming. *SIAM Journal on Optimization*, 14:914–935, 2004.

18. R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer, Dordrecht, 1996.

19. S. Kim and M. Kojima. Second order cone programming relaxation methods of nonconvex quadratic optimization problems. *Optimization Methods and Software*, 15:201–224, 2001.

20. Y. Lebbeh, M. Rueher, and C. Michel. A global filtering algorithm for handling systes of quadratic equations and inequations. In P. van Hentenryck, editor, *Lecture Notes in Computer Science: Principles and Practice of Constraint Programming: CP 2002*, volume 2470, pages 109–123. Springer, 2002.

21. J. T. Linderoth. Applying integer programming techniques to global optimization problems, 2003. Presentation at INFORMS National Meeting.

22. G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

23. Mosek ApS, 2004. `www.mosek.com`.

24. U. Raber. A simplicial branch-and-bound method for solving nonconvex all-quadratic programs. *Journal of Global Optimization*, 13:417–432, 1998.

25. U. Raber. *Nonconvex All-Quadratic Global Optimization Problems: Solution Methods, Application and Related Topics*. PhD thesis, Universität Trier, Germany, 1999.

26. G. Rote. The convergence of the sandwich algorithm for approximating convex functions. *Computing*, 48:337–361, 1992.

27. H. S. Ryoo and N. V. Sahinidis. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8:107–139, 1996.

28. N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8:201–205, 1996.

29. H. D. Sherali and A. R. Alameddine. An explicit characterization of the convex envelope of a bivariate function over special polytopes. *Annals of Operations Research, Computational Methods in Global Optimization*, pages 197–210, 1990.

30. H. D. Sherali and A. R. Alameddine. A new reformulation linearization technique for bilinear programming problems. *Journal of Global Optimization*, 2:379–410, 1992.

31. J. F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12:625–653, 1999.

32. M. Tawarmalani and N. V. Sahinidis. Semidefinite relaxations of fractional programs via novel convexifications techniques. *Journal of Global Optimization*, 20:137–158, 2001.

33. M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer Academic Publishers, Boston MA, 2002.

34. M. Tawarmalani and N. V. Sahinidis. Global optimization of mixed integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 2004. to appear.

35. A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Research report, IBM T. J. Watson Research Center, Yorktown, USA, 2004.