Journal of Global Optimization **7**: 1–31, 1995.

A Reformulation-Convexification Approach for Solving Nonconvex Quadratic Programming Problems

HANIF D. SHERALI and CIHAN H. TUNCBILEK
Department of Industrial and Systems Engineering Virginia Polytechnic Institute and State University Blacksburg, Virginia 24061-0118, U.S.A.

(Received: 27 January 1994; accepted: 9 February 1995)

Abstract. In this paper, we consider the class of linearly constrained nonconvex quadratic programming problems, and present a new approach based on a novel Reformulation-Linearization/Convexification Technique. In this approach, a tight linear (or convex) programming relaxation, or outer-approximation to the convex envelope of the objective function over the constrained region, is constructed for the problem by generating new constraints through the process of employing suitable products of constraints and using variable redefinitions. Various such relaxations are considered and analyzed, including ones that retain some useful nonlinear relationships. Efficient solution techniques are then explored for solving these relaxations in order to derive lower and upper bounds on the problem, and appropriate branching/partitioning strategies are used in concert with these bounding techniques to derive a convergent algorithm. Computational results are presented on a set of test problems from the literature to demonstrate the efficiency of the approach. (One of these test problems had not previously been solved to optimality.) It is shown that for many problems, the initial relaxation itself produces an optimal solution.

Key words: Quadratic programming, indefinite quadratic problems, reformulation-linearization technique, reformulation-convexification approach, outer-approximations, tight linear programming relaxations.

1. Introduction

This paper deals with finding a global optimal solution for nonconvex quadratic programming problems of the form

$$\begin{aligned} \text{QP}: \ & \text{Minimize} \ cx + x^t Qx \\ & \text{subject to} \ Ax \leqslant b \\ & \qquad 0 \leqslant l_k \leqslant x_k \leqslant u_k < \infty \quad k = 1, \ldots, n, \end{aligned}$$

where $c \in \mathbb{R}^n$, A is an $m \times n$ matrix, $b \in \mathbb{R}^m, Q$ is an $n \times n$ indefinite matrix, assumed to be symmetric for convenience, and where the decision variables are $x \in \mathbb{R}^n$. Our proposed methodology is equally applicable to the case where Q is a negative semidefinite matrix; however, for this pure concave case, we do not take advantage of the extremality property of the global optimum, and treat it just as another non-convex instance of the problem. Notice that we have assumed the existence of finite lower and upper bounds on all the variables. These might be prespecified bounds, or they might be derived from the other constraints of QP and imposed herein for certain algorithmic purposes. As we proceed, we will point out modifications to be made in our algorithmic constructs whenever such bounds are known to be implied by the other constraints defining the problem, but are not explicitly specified.

Pardalos and Vavasis (1991) have shown that quadratic programs are NP-hard, even when Q has just one negative eigenvalue. These problems, which arise variously in applications such as in modelling economies of scale in a cost structure, in location-allocation problems, VLSI design problems, some production planning and risk management problems, and in various other mathematical models such as the maximum clique problem and the jointly constrained bilinear programming problem, have therefore been both interesting and challenging problems to solve (see the surveys of Pardalos and Rosen (1987) and Pardalos (1991)). Earlier work in pursuit of solving general quadratic programming problems includes that of Ritter (1966), who gave an algorithm based on Tuy-cuts for concave programming, Manas (1968), who gave an algorithm based on the enumeration of vertices of the feasible region, and Mueller (1970), who used gradient projection searches in an adjacent extreme point framework. Zwart (1973) has subsequently shown via a counter example that Ritter's algorithm may converge to a non-global optimum. Balas (1975) introduced generalized polar sets, and Benacer and Pham Dinh Tao (1986) used reverse convex constraints in a cutting plane framework to find the minimum Karush–Kuhn–Tucker point. Considering QP in its transformed polar form, Kough (1979) used a generalized Bender's approach, while Tuy (1987) specialized a cutting plane procedure that was developed to solve DC-programming problems. Bomze (1992) presented sufficient optimality conditions based on ε-subdifferential calculus. Pardalos, Glick and Rosen (1987), and Phillips and Rosen (1990) employed eigen-transformations to construct a lower bounding problem using the convex envelope of the concave terms over the bounding constraints, and have developed algorithms based on partitioning the bounding intervals. Using a similar lower bounding problem, Vavasis (1992) has proposed an ε-optimum algorithm that is polynomial in $1/\varepsilon$ for a fixed number of concave variables. For mixed integer quadratic programs, Al-Khayyal and Larsson (1990) developed two forms of piecewise affine convex underestimating functions for linearizing the objective function in a branch-and-bound context. Floudas and Visweswaran (1990, 1993) proposed an ε-finite branch-and-bound algorithm employing a generalized Benders type of approach, where Benders' cuts are replaced by suitable implied Lagrange functions. Specializing this algorithm for solving certain classes of problems, including indefinite and concave programming problems, Visweswaran and Floudas (1993) have presented computational results on solving randomly generated indefinite QP's having up to 25 nonlinear variables and 100 linear variables, and a separable objective function.

For the equivalent, jointly constrained bilinear programming problem, Muu and Oettli (1991) have proposed a branch-and-bound algorithm using simplex bisection bisections and a decomposition based bounding problem. This approach can also handle convex constraints. Al-Khayyal and Falk (1983) have developed a branch-and-bound algorithm using the convex envelope of each bilinear term to generate bounding problems. These bounds were significantly improved by Sherali and Alameddine (1992) who designed a *Reformulation-Linearization Technique (RLT)* to obtain tight lower bounding linear programs. Later, Sherali and Tuncbilek (1992) generalized this algorithm to solve the class of polynomial programming problems.

In this paper, we investigate various specialized RLT designs for generating both linear (RLT-LP) and convex, nonlinear (RLT-NLP) lower bounding problems for QP, that can be suitably embedded within a branch-and-bound procedure. In general, we call this process of generating relaxations a *Reformulation-Convexification* approach. The first scheme we propose involves the generation of quadratic constraints through a construction of pairwise constraint products. These constraints are subsequently linearized to yield a lower bounding linear program. We show that the resulting linear program that involves all such products is invariant under affine transformations. However, if eigen-transformation is used as a particular linear transformation, this enables us to introduce additional nonlinear convex constraints that can further strengthen the linear programming relaxation produced by RLT. Such nonlinear constraints can be suitably handled within a Lagrangian dual procedure, without hampering the efficiency of the solution procedure as compared with that for solving the linear programming bounding problem. We also present a rule for reducing the number of new second-order constraints generated for the bounding problem, without compromising much on the quality of the resulting lower bound.

Following this development, the paper discusses implementation issues and strategies for a hybrid best-first and depth-first branch-and-bound algorithm for solving QP. Branching is performed based on the partitioning of the box constraints defined by the bounds on each variable. Lower and upper bounds on QP are then derived using the designed RLT relaxations. Several other algorithmic strategies are gainfully employed in solving the problem. In particular, the Lagrangian dual formulation is enhanced using a layering strategy, and various simple strategies are devised for range-restricting variables as well as to tighten the lower bounds in order to enhance the fathoming efficiency of the algorithm. Guidelines for other implementation details, along with computational experience on test problems from the literature, are also provided. In particular, for one of the test problems having 20 variables and 10 constraints, our procedure finds an improved solution (within 0.1% of optimality) than has been previously reported in the literature.

2. Reformulation via Quadratic Constraint Generation (RLT-LP)

In this section, we generate quadratic implied constraints, and subsequently linearize them to obtain RLT constraints. To generate the quadratic constraints, we consider pairwise products of so-called bound-factors and constraint-factors defined by individual variable bounds and structural constraints in QP. For brevity of presentation, let us combine bound and constraint-factors in a single set as follows:

$$\begin{bmatrix} (b_i - a_i x) \geqslant 0, \ i = 1, \ldots, m \\ (u_k - x_k) \geqslant 0, \ k = 1, \ldots, n \\ (x_k - l_k) \geqslant 0, \ \ k = 1, \ldots, n \end{bmatrix} \equiv \begin{bmatrix} (g_i - G_i x) \geqslant 0 \\ i = 1, \ldots, m + 2n \end{bmatrix} \tag{1}$$

where $a_i x \leqslant b_i$ is the i^{th} (structural) constraint from $Ax \leqslant b$, for $i = 1, \ldots, m$. At the reformulation step, we take all possible pairwise products of the factors in (1), including self-products, to generate the following nonlinear implied constraints that are included in the original problem QP:

$$(g_i - G_i x)(g_j - G_j x) \geqslant 0 \quad \forall 1 \leqslant i \leqslant j \leqslant m + 2n. \tag{2}$$

We then linearize the resulting augmented problem by substituting

$$w_{kl} = x_k x_l \quad \forall 1 \leqslant k \leqslant l \leqslant n. \tag{3}$$

This substitution associates a separate new variable with each distinct nonlinear term in the problem. We often refer to these variables as *RLT variables*. The resulting problem is called the *first-level* or *first-order RLT*, since it employs first-order (linear) factors to generate new constraints. Denoting $[(g_i - G_i x)(g_j - G_j x)]_\ell \leqslant 0, \quad \forall 1 \leqslant i \leq j \leqslant m + 2n$ as the resulting linearized constraints, we obtain the following lower bounding first-level RLT linear program, where $q_{kl} (= q_{lk})$ is the $(k, l)^{\text{th}}$ element of the symmetric matrix Q.

$$\mathbf{RLT-LP}: \text{ Minimize } \sum_{k=1}^{n} c_k x_k + \sum_{k=1}^{n} q_{kk} w_{kk} + 2 \sum_{k=1}^{n-1} \sum_{l=k+1}^{n} q_{kl} w_{kl} \tag{4.1}$$

$$\text{subject to } [(g_i - G_i x)(g_j - G_j x)]_\ell \geqslant 0 \quad \forall 1 \leqslant i \leqslant j \leqslant m + 2n. \tag{4.2}$$

Notice that the original constraints of QP are not included in RLT-LP, since, as we shall show in Proposition 1 later, these constraints are implied by the RLT constraints (4.2), even if the feasible region is not assumed to be bounded, provided that at least one variable has a bounded range over the feasible region.

For any feasible solution to problem QP, there exists a feasible solution to RLT-LP having the same objective function value through the definitions (3). However, the converse is not necessarily true. Therefore, RLT-LP is a relaxation of QP that yields a lower bound on the global minimum of QP. Moreover, if $(\overline{x}, \overline{w})$ solves

RLT-LP, then since $\overline{x}$ is feasible to QP, it provides an upper bound on this problem. In particular, if this solution also satisfies the definitions (3) for all the nonlinear terms appearing in QP, then $\overline{x}$ solves QP.

REMARK 1. *(Equality Constraints)*: If there is some equality constraint $G_e x = g_e$ in QP, then we only need to consider the product of the constraint-factor $(g_e - G_e x) = 0$ with each variable x_k, $k = 1, \ldots, n$, since all the other RLT constraints generated via this constraint can be obtained by suitably surrogating the constraints $[x_k(g_e - G_e x)]_\ell = 0, k = 1, \ldots, n$.

3. An Illustrative Example

To provide some insights into the RLT process and its effects, consider the following illustrative concave quadratic program:

$$\text{Minimize } \{z = -(x_1 - 12)^2 - x_2^2 : -6x_1 + 8x_2 \leqslant 48,$$
$$3x_1 + 8x_2 \leqslant 120, 0 \leqslant x_1 \leqslant 24, x_2 \geqslant 0\}.$$

The optimal solution to this problem is $(x_1^*, x_2^*) = (24, 6)$, or alternatively, (0, 6), and the optimal objective function value is $z^* = -180$. The feasible region happens to be bounded, and given by the convex hull of the vertices (0, 0), (0, 6), (8, 12), (24, 6), and (24, 0). Before generating the first-level RLT for this problem, let us identify the bound-factors as $(24 - x_1) \geqslant 0, x_1 \geqslant 0, x_2 \geqslant 0$, and denote the constraint-factors as $s_1 = (48 + 6x_1 - 8x_2) \geqslant 0$, and $s_2 = (120 - 3x_1 - 8x_2) \geqslant 0$. The first-level RLT problem is then generated by constructing the 15 pairwise (including self-products) of these factors, and then linearizing the resulting quadratically constrained QP via the variable redefinitions $w_{11} = x_1^2, w_{12} = x_1x_2$, and $w_{22} = x_2^2$.

The solution to this first-level RLT problem is given by $(\overline{x}_1, \overline{x}_2, \overline{w}_{11}, \overline{w}_{12}, \overline{w}_{22}) = (8, 6, 192, 48, 72)$ and has an objective function value of -216. Hence, -216 provides a lower bound on QP. Notice that $\overline{x}_1\overline{x}_2 = \overline{w}_{12}$; however, the same is not true for $\overline{w}_{11} = 192 \neq \overline{x}_1^2 = 64$, and $\overline{w}_{22} = 72 \neq \overline{x}_2^2 = 36$. Notice also that $\overline{x} = (8, 6)$ is feasible to QP and has an objective function value of -52. This provides an upper bound on QP.

Now, let us partition the problem via the dichotomy $x_1 \leqslant 8$ or $x_1 \geqslant 8$. When we solve a first-level RLT problem separately for each partitioned subproblem, the optimal value turns out to be -180 in both cases. That is, the original problem is solved after a single branching on x_1.

To show what RLT is trying to achieve, let us introduce cubic RLT constraints. In the following problem, we consider certain selected second-order RLT constraints along with some cubic bound/constraint-factor products to yield additional RLT constraints. (This selection is done for the purpose of illustration, and is motivated by the magnitude of the optimal dual variable values of the first-level RLT problem.) The resulting linear program given below is a (partial) second-level RLT problem, where the additional RLT variables represent the cubic terms $w_{111} = x_1^3, w_{112} = x_1^2 x_2$, and $w_{122} = x_1 x_2^2$.

$$\begin{aligned}
\text{Minimize} \quad & -w_{11} - w_{22} + 24x_1 - 144 \\
\text{subject to} \quad & [(24 - x_1)s_1]_\ell = 1152 + 96x_1 - 192x_2 - 6w_{11} + 8w_{12} \geqslant 0 \\
& [x_1 s_2]_\ell = 48x_1 + 6w_{11} - 8w_{12} \geqslant 0 \\
& [(24 - x_1)x_1]_\ell = 24x_1 - w_{11} \geqslant 0 \\
& [(24 - x_1)x_1 s_1]_\ell = 1152x_1 + 96w_{11} - 192w_{12} - 6w_{111} \\
& \quad +8w_{112} \geqslant 0 \\
& [(24 - x_1)x_2 s_1]_\ell = 1152x_2 + 96w_{12} - 192w_{22} - 6w_{112} \\
& \quad +8w_{122} \geqslant 0 \\
& [(24 - x_1)x_1 s_2]_\ell = 2880x_1 - 192w_{11} - 192w_{12} + 3w_{111} \\
& \quad +8w_{112} \geqslant 0 \\
& [x_1 x_2 s_2]_\ell = 120w_{12} - 3w_{112} - 8w_{122} \geqslant 0.
\end{aligned}$$

The solution to this revised problem is given by $(\hat{x}_1, \hat{x}_2) = (0, 6), \hat{w}_{22} = 36, \hat{w}_{11} = \hat{w}_{12} = \hat{w}_{111} = \hat{w}_{112} = \hat{w}_{122} = 0$, having the objective function value -180. Hence, the above linear bounding problem composed of selected second and third-order RLT constraints solves the original problem. In fact, we can demonstrate in this instance that these constraints effectively describe the convex envelope of the objective function for the original concave quadratic program.

To see this, let us define a particular surrogate of the above third-order RLT constraint as follows.

$$\begin{aligned}
\mathit{surr}[\mathit{cubic}]_\ell \equiv \; & 1/512[(24 - x_1)x_1 s_1]_\ell + 1/192[(24 - x_1)x_2 s_1]_\ell \\
& +1/256[(24 - x_1)x_1 s_2]_\ell + 1/192[x_1 x_2 s_2]_\ell \geqslant 0.
\end{aligned}$$

When we further surrogate this constraint with the objective function, and the second-order RLT constraints using two different sets of weights in turn, we obtain the following two constraints that yield the desired convex envelope representation.

$$\begin{aligned}
& (z + w_{11} + w_{22} - 24x_1 + 144 = 0) + 7/16[(24 - x_1)x_1]_\ell \geqslant 0) \\
& \quad +(\mathit{surr}[\mathit{cubic}]_\ell \geqslant 0) \Rightarrow z \geqslant -6x_2 - 144,
\end{aligned}$$

$$\begin{aligned}
& (z + w_{11} + w_{22} - 24x_1 + 144 = 0) + 7/144([(24 - x_1)s_1]_\ell \geqslant 0) \\
& \quad +7/144([x_1 s_2]_\ell \geqslant 0) + (\mathit{surr}[\mathit{cubic}]_\ell \geqslant 0) \Rightarrow z \geqslant (10/3)x_2 - 200.
\end{aligned}$$

Hence, it turned out that the given QP was solved as the above single linear program. In fact, as this example exhibits, the RLT scheme attempts to approximate the convex envelope of the objective function over the feasible region in deriving a lower bounding linear program. If this approximation is composed properly, a tight linear representation of the problem can be derived.

4. Insights into the First-Level RLT Application

In this section, we present some results that provide insights into the first-level RLT problem. Some implications of these results will find direct use in designing a suitable branch-and-bound algorithm. In concert with definition (1), we refer to the constraint set of QP, including the simple bound restrictions on the variables, as $Gx \leqslant g$.

The first result below shows that the original constraints $Gx \leqslant g$ need not be included in the first-level RLT problem, even under a relaxed boundedness assumption on QP.

PROPOSITION 1. *Suppose that the feasible region of QP is not necessarily bounded (possibly including unrestricted variables) but that for some variable* $k \in \{1, \ldots, n\}$, *we have*

$$U_k \equiv \max\{x_k : G_i x \leqslant g_i, i = 1, \ldots, m+2n\} < \infty \tag{5.1}$$

$$L_k \equiv \min\{x_k : -G_i x \geqslant -g_i, i = 1, \ldots, m+2n\} > -\infty \tag{5.2}$$

where $U_k > L_k$. *Then, the original constraints* $Gx \leqslant g$ *are implied by the RLT constraints* $[(g_i - G_i x)(g_j - G_j x)]_\ell \geqslant 0 \quad \forall 1 \leqslant i \leqslant j \leqslant m+2n$.

Proof. Let $\mu^u \geqslant 0$ and $\mu^\ell \geqslant 0$ be the optimal dual multiplier vectors associated with the constraints in (5.1) and (5.2), respectively. Then, we have $G^t\mu^u = -G^t\mu^\ell = e_k$, where $e_k \in \mathbb{R}^n$ is the unit vector with entry 1 at the k^{th} position, and also, $g^t\mu^u = U_k$, and $-g^t\mu^\ell = L_k$.

Now, for any $j \in \{1, \ldots, m+2n\}$, consider the surrogate of all RLT constraints involving the constraint-factor $(g_j - G_j x \geqslant 0)$, obtained by using the weights μ^u.

$$\begin{aligned} 0 &\leqslant \sum_{i=1}^{m+2n} \mu_i^u[(g_i - G_i x)(g_j - G_j x)]_\ell \\ &= \sum_{i=1}^{m+2n} \mu_i^u(g_i g_j - g_i G_j x - g_j G_i x + [(G_i x)(G_j x)]_\ell) \\ &= -g_j \sum_{i=1}^{m+2n} \mu_i^u G_i x + (g_j - G_j x)\sum_{i=1}^{m+2n} \mu_i^u g_i + \sum_{i=1}^{m+2n} \mu_i^u G_i [xx^t]_\ell G_j^t \\ &= -g_j x_k + (g_j - G_j x)U_k + e_k^t[xx^t]_\ell G_j^t. \end{aligned}$$

After rearranging the terms in the final expression, and noting that $e_k^t[xx^t]_\ell G_j^t = [x_k(G_j x)]_\ell$, we obtain,

$$U_k(g_j - G_j x) - [x_k(g_j - G_j x)]_\ell = [(U_k - x_k)(g_j - G_j x)]_\ell \geqslant 0. \tag{6.1}$$

Replacing μ^u by μ^ℓ, and then following the same steps as above, we obtain,

$$-L_k(g_j - G_jx) + [x_k(g_j - G_jx)]_\ell = [(x_k - L_k)(g_j - G_jx)]_\ell \geqslant 0. \quad (6.2)$$

The surrogate of (6.1) and (6.2) gives $(U_k - L_k)(g_j - G_jx) \geqslant 0$, which implies that $G_jx \leqslant g_j$, since $U_k > L_k$. Since this is true for any $j \in \{1, \ldots, m + 2n\}$, including the bounding constraints, the proof is complete. ■

Consider the constraints $Gx \leqslant g$, and suppose that some of these constraints are implied by the others. In particular, let us assume that the constraints $G_ix \leqslant g_i, i \in \{1, \ldots, m'\}$ imply the other constraints within $Gx \leqslant g$, and represent, say a minimal set of non-implied constraints. The second proposition below exhibits that we do not need to use any implied constraint in generating RLT constraints, because such constraints would be implied by the RLT constraints that are generated via the products of the non-implied constraints. Hence, any constraints within $Gx \leqslant g$ that are known to be implied can be discarded without loss of any tightness in the resulting RLT problem, and moreover, it is futile to use any implied constraint, such as implied bounds on variables, to generate additional RLT constraints. The following result encapsulates the foregoing comment. For convenience, let us refer to the RLT constraints generated via the non-implied set of original constraints as

$$\Gamma \equiv \{[(g_i - G_ix)(g_j - G_jx)]_\ell \geqslant 0 \quad \forall 1 \leqslant i \leqslant j \leqslant m'\}. \quad (7)$$

PROPOSITION 2. *Suppose that the assumption of Proposition 1 holds, and let* $\alpha x \leqslant \beta$ *be implied by the constraints* $G_ix \leqslant g_i, i = 1, \ldots, m'$. *Then, the RLT constraints* $[(\beta - \alpha x)(g_i - G_ix)]_\ell \geqslant 0, i = 1, \ldots, m'$, *and* $[(\beta - \alpha x)^2]_\ell \geqslant 0$ *are also implied by the RLT constraints* $[(g_i - G_ix)(g_j - G_jx)]_\ell \geqslant 0, 1 \leqslant i \leqslant j \leqslant m'$. *In particular, the constraints of the first-level RLT problem are all implied by the RLT constraints contained within* Γ.

Proof. Since $\beta \geqslant \text{maximum}\{\alpha x : G_ix \leqslant g_i, i = 1, \ldots, m'\}$, there exist dual multipliers $\mu_i \geqslant 0, i = 1, \ldots, m'$ such that

$$\sum_{i=1}^{m'} \mu_i G_i = \alpha \quad \text{and} \quad \sum_{i=1}^{m'} \mu_i g_i \leqslant \beta. \quad (8)$$

Now, for any $j \in \{1, \ldots, m'\}$, consider the surrogate of the following RLT constraints from Γ involving the constraint-factor $(g_j - G_jx) \geqslant 0$, obtained by using the weights μ.

$$\begin{aligned} 0 &\leqslant \sum_{i=1}^{m'} \mu_i[(g_i - G_ix)(g_j - G_jx)]_\ell \\ &= \sum_{i=1}^{m'} \mu_i(g_ig_j - g_iG_jx - g_jG_ix + [(G_ix)(G_jx)]_\ell) \end{aligned}$$

$$= -g_j \sum_{i=1}^{m'} \mu_i G_i x + (g_j - G_j x) \sum_{i=1}^{m'} \mu_i g_i + \sum_{i=1}^{m'} \mu_i G_i [xx^t]_\ell G_j^t$$

Since by Proposition 1, $(g_j - G_j x) \geqslant 0$ is implied by Γ, we get upon using (8) that

$$0 \leqslant -g_j \alpha x + \beta(g_j - G_j x) + \alpha[xx^t]_\ell G_j^t = [(\beta - \alpha x)(g_j - G_j x)]_\ell.$$

Hence, $[(\beta - \alpha x)(g_j - G_j x)]_\ell \geqslant 0$ is implied by Γ for all $j = 1, \ldots, m'$. Furthermore, by following the same algebraic steps as above, and using the foregoing assertion, we get $0 \leqslant \sum_{j=1}^{m'} \mu_j [(g_j - G_j x)(\beta - \alpha x)]_\ell \leqslant [(\beta - \alpha x)(\beta - \alpha x)]_\ell$. Hence, the self-product constraint $[(\beta - \alpha x)^2]_\ell \geqslant 0$ is also implied by Γ. The final assertion of the proposition now follows by inductively taking the implied constraints of $Gx \leqslant g$ one at a time, and establishing as above that the RLT constraints generated via this constraint, including the self-product constraint, are implied by the RLT constraints generated via the other remaining constraints. Discarding such implied constraints sequentially, we deduce that the constraint set Γ implies the other RLT constraints, and this completes the proof. ■

Knowing that the original constraints are implied by the RLT constraints, if an original constraint from $Gx \leqslant g$ is binding at some feasible solution $(\overline{x}, \overline{w})$ to RLT-LP, then one would expect that some of the RLT constraints would also be binding at $(\overline{x}, \overline{w})$. The next result shows that the RLT constraints generated using a constraint-factor that turns out to be binding, are themselves binding. However, for this result, we need the original assumption that the feasible region of QP is bounded.

PROPOSITION 3. *Assume that the feasible region of QP is bounded, and suppose that at a given point* $(\overline{x}, \overline{w})$, *we have* $G_i \overline{x} = g_i$, *for some* $i \in \{1, \ldots, m + 2n\}$. *Then, the linearized product of this constraint with any original variable* $x_k, k \in \{1, \ldots, n\}$ *at* $(\overline{x}, \overline{w})$ *is zero. That is, denoting* $[\cdot]_\ell$ *evaluated at* $(\overline{x}, \overline{w})$ *by* $[\cdot]_\ell|_{(\overline{x}, \overline{w})}$, *we have,*

$$[x_k(g_i - G_i x)]_\ell|_{(\overline{x},\overline{w})} \equiv g_i \overline{x}_k - \sum_{i=1}^{k} G_{il} \overline{w}_{lk} - \sum_{l=k+1}^{n} G_{il} \overline{w}_{kl} = 0, \quad \forall k = 1, \ldots, n. \tag{9}$$

In particular, the RLT constraint generated by multiplying this constraint with any other constraint is also binding.

Proof. Under the boundedness assumption of the feasible region, consider the constraints (6.1) and (6.2), which are implied by or which already exist in Γ by Proposition 2, in a combined form as follows:

$$L_k(g_i - G_i x) \leqslant [x_k(g_i - G_i k)]_\ell \leqslant U_k(g_i - G_i x) \quad \forall k = 1, \ldots, n. \tag{10}$$

Evaluating (10) at $(\overline{x}, \overline{w})$, since $g_i - G_i\overline{x} = 0$, we obtain (9) holding true. For any $j \in \{1, \ldots, m+2n\}$, when we evaluate the following RLT constraint at $(\overline{x}, \overline{w})$,

$$\begin{aligned}[(g_i - G_i x)(g_j - G_j x)]_\ell &= g_j(g_i - G_i x) \\ &\quad - \sum_{k=1}^{n} G_{jk}[x_k(g_i - G_i x)]_\ell \geqslant 0 \end{aligned} \tag{11}$$

we get by (9) and $(g_i - G_i\overline{x}) = 0$, that (11) holds as equality. This completes the proof. ■

Next, we address the question whether an application of RLT after using some affine transformation can possibly produce a different relaxation. This might be of interest, in particular, if one wished to investigate the effect of applying RLT to different nonbasic space representations of the linear constraint defining QP, or the effect of employing an eigen-transformation on QP before applying RLT. More specifically, given the quadratic programming problem QP: minimize$\{cx + x^tQx : Gx \leqslant g\} < \infty$, define a nonsingular affine transformation $s = Bx + p$ to represent QP in s-space as follows, using the substitution $x \equiv B^{-1}s - B^{-1}p$, and where $B^{-t} \equiv (B^{-1})^t$.

$$\begin{aligned} \text{QP}' : -cB^{-1}p + p^tB^{-t}QB^{-1}p + \ \text{Minimize} \ & (cB^{-1} - 2p^tB^{-t}QB^{-1})s \\ & + s^tB^{-t}QB^{-1}s \\ \text{subject to} \ & GB^{-1}s \leqslant g + GB^{-1}p \end{aligned}$$

Let RLT-LP and RLT-LP′ be the linear programs obtained by applying the first-level RLT to QP and QP′, respectively, using all possible pairwise constraint-factor products. These problems can be stated as follows, where G_i is the i^{th} row of G, for $i \in \{1, \ldots, m+2n\}$.

RLT-LP:

$$\begin{aligned} \text{Minimize} \ & cx + [x^tQx]_\ell \\ \text{subject to} \ & g_ig_j - (g_iG_j + g_jG_i)x + [(G_ix)(G_jx)]_\ell \geqslant 0 \\ & \forall (i,j) \in M_R \equiv \{(i,j) : 1 \leqslant i \leqslant j \leqslant m+2n\} \end{aligned}$$

RLT-LP′ :

$$\begin{aligned} & -cB^{-1}p + p^tB^{-t}QB^{-1}p \\ & + \text{Minimize} \ (cB^{-1} - 2p^tB^{-t}QB^{-1})s + [s^tB^{-t}QB^{-1}s]_\ell \\ & \text{subject to} \ (g_i + G_iB^{-1}p)(g_j + G_jB^{-1}p) - [(g_i + G_iB^{-1}p)(G_jB^{-1}) \\ & \qquad + (g_j + G_jB^{-1}p)(G_iB^{-1})]s + [(G_iB^{-1}s)(G_jB^{-1}s)]_\ell \geqslant 0 \\ & \qquad \forall (i,j) \in M_R \end{aligned}$$

Also, in accordance with our foregoing discussion, let us define the RLT variables for each quadratic term as $w_{kl} \equiv [x_k x_l]_\ell$ and $y_{kl} \equiv [s_k s_l]_\ell, 1 \leqslant k \leqslant l \leqslant n$. Henceforth, we will let $v[\cdot]$ denote the value at optimality of the corresponding problem $[\cdot]$. The next proposition shows that the first-level RLT is invariant under affine transformations.

PROPOSITION 4. *v[RLT-LP] $=$ v[RLT-LP′]. In particular, if (x^*, w^*) solves RLT-LP, and if u^* is the corresponding optimal dual solution, then*

$$\begin{aligned}(s^*,[y^*]) &\equiv (s^*,[ss^t]_\ell^*)\\ &= (Bx^*+p, B[xx^t]_\ell^* B^t + p(x^*)^t B^t + Bx^* p^t + pp^t)\end{aligned} \tag{12}$$

solves RLT-LP′, with the corresponding optimal dual solution being u^*, where $[y^*]$ denotes an $n \times n$ matrix representation of y^*, such that for $1 \leqslant k \leqslant l \leqslant n$, $[y^*]_{kl} = [y^*]_{lk} = y^*_{kl}$, and where $[\cdot]_{kl}$ is the $(k,l)^{\text{th}}$ entry of the $n \times n$ symmetric matrix $[\cdot]$.

Proof. From the KKT conditions of the linear program RLT-LP, we have

$$-c - \sum_{(i,j)\in M_R} u^*_{ij}(g_j G_i + g_i G_j) = 0 \tag{13.1}$$

and

$$-2Q + \sum_{(i,j)\in M_R} u^*_{ij}(G_j^t G_i + G_i^t G_j) = 0. \tag{13.2}$$

(Note that in the dual feasibility conditions (13.2), there are $n(n-1)/2$ more equations than the number of w-variables, but since the left-hand side is symmetric, (13.2) is valid, having $n(n-1)/2$ duplicated equations.) Dual feasibility of u^* for RLT-LP′ then follows directly from (13.1) and (13.2), since we have,

$$\begin{aligned}0 &= (13.1)B^{-1} + p^t B^{-t}(13.2)B^{-1}\\ &\equiv -cB^{-1} + 2p^t B^{-t} Q B^{-1} - \sum_{(i,j)\in M_R} u^*_{ij}[(g_i + G_i B^{-1}p)(G_j B^{-1})\\ &\quad +(g_j + G_j B^{-1}p)(G_i B^{-1})]\end{aligned} \tag{14.1}$$

and

$$\begin{aligned}0 &= B^{-t}(13.2)B^{-1} \equiv -2B^{-t}QB^{-1}\\ &\quad + \sum_{(i,j)\in M_R} u^*_{ij} B^{-t}(G_j^t G_i + G_i^t G_j)B^{-1}.\end{aligned} \tag{14.2}$$

For verifying the primal feasibility and complementary slackness conditions with respect to (12), define

$$s^* = Bx^* + p, \quad \text{so that} \quad x^* = B^{-1}(s^* - p) \tag{15.1}$$

and

$$\begin{aligned}[y^*] &= B[w^*]B^t + p(x^*)^t B^t + Bx^*p^t + pp^t \\ &= B[w^*]B^t - (pp^t - p(s^*)^t - s^*p^t), \end{aligned} \tag{15.2}$$

where $[w^*]$ is the matrix representation of w^*, defined similarly as $[y^*]$. Then, consider any constraint of RLT-LP′ evaluated at (s^*, y^*). Recognizing that ss^t in this constraint is replaced by $[y]$, we have, using (15.1) and (15.2),

$$\begin{aligned} &(g_i + G_iB^{-1}p)(g_j + G_jB^{-1}p) - [(g_i + G_iB^{-1}p)(G_jB^{-1}) \\ &\quad + (g_j + G_jB^{-1}p)(G_iB^{-1})]s^* + G_iB^{-1}[y^*]B^{-t}G_j^t \\ &= g_ig_j - (g_jG_i + g_iG_j)B^{-1}(s^* - p) \\ &\quad + G_iB^{-1}([y^*] + pp^t - p(s^*)^t - s^*p^t)B^{-t}G_j^t \\ &= g_ig_j - (g_jG_i + g_iG_j)x^* + G_i[w^*]G_j^t \geqslant 0 \end{aligned}$$

since (x^*, w^*) is primal feasible for RLT-LP. Moreover, this also exhibits that the slack values of constraints in RLT-LP and RLT-LP′ are the same when they are evaluated at (x^*, w^*) and (s^*, y^*), respectively. Hence the complementary slackness conditions for RLT-LP′ follow directly from the KKT conditions of RLT-LP. To complete the proof, using (13.1) and (13.2), we show the equivalence of the optimal objective function values as follows.

$$\begin{aligned} v[\text{RLT-LP}'] &= -cB^{-1}p + p^tB^{-t}QB^{-1}p \\ &\quad - \sum_{(i,j)\in M_R} u_{ij}^*[(g_i + G_iB^{-1}p)(g_j + G_jB^{-1}p)] \\ &= - \sum_{(i,j)\in M_R} u_{ij}^* g_ig_j \\ &\quad + \left[-c - \sum_{(i,j)\in M_R} u_{ij}^*(g_iG_j + g_jG_i)\right] B^{-1}p \\ &\quad + p^tB^{-t}\left[Q - \frac{1}{2}\sum_{(i,j)\in M_R} u_{ij}^*(G_i^tG_j + G_j^tG_i)\right] B^{-1}p \\ &= - \sum_{(i,j)\in M_R} u_{ij}^* g_ig_j = v[\text{RLT-LP}]. \end{aligned}$$

This completes the proof. ■

5. Enhancements of the RLT Relaxation: RLT-NLP

We now propose two enhancements for the foregoing RLT relaxation. First, we employ an eigen-transformation on QP to generate an equivalent problem for which the first-level RLT representation can be constructed. However, the separable structure of the revised objective function now enables us to identify the RLT constraints that play an important role in providing a tight representation and the ones that do not. Translating this information for RLT-LP, permits us to reduce the size of the latter problem, without compromising on its tightness. Second, motivated by the same constructs, we include certain separable convex quadratic constraints within RLT-LP to derive a revised relaxation RLT-NLP. The structure of these constraints is such that they pose no additional burden in the context of a Lagrangian Relaxation optimization scheme, while they contribute additional strength to the resulting relaxation. Each of these enhancements is presented in separate subsections below.

5.1. Eigen-Transformation (RLT-NLPE) and Identification of Selected Constraints (SC)

Let us begin by using a particular linear transformation based on the eigenstructure of the quadratic objective function. Let $Q = PDP^t$, where D is a diagonal matrix whose diagonal elements correspond to the eigenvalues λ_i of Q, $i = 1, \ldots, n$, and P is a matrix whose columns correspond to orthonormal eigenvectors of Q (see Golub and Van Loan (1989), for example). Define $x = Pz$, so that $z = P^t x$. The resulting eigen-transformed quadratic program is then obtained as follows:

$$\underset{z \in \mathbb{R}^n}{\text{Minimize}} \{cPz + z^t Dz : APz \leqslant b, l \leqslant Pz \leqslant u\}. \tag{16}$$

We next construct the first-level RLT for this quadratic problem (16) as before, defined in terms of z and the RLT-variables $y_{kl}, 1 \leqslant k \leqslant l \leqslant n$, but now, we further tighten the relaxation by including the nonlinear constraints $z_k^2 \leqslant y_{kk} \equiv [z_k^2]_\ell, \quad \forall k = 1, \ldots, n$, in this model. Let us refer to the resulting problem as **RLT-NLPE**.

Let us now attempt to reduce the size of the problem RLT-NLPE by eliminating those constraints that might not contribute significantly to determining an optimal solution. Toward this end, denote the index set of *"concave" variables* by $N_v \equiv \{k : \lambda_k < 0, k = 1, \ldots, n\}$. Renaming the constraints in (16) collectively as $F_i z \leqslant f_i, i = 1, \ldots, m + 2n$, let us re-organize each second-order RLT constraint $[(f_i - F_i z)(f_j - F_j z)]_\ell \geqslant 0$ from RLT-NLPE as $\sum_{k \in N_v} -F_{ik} F_{jk} y_{kk} \leqslant$ [the rest of the constraint]. On the left-hand side of the re-organized RLT constraint, if the positive coefficients have at least as much weight as the negative coefficients, then it is more likely that this constraint produces relatively strong upper bounds on the y_{kk}-variables that have positive coefficients, i.e., for which $-F_{ik} F_{jk} > 0, k \in N_v$. This is useful since the objective coefficients λ_k of these y_{kk} variables are negative. Therefore, assuming that the original problem has been scaled so that all variables are roughly commensurate with each other, we suggest generating only those RLT constraints that satisfy the condition $(\sum_{k \in N_v} -F_{ik} F_{jk}) \geqslant 0$, and suppressing the rest of them. *The corresponding RLT* *constraints in the previous representation RLT-LP are accordingly retained, while the remaining are discarded.* Observe that we can no longer guarantee that the original constraints are implied by the selected RLT constraints, and so, we include the original constraints $Gx \leqslant g$ in the reduced RLT-LP problem. Let us call this reduced relaxation having only selected constraints as **RLT-LP(SC)**.

5.2. Reformulation-Convexification Approach: Inclusion of Suitable Nonlinear Constraints in RLT-LP to Derive RLT-NLP

Observe that in RLT-LP, the following linear RLT constraints

$$[(x_k - l_k)^2]_\ell \geqslant 0, [(u_k - x_k)^2]_\ell \geqslant 0,$$
$$[(x_k - l_k)(u_k - x_k)]_\ell \geqslant 0, k = 1, \ldots, n \tag{17}$$

approximate the relationship $w_{kk} = x_k^2$ over the interval $l_k \leqslant x_k \leqslant u_k$, for $k = 1, \ldots, n$. Motivated by RLT-NLPE, we propose to replace (17) by the nonlinear constraints

$$x_k^2 \leqslant w_{kk} \leqslant (u_k + l_k)x_k - u_k l_k, l_k \leqslant x_k \leqslant u_k, k = 1, \ldots, n \tag{18}$$

and call the resulting nonlinear problem as **RLT-NLP**, and its corresponding reduced version as derived in Section 5.1 as **RLT-NLP(SC)**. Notice that the upper bounding linear function in (18) is precisely the last constraint in (17). The improvement via (18), which implies (17), appears in the case when the problem RLT-LP tries to reduce the value of w_{kk} for some $k \in \{1, \ldots, n\}$ in the relaxed solution. Note that first two constraints in (17) merely *approximate* the function $w_{kk} = x_k^2$ from below via tangential supports at the points l_k and u_k. On the other hand, since (18) produces the exact lower envelope, it is equivalent to having an additional tangential support at the optimal point included within RLT-LP. Therefore, the enhancement (18) corresponds to a tighter bounding nonlinear problem than the linear program RLT-LP.

REMARK 2. Note that RLT-NLP and RLT-NLPE are no longer equivalent relaxations. Although RLT-NLPE usually turns out to yield a tighter representation because the additional nonlinear constraints provide a better support for the *"convex" variables* (those associated with $\lambda_k > 0$), the loss in structure due to the increased density in the bounding constraints inhibits the development of an efficient solution scheme. Hence, we implemented RLT-NLP. (See Section 7 for some related computational results.)

REMARK 3. To further strengthen the bounding problem, we can derive two additional classes of linear constraints based on projecting cubic bound-factor products onto the quadratic space, and on squaring *differences* of bound (or constraint) factors. However, although these constraints serve to tighten the relaxation somewhat, they increase the problem size considerably. Hence, due to the ensuing computational burden, they will not be used in the overall branch-and-bound algorithm.

(We refer the interested reader to Tuncbilek (1994) for the generation of such constraints; also, see Section 7 for some related computational results.)

6. A Lagrangian Dual Approach for Solving RLT Relaxations

Given a problem QP that has n variables and (up to) $m + 2n$ constraints, the corresponding (non-reduced) first-level RLT problems has $n(n + 1)/2$ additional variables, and a total of $(m + 2n)(m + 2n + 1)/2$ constraints. It is clear that the size of RLT-LP, or even RLT-NLP, gets quite large as the size of QP increases. If we can obtain a tight lower bound on RLT-NLP relatively easy, then we can trade-off between the quality of the bound and the effort necessary to obtain it. For this purpose, we propose to use a Lagrangian relaxation of RLT-NLP (see Fisher, 1981, for example), and solve – not necessarily exactly – the Lagrangian Dual Problem **LD-RLT-NLP**.

To define the proposed Lagrangian dual for RLT-NLP, we dualize all but the constraints $[(x_k - l_k)(x_l - l_l)]_\ell \geqslant 0$, $[(u_k - x_k)(x_l - l_l)]_\ell \geqslant 0$, $[(u_k - x_k)(u_l - x_l)]_\ell \geqslant 0$ and $[(x_k - l_k)(u_l - x_l)]_\ell \geqslant 0, \quad \forall 1 \leqslant k < l \leqslant n$, from the set (4.2), and the constraints (18) which have replaced (17). These constraints, along with the bounds $l_k \leqslant x_k \leqslant u_k, k = 1, \ldots, n$, on the x-variables, comprise the Lagrangian subproblem constraints (see Fisher, 1981). Note that the foregoing linearized bound-factor product constraints in the subproblem yield lower and upper bounding linear functions for the linearized cross product terms w_{kl}, for all $1 \leqslant k < l \leqslant n$. These constraints can be expressed in open form as follows:

$$l_l x_k + l_k x_l - l_k l_l \leqslant w_{kl} \leqslant l_l x_k + u_k x_l - u_k l_l \quad \forall 1 \leqslant k < l \leqslant n \tag{19.1}$$

$$u_l x_k + u_k x_l - u_k u_l \leqslant w_{kl} \leq u_l x_k + l_k x_l - l_k u_l \quad \forall 1 \leqslant k < l \leqslant n. \tag{19.2}$$

In addition, following the layering strategy proposed by Guignard and Kim (1987), in order to facilitate the solution of the Lagrangian subproblems, we replace w_{kl} by $w'_{kl} \quad \forall 1 \leqslant k < l \leqslant n$ in (19.2), and we include in RLT-NLP the constraints

$$w'_{kl} = w_{kl} \quad \forall 1 \leqslant k < l \leqslant n. \tag{20}$$

These new constraints (20) are also dualized using some defined Lagrange multipliers. Hence, in order to solve the Lagrangian subproblem, depending on the signs of the coefficients for each w_{kl} and w'_{kl} variable in the Lagrangian subproblem objective function, we first replace this variable in the objective function appropriately by either its lower or upper bounding function in terms of x as defined in (19). The resulting subproblem objective function is then given in terms of the variables (x_k, and w_{kk}, for $k = 1, \ldots, n$) alone, and the subproblem constraints are now defined by (18) and the bounds on the x-variables. This reduced separable

Lagrangian dual subproblem can be stated as follows:

$$\sum_{k=1}^{n} \text{Minimize}\{\hat{c}_k x_k + \hat{q}_{kk} w_{kk} : \\ x_k^2 \leqslant w_{kk} \leqslant (u_k + l_k)x_k - u_k l_k, l_k \leqslant x_k \leqslant u_k\}. \tag{21}$$

Suppose that instead of the constraints in (21), we use the restrictions

$$w_{kk} = x_k^2, l_k \leqslant x_k \leqslant u_k, k = 1, \ldots, n \tag{22}$$

to obtain the reduced Lagrangian dual subproblem in the following more favorable form:

$$\text{Minimize}\left\{\sum_{k=1}^{n} \hat{c}_k x_k + \sum_{k=1}^{n} \hat{q}_{kk} w_{kk} : (22)\right\} \\ = \sum_{k=1}^{n} \underset{l_k \leqslant x_k \leqslant u_k}{\text{Minimize}} \{\hat{c}_k x_k + \hat{q}_{kk} x_k^2\}. \tag{23}$$

The following proposition asserts that the simpler problem (23) equivalently solves (21).

PROPOSITION 5. *Problem* (21) *is equivalently solved via Problem* (23).

Proof. For any $k \in \{1, \ldots, n\}$, if $\hat{q}_{kk} > 0$, then in (21), w_{kk} can be replaced by x_k^2 in the objective function, which makes it equivalent to (23). If $\hat{q}_{kk} \leqslant 0$, then in (21), if w_{kk} is replaced by $(u_k + l_k)x_k - u_k l_k$, the revised objective function becomes a linear function of the variable x_k. Consequently, an optimal value for x_k occurs at either of its bounds, and so, the constraints on w_{kk} in (21) are satisfied as equalities at optimality. Examining the same case for (23), we similarly obtain a concave univariate quadratic minimization problem over simple bounds, and so, an optimal value for x_k occurs at either of the bounds. Also, since $(u_k + l_k)x_k - u_k l_k = x_k^2$ at either bound, both (23) and (21) produce the same optimal solutions. This completes the proof. ■

REMARK 4. The Lagrangian dual problem is a nondifferentiable optimization problem. To solve this problem, we adopted the subgradient deflection algorithm of Sherali and Ulular (1989), using their recommended parameter values. After running this algorithm for up to 200 iterations, another 50 iterations were performed in a reduced subspace by fixing the dual variables having relatively small magnitudes (less than half of the average magnitude of all the dual values) at their current level in the given incumbent solution, and polishing the remaining dual variable values. Finally, a dual ascent was performed by finding optimal values for the dual variables associated with the constraints (20) one at a time, in a Gauss-Seidel fashion, given the remaining dual variable values. This can be executed with little additional effort, and so, it provides a quick dual ascent step (see Tuncbilek, 1994, for details).

7. A Preliminary Computational Comparison of the Bounding Problems

To evaluate and compare the different approaches for generating and solving a first-level RLT relaxation, we performed some preliminary empirical experiments using test problems from the literature. Using three bilinear programming problems (BLP1, BLP2, BLP3) from Al-Khayyal and Falk (1983), and two concave programming problems (CQP1, CQP2) from Floudas and Pardalos (1990), we solved the first-level linear RLT problem RLT-LP, the nonlinear RLT problem RLT-NLP, the eigen space first-level nonlinear RLT problem RLT-NLPE, and the reduced RLT-NLP that uses only selected constraints (denoted RLT-NLP(SC)), as well as the Lagrangian dual problems corresponding to the latter three (denoted by the prefix **LD-**). For these computations, the linear and nonlinear programs were solved using GAMS along with the solver MINOS 5.2, and a Fortran code was written for solving the Lagrangian dual problems as per Remark 4. All the runs were conducted on an IBM 3090 mainframe computer. Table I reports the resulting lower bounds obtained along with the solution times required to generate these bounds.

RLT-LP yields lower bounds very close to the actual global minimum for all the problems except for BLP1 (BLP2 and BLP3 are solved exactly). RLT-NLPE, where we have included the nonlinear constraints $z_k^2 \leqslant y_{kk}$ only for $k \in \{1, \ldots, n\}$ such that $\lambda_k \geqslant 0$, solves all problems to (near) optimality. However, the computational effort is considerably increased since this lower bounding problem is a nonlinear programming problem. When we include the nonlinear constraints $z_k^2 \leqslant y_{kk}$ for all $k \in \{1, \ldots, n\}$ in RLT-NLPE, the imposed resource limit of 1000 cpu seconds for GAMS is exceeded for BLP2. On the other hand, we were able to solve RLT-NLP, even while including all n nonlinear constraints, and it produced solutions comparable to those obtained by RLT-NLPE. Reconsidering RLT-LP, but this time, including the additional two classes of constraints mentioned in Remark 3 of Section 5, improves the lower bound for BLP1 to -1.125, and that for CQP2 to -39.47, the latter of which is greater than any of the bounds reported in Table I. However, the computational effort for CQP2 increased to 13.36 cpu seconds.

Although the Lagrangian dual problem LD-RLT-NLP should give the same lower bound as does RLT-NLP, the bounds for BLP3 and CQP2 are somewhat worsened due to the inherent difficulty in solving nondifferentiable optimization problems. Nevertheless, the attractive computational times, especially when the problem size increases, makes this method our first choice to be used in the proposed branch-and-bound algorithm.

In adopting the Lagrangian dual approach of Section 6 for RLT-NLPE, the $2n(n-1)$ RLT constraints that are generated using the constraints $l \leqslant Pz \leqslant u$ were dualized. Implied simple bounds on z-variables were computed by minimizing and then maximizing each row of $z = P^t x$ over $l \leqslant x \leqslant u$ to obtain counterparts of both the constraints (19) and the subproblem (23) in z-space. Ideally, we would like to use LD-RLT-NLPE as the bounding problem for the overall algorithm.

TABLE I. Comparison of RLT schemes

Problem	m	n	Known v[QP]	v[RLT-LP]	cpu secs.
BLP1	2	2	−1.083	−1.50	0.042
BLP2	10	10	−45.38	−45.38	3.75
BLP3	13	10	−794.86	−794.86	22.74
CQP1	11	10	−267.95	−268.02	9.72
CQP2	5	10	−39.00	−39.83	3.23

Problem	v[RLT-NLPE]	cpu secs.	v[LD-RLT-NLPE]	cpu secs.
BLP 1	−1.083	0.086	−1.097	0.09
BLP 2	−45.38	31.23	−69.17	0.81
BLP 3	−794.76	48.55	−11586.88	0.97
CQP1	−268.02	12.16	−269.87	0.85
CQP2	−39.83	4.69	−43.38	0.65

Problem	v[RLT-NLP]	cpu secs.	v[LD-RLT-NLP]	cpu secs.
BLP 1	−1.089	0.105	−1.089	0.02
BLP 2	−45.38	9.79	−46.10	0.43
BLP 3	−794.86	78.41	−829.52	0.57
CQP1	−268.02	13.30	−269.83	0.48
CQP2	−39.83	4.69	−43.93	0.25

Problem	v[RLT-NLP(SC)]	cpu secs.	$\frac{\text{used}}{\text{total}}$ constr.	v[LD-RLT-NLP(SC)]
BLP1	−1.089	0.091	15/21	−1.089
BLP2	−45.38	6.07	339/465	−45.62
BLP3	−806.53	32.84	387/564	−841.60
CQP1	−268.02	13.61	352/497	−269.68
CQP2	−40.10	2.54	261/320	−42.99

Legend: (m, n) = size of the problem QP, $v[\cdot]$ = optimal solution of problem $(\cdot)$, cpu secs.=cpu seconds to solve the problem on an IBM 3090 computer.

However, due to its dense structure, LD-RLT-NLPE tends to perform poorly as for problems BLP2 and BLP3, although the bounds for CQP1 and CQP2 are slightly improved over those obtained via LD-RLT-NLP. Upon using the tightest simple bounds on the z-variables that contain the feasible region, the lower bound for BLP3 improved considerably to -1507.64, but it is still 81% lower than that obtained via LD-RLT-NLP.

Applying the constraint selection strategy of Section 5.1 on RLT-NLP we solved the reduced problem RLT-NLP(SC) with reduced computational effort, although at an expense of a 1.5% decrease in the lower bound for BLP3. (Roughly 18–30% of the constraints are deleted by this strategy.) For the same problem, v[LD-RLT-

NLP(SC)] has also worsened by 1.5% compared to v[LD-RLT-NLP]. However, for the rest of the problems, LD-RLT-NLP(SC) actually improved the lower bounds. We have also observed that there is only a negligible increase in the required computational effort compared to that consumed by LD-RLT-NLP. Section 10 reports on computational experiments using RLT-NLP, LD-RLT-NLP, and LD-RLT-NLP(SC) within the proposed branch-and-bound algorithm.

8. Additional Features of the Proposed Algorithm

8.1. Scaling

Our experiments with the conjugate subgradient deflection algorithm have indicated that scaling plays an important role in the performance of this algorithm. Among several scaling methods we tried, including a sophisticated iterative method used in the package MINOS (see Murtagh and Saunders, 1987), a simple one seemed to work well. This method scales the variables such that the lower and upper bounds are mapped onto the hypersquare $[0, 1]^n$. In addition, a row scaling is employed that divides each constraint $a_i x \leqslant b_i, i \in \{1, \ldots, m\}$, by the ℓ_∞-norm of (a_i, b_i).

8.2. Branch-and-Bound Search Strategy

The branch-and-bound search strategy employed uses a hybrid of the depth-first search and the best-first search strategies as suggested by Sherali and Myers (1985/6), where at most a fixed *(MAXACT)* number of nodes are kept active in the branch-and-bound tree. We used the largest value of *MAXACT* as permissible by storage limitations, depending on the size of the problem. In this approach, branching is performed by splitting the bounding interval of a variable as stated in Section 8.5.

8.3. Optimality Criterion

To avoid undue excessive computations involved in sifting through alternative optimal solutions or close to global optimal solutions, we adopt the fathoming criterion

$$\mathrm{LB} \geqslant \mathrm{UB} - \varepsilon|\mathrm{UB}| \tag{24}$$

where $0 < \varepsilon < 1$, and where LB is a valid lower bound at the current branch-and-bound node, and UB is the current best (incumbent) solution value for QP. Hence, when the algorithm stops, we can claim that the global minimum is within $100\varepsilon\%$ of the current best solution.

8.4. Range Reductions

The tightness of the lower and upper bounds that define the *box constraints* on the variables play a major role in the performance of the Lagrangian dual bounding scheme. Fast and effective procedures to improve simple bounds, known as "logical tests", have acquired a good reputation in (mixed) integer 0-1 programming. In addition to showing that these types of tests have their counterparts in global optimization, Hansen, Jaumard and Lu (1991) have also developed new tests using analytical and numerical methods. In the same spirit, we also propose some suitable strategies for improving lower and upper bounds on the variables at each node of the branch-and-bound tree. The first procedure is based on knapsack problems defined by individual linear functional constraints along with the box constraints. The second procedure is motivated by the number of constraints that have to be binding at optimality. The third and the fourth procedures are applied to cutting planes based on, respectively, the Lagrangian dual objective function, and the eigen-transformed separable objective function. The latter three strategies are based on some appropriate optimality conditions, by which we can further tighten the box constraints beyond feasibility considerations, by discarding regions that cannot contain an optimal solution. Since the bounds are tightened based on optimality related considerations, they are possibly not implied by the already existing constraints, and in the light of Proposition 2, this helps to generate tighter RLT bounding problems as well. Below, we summarize these range reduction strategies; for further details, see Tuncbilek (1994).

8.4.1. *Range Reduction Strategy 1*

This is a strategy for tightening variable bounds by virtue of a simple feasibility check. By minimizing the left-hand side expressions of the less than or equal to type functional constraints of QP in turn over the box constraints, we can obtain a maximum slack for each constraint. The procedure then discards that portion of the interval for each variable for which the maximum slack of some constraint would become negative.

8.4.2. *Range Reduction Strategy 2*

Denoting the number of nonpositive eigenvalues of the matrix Q by q, this procedure is based on the result that at optimality, at least q out of the $m + 2n$ constraints can be required to hold as equalities (see Phillips and Rosen, 1990, and Mueller, 1970). The proposed strategy computes a measure of redundancy for each functional constraint with respect to the box constraints by maximizing the left-hand side of the less than or equal to type functional constraints. If the number of nonredundant constraints thus detected, plus the number of x-variables that have an *original* lower or upper bound restricting its interval at the current node, is less than q, then we can fathom the current node. If this condition does not hold, then we can attempt to improve the bounds on the variables by using the same concept over the feasible range of each variable in turn. For any given variable, we can readily identify in closed-form the range of its interval for which the foregoing criterion would hold, if at all. This range is then discarded unless the remaining interval becomes disjoint, and the procedure continues with another variable in a cyclic fashion until no further restrictions are effected in a complete cycle.

8.4.3. *Range Reduction Strategy 3*

For each node, given the current bound restrictions on the x-variables, consider the Lagrangian dual subproblem (23) corresponding to the incumbent dual solution of the parent subproblem. Since this gives a valid lower bounding problem, we examine each variable in turn and identify a subinterval of its range for which, if the variable was so restricted, the resulting Lagrangian based bound would fathom the node. Any such range identified for a variable is discarded, unless the remaining interval becomes disjoint. In a similar fashion, after having solved the current node's problem via the Lagrangian dual, we perform this range reduction before making any branching decision. This restriction also serves as an input for the immediate descendent nodes, for which this strategy will be applied after imposing a branching decision.

8.4.4. *Range Reduction Strategy 4*

This procedure is performed on the eigen-transformed problem (16). We first derive lower and upper bounds on the z-variables, say $[L, U]$, by minimizing and then maximizing for each $i = 1, \ldots, n$, the definition function $z_i = (P^t_{i\bullet})x$, where $(P^t_{i\bullet})$, is the i^{th} row of P^t, over the box constraints on the x-variables corresponding to the current node restrictions. Let z° minimize the separable eigen-transformed objective function over $L \leqslant z \leqslant U$. Then, considering one z-variable at a time, while fixing the others at their values in z°, we determine a subinterval to be eliminated for that z-variable by finding the range for which the objective function value would exceed the incumbent solution value. We eliminate such a subinterval, unless the remaining interval becomes disjoint, and continue this procedure in a cyclic fashion until there is no further reduction in the bounding intervals for the z-variables. Let $[L^{\text{new}}, U^{\text{new}}]$ be the final bounds thus obtained. Then, by including the constraints $L^{\text{new}}_i \leqslant z_i = (P^t_{i\bullet})x \leqslant U^{\text{new}}_i$ for those $z_i, i = 1, \ldots, n$, for which an interval reduction has resulted, we perform Range Reduction Strategy 1 again to possibly further restrict the x-variable bounds.

8.5. Branching Variable Selection

We describe below a branching rule that attempts to resolve the discrepancy between the values of the RLT variables and the corresponding nonlinear terms they represent. If there is no such discrepancy, that is, if (3) holds at an optimal solution to the bounding problem, then this solution is also optimal to QP. In particular, to identify the RLT variables that contribute toward reducing the lower bound below the true QP objective value, let us compute the quantity

$$d_1 = \underset{1 \leqslant k \leqslant l \leqslant n}{\text{minimum}} \left\{ \min \left\{ 0, \begin{bmatrix} q_{kl}, & \text{if } k = l \\ 2q_{kl}, & \text{if } k < l \end{bmatrix} (\overline{w}_{kl} - \overline{x}_k \overline{x}_l) \right\} \right\}. \tag{25}$$

The next proposition provides a sufficient condition under which the x-variable part of the solution to the RLT bounding problem optimality solves QP.

PROPOSITION 6. *Suppose that* $(\overline{x}, \overline{w})$ *solves the RLT bounding problem of QP, with the corresponding objective value being* $\overline{z}$. *Let* d_1 *be computed as in (25). If* $d_1 = 0$ *then* $\overline{x}$ *solves QP with the corresponding objective value being* $\overline{z}$.

Proof. By Proposition 2, the constraints of QP are implied in the RLT problem. (Alternatively, if a reduced RLT is being employed, then these constraints are directly present in the RLT problem.) Therefore, $\overline{x}$ is feasible to QP, and its objective function value $c\overline{x} + \overline{x}^t Q \overline{x}$ gives an upper bound on the actual optimum $v[QP]$. Due to the inner minimization operation in (25), $d_1 = 0$ implies that $q_{kk}\overline{w}_{kk} \geqslant q_{kk}\overline{x}_k^2 \quad \forall k = 1, \ldots, n$, and $q_{kl}\overline{w}_{kl} \geqslant q_{kl}\overline{x}_k\overline{x}_l \quad \forall 1 \leqslant k < l \leqslant n$. Hence, $\overline{z} \geqslant c\overline{x} + \overline{x}^t Q \overline{x}$. But since $\overline{x}$ is feasible to QP, and $\overline{z}$ is a lower bound on QP, we have, $c\overline{x} + \overline{x}^t Q \overline{x} \geqslant v[QP] \geqslant \overline{z} \geqslant c\overline{x} + \overline{x}^t Q \overline{x}$, which implies that equality holds throughout. Hence, $\overline{x}$, of objective value $\overline{z}$, solves QP, and this completes the proof. ∎

Now suppose that d_1 given by some indices $(\underline{k}, \underline{l})$ in (25) is negative. (If there are ties in (25), we break it in favor of the maximum discrepancy $\overline{x}_k\overline{x}_l - \overline{w}_{kl}$, with further ties broken arbitrarily.) In order to choose between $x_{\underline{k}}$ and $x_{\underline{l}}$ for a branching variable, using the same motivation as in (25), we compute for $t = \underline{k}$ and $\underline{l}$,

$$\begin{aligned} d_2(t) &= \sum_{j=1}^{t-1} \min\{0, 2q_{jt}(\overline{w}_{jt} - \overline{x}_j\overline{x}_t)\} + \sum_{j=t+1}^{n} \min\{0, 2q_{tj}(\overline{w}_{tj} - \overline{x}_t\overline{x}_j)\} \\ &\quad + \min\{0, q_{tt}(\overline{w}_{tt} - \overline{x}_t^2)\}. \end{aligned} \tag{26}$$

The branching variable index is then selected as follows:

$$br = \text{argmin}\{d_2(t) : t = \underline{k}, \underline{l}\}. \tag{27}$$

REMARK 5. The foregoing branching scheme is based on an optimal solution $(\overline{x}, \overline{w})$ to the RLT bounding problem. However, note that since we are solving the Lagrangian dual of the enhanced first-level RLT problem, we do not directly obtain the optimal primal (x, w) variable values. Based on Theorem 1 of Sherali and Tuncbilek (1992), we could solve a linear program obtained by surrogating all the constraints of RLT-LP using the optimal dual solution obtained via LD-RLT-NLP, except for the bound factor product constraints and the original constraints of QP, and guarantee convergence of the algorithm. However, this linear program itself might be a computational bottleneck. Hence, for theoretical purposes, one could resort to solving the latter linear program only finitely often along any branch of the branch-and-bound tree, but for the most part, resort to the following branching scheme that is motivated by the above theory.

Step 1. Consider the Lagrangian subproblem associated with the dual incumbent solution to LD-RLT-NLP at the stage (23), where $[l, u]$ are the x-variable bounds associated with the *current node*. Letting $f_k(x_k) = \hat{c}_k x_k + \hat{q}_{kk} x_k^2$, select a branching variable index as follows:

$$br = \underset{\substack{k \in \{1,\ldots,n\} \\ \ni l_k < \overline{x}_k < u_k}}{\text{argmax}} \; [\min\{|f_k(u_k) - f_k(\overline{x}_k)|, |f_k(l_k) - f_k(\overline{x}_k)|\}] \tag{28}$$

where $\overline{x}$ solves (23), breaking ties in favor of the variable that has the largest feasible interval at the current node. If all variables are at their bounds in $\overline{x}$, proceed to Step 2.

Step 2. Using the incumbent dual solution to LD-RLT-NLP as the starting solution, continue the conjugate subgradient procedure for 50 more iterations omitting the resetting strategy, and accumulate $(\overline{x}, \overline{w})$ as the average of the Lagrangian subproblem solutions. (In theory, Larsson and Liu (1989) show that this should ultimately converge to the optimal primal solution to RLT-NLP under a very restricted step-size strategy. To aid this process, we also attempt to project $\overline{x}$, if infeasible, onto the feasible region of QP by taking a single step toward this region along a direction defined by the violated constraints.) Select a branching variable using (25)-(27) provided that $d_1 < 0$ in (25), and otherwise, if $d_1 = 0$, then proceed to Step 3.

Step 3. Select the x-variable that has the largest feasible interval in the current node problem as the branching variable. That is, let

$$br = \underset{k \in \{1,\ldots,n\}}{\text{argmax}} \; \{(u_k - l_k)\} \tag{29}$$

and exit this procedure.

Partitioning Phase. If the branching variable x_{br} is selected using Step 1 or Step 2 of the above procedure, then split its current interval at the value $\overline{x}_{br}$, creating the partitions $[l_{br}, \overline{x}_{br}]$ and $[\overline{x}_{br}, u_{br}]$, provided that the length of each resulting partition is at least 5% of the length of the current interval $[l_{br}, u_{br}]$. Otherwise, partition the current interval of x_{br} by simply bisecting it.

8.6. FINDING GOOD QUALITY FEASIBLE SOLUTIONS

In the branch-and-bound algorithm, besides a tight lower bound, we should also actively seek good quality solutions early in the algorithm. Although RLT-(N)-LP yields a feasible solution by construction, this may not be true for the case of

LD-RLT-NLP. Therefore, we developed and tested the following three heuristic procedures, and composed them in the manner described below.

In the first procedure, we formulate an l_1-norm penalty function for the problem QP that incorporates absolute violations in the functional constraints into the penalty term, and then approximately minimize this penalty function using the conjugate subgradient algorithm described in Remark 4. As the starting solution, we used the average of the x-variable part of the subproblem solutions obtained at improving iterations of the conjugate subgradient algorithm while solving the Lagrangian dual problem at the current branch-and-bound node. During this procedure, we attempt to project promising near feasible points onto the original feasible region of QP by taking a single step toward this region along a direction defined by the violated constraints. We consider a feasibility tolerance of 10^{-6} in Euclidean distance to be compatible with the default setting for MINOS (see Murtagh and Saunders, 1987).

In the second procedure, we simply apply MINOS to the original problem QP, using the resulting point of the foregoing penalty approach as the starting solution.

In the third procedure, at each node, having solved the Lagrangian dual problem, we formulate (21) corresponding to the incumbent dual solution, but now, we also include the functional constraints of QP in this subproblem. The resulting convex program is then solved using MINOS. Notice that this procedure also happens to be a dual ascent step for the Lagrangian dual problem.

In the overall branch-and-bound algorithm, we implemented the following heuristic scheme. For the first 10 nodes of the branch-and-bound tree, we employed the second procedure, using the solution of the first procedure as a starting solution. At all subsequent nodes, we employed the first and the third procedures, except that whenever the incumbent solution improved, we executed the second procedure using this new incumbent point as the starting solution, to possibly further improve this incumbent solution.

9. Summary of the Algorithm

Step 0. Initialization. Apply the heuristic procedure of Section 8.6 using $x^\circ = Pz^\circ$ as the starting solution, where z° is defined in Section 8.4.4, to obtain an initial incumbent solution. Initialize the branch-and-bound tree as node 0, and let the present set of bounds on the x-variables be as given in Problem QP. Flag node 0 and proceed to Step 1.

Step 1. Range Reductions. Designate the most recently flagged node as the current active node. For the given set of bounds, apply the Range Reduction Strategies 1-4. If this indicates a fathoming of the current node, then go to Step 3. Otherwise, proceed to Step 2.

Step 2. Bounding and Branching Step. Scale the node subproblem using the scheme of Section 8.1, and solve the problem LD-RLT-NLP to obtain a lower bound on the node subproblem. (If LD-RLT-NLP(SC) is used, then prior to the constraint selection process, scale the problem (16) as described in Section 8.1 using the bounds $[L^{\text{new}}, U^{\text{new}}]$ on z-variables, which are readily available as a byproduct of the Range Reduction Strategy 4.) During the process of solving the Lagrangian dual problem, whenever the incumbent dual solution improves during the conjugate subgradient optimization iterations, check if the fathoming condition (24) is satisfied, and proceed to Step 3 if this is the case. Apply the heuristic procedure of Section 8.6, to possibly improve the incumbent solution. Again, if the fathoming rule (24) holds, then proceed to Step 3. Otherwise, apply the Range Reduction Strategy 3 using the current incumbent dual solution, and select a branching variable according to the branching rule of Section 8.5. Accordingly, partition the current node subproblem by creating two nonactive descendent nodes corresponding to the resulting two sets of (revised) bounds on the branching variable x_{br}, and go to Step 4.

Step 3. Fathoming Step. Fathom the current node. If the sibling of the fathomed node is not active (see Section 8.2) then flag that node. Otherwise, flag the nonactive sibling of the highest level node on the path from the current node to the root node, and return to Step 1. If there is no such node, then either stop if there exist no active end nodes, or else, proceed to Step 4.

Step 4. Node Selection Step. If the incumbent solution has improved since the last time Step 4 has been visited, then fathom any active node that satisfies the criterion (24). If the number of active end nodes equals *MAXACT*, then select an active end node that has the least lower bound, and flag one of its descendent nodes. On the other hand, if the number of active end nodes is less than *MAXACT*, then along the branch of each such end node, find the lowest level node (closest to the root) that has at least one nonactive descendent node, and among these nodes, flag the nonactive descendent node of the one that has the least lower bound. Return to Step 1.

The convergence of the above algorithm follows from Sherali and Tuncbilek (1992), where it is shown that for a more general procedure, any accumulation point of the sequence of solutions generated for the RLT relaxations along any infinite branch solves the Problem QP. Hence, finite convergence to an ε-optimal solution can be obtained.

10. Computational Results and Conclusions

We now evaluate the proposed algorithm using a set of test problems chosen from the literature. In addition to the five problems used in Section 7, six larger sized ($m = 10, n = 20$) standard test problems from Floudas and Pardalos (1990), and seven randomly generated problems using the generation scheme of Phillips and Rosen (1990) and Visweswaran and Floudas (1993) of size upto ($m = 20, n = 50$) are solved.

TABLE II. Performance of the branch-and-bound algorithm using LD-RLT-NLP(SC)

Problem (m, n)		Known v[QP]	v[B&B]	cpu secs.	No. of B&B nodes	Node 0 LB
BLP1	(2, 2)	−1.083	−1.083	0.71	1	−1.089
BLP2	(10, 10)	−45.38	−45.38	1.08	1	−45.81
BLP3	(13, 10)	−794.86	−794.86	3.02	5	−838.85
CQP1	(11, 10)	−267.95	−268.01	1.17	1	−270.69
CQP2	(5, 10)	−39.00	−39.00	1.72	5	−42.95
CQP3	(10, 20)	−394.75	−394.75	3.29	3	−423.32
CQP4	(10, 20)	−884.75	−884.75	2.61	1	−904.02
CQP5	(10, 20)	−8695.01	−8695.01	2.55	1	−9097.99
CQP6	(10, 20)	−754.75	−754.75	2.61	1	−787.60
CQP7	(10, 20)	−4105.28	−4150.41	15.94	11	−5126.68
IQP1	(10, 20)	49318.0	49317.97	2.73	3	44937.40

Legend: Problem=problem name, (m, n)=size of the problem QP, v[QP]=known optimal (best) solution of problem QP, v[B&B]=branch-and-bound algorithm incumbent value, cpu secs.=cpu seconds to solve the problem on an IBM 3090 computer, No. of B&B nodes=number of branch-and-bound nodes generated, Node 0 LB=lower bound on QP at root node (optimality criterion is 1% for the first 5 problems, and 5% for the rest of the problems).

TABLE III. Performance of the branch-and-bound algorithm using LD-RLT-NLP

Problem (m, n)		Known v[QP]	v[B&B]	cpu secs.	No. of B&B nodes	Node 0 LB
BLP1	(2, 2)	−1.083	−1.083	0.71	1	−1.089
BLP2	(10, 10)	−45.38	−45.38	1.37	3	−46.02
BLP3	(13, 10)	−794.86	−794.86	2.66	5	−839.02
CQP1	(11, 10)	−267.95	−268.01	1.12	1	−270.68
CQP2	(5, 10)	−39.00	−39.00	1.61	5	−42.96
CQP3	(10, 20)	−394.75	−394.75	8.13	7	−439.03
CQP4	(10, 20)	−884.75	−884.75	2.54	1	−928.92
CQP5	(10, 20)	−8695.01	−8695.01	13.26	11	−9541.67
CQP6	(10, 20)	−754.75	−754.75	5.04	5	−803.31
CQP7	(10, 20)	−4105.28	−4150.41	27.00	25	−5262.57
IQP1	(10, 20)	49318.0	49317.97	2.61	3	45776.43

Legend: Problem=problem name, (m, n)=size of the problem QP, v[QP]=known optimal (best) solution of problem QP, v[B&B]=branch-and-bound algorithm incumbent value, cpu secs.=cpu seconds to solve the problem on an IBM 3090 computer, No. of B&B nodes=number of branch-and-bound nodes generated, Node 0 LB=lower bound on QP at root node (optimality criterion is 1% for the first 5 problems, and 5% for the rest of the problems).

Tables II and III present results on the standard test problems using LD-RLT-NLP(SC) and LD-RLT-NLP, respectively, as the lower bounding problem. The optimality tolerance (see Section 8.3) is taken as 1% for the first five problems and as 5% for the remaining ones. Using LD-RLT-NLP(SC), all the problems are solved under 16 cpu seconds. Compared to Table III, although a more relaxed problem is being solved in Table II, we observe an improvement in several of the root node lower bounds, especially for larger sized problems. This is principally due to the reduction of the dual search space, which improves the performance of the conjugate subgradient algorithm, while not significantly sacrificing the tightness of the theoretical lower bound. In Table III, the run times have somewhat increased for the larger problems, where all the problems, except one, are solved under 14 cpu seconds. Problem CQP7 required by far the greatest effort, taking 27 cpu seconds to be solved. However, even when using an optimality criterion of 5%, a better solution than the best known one to CQP7 is found. (This previous best solution reported in Floudas and Pardalos (1990) has an objective value of −4105.2779.) Upon reducing the optimality criterion to 1% and then to 0.1%, CQP7 is solved in 64 cpu seconds and in 205 cpu seconds, respectively, and in both cases, the same solution of value −4150.4087 is obtained (non-zero variables are $x_3 = 1.0429, x_{11} = 1.746744, x_{13} = 0.4314709, x_{16} = 4.43305, x_{18} = 15.85893, x_{20} = 16.4869$). For both of these cases, 86% of the overall effort is spent in solving the Lagrangian dual problem, showing that this is the determining factor for the total computational effort required. Using the reduced problem LD-RLT-NLP(SC), when we set the optimality criterion to 1% for CQP7, the algorithm consumed 90 cpu seconds, while for an accuracy tolerance of 0.1%, the algorithm was prematurely terminated after enumerating the preset limit of 200 nodes in 290 cpu seconds. Therefore, if a higher degree of accuracy is required, we recommend using the non-reduced problem LD-RLT-NLP; where overall, the marginally tighter representation does play an important role.

The heuristic procedure of Section 8.6 performed well by identifying the incumbent solution at the root node for all the problems, except for Problem CQP2, for which the optimum was found at the second node. The range reduction strategies of Section 8.4 prove to be very fast and effective; for example, if these reductions are not performed for Problem CQP7, the number of branch-and-bound nodes enumerated increases to 45 from 25 (requiring 57 cpu seconds) in Table III.

Although by Proposition 2, the tightness of the implied bounds on the variables should not affect the result of the bounding problems, this seems to play an important role in the performance of the Lagrangian dual solution procedure. As originally stated, problems CQP3-CQP7 do not include upper bounds on the variables, and for the purpose of the branch-and-bound algorithm, we used the smallest hyperrectangle that contains the feasible region found by minimizing and maximizing each variable over the feasible region. As a comparison, upon using a looser upper bound of 40 on each variable, which is trivially implied by a generalized upper bounding type of constraint present in these problems, CQP3 and CQP7 are solved in 35 and 43 cpu seconds, respectively, using LD-RLT-NLP(SC) as the bounding problem.

Table IV presents results for the smaller sized problems using RLT-NLP as the bounding problem and solving this by using MINOS 5.1, in lieu of the Lagrangian

TABLE IV. Performance of the branch-and-bound algorithm using RLT-NLP

Problem (m, n)		Known v[QP]	v[B&B]	cpu secs.	No. of B&B nodes	Node 0 LB
BLP1	(2, 2)	−1.083	−1.083	1.15	1	−1.089
BLP2	(10, 10)	−45.38	−45.38	12.90	1	−45.38
BLP3	(13, 10)	−794.86	−794.86	63.60	1	−794.86
CQP1	(11, 10)	−267.95	−268.01	15.25	1	−268.01
CQP2	(5, 10)	−39.00	−39.00	16.68	3	−39.82

Legend: Problem=problem name, (m, n)=size of the problem QP, v[QP]=known optimal (best) solution of problem QP, v[B&B]=branch-and-bound algorithm incumbent value, cpu secs.=cpu seconds to solve the problem on an IBM 3090 computer, No. of B&B nodes=number of branch-and-bound nodes generated, Node 0 LB=lower bound on QP at root node (optimality criterion is 1% for the first 5 problems, and 5% for the rest of the problems).

TABLE V. Performance of the branch-and-bound algorithm for randomly-generated problems using LD-RLT-NLP(SC)

(m, n)	v[B&B]	Node 0 LB	cpu seconds: 5% optimality	cpu seconds: Full Node 0	No. of B&B nodes	Node 0 relative gap 100(UB-LB)/|UB|
(20, 25)	365.91	365.64	1.71	6.06	1	0.076%
(20, 25)	1170.54	1169.98	1.52	5.88	1	0.048%
(20, 40)	−234.55	−234.95	3.58	12.91	1	0.172%
(20, 40)	−1264.53	−1264.93	2.30	12.44	1	0.031%
(20, 50)	−1311.48	−1326.01	3.84	17.21	1	1.109%
(20, 50)	−1259.01	−1319.69	12.77	18.50	1	4.82%
(20, 50)	−1215.62	−1241.04	4.98	18.08	1	2.091%

dual approach used in Table III. We observe that although the computational time has increased for all problems, the first four problems are solved at the root node itself, while the fifth problem returns an initial lower bound of −39.82 at the root node, the optimum value being −39.00. Note that while there is some loss in the tightness of the bounds due to the inaccuracy in solving RLT-NLP via the Lagrangian dual approach, the overall gain in efficiency is quite significant. Hence, there exists a great potential for further improvement if the lower bounding problem RLT-NLP could be solved more accurately by the Lagrangian dual scheme.

Using the random problem generator kindly shared by Visweswaran and Floudas (1993) and Phillips and Rosen (1990), we solved several larger sized problems having upto 20 constraints and 50 variables using the bounding problem LD-RLT-NLP(SC), and an optimality criterion of 5%. These problems are of the form $\min\{\theta_1(0.5\sum_{i=1}^{n}\lambda_i(x_i-\overline{w}_i)^2) : Ax \leqslant b, x \geqslant 0\}$, where $\theta_1 = -0.001$, and the number of positive and negative components of λ are roughly equal. As reported in Table V, all the problems are solved at the root node with a reasonable computational effort. Note that for several of these problems, the 5% optimality tolerance was detected even before the node zero analysis was completed (see the cpu seconds columns). However, the results given in Table V correspond to a full node zero analysis. The final column in Table V shows that the proven accuracy of the solutions obtained at node zero is typically significantly better than 5%. All incumbent solutions obtained were subsequently verified to be at lest within 1% of optimality, except for the sixth problem, for which a better incumbent solution of value −1281.0 was obtained when we enumerated two more nodes. This shows that the actual accuracy of the node zero lower bound for this problem is at least 3%.

To summarize, in this paper, we have investigated Reformulation-Convexification based relaxations embedded within a branch-and-bound algorithm for solving non-convex quadratic programming problems. Tight nonlinear programming relaxations have been defined, and a suitable Lagrangian dual procedure has been designed to solve the relaxations efficiently. The proposed algorithm has been further enhanced by incorporating fast and effective range reduction procedures. Test problems from the literature having upto 20 variables, and randomly generated problems having upto 50 variables have been solved with a reasonable computational effort.

For implementation, we recommend the use of LD-RLT-NLP when a better than 5% accuracy is desired, and the use of the reduced relaxation LD-RLT-NLP(SC) otherwise. For specially structured QPs, especially in the light of Remark 1 and Proposition 2, we strongly suggest that specialized, reduced RLT relaxations be investigated. The eigen-space based relaxation LD-RLT-NLPE is also recommended to be used whenever it can be conveniently constructed, and if there is a significant gap observed between the lower bounds generated via RLT-NLP and RLT-NLPE. For large sized problems, we recommend that the heuristic of Section 8.6 be used, perhaps in concert with solving the RLT based relaxation LD-RLT-NLP(SC) at a limited number of nodes.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. DMII-9121419.

References

1. Al-Khayyal, F. A. and J. E. Falk (1983), Jointly Constrained Biconvex Programming, *Math. of Oper. Res.* **8**, 273–286.
2. Al-Khayyal, F. A. and C. Larson (1990), Global Minimization of a Quadratic Function Subject to a Bounded Mixed Integer Constraint Set, *Annals of Operations Research* **25**, 169–180.
3. Balas, E. (1975), Nonconvex Quadratic Programming via Generalized Polars, *SIAM Journal on Applied Math.* **28**, 335–349.

4. Benacer, R. and Pham Dinh Tao (1986), Global Maximization of a Nondefinite Quadratic Function over a Convex Polyhedron, pp. 65–76, in *Fermat Days 85: Mathematics for Optimization*, J. B. Hiriart-Urruty (ed.), North-Holland, Amsterdam.
5. Bomze, I. M. (1992), Copositivity Conditions for Global Optimality Indefinite Quadratic Programming Problems, *Czechoslovak Journal for Operations Research* **1**,1–19.
6. Fisher, M. L. (1981), The Lagrangian Relaxation Methods for Solving Integer Programming Problems, *Management Science* **27**, 1–18.
7. Floudas, C. A. and P. M. Parsalos (1990), *A Collection of Test Problems for Constrained Global Optimization Algorithms*, Springer-Verlag, Berlin.
8. Floudas, C. A. and V. Visweswaran (1990), A Global Optimization Algorithm (GOP) for Certain Classes of Nonconvex NLP's-1. Theory, *Computers and Chemical Engineering* **14**, 1419.
9. Floudas, C. A. and V. Visweswaran (1993), A Primal-Relaxed Dual Global Optimization Approach: Theory, *JOTA* **78**, 187–225.
10. Golub, G. H. and C. F. Van Loan (1989), *Matrix Computations,* Second Edition, The Johns Hopkins University Press, Baltimore.
11. Guignard, M. and S. Kim (1987), Lagrangian Decomposition: A Model Yielding Stronger Lagrangian Bounds, *Mathematical Programming* **39**, 215–228.
12. Hansen, P., B. Jaumard, and S. Lu (1991), An Analytical Approach to Global Optimization, *Math. Programming* **52**, 227–254.
13. Kough, P. F., The Indefinite Quadratic Programming Problem, *Operations Research* **27**(**3**), 516–533.
14. Larsson, T. and Z. Liu (1989), A Primal Convergence Result for Dual Subgradient Optimization with Applications to Multicommunity Network Flows, Research Report, Department of Mathematics, Linkoping Institute of Technology, S-581 83, Linkoping, Sweden.
15. Manas, M. (1968), An Algorithm for a Nonconvex Programming Problem, *Econ Math Obzor Acad. Nacl. Ceskoslov* **4**(**2**), 202–212.
16. Meyer, G. G. (1988), Convergence of Relaxation Algorithms by Averaging, *Mathematical Programming* **40**, 205–212.
17. Mueller R. K. (1970), A Method for Solving the Indefinite Quadratic Programming Problem, *Management Science* **16**(**5**), 333–339.
18. Murtagh, B. A. and M. A. Saunders (1987), MINOS 5.1 User's Guide, Technical Report Sol 83-20R, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California.
19. Muu L. D. and W. Oettli (1991), An Algorithm for Indefinite Quadratic Programming with Convex Constraints, *Operations Research Letters* **10**, 323–327.
20. Pardalos, P. M. (1991), Global Optimization Algorithms for Linearly Constrained Indefinite Quadratic Programs, *Computers Math. Applic.* **21**, 87–97.
21. Pardalos, P. M., J. H. Glick, and J. B. Rosen (1987), Global Minimization of Indefinite Quadratic Problems, *Computing* **39**, 281–291.
22. Pardalos, P. M. and J. B. Rosen (1987), *Constrained Global Optimization: Algorithms and Applications*, Springer-Verlag, Berlin.
23. Pardalos, P. M. and S. A. Vavasis (1991), Quadratic Programming with One Negative Eigenvalue Is NP-Hard, *Journal of Global Optimization* **1**, 15–22.
24. Phillips, A. T. and J. B. Rosen (1990), Guaranteed ε-Approximate Solution for Indefinite Quadratic Global Minimization, *Naval Research Logistics* **37**, 499–514.
25. Ritter K. (1966), A Method for Solving Maximum Problems with a Nonconcave Quadratic Objective Function, *Z. Wahrscheinlichkeitstheorie,* **4**, 340–351.
26. Sherali, H. D. and A. R. Alameddine (1992), A New Reformulation-Linearization Technique for Bilinear Programming Problems, *Journal of Global Optimization* **2**(**3**), 379–410.
27. Sherali, H. D. and D. C. Myers (1985/6), The Design of Branch and Bound Algorithms for a Class of Nonlinear Integer Programs, *Annals of Oper. Res.* **5**, 463–484.
28. Sherali, H. D. and C. H. Tuncbilek (1992), A Global Optimization Algorithm for Polynomial Programming Problems Using a Reformulation-Linearization Technique, *The Journal of Global Optimization* **2**, 101–112.

29. Sherali, H. D. and O. Ulular (1989), A Primal-Dual Conjugate Subgradient Algorithm for Specially Structured Linear and Convex Programming Problems, *Appl. Math. Optim.* **20**, 193–221.
30. Tuncbilek, C. H. (1994), *Polynomial and Indefinite Quadratic Programming Problems: Algorithms and Applications,* PhD Dissertation, Industrial and Systems Engineering, Virginia Polytechnic Institute and State University.
31. Tuy, H. (1987), Global Minimization of a Difference of Two Convex Functions, *Mathematical Programming Study* **30**, 150–182.
32. Vavasis, S. A. (1992), Approximation Algorithms for Indefinite Quadratic Programming, *Mathematical Programming* **57**, 279–311.
33. Visweswaran, V. and C. A. Floudas, (1993), New Properties and Computational Improvement of the GOP Algorithm for Problems with Quadratic Objective Function and Constraints, *Journal of Global Optimization* **3**, 439–462.
34. Zwart, P. B. (1973), Nonlinear Programming: Counterexamples to Two Global Optimization Algorithms, *Operations Research* **21(6)**, 1260–1266.