RESEARCH ARTICLE

Dynamically-Generated Cutting Planes for Mixed-Integer Quadratically-Constrained Quadratic Programs and Their Incorporation into GloMIQO 2

Ruth Misener\textsuperscript{a,b}, James B. Smadbeck\textsuperscript{a}, and Christodoulos A. Floudas\textsuperscript{a,*}

\textsuperscript{a}Department of Chemical and Biological Engineering; Princeton University; Princeton, NJ 08544-5263; \textsuperscript{b}Department of Chemical Engineering; Imperial College London; South Kensington Campus, London, SW7 2AZ

(Received 00 Month 200x; in final form 00 Month 200x)

The Global Mixed-Integer Quadratic Optimizer, GloMIQO, addresses mixed-integer quadratically-constrained quadratic programs (MIQCQP) to $\varepsilon$-global optimality. This paper documents the branch-and-cut framework integrated into GloMIQO 2. Cutting planes are derived from reformulation-linearisation technique equations, convex multivariable terms, $\alpha$BB convexifications, and low- and high-dimensional edge-concave aggregations. Cuts are based on both individual equations and collections of nonlinear terms in MIQCQP. Novel contributions of this paper include: development of a corollary to Crama’s [35] necessary and sufficient condition for the existence of a cut dominating the termwise relaxation of a bilinear expression; algorithmic descriptions for deriving each class of cut; presentation of a branch-and-cut framework integrating the cuts. Computational results are presented along with comparison of the GloMIQO 2 performance to several state-of-the-art solvers.

Keywords: quadratically constrained quadratic programming, cutting planes, global optimisation

AMS Subject Classification: 90C26, 90C20, 90C57, 65K05

1. Introduction and Problem Definition

Recent work developed the Global Mixed-Integer Quadratic Optimizer, GloMIQO, a generic computational framework for addressing mixed-integer quadratically-constrained quadratic programs (MIQCQP) to $\varepsilon$-global optimality [75, 77]. Expanding the framework, we describe a hierarchy of dynamic cuts designed to cutoff the current feasible point of an individual node.

Important applications of MIQCQP include quality blending in process networks [17, 39, 59, 74, 79], separating objects in computational geometry [11, 53], portfolio optimisation in finance [70, 86, 90], quadratic assignment [3, 10, 66], maximum clique [28, 49, 85], and \textit{de novo} protein design [55].

We define MIQCQP as:

$$
\begin{aligned}
\min & \ x^T \cdot Q_0 \cdot x + a_0 \cdot x \\
\text{s.t.} & \ b_m^{LO} \leq x^T \cdot Q_m \cdot x + a_m \cdot x \leq b_m^{UP} & \forall \ m \in \{1, \ldots, M\} \\
& \ x \in \mathbb{R}^C \times \{0, 1\}^B \times \mathbb{Z}^I 
\end{aligned}
\quad (\text{MIQCQP})
$$

where $C$, $B$, $I$, and $M$ represent the number of continuous variables, binary variables, integer variables, and constraints, respectively. We assume that it is possible to infer finite bounds $[x^L_i, x^U_i]$ on the variables participating in nonlinear terms and that a relaxation of MIQCQP can...
be formulated as a bounded MILP. MIQCQP slightly generalises the GloMIQO 1.0 definition [77]; GloMIQO 2 accepts general integer variables and products involving discrete variables (i.e., products of continuous-continuous, continuous-binary, continuous-integer, binary-binary, binary-integer, and integer-integer).

GloMIQO replaces each distinct bilinear and quadratic term $x_i \cdot x_j$ in MIQCQP with an auxiliary variable $w_{i,j}$ using the McCormick hull [71]:

$$w_{i,j} \geq \max \left\{ x_i \cdot x^L_j + x^L_i \cdot x_j - x^L_i \cdot x^L_j; x_i \cdot x^U_j + x^U_i \cdot x_j - x^U_i \cdot x^U_j \right\}, \quad (1)$$

$$w_{i,j} \leq \min \left\{ x_i \cdot x^U_j + x^L_i \cdot x_j - x^L_i \cdot x^U_j; x_i \cdot x^L_j + x^U_i \cdot x_j - x^U_i \cdot x^L_j \right\}. \quad (2)$$

The questions this paper asks are: (i) When can we develop a cut that tightens the termwise relaxation? and (ii) Of the cuts dominating Equations (1) and (2), which ones are easy to generate? GloMIQO 1.0 implements necessary conditions for dominant vertex polyhedral cuts to exist in MIQCQP [73, 75, 77, 104–106]; this paper uses a property of Crama [35] to establish necessary and sufficient conditions for the existence of hyperplanes dominating the termwise linear programming (LP) relaxation of a bilinear expression.

We describe the cutting planes GloMIQO 2 dynamically generates after the solution of each branch-and-bound tree node. The pre-processing step of detecting special mathematical structure determines which classes of cuts may be applicable to the particular MIQCQP; that information is used to generate any supporting hyperplanes in the initial tree nodes. In the first few tree nodes, GloMIQO 2 records whether or not specific classes of separating hyperplanes improve the objective function; GloMIQO 2 then decides to continue generating each class of cutting plane or not.

We begin in Section 2 by discussing prior work relevant to our own, continue in Section 3 by describing necessary and sufficient conditions for cuts dominating Equations (1) and (2) to exist in $x^T \cdot Q_m \cdot x$, discuss several classes of hyperplanes in Section 4, and present our hierarchy of dynamically-generated cuts in Section 4.6. This hierarchy is based on complexity analysis for each of the cut classes and observations as to the global/local validity of each supporting hyperplane. We present illustrative examples and a comparative study in Section 5 and conclude in Section 6.

None of the supporting hyperplanes we describe in this paper are entirely new. Our contributions are in the areas of (i) automatically recognising which classes of cuts will be valuable or not, (ii) developing effective algorithms for constructing cuts and (iii) ordering the cut-generation scheme into a hierarchy. Effectively, GloMIQO 2 runs several low-complexity algorithms before attempting the most computationally expensive methods that may generate the individually deepest cuts.

2. Literature Review

GloMIQO is a framework for the deterministic global optimisation of MIQCQP [77]. GloMIQO is centrally focused on detecting and exploiting special mathematical structure, so it integrates a variety of algorithmic components. For a more comprehensive view on global optimisation algorithms, the reader is referred to the excellent reviews of Floudas and co-workers [42, 44], a recent MIQCQP-specific review [24], and a variety of texts [40, 41, 50, 84, 99, 108].

The GloMIQO software implementation addresses MIQCQP to $\varepsilon$-global optimality. See the review of Bussieck and Vigerske [27] for a general discussion of numerical optimisation software; we limit our focus to code bases relevant to GloMIQO. Global optimisation software addressing MIQCQP includes: $\alpha$BB [5, 6, 9, 41, 69]; ANTIGONE [76, 78]; BARON [108, 109]; branch-and-cut for QCQP [13–16]; Couenne [19, 67]; LINDO [45, 64]; and SCIP [1, 2, 20, 21]. Except for the methodology described in this paper, the GloMIQO 1.0 framework of (i) reformulating user input, (ii) detecting special mathematical structure, and (iii)
branch-and-bound global optimisation remains unchanged. In the case of MIQCQP, the solvers GloMIQO and ANTIGONE are equivalent, so all material in this manuscript additionally applies to ANTIGONE [76, 78].

Global optimisation of quadratically-constrained quadratic programs (QCQP) using dynamically-generated supporting hyperplanes was first introduced by Audet et al. [14]. The GloMIQO 2 cutting plane framework differs from Audet et al. [14] as follows:

- GloMIQO 2 incorporates a broader array of separating hyperplanes (see Section 4).
- GloMIQO 2 cuts may be based on three or higher dimensions (see Section 4).
- GloMIQO 2 cuts are based not only on individual terms or equations but also on the collection of quadratic and bilinear terms existing in MIQCQP.
- GloMIQO 2 may not derive cuts even when it deduces that they must exist; GloMIQO 2 may rather stop generating cuts when it infers that the advantages of additional cutting planes are outweighed by the cost of calculating them (see Section 4.6).
- GloMIQO 2 may not generate every class of hyperplane at every node; if the computationally inexpensive cuts significantly improve the relaxation bound, GloMIQO 2 will not derive more expensive cuts (see Section 4.5 and Section 4.6).
- GloMIQO 2 does not automatically add cuts generated in a parent node into the initial LP relaxation of a child node. GloMIQO 2 rather maintains a pool of previously-generated cuts that it integrates or not depending on whether or not the current LP solution violates that cut (see Section 4.6).
- GloMIQO 2 distinguishes between globally valid cuts that are applicable to any node of the branch-and-bound tree and locally valid cuts that may only be applied to descendants of the node where the cut is derived (see Section 4 and Section 4.6).

Recall that most state-of-the-art optimisation solvers will dynamically-generate hyperplanes using a cutting plane framework satisfying similar goals. As mentioned in Section 1, a contribution of this manuscript is to develop a cut-generation framework for MIQCQP that effectively incorporates the previous seven points.

GloMIQO 1.0 already incorporates two important classes of dynamically-generated cuts [77]: redundant constraints based on the Reformulation-Linearisation Technique (RLT) [63, 99–102] and outer-approximation cuts based on convex multivariable terms and univariate quadratics [14, 20, 21]. These two classes of cuts also are integrated into GloMIQO 2. Liberti and Pantelides [63] proposed an extension to the RLT technique that adds specific bilinear terms to the model formulation; we consider a variant of this strategy that, in the special case of quadratic assignment problems, automatically reduces to the Adams and Sherali [4] and Adams and Johnson [3] RLT-1 formulation.

A recurring idea in MIQCQP is convexifying the matrix $Q_m; m \in \{0, \ldots, M\}$ using a difference of convex (D.C.) underestimators [22, 87, 118]; we refer to these as $\alpha$BB underestimators because they specialise the generic results of Floudas and co-workers to MIQCQP [5, 6, 9, 65]. Anstreicher [12] has shown that, no matter the choice of $\alpha$ parameter, a D.C. relaxation of MIQCQP is dominated by a relaxation combining McCormick [71] envelopes and a semidefinite condition. Based on this result [12], we avoid extensive computation generating the $\alpha$ parameters (e.g., we do not solve an LP as proposed by Zheng et al. [118]). Section 4.6 presents the proposed $\alpha$BB convexifications and demonstrates their importance to the GloMIQO 2 cutting plane strategy; generating $\alpha$BB cuts is less computationally demanding than, for example, deriving vertex polyhedral cuts.

The statically-generated underestimators in GloMIQO 1.0 already use low-dimensional ($d = 3$) edge-concave relaxations [75, 77]; these cuts specialise work of Tardella [104–106] and Meyer and Floudas [73] to MIQCQP. For higher-dimensional ($d = 4 − 7$) cutting planes, this paper extends the methodology of Meyer and Floudas [73]. Similar cuts can be generated by solving an LP [18, 98].

Multiple relaxations have been proposed based on a semidefiniteness condition (e.g., [11, 25,
The relaxations are variants on the constraint:

\[ X - x \cdot x^T \succeq 0. \]  

(3)

For MIQCQP with \( C \times B \times I \) nonlinearly-participating variables, these relaxations typically require order \( |C \times B \times I|^2 \) nonlinear terms when, in many practical instantiations of MIQCQP, the quadratic and bilinear terms participating in MIQCQP sparsely populate the possible nonlinearities. Observe, for instance, in the large-scale ten-plant generalised pooling problem of Meyer and Floudas [74] that there are 130 nonlinearly participating variables but only 750 bilinear terms; fewer than 4.5% of the \( 1.69 \times 10^4 \) possible nonlinear terms are in the model formulation.

GloMIQO is centrally focussed on large-scale MIQCQP, so even carefully-constructed semidefinite convexifications of Saxena et al. [94] and Qualizza et al. [88] are too unwieldy. GloMIQO 2 does not add quadratic or bilinear terms specifically for semidefinite relaxations, but it does consider an additional expression representing the collection of nonlinear terms in MIQCQP:

\[ \sum_{(i,j) \in T_Q} x_i \cdot x_j, \]  

(4)

where set \( T_Q \) is the set of pairs for which nonlinear term \( Q_{m,i,j} \) exists in an equation \( m \). Expression (4) is used to develop cutting planes. The GloMIQO 2 implementation breaks expression (4) into disjoint sets before deriving cuts. Non-interacting terms are sum-decomposable [104, 105]; cuts including unconnected bilinear terms introduce unnecessary complexity.

### 3. MIQCQP as an Undirected Graph

In the special mathematical structure detection phase, GloMIQO analyses each equation \( m \in \{0, \ldots, M\} \) as an undirected graph where the nodes represent variables and the edges denote nonzero coefficients [77]. GloMIQO 2 additionally analyses the collection of nonlinear terms existing in the model by creating an undirected graph representation of the collection of bilinear and quadratic terms in MIQCQP.

The key question that GloMIQO 2 asks is, given a quadratic expression \( x^T \cdot Q_m \cdot x \), do there exist cutting planes dominating the termwise relaxation of the expression? The answer suggests whether or not GloMIQO 2 should add low-dimensional (\( d = 3 \)) edge-concave underestimators to the relaxation of MIQCQP or generate high-dimensional cutting planes.

From a result of Crama [35], this section develops a necessary and sufficient condition in Theorem 3.10 for there to exist a cutting plane strictly dominating the termwise relaxation of bilinear expression \( x^T \cdot Q_m \cdot x \) (i.e., matrices \( Q_m \) with diagonal elements \( Q_{m,i,i} = 0 \)). GloMIQO 2 uses Theorem 3.10 to determine whether or not it is worthwhile to derive high-dimensional cuts. Graph theoretical methods have been previously used to find the largest possible difference between the McCormick relaxation and the convex hull of general bilinear functions [68], but Theorem 3.10 provides a useful screen to find if the minimum necessary difference is zero or not. To begin, define a balanced \( \{0, \pm 1\} \) matrix [95]:

**Definition 3.1 [95]:** A \( \{0, \pm 1\} \)-matrix \( A \) is balanced if for each submatrix \( B \) of \( A \) with exactly two nonzeros in each row and each column, the sum of all the components in \( B \) is divisible by 4.

**Illustration 3.2:** Consider the matrices \( M_1 \) and \( M_2 \) below. The only submatrices of matrices \( M_1 \) and \( M_2 \) with exactly two nonzeros in each row and each column are \( M_1 \) and \( M_2 \) themselves. \( M_1 \) is balanced because the sum of all the components is 0 and 0 mod (4) = 0. But \( M_2 \) is not balanced because the sum of all the components is 6 and 6 mod (4) = 2. Conforti et al. [33] refer to \( M_2 \) as an unbalanced hole of length 6.
\[
M_1 = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & -1 \\
0 & -1 & -1
\end{pmatrix};
M_2 = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix}
\]

**Illustration 3.3:** Consider matrix \(M_3\) with 8 rows and 9 columns.

\[
M_3 = \begin{pmatrix}
c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \\
r_1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
r_2 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\
r_3 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
r_4 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
r_5 & 0 & 1 & 1 & 0 & 0 & 0 & -1 & 0 \\
r_6 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & -1 \\
r_7 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
r_8 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

Some submatrices in \(M_3\) with exactly two nonzero elements in each row and column have component sums divisible by 4. For example, submatrix \(M_{3,1}\) consisting of \(r_5 - r_8\) and \(c_2 - c_4, c_9\) has component sum 4:

\[
M_{3,1} = \begin{pmatrix}
1 & 1 & 0 & 0 \\
r_5 \\
r_6 \\
r_7 \\
r_8
\end{pmatrix}
\]

But the component sums of other submatrices in \(M_3\) are not divisible by 4. Submatrix \(M_{3,2}\) consisting of \(r_1 - r_4, r_6\) and \(c_1, c_2, c_4 - c_6\) has component sum 2:

\[
M_{3,2} = \begin{pmatrix}
c_1 & c_2 & c_4 & c_5 & c_6 \\
r_1 & -1 & 0 & 0 & 1 & 0 \\
r_2 & 0 & -1 & 0 & 1 & 0 \\
r_3 & -1 & 0 & 0 & 0 & 1 \\
r_4 & 0 & 0 & -1 & 0 & 1 \\
r_6 & 0 & 1 & 1 & 0 & 0
\end{pmatrix}
\]

\(M_3\) is not balanced because a submatrix (e.g., \(M_{3,2}\)) does not satisfy Definition 3.1.

Next, we state without proof a theorem of Crama [35]; the only difference between our wording and the original result is that we refer to convex rather than concave extensions (MIQCQP assumes minimisation):

**Theorem 3.4 [35]:** For every canonical pseudo-Boolean expression:

\[
\psi(x) = \sum_{T \in \Gamma} q_T \prod_{i \in T} x_i
\]

where \(x \in [0, 1]^d\), \(\Gamma \subseteq \{T : T \subseteq \{1, 2, \ldots, d\}\}\), and \(q_T \neq 0 \forall T \in \Gamma\), the standard extension \(\psi^S\) of \(\psi\):

\[
\psi^S(x) = \sum_{T \in \Gamma} q_T \begin{cases}
\min \{x_i : i \in T\} & \text{if } q_T < 0 \\
0, 1 - |T| + \sum_{i \in T} x_i & \text{if } q_T > 0
\end{cases}
\]
is equal to the unique convex extension $\psi^C$ of $\psi$ if and only if $M(\psi)$, the constraint matrix of:

$$\min \sum_{T \in \Gamma} q_T \cdot w_T$$

s.t. \[ w_T - x_i \leq 0 \] for $T \in \Gamma, i \in T, q_T < 0,$ \[ -w_T + \sum_{c \in T} x_c \leq |T| - 1 \] for $T \in \Gamma,$ $q_T > 0,$ \[ 0 \leq w_T \leq 1 \] for $T \in \Gamma,$ \[ 0 \leq x_i \leq 1 \] for $i \in \{1, 2, \ldots, d\}$

is balanced.

**Illustration 3.5:** Consider canonical pseudo-Boolean expression:

$$\psi(x_1, x_2, x_3) = Q_{12} \cdot x_1 \cdot x_2 + Q_{13} \cdot x_1 \cdot x_3 + Q_{23} \cdot x_2 \cdot x_3$$

(10)

where $x_1, x_2, x_3 \in [0, 1]$, $Q_{12}, Q_{23} > 0$ and $Q_{13} < 0$. We will show that $\psi^S = \psi^C$ by showing that the constraint matrix of Equation (10) is balanced.

In the notation of Equation (7), $\Gamma = \{T_1, T_2, T_3\}$ where $T_1 = \{1, 2\}, T_2 = \{1, 3\}, T_3 = \{2, 3\}, q_{T1} = Q_{12}, q_{T2} = Q_{13}, q_{T3} = Q_{23}$, and $|T_i| = 2$ for $i = 1, 2, 3$. Linear program (9) becomes:

$$\min Q_{12} \cdot w_{12}^{xx} + Q_{13} \cdot w_{13}^{xx} + Q_{23} \cdot w_{23}^{xx}$$

s.t. \[ w_{12}^{xx} - x_1 \leq 0, \]
\[ w_{13}^{xx} - x_3 \leq 0, \]
\[ -w_{12}^{xx} + x_1 + x_2 \leq 1, \]
\[ -w_{23}^{xx} + x_2 + x_3 \leq 1, \]
\[ 0 \leq w_{12}^{xx}, w_{13}^{xx}, w_{23}^{xx} \leq 1, \]
\[ 0 \leq x_i \leq 1 \] for $i \in \{1, 2, 3\}$

(11)

Matrix $M_4$ represents the constraint matrix of linear program (11). In the $\{0, \pm 1\}$ matrix, columns $c_1 - c_3$ represent original variables $x_1 - x_3$ and columns $c_4 - c_6$ are auxiliary variables. Rows $r_1, r_2$ correspond to $Q_{13} < 0$; rows $r_3$ and $r_4$ are for $Q_{12}, Q_{23} > 0$; rows $r_5 - r_{16}$ are variable bounds. Recall that right-hand side coefficients are excluded from an LP constraint matrix.
To see that $M_4$ is balanced, recall from Definition 3.1 that we only need to consider submatrices with two nonzeros in each row and each column. It is therefore sufficient to limit our analysis to rows $r_1 - r_4$; variable bound rows $r_5 - r_{16}$ have exactly one entry each. After excluding rows $r_5 - r_{16}$, columns $c_4$ and $c_6$ have exactly one entry each and are additionally excluded. Therefore, $M_4$ is balanced if and only if submatrix $M_{4,1}$ consisting of rows $r_1 - r_4$ and columns $c_1 - c_3, c_5$ is balanced:

\[
M_{4,1} = \begin{pmatrix}
 c_1 & c_2 & c_3 & c_5 \\
 r_1 & -1 & 0 & 0 & 1 \\
 r_2 & 0 & 0 & -1 & 1 \\
 r_3 & 1 & 1 & 0 & 0 \\
 r_4 & 0 & 1 & 1 & 0
\end{pmatrix}
\]

To see that $M_{4,1}$ is balanced, observe that the only nonempty submatrix of $M_{4,1}$ with exactly two nonzeros in each row and each column is $M_{4,1}$ itself. The sum of the components in $M_{4,1}$ is 4. By Definition 3.1, $M_4$ is balanced because each submatrix of $M_4$ with exactly two nonzeros in each row and each column has a component sum divisible by 4. Concluding Illustration 3.5, it is therefore sufficient to limit our analysis to rows $r_1 - r_4$ with two nonzeros in each row and each column. It is therefore sufficient to limit our analysis to rows $r_1 - r_4$ with two nonzeros in each row and each column.

To begin, we show that $M_4$ is balanced or not, $M_4$ is balanced if and only if submatrix $M_{4,1}$ consisting of rows $r_1 - r_4$ and columns $c_1 - c_3, c_5$ is balanced:

\[
\psi = x^T \cdot Q_m \cdot x + x^T \cdot (Q_m + Q_m) \cdot (x - x^L) + (x^L)^T \cdot Q_m \cdot x^L
\]

Although GloMIQO assumes bounds $x \in [x^L, x^U]^d$ rather than $x \in [0, 1]^d$, Theorem 3.6 shows that Theorem 3.4 is easily extended to the case $x \in [x^L, x^U]^d$.

**Theorem 3.6:** The triangulation representing the convex extension $\psi^C$ of quadratic expression $x^T \cdot Q_m \cdot x$ for $m \in \{0, \ldots, M\}$ dominates the termwise relaxation $\psi^S$ if and only if the triangulation representing the convex extension $\hat{\psi}^C$ of $\hat{x}^T \cdot \hat{Q}_m \cdot \hat{x}$ where $\hat{x} \in [0, 1]^d$ and $\hat{Q}_{m,i,j} = Q_{m,i,j} \cdot (x^L - x^L) \cdot (x^L - x^L)$ dominates the standard extension $\hat{\psi}^S$.

**Proof:** To begin, we show that $x^T \cdot Q_m \cdot x$ is equivalent to $\hat{x}^T \cdot \hat{Q}_m \cdot \hat{x}$ up to an affine shift. Using basic algebra:

\[
x^T \cdot Q_m \cdot x = (x - x^L)^T \cdot Q_m \cdot (x - x^L) + (x^L)^T \cdot (Q_m + Q_m) \cdot (x - x^L) + (x^L)^T \cdot Q_m \cdot x^L
\]

\[
= \hat{x}^T \cdot \hat{Q}_m \cdot \hat{x} + \hat{x} + \hat{Q}_m \cdot \hat{x} + \hat{Q}_m \cdot \hat{x} + \hat{Q}_m \cdot \hat{x}
\]

\[
= \hat{x}^T \cdot \hat{Q}_m \cdot \hat{x} + \hat{a}_m \cdot \hat{x} + \hat{b}_m
\]

where $\hat{x} = \frac{x^L - x^L}{x^L - x^L} \in [0, 1]^d$, $\hat{a}_m = \left(\frac{x^L - x^L}{x^L - x^L}\right)^T$, and $\hat{b}_m = \left(\frac{x^L - x^L}{x^L - x^L}\right)^T$. By Equation (12), $x^T \cdot Q_m \cdot x$ and $\hat{x}^T \cdot \hat{Q}_m \cdot \hat{x} + \hat{a}_m \cdot \hat{x} + \hat{b}_m$ have identical triangulations. By Proposition 2.2 of Tardella [105], linear and constant terms $\hat{a}_m \cdot \hat{x} + \hat{b}_m$ have standard extensions equal to their convex extensions. Therefore, deviations between $\psi^S$ and $\psi^C$ are entirely due to a difference between $\hat{\psi}^S$ and $\hat{\psi}^C$.

Practically, we do not calculate $\hat{Q}_m$ because each element $\hat{Q}_{m,i,j}$ of the scaled matrix is $Q_{m,i,j}$ in the original matrix $Q_m$ multiplied by the positive constant $(x^L - x^L) \cdot (x^L - x^L)$. From linear program (9), the test for balancedness of matrix $M(\psi)$ is based only on the signs of the elements in $\hat{Q}_m$; the magnitudes are irrelevant. Determining if matrix $M(\psi)$ is balanced or not has received considerable attention in the literature (e.g., [33, 95, 112]); our characterisation is with respect to MIQCP.

**Definition 3.7:** A cycle $C_n$ of length $n$ in the undirected graph representation of $x^T \cdot Q_m \cdot x$ denotes an ordered subset of variables $x^L_1, x^L_2, \ldots, x^L_n$ such that $Q_{m,1,2}, Q_{m,2,3}, \ldots, Q_{m,n-1,n}, Q_{m,n,1}$ are all nonzero.
Proof. \(\psi_i\) is a function of positively-weighted edges; we will show \(\psi_i\) is balanced.

Consider submatrix \(B\) of the constraint matrix of Equation (13) and there exists a cutting plane dominating the termwise relaxation of Equation (13). Columns \(r_5\) and \(r_6\) correspond to positively-valued coefficients \(Q_{23}, Q_{24} > 0\); the remaining rows correspond to the negatively-valued coefficients. We ignore rows corresponding to variable bounds; as in Illustration 3.5, each variable bound row has exactly one element and does not effect whether the constraint matrix is balanced or not.

Equation (5) corresponds to Cycle 1 of Equation (13) and Equation (6) is related to Cycle 3. In each case, the submatrix columns include both the participating variables and the auxiliary variables with negatively-valued coefficients. The submatrix rows represent nonzero coefficients (i.e., the edges in Figure 1).

Illustration 3.3 showed that \(M_{3,1}\) is balanced; Cycle 1 has two positively-weighted edges (two is even). We also demonstrated that \(M_{3,2}\) is unbalanced; Cycle 3 has one positively-weighted edge (one is odd).

We want an efficient test for the existence of cuts dominating the termwise relaxation of a bilinear expression. Theorem 3.4 [35] shows that these dominant cuts exist if and only if the constraint matrix is unbalanced. Theorem 3.10 uses Theorem 3.4 to prove that the existence of a dominant cut in a bilinear expression is equivalent to the existence of a cycle with an odd number of positively-weighted edges. If there is a cycle with an odd number of positively-weighted edges (e.g., Cycle 2 or 3 in Equation 13), then there exists a cutting plane dominating the termwise relaxation.

**Theorem 3.10:** There exists a cutting plane dominating the termwise relaxation of bilinear expression \(x^T \cdot Q_m \cdot x\) (matrix \(Q_m\) has diagonal elements \(Q_{m,i,i} = 0\)) if and only if \(x^T \cdot Q_m \cdot x\) has a cycle \(C_n\) of length \(n \geq 3\) with an odd number of positively-weighted edges \(Q_{m,i,j}\) such that \(i, j \in \{c_1, \ldots, c_n\}\) and \(i + 1 = j\).

**Proof:** \((\Leftarrow)\) Assume that \(\psi(x) = x^T \cdot Q_m \cdot x\) has a cycle \(C_n\) of length \(n \geq 3\) with an odd number of positively-weighted edges; we will show \(\psi^C(x) \neq \psi^S(x)\) by proving that \(M(\psi)\) is not balanced.

Consider submatrix \(B\) of \(M(\psi)\) with rows corresponding to the constraints in linear program.
(9) that relax the terms $Q_{m,1,2}, Q_{m,2,3}, \ldots, Q_{m,n-1,n}, Q_{m,n,1}$ participating in cycle $C_n$. There are two rows:

$$w^x_{i,j} - x_j$$
$$w^x_{i,j} - x_j$$

for each $Q_{m,i,j} < 0$ and there is one row:

$$-w^x_{i,j} + x_i + x_j$$

for each $Q_{m,i,j} > 0$. Restrict the columns of $B$ to $x^C_1, x^C_2, \ldots, x^C_n$ and to $w^{xx}_{i,j}, i, j \in c_1, \ldots, c_n$ such that $Q_{m,i,j} < 0$. This matrix $B$ has exactly two nonzeros in each row ($Q_{m,i,j} > 0$ implies that there are two 1’s in the the row; $Q_{m,i,j} < 0$ implies that there is a 1 and a -1 in the row) and exactly two nonzeros in each column (the cycle includes each variable twice, so each $x_i$ participates twice; $w^{xx}_{i,j}$ participates twice if $Q_{m,i,j} < 0$ and is excluded from the submatrix if $Q_{m,i,j} > 0$).

Summing the nonzeros in $B$, we get a sum of 0 for each row where $Q_{m,i,j} < 0$ and a sum of 2 for each row where $Q_{m,i,j} > 0$; the sum of components in $B$ is $2 \cdot k$ where $k$ is the number of positively-weighted $Q_{m,i,j}$. By assumption, $k$ is odd, so $(2 \cdot k) \mod 4 = 2$ and $M(\psi)$ is not balanced by Definition 3.7. Applying Theorems 3.4 and 3.6, there exists a cutting plane in $x^T \cdot Q_m \cdot x$ dominating the termwise relaxation.

($\Rightarrow$) Assume that $\psi(x) = x^T \cdot Q_m \cdot x$ has no cycle $C_n$ of length $n \geq 3$ with an odd number of positively-weighted edges; we will show $\psi^S(x) = \psi^C(x)$ by proving $M(\psi)$ is balanced.

Consider an arbitrary submatrix $B$ of $M(\psi)$ with exactly two nonzeros in each row and each column. Rows corresponding to variable bounds of linear program (9) cannot exist in $B$ because there is only one nonzero in each of those rows (see Illustration 3.5). Columns corresponding to auxiliary variables $w^{xx}_{i,j}$ such that $Q_{m,i,j} > 0$ cannot exist in $B$ because there is only one row where $w^{xx}_{i,j}$ participates other than in excluded variable bound $0 \leq w^{xx}_{i,j} \leq 1$ (see Illustration 3.5).

By assumption, the rows of $B$ have exactly two nonzero entries. For rows corresponding to $Q_{m,i,j} < 0$, these will be 1 (for the auxiliary variable) and -1 (for one of the two original problem variables); for rows corresponding to $Q_{m,i,j} < 0$ these will be 1 and 1 (for the two original problem variables; the previous paragraph excludes the auxiliary variable in this row). Columns in $B$ corresponding to $x_i$ must have exactly two nonzero entries. Therefore, nonempty $B$ must correspond to one or more disjoint cycles (the cycles cannot share variables; otherwise there would be more than two nonzeros in a column $x_i$, $i \in \{1, \ldots, d\}$). Cycles must have length at least three because $x_i \cdot x_j$ is not distinct from $x_j \cdot x_i$.

For each disjoint cycle $C_n$ in $B$, the sum of components in rows corresponding to $Q_{m,i,j} < 0$ is 0 and the sum of components in rows corresponding to $Q_{m,i,j} > 0$ is 2; the sum of the components in each cycle is therefore $2 \cdot k$ where $k$ is the number of positively-weighted $Q_{m,i,j}$. By assumption, $k$ is even for each cycle, so each cycle has $(2 \cdot k) \mod 4 = 0$. The cycles are disjoint, so summing the components in each cycle is equivalent to summing $B$; we conclude that the sum of components in $B$ is divisible by 4 and that $\psi^S(x) = \psi^C(x)$ because $M(\psi)$ is balanced.

GloMIQO 2 uses the test in Theorem 3.10 to decide whether or not to look for dominant cuts in MIQCQP. Given a quadratic expression, GloMIQO 2 only generates edge-concave cutting planes (Section 4.4) after confirming the existence of cycles with an odd number of positively-weighted edges.

We discuss Corollaries 3.11 − 3.13, Corollary 3.15, and Examples 3.16 − 3.19 as interesting
special cases of Theorem 3.10. Figure 3 illustrates several problem classes of MIQCQP. Corollary 3.11 represents an alternative derivation of Example 5 from Meyer and Floudas [73]. Corollary 3.12 is equivalent to saying that convex envelope of the sum is equal to the sum of the convex envelopes for bipartite graphs with all positive coefficients [48, 68]. Corollary 3.13 is related to Proposition 2 from Meyer and Floudas [73]. Submodular functions are useful in the context of relaxing general nonlinear functions [107], but Corollary 3.15 shows that submodularity-based cuts cannot tighten the termwise relaxation of MIQCQP; the advantage for using submodularity in MIQCQP would have to stem from using a reduced number of constraints to represent the relaxation. Example 3.16 is equivalent to the test of Conforti et al. [33] for an unbalanced hole of length 6. Corollaries 3.11 – 3.13 and 3.15 assume that $Q_m$ is bilinear (diagonal elements are 0).

**Corollary 3.11:** If every nonzero in $Q_m$ is negative, no cut in $x^T \cdot Q_m \cdot x$ dominates the termwise relaxation.

**Proof.** There are no positively-weighted edges in $Q_m$, so there can be no cycle with an odd number of positively-weighted edges. \hfill \Box

**Corollary 3.12:** If every element of $Q_m$ is nonnegative, there is a cut in $x^T \cdot Q_m \cdot x$ dominating the termwise relaxation if and only if there is a cycle $C_n$ of length $n = 2 \cdot k + 1$ where $k \in \mathbb{Z}^+$. 

**Proof.** Odd-length cycles in $Q_m$ have an odd number of positively-weighted edges. \hfill \Box

**Corollary 3.13:** If $Q_m$ is bipartite, then $Q_m$ must have both positive and negative elements for there to exist a cutting plane dominating the termwise relaxation of $x^T \cdot Q_m \cdot x$.

**Proof.** Every cycle in bipartite $Q_m$ has an even number of elements, so finding a cycle with an odd number of positively-weighted edges requires that some nonzeros in the cycle be negative. \hfill \Box

**Definition 3.14** [34, 107]: A bilinear function $f(x_1, \ldots, x_n)$ is unimodular if, by switching a subset $S$ of the variables $x_i; i \in S$ via a transformation such as $\bar{x}_i = x_i^U - x_i$, the bilinear function $f$ can be equivalently represented as:

$$\sum_f q_f^I \cdot x_{j1} \cdot x_{j2} + \sum_f q_f^J \cdot (1 - x_{i1}) \cdot (1 - x_{i2})$$  \hspace{0.5cm} (14)

where $q_f^I$, $q_f^J$ are negative coefficients.

**Corollary 3.15:** The termwise relaxation of a unimodular bilinear function is equal to its convex hull.

**Proof.** By Corollary 3.11, Equation 14 has a McCormick relaxation equal to its convex hull. Any variable switching transformation will preserve the convex hull up to an affine shift; therefore any unimodular bilinear function has a convex envelope equal to its termwise relaxation. \hfill \Box

**Example 3.16:** As illustrated in Figure 2, any $Q_m$ with positive $Q_{m,i,j}$, $Q_{m,i,k}$, and $Q_{m,j,k}$ has a cut dominating the termwise relaxation because the constraint matrix has submatrix $M_1$ from Illustration 3.2. The unbalanced hole of length 6 corresponding to the analysis of Conforti et al. [33] is the cycle on row nodes $x_i \cdot x_j, x_i \cdot x_k, x_j \cdot x_k$ and column nodes $x_i, x_j, x_k$. 

![Figure 2: Bilinear expression $Q_{m,i,j} \cdot x_i \cdot x_j + Q_{m,i,k} \cdot x_i \cdot x_k + Q_{m,j,k} \cdot x_j \cdot x_k$ with $Q_{m,i,j}, Q_{m,i,k}, Q_{m,j,k} > 0$ is an unbalanced hole of length 6 [33]; the constraint matrix has submatrix $M_1$ from Illustration 3.2.](image)
Examples 3.17 - 3.19 discuss the special cases of: process networks [17, 39, 59, 74, 79], separating objects in computational geometry [11, 53], quadratic assignment [3, 10, 66], maximum clique [28, 49, 85], and de novo protein design [55]. The examples are also illustrated in Figure 3. Recall that GloMIQO 2 only considers nonlinear terms that are either natural to the original problem or added as in Section 4.1.

Example 3.17: Figures 3(a) and 3(b) illustrate typical process networks blending problems with intermediate nodes [17, 39, 59, 74, 79]. Bilinear terms arise from multiplying qualities $p$ (light green) by flowrates $f$ (dark purple). Individual process networks equations as depicted in Figure 3(a) have no cycles and therefore no dominant cuts [75]. Figure 3(b) graphs the collection of bilinear terms in a process networks problem; the graph is bipartite with one disjoint vertex set per pool-like structure. By Corollary 3.15, getting a dominant cut based on the collection of bilinear terms in Figure 3(b) requires that there be both negative and positive terms in a bilinear expression.

Example 3.18: Figures 3(c) and 3(d) represent computational geometry problems [11, 53]; here the light red and dark blue represent a two-dimensional problem; an $n$-dimensional problem would have one bow tie per dimension in each equation and one complete subgraph per dimension in the collection of bilinear terms. There are no cycles of length $n \geq 3$ in Figure 3(c), so no dominant cuts can be introduced to an equation.

There are many odd-length cycles in Figure 3(d), so cuts convexifying Equation (3) are attractive (e.g., [11]). However, departing from the methodology of Anstreicher [11], GloMIQO 2 will not directly convexify Equation (3) by introducing new bilinear terms like a typical semidefinite relaxation. Rather, as illustrated in Figure 3(d), GloMIQO 2 divides the bilinear terms into one expression per disjoint vertex set.

Example 3.19: The quadratic objective of a quadratic assignment [3, 10, 66], maximum clique [28, 49, 85], or de novo protein design [55] problem is one connected graph. After GloMIQO 2 adds the bilinear terms described in Section 4.1, the terms in an assignment problem form a complete graph. The cycles illustrated in Figure 3(e) justify alternatively convexifying Equation (3) (e.g., [117]) or finding vertex polyhedral cuts (e.g., [52]) for assignment problems.

The preceding analysis decides whether or not a cut dominating the termwise relaxation exists
but cannot tell whether the current feasible point can be cutoff (the cut has to be constructed for that to be determined). However, GloMIQO 2 uses the results in this section to test the possible advantage of deriving the expensive vertex polyhedral cuts presented in Section 4.4.

4. Classes of Separating Hyperplanes for MIQCQP

This section describes how GloMIQO 2.0 derives classes of supporting hyperplanes. These include: RLT constraints (Section 4.1), convex gradient-based cuts (Section 4.2), αBB relaxations (Section 4.3), and vertex polyhedral hyperplanes (Section 4.4). Section 4.5 discusses the computational complexity associated with deriving each class of cut. This analysis is the foundation for the framework described in Section 4.6.

As illustrated in Figure 4, GloMIQO 2 distinguishes between globally and locally valid cuts. After a convex gradient-based relaxation, αBB cut, or edge-concave facet is derived, it is stored in a cut pool for potential integration into later nodes. A globally valid cut can be applied to any node of the branch-and-bound tree; a locally valid cut can only be added to direct descendants of the node where the cut is derived.

GloMIQO 2 stores and updates RLT equations at every node of the branch-and-bound tree [77]; properly-updated RLT equations can be applied universally to any node. The cuts described in Sections 4.2 and 4.3 are based on outer-approximation of convex expressions and are therefore globally valid; the edge-concave facets in Section 4.4 are only locally valid.

4.1. Reformulation-linearisation Technique (RLT) Equations

GloMIQO 2 permits the addition of bilinear terms to the model formulation for the purpose of generating linear equality equation/variable products:

\[ \left( \sum_j a_{m,j} \cdot x_j - b_m \right) \cdot x_i = \sum_j a_{m,j} \cdot x_j \cdot x_i - b_m \cdot x_i = 0 \]  

Liberti and Pantelides [63] proposed a graph-theoretical algorithm that adds bilinear terms to the model formulation with a final goal of having the number of equations with form (15) be greater than the number of additional bilinear terms added to the model formulation. Liberti and Pantelides [63] specifically use Equation (15) to eliminate variables from MIQCQP.

The GloMIQO 2 strategy for adding bilinear terms also addresses equality equations but does not use the graph-theoretical algorithm of Liberti and Pantelides [63] or eliminate variables from MIQCQP. GloMIQO 2 does not compare numbers of equations and bilinear terms because the addition of a single deep cut may justify adding several bilinear terms. GloMIQO 2 does not eliminate variables because preserving Equation (15) is important for its variable bounding strategy.
Figure 5.: GloMIQO 2 adds bilinear terms to MIQCQP for RLT-1 Equation (15) but will not connect otherwise disjoint vertex sets; this is equivalent to allowing additional chords only when nodes are already connected.

To integrate Equation (15), GloMIQO 2 will add any necessary bilinear terms to the model formulation with the caveats that:

1. Linearly-participating variables do not become nonlinearly-participating variables (i.e., bilinear terms are not formed using a variable that participates only linearly in MIQCQP).
2. The disjoint vertex sets in the undirected graph representation of Equation (4) do not change (e.g., disjoint pool-like structures in Figure 3(b) are not be connected). Figure 5 illustrates this rule.
3. Bilinear terms that must be equal to 0 are not added to the model formulation. For variables $x \in \{0, 1\}^I$ and a pair of constraints:
   \[
   x_1 + \sum_{i=2}^{I-1} x_i = 1; \quad x_1 + \sum_{i=I'}^I x_i = 1
   \]
   GloMIQO 2 will infer $x_1 \cdot x_i = 0$ $\forall i \in \{2, \ldots, I\}$ and not incorporate those terms when it multiplies $x_i$ by a constraint equation. This strategy is highly relevant in quadratic assignment problems.
4. In Equation (15), $b_m = 1$. This effectively restricts RLT-1 generation to assignment-like problems.

Observe that, in the case of quadratic assignment, adding the proposed bilinear terms is equivalent to the Adams and Sherali [4] and Adams and Johnson [3] RLT-1 formulation. Incorporating many bilinear terms without eliminating constraint equations would be disallowed by the Liberti and Pantelides [63] algorithm but is imperative to attain the Adams and Sherali [4] and Adams and Johnson [3] bound. Similarly, Section 5.1 discusses GLOBALLib Ex. 2.1.9 [43, 72]; adding 33 bilinear terms but only 10 RLT equations is disallowed by Liberti and Pantelides [63] but required for the tight GloMIQO 2 root node.

GloMIQO 2 automatically adds Equation (15) to MIQCQP in the special structure phase; it integrates every equation that does not add bilinear terms to the model formulation (beyond the ones added as described above). But GloMIQO 2 does not include Equation (15) when it solves upper bounding, feasibility problems; Equation (15) is a difficult constraint for local nonlinear programming (NLP) solvers [38].

The preceding describes features unique to GloMIQO 2. Like GloMIQO 1.0, GloMIQO 2 still considers dynamically adding inequality equation/variable and equation/equation products to cutoff a point $\hat{x}$ feasible to the LP relaxation of MIQCQP [77].

4.2. Hyperplanes for Convex Univariate Quadratics and Convex Multivariable Terms

Given a point $\hat{x}$, $\hat{w}^{xx}$ feasible to the LP relaxation of MIQCQP, GloMIQO 2 will, if possible, cutoff the current feasible solution using outer approximation for univariate quadratics and convex multivariable terms [77]. If a univariate quadratic or convex multivariable term violates its convex hull by at least $\varepsilon_V > 0$, GloMIQO 2 adds a hyperplane to cutoff the current feasible
point:

\[
\begin{align*}
\text{Univariate Quadratics} & \quad \begin{cases} 
\hat{x}_i^2 - \hat{w}_{i,i}^{xx} \geq \varepsilon_V \\
\end{cases} \\
\quad \begin{cases} 
\downarrow \\
\quad \begin{cases} 
\hat{w}_{i,j}^{xx} - 2 \cdot \hat{x}_i \cdot x_j + \hat{x}_i \cdot \hat{x}_j \geq 0
\end{cases}
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\text{Multivariable Terms} & \quad \begin{cases} 
\sum_{i=0}^{C} \sum_{j=0}^{C} Q_{i,j} \cdot (\hat{x}_i \cdot \hat{x}_j - \hat{w}_{i,j}^{xx}) \geq \varepsilon_V \\
\quad \begin{cases} 
\downarrow \\
\quad \begin{cases} 
\sum_{i=0}^{C} \sum_{j=0}^{C} Q_{i,j} \cdot (w_{i,j}^{xx} - \hat{x}_i \cdot x_j - \hat{x}_i \cdot \hat{x}_j + \cdot \hat{x}_j) \geq 0
\end{cases}
\end{cases}
\end{cases}
\end{align*}
\]

Observe that cuts (16) and (17) are globally valid because they outer-approximate convex expressions. Cuts (16) and (17) are applicable to any node of the tree (including non-descendants).

4.3. \textbf{αBB Cuts}

The \textbf{αBB} cuts in GloMIQO 2 are based on convexifying a quadratic matrix; they specialise the \textbf{αBB} relaxations, which are applicable to any \(\mathcal{G}^2\) function, to MIQCQP [5, 6, 9, 65].

\textbf{Lemma 4.3.1:} For quadratic expression \(x^T \cdot Q_m \cdot x\), there exist convexifying vectors \(\alpha \in R^{C \times B \times I}\) such that:

\[
x^T \cdot (Q_m + \text{Diag}(\alpha)) \cdot x \geq 0.
\]

\textit{Proof.} Consider a nonuniform diagonal shift of \(Q_m\):

\[
\alpha_i = \max \left\{ 0, \min \left\{ -\lambda_{\min}, -Q_{i,i} + \sum_{i \neq j} |Q_{m,i,j}| \right\} \right\} \quad \forall \ i = 1, \ldots, C \times B \times I
\]

where \(\lambda_{\min}\) is the minimum eigenvalue of \(Q_m\). According to the Gerschgorin circle theorem [46], the eigenvalues of \(Q_m\) lie within \(-Q_{i,i} + \sum_{i \neq j} |Q_{m,i,j}|\); the eigenvalues of \(Q_m\) are also bounded by \(\lambda_{\min}\). Then \(\text{Diag}(\alpha)\) convexifies \(Q_m\) as in Equation (18) by ensuring that the shifted matrix is positive semidefinite. \(\Box\)

Equation (19) combines the uniform diagonal shift \(\lambda_{\min}\) of Androulakis et al. [9] with the nonuniform Gerschgorin shift of Adjiman et al. [6]. Note the distinction between Equation (18) and a standard \(\alpha\text{BB}\) approach: Equation (18) is not an underestimator matching \(x^T \cdot Q_m \cdot x\) at the corner points; GloMIQO 2 already has underestimators matching the function at its corner points due to Equations (1) and (2). The purpose of Equation (18) is rather to find a convex expression that is violated at the current LP relaxation.

Given a point \(\hat{x}, \hat{w}^{xx}\) feasible to the LP relaxation of MIQCQP and an expression (18) that is violated by at least \(\varepsilon_V\), an \(\alpha\text{BB}\) cut is similar to Equation (17):

\[
\alpha\text{BB Cut} \begin{cases} 
\sum_{i=0}^{C} \sum_{j=0}^{C} (Q_{i,j} + \alpha_i \cdot \delta_{i,j}) \cdot (\hat{x}_i \cdot \hat{x}_j - \hat{w}_{i,j}^{xx}) \geq \varepsilon_V \\
\quad \begin{cases} 
\downarrow \\
\quad \begin{cases} 
\sum_{i=0}^{C} \sum_{j=0}^{C} (Q_{i,j} + \alpha_i \cdot \delta_{i,j}) \cdot (w_{i,j}^{xx} - \hat{x}_i \cdot x_j - \hat{x}_i \cdot \hat{x}_j + \cdot \hat{x}_j) \geq 0
\end{cases}
\end{cases}
\end{cases}
\]

where \(\delta_{i,j}\) is the Kronecker delta function.

For general \(\mathcal{G}^2\) functions, \(\alpha\text{BB}\) relaxations are locally valid relaxations derived using variable bounds. However, in the quadratic case, \(\alpha\text{BB}\) cuts proposed in Equation (20) directly convexify
Anstreicher [12] has shown that, no matter the choice of $\alpha$, cuts of the form in Equation (20) are dominated by alternative cuts. GloMIQO 2 therefore avoids excessive computation in deriving the best possible $\alpha$ parameter and finds a convexifying vector using Equation (19).

To construct an $\alpha$BB cut, GloMIQO 2 considers every multivariable term $x^T \cdot Q \cdot x$ in (i) the $M$ equations natural to MIQCQP and (ii) the collection of nonlinearities as represented by Equation (4). GloMIQO 2 develops $\alpha$BB cuts for multivariable terms when the relevant univariate quadratics already exist in MIQCQP; it will not add nonlinearities to the model formulation for cut generation.

To develop a violated expression that can be cutoff as in (20), GloMIQO 2 takes a multivariable term $x^T \cdot Q \cdot x$ and changes the signs of the nonzero $Q_{i,j}$, so that the expression is maximally violated at the point $\hat{x}$, $\hat{x}$ feasible to the current LP relaxation and requires an underestimating (rather than overestimating) cut:

$$
\sum_{i=0}^{C} \sum_{j=i}^{C} s_{i,j} \cdot |Q_{i,j}| \cdot x_i \cdot x_j
$$

where $s_{i,j} = \begin{cases} 
1 & \hat{w}_{i,j}^{xx} < \hat{x}_i \cdot \hat{x}_j \\
-1 & \hat{w}_{i,j}^{xx} > \hat{x}_i \cdot \hat{x}_j 
\end{cases}$

(21)

If, by the analysis in Section 3, there exists a cutting plane dominating the termwise relaxation of Equation (21), GloMIQO 2 will derive the nonuniform shift vector $\alpha$ in Equation (19). If shifted Equation (20) is violated in the LP relaxation of MIQCQP, then GloMIQO 2 adds an $\alpha$BB cut.

**Illustration 4.3.2:** At the root node of pnt_pack_05.ORD [11], GloMIQO 2 identifies five variables in expression (4) forming a complete, connected graph (equivalent to Figure 3(d)). Using the procedure in Equation (21), GloMIQO constructs matrix $Q_m$ and shifted matrix $Q_m + \text{Diag}(\alpha)$ from the minimum eigenvalue $\lambda_{\text{min}} = -0.5$ of $Q_m$:

$$
Q_m = \begin{pmatrix} 
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 \\
\end{pmatrix}
\Rightarrow Q_m + \text{Diag}(\alpha) = \begin{pmatrix} 
0.5 & 1 & 1 & 1 & 1 \\
1 & 0.5 & 1 & 1 & 1 \\
1 & 1 & 0.5 & 1 & 1 \\
1 & 1 & 1 & 0.5 & 1 \\
1 & 1 & 1 & 1 & 0.5 \\
\end{pmatrix}
$$

$Q_m + \text{Diag}(\alpha)$ is violated at the root node LP relaxation; the relaxed expression has value 1.062 while the true value is 2.258. GloMIQO 2 therefore adds a cut as described in Equation (20).

### 4.4. High-Dimension Edge-Concave Facets

GloMIQO 1.0 includes statically-generated relaxations based on low-dimensional ($n = 3$) edge-concave aggregations [77]; the GloMIQO 2 branch-and-cut framework dynamically integrates higher-dimensional ($n = 3 - 7$) vertex polyhedral cutting planes. These cuts derive facets of the convex envelope and represent the deepest possible cut on an edge-concave multivariable term. The tradeoff is that the cuts are based on the vertices of the hypercube; the complexity for developing higher-dimensional edge-concave facets scales (at best) as $n!$ while $\alpha$BB-based cuts scale with the complexity of finding a minimum eigenvalue.

Another limitation is that the edge-concave facets are locally rather than globally valid; bounds-based cuts can only be integrated into nodes that are direct descendants of the generating node. Note that the preceding is a slight falsification; edge-concave cuts are not strictly local but rather applicable to nodes where the variables relevant to the cut have equal or tighter bounds than the generating node. For large branch-and-bound trees with disparate branches that have similar variable bounds, an ostensibly local cut may actually apply to non-descendant
nodes. However, GloMIQO 2 does not consider this detail and labels every edge-concave cut as local. Because edge-concave cuts are difficult to generate and only locally applicable, observe in Section 4.6 and Figure 8 that GloMIQO 2 applies these cuts cautiously.

In the sequel, we discuss the GloMIQO 2 strategy for developing a hyperplane to cutoff the current feasible point \( \hat{x}, \hat{w}^{xx} \) on an edge-concave expression of the form:

\[
f_Q(x) = \sum_{i=0}^{C} \sum_{j=0}^{C} Q_{i,j} \cdot x_i \cdot x_j.
\]

Assume: (1) Equation (22) has been developed identically to the process in Equation (21), (2) Equation (22) is edge-concave (i.e., \( Q_{i,j} \leq 0 \) when \( i = j \), see [75]), (3) a cut may exist as discussed in Section 3, and (4) a less complex cut cannot be derived.

We present two methods for developing the facets of the convex hull \( \text{conv}(f_Q) \). The naïve approach (Section 4.4.1) exhaustively enumerates all candidate facets of the convex envelope and preserves the valid underestimator that cuts off the relaxed expression \( f_Q(\hat{w}^{xx}) \) by the greatest magnitude. The second method (Section 4.4.2) markedly reduces the number of candidate facets by exploiting (1) dominance relations, (2) linear independence tests, and (3) facets composed of merged simplices.

4.4.1. Naïve approach

The naïve approach:

1. Exhaustively enumerates every combination of \((n+1)\) vertices on the hypercube \([x^L, x^U]^n\) by addressing every \( \xi = [\xi_1, \ldots, \xi_{n+1}] | \xi_i \in [x^L, x^U]^n ; i \neq j \implies \xi_i \neq \xi_j \} \).
2. Discards \( \xi \) with linearly dependent combinations of \((n+1)\) vertices.
3. Fits affine function \( h(\xi) \) to linearly independent \( \xi \) such that \( h(\xi) = f_Q(\xi) \).
4. Tests if \( h(x) \) is a valid underestimator of \( f_Q(x) \) \( \forall x \in [x^L, x^U]^n \).
5. Preserves the valid underestimator of \( f_Q \) in Equation (22) that cuts off relaxed expression \( f_Q(\hat{w}^{xx}) \) by the greatest magnitude.

Equation (22) is vertex polyhedral, so this brute force method yields \( \text{conv}(f_Q) \) by exhaustively considering every simplex in \( x \) as a possible generating set for a facet of \( \text{conv}(f_Q) \).

Step 1 of this approach is straightforward, albeit combinatorially complex. There are \( 2^n \) vertices of the \( n \)-dimensional cube, so exhaustive enumeration grows as \( \binom{2^n}{n+1} \). We systematise the enumeration by mapping each vertex of the \( n \)-cube to an integer. The integers correspond to the binary representation of the vertex:

\[
\begin{align*}
\{x_1^L, x_2^L, \ldots, x_{n-1}^L, x_n^L\} & \rightarrow \{0, 0, \ldots, 0, 0\} = 0 \\
\{x_1^L, x_2^L, \ldots, x_{n-1}^L, x_n^U\} & \rightarrow \{0, 0, \ldots, 0, 1\} = 1 \\
\{x_1^L, x_2^L, \ldots, x_{n-1}^U, x_n^L\} & \rightarrow \{0, 0, \ldots, 1, 0\} = 2 \\
\{x_1^U, x_2^L, \ldots, x_{n-1}^L, x_n^L\} & \rightarrow \{0, 0, \ldots, 1, 1\} = 3 \\
& \quad \vdots \\
\{x_1^L, x_2^L, \ldots, x_{n-1}^U, x_n^U\} & \rightarrow \{1, 1, \ldots, 1\} = 2^n - 1
\end{align*}
\]

We begin with vertices \( \xi = \{0, 1, \ldots, n\} \) and increment \( \xi \) according to a flattened, last-in-first-out depth-first search (LIFO DFS). Relative problem sizes for Equation (22) of dimension \( n \) are delineated in Table 1; there are \( \binom{2^n}{n+1} \) candidate facets.

Step 2 is also simple. Linear independence of the \((n+1)\) vertices is tested via modified Gram-Schmidt orthonormalisation. Polyhedra \( \xi \) with linearly dependent vertices are discarded.
April 3, 2014 Optimization Methods and Software MisenerSmaedgeckFloudas MIQCQP CuttingPlanes

For $\xi$ composed of $(n + 1)$ linearly independent vertices, Step 3 proposes candidate facet $h(x) = \sum_{i=1}^{n} \alpha_i \cdot x_i + \alpha_{n+1}$ where the coefficients $\alpha$ are determined by solving Equation (23):

$$
\begin{bmatrix}
\xi_1, 1 \\
\xi_2, 1 \\
\vdots \\
\xi_{n+1}, 1 \\
\end{bmatrix}
\begin{bmatrix}
\alpha_1 \\
\alpha_2 \\
\vdots \\
\alpha_{n+1} \\
\end{bmatrix} =
\begin{bmatrix}
f_Q(\xi_1) \\
f_Q(\xi_2) \\
\vdots \\
f_Q(\xi_{n+1}) \\
\end{bmatrix}
$$

(23)

The Step 4 test for whether the candidate hyperplane $h(x)$ is in $\text{conv}(f_Q)$ is due to Rikun [89]. Repeating a result of Rikun [89] without proof:

**Lemma 4.4.1.1 [89]:** Let $f(x)$ be a continuously differentiable function on $n$-dimensional convex polytope $P \subset \mathbb{R}^n$ and $\text{conv}_P(f(x))$ be a polyhedral function. Let $h(x)$ be an affine function and there exist $n + 1$ linearly independent vertices of $P: \xi_i, i = 1, \ldots, n + 1$ such that:

$$
h(\xi_i) = f(\xi_i), \quad i = 1, \ldots, n + 1
$$

(Leon 1)

and

$$
h(x) \leq f(x), \quad \forall x \in \text{vert}(P)
$$

(Leon 2)

then $h(x)$ is an element of $\text{conv}_P(f(x))$ and $h(x) = \text{conv}_S(f(x)) = \text{conv}_P(f(x)) \forall x \in S$ where $S$ is defined $\text{conv}\{\xi_i : i = 1, \ldots, n + 1\}$.  

**Theorem 4.4.1.2 [89]:** Let function $f(x)$ be continuously differentiable on convex compact polytope $P$ and its convex envelope $\text{conv}_P(f(x))$ be a polyhedral function. Let there exist a collection of $m$ affine functions $h_i(x)$ so that each function $h_i$ fills the conditions of Lemma 4.4.1.1. Then function $\text{conv}_P(f(x))$ coincides with $\psi(x) = \max\{h_i(x) | i = 1, \ldots, m; x \in P\}$ iff: (a) the generating set of $\psi(x)$ coincides with $\text{vert}(P)$ and (b) for each vertex $\xi \in P, \exists i \in \{1, \ldots, m\}$ such that $h_i(\xi) = f(\xi)$.  

By Lemma 4.4.1.1, linearly independent $\xi$ of $(n + 1)$ vertices that pass both (Leon 1) and (Leon 2) are facets of $\text{conv}(f_Q)$. We preserve the facet passing both tests which cuts off the relaxed expression $f_Q(\hat{\n^*})$ by the greatest magnitude.

**Complexity analysis of the naïve approach**

The naïve approach is combinatorially complex and there is no way of avoiding the curse of dimensionality. But we mention two computationally expensive steps as motivation for a reduced complexity approach.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\text{Dim}$</th>
<th>$\text{2}^n \text{ Vertices}$</th>
<th>$\binom{\text{2}^n}{\text{n+1}}$</th>
<th>Candidates</th>
<th>$n! \text{ Max Possible Facets}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>$4.000 \times 10^3$</td>
<td>$2.000 \times 10^3$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>$7.000 \times 10^4$</td>
<td>$6.000 \times 10^4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>$4.368 \times 10^5$</td>
<td>$2.400 \times 10^1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>$\approx 9.062 \times 10^5$</td>
<td>$1.200 \times 10^2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>$\approx 6.212 \times 10^8$</td>
<td>$7.200 \times 10^2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>$\approx 1.430 \times 10^{12}$</td>
<td>$5.040 \times 10^3$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>$\approx 1.129 \times 10^{16}$</td>
<td>$4.032 \times 10^4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>$\approx 3.123 \times 10^{20}$</td>
<td>$\approx 3.629 \times 10^5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>$\approx 3.081 \times 10^{25}$</td>
<td>$\approx 3.629 \times 10^6$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The first expensive operation in the brute force method is testing for linear independence via modified Gram-Schmidt orthonormalisation. Gram-Schmidt orthonormalisation involves normalising each of the \((n + 1)\) elements in a candidate \(\xi\), so each Gram-Schmidt operation requires \(n \cdot (n + 1)\) floating point divisions. Therefore, the naïve approach may require as many as \(n \cdot (n + 1) \cdot \binom{2n}{n+1}\) floating point divisions.

The second computationally expensive step in the brute force method is solving Equation (23) to generate a possible member facet of \(\text{conv}(f_Q)\). Although testing for linear independence reduces the number of problems of type Equation (23), we still solve linear equations a number of times that scales with \(\binom{2n}{n+1}\).

The following reduced complexity approach sharply reduces the number of Gram-Schmidt orthonormalisations and Equation (23) problems. Steps 1 and 3 – 5 remain identical to the exhaustive enumeration method, but we augment Step 2 with additional elimination criteria for a candidate \(\xi\).

4.4.2. Reduced complexity approach

This section describes reducing the number candidate facets to avoid inverting matrices and performing floating point division.

**Detecting linear independence**

Given a candidate polyhedron \(\xi\), GloMIQO 2 tests for linear independence by using the modified Gram-Schmidt process that subtracts from every new vector its projections in the directions already set [103]. As soon as a vertex in \(\xi\) is linearly dependent on the previous rows, GloMIQO 2 increments \(\xi\) according to LIFO DFS and does not test the independence of the remaining rows. The incremental combinations of vertices allows GloMIQO 2 to cache rows of orthonormalised vectors persisting from previous steps.

**Exploiting dominance relations**

The most dramatic improvements of the reduced complexity approach comes from exploiting dominance relations. GloMIQO 2 determines dominant minimal affine dependencies as proposed by Meyer and Floudas [73]. The key advantage of the Meyer and Floudas [73] approach is determining the appropriate triangulation before calculating candidate supporting hyperplanes. Stated differently, they deduce the vertex points corresponding to each facet of the convex hull before solving linear algebra problems of type Equation (23). On the 3-cube, Meyer and Floudas [73] show that they can establish the appropriate triangulation by comparing the intersection points of (1) two-dimensional diagonals, (2) three-dimensional diagonals, and (3) corner slicing simplices to the three-dimensional diagonals.

To motivate our approach, we paraphrase a result from Meyer and Floudas [73] without proof:

**Proposition 4.4.2.1** [73]: Let \(X := \{x_1, \ldots, x_V\}\) be the set of vertices of a convex polytope in \(\mathbb{R}^n\), and let \(f : \mathbb{R}^n \to \mathbb{R}\) be an edge-concave function on \(\text{conv}(X)\). Let \(T\) be a triangulation of \(X\). Triangulation \(T\) of \(X\) is a finite set of \(n\)-simplices \(\{\triangle^1, \triangle^2, \ldots\}\) satisfying:

- The vertices of each \(\triangle^i\) are \(n + 1\) linearly independent vertices of \(X\).
- Two simplices \(\triangle^i\) and \(\triangle^j\), \(i \neq j\), have no common interior point.
- The union of the simplices, \(\triangle^1 \cup \triangle^2 \cup \ldots\), is \(\text{conv}(X)\).

Further, let \(h_T : \text{conv}(X) \to \mathbb{R}\) be the following piecewise affine function defined by \(T\):

\[
h_T(x) := \sum_{i=1}^{d+1} \lambda_i \cdot f(x^{(i)})
\]

where \(\lambda \in [0, 1]^{d+1}\), \(\sum_{i=1}^{d+1} \lambda_i = 1\), and \(x = \sum_{i=1}^{d+1} \lambda_i \cdot x^{(i)}\), for some \(\{x^{(1)}, \ldots, x^{(d+1)}\} \in T\). \(h_T\) is the
convex envelope \( \text{conv}_X(f) \) if and only if:

\[
\sum_{i \in P(\gamma)} \gamma_i \cdot f(x^i) \leq - \sum_{i \in N(\gamma)} \gamma_i \cdot f(x^i)
\]

for all minimal affine dependencies \( \gamma \) where \( \{x^i : i \in P(\gamma)\} \subseteq \triangle' \subseteq T \) and \( \{x^i : i \in N(\gamma)\} \subseteq \triangle'' \notin T \). The affine dependencies of \( X \) are vectors \( \gamma \in \mathbb{R}^V \) corresponding to \( V \) weights of vert(\( X \)) with \( \sum_{i=1}^n \gamma_i = 0 \) and \( \sum_{i=1}^n \gamma_i \cdot x_i \) for some point \( x_i \) in the interior of \( X \). For minimal affine dependencies (i.e., affine dependencies where every proper subset of the nonzero \( \gamma \) correspond to linearly independent vertices), the positive \( (P(\gamma) = \{i : \gamma_i > 0\}) \) and negative \( (N(\gamma) = \{i : \gamma_i < 0\}) \) components of vector \( \gamma \) are simplices with relative interiors intersecting at unique point \( x^* \).

GloMIQO 2 uses Proposition 4.4.2.1 to eliminate many candidates \( \xi \) before performing Gram-Schmidt orthonormalisation or fitting a hyperplane to a set of points. As in Step 1 of the naive approach, GloMIQO 2 enumerates all \( \binom{2^n}{n+1} \) candidates \( \xi \) of \( (n+1) \) vertices, but does not proceed to Step 2 and check for linear independence or construct a candidate facet of \( \text{conv}(f_0) \) when \( \xi \) is a member of a triangulation that violates Equation (24). If, in a preprocessing step, we successfully determine all dominant minimal affine dependencies on the \( n \)-dimensional cube, then every potential supporting hyperplane determined by Equation (23) will be a member of \( \text{conv}(f_0) \) and the computationally expensive operations in Steps 3 & 4 from the naive approach scale as \( n! \) rather than \( \binom{2^n}{n+1} \). The factor \( n! \), the maximum number of simplices in a triangulation of the \( n \)-cube [36], is the lowest order that guarantees deriving the convex hull of Equation (22) on the \( n \)-cube.

We describe the diagonal-to-diagonal and diagonal-to-corner comparisons GloMIQO 2 uses to establish dominance relationships. These two sets of comparisons do not fully characterise the triangulation in dimension four or higher (e.g., certain plane-to-plane intersections are not covered by the intersections discussed here). Because the GloMIQO 2 implementation does not incorporate every dominance test, it performs Steps 3 – 4 significantly more than \( n! \) times (particularly noticeable for dimension 6 and higher). But note in Section 5 the performance of GloMIQO 2 on a wide array of test problems; we see significant advantage in exploiting dominance relations even without exhaustively defining them.

**Diagonal-to-diagonal comparisons**

To take advantage of the Meyer and Floudas [73] result, we first consider diagonal-to-diagonal comparisons in multiple dimensions by addressing the:

\[
\begin{cases}
2^1 = 2 & \text{intersecting diagonals on each of the } \binom{2^{n-2}}{2} \text{ 2D faces of the } n \text{-cube} \\
2^2 = 4 & \text{intersecting diagonals on each of the } \binom{2^{n-3}}{3} \text{ 3D faces of the } n \text{-cube} \\
2^3 = 8 & \text{intersecting diagonals on each of the } \binom{2^{n-4}}{4} \text{ 4D faces of the } n \text{-cube} \\
& \vdots \\
2^{n-1} & \text{intersecting diagonals on the } n \text{-dimensional cube}
\end{cases}
\]

Because Equation (22) is a quadratic function, we can limit these diagonal comparisons to:

\[
\begin{cases}
\binom{n}{2} \cdot 2^1 & \text{intersecting diagonals on 2D faces of the } n \text{-cube} \\
\binom{n}{3} \cdot 2^2 & \text{intersecting diagonals on 3D faces of the } n \text{-cube} \\
\binom{n}{4} \cdot 2^3 & \text{intersecting diagonals on 4D faces of the } n \text{-cube} \\
& \vdots \\
2^{n-1} & \text{intersecting diagonals on the } n \text{-dimensional cube}
\end{cases}
\]

Theorem 4.4.2.3 justifies eliminating dominance tests as in (25) by considering two diagonals...
Figure 6.: The $2^{n-i}$ diagonal-to-diagonal dominance relations on an $i$-face of an $n$-cube are equivalent; there is no need to calculate every one. The relationship between $f_Q(v_0) + f_Q(v_6)$ and $f_Q(v_2) + f_Q(v_4)$ is the same as the one between $f_Q(v_1) + f_Q(v_7)$ and $f_Q(v_2) + f_Q(v_4)$.

$D_1 - D_2^C$ and $D_2 - D_2^C$ on an $i$-dimensional face of an $n$-cube. These diagonals are represented by two hypercube vertex points ($D_1$ and $D_2$) and their complements ($D_1^C$ and $D_2^C$). We partition the $n$ variables in $x$ into the $i$ variables participating in the diagonal and the $n-i$ fixed variables on the cube face $F$ so that $x = (x_D, x_F)$ where $x_D \in R^i$ and $x_F \in R^{n-i}$.

**Example 4.4.2.2:** Consider the diagonals on a 2-face of the 3-cube (face where $x_3 = x_3^L$; illustrated with vertices $v_0, v_2, v_4, v_6$ in Figure 6):

$$D_1 = \{x_1^L, x_2^L, x_3^L\}; D_1^C = \{x_1^U, x_2^U, x_3^L\}$$

$$D_2 = \{x_1^L, x_2^U, x_3^L\}; D_2^C = \{x_1^U, x_2^L, x_3^L\}.$$

Theorem 4.4.2.3 shows that there is no reason to additionally calculate dominance on the face where $x_3 = x_3^L$ ($D_1 = \{x_1^L, x_2^L, x_3^L\}; D_1^C = \{x_1^U, x_2^L, x_3^L\}; D_2 = \{x_1^L, x_2^U, x_3^U\}; D_2^C = \{x_1^U, x_2^U, x_3^L\}$; represented with vertices $v_1, v_3, v_5, v_7$ in Figure 6).

**Theorem 4.4.2.3:** Assume $D_1 - D_2^C$ dominates $D_2 - D_2^C$:

$$f_Q(x_D, x_F) + f_Q(x_D^C, x_F) < f_Q(x_D, x_F) + f_Q(x_D^C, x_F)$$

(26)

Then analogous diagonal $D_1 - D_1^C$ dominates $D_2 - D_2^C$ on all the other $2^{n-i} - 1$ faces $F'$.

**Proof.** Consider the four vertices on arbitrary face $F'$:

$$f_Q(x_D, x_F) = \sum_{i \leq j; i, j \in D_1} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \in D_1, j \in F'} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \leq j; i, j \in F'} Q_{i,j} \cdot x_i \cdot x_j$$

(27)

$$f_Q(x_D^C, x_F) = \sum_{i \leq j; i, j \in D_1^C} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \in D_1^C, j \in F'} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \leq j; i, j \in F'} Q_{i,j} \cdot x_i \cdot x_j$$

(28)

$$f_Q(x_D, x_F') = \sum_{i \leq j; i, j \in D_2} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \in D_2, j \in F'} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \leq j; i, j \in F'} Q_{i,j} \cdot x_i \cdot x_j$$

(29)

$$f_Q(x_D^C, x_F') = \sum_{i \leq j; i, j \in D_2^C} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \in D_2^C, j \in F'} Q_{i,j} \cdot x_i \cdot x_j + \sum_{i \leq j; i, j \in F'} Q_{i,j} \cdot x_i \cdot x_j$$

(30)

In the dominance test that compares $f_Q(x_D, x_F') + f_Q(x_D^C, x_F')$ to $f_Q(x_D, x_F') + f_Q(x_D^C, x_F')$, we can cancel the final, identical, term in Equations (27) – (30) and combine the first and second terms of diagonal $f_Q(x_D, x_F') - f_Q(x_D^C, x_F')$ and diagonal $f_Q(x_D, x_F') - f_Q(x_D^C, x_F')$.
Figure 7.: The point intersecting a 3D diagonal with a 3D corner slice is equidistant from \( v_1, v_2, \) and \( v_4 \) and one-third of the distance from \( v_0 \) to \( v_7 \).

to achieve:

\[
Q_{i,j}(x_i + x^C_i)x_j + \sum_{i,j \in D_1} Q_{i,j}(x_i x_j + x^C_i x^C_j) + \sum_{i,j \in D_2} Q_{i,j}(x_i + x^C_i)x_j
\]

where \( x^C_i \) is the complement of \( x_i \) (i.e., if \( x_i = x^L_i \) then \( x^C_i = x^U_i \)). Because the second summands on either side of the test are equal to one another \((x_i + x^C_i = x^L_i + x^U_i)\) no matter what the diagonal is, the test for dominance on face \( F' \) has reduced to a test for dominance on face \( F \). Therefore, we need not consider all \( 2^n - 1 \) projections of a diagonal on an \( i \)-face of an \( n \)-cube; the simplification in comparison (25) is justified.

Diagonal-to-corner comparisons

Recall that the Equation (26) dominance test compares the two diagonals at their intersecting midpoint [73]. Figure 7 illustrates the second GloMIQO 2 dominance test; GloMIQO 2 generalises the 3D diagonal to a 3D corner slice of Meyer and Floudas [73] to higher dimensions:

\[
\begin{align*}
(\binom{n}{3}) \cdot 2^3 & \quad \text{3D diagonals to 3D corner slices} \\
(\binom{n}{4}) \cdot 2^4 & \quad \text{4D diagonals to 4D corner slices} \\
& \quad \vdots \\
2^n & \quad n\text{-D diagonals to } n\text{-D corner slices}
\end{align*}
\]

There are \( 2^n - 1 \) diagonals in an \( n \)-cube and two corner slices per diagonal. As in Figure 7, the intersection point is equidistant from the points participating in the corner slice and \( 1/n \) the distance along the diagonal from the closer vertex. Analogous to Theorem 4.4.2.2, there are \( 2^{n-i} \cdot \binom{n}{i} \) \( i \)-faces in the \( n \)-cube, but we only need to consider \( \binom{n}{i} \) of them.

GloMIQO 2 only attempts edge-concave cuts on the collection of nonlinear terms in MIQCQP (Equation 4); it uses the technique described in Equation (21) to produce a maximally-violated edge-concave expression. Recall from Equation (21) that this resulting edge-concave expression has coefficients of 1 or \(-1\) depending on the relaxation solution and requires an underestimating cutting plane. GloMIQO 2 considers combinations of first 3 variables, then 4 variables, and then 5 variables to try to cutoff the current feasible point; the default GloMIQO behaviour is that if at least 5 dominant cuts have been generated with combinations of 3 (3 and 4) variables, then GloMIQO does not go on to test combinations of 4 (5) variables. The default behaviour is that only the most effective cut is saved for each aggregation of variables. After GloMIQO has developed a sufficient number of cuts (default maximum in GloMIQO: 1000), it returns and solves the augmented LP (e.g., it will not continue searching through combinations of 3 variables if it has already found 1000 combinations of 3 variables...
Table 2.: Comparison of the GloMIQO 2 Cut Classes

<table>
<thead>
<tr>
<th>Cut Class</th>
<th>Validity</th>
<th>Generation Complexity</th>
<th>Efficacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>RLT</td>
<td>Sec. 4.1</td>
<td>Global†</td>
<td>Evaluating an expression</td>
</tr>
<tr>
<td>Convexity</td>
<td>Sec. 4.2</td>
<td>Global</td>
<td>Evaluating an expression</td>
</tr>
<tr>
<td>αBB</td>
<td>Sec. 3.3</td>
<td>Global</td>
<td>Calculating the min. eigenvalue</td>
</tr>
<tr>
<td>Edge-Concave</td>
<td>Sec. 3.4</td>
<td>Local</td>
<td>Generating $n!$ facets</td>
</tr>
</tbody>
</table>

† If updated at each node [77]

which each yield a dominant cut).

GloMIQO 2 looking for low-dimensional cuts first is consistent with the computational work of Qualizza et al. [88] indicating the practical usefulness of cuts with fewer variables. The GloMIQO 2 strategy also fits with the result of Luedtke et al. [68] showing that density in the undirected graph representation of a bilinear expression generally correlates with deeper cuts. So GloMIQO 2 is effectively combining these two results when it looks for combinations of first 3, then 4, then 5 variables; it looks for low-dimensional cuts first in areas where the graph is dense and then applies the condition in Theorem 3.10 to screen the remaining cuts.

4.5. Comparison of the Cut Classes

Table 2 summarises Sections 4.1 – 4.4. The multivariable convex term cuts are the tightest and easiest to generate, but they require that convex multivariable terms exist in MIQCQP. The RLT cuts differ from GloMIQO 1.0 in that they allow addition of bilinear terms, the convex separating hyperplanes are equivalent to GloMIQO 1.0, and the αBB and the higher dimension edge-concave cuts are new to GloMIQO 2.

4.6. A Framework for Dynamically Generating Cuts

GloMIQO 1 is a spatial branch-and-bound global optimisation algorithm that solves an MILP relaxation of the MIQCQP at each node of the tree [77]. The primary addition in GloMIQO 2 is the framework for dynamically generating cuts diagrammed in Figure 8. Newly-generated cuts are stored in a cut pool and marked according to their global/local validity (see Figure 4). At the beginning of each branch-and-cut tree node, GloMIQO 2 generates a static mixed-integer linear program (MILP) relaxation of the MIQCQP [77]. After solving an LP relaxation of the MILP, GloMIQO 2 iteratively adds RLT constraints and previously-generated cuts. Globally valid cuts can always be applied; locally valid cuts are only applied when the current node descends from the generating node. When the cuts are no longer helping the LP objective function by a predefined $\epsilon$, GloMIQO enforces integrality of the discrete variables and solves the MILP relaxation using a sub-solver.

For the first three levels of the branch-and-bound tree, GloMIQO 2 follows Figure 8 and tries to generate convex hyperplanes and αBB cuts before generating more expensive vertex polyhedral cuts. After the first few levels of the branch-and-bound tree, GloMIQO 2 applies each cut class or not depending on whether that cut class previously tightened the LP relaxation (i.e., GloMIQO may modify Figure 8 in later nodes of the branch-and-cut tree).

5. Comparative Computational Studies

This section illustrates the GloMIQO 2 branch-and-cut strategies and uses a computational study to justify integrating the new framework. Section 5.1 describes the GloMIQO 2 treatment
Figure 8.: The GloMIQO 2 framework for dynamically generating cuts is used at every node of a branch-and-cut tree; recall that GloMIQO solves an MILP relaxation at each node [77].

of GLOBALLib Ex. 2.1.9, Section 5.2 presents computational results for randomly-generated QCQP, Section 5.3 presents results for process networks problems, and Section 5.4 discusses the performance of GloMIQO 2 on the 1041 bounded MIQCQP in its test set.

We present performance comparisons demonstrating the efficacy of each algorithmic component discussed in this manuscript. We also discuss the performance of GloMIQO 2 with respect to four state-of-the-art solvers: BARON 12.3.3, Couenne 0.4, LINDO 8.0, and SCIP 3.0; all four solvers are from the GAMS 24.1.2 distribution (released June 2013). It is important to note that these results represent a snapshot in time as MIQCQP solver technology develops. To represent solver evolution, Figures 14(d) and 14(e) diagram the solvers available in GAMS 24.2.1 (released December 2013).

Our computational studies were completed running 64 bit Linux on an Intel Xeon X5650 2.67 GHz processor with 24 GB RAM. We ran each computational experiment under 4 termination criteria: (1) a relative optimality gap $\epsilon = (UB - LB)/|LB| \leq 1 \times 10^{-6} = 1 \times 10^{-4}\%$; (2) an absolute optimality gap $\epsilon = UB - LB = 1 \times 10^{-6}$; (3) a 7200 s time limit; (4) an iteration limit of $9 \cdot 10^9$. No other parameters were altered from default. After solving each of the 1041 test cases using GAMS, a Perl script assert the feasibility of the solution returned by each solver. Instances where solvers returned infeasible points are automatically relabelled as failures ($fl$) (violating a variable bound or constraint equation by max$(10^{-4}, 10^{-4} \cdot |V|)$ or more where $V$ is the considered value).

Figures 10 and 14 were generated using GAMS Performance Tools with options: colselect; useobject; bnd=1e-4; gaptol=1e-4 (PTOOLS 1.1; http://www.gamsworld.org/performance/paver/; accessed 11 March 2013). The only processing done in transit between the solver-generated trace files and the PTOOLS input was to penalise solvers for returning infeasible solutions and add a column for the colselect option.

Subfigures 10(a) and 14(a) are the output of PTOOLS using solver reported time as a resource measurement and the object option to enforce an objective gap of $10^{-4}$. Paraphrasing the PTOOLS output, a solver is considered optimal if it has proper model and solver return codes and the relative objective value error is within $10^{-4}$ of the best possible solution. If the best found objective value is 0, then the absolute objective value error is used. Subfigures 10(b)
and 14(b) use the colselect command (and switch off useobjest); this allows us to compare the solvers with respect to their optimality gap at termination (100 · \( \frac{UB - LB}{LB} \)); this column in the GAMS trace file was created with the same Perl script that asserts feasibility of the solution.

It is important to note that the relevant characteristics of GloMIQO 2 change depending on the problem; GloMIQO must adapt on-the-fly to each new problem class. Therefore the analysis in Sections 5.1 – 5.3 will focus on individual problems or problem classes whereas Section 5.4 will demonstrate aggregate GloMIQO 2 performance.

As motivated in Figure 3 and Examples 3.17 – 3.19, it is naïve to assume that cuts affecting one class of MIQCQP will have a similar effect on structurally different classes of MIQCQP. In our experience, practically-relevant, well-written models each have interesting special mathematical structure; GloMIQO 2 dynamically determines solution strategies appropriate for a particular class. Section 5.2 highlights the Section 3 loop search and computationally motivates the Section 4.4 edge-concave cuts with an emphasis on the Section 4.4.2 reduced complexity approach. But the 239 problems featured in Section 5.2 are all from randomly generated test sets and therefore have somewhat similar structure. Other test sets show different advantages; Section 5.3 uses the process networks examples to highlight advantages in the αBB and RLT cutting planes. The convexity cuts described in Section 4.2 are motivated elsewhere [20, 21].

5.1. GLOBALLib Example 2.1.9

\[
\text{min } \sum_{i=1}^{9} x_i \cdot x_{i+1} + \sum_{i=1}^{8} x_i \cdot x_{i+2} + x_1 \cdot x_9 + x_1 \cdot x_{10} + x_2 \cdot x_{10} + x_1 \cdot x_5 + x_4 \cdot x_7 \\
\text{s.t. } \sum_{i=1}^{10} x_i = 1; \ x \in [0, 1]^{10}
\]

(32)

There are 22 bilinear terms in GLOBALLib Example 2.1.9 [43, 72]. The known global solution has objective value \(-0.375\). GloMIQO 1.0, BARON 12.3.3, and LINDO 8.0 report an initial relaxation of \(-22\), and the initial SCIP 3.0 relaxation is \(-4\). But the initial GloMIQO 2 relaxation is \(-0.5\); this is because GloMIQO 2 adds (via the analysis in Section 4.1), 33 nonlinear terms and 10 RLT-1 equations to the model formulation. GloMIQO 2.3 then solves Example 2.1.9 at the root node by introducing 81 univariate cuts (Equation 16), 472 αBB hyperplanes (Equation 20), and 118 edge-concave cuts. In comparison, GloMIQO 1.0 explores 249 nodes, BARON 12.3.3 uses 1615 nodes, LINDO takes 2 nodes, and SCIP 3.0 uses 3995 nodes.

5.2. QCQP: Box QP; Standard QP; Randomly-Generated QCQP

GLOBALLib Example 2.1.9 is a toy problem that all solvers address in seconds. To illustrate the advantages of the GloMIQO 2 branch-and-cut framework, we consider: (1) the Box-Constrained Quadratic Programming (BoxQP) problems of Vandenbussche and Nemhauser [113, 114]; (2) the Standard Quadratic Programming (StQP) test cases of Scozzari and Tardella [96]; (3) the 135 randomly-generated QCQP of Bao et al. [18]. The BoxQP test set was expanded to 90 problems by Burer and Vandenbussche [25]; we used the problems at http://dollar.biz.uiowa.edu/~sburer/pmwiki/pmwiki.php.html. There are 14 StQP case studies available at http://xoomer.virgilio.it/andreascozzari/test_StQP/. We transformed all 239 QP instances into GAMS format (availability: http://helios.princeton.edu/GloMIQO/).

Table 1 in the online supplement displays the relative complexity of the 90 BoxQP test cases (e.g., spar020-100-1 has 20 variables and 205 nonlinear terms). The problems are indexed sparNUMVARS- SPARSITY-EXAMPLE where the number of variables NUMVARS ranges from 20 – 100, the sparsity SPARSITY ranges from 25 – 100%, and the test EXAMPLE indexes
Figure 9: **239 BoxQP/StQP/QCQP**: Performance Profile illustrating the effect of knocking out each of the algorithmic components in GloMIQO 2

Figure 10: **239 BoxQP/StQP/QCQP**: Performance Profile (a) compares the time for solving 239 QP test cases; Profile (b) diagrams the percent gap remaining at 7200 s for the 239 test cases

the problem number (each complexity level has three similar problems). Online supplement Table 2 records the complexity of the 14 StQP instances. The StQP case studies are fully dense; different problems of the same size have varying convexity characteristics. Table 3 in the online supplement records the size of the 135 QCQP [18]; observe that there are several similar problems with each size demarcation.

Online supplement Table 5 reports complete analysis into the behaviour of the cutting plane framework for each of the 90 BoxQP, 14 StQP, and 135 QCQP. Table 4 in the online supplement summarises the framework output in its full form and without each of five critical algorithmic components. Individually knocking out each algorithmic component displays the relative contribution of that strategy for this class of MIQCQP. Figure 9 considers the performance of GloMIQO 2 in competition with: (1) naïve facet generation rather than the reduced complexity approach (i.e., the algorithm in Section 4.4.1 without the additional components in Section 4.4.2); (2) no αBB cuts (Section 4.3); (3) no convexity cuts (Section 4.2); (4) no edge-concave cuts (Section 4.4); (5) no loop search for whether a cycle may have a cut dominating the termwise relaxation (Theorem 3.10).

Observe in Figure 9 that, for these 239 examples with many interconnected bilinear terms and exclusively nonconvex nonlinear equations, the greatest contributing factor to the high performance of GloMIQO 2 is the presence of the dynamically-generated edge-concave facets. Using these edge-concave cuts, GloMIQO 2 solves 185 of the 239 problems whereas with-
out the edge-concave cuts GloMIQO 2 is only solving 112 of the 239 test cases. The second most important algorithmic component is replacing the naïve facet generation strategy with the reduced complexity approach; see from Figure 9 that a user has to be willing to accept a slow-down factor of $\approx 40$ before the naïve approach is equivalent to the reduced complexity approach. Although both the naïve and the reduced-complexity approach are exponential algorithms, the pre-factor associated with the reduced-complexity approach makes a significant impact. The final algorithmic component making a significant impact for these 239 problems is the loop search that checks the condition of Theorem 3.10; see in Figure 9 that we have to accept a slow-down factor of $\approx 2.3$ without the loop search.

It is important that the convexity and $\alpha$BB cuts, the two components that do not significantly help these 239 problems, do not significantly hurt the performance, either. The goal of designing a framework such that in GloMIQO 2 is that the appropriate cut classes activate at appropriate times and otherwise do not damage the framework. For other classes of problems, we find that other algorithmic components are the most important. For example, matching the results discussed by Berthold et al. [20, 21], we find that the convexity cuts are the most important class of cuts for MINLPLib [26, 43]. The convexity cuts are also important in the point packing problems [11] where a user has to accept $\approx 3$ times slower performance without the convexity cuts. As shown in Section 5.3, the $\alpha$BB cuts help close the optimality gap for water treatment system test cases [29, 31]; they also help tighten problems in GlobalLib [43, 72].

Tables 6 – 8 in the online supplement display complete computational results for the 90 BoxQP, 14 StQP, and 135 QCQP with respect to BARON 12.3.3 and Couenne 0.4; Figure 10 diagrams performance profiles for the 239 examples [37]. GloMIQO 2 is a powerful framework for addressing QCQP; it addresses $\approx 77\%$ of the problems to global optimality and is consistently the fastest solver. The next two most effective solvers, BARON 12.3.3 and Couenne 0.4, globally optimise $\approx 68\%$ and $\approx 40\%$ of the 239 problems. It is reasonable that BARON 12.3.3 and Couenne 0.4 are, for QP, the most effective solvers after GloMIQO 2; vertex polyhedral cuts similar to those documented in Section 4.4 are in BARON [18] and Couenne [88]. These results are a snapshot in time; BARON 12.7.3 (released December 2013) exhibits a positive step-change improvement on these 239 problems.

But it is important to note that the GloMIQO 2 branch-and-cut framework documented in this manuscript represents a significant advancement; begin by noting that GloMIQO 2 solves 17 more of the 135 randomly-generated QCQP than BARON 12.3.3; this test set was originally used to develop the BARON multiterm cuts [18]. This manuscript has focussed on perfecting an approach for cuts in lower dimensions than the ones designed by Bao et al. [18]; observe that the payoff is significantly faster performance than BARON 12.3.3. It is also evident that the GloMIQO 2 cutting plane framework is more effective than the one in Couenne which is also based on semidefinite programming but does not have any built-in intelligence with respect to which aggregates of bilinear terms will or will not have cuts deeper than their convex hull [88].

GloMIQO 2 solves 70 of the 90 BoxQP problems within a 2 hour time limit and addresses an additional 8 problems to within a 10% gap; given a 10 hour time limit, GloMIQO solves 72 problems to a $1 \times 10^{-4}\%$ gap. Observe that GloMIQO 1.0 globally optimises 19 of the 90 BoxQP problems within the 2 hour time limit; GloMIQO 1.0 and 2.3 perform equivalently on these BoxQP except for the branch-and-cut framework described in this manuscript. GloMIQO 2 is therefore comparable to the specialised QP solvers of Burer and Vandenbussche [25] and Chen and Burer [32] that address 78 and 85, respectively, of the 90 problems in a 10 hour time limit. An earlier algorithm designed by Vandenbussche and Nemhauser [113] was tested on a subset of the BoxQP problems and addressed 53 of the 54 problems; GloMIQO solves all 54 problems considered by Vandenbussche and Nemhauser [113] in a 2 hour time limit.
Figure 11: 32 Water Treatment Systems [29, 31]: Performance Profile illustrating the effect of knocking out each of the algorithmic components in GloMIQO 2

5.3. Process Networks Examples

The edge-concave, αBB, and convexity cuts analysed in Section 5.2 have either a positive or negligible effect on process networks problems (e.g., [7, 8, 29–31, 54, 56–62, 80–82, 91, 97, 110, 111]). For example, please see in Figure 11 that, with respect to solving speed, the Section 4.4.2 reduced complexity approach prevents the edge-concave cuts from damaging 32 water treatment system test cases [29, 31]. The αBB cuts help tighten the duality gap.

The RLT cuts, which were irrelevant in Section 5.2, also have an impact here. Figure 12 represents an analysis of all 141 process networks problems in the GloMIQO test set; the pooling transformations discussed in Misener et al. [77] work synergistically with the reformulation linearisation cuts (Section 4.1) to solve as many of the models and have the remaining models be as tight as possible. Further note in Figure 13 that these pre-processing transformations can be applied to other state-of-the-art global optimisation solvers by having GloMIQO 2 print out models equivalent to its pre-processing transformations. The GloMIQO transformations, in addition to improving GloMIQO, also improve three of the four other global optimisation solvers available in GAMS; the improvement is especially noticeable for Couenne.

Analysing Figure 13 in further detail, observe that the Final Root Node cyan marker is always exactly on top of the Trans + RLT marker; this means that the optimality-based bounds tightening (OBBT) operations in GloMIQO do not change the performance of the other four solvers. This is sensible; all four other solvers have optimality-based bounds tightening as one of their constitutive algorithms [19, 45, 47, 64, 108] and using the GloMIQO OBBT strategies is no additional advantage. BARON, Couenne and LINDO all show an advantage stemming from the pooling problem transformations [77], but only Couenne shows an additional advantage in adding the RLT cuts; this is in correspondence with recent BARON implementation of a variant of RLT cuts [120] and a lacking analogue within Couenne. There is no public documentation of incorporating RLT cuts within LINDO, but our internal testing suggests that the LINDO solver implemented some RLT variant after version 6.1; this deduction is due to the improved performance of LINDO 8.0 on GLOBALLib ex5_2_5 (see Misener and Floudas [77] for a discussion on RLT and GLOBALLib ex5_2_5). Therefore, the improved performance of BARON, Couenne and LINDO in Figure 13 strongly reinforces the synergistic interplay between pooling transformations and RLT cuts.

5.4. Large Scale Computational Comparisons

Although Figure 10 shows that there is a strong advantage for QP and randomly-generated QCQP in using the proposed branch-and-cut framework, we also need to show that the framework does not hinder the performance of GloMIQO 2 on other MIQCQP classes. Figure 14
Figure 12.: **141 Process Networks Problems**: Performance Profile illustrating the effect of several pre-processing algorithmic components in GloMIQO 2

therefore diagrams the performance profiles for the complete bounded test set of GloMIQO 2 (1041 MIQCQP case studies outlined in Table 3 and fully documented at http://helios.princeton.edu/GloMIQO/). GloMIQO 2 takes no performance hit on other test problems due to its branch-and-cut framework. Figure 14 also represents the dynamics in MIQCQP solver software; Subfigures (a) and (b) were created using the solvers available in GAMS 24.1.2 whereas (d) and (e) are the corresponding figures for GAMS 24.2.1.

6. Conclusion

We have presented the GloMIQO 2 branch-and-cut framework; GloMIQO 2 is available in the GAMS modelling system (beginning version 23.9; current GAMS version is GloMIQO 2.3). The contributions of this paper include: development of a necessary and sufficient condition for there to exist a cutting plane dominating the termwise relaxation, justification and algorithmic description of the GloMIQO 2 cut classes, and organisation of the cut classes into a cohesive framework. From the computational study, we see a strong advantage to integrating many cut classes into a whole.

Acknowledgements

C. A. F. is thankful for support from the National Science Foundation (CBET – 0827907) and the National Institutes of Health R01 GM052032. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship to R. M. under Grant No. DGE-0646086 and by a Royal Academy of Engineering Research Fellowship awarded to R. M. J. B. S. was supported by the National Institutes of Health Grants P50GM071508-06 and R01 GM052032.

References


Figure 13: **133 Process Networks Problems**: Performance profiles comparing the performance of four state-of-the-art solvers when given the GloMIQO 2 preprocessing strategies; these are all the process networks problems that GloMIQO 2 does not solve at the root node. *Transformations only* indicates that only the pooling transformations discussed in Misener et al. [77] were applied; *Trans + RLT* includes the RLT cuts; *Final Root Node* also includes all optimality-based bounds tightening operations.


Table 3.: MINLP Test Suite of 1041 Problems

<table>
<thead>
<tr>
<th>Problem Class</th>
<th># Cases</th>
<th>Discrete</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Networks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crude Oil Scheduling</td>
<td>31</td>
<td>✓</td>
<td>[58–60, 81, 82]</td>
</tr>
<tr>
<td>Multi-Period Blending</td>
<td>7</td>
<td>✓</td>
<td>[56, 57]</td>
</tr>
<tr>
<td>Natural Gas Production</td>
<td>3</td>
<td>✓</td>
<td>[61, 62, 97]</td>
</tr>
<tr>
<td>Pooling</td>
<td>19</td>
<td>✓</td>
<td>[80]</td>
</tr>
<tr>
<td>Supply Chain Design</td>
<td>1</td>
<td>✓</td>
<td>[115, 116]</td>
</tr>
<tr>
<td>Unit Commitment</td>
<td>1</td>
<td>✓</td>
<td>[83, 119]</td>
</tr>
<tr>
<td>Water Treatment Network</td>
<td>8</td>
<td>✓</td>
<td>[7, 8, 54, 91]</td>
</tr>
<tr>
<td>Water Treatment System</td>
<td>32</td>
<td></td>
<td>[29, 31]</td>
</tr>
<tr>
<td>Water-Using Network</td>
<td>35</td>
<td></td>
<td>[30, 110, 111]</td>
</tr>
<tr>
<td>Computational Geometry</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Circles &amp; Polygon Nesting</td>
<td>36</td>
<td></td>
<td>[53]</td>
</tr>
<tr>
<td>Nesting</td>
<td>5</td>
<td></td>
<td>[51]</td>
</tr>
<tr>
<td>Point Packing</td>
<td>14</td>
<td></td>
<td>[11]</td>
</tr>
<tr>
<td>QCQP Test Sets</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BoxQP</td>
<td>90</td>
<td></td>
<td>[25, 113, 114]</td>
</tr>
<tr>
<td>Randomly-Generated QCQP</td>
<td>135</td>
<td></td>
<td>[18]</td>
</tr>
<tr>
<td>Standard QP</td>
<td>14</td>
<td></td>
<td>[96]</td>
</tr>
<tr>
<td>Reformulated Libraries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reform. GLOBALLib</td>
<td>32</td>
<td>✓</td>
<td>[43, 72, 93]</td>
</tr>
<tr>
<td>Reform. MINLPLib</td>
<td>10</td>
<td>✓</td>
<td>[26, 43]</td>
</tr>
<tr>
<td>Test Libraries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bonmin Test Set</td>
<td>20</td>
<td>✓</td>
<td>[23, 92]</td>
</tr>
<tr>
<td>GLOBALLib</td>
<td>172</td>
<td></td>
<td>[43, 72]</td>
</tr>
<tr>
<td>LINLib</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MacMOOPLib</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPLLib</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MINLPLib</td>
<td>102</td>
<td>✓</td>
<td>[26, 43]</td>
</tr>
<tr>
<td>PrincetonLib</td>
<td>146</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QAP</td>
<td>67</td>
<td></td>
<td>[10, 66]</td>
</tr>
</tbody>
</table>

References:

REFERENCES

Figure 14.: 1041 MIQCQP: Performance Profiles (a) and (d) compare the time for solving the 1041 bounded MIQCQP problems in the GloMIQO 2 test set; Profiles (b) and (e) diagram the percent gap remaining at 7200 s for the 1041 test cases. Performance profiles (a) and (b) were created using the solvers available in GAMS 24.1.2; (d) and (e) are the corresponding figures for GAMS 24.2.1

REFERENCES


[58] L. Liberti and C.C. Pantelides, An exact reformulation algorithm for large nonconvex NLPs involving bilinear
REFERENCES

33

REFERENCES


