Journal of Global Optimization (2021) 81:293–321
https://doi.org/10.1007/s10898-021-01017-y

Global solutions of nonconvex standard quadratic programs via mixed integer linear programming reformulations

Jacek Gondzio[1] · **E. Alper Yıldırım**[1]

Received: 22 June 2020 / Accepted: 20 March 2021 / Published online: 20 April 2021

Abstract
A standard quadratic program is an optimization problem that consists of minimizing a (nonconvex) quadratic form over the unit simplex. We focus on reformulating a standard quadratic program as a mixed integer linear programming problem. We propose two alternative formulations. Our first formulation is based on casting a standard quadratic program as a linear program with complementarity constraints. We then employ binary variables to linearize the complementarity constraints. For the second formulation, we first derive an overestimating function of the objective function and establish its tightness at any global minimizer. We then linearize the overestimating function using binary variables and obtain our second formulation. For both formulations, we propose a set of valid inequalities. Our extensive computational results illustrate that the proposed mixed integer linear programming reformulations significantly outperform other global solution approaches. On larger instances, we usually observe improvements of several orders of magnitude.

Keywords Nonconvex optimization · Quadratic programming · Mixed integer linear programming · Global optimization

Mathematics Subject Classification 90C20 · 90C11 · 90C26

1 Introduction

A standard quadratic program is an optimization problem in which a (nonconvex) homogeneous quadratic function, also known as a quadratic form, is minimized over the unit simplex. An instance of a standard quadratic program is given by

$$\text{(StQP)} \quad \nu(Q) := \min_{x \in \Delta_n} x^T Q x,$$

✉ E. Alper Yıldırım
E.A.Yildirim@ed.ac.uk

Jacek Gondzio
J.Gondzio@ed.ac.uk

[1] School of Mathematics, The University of Edinburgh, James Clerk Maxwell Building, Edinburgh EH9 3FD, UK

where $Q \in \mathcal{S}^n$ and $\mathcal{S}^n$ denotes the set of $n \times n$ real symmetric matrices, $x \in \mathbb{R}^n$, and Δ_n denotes the unit simplex in $\mathbb{R}^n$ given by

$$\Delta_n := \left\{ x \in \mathbb{R}^n_+ : e^T x = 1 \right\}, \tag{1}$$

where $e \in \mathbb{R}^n$ is the vector of all ones and $\mathbb{R}^n_+$ denotes the nonnegative orthant in $\mathbb{R}^n$.

We remark that having a quadratic form in the objective function is not restrictive since the problem of minimizing a nonhomogeneous quadratic function over the unit simplex can be reformulated in the form of (StQP) using the following identity:

$$x^T Q x + 2c^T x = x^T (Q + ec^T + ce^T)x, \quad \text{for each } x \in \Delta_n.$$

Standard quadratic programs arise in a variety of applications ranging from the classical portfolio optimization problem [26] to population genetics [24]; from quadratic resource allocation [21] to selection replicator dynamics and evolutionary game theory [5]. For a given matrix $Q \in \mathcal{S}^n$, Q is copositive if and only if $\nu(Q) \geq 0$. Therefore, standard quadratic programs can be used to check if a matrix is copositive. We refer the reader to the paper [3] for other applications, in which the term standard quadratic program was coined. The maximum stable set problem in graph theory [27] and its weighted version [17] can be formulated as instances of (StQP), which implies that (StQP) is, in general, NP-hard.

There is an extensive amount of literature on standard quadratic programs. In this paper, we are concerned with computing a global solution of (StQP). We therefore restrict our literature review to the exact solution approaches. All of these approaches, in general, are based on a branch-and-bound scheme and differ only in terms of the subroutines used for computing upper and lower bounds, and subdividing the feasible region. For instance, a DC (difference of two convex functions) programming approach is employed in [4] to compute a lower bound and a local optimization method is used to find an upper bound. Using a relation between global solutions of (StQP) and the set of cliques in an associated graph, referred to as the *convexity graph* (see Sect. 2.3), a branch-and-bound method based on an implicit enumeration of cliques is proposed in [29]. More recently, another branch-and-bound method was proposed in [25], in which both convex envelope estimators and polyhedral underestimators are employed for computing lower bounds, and an implicit enumeration of the KKT points is utilized using the relation with the set of cliques in the convexity graph. In another recent paper [10], a set of cutting planes is proposed in the context of a spatial branch-and-bound scheme.

Standard quadratic programs can also be solved by finite branch-and-bound methods proposed for solving more general nonconvex quadratic programming problems (see, e.g., [12,13]). These approaches are based on an implicit enumeration of the complementarity constraints in the KKT conditions. The resulting subproblems are approximated by semidefinite relaxations or by polyhedral semidefinite relaxations. By a simple manipulation of the KKT conditions, a general quadratic program can be formulated as a linear program with complementarity constraints (LPCC) (see, e.g., [20] and the references therein) and the resulting LPCC can be solved by an enumerative scheme such as branch-and-bound. An LPCC can also be formulated as a mixed integer linear programming (MILP) problem and can be solved using Benders decomposition [20] or by branch-and-cut [32]. A similar MILP formulation is proposed in [31] under the assumption of a bounded feasible region. Alternatively, using the completely positive reformulation of (StQP) (see, e.g., [7]), adaptive inner and outer polyhedral approximations of completely positive programs can be employed [11]. Clearly, one can also use general purpose nonlinear programming solvers such as BARON [30] and Couenne [2].

In this paper, we propose globally solving a standard quadratic program by reformulating it as a mixed integer linear programming (MILP) problem. We choose MILP reformulations due to the existence of powerful state-of-the-art MILP solvers such as CPLEX [22] and Gurobi [18]. We propose two different MILP reformulations. Our first formulation is based on casting (StQP) as a linear program with complementarity constraints and linearizing the complementarity constraints by using binary variables and big-M constraints. We discuss how to obtain valid bounds for the big-M parameters by exploiting the structure of (StQP). The second formulation is obtained by replacing the quadratic objective function in (StQP) by an overestimating function given by the maximum of a finite number of linear functions associated with the positive components of a feasible solution, referred to as the support. We show that the overestimating function is exact at all KKT points of (StQP), which leads to the second MILP formulation by introducing binary variables for modeling the support of a feasible solution. We further show that our second MILP formulation is, in fact, an exact relaxation of the first one. Furthermore, using the relation between the support of a global minimizer of (StQP) and the set of cliques in the convexity graph, we propose a set of valid inequalities for both of our MILP formulations. We conduct extensive computational experiments to assess the performances of our MILP formulations in comparison with several other global solution approaches. The computational results indicate that the proposed MILP formulations consistently outperform other global solution approaches. Furthermore, especially on larger instances, we observe improvements of several orders of magnitude.

Our work is related to the previous work on reformulations of a quadratic program as an instance of a linear program with complementarity constraints (LPCC) [20,31]. For a general quadratic program, the paper [20] proposes a two-stage LPCC approach. In the first stage, an LPCC is solved to determine if the quadratic program is bounded below, in which case a second LPCC is formulated to compute a global solution. The resulting complementarity problems are formulated as MILP problems and solved using a parameter-free approach via Benders decomposition, which eliminates the need for big-M parameters [19] (see [32] for a branch-and-cut approach). In contrast, the paper [31] explicitly uses big-M parameters. By using a Hoffman type error bound, the authors show that there exists a valid upper bound for the big-M parameters under the assumption of a bounded feasible region. They give a closed form expression of this bound for (StQP). Our first MILP formulation, which is based on a similar approach as in these previous MILP formulations, also employs big-M parameters as in [31]. In contrast with their approach, we exploit the specific structure of (StQP) in an attempt to obtain much tighter bounds for big-M parameters. Furthermore, our second MILP formulation is based on specifically taking advantage of the particular structure of (StQP). Therefore, in contrast with the previous approaches in the literature, we propose stronger MILP formulations for a more specific class of quadratic programs.

This paper is organized as follows. In Sect. 1.1, we briefly review our notation. Section 2 discusses several useful properties of standard quadratic programs. We present our MILP formulations as well as a set of valid inequalities in Sect. 3. Section 4 is devoted to the results of our computational experiments. We conclude the paper in Sect. 5.

1.1 Notation

We use $\mathbb{R}^n$, $\mathbb{R}^n_+$, and $\mathcal{S}^n$ to denote the n-dimensional Euclidean space, the nonnegative orthant, and the space of $n \times n$ real symmetric matrices, respectively. For $u \in \mathbb{R}^n$, we denote its jth component by u_j, $j = 1, \ldots, n$. Similarly, U_{ij} denotes the (i, j) entry of a matrix $U \in \mathcal{S}^n$, $i = 1, \ldots, n$; $j = 1, \ldots, n$. We denote the unit simplex in $\mathbb{R}^n$ by Δ_n. For any $U \in \mathcal{S}^n$

and $V \in \mathcal{S}^n$, the trace inner product of U and V is given by $\langle U, V \rangle := \sum_{i=1}^{n} \sum_{j=1}^{n} U_{ij} V_{ij}$. The unit vectors in $\mathbb{R}^n$ are denoted by e_j, $j = 1, \ldots, n$. We reserve $e \in \mathbb{R}^n$ and $E = ee^T \in \mathcal{S}^n$ for the vector of all ones and the matrix of all ones, respectively. We use 0 to denote the real number zero, the vector of all zeroes as well as the matrix of all zeroes in the appropriate dimension, which will always be clear from the context. We use $\mathrm{conv}(\cdot)$ to denote the convex hull. We define the following convex cones in $\mathcal{S}^n$:

$$\mathcal{N}^n = \left\{ M \in \mathcal{S}^n : M_{ij} \geq 0, \quad i = 1, \ldots, n; \ \ j = 1, \ldots, n \right\}, \tag{2}$$

$$\mathcal{S}^n_+ = \left\{ M \in \mathcal{S}^n : u^T M u \geq 0, \quad \forall\, u \in \mathbb{R}^n \right\}, \tag{3}$$

$$\mathcal{COP}^n = \left\{ M \in \mathcal{S}^n : u^T M u \geq 0, \quad \forall\, u \in \mathbb{R}^n_+ \right\}, \tag{4}$$

$$\mathcal{CP}^n = \mathrm{conv} \left\{ uu^T : u \in \mathbb{R}^n_+ \right\}, \tag{5}$$

$$\mathcal{DNN}^n = \mathcal{S}^n_+ \cap \mathcal{N}^n, \tag{6}$$

namely, the cone of component-wise nonnegative matrices, the cone of positive semidefinite matrices, the cone of copositive matrices, the cone of completely positive matrices, and the cone of doubly nonnegative matrices, respectively. The following relations easily follow from these definitions:

$$\mathcal{CP}^n \subseteq \mathcal{DNN}^n \subseteq \mathcal{COP}^n. \tag{7}$$

2 Preliminaries

In this section, we review several basic properties of standard quadratic programs that will be useful in the subsequent sections. We remark that these results can be found in the literature (see, e.g., [3,8]). We include proofs of some of these results for the sake of completeness.

2.1 Optimality conditions

Since a standard quadratic program has linear constraints, constraint qualification is satisfied at every feasible solution. Given an instance of (StQP), if $x \in \Delta_n$ is an optimal solution, then there exist $s \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$ such that the following KKT conditions are satisfied:

$$Qx - \lambda e - s = 0, \tag{8}$$

$$e^T x = 1, \tag{9}$$

$$x \in \mathbb{R}^n_+, \tag{10}$$

$$s \in \mathbb{R}^n_+, \tag{11}$$

$$x_j s_j = 0, \quad j = 1, \ldots, n. \tag{12}$$

We remark that the Lagrange multipliers are scaled by a factor of $1/2$ in (8).

For an instance of (StQP), $x \in \Delta_n$ is said to be a *KKT point* if there exist $s \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$ such that conditions (8)–(12) are satisfied. For any KKT point $x \in \Delta_n$ of (StQP), it follows from (8), (9), (10), and (12) that (see also [16])

$$\lambda = x^T Q x \geq \nu(Q), \tag{13}$$

where the inequality is exact if and only if x is a global minimizer of (StQP).

2.2 Properties of $\nu(Q)$

The following lemma presents several useful properties about the optimal value function $\nu(\cdot)$.

Lemma 1 *For any $Q \in \mathcal{S}^n$, $Q_1 \in \mathcal{S}^n$, $Q_2 \in \mathcal{S}^n$, and $\gamma \in \mathbb{R}$, the following relations are satisfied:*

(i) $\nu(Q + \gamma ee^T) = \nu(Q) + \gamma$.
(ii) *If $Q_1 - Q_2 \in \mathcal{N}^n$, then $\nu(Q_1) \geq \nu(Q_2)$.*
(iii) *If Q is a diagonal matrix with strictly positive diagonal entries $Q_{11}, \ldots, Q_{nn}$, then*

$$\nu(Q) = \frac{1}{\sum_{k=1}^{n} (1/Q_{kk})}.$$

(iv) *Let $\gamma_0 = \min_{1 \leq i \leq j \leq n} Q_{ij}$ and $\gamma_1 = \min_{k=1,\ldots,n} Q_{kk}$. Then, $\gamma_0 \leq \nu(Q) \leq \gamma_1$.*

Proof (i) For any $Q \in \mathcal{S}^n$ and any $\gamma \in \mathbb{R}$, we have

$$\nu(Q + \gamma ee^T) = \min\{x^T(Q + \gamma ee^T)x : x \in \Delta_n\} = \gamma + \min\{x^T Q x : x \in \Delta_n\} = \nu(Q) + \gamma.$$

(ii) Let $Q_1 - Q_2 \in \mathcal{N}^n$. Then, for any $x \in \Delta_n$, we have $x^T(Q_1 - Q_2)x \geq 0$, which implies that

$$x^T Q_1 x \geq x^T Q_2 x, \quad \forall\, x \in \Delta_n,$$

from which the assertion follows.

(iii) If Q is a diagonal matrix with strictly positive diagonal entries $Q_{11}, \ldots, Q_{nn}$, then $Q \in \mathcal{N}^n$, which implies that $\nu(Q) \geq 0$. For any KKT point $x \in \Delta_n$, one obtains

$$Q_{jj} x_j - \lambda - s_j = 0, \quad j = 1, \ldots, n.$$

First, it is easy to see that $s_j = 0$ for each $j = 1, \ldots, n$, since if $s_k > 0$ for some $k \in \{1, \ldots, n\}$, then we would have $x_k = 0$ by (12), which would imply that $\lambda = x^T Q x = -s_k < 0$ by (13), which contradicts $\nu(Q) \geq 0$. Therefore, we obtain $x_j = \lambda / Q_{jj}$ for each $j = 1, \ldots, n$. Combining this with $e^T x = 1$ yields that the unique KKT point satisfies

$$x_k = \frac{1/Q_{kk}}{\sum_{k=1}^{n} (1/Q_{kk})}, \quad k = 1, \ldots, n.$$

Substituting this solution in the objective function yields the result.

(iv) Let $\gamma_0 = \min_{1 \leq i \leq j \leq n} Q_{ij}$. Then, $Q - \gamma_0 ee^T \in \mathcal{N}^n$, which implies that $0 \leq \nu(Q - \gamma_0 ee^T) = \nu(Q) - \gamma_0$, where we used (i). Therefore, $\gamma_0 \leq \nu(Q)$. Furthermore, since $e_k \in \Delta_n$ for each $k = 1, \ldots, n$, we have $\nu(Q) \leq \min_{k=1,\ldots,n} e_k^T Q e_k = \min_{k=1,\ldots,n} Q_{kk} = \gamma_1$.

□

2.3 Properties of an optimal solution

In this section, we present a useful relation between an optimal solution of a standard quadratic program and a related graph.

Given an instance of (StQP) with $Q \in \mathcal{S}^n$, we can associate with it an undirected graph $G = (V, E)$, called the *convexity graph of* Q, where $V = \{1, 2, \ldots, n\}$ with node j corresponding to the vertex of the unit simplex $e_j,\ j = 1, \ldots, n$. There is an edge between node i and node j if the restriction of the quadratic form $x^T Qx$ to the edge of the unit simplex between the vertices e_i and e_j is strictly convex, i.e.,

$$E = \{(i, j) : Q_{ii} + Q_{jj} - 2Q_{ij} > 0, \quad 1 \le i < j \le n\}.$$

For $x \in \Delta_n$, we introduce the following index sets:

$$\mathcal{P}(x) = \left\{j \in \{1, \ldots, n\} : x_j > 0\right\}, \tag{14}$$

$$\mathcal{Z}(x) = \left\{j \in \{1, \ldots, n\} : x_j = 0\right\}. \tag{15}$$

The indices in $\mathcal{P}(x)$ are referred to as the *support* of x. For a given undirected graph $G = (V, E)$, a set $C \subseteq V$ of nodes is called a *clique* if each pair of nodes is connected by an edge. Similarly, a set $S \subseteq V$ of nodes is called a *stable set* if no two nodes in S are connected by an edge. The following theorem [29] establishes a useful connection between the support of a global solution of (StQP) and the convexity graph $G = (V, E)$.

Theorem 1 (Scozzari and Tardella, 2008) *Given an instance of (StQP), let* $G = (V, E)$ *denote the convexity graph of* Q*. Then, there exists a globally optimal solution* $x^* \in \Delta_n$ *of (StQP) such that the nodes corresponding to the indices in* $\mathcal{P}(x^*)$ *(i.e., the support of* x^**) in* G *form a clique (or, equivalently, a stable set in the complement of* G*).*

2.4 Lower bounds on $\nu(Q)$

In this section, given an instance of (StQP), we review two lower bounds on $\nu(Q)$.

2.4.1 A simple lower bound

We start with a simple lower bound on $\nu(Q)$. By Lemma 1(iv),

$$\nu(Q) \ge \gamma_0 = \min_{1 \le i \le j \le n} Q_{ij},$$

with equality if there exists $k \in \{1, \ldots, n\}$ such that $\gamma_0 = \gamma_1 = Q_{kk}$.

This lower bound can be slightly sharpened if the minimum entry of Q is not on the main diagonal, i.e. if $\gamma_0 < \gamma_1$. In this case, we have $Q - \gamma_0 ee^T \in \mathcal{N}^n$ with strictly positive diagonal elements, which can be decomposed as $Q - \gamma_0 ee^T = D + F$, where $D \in \mathcal{N}^n$ and $F \in \mathcal{N}^n$ are such that D is a diagonal matrix with strictly positive entries given by $D_{kk} = Q_{kk} - \gamma_0,\ k = 1, \ldots, n$, along the main diagonal, and all diagonal entries of F are equal to zero. Since $(Q - \gamma_0 ee^T) - D = F \in \mathcal{N}^n$, it follows by Lemma 1(i), (ii), and (iii) that

$$\nu(Q - \gamma_0 ee^T) = \nu(Q) - \gamma_0 \ge \nu(D) = \frac{1}{\sum_{k=1}^{n} (1/D_{kk})} = \frac{1}{\sum_{k=1}^{n} (1/(Q_{kk} - \gamma_0))}.$$

This gives rise to the following lower bound on $\nu(Q)$ (see, e.g., [8]):

$$\text{(LB1)} \quad \nu(Q) \geq \ell_1(Q) := \min_{1\leq i\leq j\leq n} Q_{ij} + \frac{1}{\sum_{k=1}^{n}\left(1/(Q_{kk} - \min_{1\leq i\leq j\leq n} Q_{ij})\right)},$$

where we define $1/0 = \infty$, $1/\infty = 0$, and $\beta + \infty = \infty$ for any $\beta \in \mathbb{R}$. These definitions imply that $\ell_1(Q) = \min_{1\leq i\leq j\leq n} Q_{ij} = \gamma_0$ if and only if $\min_{1\leq i\leq j\leq n} Q_{ij} = \min_{k=1,\ldots,n} Q_{kk}$.

2.4.2 Lower bound from doubly nonnegative relaxation

In this section, we present another lower bound on $\nu(Q)$ using an alternative formulation of (StQP).

A standard quadratic program can be equivalently reformulated as the following instance of a linear optimization problem over the convex cone of completely positive matrices [7]:

$$\text{(CPP)} \quad \nu(Q) = \min\{\langle Q, X\rangle : \langle E, X\rangle = 1, \quad X \in \mathcal{CP}^n\},$$

where $X \in \mathcal{S}^n$ and $\mathcal{CP}^n$ is given by (5). Despite the fact that (CPP) is a convex reformulation of (StQP), it remains NP-hard since the membership problem for the cone of completely positive matrices is intractable (see, e.g., [14]).

By (7), one can replace the intractable cone of completely positive matrices in (CPP) by the larger but tractable cone of doubly nonnegative matrices so as to obtain the following doubly nonnegative relaxation of (CPP):

$$\text{(DNN)} \quad \min\left\{\langle Q, X\rangle : \langle E, X\rangle = 1, \quad X \in \mathcal{DNN}^n\right\},$$

where $\mathcal{DNN}^n$ is given by (6).

Therefore, another lower bound on $\nu(Q)$ is given by

$$\text{(LB2)} \quad \nu(Q) \geq \ell_2(Q) := \min\left\{\langle Q, X\rangle : \langle E, X\rangle = 1, \quad X \in \mathcal{DNN}^n\right\}.$$

By [8, Theorem 13], we have the following relation:

$$\ell_1(Q) \leq \ell_2(Q) \leq \nu(Q), \tag{16}$$

i.e., the lower bound $\ell_2(Q)$ is at least as tight as $\ell_1(Q)$. By Lemma 1(iv) and (16), both lower bounds are exact if the minimum entry of Q lies along the diagonal. However, as illustrated by our computational results in Sect. 4, $\ell_2(Q)$ is, in general, much tighter than $\ell_1(Q)$.

Note that $\ell_1(Q)$ can be computed in $O(n^2)$ time whereas $\ell_2(Q)$ requires solving a computationally expensive semidefinite program. We remark that there exist other lower bounds in the literature (see, e.g., [1,6,8,8,28], and see [8] for a comparison of different lower bounds). Usually, there is a trade-off between the quality of the lower bound and its computational cost.

3 Mixed integer linear programming formulations

In this section, we present two different mixed integer linear programming (MILP) reformulations of standard quadratic programs. We then propose a set of inequalities that are valid for both formulations.

3.1 A formulation based on KKT conditions

Our first MILP formulation is obtained by exploiting the KKT conditions. We discuss how the nonlinear complementarity constraints can be linearized by employing binary variables. We also discuss how to obtain valid upper bounds for the big-M parameters that arise from this linearization.

Given an instance of (StQP), it follows from (13) that $\lambda = x^T Qx$ for any KKT point $x \in \Delta_n$. Therefore, (StQP) can be equivalently formulated as the following linear program with complementarity constraints (see also [16]):

$$\begin{aligned} \text{(LPCC1)} \quad \min \quad & \lambda \\ \text{s.t.} \quad & Qx - \lambda e - s = 0, \\ & e^T x = 1, \\ & x_j s_j = 0, \quad j = 1, \ldots, n, \\ & x \geq 0, \\ & s \geq 0. \end{aligned}$$

We can linearize the nonconvex complementarity constraints in (LPCC1) by using binary variables and big-M constraints, which gives rise to the following MILP reformulation of (StQP):

$$\begin{aligned} \text{(MILP1)} \quad \min \quad & \lambda && (17) \\ \text{s.t.} \quad & Qx - \lambda e - s = 0, && (18) \\ & e^T x = 1, && (19) \\ & x_j \leq y_j, \quad j = 1, \ldots, n, && (20) \\ & s_j \leq M_j(1 - y_j), \quad j = 1, \ldots, n, && (21) \\ & x \geq 0, && (22) \\ & s \geq 0, && (23) \\ & y_j \in \{0, 1\}, \quad j = 1, \ldots, n. && (24) \end{aligned}$$

Note that, by (20) and (21), the binary variable y_j ensures that x_j and s_j cannot simultaneously be positive for any $j = 1, \ldots, n$. In particular, if $y_j = 1$, then $s_j = 0$ by (21) and x_j is allowed to be positive. Since $x \in \Delta_n$, we have $0 \leq x_j \leq 1$, which implies that (20) yields a valid upper bound on x_j. On the other hand, if $y_j = 0$, we have $x_j = 0$ by (20). In this case, we need valid upper bounds on the variable s_j in (21), which we discuss next.

By (18),

$$s_j = e_j^T Qx - \lambda, \quad j = 1, \ldots, n. \tag{25}$$

We can obtain an upper bound on s_j by deriving an upper bound for each of the terms on the right-hand side of (25). For the first term, since $x \in \Delta_n$, we have

$$e_j^T Qx = x^T Qe_j \leq \max_{i=1,\ldots,n} Q_{ij}, \quad j = 1, \ldots, n. \tag{26}$$

In order to bound the second term from above, any lower bound on λ can be employed. Since $\lambda \geq \nu(Q)$ for any feasible solution of (MILP1), it follows that any lower bound on $\nu(Q)$ can be used to obtain an upper bound on s_j, $j = 1, \ldots, n$. Indeed, let ℓ denote an arbitrary

lower bound on $\nu(Q)$. For any feasible solution (x, y, s, λ) of (MILP1), it follows from (25) and (26) that

$$s_j = e_j^T Qx - \lambda \leq \max_{i=1,\ldots,n} Q_{ij} - \nu(Q) \leq \max_{i=1,\ldots,n} Q_{ij} - \ell, \quad j = 1, \ldots, n,$$

which implies that

$$M_j = \max_{i=1,\ldots,n} Q_{ij} - \ell, \quad j = 1, \ldots, n \tag{27}$$

would be a valid choice in (MILP1). In particular, we use $\ell \in \{\ell_1(Q), \ell_2(Q)\}$ in our computational experiments.

3.2 An alternative formulation

In this section, we present an alternative MILP formulation. Given an instance of (StQP), we first derive an underestimator and an overestimator for the quadratic objective function. We then establish useful properties of these two functions, which form the basis of our second formulation.

We start with a lemma that presents an underestimator and an overestimatior for the objective function of (StQP), both of which depend on the support of a feasible solution defined in (14).

Lemma 2 *For any $Q \in \mathcal{S}^n$ and $x \in \Delta_n$, we have*

$$\min_{j \in \mathcal{P}(x)} e_j^T Qx \leq x^T Qx \leq \max_{j \in \mathcal{P}(x)} e_j^T Qx, \tag{28}$$

where $\mathcal{P}(x)$, given by (14), denotes the support of x. Furthermore, if $x \in \Delta_n$ is a KKT point of (StQP), then

$$\min_{j \in \mathcal{P}(x)} e_j^T Qx = x^T Qx = \max_{j \in \mathcal{P}(x)} e_j^T Qx. \tag{29}$$

Proof For any $Q \in \mathcal{S}^n$ and $x \in \Delta_n$,

$$x^T Qx = \left(\sum_{j=1}^n x_j e_j^T \right) Qx = \sum_{j=1}^n x_j \left(e_j^T Qx \right) = \sum_{j \in \mathcal{P}(x)} x_j \left(e_j^T Qx \right),$$

i.e., $x^T Qx$ is a convex combination of $e_j^T Qx$ for $j \in \mathcal{P}(x)$, from which (28) follows.

Let $x \in \Delta_n$ be a KKT point of (StQP). Then, by (8)–(12),

$$e_j^T Qx = \lambda, \quad j \in \mathcal{P}(x),$$
$$e_j^T Qx \geq \lambda, \quad j \in \mathcal{Z}(x).$$

Furthermore $x^T Qx = \sum_{j \in \mathcal{P}(x)} x_j \left(e_j^T Qx \right) = \lambda \left(\sum_{j \in \mathcal{P}(x)} x_j \right) = \lambda$, which establishes (29). □

Using Lemma 2, we next present an alternative characterization of $\nu(Q)$.

Proposition 1 *Given an instance of (StQP),*

$$\nu(Q) = \min_{x \in \Delta_n} \max_{j \in \mathcal{P}(x)} e_j^T Qx. \tag{30}$$

Proof For any $x \in \Delta_n$, we have $x^T Qx \leq \max_{j \in \mathcal{P}(x)} e_j^T Qx$ by Lemma 2, which implies that $\nu(Q) \leq \min_{x \in \Delta_n} \max_{j \in \mathcal{P}(x)} e_j^T Qx$. Conversely, let $x^* \in \Delta_n$ be an optimal solution of (StQP). Then, x^* is a KKT point, which implies that $\nu(Q) = (x^*)^T Qx^* = \max_{j \in \mathcal{P}(x^*)} e_j^T Qx^*$ by Lemma 2, which establishes the reverse inequality. □

We are now in a position to propose an alternative MILP formulation based on the characterization (30) stated in Proposition 1.

$$\begin{aligned}
\text{(MILP2)} \quad \min \quad & \alpha && (31)\\
\text{s.t.} \quad & \\
& e_j^T Qx \leq \alpha + z_j, \quad j = 1, \ldots, n, && (32)\\
& e^T x = 1, && (33)\\
& x_j \leq y_j, \quad j = 1, \ldots, n, && (34)\\
& z_j \leq U_j(1 - y_j), \quad j = 1, \ldots, n, && (35)\\
& x \geq 0, && (36)\\
& z \geq 0, && (37)\\
& y_j \in \{0, 1\}, \quad j = 1, \ldots, n. && (38)
\end{aligned}$$

Note that the auxiliary variable α is introduced and used in (31) and (32) to linearize the maximum function on the right-hand side of (30) and the binary variables y_j are employed in (35) to ensure that the maximum is restricted only to the linear functions corresponding to the support of x in (32). Indeed, if $j \in \mathcal{P}(x)$, then $x_j > 0$, which forces $y_j = 1$ by (34) and $z_j = 0$ by (35). Otherwise, (32) is a redundant constraint since z_j can then take a positive value. Note that we again rely on big-M parameters U_j in (35). The next proposition presents a valid bound for these parameters.

Proposition 2 *Given an instance of (StQP), (MILP2) is an equivalent reformulation of (StQP) if*

$$U_j \geq M_j, \quad j = 1, \ldots, n,$$

where M_j is defined as in (27) and ℓ is any lower bound on $\nu(Q)$.

Proof Let $x \in \Delta_n$. Then, for each $j \in \mathcal{P}(x)$, we have $y_j = 1$ by (34), which implies that $z_j = 0$ by (35) and $\alpha \geq \max_{j \in \mathcal{P}(x)} e_j^T Qx \geq x^T Qx$ by (32) and by Lemma 2. Since the objective function minimizes α, the best choice of α would be given by $\alpha = \max_{j \in \mathcal{P}(x)} e_j^T Qx$.

Consider an index $j \notin \mathcal{P}(x)$. Let us define $z_j = \max\{0, e_j^T Qx - \alpha\} \geq 0$. Then, if $e_j^T Qx - \alpha < 0$, we have $z_j = 0 \leq M_j$, which satisfies the constraints of (MILP2). Otherwise,

$$z_j = e_j^T Qx - \alpha \leq \max_{i=1,\ldots,n} Q_{ij} - \alpha \leq \max_{i=1,\ldots,n} Q_{ij} - x^T Qx \leq \max_{i=1,\ldots,n} Q_{ij} - \nu(Q),$$

which implies that $z_j \leq \max_{i=1,\ldots,n} Q_{ij} - \ell = M_j$, where ℓ is any lower bound on $\nu(Q)$. It follows that for each $x \in \Delta_n$, we can construct $y \in \mathbb{R}^n$, $z \in \mathbb{R}^n$, and $\alpha \in \mathbb{R}$ such that $\alpha = \max_{j \in \mathcal{P}(x)} e_j^T Qx \geq x^T Qx$. By Lemma 2, if $x \in \Delta_n$ is a KKT point, then we can choose $\alpha = x^T Qx$. The equivalence follows. □

By Proposition 2, (MILP2) can be viewed as a majorization minimization approach for (StQP), where the majorizing function is exact at any KKT point.

We close this section by a brief comparison of (MILP1) and (MILP2). Note that the constraint set (32) in (MILP2) can be rewritten as

$$Qx - \alpha e - z \leq 0.$$

Identifying the variables z with s and α with λ, a comparison with (18) in (MILP1) reveals that (MILP2) is in fact a relaxation of (MILP1). It follows that, for any feasible solution (x, y, s, λ) of (MILP1), we can define $z = s$ and $\alpha = \lambda$ so that (x, y, z, α) is a feasible solution of (MILP2). On the other hand, while each feasible solution (x, y, s, λ) of (MILP1) necessarily corresponds to a KKT point of (StQP), we can construct a feasible solution (x, y, z, α) for *any* $x \in \Delta_n$ such that $\alpha \geq x^T Qx$, with equality if x is a KKT point of (StQP). Therefore, (MILP2) can be viewed as an *exact relaxation* of (MILP1).

3.3 Valid inequalities

In this section, we present a set of inequalities that are valid for both formulations (MILP1) and (MILP2).

Given an instance of (StQP), Theorem 1 presents a relation between the support of an optimal solution and the convexity graph of Q. This relation gives rise to the following theorem.

Theorem 2 *The following inequalities are valid for both formulations (MILP1) and (MILP2):*

$$y_i + y_j \leq 1, \qquad 1 \leq i < j \leq n \;\; s.t. \;\; Q_{ii} + Q_{jj} - 2Q_{ij} \leq 0. \tag{39}$$

Proof In both formulations (MILP1) and (MILP2), the binary variables y_j are equal to one if $j \in \mathcal{P}(x)$ for any feasible solution $x \in \Delta_n$. By Theorem 1, there exists a global solution of (StQP) whose support set forms a stable set in the complement of the convexity graph $G = (V, E)$ of Q. The assertion follows. □

4 Computational results

We report the results of our computational experiments in this section. We first describe the set of instances used in our experiments. Then, we explain our experimental setup in detail. Finally, we report performances of the proposed MILP formulations in comparison with several other global solution approaches from the literature.

4.1 Set of instances

In an attempt to accurately assess the performances of the MILP formulations (MILP1) and (MILP2), we conducted extensive experiments on the following set of instances from the literature:

(i) *BLST instances* [10]: This set consists of 150 instances[1] with $n = 30$ (BLST30) and 150 instances with $n = 50$ (BLST50). Each entry of Q is randomly generated from a

[1] Publicly available at http://or.dei.unibo.it/library/msc.

triangular distribution with parameters $a < c < b$, where a and b denote the minimum and maximum values and c is the mode of the distribution.

(ii) *ST instances* [29]: This set consists of 24 instances[2] with $n = 100$ (ST100), 18 instances with $n = 200$ (ST200), 11 instances with $n = 500$ (ST500), and 1 instance with $n = 1000$ (ST1000). For each of these instances, the matrix Q is randomly generated so that its convexity graph $G = (V, E)$ has a prespecified density $\delta \in [0, 1]$, where the density of an undirected graph is given by the ratio of the number of edges to the maximum possible number of edges. Note that these instances are generated by constructing a matrix $Q \in \mathcal{S}^n$ such that $Q_{ij} = 0.5(Q_{ii}+Q_{jj})-R_{ij}$, for $1 \leq i < j \leq n$, where $R_{ij} > 0$ with probability δ and $R_{ij} < 0$ with probability $1 - \delta$ (see [28]).

(iii) *DIMACS instances*: It is well-known that the maximum stable set problem in graph theory can be formulated as an instance of (StQP) [27]. This set consists of (StQP) instances obtained from the complements of the 30 instances of the maximum clique problem[3] from the Second DIMACS Implementation Challenge with $n \in [28, 300]$. These instances are divided into two groups based on the number of vertices. DIMACS1 consists of 8 instances with $n \in [28, 171]$ and DIMACS2 is comprised of 22 instances with $n \in [200, 300]$.

(iv) *BSU instances* [9]: This set consists of 20 "hard" instances with $n \in [5, 24]$. Each of these instances is specifically constructed to harbor an exponential number of strict local minimizers. In particular, the number of strict local minimizers varies between 1.38^n and 1.49^n.

Note that Theorem 1 establishes a relation between the support of a global minimizer of an instance of (StQP) and the set of cliques of the associated convexity graph $G = (V, E)$. Denoting the density of the convexity graph by $\delta \in [0, 1]$, it follows that the number of cliques in G tends to increase as δ increases. Therefore, instances of (StQP) with larger values of δ contain a larger number of possible support sets for a global minimizer. Indeed, this difficulty is also reflected in earlier computational experiments (see, e.g. [25,29]). It is also worth mentioning that the number of valid inequalities (39) is given by $(1-\delta)n(n-1)/2$. Therefore, for fixed n, the number of valid inequalities decreases as δ increases. For each set of instances, we therefore report the range of the parameter δ.

Recall that, for an instance of (StQP), if the minimum entry of Q lies along the main diagonal, then $\nu(Q)$ equals that entry by Lemma 1(iv). Therefore, we use this criterion as a preprocessing step in order to eliminate trivial instances. Apart from this, we do not use any other preprocessing procedure. After eliminating such trivial instances, we obtain a test bed that consists of a total of 376 instances. We summarize the statistics on the set of instances in Table 1.

As illustrated by Table 1, our test bed encompasses a large number of instances of (StQP) with varying sizes and characteristics. In particular, we note the higher density of the convexity graphs associated with the hard BSU instances. Indeed, 14 instances in this set have a density of 1; 5 instances have densities between 0.92 and 0.95; and only one instance has a density of 0.6, supporting our previous observation regarding the correlation between the difficulty of an instance and the density of the associated convexity graph.

[2] Publicly available at http://www.iasi.cnr.it/~liuzzi/StQP/.

[3] Publicly available at http://wwwdimacs.dimacs.rutgers.edu/pub/challenge/graph/benchmarks/clique/.

Table 1 Summary of instances

Instance family	Number of instances	n	δ
BLST30	134	30	[0.10,1.00]
BLST50	138	50	[0.11,1.00]
ST100	24	100	[0.25,0.90]
ST200	18	200	[0.25,0.75]
ST500	11	500	[0.25,0.50]
ST1000	1	1000	0.25
DIMACS1	8	[28,171]	[0.35,0.93]
DIMACS2	22	[200,300]	[0.08,0.97]
BSU	20	[5,24]	[0.6,0.1]
Overall	376	[5,1000]	[0.08,1]

4.2 Experimental setup

Note that we propose two MILP formulations (MILP1) and (MILP2). For each formulation, one can use the lower bound $\ell_1(Q)$ or $\ell_2(Q)$. Finally, for each choice of the lower bound, we have the option of adding the valid inequalities (39) or not, which implies that we have a total of 8 variants. For each variant with (MILP1), we add the following bound constraints on the variable λ:

$$\ell_1(Q) \leq \lambda \leq \min_{k=1,\ldots,n} Q_{kk}, \tag{40}$$

where the upper bound follows from Lemma 1(iv). Similarly, the corresponding constraints are added for each variant with (MILP2):

$$\ell_2(Q) \leq \alpha \leq \min_{k=1,\ldots,n} Q_{kk}. \tag{41}$$

We compare the performances of our MILP formulations with three other global solution approaches, namely, the MILP formulation of [31], which is publicly available at https://github.com/xiawei918/quadprogIP, the quadratic programming (QP) solver of CPLEX, and the nonlinear programming (NLP) solver BARON.

We solved all MILP formulations in MATLAB (version R2017b) using CPLEX (version 12.8.0) with the CPLEX Class API provided in CPLEX for MATLAB Toolbox. Similarly, the QP solver of CPLEX was called in MATLAB using the same API. The NLP solver BARON (version 17.4.1) was called from GAMS (version 24.8.5) using the MATLAB interface. The computation of $\ell_2(Q)$ requires a semidefinite programming solver. For that purpose, we employed MOSEK (version 8.1.0.49) using the MOSEK Optimization Toolbox for MATLAB (version 8.1.0.82).

We measured the running times in terms of wall clock time. In our experiments with CPLEX and BARON, we imposed a time limit of 3600 s for each MILP and QP problem. No time limit was imposed on MOSEK for the computation of $\ell_2(Q)$. Our computational experiments were carried out on a 64-bit HP workstation with 24 threads (2 sockets, 6 cores per socket, 2 threads per core) running Ubuntu Linux with 48 GB of RAM and Intel Xeon CPU E5-2667 processors with a clock speed of 2.90 GHz. In our experiments with CPLEX, we chose the deterministic parallel mode by setting `cplex.parallelmode = 1` in order to have reproducible results. We remark that both CPLEX and MOSEK can take advantage of

multiple threads. Similarly, we employed `option threads = 24` in GAMS. However, we noticed that the wall clock time and the CPU time reported by BARON were virtually identical on all instances, suggesting that BARON did not take advantage of the multiple threads. Therefore, we caution the reader about the interpretation of the run times in our experiments with BARON.

In CPLEX and BARON, we set the optimality gap tolerance to 10^{-6}, which is given by

$$\frac{|\texttt{bestbound} - \texttt{bestsolution}|}{10^{-10} + |\texttt{bestsolution}|}. \tag{42}$$

We employed the default settings for all the other parameters of CPLEX, BARON, and MOSEK.

We denote by MILP1-L1 and MILP1-L1-VI the MILP formulation (MILP1) using the lower bound $\ell_1(Q)$ with and without the set of valid inequalities (39), respectively. We replace the suffix L1 by L2 for the lower bound $\ell_2(Q)$. We use a similar convention for (MILP2). The MILP formulation of [31] is denoted by QP-IP, whereas CPLEX QP and BARON refer to the QP formulations solved by CPLEX and BARON, respectively. The doubly nonnegative relaxation (DNN) is denoted by DNN. For our MILP formulations with the lower bound $\ell_2(Q)$, the solution times exclude the computational effort for solving the corresponding doubly nonnegative relaxation (DNN) required to compute this lower bound, which is reported separately. Finally, for a solution $x \in \Delta_n$ reported by a solver, an index $j \in \{1, \ldots, n\}$ is considered to be in the support of x (i.e., $j \in \mathcal{P}(x)$) if $x_j > 10^{-8}$.

For each data set, we report our results using a table that presents various summary statistics and a performance profile. In each table, we have a row for each of the eight variants using (MILP1) and (MILP2), and a row for each of QP-IP, CPLEX QP, BARON, and DNN. The first set of columns reports the number of instances solved to optimality within the time limit of 3600 s, the average solution time over these instances, and the standard deviation. We present the number of instances on which the corresponding approach hits the time limit and the average optimality gap defined as in (42) over these instances in the second set of columns. Recall that we do not impose any time limit for solving the DNN relaxation. In an attempt to shed more light on the comparison of the performances of different approaches, we report performance profiles [15], which are frequently used for benchmarking purposes in optimization. For a given problem instance, the performance ratio of a particular approach is defined as the ratio of the solution time of that approach to the best solution time among all approaches on that particular instance, which is defined to be $+\infty$ if the approach fails to solve the instance within the given time limit. Then, for each approach a, a cumulative distribution function $P_a(\tau)$ is defined to be the percentage of the number of instances that can be solved by that approach within a factor of τ of the solution time of the best approach, where $\tau \in [1, \infty)$. Note that only the instances that can be solved by at least one approach within the time limit are included in the comparison. We remark that performance ratios are reported in logarithmic scale. Furthermore, the markers are included for every kth instance, where k is judiciously chosen according to the number of instances in each set in such a way that each figure has roughly the same number of markers.

4.3 BLST instances

In this section, we report the performances on the BLST data set [10]. Recall that BLST30 consists of 134 instances with $n = 30$ and BLST50 is comprised of 138 instances with $n = 50$.

Table 2 Performance on BLST30 (134 instances; $n = 30$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	134	0.20	0.11	0	–
MILP1-L1-VI	134	0.23	0.13	0	–
MILP2-L1	134	0.23	0.12	0	–
MILP2-L1-VI	134	0.25	0.11	0	–
MILP1-L2	134	0.26	0.12	0	–
MILP1-L2-VI	134	0.18	0.11	0	–
MILP2-L2	134	0.19	0.11	0	–
MILP2-L2-VI	134	0.22	0.11	0	–
QP-IP	134	0.28	0.10	0	–
CPLEX QP	124	6.84	13.94	10	6.14
BARON	96	533.16	793.44	38	115.91
DNN	134	0.11	0.01	–	–

Table 3 Performance on BLST50 (138 instances; $n = 50$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	138	0.36	0.42	0	–
MILP1-L1-VI	138	0.37	0.42	0	–
MILP2-L1	138	0.36	0.35	0	–
MILP2-L1-VI	138	0.40	0.37	0	–
MILP1-L2	138	0.33	0.46	0	–
MILP1-L2-VI	138	0.30	0.47	0	–
MILP2-L2	138	0.22	0.13	0	–
MILP2-L2-VI	138	0.26	0.14	0	–
QP-IP	138	0.62	0.87	0	–
CPLEX QP	124	211.45	594.28	14	13.74
BARON	19	138.61	512.04	95	246.14
DNN	138	0.72	0.09	–	–

On each of the BLST30 and BLST50 instances, the lower bound $\ell_2(Q)$ (i.e., the optimal value of (DNN)) is either equal to $\nu(Q)$ or the difference between the two is at most 10^{-6}, whereas the simple lower bound $\ell_1(Q)$ is always smaller than $\nu(Q)$. Therefore, $\ell_2(Q)$ is significantly tighter than $\ell_1(Q)$ over all instances. The support sizes of optimal solutions are in the range [1, 10].

We report the summary of the results in Tables 2 and 3 for the BLST30 and BLST50 instances, respectively.

As illustrated by Tables 2 and 3, our MILP formulations as well as QP-IP can solve each instance to optimality in a fraction of a second on average. On the other hand, each of CPLEX QP and BARON hits the time limit on some of the instances, with CPLEX QP exhibiting a better performance than BARON. Note that the average gaps of BARON are usually considerably higher than those reported by CPLEX QP. On each set, the average time taken by each of CPLEX QP and BARON is significantly larger than that required for

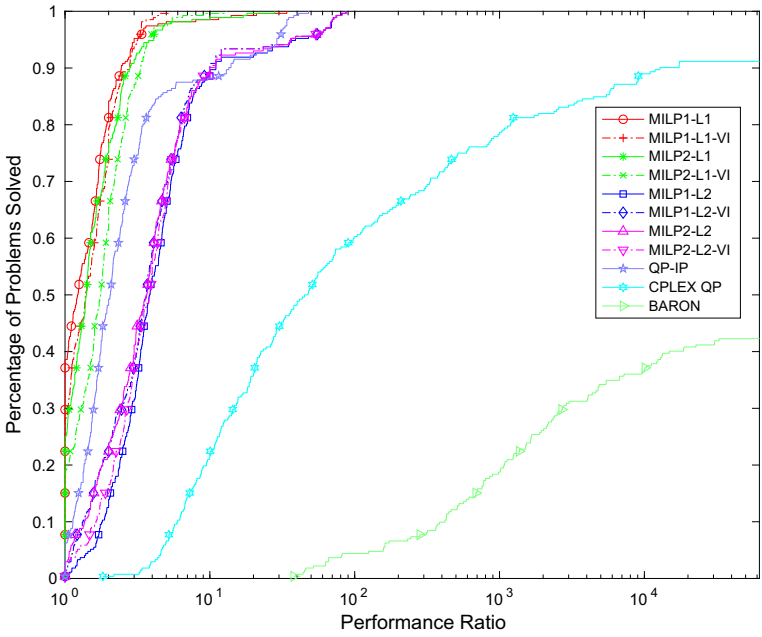

Fig. 1 Performance profile on BLST instances

each MILP formulation. In terms of average solution times of MILP models, each of the eight variants of (MILP1) and (MILP2) slightly outperforms QP-IP, where the improvement is more pronounced on BLST50. We remark that the average time of each of our MILP formulations on BLST50 increases only modestly in comparison with that on BLST30. We particularly highlight the better and more robust performances of MILP2-L2 and MILP2-L2-VI on BLST50. Finally, while the average computational effort for solving the DNN relaxation is somewhat negligible on BLST30, it increases significantly on BLST50, even surpassing the average time required for solving each MILP formulation.

The performance profile on BLST instances is illustrated in Fig. 1. In an attempt to illustrate the total computational effort in the performance profile, we add the computation time of the DNN relaxation to the solution time of each variant of our model that uses the lower bound $\ell_2(Q)$. As indicated by Fig. 1, each of the eight variants of our MILP formulations outperforms CPLEX QP and BARON, even with the inclusion of the computational effort for solving the DNN relaxation. On the other hand, while each of the four variants of our MILP formulations that uses the simple lower bound $\ell_1(Q)$ outperforms QP-IP, the additional computational effort for the DNN relaxation outweighs the benefits of the MILP formulations that rely on the tighter lower bound $\ell_2(Q)$. In an attempt to give a better comparison of the MILP based approaches, we also include the performance profile excluding CPLEX QP and BARON in Fig. 2, which clearly illustrates that the best performance ratios are achieved by MILP1-L1, MILP1-L1-VI, and MILP2-L1. While the valid inequalities seem to slightly improve the performance ratios of MILP1-L2, they do not seem to have a positive effect on the remaining variants.

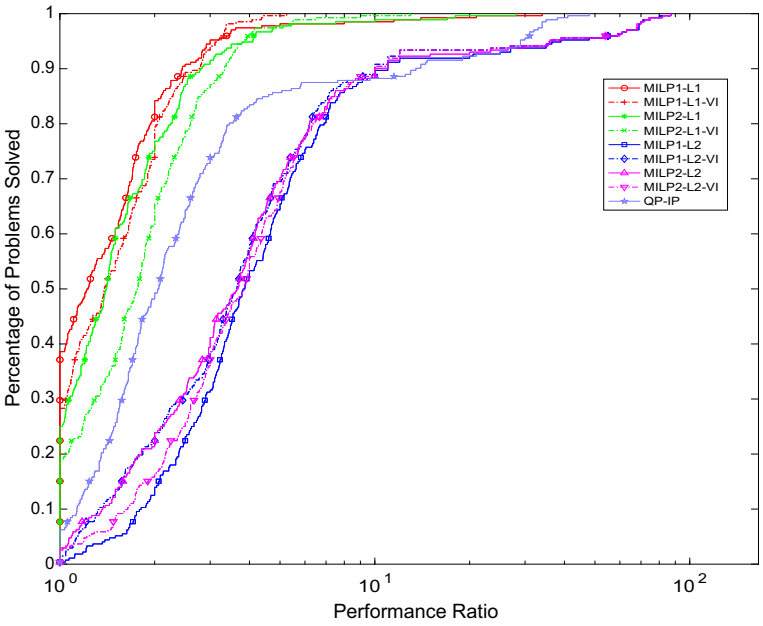

Fig. 2 Performance profile on BLST instances (excluding CPLEX QP and BARON)

4.4 ST instances

In this section, we report our computational results on the ST data set [29], which consists of 24 instances with $n = 100$ (ST100), 18 instances with $n = 200$ (ST200), 11 instances with $n = 500$ (ST500), and one instance with $n = 1000$ (ST1000).

We first focus on ST100 and ST200. On each instance in these data sets, the difference between the lower bound $\ell_2(Q)$ and $\nu(Q)$ is less than 10^{-5}. On the other hand, the simple lower bound $\ell_1(Q)$ is considerably smaller than $\nu(Q)$ on all instances. The support sizes of optimal solutions vary between 3 and 7.

We report our results in Tables 4 and 5 , each of which is organized similarly to Table 2. A close examination of Tables 4 and 5 reveals that MILP formulations significantly outperform each of CPLEX QP and BARON on ST100 and ST200. In particular, all of the instances in these two sets can be solved to optimality by each of our MILP formulations and by QP-IP, whereas CPLEX QP can solve only 6 instances out of 24 in ST100 to optimality and BARON can solve none within the time limit. Furthermore, BARON generally reports considerably higher optimality gaps in comparison with CPLEX QP. Note again that the average computational effort required for each MILP formulation is significantly smaller than that for CPLEX QP and BARON. The average computational effort for solving the DNN relaxation significantly exceeds that required for each MILP formulation. In particular, the average solution time of DNN increases by more than a factor of 25 from ST100 to ST200, despite the fact that n only increases by a factor of 2.

In terms of average solution times of MILP models, each of the eight variants of our MILP formulations outperforms QP-IP on both ST100 and ST200. MILP2-L2 stands out as a clear winner among the other MILP formulations. Note, in particular, that MILP2-L2 is about 13 times faster than QP-IP on ST100 and more than 30 times faster on ST200 on average. In

Table 4 Performance on ST100 (24 instances; $n = 100$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	24	1.59	1.34	0	–
MILP1-L1-VI	24	2.21	1.38	0	–
MILP2-L1	24	0.87	0.66	0	–
MILP2-L1-VI	24	1.17	0.70	0	–
MILP1-L2	24	1.84	2.25	0	–
MILP1-L2-VI	24	2.66	2.85	0	–
MILP2-L2	24	0.45	0.14	0	–
MILP2-L2-VI	24	0.70	0.18	0	–
QP-IP	24	5.63	2.94	0	–
CPLEX QP	6	1008.10	554.62	18	38.18
BARON	0	–	–	24	1131.58
DNN	24	16.18	1.45	–	–

Table 5 Performance on ST200 (18 instances; $n = 200$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	18	32.01	35.86	0	–
MILP1-L1-VI	18	28.47	21.33	0	–
MILP2-L1	18	10.52	10.94	0	–
MILP2-L1-VI	18	12.56	5.18	0	–
MILP1-L2	18	25.75	50.03	0	–
MILP1-L2-VI	18	23.21	27.00	0	–
MILP2-L2	18	2.11	2.50	0	–
MILP2-L2-VI	18	6.93	6.17	0	–
QP-IP	18	66.52	64.07	0	–
CPLEX QP	0	–	–	18	44.63
BARON	0	–	–	18	4249.18
DNN	18	429.43	61.89	–	–

terms of the overall performance, MILP2-L2 is followed by MILP2-L2-VI, MILP2-L1, and MILP2-L1-VI. In particular, the lower bound $\ell_2(Q)$ seems to have a remarkably positive effect on the performance of (MILP2), both with and without valid inequalities and, to a lesser extent, on the performance of (MILP1) on ST200. The inclusion of valid inequalities has a mixed effect on (MILP1) and (MILP2). It is worth mentioning that instances with sparse convexity graphs give rise to a larger number of valid inequalities, which may adversely affect the overall performance of the variants with valid inequalities.

The performance profile on ST100 and ST200 instances, which is illustrated in Fig. 3 and includes the additional computation time of DNN relaxations in the variants of our formulations that rely on the lower bound $\ell_2(Q)$ as in Figs. 1 and 2, reveals that each of the eight variants of our MILP formulations outperforms CPLEX QP and BARON, even with the inclusion of the computational effort for solving the DNN relaxation. On the other hand, this additional computational effort outweighs the benefits of the MILP formulations

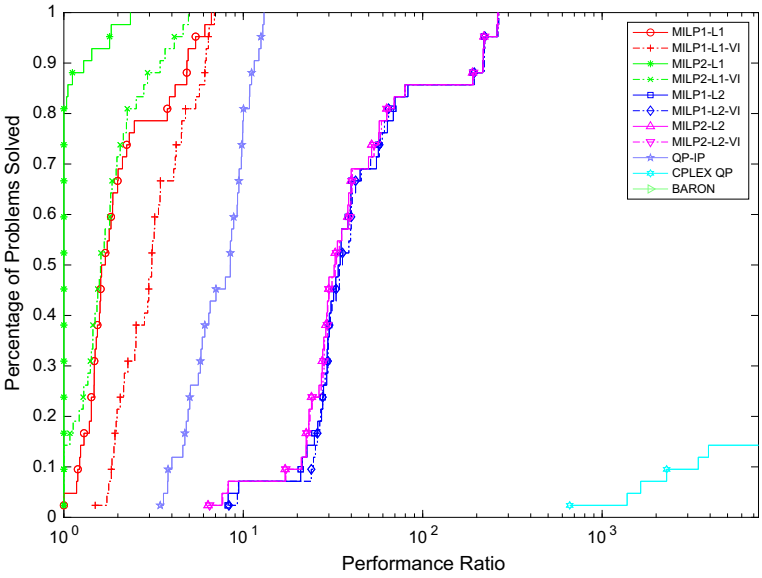

Fig. 3 Performance Profile on ST100 and ST200 Instances

Table 6 Performance on ST500 (11 instances; $n = 500$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	11	598.46	476.870	0	–
MILP1-L1-VI	11	814.55	634.73	0	–
MILP2-L1	11	229.09	176.13	0	–
MILP2-L1-VI	11	345.71	257.63	0	–
QP-IP	6	379.06	37.07	5	6.97
CPLEX QP	0	–	–	11	44660285.88
BARON	0	–	–	11	11376.00

in comparison with QP-IP. In terms of the performance profiles, the ranking is given by MILP2-L1, MILP2-L1-VI, MILP1-L1, and MILP1-L1-VI, followed by QP-IP, which, in turn, is followed by the variants that rely on the lower bound $\ell_2(Q)$. Once again, we note that the inclusion of valid inequalities does not seem to lead to an improved performance on this set. Recall that BARON is not included in the figure since it hits the time limit on all instances in this set.

We now focus on the larger instances ST500 and ST1000. On each of these instances, $\ell_1(Q)$ again turns out to be a rather loose lower bound on $\nu(Q)$. The sizes of the support of optimal solutions are between 4 and 6. Note that the DNN relaxation could not be solved for any instance in ST500 and ST1000. Therefore, the lower bound $\ell_2(Q)$ is not available on this set.

Tables 6 and 7 report the results on ST500 and ST1000, respectively. Note that we omit the rows corresponding to the variants of our MILP formulations with the lower bound $\ell_2(Q)$ and the DNN relaxation in Tables 6 and 7 accordingly. Therefore, we present the results only for the variants of our MILP formulations with the simple lower bound $\ell_1(Q)$. On ST500,

Table 7 Performance on ST1000 (1 instance; $n = 1000$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	0	–	–	1	7.63
MILP1-L1-VI	0	–	–	1	46.74
MILP2-L1	1	2100.22	0	0	–
MILP2-L1-VI	0	–	–	1	11.59
QP-IP	0	–	–	1	22.05
CPLEX QP	0	–	–	1	86423971.55
BARON	0	–	–	1	18020.23

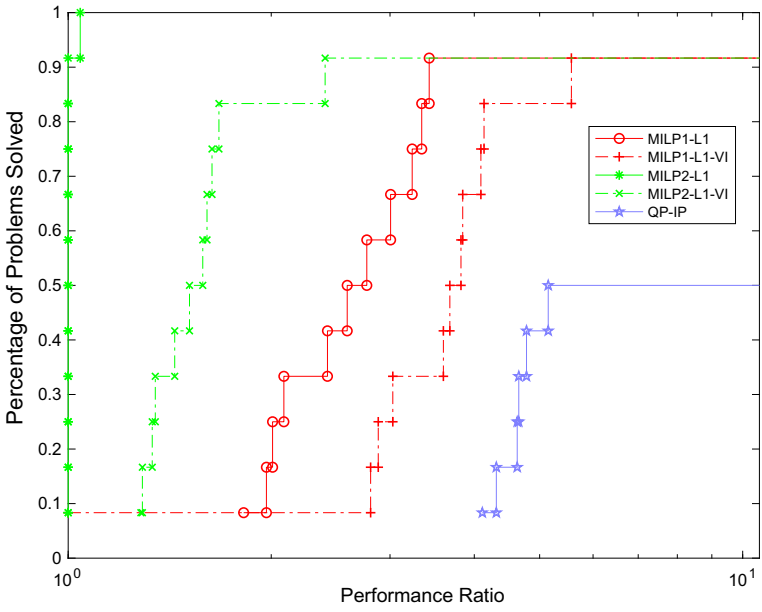

Fig. 4 Performance profile on ST500 and ST1000 instances

while each of our formulations can solve all instances to optimality, QP-IP hits the time limit on 5 of the 11 instances. Furthermore, both CPLEX QP and BARON also hit the time limit on all of the instances. A comparison of average computational effort indicates that our MILP formulations are far more effective than the other approaches, with MILP2-L1 exhibiting the best and most robust performance, followed by MILP2-L1-VI, MILP1-L1, and MILP1-L1-VI.

ST1000 consists of a single instance with $n = 1000$. We include this instance in our experiments to assess the performances of different global approaches on a very large sample instance. This instance indeed turns out to be very challenging for each approach, with only MILP2-L1 being able to solve it to optimality within the time limit. We believe that this instance provides a remarkable computational evidence about the effectiveness of (MILP2), even when used with the simple and fairly loose lower bound $\ell_1(Q)$. We also point out the extremely large optimality gaps reported by the QP solvers, which illustrates that instances of this scale are well beyond the reach of current state-of-the-art QP and NLP solvers. On

this instance, only MILP1-L1-VI reports a larger gap than that of QP-IP. Note that the valid inequalities do not seem to help on these two sets since the total number of such inequalities can be fairly large. For instance, the number of valid inequalities is about 375,000 on the single instance in ST1000. On both of ST500 and ST1000, MILP2-L1 clearly outperforms all the other MILP formulations.

In terms of the performance ratios, the performance profile in Fig. 4 clearly illustrates the superior performance of MILP2-L1, supporting our previous observations, and the better performance of each of our MILP formulations in comparison with QP-IP. Recall again that CPLEX QP and BARON are not included in the figure since each of them hits the time limit on all instances in this set.

4.5 DIMACS instances

In this section, we report our computational results on the DIMACS data set, consisting of 8 instances in DIMACS1 with $n \in [28, 171]$, and 22 instances in DIMACS2 with $n \in [200, 300]$. Each of these instances is obtained from the reformulation of the maximum stable set problem as an instance of (StQP) [27]. We first review this formulation.

Let $G = (V, E)$ be a simple, undirected graph, where $V = \{1, \ldots, n\}$. Recall that a set $S \subseteq V$ is a stable set if no two nodes in S are connected by an edge. The maximum stable set problem is concerned with computing the largest stable set in G, whose size is denoted by $\alpha(G)$.

By defining a binary variable $y_j \in \{0, 1\}$ for each $j \in V$ to indicate whether node j belongs to a maximum stable set, the maximum stable set problem can be formulated as an integer linear programming problem as follows:

$$\text{(ILP)} \quad \alpha(G) = \max \left\{ \sum_{j=1}^{n} y_j : y_i + y_j \leq 1, \quad (i, j) \in E, \quad y_j \in \{0, 1\}, \quad j = 1, \ldots, n \right\}.$$

Motzkin and Straus [27] proposed the following formulation in the form of (StQP):

$$\text{(MS-StQP)} \quad \frac{1}{\alpha(G)} = \min_{x \in \Delta_n} x^T (I + A_G) x,$$

where $A_G \in \mathcal{S}^n$ denotes the adjacency matrix of G. Furthermore, if $S^* \subseteq V$ denotes a maximum stable set of G, then an optimal solution of (MS-StQP) is given by $x_j = 1/|S^*|$ if $j \in S^*$, and $x_j = 0$ otherwise.

While we think that reformulating a combinatorial optimization problem as an instance of (StQP) and then reformulating it as an MILP problem may not necessarily be the best solution approach, we still include this set of instances in our experiments due to the existence of a global optimal solution of (MS-StQP) with a support of size $\alpha(G)$, which can be fairly large for certain classes of graphs. Therefore, these instances can be particularly challenging for global solution approaches for (StQP).

Given an instance of (MS-StQP) corresponding to a graph $G = (V, E)$, it is easy to verify that the convexity graph of $Q = I + A_G$ (see Sect. 2.3) is precisely given by the complement of G. Since the valid inequalities (39) are defined for each edge of the complement of the convexity graph, which coincides with G, it follows that the valid inequalities in our MILP formulations are given by

$$y_i + y_j \leq 1, \quad (i, j) \in E,$$

Table 8 Performance on DIMACS1 (8 instances; $n \in [28, 171]$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	7	12.83	29.36	1	72.80
MILP1-L1-VI	8	6.54	12.15	0	–
MILP2-L1	7	2.49	4.84	1	72.80
MILP2-L1-VI	8	7.46	12.74	0	–
MILP1-L2	7	18.71	42.03	1	9.46
MILP1-L2-VI	8	17.60	32.66	0	–
MILP2-L2	7	2.39	4.92	1	9.46
MILP2-L2-VI	8	6.48	11.48	0	–
QP-IP	7	180.46	470.21	1	100.00
CPLEX QP	0	–	–	8	61.79
BARON	0	–	–	8	83.13
ILP	8	2.02	3.49	0	–
DNN	8	44.59	91.00	–	–

Table 9 Performance on DIMACS2 (22 instances; $n \in [200, 300]$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	7	498.14	1052.98	15	81.91
MILP1-L1-VI	18	389.90	754.65	4	71.60
MILP2-L1	7	195.19	449.00	15	82.38
MILP2-L1-VI	17	417.06	749.69	5	79.65
MILP1-L2	7	265.87	596.93	15	18.29
MILP1-L2-VI	17	583.17	988.58	5	9.40
MILP2-L2	8	586.21	1246.52	14	20.60
MILP2-L2-VI	15	379.44	847.51	7	7.65
QP-IP	5	93.20	179.34	17	99.98
CPLEX QP	1	43.80	0	21	84.68
BARON	1	7.95	0	21	92.79
ILP	21	50.57	85.08	1	14.07
DNN	22	1139.63	1383.93	–	–

which are precisely the same constraints as in (ILP). Therefore, our MILP formulations with valid inequalities can in some sense be viewed as extended formulations of (ILP).

In contrast with BLST instances and ST instances, the lower bound $\ell_2(Q)$ is almost tight on 5 out of 8 instances in DIMACS1 and only on 6 out of 22 instances in DIMACS2. The lower bound $\ell_1(Q)$ is quite loose across all instances. The support size of an optimal solution is in the range of [4,44] and [8,128] in DIMACS1 and DIMACS2, respectively.

We report our computational results on DIMACS1 and DIMACS2 in Tables 8 and 9, respectively. For comparison purposes, we also include the performance of the integer linear programming formulation (ILP) on these instances, denoted by ILP.

Tables 8 and 9 illustrate that these instances are indeed challenging for each of the global solution approaches. We discuss the computational results on DIMACS1 and DIMACS2 separately.

On DIMACS1, as illustrated by Table 8, each variant of our MILP formulations without valid inequalities can solve 7 of the 8 instances to optimality within the time limit. The addition of the valid inequalities not only helps to close the gap on the remaining instance on all variants but also improves the average solution time of (MILP1) regardless of the particular lower bound employed. Similarly, QP-IP can solve 7 out of 8 instances to optimality. Each of CPLEX QP and BARON hits the time limit on each of the 8 instances, with BARON reporting a higher average optimality gap compared to CPLEX QP. The average computational effort for solving the DNN relaxation is considerably larger than that required for each variant of our MILP formulations. Finally, ILP can solve all of these instances to optimality fairly quickly. We conjecture that this favorable outcome can be attributed to CPLEX's capability of identifying the particular structure of this formulation and adding other well-known inequalities and cuts.

In terms of the average solution time of MILP problems, each of our eight MILP variants requires less computational effort than QP-IP, either achieving significantly smaller average solution times or terminating with smaller optimality gaps. While the performances of MILP1-L1-VI and MILP2-L1-VI are similar, MILP2-L2-VI exhibits a considerably better performance than MILP1-L2-VI.

On DIMACS2, which consists of larger instances, Table 9 reveals that each approach, including ILP, hits the time limit on various subsets of instances, illustrating the challenging structure of these instances. ILP achieves the best performance in terms of the number of instances solved to optimality. In particular, ILP is terminated due to the time limit only on one instance out of 22. MILP1-L1-VI ranks in the second place, with the second largest number of instances solved to optimality, followed by MILP2-L1-VI and MILP1-L2-VI, respectively. Once again, QP-IP is outperformed by each of our eight MILP variants in terms of both the number of instances solved to optimality and average optimality gap. Both CPLEX QP and BARON can solve only one instance to optimality and hit the time limit on each of the remaining instances, with CPLEX QP reporting a slightly lower average gap compared to BARON. The average computational effort required for solving the DNN relaxation is quite significant, exceeding the average solution time of each MILP formulation.

We again observe notable improvements due to the addition of valid inequalities for both (MILP1) and (MILP2). Furthermore, it is worth noting that both formulations (MILP1) and (MILP2) significantly benefit in terms of average optimality gaps when used with the better lower bound $\ell_2(Q)$.

The performance profile on DIMACS instances is illustrated in Fig. 5. In the figure, we once again recall that the computation time of the tighter lower bound $\ell_2(Q)$ is added to the solution times of each variant that relies on this bound. As demonstrated by Fig. 5, we observe that each of the eight variants of our MILP formulations outperforms CPLEX QP and BARON, even with the inclusion of the additional effort for the computation of $\ell_2(Q)$. MILP1-L1-VI achieves the best performance ratio, followed closely by MILP2-L1-VI. It is also worth noticing that each of MILP2-L1 and MILP1-L1 outperforms QP-IP on this set. Finally, even with the additional computational effort, we remark that each of MILP1-L2-VI and MILP2-L2-VI exhibits a similar performance to QP-IP, while solving a larger number of instances to optimality. On this set of instances, we can clearly see the performance improvement due to the inclusion of valid inequalities on each variant.

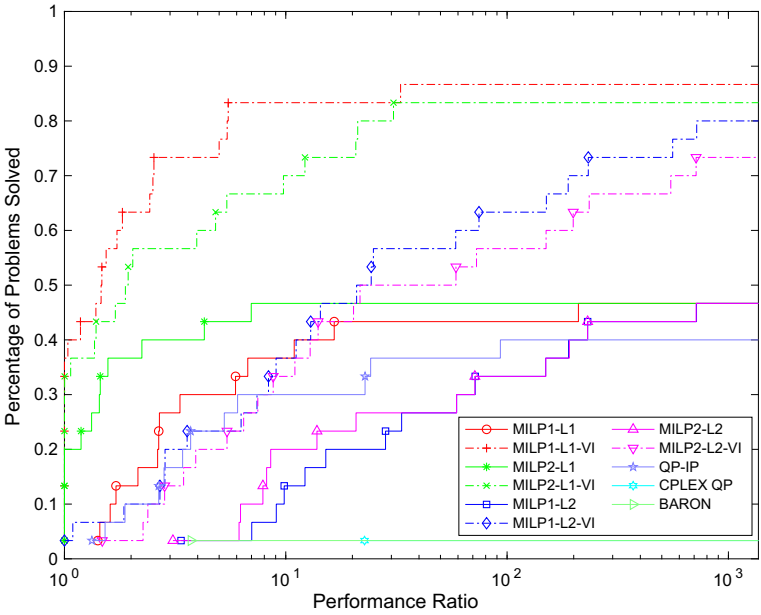

Fig. 5 Performance profile on DIMACS instances

Table 10 Performance on BSU (20 instances; $n \in [5, 24]$)

	Optimal	Average time	Std. Dev.	Time limit	Average gap (%)
MILP1-L1	20	0.34	0.09	0	–
MILP1-L1-VI	20	0.30	0.09	0	–
MILP2-L1	20	0.44	0.61	0	–
MILP2-L1-VI	20	0.37	0.58	0	–
MILP1-L2	20	0.35	0.21	0	–
MILP1-L2-VI	20	0.25	0.11	0	–
MILP2-L2	20	0.35	0.57	0	–
MILP2-L2-VI	20	0.37	0.58	0	–
QP-IP	20	0.60	0.39	0	–
CPLEX QP	5	723.51	1351.99	15	14.82
BARON	5	803.28	1173.36	15	54.53
DNN	20	0.03	0.01	–	–

4.6 BSU instances

In this section, we report our computational results on the BSU data set, which consists of one instance for each value of n in the range $[5, 24]$. Each problem is specifically constructed to have an exponential number of strict local minimizers.

The lower bound $\ell_2(Q)$ is tight only on one instance in this set, namely for $n = 6$. On the other hand, the lower bound $\ell_1(Q)$ is considerably weaker than $\ell_2(Q)$ on all instances. The sizes of the support of an optimal solution range between 2 and 13.

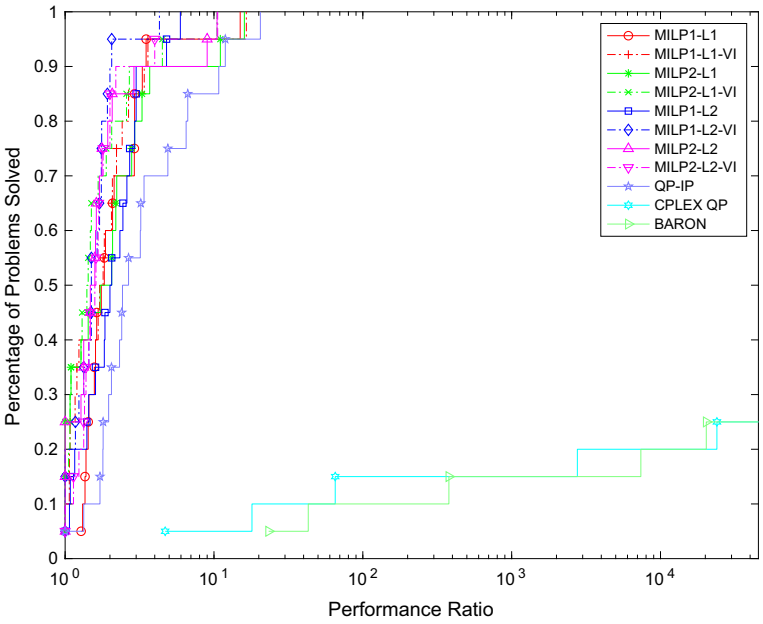

Fig. 6 Performance profile on BSU instances

We report our results in Table 10, which is organized similarly to Table 2. Table 10 reveals that each instance in this set can be solved by each MILP formulation in a fraction of a second on average, illustrating that the existence of an exponential number of strict local minimizers does not seem to hinder the performances of MILP formulations. In particular, each of our eight MILP variants slightly outperforms QP-IP on this set. On the other hand, each of CPLEX QP and BARON can solve only the five smallest instances to optimality within the time limit, each hitting the time limit on the remaining set of 15 instances. Therefore, these instances seem to be particularly challenging for QP and NLP solvers despite the small values of n. Note that DNN requires a negligible average computational effort on this set.

The performance profiles, provided in Fig. 6 for all approaches and in Fig. 7 for MILP based approaches only, support our previous observations. We again recall that the computation of the DNN relaxation is added to the solution time of each variant that relies on the lower bound $\ell_2(Q)$. In particular, as demonstrated by these two figures, it is worth noticing that each of our eight variants outperforms QP-IP, even with the inclusion of the computational effort for $\ell_2(Q)$. In addition, this set of instances provides strong computational evidence that MILP formulations are less likely to be affected by the possibility of an exponential number of local minimizers, which otherwise poses a great challenge for general purpose QP and NLP solvers. These results illustrate that MILP formulations can be much more effective for solving standard quadratic programs in comparison with general purpose QP and NLP solvers.

4.7 Overall comparison

In this section, we give a discussion on overall comparison of each of the global solution approaches on all of the instances in our test bed. In an attempt to make a fair comparison,

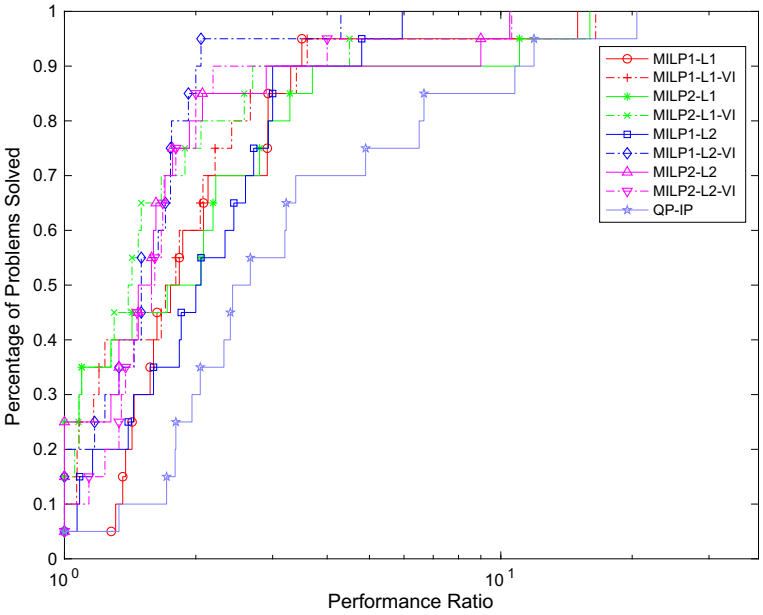

Fig. 7 Performance profile on BSU instances (excluding CPLEX QP and BARON)

we first divide the set of instances into two subsets. IS1 consists of all instances on which each global solution approach is attempted, i.e., the set of all instances excluding the sets ST500 and ST1000. We collect these two sets of larger instances in the set IS2. Recall that the DNN relaxation cannot be solved on IS2.

Our first comparison is based on the summary of the performances of all of the global solution approaches on our test bed.

In Table 11, we summarize the results of our computational experiments. For each instance set and each global solution approach, we report the total number of instances solved to optimality and the total number of instances terminated due to the time limit. We organize the results similarly as in Table 2, except that we have a separate set of columns for each of IS1 and IS2. Furthermore, we do not include the average solution times and average optimality gaps due to the high variability.

We first focus on instances in IS1. Table 11 reveals that MILP1-L1-VI dominates all the other approaches in terms of the total number of instances solved to optimality within the time limit. MILP2-L1-VI ranks in the second place, followed closely by MILP1-L2-VI and MILP2-L2-VI, respectively. Each of our 8 MILP variants outperforms QP-IP on this performance metric. CPLEX QP and BARON fall behind significantly, with CPLEX QP exhibiting better performance than BARON. Note that the addition of valid inequalities improves this performance metric on both formulations regardless of the lower bound employed. On the other hand, this performance metric does not seem to be significantly affected by the choice of the lower bound $\ell_1(Q)$ or $\ell_2(Q)$.

Recall that IS2 consists of 12 larger instances. On this set, MILP2-L1 can solve all of the instances to optimality and outperforms all other approaches in terms of this performance metric, which is followed, in turn, by MILP2-L1-VI, MILP1-L1, and MILP1-LI-VI, respectively. Once again, each of our MILP variants exhibits better performance than QP-IP on this

Table 11 Summary of results (IS1: 364 instances; $n \in [5, 300]$; IS2: 12 instances; $n \in [500, 1000]$)

	IS1		IS2	
	Optimal	Time limit	Optimal	Time limit
MILP1-L1	348	16	11	1
MILP1-L1-VI	360	4	11	1
MILP2-L1	348	16	12	0
MILP2-L1-VI	359	5	11	1
MILP1-L2	348	16	–	–
MILP1-L2-VI	358	5	–	–
MILP2-L2	349	15	–	–
MILP2-L2-VI	357	7	–	–
QP-IP	346	18	6	6
CPLEX QP	260	104	0	12
BARON	121	243	0	12

metric. Each of CPLEX QP and BARON hits the time limit on each instance in this set. Note that valid inequalities and the choice of the lower bound do not seem to affect this metric on this set.

In an attempt to shed more light on the comparison of the behavior of different approaches, we now focus on the results reported in Tables 2, 3, 4, 5, 6, 7, 8, 9, and 10 and Figs. 1, 2, 3, 4, 5, 6, and 7. These results clearly illustrate that MILP formulations constitute an effective approach for solving standard quadratic programs in comparison with state-of-the-art QP and NLP solvers. In terms of average solution times, we observe that each of the eight variants of our MILP formulations outperforms QP-IP. On the other hand, when we take into account the additional computational effort for the computation of the tighter lower bound $\ell_2(Q)$, we see that this effort outweighs the benefits of our MILP formulations on larger instances. It is worth noticing that each of the four variants that relies on the simple lower bound $\ell_1(Q)$ consistently outperforms QP-IP on all instances, demonstrating the robustness of these variants. The valid inequalities do not seem to provide a notable computational advantage in general and may even worsen the performance, with the exception of maximum stable set instances. We recommend using the variants of our formulations that rely on the tighter lower bound $\ell_2(Q)$ unless this bound is too costly to compute (e.g., for $n \leq 50$). For larger instances, we suggest using (MILP2) with the simple lower bound $\ell_1(Q)$ due its more robust performance. Finally, we do not recommend the addition of valid inequalities, unless the instance is known to arise from a maximum stable set problem.

5 Concluding remarks

In this paper, we propose solving standard quadratic programs by using two alternative MILP reformulations. The first MILP formulation arises from a simple manipulation of the KKT conditions. The second MILP formulation is obtained by exploiting the specific structure of a standard quadratic program and is in fact a relaxation of the first formulation. Both of our MILP formulations involve big-M parameters. We derive bounds on these parameters by taking advantage of the particular structure of (StQP). We show that these bounds are functions of a lower bound on the optimal value. We consider two different lower bounds, which differ in terms of the computational effort and tightness. Our extensive computational

experiments illustrate that the proposed MILP formulations outperform another recently proposed MILP approach in [31] in terms of average solution times and are much more effective than general purpose quadratic programming and nonlinear programming solvers.

Since the tighter bound $\ell_2(Q)$ requires a considerable computational effort for large instances, one can use cheaper methods for computing or approximating this bound. For instance, the dual problem of (DNN) can be solved using a combination of proximal gradient method and binary search (see, e.g., [23]), which may be cheaper than using a semidefinite programming solver. Alternatively, based on the encouraging computational results reported in [25], a tight linear programming based lower bound can be employed in lieu of $\ell_2(Q)$. We leave these problems for future work.

Another interesting research direction is the investigation of decomposition and cutting plane approaches for the proposed MILP formulations arising from large-scale standard quadratic programs. In addition, encouraged by our computational results, we intend to identify other classes of nonconvex quadratic programs that would be amenable to a similar effective MILP formulation.

Acknowledgements We are grateful to two anonymous referees for their insightful comments and suggestions.

References

1. Anstreicher, K.M., Burer, S.: DC versus copositive bounds for standard QP. J. Global Optim. **33**(2), 299–312 (2005)
2. Belotti, P., Lee, J., Liberti, L., Margot, F., Wächter, A.: Branching and bounds tightening techniques for non-convex MINLP. Optim. Methods Softw. **24**(4–5), 597–634 (2009)
3. Bomze, I.M.: On standard quadratic optimization problems. J. Global Optim. **13**(4), 369–387 (1998)
4. Bomze, I.M.: Branch-and-bound approaches to standard quadratic optimization problems. J. Global Optim. **22**(1–4), 17–37 (2002)
5. Bomze, I.M.: Regularity versus degeneracy in dynamics, games, and optimization: A unified approach to different aspects. SIAM Rev. **44**(3), 394–414 (2002)
6. Bomze, I.M., De Klerk, E.: Solving standard quadratic optimization problems via linear, semidefinite and copositive programming. J. Global Optim. **24**(2), 163–185 (2002)
7. Bomze, I.M., Dür, M., De Klerk, E., Roos, C., Quist, A.J., Terlaky, T.: On copositive programming and standard quadratic optimization problems. J. Global Optim. **18**(4), 301–320 (2000)
8. Bomze, I.M., Locatelli, M., Tardella, F.: New and old bounds for standard quadratic optimization: dominance, equivalence and incomparability. Math. Program. **115**(1), 31 (2008)
9. Bomze, I.M., Schachinger, W., Ullrich, R.: The complexity of simple models - A study of worst and typical hard cases for the standard quadratic optimization problem. Math. Oper. Res. **43**(2), 651–674 (2018)
10. Bonami, P., Lodi, A., Schweiger, J., Tramontani, A.: Solving standard quadratic programming by cutting planes. SIAM J. Optim. **29**(2), 1076–1105 (2019)
11. Bundfuss, S., Dür, M.: An adaptive linear approximation algorithm for copositive programs. SIAM J. Optim. **20**(1), 30–53 (2009)
12. Burer, S., Vandenbussche, D.: A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations. Math. Program. **113**(2), 259–282 (2008)

13. Chen, J., Burer, S.: Globally solving nonconvex quadratic programming problems via completely positive programming. Math. Program. Comput. **4**(1), 33–52 (2012)
14. Dickinson, P.J.C., Gijben, L.: On the computational complexity of membership problems for the completely positive cone and its dual. Comput. Optim. Appl. **57**(2), 403–415 (2014)
15. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
16. Giannessi, F., Tomasin, E.: Nonconvex quadratic programs, linear complementarity problems, and integer linear programs. In: IFIP Technical Conference on Optimization Techniques, pp. 437–449. Springer (1973)
17. Gibbons, L.E., Hearn, D.W., Pardalos, P.M., Ramana, M.V.: Continuous characterizations of the maximum clique problem. Math. Oper. Res. **22**(3), 754–768 (1997)
18. Gurobi Optimization, LLC: Gurobi optimizer reference manual (2018). http://www.gurobi.com
19. Hu, J., Mitchell, J., Pang, J., Bennett, K., Kunapuli, G.: On the global solution of linear programs with linear complementarity constraints. SIAM J. Optim. **19**(1), 445–471 (2008)
20. Hu, J., Mitchell, J.E., Pang, J.S.: An LPCC approach to nonconvex quadratic programs. Math. Program. **133**(1), 243–277 (2012)
21. Ibaraki, T., Katoh, N.: Resource Allocation Problems: Algorithmic Approaches. MIT Press, Cambridge, MA, USA (1988)
22. IBM: IBM ILOG CPLEX Optimization Studio (2018). https://www.ibm.com/products/ilog-cplex-optimization-studio
23. Kim, S., Kojima, M., Toh, K.C.: A Lagrangian-DNN relaxation: a fast method for computing tight lower bounds for a class of quadratic optimization problems. Math. Program. **156**(1), 161–187 (2016)
24. Kingman, J.F.C.: A mathematical problem in population genetics. Math. Proc. Cambridge Philos. Soc. **57**(3), 574–582 (1961)
25. Liuzzi, G., Locatelli, M., Piccialli, V.: A new branch-and-bound algorithm for standard quadratic programming problems. Optim. Methods Softw. **34**(1), 79–97 (2019)
26. Markowitz, H.: Portfolio selection. The J. Finance **7**(1), 77–91 (1952)
27. Motzkin, T.S., Straus, E.G.: Maxima for graphs and a new proof of a theorem of Turán. Canad. J. Math **17**(4), 533–540 (1965)
28. Nowak, I.: A new semidefinite programming bound for indefinite quadratic forms over a simplex. J. Global Optim. **14**(4), 357–364 (1999)
29. Scozzari, A., Tardella, F.: A clique algorithm for standard quadratic programming. Discret. Appl. Math. **156**(13), 2439–2448 (2008)
30. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**, 225–249 (2005)
31. Xia, W., Vera, J., Zuluaga, L.F.: Globally solving nonconvex quadratic programs via linear integer programming techniques. Informs J. Comput. **32**(1), 40–56 (2020)
32. Yu, B., Mitchell, J.E., Pang, J.S.: Solving linear programs with complementarity constraints using branch-and-cut. Math. Program. Comput. **11**(2), 267–310 (2019)