



Enhancing the normalized multiparametric disaggregation technique for mixed-integer quadratic programming

Tiago Andrade¹ · Fabricio Oliveira² · Silvio Hamacher¹ · Andrew Eberhard³

Received: 11 January 2018 / Accepted: 3 November 2018 / Published online: 17 December 2018
© The Author(s) 2018

Abstract

We propose methods for improving the relaxations obtained by the normalized multiparametric disaggregation technique (NMDT). These relaxations constitute a key component for some methods for solving nonconvex mixed-integer quadratically constrained quadratic programming (MIQCQP) problems. It is shown that these relaxations can be more efficiently formulated by significantly reducing the number of auxiliary variables (in particular, binary variables) and constraints. Moreover, a novel algorithm for solving MIQCQP problems is proposed. It can be applied using either its original NMDT or the proposed reformulation. Computational experiments are performed using both benchmark instances from the literature and randomly generated instances. The numerical results suggest that the proposed techniques can improve the quality of the relaxations.

Keywords Normalized multiparametric disaggregation technique · Nonconvex mixed-integer quadratically constrained quadratic programs · McCormick envelopes · Convex relaxation

This work was partially supported by the Brazilian National Council for Scientific and Technological Development (CNPq) and the Coordination for the Improvement of Higher Education Personnel, Brazil (CAPES) under grants 03863/2016-3 and 306802/2015-5. The research of the last author was supported by the Australian Research Council, Discovery Grant DP140100985.

✉ Andrew Eberhard
andy.eberhard@rmit.edu.au

Tiago Andrade
tiago.cout.andrade@gmail.com

Fabricio Oliveira
fabricio.oliveira@aalto.fi

Silvio Hamacher
hamacher@puc-rio.br

¹ Pontifícia Universidade Católica do Rio de Janeiro (Puc-Rio) R. Marquês de São Vicente, 225 - Gávea, Rio de Janeiro, RJ 22451-000, Brazil

² Systems Analysis Laboratory, Department of Mathematics and Systems Analysis, School of Science, Aalto University, PO Box 11100, 00076 AALTO, Finland

³ RMIT University, GPO Box 2476, Melbourne, VIC 3001, Australia

1 Introduction

In this study, the following general nonconvex (mixed-integer) quadratically constrained quadratic programming ((MI)QCQP) problems with box constraints are considered.

$$\min x^T Q_0 x + f_0(x, y) \quad (1)$$

s.t.:

$$x^T Q_r x + f_r(x, y) \leq 0, \quad \forall r \in I_{1,m} \quad (2)$$

$$x_i \in [X_i^L, X_i^U], \quad \forall i \in I_{1,n_1} \quad (3)$$

$$y_i \in \{Y_i^L, \dots, Y_i^U\}, \quad \forall i \in I_{1,n_2}, \quad (4)$$

where $I_{a,b} = \{a, \dots, b\}$ is the subset of integers between a and b (inclusive), Q_r , for all $r \in I_{0,m}$, is a symmetric matrix, $f_0 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \rightarrow \mathbb{R}$ is a linear function, and $f_r : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \rightarrow \mathbb{R}$, for all $r \in I_{1,m}$, are affine functions. The variable x can assume any value between its bounds X^L and X^U , and y can assume any integer value between Y^L and Y^U . One implicit assumption in formulation (1)–(4) is that all variables that appear in product terms are continuous, as a product term containing at least one integer variable can be trivially linearized. If $n_2 = 0$, the problem is reduced to a nonconvex QCQP problem.

(MI)QCQP can naturally be used for modeling various important processes in areas such as heat integration networks, separation systems, reactor networks, batch processes, pooling problems, and refinery operation planning problems [2,5,23,32,46]. It is a general method with important subclasses including (nonconvex) quadratic programming (QP), mixed-integer programming (MIP), (nonconvex) quadratically constrained quadratic programming (QCQP), and linear programming (LP).

An (MI)QCQP problem is called convex if its continuous relaxation is convex, regardless of the nonconvexity introduced by the integrality constraints of the decision variables. This problem is convex if Q_r is positive semi-definite for all $r \in I_{0,m}$, and nonconvex otherwise. In this study, the latter case is considered, i.e., when Q_r is not positive semidefinite.

The (MI)QCQP problem with box constraints is known to be NP-hard [35], even without quadratic constraints. It should be noted that if the box constraints are removed, the problem is undecidable [22]. A detailed definition and implications of undecidability and NP-hardness may be found in [18].

It is known that MIQCQP problems are equivalent to QCQP problems, as any integer variable can be defined as a sum of binary variables, and the constraint $y = y^2$ can be added to represent the integrality condition $y \in \{0, 1\}$. Although this transformation is possible, it is usually undesirable because it generally results in more computationally difficult nonconvex problems.

As (MI)QCQP problems and their variants are difficult to solve, many alternative solution methods have been proposed. They can be classified into exact methods, such as spatial Branch-and-Bound (BnB), and heuristic methods. The former can ensure that a globally optimal solution will be achieved, whereas the latter can only ensure local optimality of the solutions. Furthermore, nearly all methods involve relaxation techniques.

A commonly used exact algorithm for solving MIQCQP problems is BnB and its variants. If the problem is convex, BnB obtains bounds and eventually globally optimal solutions by relaxations of the integrality constraints. If the problem is nonconvex, spatial BnB is typically used, and the nonconvexity that arises from nonlinearity must also be relaxed via convex relaxations. In spatial BnB, both integer and continuous variables are usually branched by partitioning the feasible region into hyper-rectangles within the search space. Other forms

of branching using different types of polyhedra have also been employed [27]. In general, it is recommended that a domain reduction step be performed to accelerate BnB [19,43]. A survey on BnB applied to nonconvex problems may be found in [40].

The relaxations for MIQCQP problems can be classified into five categories. The first category consists of linear relaxations whereby the problem is relaxed to a mixed-integer programming (MIP) problem without auxiliary integer variables. A classic approach of this type relies on McCormick envelopes [31], where bounded auxiliary variables representing the product of two variables are added to the problem. If the variables that appear in the product assume their bounds, the auxiliary variable will assume the value of the product; therefore, the relaxation will then be exact. Al-Khayyal and Falk [1] showed that McCormick envelopes represent the convex and concave envelopes of the function $f(x_1, x_2) = x_1x_2$ defined in a rectangle, that is we assume, $f : [x_1^L, x_1^U] \times [x_2^L, x_2^U] \rightarrow \mathbb{R}$. Bao et al. [7] proposed a tighter relaxation, namely a polyhedral multiterm relaxation obtained by determining the convex envelope of the sum of the quadratic terms. Sherali and Adams [38] proposed the reformulation-linearization technique (RLT), which is a systematic approach for generating valid constraints to a problem using linear equations and inequalities including the bounding constraints, and thus strengthening its linear relaxation. In particular, McCormick envelopes can be derived using RLT.

The second category uses convex relaxations. The resulting relaxed problem is still nonlinear, possibly with integer variables, but its continuous relaxation is convex. This can be achieved by adding convex terms with sufficiently large coefficients [3] or by decomposing the quadratic matrices into a sum of positive and negative matrices and then linearizing the second term only. This method is known as the difference of convex functions (DC) approach. Fampa et al. [16] proposed several approaches for decomposing the quadratic functions as sums of a convex and a concave function. Another possible strategy is to determine envelopes for the quadratic terms over regions other than rectangles [27].

The third category uses Lagrangian relaxations and Lagrangian bounds [33,44,45]. Although the relaxed problem is convex, the resulting Lagrangian relaxation subproblem is usually nonconvex and as difficult to solve as the original problem. To obtain a tighter relaxation with convex subproblems, augmented Lagrangian relaxation [9,36] can be used, at the expense of introducing nonlinear terms. Moreover, both traditional Lagrangian relaxation and the augmented version usually result in nonsmooth problems.

The fourth category is based on conic programming, which can be considered a generalization of linear programming. The most common approach is semidefinite programming (SDP) [20,29,30]. Another common approach is second-order conic programming (SOC). Sherali and Fraticelli [39] proposed a cutting plane generation method using SDP. Anstreicher [4] compared SDP and RLT relaxations and proposed an integrated approach for using SDP to tighten RLT relaxations. Linderoth [27] showed that the relaxation of quadratic terms over triangular regions of the form $\{(x_1, x_2, w) | w = x_1x_2, x_1^L \leq x_1 \leq x_1^U, x_2^L \leq x_2 \leq x_2^U, x_1 + x_2 \leq c\}$ can be formulated using SOC. More recently, Bomze et al. [11] and Bomze [10] proposed copositive programming (a subclass of conic programming) for solving quadratic problems. Although it provides a tighter relaxation, the resulting conic programming is nonconvex.

The fifth category is based on partitioning the solution space and relaxing each partition independently. The partitions can be obtained by adding binary variables or using disjunctive programming [6]. The most traditional approach is based on piecewise McCormick envelopes [12,21,24,47]. A recent alternative method, which also relies on convex envelopes, is the normalized multiparametric disaggregation technique (NMDT) proposed by Castro [13]. NMDT and its variations will be reviewed in more detail in Sect. 2.

A relaxation method of the fifth class was chosen in this study because this class can generate arbitrarily tight relaxations without using spatial BnB. Moreover, these relaxations yield problems that can be solved using off-the-shelf MIP solvers, such as CPLEX [14], GUROBI [34], and XPRESS [17], which are known to be reliable and efficient.

In this study, we provide a formal proof that NMDT can be used for generating arbitrarily tight relaxations for (MI)QCQP problems. Moreover, we investigated whether NMDT can be improved using a more efficient reformulation that requires fewer additional binary and continuous variables and constraints as compared to the formulation in [13]. Furthermore, we propose an improvement to the traditional algorithm for solving the (MI)QCQP problems using this relaxation. The new algorithm requires fewer binary variables to be added per iteration by choosing the variables that have more potential to impact on the quality of the relaxation via the refinement of the underlying discretization. To assess the effectiveness of these improvements a comparison is made with the corresponding implementation of the original algorithm proposed by Castro [13] with that proposed in this paper. We compare these two implementations using the instances available in the literature and also using a novel class of randomly generated instances.

The paper is organized as follows. In Sect. 2, a review of NMDT and its variations is provided. In Sect. 3, it is shown that NMDT can be reformulated so that the number of required additional variables and constraints may be reduced. Furthermore, numerical experiments are presented in Sect. 4, where the proposed methods are assessed considering both instances in the literature and randomly generated instances. Finally, in Sect. 5, conclusions are drawn and future research directions are discussed.

2 Normalized multiparametric disaggregation

In this section the mathematical background of NMDT is reviewed, and the related notation is introduced. Moreover, an initial formulation of NMDT is presented that will be central to the subject matter of this study.

NMDT appears as a natural progression of relaxations that have recently been used for solving either (MI)QCQP problems or certain subclasses of these problems such as bilinear programming problems. The ideas that led to the development of NMDT are reviewed below.

Li and Chang [26] proposed an approximation to the quadratic problem using a binary expansion of all variables. Based on this idea, Teles et al. [42] proposed the multiparametric disaggregation technique (MDT) as an approximation to polynomial programming.

Kolodziej et al. [25] proposed a relaxation for QCQP problems based on MDT by performing a decimal expansion on a subset of the variables and by including additional continuous variables with arbitrarily tight bounds. The products of binary variables and continuous variables were linearized exactly, and the products of two continuous variables were relaxed using McCormick envelopes. Additionally, they showed that their formulation can be obtained using disjunctive programming.

Later, Castro [13] proposed the normalized multiparametric disaggregation technique (NMDT) and showed that it is advantageous to normalize the variables before performing the decimal expansion, as the number of partitions for all variables is more controllable.

In the remainder of this section, the formulation of NMDT is presented. Given a (MI)QCQP problem, let $QT = \{(i, j) \in I_{1,n_1}^2 \mid j \geq i, \exists r \in I_{0,m}, |Q_{r,i,j}| > 0\}$ and $DS = \{j \in I_{1,n_1} \mid \exists i \in I_{1,n_1}, (i, j) \in QT\}$. The set QT consists of the indices of variables that appear in at least one quadratic term, whereas DS corresponds to the set of variables that will be discretized.

The variables x_j for all $j \in DS$ are normalized as follows:

$$x_j = (X_j^U - X_j^L)\lambda_j + X_j^L, \quad \forall j \in DS. \tag{5}$$

$\lambda_j \in [0, 1]$ is discretized in partitions of size 10^p each, where p corresponds to a precision factor. The variables $\Delta\lambda_j$ are added to allow λ_j to attain all values in the interval $[0, 1]$. Thus,

$$\lambda_j = \sum_{k \in I_{0,9}, l \in I_{p,-1}} k10^l z_{j,k,l} + \Delta\lambda_j, \quad \forall j \in DS \tag{6}$$

$$0 \leq \Delta\lambda_j \leq 10^p, \quad \forall j \in DS. \tag{7}$$

The following relations are obtained by multiplying both sides of (5) and (6) by x_i for all $i \in I_{1,n}$.

$$x_i x_j = (X_j^U - X_j^L)x_i \lambda_j + x_i X_j^L, \quad \forall i, j \in QT \tag{8}$$

$$x_i \lambda_j = \sum_{k \in I_{0,9}, l \in I_{p,-1}} k10^l x_i z_{j,k,l} + x_i \Delta\lambda_j, \quad \forall i, j \in QT. \tag{9}$$

Subsequently, the auxiliary variables $w_{i,j}$, $\hat{x}_{i,j,k,l}$, $v_{i,j}$, and $\Delta v_{i,j}$ are included to represent the products $x_i x_j$, $x_i z_{j,k,l}$, $x_i \lambda_j$, and $x_i \Delta\lambda_j$, respectively. Using these auxiliary variables, we obtain

$$w_{i,j} = (X_j^U - X_j^L)v_{i,j} + x_i X_j^L, \quad \forall i, j \in QT \tag{10}$$

$$v_{i,j} = \sum_{k \in I_{0,9}, l \in I_{p,-1}} k10^l \hat{x}_{i,j,k,l} + \Delta v_{i,j}, \quad \forall i, j \in QT. \tag{11}$$

Constraints (12) and (13) are known as the McCormick envelopes and provide a relaxation of the product of two continuous variables. The product of binary and continuous variables is exactly linearized by constraints (14)–(16).

$$X_i^L \Delta\lambda_j \leq \Delta v_{i,j} \leq X_i^U \Delta\lambda_j, \quad \forall i, j \in QT \tag{12}$$

$$10^p(x_i - X_i^U) + X_i^U \Delta\lambda_j \leq \Delta v_{i,j} \leq 10^p(x_i - X_i^L) + X_i^L \Delta\lambda_j, \quad \forall i, j \in QT \tag{13}$$

$$\sum_{k \in I_{0,9}} z_{j,k,l} = 1, \quad \forall j \in DS, l \in I_{p,-1} \tag{14}$$

$$\sum_{k \in I_{0,9}} \hat{x}_{i,j,k,l} = x_i, \quad \forall i, j \in QT \tag{15}$$

$$X_i^L z_{j,k,l} \leq \hat{x}_{i,j,k,l} \leq X_i^U z_{j,k,l}, \quad \forall i, j, k, l. \tag{16}$$

Furthermore, using the variable $w_{i,j}$, the objective function (1) and the original constraints (2) are replaced by Eqs. (17) and (18), respectively.

$$\min \sum_{i|(i,i) \in QT} Q_{0,i,i} w_{i,i} + 2 \sum_{(i,j) \in QT | j > i} Q_{0,i,j} w_{i,j} + f_0(x, y) \tag{17}$$

$$\sum_{i|(i,i) \in QT} Q_{r,i,i} w_{i,i} + 2 \sum_{(i,j) \in QT | j > i} Q_{r,i,j} w_{i,j} + f_r(x, y) \leq 0, \quad \forall r \in I_{1,m}. \tag{18}$$

We need to define one additional constraint that will serve the purpose of simplifying the technical results stated later on. Constraint (19) represents an alternative nonlinear definition of the variable Δv .

$$\Delta v_{i,j} = x_i \Delta\lambda_j \tag{19}$$

Definition 1 For every p , $EQUIV_p$ is defined as the problem of minimizing the objective function (17), subject to the constraints (5)–(7), (10), (11), (14)–(16), (18), and (19).

Definition 2 For every p , the set $FS-EQUIV_p$ is defined as the feasible set of problem $EQUIV_p$. That is, $(x, y, w, z, v, \hat{x}, \lambda, \Delta\lambda, \Delta v) \in FS-EQUIV_p$ if and only if it satisfies constraints (5)–(7), (10), (11), (14)–(16), (18), and (19).

Lemma 1 For all $p \leq 0$, $EQUIV_p$ is equivalent to the original problem (1)–(4).

Lemma 1 is trivial, as all additional constraints (and associated variables) are redundant and the linearizations are exact. Problem $EQUIV_p$ is useful as an intermediate step in proving that $NMDT_p$ is a relaxation of the original (MI)QCQP problem.

Definition 3 For every p , $NMDT_p$ is defined as the problem of minimizing the objective function (17) subject to the constraints (5)–(7), (10)–(16), and (18).

Definition 4 For every p , the set $FS-NMDT_p$ is defined as the feasible set of problem $NMDT_p$. That is, $(x, y, w, z, v, \hat{x}, \lambda, \Delta\lambda, \Delta v) \in FS-NMDT_p$ if and only if it satisfies constraints (5)–(7), (10)–(16), and (18).

Proposition 1 $NMDT_p$ is a relaxation of $EQUIV_p$ for every $p \leq 0$.

Proof Both problems have the same objective function. Thus, $NMDT_p$ will be a relaxation of $EQUIV_p$ if $FS-NMDT_p \supseteq FS-EQUIV_p$. The constraints that are used to define both feasible sets are nearly the same. The only difference is that $FS-NMDT_p$ has constraints (12) and (13) instead of (19). As the former are the McCormick envelopes of the product that appears in the latter, it follows that constraints (12) and (13) are implied from constraint (19), whereas the converse is not true. Therefore, $FS-NMDT_p \supseteq FS-EQUIV_p$, and the proposition follows. \square

Proposition 2 $NMDT_p$ is a relaxation of the original (MI)QCQP problem for every $p \leq 0$.

Proof The result follows directly from Lemma 1 and Proposition 1. \square

Theorem 1 For any pair (p_1, p_2) with $p_1 < p_2 \leq 0$, $NMDT_{p_2}$ is a relaxation of $NMDT_{p_1}$.

Proof It should be noted that $NMDT_{p_1}$ has more variables than $NMDT_{p_2}$. Thus, the feasible sets $FS-NMDT_{p_1}$ and $FS-NMDT_{p_2}$ cannot be compared directly, as they have different dimensions. To allow such a comparison, a mapping $M : FS-NMDT_{p_1} \rightarrow FS-NMDT_{p_2}$ is constructed so that every element $(x, y, w, z, v, \hat{x}, \lambda, \Delta\lambda, \Delta v) \in FS-NMDT_{p_1}$ evaluated in the objective function of $NMDT_{p_1}$ is equal to $M(x, y, w, z, v, \hat{x}, \lambda, \Delta\lambda, \Delta v)$ evaluated in the objective function of $NMDT_{p_2}$. Let M be defined as

$$\begin{aligned} x_i^{NMDT_{p_2}} &= x_i^{NMDT_{p_1}}, & \forall i \in I_{1,n_1} \\ y_i^{NMDT_{p_2}} &= y_i^{NMDT_{p_1}}, & \forall i \in I_{1,n_2} \\ w_i^{NMDT_{p_2}} &= w_i^{NMDT_{p_1}}, & \forall i, j \in QT \\ z_{j,k,l}^{NMDT_{p_2}} &= z_{j,k,l}^{NMDT_{p_1}}, & \forall j \in DS, k \in I_{0,9}, l \in I_{p_2,0} \\ v_{i,j}^{NMDT_{p_2}} &= v_{i,j}^{NMDT_{p_1}}, & \forall i, j \in QT \\ \hat{x}_{i,j,l}^{NMDT_{p_2}} &= \hat{x}_{i,j,l}^{NMDT_{p_1}}, & \forall i, j \in QT, k \in I_{0,9}, l \in I_{p_2,0} \end{aligned}$$

$$\begin{aligned} \lambda_j^{NMDT_{p_2}} &= \lambda_j^{NMDT_{p_1}}, \quad \forall j \in DS \\ \Delta\lambda_j^{NMDT_{p_2}} &= \Delta\lambda_j^{NMDT_{p_1}} + \sum_{l \in I_{p_1, p_2-1}} k2^l z_{j,k,l}^{NMDT_{p_1}}, \quad \forall j \in DS \\ \Delta v_{i,j}^{NMDT_{p_2}} &= \Delta v_{i,j}^{NMDT_{p_1}} + \sum_{l \in I_{p_1, p_2-1}} k2^l \hat{x}_{i,j,l}^{NMDT_{p_1}}, \quad \forall i, j \in QT \end{aligned}$$

It is straightforward to verify that the image of this mapping is in the feasibility set $FS-NMDT_{p_2}$, completing the proof. \square

Theorem 2 For any pair (p_1, p_2) with $p_1 < p_2 \leq 0$, $NMDT_{p_1}$ is a tighter (or equal) relaxation of the original (MI)QCQP problem than $NMDT_{p_2}$.

Proof By Proposition 2, both problems are relaxations of the original problem. By Theorem 1, $NMDT_{p_2}$ is a relaxation of $NMDT_{p_1}$, it follows that $NMDT_{p_1}$ is a tighter relaxation of the original problem than $NMDT_{p_2}$. \square

2.1 Algorithm

Algorithm 1 was originally proposed in Castro [13] for solving an (MI)QCQP problem using NMDT. It is similar to the algorithm in Kolodziej et al. [25].

Algorithm 1 Algorithm to NMDT

- Step 0. Choose $p = 0$ and let $UB = +\infty$ and $iteration = 0$.
 - Step 1. $iteration = iteration + 1$.
 - Step 2. Solve relaxation $NMDT_p$, obtaining LB and (x^R, y^R) .
 - Step 3. Solve original problem with a local solver with initial solution (x^R, y^R) and fix integer variables at y^R . If a new best solution is found, store the incumbent solution (x^*, y^*) and update UB .
 - Step 4. If one of the stopping criteria (discussed below) is met, stop. Otherwise, set $p = p - 1$ and return to Step 1.
-

The principle of this algorithm is to tighten the relaxation as the iterations progress by decreasing the parameter p , thus gradually increasing the lower bound (LB). Feasible solutions are obtained using local methods with warm starts and fixing integer variables, which in turn provides upper bounds (UB). An incumbent solution is the best feasible solution obtained during the execution of the algorithm. Common stop criteria are the maximum number of iterations, the maximum time elapsed, and the relative or absolute gap with respect to a certain threshold.

If the feasible space of the original (MI)QCQP problem is not empty, the algorithm converges to the optimal value of the (MI)QCQP problem since $\lim_{p \rightarrow -\infty} \Delta\lambda_j = 0$ and, if $\Delta\lambda_j = 0$, then $w_{i,j} = x_i x_j$. Thus, the relaxed solution is feasible for the original problem and $UB = LB$.

Although convergence is only asymptotically guaranteed, it is often observed (as will also be seen in the computational experiments presented later) that feasible solutions are obtained within a few iterations of the algorithm. Furthermore, the lower bound in each iteration is at least as good as the bound in the previous iteration, as stated in the following theorem.

Theorem 3 The sequence of lower bounds generated by Algorithm 1 is monotonic.

Proof As in each iteration, the value of p is decreased, the result follows from Theorem 2. \square

3 Reformulated normalized multiparametric disaggregation

Herein, the main contributions of this study are presented. It is first shown that a binary expansion is preferable to a decimal expansion in NMDT. Subsequently, a reformulation of the problem is presented in which the number of variables (both binary and continuous) and constraints are reduced. Finally, an alternative algorithm is developed.

3.1 Reformulation using binary expansion

The first reformulation consists in changing from 10 to 2 the numerical base that is used for representing the continuous variables. It should be noted that this idea is not new and has already been successfully applied to other techniques related to RNMDT [41]. Nevertheless, Castro [13] used decimal representation for the NMDT. Despite a brief mention in that other bases may be chosen, to the best of our knowledge [13], base-2 (or binary) expansions have not been applied in this context. Other key difference between the base-2 expansion used in Teles et al. [41] and the proposed approach is that, while the former uses base-2 for the MDT only as a means to reduce the total of auxiliary variables while approximately maintaining the same precision level, our focus is to use the base-2 formulation to control how the model grows as the precision level increase between iterations.

The formulation using a binary expansion (i.e., a representation in which each variable is replaced by a base-2 expansion) instead of the traditional decimal expansion is obtained by modifying constraints (6), (7), (11), (13), (14), and (15). The new constraints are obtained by replacing the number 10 by 2 and 9 by 1, respectively, wherever they appear in these constraints. This procedure results in the new constraints (20)–(25).

$$\lambda_j = \sum_{k \in I_{0,1}, l \in I_{p,-1}} 2^l k z_{j,k,l} + \Delta \lambda_j, \quad \forall j \in DS \quad (20)$$

$$0 \leq \Delta \lambda_j \leq 2^p, \quad \forall j \in DS \quad (21)$$

$$v_{i,j} = \sum_{k \in I_{0,1}, l \in I_{p,-1}} 2^l k \hat{x}_{i,j,k,l} + \Delta v_{i,j}, \quad \forall i, j \in QT \quad (22)$$

$$2^p(x_i - X_i^U) + X_i^U \Delta \lambda_j \leq \Delta v_{i,j} \leq 2^p(x_i - X_i^L) + X_i^L \Delta \lambda_j, \quad \forall i, j \in QT \quad (23)$$

$$\sum_{k \in I_{0,1}} z_{j,k,l} = 1, \quad \forall j \in DS, l \in I_{p,-1} \quad (24)$$

$$\sum_{k \in I_{0,1}} \hat{x}_{i,j,k,l} = x_i, \quad \forall i, j \in QT \quad (25)$$

Despite its simplicity, this reformulation allows a significant reduction in the number of auxiliary binary variables required in the variable expansion for a given precision 10^p . The following propositions allow the comparison of the total number of binary variables required in the base-10 and the base-2 expansions.

Proposition 3 *The number of auxiliary binary variables z for NMDT in base 10 is $10(-p)|DS|$ for a given value of the parameter $p < 0$, where $|DS|$ is the cardinality of the set QT , i.e., the number of quadratic terms in the original (MI)QCQP problem.*

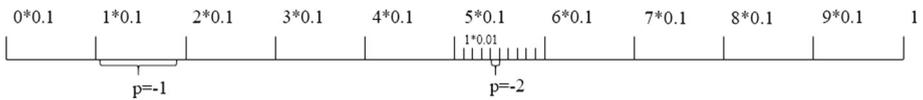


Fig. 1 Discretization using decimal expansion

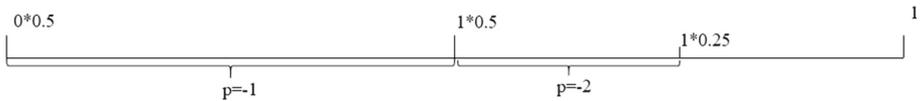


Fig. 2 Discretization using binary expansion

Proof For every $j \in DS$, $\sum_{k \in I_{0,9}} \sum_{l \in I_{p,-1}} 1$ binary variables are added to the problem. Therefore, the number of added auxiliary binary variables z is

$$|DS| \sum_{k \in I_{0,9}} \sum_{l \in I_{p,-1}} = |DS| \times |I_{0,9}| \times |I_{p,-1}| = 10(-p)|DS|. \quad \square$$

Proposition 4 The number of auxiliary binary variables z for NMDT in base 2 is $2(-p)|DS|$ for a given value $p < 0$.

Proof For every $j \in DS$, $\sum_{k \in I_{0,9}} \sum_{l \in I_{p,-1}} 1$ binary variables are added to the problem. Therefore, the number of added auxiliary binary variables z is

$$|DS| \sum_{k \in I_{0,9}} \sum_{l \in I_{p,-1}} = |DS| \times |I_{0,1}| \times |I_{p,-1}| = 2(-p)|DS|. \quad \square$$

Figures 1 and 2 illustrate the discretization of λ using decimal and binary base, respectively. Even though for a given p , the binary expansion provides less precision than the decimal expansion, it also requires fewer binary variables. Alternatively, for a given desired precision, fewer binary variables are required, as will be further discussed in Sect. 3.4.

3.2 Eliminating redundant variables and constraints

The original formulation of NMDT presents redundancy in both variables and constraints. Therefore, the first step of the reformulation process is to eliminate these redundant terms. The variables λ and v can be eliminated by replacing them in every constraint they appear with the form given by constraints (20) and (22), respectively.

The second step consists of replacing $z_{j,0,l}$ with $1 - z_{j,1,l}$ for all $j \in DS, l \in I_{p,-1}$. This renders Eq. (24) redundant. Similarly, variable $\hat{x}_{i,j,0,l}$ is replaced by $x_i - \hat{x}_{i,j,0,l}$, thus rendering Eq. (25) redundant.

The last two steps do not involve elimination of constraints, but rather rearrangement of variable labels and indices to accommodate the previous simplifications. First, index k can be dropped, as it refers to the singleton set ($\{1\}$). It should be noted that $k = 0$ can be disregarded [see, for example, constraint (20)], as it only adds variables with null coefficient to the summation. The last step consists of replacing $\Delta\lambda$ and Δv with Δx and Δw , respectively, as λ and v no longer exist.

The simplified model (26)–(37) is hereinafter referred to as reformulated normalized multiparametric disaggregation technique (RNMDT). For every $p \leq 0$ it is denoted as RNMDT $_p$, following the notation used for the previous models.

$$\min \sum_{i|(i,i) \in QT} Q_{0,i,i} w_{i,i} + 2 \sum_{(i,j) \in QT | j > i} Q_{0,i,j} w_{i,j} + f_0(x, y) \tag{26}$$

$$\text{s.t.} \sum_{i|(i,i) \in QT} Q_{r,i,i} w_{i,i} + 2 \sum_{(i,j) \in QT | j > i} Q_{r,i,j} w_{i,j} + f_r(x, y) \leq 0, \quad \forall r \in I_{1,m} \tag{27}$$

$$x_j = (X_j^U - X_j^L) \left(\sum_{l \in I_{p,-1}} 2^l z_{j,l} + \Delta x_j \right), \quad \forall j \in DS \tag{28}$$

$$w_{i,j} = (X_j^U - X_j^L) \left(\sum_{l \in I_{p,-1}} 2^l \hat{x}_{i,j,l} + \Delta w_{i,j} \right), \quad \forall i \in I_{1,n}, j \in I_{1,n} | (i, j) \in QT \tag{29}$$

$$0 \leq \Delta x_j \leq 2^p, \quad \forall j \in DS \tag{30}$$

$$2^p(x_i - X_i^U) + X_i^U \Delta x_j \leq \Delta w_{i,j} \leq 2^p(x_i - X_i^L) + X_i^L \Delta x_j, \quad \forall i, j | (i, j) \in QT \tag{31}$$

$$x_i^L \Delta x_j \leq \Delta w_{i,j} \leq x_i^U \Delta x_j, \quad \forall i, j | (i, j) \in QT \tag{32}$$

$$X_i^L z_{j,l} \leq \hat{x}_{i,j,l} \leq X_i^U z_{j,l}, \quad \forall i, j, l \in QT \times I_{0,p} \tag{33}$$

$$X_i^L(1 - z_{j,l}) \leq x_i - \hat{x}_{i,j,l} \leq X_i^U(1 - z_{j,l}), \quad \forall i, j, l \in QT \times I_{0,p} \tag{34}$$

$$x_i \in [X_i^L, X_i^U], \quad \forall i \in I_{1,n_1} \tag{35}$$

$$y_i \in \{Y_i^L, \dots, Y_i^U\}, \quad \forall i \in I_{1,n_2} \tag{36}$$

$$z_{j,l} \in \{0, 1\}, \quad \forall j, l \in DS \times I_{0,p}. \tag{37}$$

The following technical results concern the reduction in the number of binary variables necessary for representing the expansions, to a given precision 10^p , after performing the proposed reformulations. The total reduction in the number of binary variables is such that only one tenth of the original number of binary variables is required when combining the proposed reformulation and the change of base.

Proposition 5 *The number of auxiliary binary variables z for RNMDT is $(-p)|DS|$ for a given parameter $p < 0$.*

Proof For every $j \in DS$, $\sum_{l \in I_{p,-1}} 1$ binary variables are added to the model. Therefore, the number of added binary variables z is $|DS| \times \sum_{l \in I_{p,-1}} 1 = |DS| \times |I_{p,-1}| = (-p)|DS|$. \square

Theorem 4 *For every $p \leq 0$, the number of auxiliary binary variables z for RNMDT $_p$ is one tenth of the number of binary variables of the problem NMDT $_p$.*

Proof The proof follows from Propositions 3 and 5. \square

Moreover, most of the technical results previously presented concerning the reformulation and the change of base still hold. They are reproduced for the sake of completeness.

Proposition 6 *RNMDT $_p$ problem is a relaxation of the original (MI)QCQP problem of every $p \leq 0$.*

Proof The proof is analogous to that of Proposition 2. □

Theorem 5 For any pair of (p_1, p_2) with $p_1 < p_2 \leq 0$, $RNMDT_{p_2}$ is a relaxation of $NMDT_{p_1}$.

Proof The proof is analogous to that of Theorem 1. □

Theorem 6 For any pair of (p_1, p_2) with $p_1 < p_2 \leq 0$, $RNMDT_{p_1}$ is a tighter (or equal) relaxation of the original (MI)QCQP problem than $RNMDT_{p_2}$.

Proof The proof is analogous to that of Theorem 2. □

3.3 Dynamic-precision RNMDT algorithm

One disadvantage of Algorithm 1 is that all discretized variables are expanded using the same number of partitions (or the same precision in the MDT case), which can result in a rapid increase in the number of binary variables that are added to the problem. In this section, an alternative algorithm is proposed for solving the (MI)QCQP problem using RNMDT, whereby the number of partitions is increased only for the variables that will potentially improve (i.e., tighten) the relaxation. Initially, the single precision parameter p is replaced with a parameter vector p_j for all $j \in DS$, where each entry represents the number of partitions that will be used to expand the variable x_j for all $j \in DS$. The variables that will have their precision increased are then chosen dynamically in each iteration.

This procedure is summarized in Algorithm 2. In each iteration, the variables for which the number of partitions will be increased are chosen by ranking them using the function f_{rank} given in (38). The first term of this function represents the absolute error of the relaxation for the pure quadratic terms in which a given variable is present. The second term is the error in the bilinear terms in which the variable appears. The first N_1 variables with the largest function value are selected and their precision is increased, i.e., p_j is reduced by one unit. For every N_2 iterations, each p_j for all j , is reduced by one unit to ensure convergence (i.e., to ensure that for every $j \in DS$, $p_j \rightarrow -\infty$; therefore, $w_{i,j} \rightarrow x_i x_j$).

$$f_{rank}(j) = \sum_r |Q_{r,j}(w_{j,j} - x_j^2)| + 2 \sum_{((r,i)|i>j|(i,j) \in QT)} |Q_{r,i,j}(w_{i,j} - x_i x_j)| \quad (38)$$

Algorithm 2 Dynamic-precision RNMDT algorithm

Step 0. For all $j \in DS$, set $p_j = 0$ and let $UB = +\infty$ and $iteration = 0$.

Step 1. $iteration = iteration + 1$.

Step 2. Solve relaxation and obtain LB and point (x^R, y^R) .

Step 3. Solve original problem with a local solver with initial solution (x^R, y^R) and fixing integer variables at y^R . If a new best solution is found, save the incumbent solution (x^*, y^*) and update UB .

Step 4. If some of the stopping criteria is met, stop. Otherwise continue.

if $iteration + 1$ is not a multiple of N_2 **then**

Step 5. Rank j using f_{rank} , and for the first N_1 indexes j ranked by f_{rank} , set $p_j = p_j - 1$. return to step 1.

else

Step 5. For all j , set $p_j = p_j - 1$. return to Step 1.

end if

Table 1 Precision \times number of binary variables

p	NMDT—base 10		NMDT—base 2		RNMDT	
	Precision	Binary variables	Precision	Binary variables	Precision	Binary variables
0	1.00E+00	0	1.00E+00	0	1.00E+00	0
−1	1.00E−01	10	5.00E−01	2	5.00E−01	1
−2	1.00E−02	20	2.50E−01	4	2.50E−01	2
−3	1.00E−03	30	1.25E−01	6	1.25E−01	3
−4	1.00E−04	40	6.25E−02	8	6.25E−02	4
−5	1.00E−05	50	3.13E−02	10	3.13E−02	5
−6	1.00E−06	60	1.56E−02	12	1.56E−02	6
−7	1.00E−07	70	7.81E−03	14	7.81E−03	7
−8	1.00E−08	80	3.91E−03	16	3.91E−03	8
−9	1.00E−09	90	1.95E−03	18	1.95E−03	9
−10	1.00E−10	100	9.77E−04	20	9.77E−04	10

Theorem 7 *The sequence of lower bounds generated by Algorithm 2 is monotonic.*

Proof As the parameter p is point-wise decreased in each iteration, monotonicity follows from Theorem 6. \square

3.4 Discussions

It should be noted that the proposed changes aim at reducing the total number of binary variables required for obtaining the relaxation at each iteration. In that sense, the change of base reduces the number of binary variables necessary for expanding the continuous variables, the elimination of redundant variables and constraints reduces the overall model size. The proposed algorithm controls the increase in the model size between iterations.

Table 1 shows the precision and the number of additional binary variables for each choice of the parameter p . The first column represents different choices of p . The remaining columns are grouped in pairs. The first column for each pair represents the precision for the chosen p , i.e., the tightness of the bounds of $\Delta\lambda$ or Δx depending on whether the model is NMDT or RDNMT, respectively. The second column represents the number of auxiliary variables z that are added for each continuous variable that is discretized. There are three pairs of columns, the first is for NDMT using base 10, the second for NMDT using base 2, and the last for RNMDT.

It is easily seen that the binary expansion has two major advantages compared to the decimal expansion; namely, it allows more control over accuracy and generates fewer binary variables for each chosen accuracy. As an illustrative example, if NMDT in base 10 is chosen, ten binary variables are necessary for a precision of 10^{-1} , whereas for RNMDT, the same number of variables results in a precision of $9.77E-04$.

Table 2 shows the model size before and after the elimination of redundant variables and constraints. It is noticeable that the number of additional binary variables (represented by variable z) is reduced by a factor of 5, owing to the base change, and by half after the redundancy elimination. Clearly, the remainder of the model size decreases as well. However, it should be noted that comparing the model sizes for the same parameter p value

Table 2 Model sizes

	NMDT—base 10	NMDT—base 2	RNMDT
x	n_1	n_1	n_1
y	n_2	n_2	n_2
w	$ QT $	$ QT $	$ QT $
z	$10(-p) DS $	$2(-p) DS $	$(-p) DS $
λ	$ DS $	$ DS $	0
$\Delta x/\Delta\lambda$	$ DS $	$ DS $	$ DS $
v	$ QT $	$ QT $	0
\hat{x}	$10(-p) QT $	$2(-p) QT $	$(-p) QT $
$\Delta v/\Delta w$	$ QT $	$ QT $	$ QT $
Binary variables	$10(-p) DS $	$2(-p) DS $	$(-p) DS $
Integer variables	n_2	n_2	n_2
Continuous variables	$n_1 + (10(-p) + 3) QT + 2 DS $	$n_1 + (2(-p) + 3) QT + 2 DS $	$n_1 + ((-p) + 2) QT + DS $
Constraints	$m + (2 + (-p))DS + (4 + 11(-p)) QT $	$m + (2 + (-p)) DS + (4 + 3(-p)) QT $	$m + (-p) DS + (3 + 2(-p)) QT $

can be misleading, since the same value of p leads to different precisions in the different formulations. Nevertheless, if one compare the formulations for a given precision level, the reduction in the number of auxiliary binary variables from base 10 to the RNMDT formulation is approximately by a factor of 3, as can be observed in Table 1.

As the Algorithm 2 (dynamic-precision algorithm) may require different number of iterations from Algorithm 1, and each iteration may have different computational cost, their efficiency is not directly comparable by theoretical analysis. However, the advantage of the dynamic-precision RNMDT algorithm will become clear in the next section in which computational experiments are presented.

4 Computational experiments

In this section, the results obtained using the proposed relaxation and algorithm are presented. The QCQP problem instances were obtained from the literature and we also consider some randomly generated MIQCQP problem instances. All instances were solved by four methods: (1) Algorithm 1 and NMDT in base 10, (2) Algorithm 1 and NMDT in base 2, (3) Algorithm 1 and RNMDT, and (4) Algorithm 2 and RNMDT.

The algorithms were implemented in GAMS on an Intel i7-3612QM with 8GB. The LP/MIP solver was CPLEX 12.6, and the local nonlinear solver was CONOPT 3.17. For the dynamic-precision RNMDT algorithm, N_1 and N_2 were set to 3 and 10, respectively. These values were selected based on early experiments that will be discussed next. A time limit of 1000s and an absolute gap $|UB - LB|$ (optimality tolerance) of 0.001 were set as stop criteria.

4.1 Literature instances

These instances were originally presented in [7]. They were provided by the Optimization Firm, which is responsible for the development of the BARON solver [37].

Table 3 Instances sizes

	Variables	Constraints	Density (%)	Number of instances
Small	8–10	8–40	25	18
	10–20	10–40	50	18
	10–20	10–40	100	18
Medium	28–40	28–80	25	18
	30–40	30–80	50	18
	30–40	30–80	100	18
Large	48	48–96	25	9
	50	50–100	50	9
	50	50–100	100	9

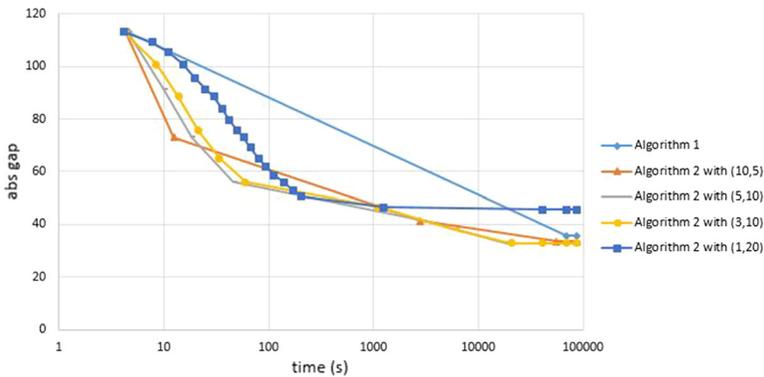


Fig. 3 Setting parameter N_1 and N_2 (logarithm)

All instances are QCQP minimization problems. All variables are continuous except for the auxiliary z variables in the relaxation, which are discrete.

There are 135 instances, and all were nonconvex. The number of variables ranged from 10 to 50, and the number of constraints from 10 to 100. The density of the quadratic matrices Q was 25%, 50%, or 100%. The linear part was 100% dense for all problems. The coefficients from both the quadratic and the linear terms were chosen to be randomly generated numbers chosen uniformly between 0 and 1. Table 3 classifies instances according to problem size and density.

4.1.1 Parameterizing the dynamic-precision algorithm

Algorithm 2 requires the parameters N_1 and N_2 to be set beforehand. To select these values, a representative instance was chosen from the group of large instances and solved with a time limit of 48 h using Algorithm 1 (classic algorithm) and Algorithm 2 (Dynamic-precision algorithm) setting the parameter pair (N_1, N_2) to (1, 20), (3, 10), (5, 10), and (10, 5). The results are shown in Fig. 3 using log-transformation on the time axis. In this figure, each dot represents an iteration and each line a different setting.

As can be seen in Fig. 3, Algorithm 2 was more efficient than Algorithm 1, as the latter required considerably more time to complete iteration 2, primarily owing to the number

Table 4 Solved instances

Size	Density (%)	Total instances	NMDT + base 10	NMDT + base 2	RNMDT + Algorithm 1	RNMDT + Algorithm 2
Small	25	18	18	18	18	18
	50	18	18	18	18	18
	100	18	9	16	16	16
	Total	54	45	52	52	52
Medium	25	18	18	18	18	18
	50	18	9	9	9	10
	100	18	–	–	–	–
	Total	54	27	27	27	28
Large	25	9	1	5	5	6
	50	9	–	–	–	–
	100	9	–	–	–	–
	Total	27	1	5	5	6
Total		135	66	84	84	86

of binary variables added in the relaxation. If a significantly small number of continuous variables have their discretization refined per iteration, then several consecutive iterations with little or no improvement may be observed. For example, this can be seen in the setting (1, 20). In contrast, if a considerably large number of variables are expanded the algorithm requires a larger amount of time to complete the first iterations, as can be seen, for example, in the setting (10, 5). Figure 3 shows that the two intermediate settings are nearly equivalent; however, the setting (3, 10) was chosen as a conservative option in terms of problem growth (as it expands fewer variables per iteration). This experiment showed that, in this case, the performance of the dynamic-precision algorithm for the given parameters was, to a certain degree, robust. A similar behavior was also observed in preliminary experiments with other instances.

4.1.2 Numerical results

Given the choice of parameters $(N_1, N_2) = (3, 10)$ for the dynamic-precision algorithm, all four methods were used for solving the 135 instances. In these experiments, the same optimality tolerance of 0.001 was used; however, the time limit was reduced to 1000 s. One can conclude from the numerical results, the three proposed improvements surpassed, in terms of performance, the relaxation and the algorithm in Castro [13]. The instances solved are summarized in Table 4.

NMDT using base 2 and RNMDT solved 18 additional instances when compared with NMDT using base 10. In particular, they solved seven additional small instances with 100% density. RNMDT with the dynamic-precision algorithm solved two additional instances compared with RNMDT combined with the classic algorithm, namely, one large instance with 25% density and one medium instance with 50% density. To compare the performance of the methods in the instances for which the optimality gap was not closed, Table 5 presents the average relative gaps after termination of the algorithm due to the time limit criterion.

The proposed improvements over the NMDT formulation and the algorithm presented in Castro [13] were both successful in terms of the number of instances solved and also the

Table 5 Relative gaps

Size	Density (%)	NMDT + base 10 (%)	NMDT + base 2 (%)	RNMDT + Algorithm 1 (%)	RNMDT + Algorithm 2 (%)
Small	25	0.0	0.0	0.0	0.0
	50	0.0	0.0	0.0	0.0
	100	14.5	0.0	0.0	0.0
	Total	4.8	0.0	0.0	0.0
Medium	25	1.9	0.0	0.0	0.0
	50	30.3	3.4	3.5	3.7
	100	112.0	65.8	65.7	53.3
	Total	48.1	23.1	23.1	19.0
Large	25	23.8	2.0	2.2	2.4
	50	98.0	56.4	53.5	49.4
	100	185.8	155.2	152.2	115.6
	Total	102.5	71.2	69.3	55.8
Total		41.7	23.5	23.1	18.8

quality of the bounds obtained for the instances that could not be solved to optimality. The single most significant improvement was due to the change in the base of the expansion from 10 to 2 which reducing the average relative gap by nearly half. Furthermore, additional gains were obtained, albeit to a lesser degree, using the reformulation and Algorithm 2. The reformulation and proposed algorithm were significantly more successful for the larger and denser instances, as these instances had many quadratic terms and thus, many variables are needed to be expanded.

4.2 Generated instances

Six MIQCQP instances with 100% density were generated to test the proposed reformulations and algorithms in the mixed-integer case. These instances are available from the authors upon request. As MIQCQP problems are typically more computationally demanding, the time limit was increased to 7200 s.

Table 6 shows the model sizes for the six instances. The average relative gap for NMDT in base 10, NMDT in base 2, RNMDT, and RNMDT with the dynamic-precision algorithm (Algorithm 2) is 163.4%, 121.7%, 124.7%, and 111.3%, respectively. It is clear that Algorithm 2 exhibited the best performance for these instances as well. Moreover, the three methods outperform the formulation and the algorithm in [13], as the lower bound was improved in all cases. Notice that RNMDT presented similar performance than NMDT (in fact, the relative average gap for NMDT is 3% smaller in Table 7) before the introduction of Algorithm 2.

4.3 Comparison with open-source solver

In the experiments that we presented so far, we compared our improvements with the approach proposed in Castro [13]. Next, we present results obtained from comparing the RNMDT with Algorithm 2, the best performing of the four configurations tested, with Couenne [8], a state-of-art open-source global solver for MINLP (MIQCQP inclusive) made available by the

Table 6 Mixed-integer instances

Instance	Continuous variables	Integer variables	Constraints	Density (%)
1	20	10	20	100
2	30	10	30	100
3	30	30	30	100
4	30	30	60	100
5	50	30	100	100
6	100	100	100	100

Table 7 Relative gaps for generated instances

Instance	NMDT + base 10 (%)	NMDT + base 2 (%)	RNDMT + Algorithm 1 (%)	RNDMT + Algorithm 2 (%)
1	15.9	0.1	0.1	0.1
2	114.7	51.1	47.2	54.1
3	58.2	20.1	18.6	18.4
4	89.0	50.3	50.8	48.4
5	340.9	258.8	257.5	222.5
6	361.6	349.8	373.4	324.4
Total	163.4	121.7	124.6	111.3

COIN-OR [28] initiative. Couenne relies on convex over and under envelopes and spatial BnB.

Table 8 details the size of generated instances. Note that, as is the case for the previously subsections, these are fully dense instances, e.g., an instance with 50 constraints and 50 variables has $(50 \times 49/2 + 50) \times (50 + 1) = 65,025$ bilinear terms (that is, nonzero entries in the Hessian matrices). We opted for this setting so that we could asses the performance of the algorithm under the most challenging instances possible using a similar number of variables and constraints of those instances available in the literature. Nevertheless, we highlight that practical problems of that nature are typically much sparser, meaning that larger instances could potentially be solved, if those instances were available. Considering the computational platform used, we were not able to solve instances larger than instance 8 in Table 8 due to memory shortage caused by the size of the dense Hessian matrix.

Table 9 shows the results in terms of relative gaps for both RNMDT with Algorithm 2 and for Couenne. All experiments were terminated due to the time limit of 3600 s. In Sect. 4.4 we present the performance profiles for these results.

4.4 Performance profiles

To provide a structured comparison between the configurations being compared, performance profiles based on Dolan and Moré [15] are presented. Let $t_{s,ip}$ be the time taken by a given solver or algorithm s to solve the instance problem ip . Let $r_{s,ip}$ be defined as follows $r_{s,ip} = t_{s,ip} / \min\{t_{s,ip} : s \in S\}$ where S is the set of all solvers and algorithm that are being compared in the experiment. Let the time performance profile $\rho_t(\tau)$ be defined as $\rho_t(\tau) = |\{ip \in IP : r_{s,ip} \leq \tau\}| / |IP|$.

Table 8 Instances size—comparison with open-source solver

Instance	Continuous variable	Constraints	Integer variables
1	50	50	0
2	50	50	10
3	50	50	50
4	60	60	50
5	60	60	60
6	70	70	50
7	100	100	0
8	100	100	10

Table 9 Results—relative gap—comparison with open-source solver

Instance	Couenne (%)	RNMDT + Algorithm 3 (%)
1	315	262
2	294	234
3	229	167
4	305	278
5	267	211
6	302	265
7	530	464
8	554	465

$r_{s,ip} \leq \tau\} / |S|$, where IP is the set of all instance problems of the experiment and $|x|$ denotes the cardinality of x . Similarly, let $g_{s,ip}$ be the relative gap achieved by the solver or algorithm s for the instance problem ip . Let the relative gap performance profile $\rho_g(\tau)$ be defined as $\rho_g(\tau) = |\{ip \in IP : g_{s,ip} \leq \tau\}| / |S|$. Figures 4 and 5 presents the time and gap performance profile, respectively, for the computational experiments performed in Sects. 4.1 and 4.2 combined, while Fig. 6 presents the gap profile for the instances used in Sect. 4.3. Notice that, in Figs. 4 and 5, the x-axes are plotted in logarithmic scale, while in Fig. 6 it is used linear scale.

The NMDT with basis 2 and the RNMDT with Algorithm 1 presented similar performance profiles, and both were faster and achieved better bounds than NMDT with basis 10. The RNMDT with Algorithm 2 was slower than with Algorithm 1, since it requires more iterations for the instances that both could solve, which explains the behavior depicted on the beginning (left-hand side) of the time performance profile. However, the gap performance profile shows its superior performance in terms of reaching smaller optimality gaps.

5 Conclusions

In this paper, three key improvements to the NMDT were proposed. Namely, the replacement of decimal expansion with binary expansion, the reduction of model size, thus eliminating redundant variables and constraints in the formulation, and a new algorithm for solving (MI)QCQP problems using this relaxation that allows the control of the number of binary variables added per iteration.

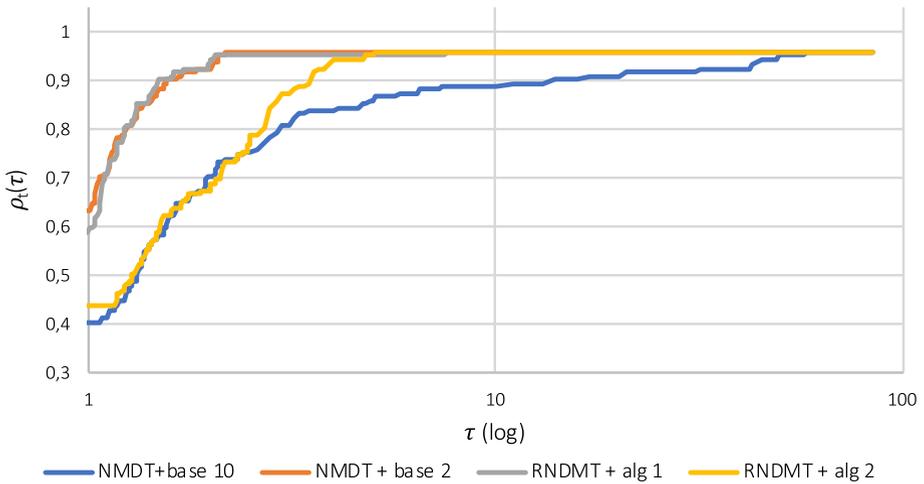


Fig. 4 Time performance profile—Sects. 4.1 and 4.2

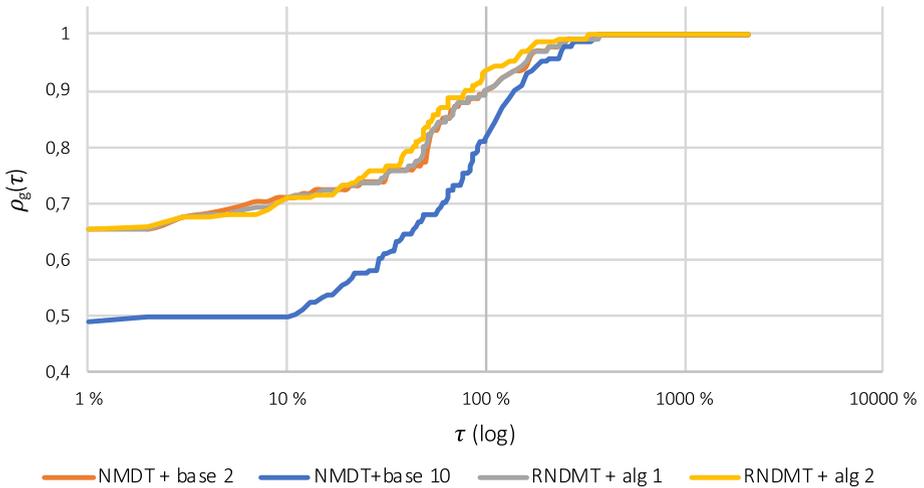


Fig. 5 Relative gap performance profile—Sects. 4.1 and 4.2

Instances from the literature and also a set of randomly generated instances were used to assess the performance of the reformulations and the new algorithm. The results showed that the reformulation is easier to solve than the formulation available in the literature, thus providing better bounds at the same computational cost and achieving global optimality for more instances. The proposed algorithm appears to be particularly useful in the presence of many quadratic terms, as in the case of high-density problems. Despite having more parameters to configure, preliminary experiments suggest that its performance is robust for different parameter settings.

The proposed method (RNDMT + Algorithm 2) also showed good results when compared to the state-of-art (open-source) global solver Couenne. Future work include to incorporate cuts and other primal heuristics in our method to increase performance (such as those available

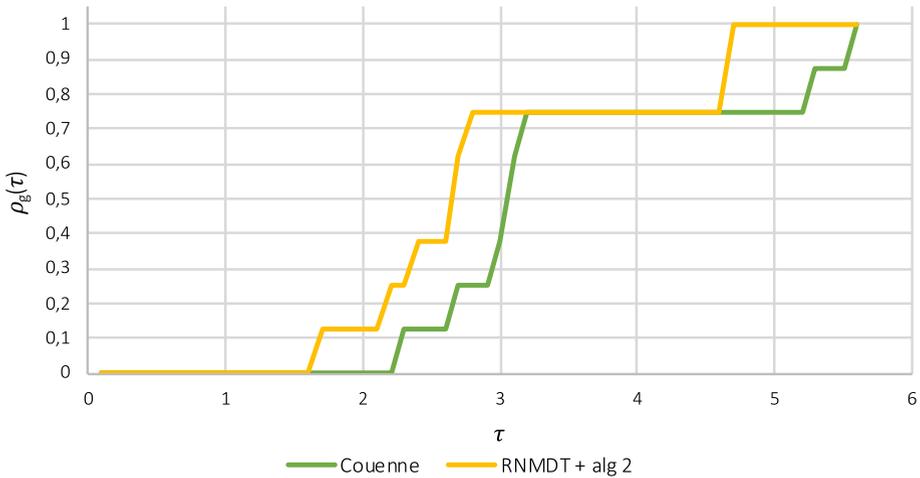


Fig. 6 Relative gap performance profile—Sect. 4.3

in global solvers such as Couenne), and to compare with other global solvers using instances derived from real-world problems.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Al-Khayyal, F.A., Falk, J.E.: Jointly constrained biconvex programming. *Math. Oper. Res.* **8**(2), 273–286 (1983)
2. Andrade, T., Ribas, G., Oliveira, F.: A strategy based on convex relaxation for solving the oil refinery operations planning problem. *Ind. Eng. Chem. Res.* **55**(1), 144–155 (2016)
3. Androulakis, I.P., Maranas, C.D., Floudas, C.A.: α BB: a global optimization method for general constrained nonconvex problems. *J. Glob. Optim.* **7**(4), 337–363 (1995)
4. Anstreicher, K.M.: Semidefinite programming versus the reformulation-linearization technique for non-convex quadratically constrained quadratic programming. *J. Glob. Optim.* **43**(2), 471–484 (2009)
5. Bagajewicz, M.: A review of recent design procedures for water networks in refineries and process plants. *Comput. Chem. Eng.* **24**(9), 2093–2113 (2000)
6. Balas, E.: Disjunctive programming. *Ann. Discrete Math.* **5**, 3–51 (1979)
7. Bao, X., Sahinidis, N.V., Tawarmalani, M.: Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs. *Optim. Methods Softw.* **24**(4–5), 485–504 (2009)
8. Belotti, P.: Couenne: a user’s manual. Technical Report, Lehigh University (2009)
9. Boland, N., Christiansen, J., Dandurand, B., Eberhard, A., Oliveira, F.: A parallelizable augmented lagrangian method applied to large-scale non-convex-constrained optimization problems. *Math. Prog.* (2018). <https://doi.org/10.1007/s10107-018-1253-9>
10. Bomze, I.M.: Copositive optimization—Recent developments and applications. *Eur. J. Oper. Res.* **216**(3), 509–520 (2012)
11. Bomze, I.M., Dür, M., De Klerk, E., Roos, C., Quist, A.J., Terlaky, T.: On copositive programming and standard quadratic optimization problems. *J. Glob. Optim.* **18**(4), 301–320 (2000)
12. Castro, P.M.: Tightening piecewise McCormick relaxations for bilinear problems. *Comput. Chem. Eng.* **72**, 300–311 (2015)

13. Castro, P.M.: Normalized multiparametric disaggregation: an efficient relaxation for mixed-integer bilinear problems. *J. Glob. Optim.* **64**(4), 765–784 (2016)
14. Cplex, I.: Ilog cplex 12.6 optimization studio (2014)
15. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. *Math. Program.* **91**(2), 201–213 (2002)
16. Fampa, M., Lee, J., Melo, W.: On global optimization with indefinite quadratics. *EURO J. Comput. Optim.* **5**(3), 309–337 (2017)
17. FICO, T.: Xpress optimization suite. Xpress-Optimizer, Reference manual, Fair Isaac Corporation (2009)
18. Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP*, vol. 29. W.H. Freeman, New York (2002)
19. Gleixner, A.M., Berthold, T., Müller, B., Weltge, S.: Three enhancements for optimization-based bound tightening. *J. Glob. Optim.* **67**(4), 1–27 (2016)
20. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM (JACM)* **42**(6), 1115–1145 (1995)
21. Gounaris, C.E., Misener, R., Floudas, C.A.: Computational comparison of piecewise-linear relaxations for pooling problems. *Ind. Eng. Chem. Res.* **48**(12), 5742–5766 (2009)
22. Jeroslow, R.: There cannot be any algorithm for integer programming with quadratic constraints. *Oper. Res.* **21**(1), 221–224 (1973)
23. Jezowski, J.: Review of water network design methods with literature annotations. *Ind. Eng. Chem. Res.* **49**(10), 4475–4516 (2010)
24. Karuppiah, R., Grossmann, I.E.: Global optimization for the synthesis of integrated water systems in chemical processes. *Comput. Chem. Eng.* **30**(4), 650–673 (2006)
25. Kolodziej, S., Castro, P.M., Grossmann, I.E.: Global optimization of bilinear programs with a multiparametric disaggregation technique. *J. Glob. Optim.* **57**(4), 1039–1063 (2013)
26. Li, H.L., Chang, C.T.: An approximate approach of global optimization for polynomial programming problems. *Eur. J. Oper. Res.* **107**(3), 625–632 (1998)
27. Linderoth, J.: A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs. *Math. Program.* **103**(2), 251–282 (2005)
28. Lougee-Heimer, R.: The common optimization interface for operations research. *IBM J. Res. Dev.* **47**(1), 57–66 (2003)
29. Lovász, L.: On the shannon capacity of a graph. *IEEE Trans. Inf. Theory* **25**(1), 1–7 (1979)
30. Luo, Z.Q., Ma, W.K., So, A.M.C., Ye, Y., Zhang, S.: Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Proc. Mag.* **27**(3), 20–34 (2010)
31. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: part I convex underestimating problems. *Math. Program.* **10**(1), 147–175 (1976)
32. Misener, R., Floudas, C.A.: Advances for the pooling problem: modeling, global optimization, and computational studies. *Appl. Comput. Math.* **8**(1), 3–22 (2009)
33. Nowak, I.: Dual bounds and optimality cuts for all-quadratic programs with convex constraints. *J. Glob. Optim.* **18**(4), 337–356 (2000)
34. Optimization, G.: Inc., 2014. Gurobi optimizer reference manual (2014). <http://www.gurobi.com>
35. Pardalos, P.M., Vavasis, S.A.: Quadratic programming with one negative eigenvalue is NP-hard. *J. Glob. Optim.* **1**(1), 15–22 (1991)
36. Rockafellar, R.T.: Lagrange multipliers and optimality. *SIAM Rev.* **35**(2), 183–238 (1993)
37. Sahinidis, N.V.: Baron: a general purpose global optimization software package. *J. Glob. Optim.* **8**(2), 201–205 (1996)
38. Sherali, H.D., Adams, W.P.: *A Reformulation-linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, vol. 31. Springer, Berlin (2013)
39. Sherali, H.D., Fraticelli, B.M.: Enhancing rlt relaxations via a new class of semidefinite cuts. *J. Glob. Optim.* **22**(1), 233–261 (2002)
40. Tawarmalani, M., Sahinidis, N.V.: Global optimization of mixed-integer nonlinear programs: a theoretical and computational study. *Math. Program.* **99**(3), 563–591 (2004)
41. Teles, J.P., Castro, P.M., Matos, H.A.: Global optimization of water networks design using multiparametric disaggregation. *Comput. Chem. Eng.* **40**, 132–147 (2012)
42. Teles, J.P., Castro, P.M., Matos, H.A.: Multi-parametric disaggregation technique for global optimization of polynomial programming problems. *J. Glob. Optim.* **55**(2), 227–251 (2013)
43. Thakur, L.S.: Domain contraction in nonlinear programming: minimizing a quadratic concave objective over a polyhedron. *Math. Oper. Res.* **16**(2), 390–407 (1991)
44. Tuy, H.: On solving nonconvex optimization problems by reducing the duality gap. *J. Glob. Optim.* **32**(3), 349–365 (2005)

45. Van Voorhis, T.: A global optimization algorithm using lagrangian underestimates and the interval newton method. *J. Glob. Optim.* **24**(3), 349–370 (2002)
46. Visweswaran, V.: MINLP: applications in blending and pooling problems. In: Floudas, C., Pardalos, P. (eds.) *Encyclopedia of Optimization*, pp. 2114–2121 Springer, Boston, MA (2008)
47. Wicaksono, D.S., Karimi, I.A.: Piecewise MILP under-and overestimators for global optimization of bilinear programs. *AIChE J.* **54**(4), 991–1008 (2008)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.