Journal of Global Optimization manuscript No.
(will be inserted by the editor)

Linear and parabolic relaxations for quadratic constraints

Ferenc Domes · Arnold Neumaier

Received: date / Accepted: date

Abstract This paper presents new techniques for filtering boxes in the presence of an additional quadratic constraint, a problem relevant for branch and bound methods for global optimization and constraint satisfaction. This is done by generating powerful linear and parabolic relaxations from a quadratic constraint and bound constraints, which are then subject to standard constraint propagation techniques. The techniques are often applicable even if the original box is unbounded in some but not all variables.

As an auxiliary tool – needed to make our theoretical results implementable in floating-point arithmetic without sacrificing mathematical rigor – we extend the directed Cholesky factorization from Domes & Neumaier (SIAM J. Matrix Anal. Appl. 32 (2011), 262–285) to a partial directed Cholesky factorization with pivoting. If the quadratic constraint is convex and the initial bounds are sufficiently wide, the final relaxation and the enclosure are optimal up to rounding errors.

Numerical tests show the usefulness of the new factorization methods in the context of filtering.

Keywords quadratic constraints, non-convex constraints, interval analysis, constraint satisfaction problems, parabolic relaxations, ellipsoid relaxations, linear relaxations, interval hull, directed modified Cholesky factorization, rounding error control, verified computing

Mathematics Subject Classification (2000) 90C20 quadratic programming, 15A23, factorization of matrices, 49M27 decomposition methods

1 Introduction

This paper presents new techniques for filtering boxes in the presence of an additional quadratic constraint, a problem relevant for branch and bound methods for global optimization and constraint satisfaction. This is done by generating powerful linear

Faculty of Mathematics, University of Vienna. Oskar-Morgenstern-Platz 1, A-1090 Vienna, Austria

and parabolic relaxations from a quadratic constraint and bound constraints, which are then subject to standard constraint propagation techniques.

In order to make the technique work, one needs to find a set of variables containing all unbounded variables such that the restriction of the Hessian matrix to these variables is positive definite. Then one can bound these variables in terms of the others using inequalities derived in Domes & Neumaier [3]. This leads to a bounded box even when the original problem is unbounded. If the quadratic constraint is convex and the initial bounds are sufficiently wide, the final relaxation and the enclosure are optimal up to rounding errors.

As an auxiliary tool – needed to make our theoretical results implementable in floating-point arithmetic without sacrificing mathematical rigor – we generalize the directed Cholesky factorization, making it applicable even on indefinite matrices, in which case it returns an incomplete or modified factorization. This method is basically a rigorous version of the (approximate) modified Cholesky factorization discussed, e.g., in Schnabel & Eskow [18,19]. The development of this auxiliary tool was also motivated by our research in global optimization (Domes & Neumaier [4]) where we had to factorize the reduced Hessian for which theory requires that a certain submatrix is positive definite, but the remainder of the matrix can be indefinite. Since the previous directed Cholesky factorization methods from Domes & Neumaier [3] proved to be useful for other researchers (e.g., [8,12,13,15]) the new generalized directed Cholesky factorization presented here might also find other applications which are outside the scope of this paper.

Some of the results of this paper are based on our technical report Domes & Neumaier [5], where early versions of the directed modified Cholesky factorization are presented.

The filtering methods presented give the possibility to obtain rigorous bounds on variables that are consequences of the constraints, without the need of having finite prior bounds on them. Similarly as the original `ehull` enclosure from [3], this turns the new methods into powerful tools in branch and bound methods for solving constrained optimization problems (e.g., [6,12,13,16]).

Acknowledgment. This research was supported by the Austrian Science Fund (FWF) under the contract numbers P23554-N13 and P22239-N13.

1.1 Content

We discuss enhancements to the indefinite case of the `ehull` enclosure technique from Domes & Neumaier [3], developed for enclosing the solution set of convex quadratic inequalities. The new techniques are presented in the following logical stages: first in Section 1.2 the notation is given, followed by the problem specification in Section 1.3. After this in Section 2 three simple but interesting examples are given and serve as motivation for Section 3; the main theoretical part discussing convex quadratic relaxations. The next part (Section 4) supports the results of the previous section by introducing the generalized directed Cholesky factorization – a greatly enhanced version of the directed Cholesky factorization from [3]. In Section 5 two new methods to compute rigorous parabolic, elliptical (strictly convex) and linear relaxations for uncertain quadratic inequalities are discussed. The new methods are based on the theoretical foundations from Section 3 and on the generalized directed Cholesky factorization from Section 4. In Section 6 the methods presented in Domes

& Neumaier [3,5], as well as the ones from Section 5 are combined to filter uncertain quadratic optimization problems. A detailed algorithm is also presented, and is using directed rounding and interval computations to make sure that its results are rigorous in floating point arithmetic. Numerical tests in Section 7 show the general usefulness of the filtering method, by applying it after the quadratic constraint propagation has already reached its contraction limit. The tests also compare the older and the new versions of the methods on a large and well known testset.

1.2 **Notation**

In this section we define the notation used throughout the paper.

Matrices. $\mathbb{R}^{m\times n}$ denotes the vector space of all $m \times n$ matrices A with real entries A_{ik} $(i = 1,\ldots,m,\ k = 1,\ldots,n)$, and $\mathbb{R}^n = \mathbb{R}^{n\times 1}$ denotes the vector space of all column vectors of length n. For vectors and matrices, the relations $=$, $\neq$, $<$, $>$, $\leq$, $\geq$ and the absolute value $|A|$ of a matrix A are interpreted component-wise.

The n-dimensional identity matrix is denoted by I and the n-dimensional zero matrix is denoted by 0. The transpose of a matrix A is denoted by A^T, and A^{-T} is short for $(A^T)^{-1}$. The ith row vector of a matrix A is denoted by $A_{i:}$ and the jth column vector by $A_{:j}$. For an $n \times n$ matrix A, $\mathrm{diag}(A)$ denotes the n-dimensional vector with $\mathrm{diag}(A)_i = A_{ii}$.

We use finite index lists N for subset selection and permutations. We call N **an ordered index list** if $N_i < N_{i+1}$. The number of elements of N is denoted by $|N|$. The ordered index list $\neg N$ denotes the complement of N. Let $I \subseteq \{1,\ldots,m\}$ and $J \subseteq \{1,\ldots,n\}$ be index lists and let $n_I := |I|$, $n_J := |J|$. For an n-dimensional vector x, x_J denotes the n_J-dimensional vector built from the components of x selected by by J, i.e. $(x_J)_i := x_{J_i}$. For an $m \times n$ matrix A, the expression $A_{I:}$ denotes the $n_I \times n$ matrix built from the rows of A selected by the index list I. Similarly, $A_{:J}$ denotes the $m \times n_J$ matrix built from the columns of A selected by the index list J. Instead of using the ordered index lists $I := \{i, i+1,\ldots,k\}$ and $J := \{j, j+1,\ldots,l\}$ we sometimes also write $A_{i:k,j:l}$.

Boxes. A **box** $\mathbf{x} = [\underline{x}, \overline{x}]$, i.e., the Cartesian product of the closed real intervals $\mathbf{x}_i := [\underline{x}_i, \overline{x}_i]$, representing a (bounded or unbounded, possibly empty) axiparallel box in $\mathbb{R}^n$. $\overline{\mathbb{IR}}^n$ denotes the set of all n-dimensional boxes. To take care of one-sided bounds on variables, the values $-\infty$ and ∞ are allowed as lower and upper bounds of a box, respectively. The condition $x \in \mathbf{x}$ is equivalent to the collection of simple bounds

$$\underline{x}_i \leq x_i \leq \overline{x}_i \quad (i = 1,\ldots,n),$$

or, with inequalities on vectors and matrices interpreted component-wise, to the two-sided vector inequality $\underline{x} \leq x \leq \overline{x}$. Apart from two-sided constraints, this includes with $\mathbf{x}_i = [a, a]$ variables x_i fixed at a particular value $x_i = a$, with $\mathbf{x}_i = [a, \infty]$ lower bounds $x_i \geq a$, with $\mathbf{x}_i = [-\infty, a]$ upper bounds $x_i \leq a$, and with $\mathbf{x}_i = [-\infty, \infty]$ free variables.

Interval matrices are matrices with uncertain components whose values are known only to lie withing given intervals. All boxes may be considered as interval vectors, i.,e., $n \times 1$ interval matrices. For the notation in interval analysis we mostly follow [11]. In particular, the **midpoint**, **width** and **radius** of an interval matrix $\mathbf{A}$ are

the scalar matrices defined by

$$\mathrm{mid}(\mathbf{A}) := (\overline{A} + \underline{A})/2, \quad \mathrm{wid}(\mathbf{A}) := \overline{A} - \underline{A}, \quad \mathrm{rad}(\mathbf{A}) := \mathrm{wid}(\mathbf{A})/2,$$

respectively. An interval, interval vector, or interval matrix is called **thin** or **degenerate** if its width is zero, and **thick** if its width is positive. A real matrix A is identified with the thin interval matrix with $\underline{A} = \overline{A} = A$.

The expression $\mathbf{A} := [\underline{A}, \overline{A}] \in \overline{\mathbb{IR}}^{m\times n}$ denotes an $m \times n$ interval matrix with lower bound $\underline{A}$ and upper bound $\overline{A}$. $\mathbf{A} \in \overline{\mathbb{IR}}^{n\times n}$ is symmetric if $\mathbf{A}_{ik} = \mathbf{A}_{ki}$ for all $i, k \in \{1, \ldots, n\}$. The comparison matrix $\langle \mathbf{A} \rangle$ of a square interval matrix $\mathbf{A}$ is defined by (using the notation $|\cdot|$ and $\langle\cdot\rangle$ from [11]) as

$$\langle \mathbf{A} \rangle_{ij} := \begin{cases} -|\mathbf{A}_{ij}| & \text{for } i \neq j, \\ \langle \mathbf{A}_{ij} \rangle & \text{for } i = j. \end{cases}$$

Uncertain vectors and matrices. To rigorously account for inaccuracies in computed entries of a matrix we use interval vectors and interval matrices, standing for uncertain real vectors and matrices, respectively, whose entries are unknown apart from lying between given lower and upper bounds.

Note that the concept of uncertainty used in this paper is unrelated to the uncertainty in a stochastic process; it is bounded epistemic uncertainty (not knowing more than bounds for a number), not aleatoric uncertainty (for which a probability distribution exists).

1.3 Uncertain quadratic inequalities and optimization problems

In this section the formal definition of uncertain quadratic inequalities and uncertain quadratic optimization problems is given. These are the problem classes for which the methods discussed in this paper are applicable. Since standard quadratic inequalities and quadratic optimization problems are special cases of their uncertain variants, these are also automatically treated by our methods.

Definition 1 The uncertain quadratic inequality (UQI) is defined in the one sided form as

$$x^T A x + 2a^T x \leq \alpha, \quad A \in \mathbf{A}, \quad a \in \mathbf{a}, \tag{1}$$

or in the two sided form as

$$x^T A x + 2a^T x \in \mathbf{d}, \quad A \in \mathbf{A}, \quad a \in \mathbf{a}, \tag{2}$$

where $\mathbf{A} \in \mathbb{IR}^{n\times n}$ is a symmetric interval matrix, $\mathbf{a} \in \mathbb{IR}^n$ is an interval vector, $\alpha \in \mathbb{R}$ is a constant, and $\mathbf{d} \in \overline{\mathbb{IR}}$ is an interval.

Definition 2 A single-objective, quadratic, uncertain optimization problem (QUOP) consisting of equality and inequality constraints can be written as

$$\begin{aligned} \min \;& x^T A^0 x + 2a^{0T} x + d \leq c_0, \\ \text{s.t.} \;& x^T A^1 x + 2a^{1T} x \;\; \in \mathbf{c}_1, \\ & \qquad \vdots \\ & x^T A^m x + 2a^{mT} x \in \mathbf{c}_m, \\ & x \in \mathbf{x}, \\ \text{with } & A^i \in \mathbf{A}^i, \quad a^i \in \mathbf{a}^i \text{ for all } i \in \{0, \ldots, m\}, \end{aligned} \tag{3}$$

where $A^i \in \mathbb{R}^{n\times n}$, $a^i \in \mathbb{R}^n$, $d \in \mathbb{R}$, c_0 is (not necessary finite) upper bound on the objective and $\mathbf{c} \in \overline{\mathbb{IR}}^m$ is a box defining the bounds of the constraints; equality constraints and one-sided inequality constraints are included.

The above definitions are variations of the concept discussed in [4]. The coefficients A, a, A^i and a^i in (1), (2) and (3) are allowed to vary in narrow intervals $\mathbf{A}$, $\mathbf{a}$, $\mathbf{A}^i$ and $\mathbf{a}^i$ in order to be able to rigorously account for uncertainties due to one of the following sources:

- measurements of limited accuracy,
- conversion errors from an original representation to our normal form,
- rounding errors when creating new constraints by relaxation techniques.

The entries of A, a, A^i and a^i are *not* variables but uncertain constants, whose precise values within the bounds $A \in \mathbf{A}$, $a \in \mathbf{a}$, $A^i \in \mathbf{A}^i$ and $a^i \in \mathbf{a}^i$ are not known. Thus whether a particular vector x is a solution of the UQI or QUOP may depend on which $A \in \mathbf{A}$, $a \in \mathbf{a}$, $A^i \in \mathbf{A}^i$ and $a^i \in \mathbf{a}^i$ is the true value. This ambiguity makes working with uncertain constraints nontrivial. It requires great care in the derivation of methods to ensure the validity of an enclosure no matter which value $A \in \mathbf{A}$, $a \in \mathbf{a}$, $A^i \in \mathbf{A}^i$ and $a^i \in \mathbf{a}^i$ is the true value.

If $\mathbf{A}^0 = 0$, $\mathbf{a}^0 = 0$ and $c_0 = \infty$ the QUOP given by (3) reduces to a quadratic, uncertain constraint satisfaction problem (QUCSP).

If $\mathbf{A}$ and $\mathbf{a}$ are thin, both (1) and (2) reduce to a standard quadratic inequality and if all $\mathbf{A}^i$ and $\mathbf{a}^i$ are thin, (3) reduces to a standard quadratic optimization problem.

In global optimization, an important part is to find an upper bound on the optimal objective value by finding an approximate feasible local solution of the problem using a local solver (e.g., IPOPT [21]).

1.4 **Objective upper bound**

A good upper bound $\overline{f}$ on the objective function, obtained in practice as a rigorous upper bound of a known feasible point, can be used to eliminate much of the search space by adding to the constraint set the additional constraint

$$f(x) \leq \overline{f}.$$

In particular, when this feasible point is already an approximation to a global minimizer, it leaves only a tiny feasible region close to the set of global optimizers. Since an approximate global minimizer is usually found quite early in the branch and bound process, using this additional constraint often saves a large amount of time by speeding up the branch and bound.

In rigorous constrained global optimization, an enclosure of a truly feasible point is necessary to obtain a valid upper bound on the optimal objective function value. How to find such an enclosure when only a (typically slightly infeasible) approximation is known is the subject of articles such as [9, Section 12], [10], and [7].

The methods presented in this paper can make use of this upper bound. This explains why we have added c_0 in the definition of (3). In particular, the solution of quadratic programs benefits from the new techniques only if a constraint on the objective is added, as this gives the only quadratic constraint.

1.5 **Branch and bound**

One of the important applications of filtering methods is global optimization via branch and bound, where filtering steps are used to eliminate subproblems or to reduce the bounding box of a subproblem before branching again. Therefore it is interesting to investigate the usefulness of a method both before and during the branch and bound process.

2 Motivating examples

We begin with a few non convex toy problems that illustrate how the techniques to be discussed work in simple but instructive circumstances. Since our new techniques are applied to quadratic constraints one by one, it suffices to illustrate how problems with a single quadratic constraint can be bounded. Since the optimal enclosure of a domain defined by a single convex quadratic constraint is already treated in [3] (which is used unmodified in our implementation), the toy problems discussed below are deliberately nonconvex.

The variable bounds are chosen such that the feasible region is nicely bounded, but otherwise as large as possible (indeed unbounded) to demonstrate the possible efficiency of the method on even very large domains.

For all our toy problems, known fast filtering methods like quadratic constraint propagation yield no variable bound improvement. (We did not try higher-order consistency techniques as these tend to be quite slow.)

2.1 **Example 1: 2-dimensional hyperbolic region**

We consider the following toy problem:

$$\begin{aligned} &f := 5x_1^2 + 12x_1x_2 + 5x_2^2 - 3x_1 - x_2 \leq 6, \\ &x_1 \in [-2, 1]. \end{aligned} \tag{4}$$

We first eliminate the unbounded variable x_2. A partial Cholesky factorization produces the inequality

$$5(x_2 + 1.2x_1 - 0.1)^2 \leq 2.2x_1^2 + 1.8x_1 + 6.05 \leq 11.25,$$

where the upper bound on the right hand side is obtained by quadratic constraint propagation. Division by 5 and taking the square root gives the constraint

$$x_2 + 1.2x_1 - 0.1 \in [-1.5, 1.5].$$

This provides a linear relaxation of the quadratic constraint that is nearly perfect, as Figure 1 shows. Quadratic constraint propagation on the new constraint leads to the finite bounds $x_2 \in \mathbf{x}_2 := [-2.6, 4]$ for the previously unbounded variable x_2. For comparison, applying MATHEMATICA with

```
Reduce[Exists[{xi}, f <= 6 && x1 >= -2 && x1 <= 1], Reals]
```

(where f is given by (4) and `xi` is either `x1` or `x2`) results in the optimal interval enclosure

$$x_1 \in [-2, 1], \quad x_2 \in [-2.51774, 4].$$

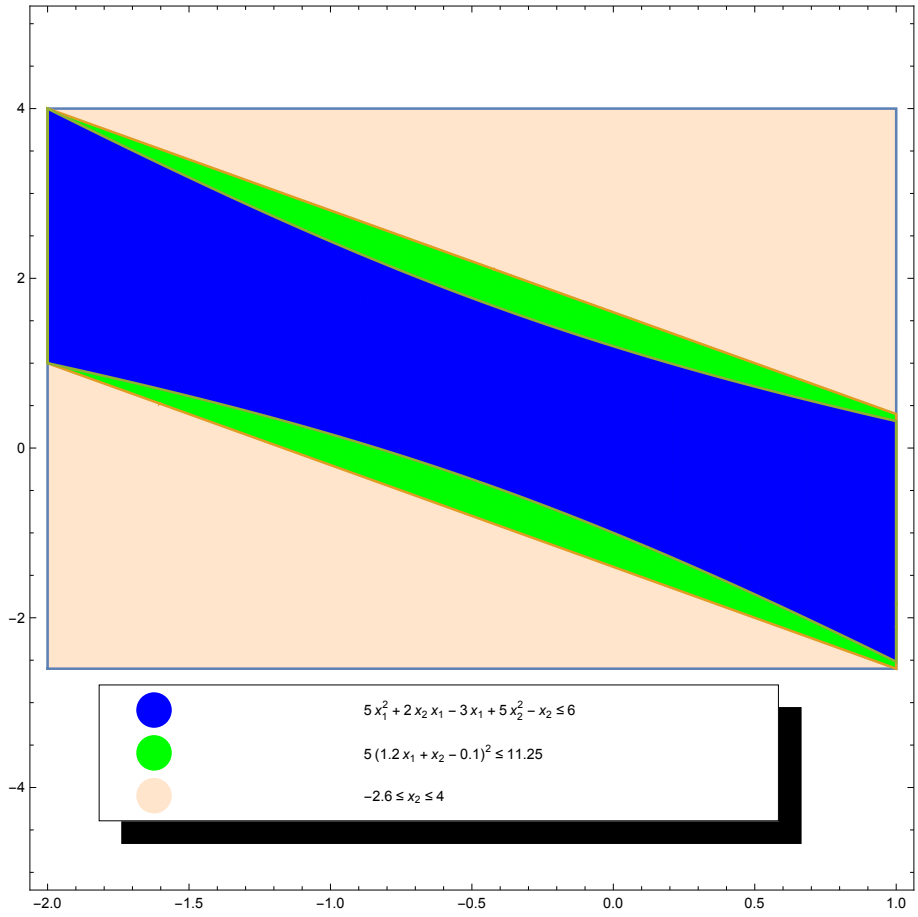

Fig. 1 Example 1.

2.2 **Example 2: 3-dimensional region with hole**

We consider the following toy problem:

$$f := -3x_1 + 5x_1^2 - x_2 + 12x_1x_2 + 5x_2^2 + 5x_3 - 15x_1x_3 - 12x_2x_3 + 6x_3^2 \le 2.75, \quad x_1 \in [-2, 1], \quad x_3 \in [0, 3]. \tag{5}$$

The quadratic constraint represents the exterior of a one-sheeted hyperboloid; in the region defined by the bound constraints, it describes a domain with a hole.

Again we first eliminate the unbounded variable x_2. A partial Cholesky factorization produces the inequality

$$5(x_2 + 1.2x_1 - 1.2x_3 - 0.1)^2 \le 2.2x_1^2 + 0.6x_1x_3 + 1.2x_3^2 + 1.8x_1 - 3.8x_3 + 2.8 \le 9.8,$$

where the upper bound on the right hand side is obtained by quadratic constraint propagation. Division by 5 and taking the square root gives the constraint

$$x_2 + 1.2x_1 - 1.2x_3 - 0.1 \in [-1.4, 1.4].$$

This provides a linear relaxation of the quadratic constraint that is nearly perfect, as Figure 2 shows. Quadratic constraint propagation on the new constraint leads to the finite bounds $x_2 \in \mathbf{x}_2 := [-2.5, 7.5]$ for the previously unbounded variable x_2. For comparison, applying MATHEMATICA with

```
Reduce[Exists[{xi, xj}, f <= 9.8 && x1 >= -2 && x1 <= 1 &&
              x3 >= 0 && x3 <= 3], Reals]
```

(where f is given by (5) and {xi, xj} is one of {x1, x2} or {x1, x3} or {x2, x3}) results in the optimal interval enclosure

$$x_1 \in [-2, 1],$$
$$x_2 \in [-2.43417, 7.18628],$$
$$x_3 \in [0, 3].$$

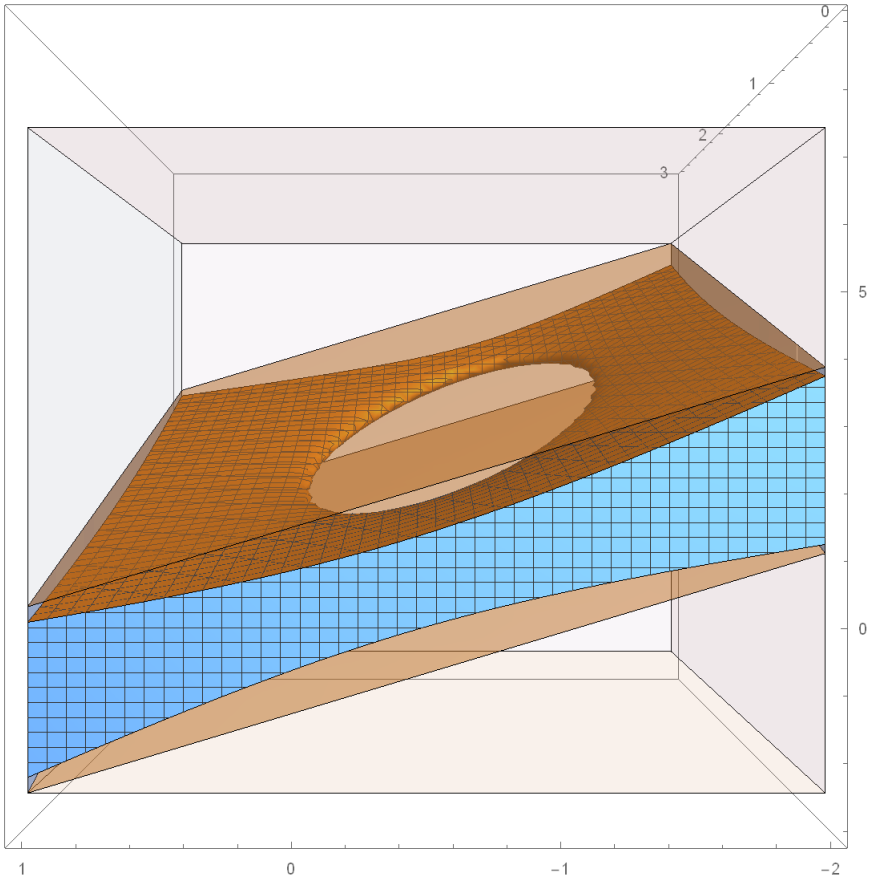

Fig. 2 Example 2.

2.3 **Example 3: Disconnected 3-dimensional region**

For

$$f := x_1^2 + 8x_1x_2 - x_1x_3 + 10x_2^2 - 10x_2x_3 + 5x_3^2 - 2x_1 - 6x_2 + 4x_3 \le -1.7, \quad x_1 \in [-2, 1] \tag{6}$$

we first eliminate the unbounded variables x_2 and x_3. A partial Cholesky factorization produces the parabolic inequality

$$10(x_2 - 0.5x_3 + 0.4x_1 - 0.3)^2 + 2.5(x_3 + 0.6x_1 + 0.2)^2 \le 1.5x_1^2 + 0.2x_1 - 0.7 \le 4.9$$

drawn in Figure 3; the upper bound on the right hand side is obtained by quadratic constraint propagation.

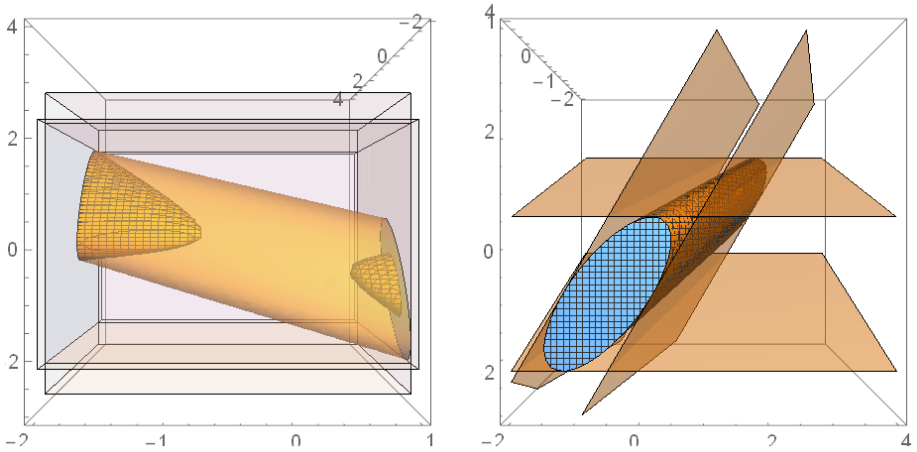

Fig. 3 Example 3.

Dropping one of the squared terms, dividing by 10 and 2.5, respectively, and taking the square root gives the constraints

$$x_2 - 0.5x_3 + 0.4x_1 \in [-0.4, 1],$$

$$x_3 + 0.6x_1 \in [-1.6, 1.2].$$

This provides a linear relaxation of the parabolic constraint that is nearly perfect, though it is less good considered as a relaxation of the original quadratic constraint. Performing quadratic constraint propagation on the new constraint leads to the finite bounds $x_2 \in \mathbf{x}_2 := [-1.9, 3]$ and $x_3 \in \mathbf{x}_3 := [-2.2, 2.4]$ for the previously unbounded variables x_2 and x_3 (see Figure 3). For comparison, applying MATHEMATICA with

```
Reduce[Exists[{xi, xj}, f <= 0 && x1 >= -2 && x1 <= 1], Reals]
```

(where f is given by (6) and {`xi, xj`} is one of {`x1, x2`} or {`x1, x3`} or {`x2, x3`}) results in the optimal interval enclosure

$$x_1 \in [-2, -0.753042] \cup [0.619709, 1],$$
$$x_2 \in [-0.947214, -0.0527864] \cup [0.545851, 2.58995],$$
$$x_3 \in [-1.4324, 2.4].$$

3 Convex quadratic relaxations

In this section we generalize the results of Domes & Neumaier [3] for finding the tightest box containing an ellipsoid to nonellipsoidal quadrics. Non ellipsoidal quadrics define unbounded sets, hence their interval hull usually has only infinite bounds. To obtain a useful problem we therefore consider the problem of bounding the intersection of a quadric and a possibly unbounded box. This may lead to useful results even in case of long and thin ellipsoids, where any enclosing box is necessarily huge, but the intersection even with infinitely long boxes may be quite small.

3.1 Exploiting general quadratic constraints

Our main result is the following theorem.

Theorem 1 *Let $\mathbf{x} \in \overline{\mathbb{IR}}^n$ be a box, and (M, N) a partition of the index set $\{1:n\}$. For $A \in \mathbb{R}^{n\times n}$, $a \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$, we consider the quadratic inequality*

$$x^T A x + 2a^T x \le \alpha. \tag{7}$$

Then for any nonsingular matrix $R_{MM} \in \mathbb{R}^{|M|\times|M|}$ such that

$$\Delta := A_{MM} - R_{MM}^T R_{MM} \tag{8}$$

is positive semidefinite, and with

$$R_{MN} := R_{MM}^{-T} A_{MN}, \quad B := R_{MN}^T R_{MN} - A_{NN}, \tag{9}$$

$$b_M := R_{MM}^{-T} a_M, \quad b_N := R_{MN}^T b_M - a_N, \tag{10}$$

$$E \in \mathbb{R}^{|M|\times n}, \quad E_{:M} = R_{MM}, \quad E_{:N} = R_{MN}, \tag{11}$$

the following statements hold:
(i) The quadratic inequality (7) with $x \in \mathbf{x}$ implies the parabolic relaxation

$$\|Ex + b_M\|_2^2 \le \gamma := \sup_{x_N \in \mathbf{x}_N} \Big(\alpha + \|b_M\|_2^2 + 2b_N^T x_N + x_N^T B x_N\Big). \tag{12}$$

(ii) If $\gamma < 0$ then (7) is violated for all $x \in \mathbf{x}$. If $\gamma \ge 0$ and $\mathbf{w}_M \in \mathbb{IR}^{|M|}$ is defined by

$$\mathbf{w}_i = [-\sqrt{\gamma}, \sqrt{\gamma}] - b_i \quad \text{for } i \in M$$

then

$$Ex \in \mathbf{w}, \tag{13}$$

is for all $x \in \mathbf{x}$ a linear relaxation of (7).

(iii) If $\gamma \geq 0$ then for all $x \in \mathbf{x}$

$$\|R_{MM}(x_M - z_M)\|_2 \leq \delta := \sup\left(\sqrt{\gamma} + \|R_{MM}z_M + R_{MN}\mathbf{x}_N + b_M\|_2\right) \tag{14}$$

is an ellipsoidal relaxation of (7).

Proof From (11) we find that

$$Ex = R_{MM}x_M + R_{MN}x_N. \tag{15}$$

Since A is symmetric, (8), (9) and (15) imply

$$\begin{aligned} x^TAx &= x_M^TA_{NN}x_M + 2x_M^TA_{MN}x_N + x_N^TA_{NN}x_N \\ &= x_M^T\Delta x_M + x_M^TR_{MM}^TR_{MM}x_M + 2x_M^TR_{MM}^TR_{MN}x_N + x_N^TA_{NN}x_N \\ &\geq (x_M^TR_{MM}^TR_{MM}x_M + 2x_M^TR_{MM}^TR_{MN}x_N) + x_N^TA_{NN}x_N \\ &= \|R_{MM}x_M + R_{MN}x_N\|_2^2 - \|R_{MN}x_Nx_N\|_2^2 + x_N^TA_{NN}x_N \\ &= \|Ex\|_2^2 - x_N^TBx_N. \end{aligned} \tag{16}$$

Now (16) and (10) give

$$\begin{aligned} \alpha &\geq x^TAx + 2a^Tx \\ &\geq \|Ex\|_2^2 - x_N^TBx_N + 2a_N^Tx_N + 2a_M^Tx_M \\ &= \|Ex\|_2^2 - x_N^TBx_N + 2a_N^Tx_N + 2b_M^TR_{MM}x_M \\ &= \|Ex + b_M\|_2^2 - x_N^TBx_N + 2a_N^Tx_N - 2b_M^TR_{MN}x_N - \|b_M\|_2^2, \\ &= \|Ex + b_M\|_2^2 - x_N^TBx_N - 2(R_{MN}^Tb_M - a_N)^Tx_N - \|b_M\|_2^2, \\ &= \|Ex + b_M\|_2^2 - x_N^TBx_N - 2b_N^Tx_N - \|b_M\|_2^2, \end{aligned}$$

and

$$\|Ex + b_M\|_2^2 \leq \alpha + \|b_M\|_2^2 + 2b_N^Tx_N + x_N^TBx_N \leq \gamma.$$

Therefore the norm inequality (12) is for all $x \in \mathbf{x}$ a relaxation of (7), proving (i).

If $\gamma \geq 0$ define $y := Ex + b_M$, then by (12) we have

$$y_i^2 \leq \sum_{i=1}^{n} y_i^2 \leq \gamma.$$

Therefore

$$y_i \in \mathbf{s} := [-\sqrt{\gamma}, \sqrt{\gamma}], \quad (Ex)_i \in \mathbf{s} - b_i,$$

proving (ii).

Statement (iii) follows from (12) and (15) since

$$\begin{aligned} \sqrt{\gamma} &\geq \|Ex + b_M\|_2 \\ &= \|R_{MM}x_M + R_{MN}x_N + b_M\|_2 \\ &= \|R_{MM}(x_M - z_M) + R_{MM}z_M + R_{MN}x_N + b_M\|_2 \\ &\geq \|R_{MM}(x_M - z_M)\|_2 - \|R_{MM}z_M + R_{MN}x_N + b_M\|_2. \end{aligned}$$

The ellipsoidal relaxation (14) can be turned into a bounding box using the following assertion, proved in [3, Proposition 1.1].

Proposition 1 *Let $B, C \in \mathbb{R}^{n\times n}$, $\beta \in \mathbb{R}$. If $d, h \in \mathbb{R}^n$ satisfy*

$$\sqrt{(CC^T)_{ii}} \le d_i \quad (i = 1, \ldots, n), \tag{17}$$

$$\langle CB\rangle d \le h, \tag{18}$$

$$\max\{h_i/d_i \mid i = 1, \ldots, n\} \le \beta, \tag{19}$$

Then the norm constraint

$$\|By\|_2 \le \delta \tag{20}$$

implies

$$|y| \le \overline{y} := \frac{\delta}{\beta} d, \quad y \in \mathbf{y} := [-\overline{y}, \overline{y}]. \tag{21}$$

Moreover, if (17)–(19) *hold with equality then* $\mathbf{y}$ *defined in* (21) *is the interval hull, the tightest box covering all y satisfying* (20)*.*

Thus if $\gamma \ge 0$ in Theorem 1, we determine the box $\mathbf{y}$ by applying Proposition 1 to $\|By\|_2 \le \delta$ with $B := R_{MM}$, $y := x_M - z_M$ and δ from (14). Therefore we have $x_M = z_M + y \in z_M + \mathbf{y}$, so that all $x \in \mathbf{x}$ with (7) are in the box $\widehat{\mathbf{x}}$ defined by

$$\widehat{\mathbf{x}}_N := \mathbf{x}_N, \quad \widehat{\mathbf{x}}_M := (z_M + \mathbf{y}) \cap \mathbf{x}_M. \tag{22}$$

3.2 Exploiting general nonlinear constraints

The preceding may be useful even for general nonlinear constraints

$$f(x) \in \mathbf{b} \tag{23}$$

if a quadratic relaxation for them is available. A variety of ways suggest themselves to find quadratic relaxations; they differ in their computational complexity and their approximation properties. This will be explored in depth in another publication.

Here we only give a simple recipe for quadratic relaxation, applicable whenever the constraint is twice continuously differentiable and all variables occurring in nonlinear terms are bounded. By splitting off the variables appearing only linearly we may assume w.l.o.g. that $\mathbf{x}$ is bounded. Then we may calculate rigorous enclosures

$$f := f(\underline{x}) \in \mathbf{f} \in \mathbb{IR}, \quad g := f[\underline{x}, \overline{x}] \in \mathbf{g} \in \mathbb{IR}^n,$$

$$S := f[\underline{x}, \overline{x}, x] \in \mathbf{S} \in \mathbb{IR}^{n\times n} \text{ for all } x \in \mathbf{x}$$

for the function value $f(\underline{x})$, the slope $f[\underline{x}, \overline{x}]$, and the second order slope $f[\underline{x}, \overline{x}, x]$ (see e.g., [15] for details). Using the defining properties of these slopes we find that

$$f(x) = f + (x - \underline{x})^T(g + S(x - \overline{x})) \ge \underline{q}(f) := \underline{f} + (x - \underline{x})^T(\underline{g} + \overline{S}(x - \overline{x})),$$

$$f(x) = f + (x - \underline{x})^T(g + S(x - \overline{x})) \le \overline{q}(f) := \overline{f} + (x - \underline{x})^T(\overline{g} + \underline{S}(x - \overline{x})).$$

Thus the constraint (23) has the quadratic relaxation

$$\underline{q}(f) \le \overline{b}, \quad -\overline{q}(f) \le -\underline{b}.$$

Thus our previous result can be applied to get convex quadratic relaxations. (If one of the bounds is infinite, the corresponding relaxation is vacuous and can be deleted.)

4 Generalized directed Cholesky factorization

To make Theorem 1 applicable to arbitrary quadratic inequality constraints (7) we need to find good matrices R_{MM} satisfying the requirement that the residual matrix Δ defined in (8) is positive semidefinite. Here "good" means that Δ should have tiny entries. To achieve this if possible we factor the symmetric A by a method called the directed generalized directed Cholesky factorization.

A **directed Cholesky factorization** of a symmetric interval matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ constructs an upper triangular matrix $R' \in \mathbb{R}^{n\times n}$ and a permutation matrix P such that $PAP^T - R'^T R'$ is positive semidefinite for all $A \in \mathbf{A}$. If we introduce the matrix $R := R'P$ (which is in general no longer upper triangular), the matrix $\Gamma := A - R^T R$ is positive semidefinite for all $A \in \mathbf{A}$. Algorithm `DirCholP` of Domes & Neumaier [3, Algorithm 5.5] finds such an R if (numerically) $\mathbf{A}$ is positive definite. If the directed Cholesky factorization fails, the partial factorization obtained satisfies at least some of the properties of the full factorization. In addition, one may in this case adjust the diagonal entries $\mathbf{A}$ such that it becomes positive definite, and a modified directed Cholesky factorization is obtained. In the following, we modify algorithm `DirCholP` such that it

- either computes a directed Cholesky factor R and a permutation matrix P such that the residual matrix Γ is positive semidefinite and very small with respect to A,
- or terminates with an error message and returns an incomplete factorization.
- or adjusts the negative diagonal elements during the factorization and stores the adjustments in a diagonal matrix D such that the factorization is successful and $\Gamma' := A + D - R^T R$ is positive semidefinite and very small with respect to $A + D$.

Since sometimes theory requires that a submatrix of $\mathbf{A}$ is positive definite (see Domes & Neumaier [4]), we also want to make sure that specific diagonal elements are selected first as pivots. For this reason we take as additional input an index list M and choose in the first $|M|$ steps only pivots α_k with $k \in M$.

4.1 Improved handling of nearly singular matrices

The method given in Domes & Neumaier [3] for selecting the parameter γ_k in line 8 of Algorithm 1 proved to have some limitations when a submatrix is nearly singular.

The improvement discussed here is based on the expectation (checked in numerical experiments to be typically valid for a suitable tolerance, e.g., $\kappa = 10^{-6}$) that each Γ^k is very small with respect to A. Therefore we choose ρ_k, r_k and γ_k in Algorithm 1 as follows:

- To make Γ positive semidefinite, we have to ensure that $\varepsilon > 0$. Therefore we need $\delta_k > 0$, which is the case when, $|\rho_k| < \sqrt{\underline{\alpha}_k}$. If we also want δ_k to be very small and assume that $\underline{\alpha}_k > 0$ (which is true if $\underline{A}$ is positive definite), we can set $\rho_k = \gamma_k \sqrt{\underline{\alpha}_k}$ with $\gamma_k < 1$. If in addition to this we choose $\gamma_k \approx 1$, the condition $\delta_k \approx 0$ is also satisfied.

- The entries of $d_k = a_k - \rho_k r_k$ can be made to vanish by setting $r_k := a_k/\rho_k$. Even when r_k and ρ_k are computed inaccurately, we can get a very small d_k by setting $r_k = \widetilde{a}_k/(2\rho_k)$ where $\widetilde{a}_k := \overline{a}_k + \underline{a}_k$.
- To make $d_k^T d_k/\delta_k$ very small, we also have to guarantee that $d_k^T d_k \ll \delta_k$. Due to rounding errors, $d_k^T d_k$ is of order

$$\widetilde{d}_k := |\overline{a}_k - \underline{a}_k| + \varepsilon|\widetilde{a}_k|,$$

where ϵ is the machine precision, so we want $1 \gg \delta_k = \alpha_k - \gamma_k^2\alpha_k \gg \widetilde{a}_k$.

In order to achieve these goals in each step we need choose a suitable γ_k such that the diagonal elements of $\underline{A}_k$ are likely to remain positive. Writing $\mu_k := \gamma_k^{-2}$, we must ensure that the diagonal of

$$\underline{A}_k = \underline{B}_k - r_k r_k^T - d_k d_k^t/\delta_k \approx \underline{B}_k - \frac{\mu_k}{4\underline{\alpha}_k}\Big(\widetilde{a}_k\widetilde{a}_k^T + \frac{\widetilde{d}_k\widetilde{d}_k^T}{\mu_k - 1}\Big)$$

is not so small. If $\widetilde{a}_k^T\widetilde{a}_k = 0$ then $\widetilde{a}_k = \widetilde{d}_k = 0$ and we may choose $\gamma_k = 1$. Otherwise we note that the trace is maximal for

$$\mu_k = 1 + \sqrt{\frac{\widetilde{d}_k^T\widetilde{d}_k}{\widetilde{a}_k\widetilde{a}_k^T}}.$$

Thus we might choose $\gamma_k = 1/\sqrt{\mu_k}$; but in order to avoid that γ_k gets small, we safeguard it by

$$\gamma_k := \begin{cases} 1/\min(2,\sqrt{\mu_k}) & \text{if } \widetilde{a}_k^T\widetilde{a}_k \neq 0, \\ 1 & \text{otherwise.} \end{cases}$$

Using these choices in Algorithm 1 (described below) makes the residual matrix not only positive semidefinite but also very small with respect to A for all $A \in \mathbf{A}$.

Theorem 5.6 of Domes & Neumaier [3] (which details the properties of the factorization resulting from the old algorithm) still holds with trivial modifications for the resulting Algorithm 1. In case the factorization failed after k steps, the incomplete factorization computed by the new algorithm still has the property that $\Gamma^k := (PAP^T)_{KK} - R_{KK}^T R_{KK}$ is positive semidefinite for $K := \{1,\dots,k\}$. This is particularly useful in case both $M \neq \emptyset$, $R^m \neq \emptyset$, and we need the fact that

$$\Gamma^m := (PAP^T)_{MM} - (R^m)^T R^m \tag{24}$$

is positive semidefinite.

4.2 **Improved pivot choosing strategy**

The pivots must be chosen to avoid an excessive growth of elements in the Schur complement. For a positive, finite variable scaling vector w, whose components may be computed, e.g., by

$$w_i := \max(\min(\overline{x}_i - \underline{x}_i, 10^4), 10^{-8})$$

the value of

$$c := w_N^T p \text{ where } p := |\mathbf{A}_{NN}|w_N, \tag{25}$$

is monitored, where $\mathbf{A}_{NN}$ is the current Schur complement to be factored. Upon choosing the pivot index i, $w_N^T|\mathbf{A}_{NN}|w_N$ changes by at most

$$\sum_{j,k\in N} w_j \Big|\frac{\mathbf{A}_{ji}\mathbf{A}_{ki}}{\mathbf{A}_{ii}}\Big| w_k = \frac{(|\mathbf{A}_{NN}|w_N)_i^2}{\underline{A}_{ii}}.$$

Since $\langle \mathbf{A}_{ii}\rangle = \underline{A}_{ii}$ for positive definite $\mathbf{A}$, it is sensible to choose as pivot the index i maximizing

$$q_i := \begin{cases} \frac{c\underline{A}_{ii}}{p_i^2} & \text{if } p_i \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

Note that by (26) the value of q_i is finite even if $p_i = 0$.

4.3 **Modified directed Cholesky factorization**

It may happen during the pivot search that after computing (26) we find that $\max q_i < \delta$ for a fixed, small positive constant δ; the value $\delta = 0.01$ works well.

In this case, we either quit and return an incomplete factorization, or we continue the algorithm and produce a modified directed Cholesky factorization in the following sense.

Definition 3 A directed modified Cholesky factorization of a symmetric interval matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, consists of a nonsingular matrix $R \in \mathbb{R}^{n\times n}$ and a non-negative diagonal matrix $D \in \mathbb{R}^{n\times n}$ such that the residual matrix

$$\Gamma := A + D - R^T R \tag{27}$$

is positive semidefinite for all $A \in \mathbf{A}$.

Suppose that in the kth step of the algorithm, $\mathbf{A}_{NN}$ is the current Schur complement to be factored, P is the permutation matrix after the $(k-1)$th the step, i denotes the selected pivot index, and $j := P_{ii}$. in view of the above pivoting strategy, the corrected pivot α' must be given by

$$\mathbf{A}'_{ii} = \mathbf{A}_{ii} + D_{jj}, \quad D_{jj} := \begin{cases} \frac{\max(\delta p_i^2, \epsilon c)}{c} - \underline{A}_{ii} & \text{if } c \neq 0, \\ \epsilon & \text{otherwise,} \end{cases} \tag{28}$$

where the constant c and the vector p is defined as in (25).

If for the pivot index i chosen, the value of q_i from (26) is saved when a modified Cholesky factorization is computed, one may still obtain the partition relevant for the partial Cholesky factorization by truncating it at the first index i in the permutation vector for which $q_i = \delta$. Algorithm 1 describes the resulting generalized directed Cholesky factorization in more precise way.

Algorithm 1: Generalized directed Cholesky factorization (GDChol)

Input: A symmetric interval matrix $\mathbf{A} \in \mathbb{IR}^{n\times n}$, the index list M (which can be empty) with $M \subseteq \{1,\dots,n\}$ and $|M| = m$, the factorization `mode` which is `inc` for the incomplete and `mod` for the modified factorization, the variable scaling vector $w \in \mathbb{R}^n$ and the diagonal modification factor δ (use $\delta = 0.01$ as default)

Success: The upper triangular matrix $R \in \mathbb{R}^{n\times n}$ and a permutation matrix $P \in \mathbb{R}^{n\times n}$ and the diagonal matrix $D \in \mathbb{R}^{n\times n}$ such that $P(A+D)P^T - R^TR$ is positive semidefinite for all symmetric $A \in \mathbf{A}$. The matrix D is always the zero matrix for `mode=inc`.

Incomplete: In case `mode=inc`, the complete factorization failed but the first $m \geq |M|$ steps were successful, return the matrix $\mathbf{A}^m \in \mathbb{IR}^{(n-m)\times(n-m)}$, P and $R^m \in \mathbb{R}^{m\times m}$

Failure: If during the factorization of $\mathbf{A}_{NN}$ a negative pivot is found.

Put $\mathbf{A}_1 = \mathbf{A}$, $N = M$, $\mathbf{A}^m = \emptyset$, $R = 0_n$, $R^m = \emptyset$, $P = I_n$, $D = 0_n$ and change to upward rounding mode;
for $k = 1,\dots,n$ **do**
  Put $S := N$ if $N \neq \emptyset$ and $S := \{k,\dots,n\}$ otherwise;
  Compute $c := |\mathbf{A}_{SS}|w_S \in \mathbb{R}^{|S|}$ and $d := w_S^T c$;
  Compute $q \in \mathbb{R}^{|S|}$ with $q_i := (d\underline{A}_{ii})/c_i^2$;
  Find the (pivot) index j with $q_j \geq q_i$ for all $i \in S$;
  if $q_j < \delta$ **then**
    if $N = \emptyset$ *and* `mode=mod` **then**
      Put $t := P_{jj}$, $D_{tt} \leftarrow \delta c_j^2/d - \underline{A}_{jj}$ and $\mathbf{A}_{jj} \leftarrow \mathbf{A}_{jj} + D_{tt}$.
    else return $\mathbf{A}^m$, P, R^m *and an error message*;
  end
  If $S = N$ remove the pivot index j from the index list N;
  Exchange the jth row with the first row and jth column with the first column of $\mathbf{A}_k$. Exchange the same rows and columns in the matrix P;
  Partition the permuted interval matrix $\mathbf{A}_k$ as:

$$\mathbf{A}_k = \begin{pmatrix} \boldsymbol{\alpha}_k & \mathbf{a}_k^T \\ \mathbf{a}_k & \mathbf{B}_k \end{pmatrix}$$

  Choose $0 < \gamma_k < 1$, $\rho_k = \gamma_k\sqrt{\underline{\alpha}_k}$ and $r_k = (\overline{a}_k + \underline{a}_k)/(2\rho_k)$;
  Set $R_{kk} = \rho_k$, $R_{k,k:n} = r_k^T$ and compute $\delta_k := -(-\underline{\alpha}_k + \rho_k^2)$, $d_k := \max(\overline{a}_k + \rho_k(-r_k),\ \rho_k r_k - \underline{a}_k)$;
  if *the residual pivot* $\delta_k \leq 0$ **then**
    return $\mathbf{A}^m$, P, R^m *and an error message*;
  else Set $\mathbf{A}_{k+1} := [\underline{B}_k - r_k r_k^T - d_k d_k^T/\delta_k,\ \overline{B}_k + (-r_k)r_k^T + d_k d_k^T/\delta_k]$;
  if $M \neq \emptyset$ *and* $k \geq |M|$ **then** put $\mathbf{A}^m = \mathbf{A}_k$ and $R^m := R_{1:k,1:k}$;
end
return *The matrix R, the diagonal matrix D and the permutation matrix P*

5 Enclosing uncertain quadratic inequalities

In this section we compute rigorous parabolic, elliptical (strictly convex) and linear relaxations for the uncertain quadratic inequality (1) with $\mathbf{x} \in \overline{\mathbb{IR}}^n$, as well as a box $\mathbf{x}' \subseteq \mathbf{x}$ such that each $x \in \mathbf{x}$ satisfying (1) is contained in the box $\mathbf{x}'$ (hence the enclosure is rigorous). We assume that $\mathbf{A}$ does not contain zero rows; this means that each variables occurs nonlinearly in (1). The case where some variables enter the constraint only linearly will be discussed separately in Section 6.1.

Since most interval linear algebra computations become vacuous when some of the intervals are unbounded we need to treat bounded and unbounded variables

differently. We therefore partition $\{1,\ldots,n\}$ into two ordered index lists

$$N_0 := \{i \mid \mathbf{x}_i \text{ is bounded}\}, \quad M_0 := \{1,\ldots n\} \setminus N_0 \tag{29}$$

indexing the bounded and unbounded variables, respectively. We use the generalized directed Cholesky factorization (Algorithm 1) in incomplete factorization mode on $\mathbf{A}$ and the index list M_0. If the factorization is successful (1) is convex and we can apply [3, Section 2]. This case is discussed in the Subsection 5.1. Subsection 5.2 considers the case where the generalized directed Cholesky factorization is used in the modified factorization mode. Finally, if the generalized directed Cholesky factorization failed, but successfully completed the first $|M_0|$ steps, the incomplete factorization can still be used to produce various relaxations and bounds for (1). This case is discussed in the Subsection 5.3. If not even $\mathbf{A}_{M_0 M_0}$ can be factorized, all methods presented in this section will fail.

5.1 **Convex quadratic relaxation using a full factorization**

If the directed Cholesky factorization was successful then (1) is strictly convex. The quadratic relaxation can be written as the norm constraint

$$\|Rx\|_2^2 + 2a^T x \le \alpha, \quad A \in \mathbf{A}, \quad a \in \mathbf{a}, \tag{30}$$

and the results of [3, Section 2] apply: Let C be an approximate inverse of the matrix R and compute

$$\widetilde{z} = C^T \operatorname{mid}(\mathbf{a}), \;\; \widetilde{x} = -C\widetilde{z} \tag{31}$$

by floating point calculations and

$$\begin{aligned} &d \in \mathbb{R}^n, \quad d_i = \inf(\sqrt{(CC^T)_{ii}}), \quad h = \langle CR \rangle d, \\ &\beta = \max\{h_i/d_i \mid i = 1,\ldots,n\} \approx 1, \quad u := \beta^{-1} d, \end{aligned} \tag{32}$$

using directed rounding as well as

$$\boldsymbol{\gamma} := \|\widetilde{z} + R\widetilde{x}\|_2 + u^T |\mathbf{a} - R^T \widetilde{z}| \tag{33}$$

and

$$\boldsymbol{\Delta} := \boldsymbol{\gamma}^2 + \alpha - 2\mathbf{a}^T \widetilde{x} - \|R\widetilde{x}\|_2^2, \tag{34}$$

using interval arithmetic. If $\underline{\Delta} \ge 0$ then by [3, Theorem 1.5 and Corollary 1.6], all $x \in \mathbf{x}$ satisfying (30) also satisfies

$$\|R(x - \widetilde{x})\|_2 \le \delta := \overline{\gamma} + \sqrt{\overline{\Delta}}. \tag{35}$$

Therefore (35) is an ellipsoid relaxation of both (30) and (1). By the same theorem we also obtain the box

$$\widehat{\mathbf{x}} := [\widetilde{x} - \delta u, \widetilde{x} + \delta u], \tag{36}$$

enclosing the solution set of (30).

5.2 **Convex quadratic relaxation using a modified factorization**

If the generalized modified Cholesky factorization presented in Section 4 is successfully applied to $\mathbf{A}$ and M in modified factorization mode, a nonsingular matrix R and a diagonal matrix $D \geq 0$ are obtained such that

$$A = R^T R - D + \Gamma \text{ for all } A \in \mathbf{A} \text{ and } D_{MM} = 0, \tag{37}$$

and the residual matrix Γ is positive semidefinite and very small with respect to $\mathbf{A} - D$ (details in Section 4).

The following result shows how (37) can be used to create from this information a convex quadratic relaxation and a box enclosure. (Of course, if $D = 0$ then the factorization was successful without needing any modification of the diagonal, and the method based on the full factorization from Section 5.1 can also be applied.)

Proposition 2 *For $x \in \mathbf{x}$ consider the uncertain quadratic inequality* (1)*. Let $M \in \{1, \ldots, n\}$ as in* (43)*, R and D be the result of the generalized Cholesky factorization given in Algorithm 1, with* `mode=mod`*, applied to $\mathbf{A}$ and M. Let for the norm inequality*

$$\|Rx\|_2^2 + 2a^T x \leq \widehat{\alpha},\ x \in \mathbf{x},\ A \in \mathbf{A},\ a \in \mathbf{a} \tag{38}$$

with

$$\widehat{\alpha} := \alpha + \sup \sum_{i \in \neg M} D_{ii} \mathbf{x}_i^2. \tag{39}$$

the box $\widehat{\mathbf{x}}$ be computed by (36) *using $\widehat{\alpha}$ in place of α, then all $x \in \mathbf{x}$ satisfying* (1) *are also contained in the box*

$$\mathbf{x}' := \mathbf{x} \cap \widehat{\mathbf{x}}. \tag{40}$$

Proof Since R and D the result of the directed modified Cholesky factorization given in Algorithm 1 applied to $\mathbf{A}$ and M, with `mode=mod` they must satisfy (37). Substituting (37) into (1) results in

$$\begin{aligned} x^T A x + 2a^T x &\leq x^T (R^T R - D + \Gamma) x + 2a^T x \\ &= \|Rx\|_2^2 - x^T D x + x^T \Gamma x + 2a^T x \leq \alpha, \end{aligned}$$

and using that $D \geq 0$ and the residual matrix Γ is positive semidefinite, we end up in

$$\|Rx\|_2^2 + 2a^T x \leq \alpha + x^T D x - x^T \Gamma x \leq \alpha + x^T D x \leq \widehat{\alpha}, \tag{41}$$

with

$$\widehat{\alpha} := \alpha + \sup_{x \in \mathbf{x}} x^T D x = \alpha + \sup \sum_{i \in \neg M} D_{ii} \mathbf{x}_i^2, \tag{42}$$

for all $x \in \mathbf{x}$, $A \in \mathbf{A}$ and $a \in \mathbf{a}$, which is exactly (30). This proves that the solution set of (1) is fully contained in the ellipsoid given by the norm constraint (30). If we apply Proposition 1 to the norm constraint (38) we obtain the convex quadratic relaxation (35) of (1) as well as the box $\widehat{\mathbf{x}}$. Therefore for all $x \in \mathbf{x}$ satisfying (1) $x \in \mathbf{x}' := \mathbf{x} \cap \widehat{\mathbf{x}}$.

Since in this case $\widehat{\alpha} = \alpha$ does not depend of the box $\mathbf{x}$ and Γ is very small with respect to A, the relative approximation error

$$\delta(x) := \frac{x^T \Gamma x}{\|Rx\|_2^2},$$

is also small. Therefore if (1) is strictly convex, (39) is a nearly optimal approximation. On the other hand if $D \neq 0$, by (43) the bound (39) has to be finite; so (30) is a nontrivial inequality.

Proposition 2 forms the basis of the rigorous method described by Algorithm 2.

Algorithm 2: Convex quadratic relaxations using the generalized Cholesky factorization in modified mode (QRelGDCholM)

Input: The uncertain quadratic inequality (1) with $x \in \mathbf{x} \in \overline{\mathbb{IR}}^n$.
Output: A convex quadratic relaxation and the rigorous box enclosure $x \in \mathbf{x}'$.

Compute M_0 by (29) and use the directed modified Cholesky factorization (Algorithm 1) on $\mathbf{A}$, M, with `mode=mod` to obtain D and R;
if $\mathbf{A}_{MM}$ *could be factorized without modifying the diagonal* **then**
  return *signaling failure*
end
else if D *is the zero matrix* **then**
  The constraint is convex and proceed as [3, Algorithm 2.1,(2)-(7)] and **return** *the ellipsoidal relaxation, given by the norm constraint* (35) *and the rigorous box enclosure* (36);
else
  Compute $\widehat{\alpha}$ by (39), using interval arithmetic;
  Compute the approximative inverse C of the matrix R;
  Compute d with $d_i = \inf(\sqrt{(CC^T)_{ii}})$ by using directed rounding;
  Use upward rounding to compute the values $h = \langle CR \rangle d$ and $\beta = \max\{h_i/d_i \mid i = 1 \dots n\} \approx 1$;
  Set $\widetilde{z} = C^T a$ and $\widetilde{x} = -C\widetilde{z}$ and compute an enclosure $[\underline{\gamma}, \overline{\gamma}]$ for (33), an enclosure $[\underline{\Delta}, \overline{\Delta}]$ for (34) using interval arithmetic;
  if $\underline{\Delta} < 0$ **then return** *signaling failure*;
  else
    Compute the interval $[\underline{\delta}, \overline{\delta}]$ from (35) by using outward rounding;
    return *The convex quadratic relaxation, given by the norm constraint* (36) *and the rigorous box enclosure* (40)
  end
end

5.3 Quadratic relaxations using an incomplete factorization

If not all $A \in \mathbf{A}$ are positive definite or some of them are too close to singularity, the generalized directed Cholesky factorization will fail. However ,if the factorization successfully passed the first $|M_0|$ steps, the index lists M and N for the factored and unfactored part, respectively, satisfy

$$M \supseteq M_0, \quad N := (\{1, \dots n\} \setminus M) \subseteq N_0. \tag{43}$$

In this case for the matrix $\mathbf{A}_{MM}$ the incomplete factorization should result in the index list M and the permuted upper triangular matrix R_{MM}. By Section 4 we know that the residual matrix

$$\Delta := A_{MM} - R_{MM}^T R_{MM}, \tag{44}$$

is positive semidefinite and tiny compared to the entries of $\mathbf{A}_{MM}$. In this we may compute

$$\begin{aligned} &C \approx R_{MM}^{-1}, \quad \mathbf{R}_{MN} := C^T \mathbf{A}_{MN}, \quad \mathbf{B} := \mathbf{R}_{MN}^T \mathbf{R}_{MN} - \mathbf{A}_{NN}, \\ &\mathbf{b}_M := C^T \mathbf{a}_M, \, \mathbf{b}_N := \mathbf{R}_{MN}^T \mathbf{b}_M - \mathbf{a}_N, \\ &\mathbf{E} \in \mathbb{IR}^{|M| \times n}, \;\; \mathbf{E}_{:M} = R_{MM}, \qquad \mathbf{E}_{:N} = \mathbf{R}_{MN}. \end{aligned} \tag{45}$$

Then conditions (9)–(11) of Theorem 1(i) are satisfied and for all $x \in \mathbf{x}$ the inequality

$$\|Ex + b_M\|_2^2 \le \gamma \text{ for all } E \in \mathbf{E}, \; b_M \in \mathbf{b}_M, \tag{46}$$

with

$$\gamma := \sup_{x_N \in \mathbf{x}_N} \Big(\alpha + \|\mathbf{b}_M\|_2^2 + 2\mathbf{b}_N^T x_N + x_N^T \mathbf{B} x_N \Big), \tag{47}$$

is a parabolic relaxation of (1). An upper bound for γ suffices which can be efficiently computed by taking the upper bound of

$$\alpha + \|\mathbf{b}_M\|_2^2 + \sum_{\substack{i \in N \\ i \ne j}} \sum_{j \in N} \mathbf{B}_{ij} \mathbf{x}_i \mathbf{x}_j + \sum_{i \in N} \sup_{x_i \in \mathbf{x}_i} (\mathbf{B}_{ii} x_i^2 + 2\mathbf{b}_i x_i), \tag{48}$$

where the exact range of the univariate quadratic expressions in the parenthesis is bounded (apart from rounding errors without overestimation) by the method presented in [2].

If $\gamma < 0$ then by (ii) of Theorem 1, (7) is inconsistent and, e.g., in a branch and bound scheme the box $\mathbf{x}$ can be rejected. On the other hand, if $\gamma \ge 0$ we may compute

$$\mathbf{u} := \sqrt{\gamma},$$

to obtain for all $x \in \mathbf{x}$ the interval linear relaxation

$$Ex \in \mathbf{w}, \; E \in \mathbf{E}, \quad \mathbf{w}_i := [-\overline{u}, \overline{u}] - (\mathbf{b}_M)_i. \tag{49}$$

of (1). Since the term

$$\|R_{MM} z_M + R_{MN} \mathbf{x}_N + b_M\|_2 \approx \|R_{MN}(\mathbf{x}_N - \operatorname{mid}(\mathbf{x}_N))\|_2$$

is expected to be small, the approximation

$$z_M \approx -C \operatorname{mid}(\mathbf{R}_{MN} \mathbf{x}_N + \mathbf{b}_M), \tag{50}$$

is a good choice, and Theorem 1(iii) gives the ellipsoidal relaxation

$$\|R_{MM}(x_M - z_M)\|_2^2 \le \delta \tag{51}$$

with

$$\delta := \sup \Big(\mathbf{u} + \|R_{MM} z_M + \mathbf{R}_{MN} \mathbf{x}_N + \mathbf{b}_M\|_2 \Big)^2. \tag{52}$$

Finally Proposition 1 with $B := R_{NN}$, $y := x_M - z_M$, and δ can be used to obtain the box $\mathbf{y}$ for which the bound (22) holds.

The following algorithm summarizes the method described by this section.

Algorithm 3: Quadratic relaxations using the generalized modified Cholesky factorization (QRelGDChol)

Input: The uncertain quadratic inequality (1) with $x \in \mathbf{x} \in \overline{\mathbb{IR}}^n$.
Output: The parabolic relaxation (46), the ellipsoidal relaxation (51), the linear relaxation (49) and the rigorous box enclosure $x \in \mathbf{x}' \subseteq \mathbf{x}$

Compute M_0 by (29) and use the generalized directed Cholesky factorization (Algorithm 1) on $\mathbf{A}$, M_0 with `mode=inc`;
if *the factorization failed but* $R^m \neq \emptyset$ **then**
  We have obtained M, P and R^m and put $R_{MM} \leftarrow R^m P_{MM}$;
  Compute the approximative inverse C of the matrix R_{MM};
  Compute $\mathbf{R}_{MN}$, $\mathbf{B}$, $\mathbf{b}_M$, $\mathbf{b}_N$, $\mathbf{E}$ as given in (45) using interval arithmetic;
  Compute γ by taking the upper bound of (48) evaluated using interval arithmetic and by computing the exact range for the separable quadratic terms;
  if $\gamma < 0$ **then return** *signaling that the constraint is inconsistent*;
  else
    Compute $u := \sup(\sqrt{\gamma})$ using the interval square root function;
    Compute z_M by (50) and δ by (52) using interval arithmetic;
    Compute d with $d_i = \inf(\sqrt{(CC^T)_{ii}})$ by using directed rounding;
    Use upward rounding to compute the values $h = \langle CR_{MM} \rangle d$ and $\beta = \max\{h_i/d_i \mid i = 1 \dots n\} \approx 1$;
    Compute $r := (\delta/\beta)d$ using upward rounding to find $\mathbf{y} := [-r, r]$;
    Compute $\mathbf{x}'$ with $\mathbf{x}'_N = \mathbf{x}_N$ and $\mathbf{x}'_M := (\mathbf{y} + z_M) \cap \mathbf{x}_M$;
    return *The parabolic relaxation* (46)*, the ellipsoidal relaxation* (51)*, the linear relaxation* (49) *and the rigorous box enclosure* $x \in \mathbf{x}'$
  end
else if *the factorization was successful* **then**
  The constraint is convex and proceed as [3, Algorithm 2.1,(2)-(7)] and **return** *the ellipsoidal relaxation, given by the norm constraint* (35) *and the rigorous box enclosure* (36);
else return *signaling failure*;

6 A filtering method for uncertain quadratic optimization problems

In this section we extend and combine the methods presented in Domes & Neumaier [3], Domes & Neumaier [5], as well as the ones presented in Section 5 in order to filter uncertain quadratic optimization problems. First we discuss how purely linear terms can be removed, then we elaborate how the success rate of the method can be enhanced by performing simple diagonal sign tests and bound checking.

6.1 Removing purely linear terms

If some variables occur only linearly in (2) the corresponding rows and columns of $\mathbf{A}$ have only zero entries, therefore $\mathbf{A}$ is singular, and depending on the method computing the relaxation becomes impossible or at least inefficient. Therefore we define the index lists

$$J := \{j \mid \mathbf{A}_{jk} = 0, \quad \text{for all } k = 1, \dots, n\}, \quad K := \neg J, \tag{53}$$

and eliminate the variables $x_J \in \mathbf{x}_J \in \overline{\mathbb{IR}}^{|J|}$ from (2) obtaining

$$x_K^T A_{KK} x_K + 2a_K^T x_K \in \mathbf{d}', \ A_{KK} \in \mathbf{A}_{KK}, \ a_K \in \mathbf{a}_K, \tag{54}$$

with

$$\mathbf{d}' := \mathbf{d} - 2\mathbf{a}_J^T \mathbf{x}_J. \tag{55}$$

After applying a method to find an improved enclosure $x_K \in \mathbf{x}''_K \subseteq \mathbf{x}_K$ for all $x_K \in \mathbf{x}_K$ satisfying (54), the improved enclosure on the original variables x can be given as

$$x \in \mathbf{x}', \quad \mathbf{x}'_K := \mathbf{x}_K \cap \mathbf{x}''_K, \quad \mathbf{x}'_J := \mathbf{x}_J. \tag{56}$$

6.2 Diagonal sign test and bound checking

Assume we are given the two sided inequality (2) with $\mathbf{A}_{j:} \neq 0$ for all $j \in \{1, \ldots, n\}$ and an index list $S \subseteq \{1, \ldots, n\}$. Performing simple diagonal sign tests and checking the bound $\mathbf{d}$ allows to transform (2) into the form (1) such that it matches the input form for the previously discussed relaxation methods. These tests can also a priori indicate that the selected relaxation method can not be successful for the given two sided inequality, thereby saving computation time.

Definition 4 For an interval matrix $\mathbf{A} \in \mathbb{IR}^{n \times n}$ and an index list $S \subseteq \{1, \ldots, n\}$ with with $A_{ii} \neq 0$ for some $i \in K$, the matrix diagonal sign function $\text{sign}(\mathbf{A}, S) \in \{-1, 0, 1\}$ is defined as

$$\text{sign}(\mathbf{A}, S) := \begin{cases} -1 \text{ if } \overline{A}_{ii} \leq 0 \text{ for all } i \in S \\ 1 \text{ if } \underline{A}_{ii} \geq 0 \text{ for all } i \in S \\ 0 \text{ otherwise.} \end{cases} \tag{57}$$

Using the well known fact that if a matrix is positive (negative) definite the diagonal entries of the matrix are non-negative (non-postive), for the two sided inequality (2) we find:

- If the index list of the unbounded variables M_0 is non-empty, $\text{sign}(\mathbf{A}, M_0) = 1$ and $\overline{d} = \infty$ then none of the relaxation methods can be successful.
- Similarly, if M_0 is non-empty, $\text{sign}(\mathbf{A}, M_0) = -1$ and $\underline{d} = -\infty$ then none of the relaxation methods can be successful.
- If $\text{sign}(\mathbf{A}, S) = 0$ for any S, then the relaxation method using a full factorization can not be successful.
- If $\text{sign}(\mathbf{A}, S) = 1$ for any S and $\overline{d} \neq \infty$ then the two sided inequality can be replaced by the one sided inequality (1) with $\alpha = \overline{d}$.
- If $\text{sign}(\mathbf{A}, S) = -1$ for any S and $\underline{d} \neq -\infty$ then the two sided inequality can be replaced by the one sided inequality (1) with $\mathbf{A} \leftarrow -\mathbf{A}$, $\mathbf{a} \leftarrow -\mathbf{a}$ and $\alpha = -\underline{d}$.

These observations form an important part of the quadratic contractor algorithm discussed in the next section.

6.3 The QuadFilter algorithm

We now present the QuadFilter algorithm for filtering a box $\mathbf{x}$ of an uncertain quadratic optimization problem. In the solution process of uncertain quadratic optimization problems, this method can be used as the filtering part of a branch and bound framework. When applied to quadratic relaxations of non quadratic problems it can even be useful for solving general uncertain optimization problems.

Algorithm 4: Quadratic filter using quadratic relaxations (QuadFilter)

Input: An uncertain quadratic optimization problem (3), the starting box $\mathbf{x}$ the maximal number of iterations `maxiter` and the minimal step gain `mingain` and the contractor `type` chosen from {`EHull`, `QRelIDChol`, `QRelMDChol`, `QRelGDChol`, `QRelGDCholM`}

Output: A rigorous box enclosure $x \in \mathbf{x}' \subseteq \mathbf{x}$ of the feasible set.

Initialize `iter` $= 0$, `gain` $= 1$ and $\mathbf{x}' := \mathbf{x}$;
while *iter* $<$ *maxiter and gain* $\geq$ *mingain and* $\mathbf{x}' \neq \emptyset$ **do**
  Put $\mathbf{x}^0 \leftarrow \mathbf{x}'$;
  foreach *Quadratic constraint* `c` *of the uncertain optimization problem* **do**
    if `c` *can be written as* (2) *with* $\mathbf{A} \neq 0$ **then**
      Compute the index list J of the variables which occur only linearly in `c` by (53) and put $K := \{1, \ldots, n\} \setminus J$;
      Eliminate the variables x_J from `c` obtaining (54) where $\mathbf{d}'$ is computed by (55), using interval arithmetic;
      if *type=EHull* **then** put $M := K$;
      else
        Compute the index list M_0 of the unbounded variables by (29) and put $M \leftarrow M_0 \cap K$;
      end
      if $M \neq \emptyset$ **then** compute $s := \mathrm{sign}(\mathbf{A}, M)$ by (57);
      else compute $s := \mathrm{sign}(\mathbf{A}, K)$ by (57);
      if *(*$s = -1$ *and* $\underline{d}' = -\infty$*) or (*$s \neq -1$ *and* $\overline{d}' = \infty$*)* **then**
        Continue with the next constraint `c`;
      end
      if $s = -1$ **then** put $\mathbf{A} \leftarrow -\mathbf{A}$, $\mathbf{a} \leftarrow -\mathbf{a}$ and $\mathbf{d} \leftarrow -\mathbf{d}$;
      Depending on the given contractor `type` apply the corresponding contraction method to (54) using the (finite) upper bound of $\mathbf{d}'$ as the right hand side of the inequality;
      if *the contraction method was successful* **then**
        if *the problem has been found infeasible* **then** set $\mathbf{x}' = \emptyset$;
        else
          Use the obtained variable bounds $\mathbf{x}'_K$ to update $\mathbf{x}_i \leftarrow \mathbf{x}_i \cap \mathbf{x}'_i$ for all $i \in K$;
          If the bounds improved perform a few steps of quadratic constraint propagation on the original problem using the new bounds;
        end
      end
    end
  end
  Compute the gain factor `gain` from $\mathbf{x}^0$ and $\mathbf{x}'$;
  Put `iter` $\leftarrow$ `iter` $+ 1$;
end

Before starting the method one of the following relaxation methods has to be chosen:

- the ellipsoid hull (`EHull`) algorithm described in [3, Section 2, Algorithm 2.1],
- the convex quadratic relaxation algorithm using incomplete directed Cholesky factorization (`QRelIDChol`) described in [5, Section 3.2, Algorithm 1],
- the convex quadratic relaxation algorithm using directed modified Cholesky factorization (`QRelMDChol`) described in [5, Section 3.3, Algorithm 2],
- the quadratic relaxation algorithm using the generalized modified Cholesky factorization (`QRelGDChol`) from Section 5.3,

– the convex quadratic relaxation algorithm using the generalized Cholesky factorization in modified mode (`QRelGDCholM`) from Section 5.2.

Then the QuadFilter Algorithm 4 can be applied to find a rigorous box enclosure of the feasible set of the uncertain quadratic problem (3).

6.4 **Computational cost**

The computational cost of the QuadFilter algorithm is dominated by the directed Cholesky factorization and therefore of the order

$$O\Big(\sum_{i\in Q}|K_i|^3\Big),$$

where Q is the index set of all nonseparable quadratic constraints and K_i is the index set of the variables which are occur in a bilinear term of the ith constraint; the separable part only contributes work of order $O(k)$ for k separable variables.

This means that the QuadFilter method is fast even for high dimensional problems, as long as only few variables appear in bilinear terms of the same constraint. This covers a very large class of problems from the applications; see Subsection 7.2.

The method remains applicable even when a constraint contains a large number p of variables in bilinear terms; it just gets slower, at a rate of $O(p^3)$. Thus use for a constraint with more than 1000 variables appearing in bilinear terms should perhaps be restricted to the root node.

7 Tests

In this section we compare the different quadratic relaxation methods selectable as listed in Section 6.3, first on the toy problems of Section 2 and then on a large number of quadratic problems from the Coconut Testset (see [20]) with the exception of a few extremely large problems with over a million of variables where global optimization with branch and bound is unlikely to succeed anyway, no matter how efficient the methods. The Coconut Testset is comprehensive global optimization problem library used for testing and comparing global optimization methods and solvers (see, Domes et al. [1] and Neumaier et al. [14]).

7.1 **Behavior on the toy problems**

The following table summarizes the test results for the toy problems from Section 2.

Test result summary for the toy problems

```
Solving toy problem 1.0:
  5*x0^2+6*x0*x1+6*x1*x0+5*x1^2-3*x0-x1 <= 6
  x0 \in [-2,1]
    EHull: Directed Cholesky factorization failed: non positive pivot=-2.2 found in the step 2
    QRelIncDirChol: x0 \in [-2,1], x1 \in [-3.846,5.246]
    QRelModDirChol: x0 \in [-2,1], x1 \in [-5.399,5.599]
    QRelGenDirCholMod: x0 \in [-2,1], x1 \in [-5.368,5.568]
    QRelGenDirChol: x0 \in [-2,1], x1 \in [-2.601,4.001]
Solving toy problem 2.0:
  5*x0^2+6*x0*x1-7.5*x0*x2+6*x1*x0+5*x1^2-6*x1*x2-7.5*x2*x0-6*x2*x1+6*x2^2-3*x0-x1+5*x2 <= 2.75
```

```
  x0 \in [-2,1], x2 \in [0,3]
    EHull: Directed Cholesky factorization failed: non positive pivot=-2.2 found in the step 2
    QRelIncDirChol: x0 \in [-2,1], x1 \in [-7.945,12.59], x2 \in [0,3]
    QRelModDirChol: x0 \in [-2,1], x1 \in [-13.268,13.468], x2 \in [0,3]
    QRelGenDirCholMod: x0 \in [-2,1], x1 \in [-13.246,13.446], x2 \in [0,3]
    QRelGenDirChol: x0 \in [-2,1], x1 \in [-2.501,7.501], x2 \in [0,3]
Solving toy problem 3.0:
  x0^2+4*x0*x1-0.5*x0*x2+4*x1*x0+10*x1^2-5*x1*x2-0.5*x2*x0-5*x2*x1+5*x2^2-2*x0-6*x1+4*x2 <= -1.7
  x0 \in [-2,1]
    EHull: Directed Cholesky factorization failed: non positive pivot=-1.5 found in the step 3
    QRelIncDirChol: x0 \in [-2,1], x1 \in [-2.789,3.889], x2 \in [-4.621,4.821]
    QRelModDirChol: x0 \in [-2,1], x1 \in [-1.297e11,98.798], x2 \in [-1.201e11,84.334]
    QRelGenDirCholMod: x0 \in [-2,1], x1 \in [-15.441,8.978], x2 \in [-14.277,7.394]
    QRelGenDirChol: x0 \in [-2,1], x1 \in [-1.49,2.59], x2 \in [-2.78,2.98]
```

The results show that except for `EHull` (which applies only to convex problems) all methods reduced the domain of the toy problems to a bounded domain. The bounds from `QRelMDChol` are quite poor for the third problem. The reason for this is that the matrix $\mathbf{A} + D$ produced by the modified Cholesky factorization in this case is very ill-conditioned. This actually was one of the motivations to improve the factorization from [5]. With this exception, all new methods produced reasonable finite bounds for all unbounded variables. The best method is `QRelGDChol` which produces quite tight bounds for all toy problems.

Table 1 P_n is the number of "typical" problems in the COCONUT Environment test set that have n variables.

n	1	2	3	4	5	6	7	8	9	10	11
P_n	10	133	97	74	73	49	25	44	26	40	8
n	12	13	14	15	16	17	19	20	21	22	24
P_n	18	6	5	6	5	2	4	10	2	4	8
n	25	26	27	28	29	30	31	32	33	36	38
P_n	2	1	3	5	2	6	4	2	1	1	5
n	40	41	43	44	46	48	49	50	51	52	55
P_n	2	1	1	1	1	1	1	5	1	1	2
n	57	58	59	60	61	62	63	64	65	66	70
P_n	1	1	1	7	2	2	1	4	4	1	1
n	72	75	78	79	80	81	82	84	85	90	96
P_n	2	2	1	1	2	1	1	1	2	2	2
n	100	101	105	108	110	111	112	115	117	118	120
P_n	17	5	1	1	12	1	1	3	1	2	6
n	125	127	130	141	142	143	150	155	157	160	192
P_n	1	2	1	1	1	2	2	2	1	1	2
n	199	200	202	203	209	222	240	249	250	288	300
P_n	2	1	1	1	1	1	1	1	1	1	2
n	301	303	320	327	343	363	372	385	400	432	436
P_n	2	1	1	1	1	1	2	2	1	1	3
n	450	468	480	496	499	500	512	517	540	600	601
P_n	1	3	1	1	1	2	1	1	3	2	2
n	640	650	699	713	735	799	800	882	900	999	1000
P_n	1	2	2	1	1	1	1	1	1	1	26
n	1024	1094	1113	1198	1201	1250	1254	1485	1499	1773	2000
P_n	4	1	1	1	1	1	1	3	1	1	1
n	2002	2005	2398	2401	2500	2504	2510	3000	3750	3754	3873
P_n	2	5	2	1	4	1	1	12	2	1	3
n	4798	5000	5004	5120	6902	7510	9598	9998	9999	10000	10001
P_n	1	9	1	1	7	1	1	1	1	18	1
n	10002	13802	14985	14996	14999	20000	20192	20200			
P_n	12	1	1	2	1	2	1	1			

7.2 **Constraint statistics**

Table 2 M_c is the number of "typical" problems in the COCONUT Environment test set used for which c is the maximum of the number variables in bilinear terms in a constraint. N_c is the number of constraints in these problems containing c variables in bilinear terms.

c	1	2	3	4	5	6	7	8	9	10	11
M_c	54	145	71	59	46	50	10	19	8	11	2
N_c	6842	3285	3399	45243	31309	8489	132	416	17	418	21
c	12	13	14	15	16	17	18	19	20	21	22
M_c	20	1	2	5	4	0	1	2	9	2	1
N_c	547	25	133	3443	113	6	12	8	186	2	1
c	24	25	27	28	29	30	38	50	55	60	64
M_c	2	1	1	2	1	3	1	5	1	1	3
N_c	13	1	1	7	1	3	1	5	1	1	3
c	75	78	84	85	96	100	109	111	143	200	231
M_c	1	2	1	1	1	10	2	1	2	1	2
N_c	1	4	1	1	1	11	2	1	2	1	2
c	300	349	372	385	400	432	501	512	698	900	1000
M_c	1	1	2	5	1	1	1	1	1	1	18
N_c	1	1	2	5	1	1	1	1	1	1	19
c	1250	2002	2005	2500	2673	3750	3873	5000	5001	9997	10000
M_c	1	2	5	4	1	2	2	5	1	1	9
N_c	2	2	5	5	1	3	2	6	1	1	9
c	10002	14996	19800	20000	20200						
M_c	12	2	1	2	1						
N_c	12	2	1	2	1						

Our new techniques are designed to improve existing methods only for problems involving quadratic constraints with at least one bilinear term. As mentioned in Subsection 6.4, the work per constraint is cubic in the number of bilinear terms in a constraint, hence significant only in "long" nonseparable quadratic constraints containing a substantial number of variables in bilinear terms. This happens very rarely; more than 94.5% of the constraints, even in high-dimensional problems, only contain 6 or fewer variables in bilinear terms, and less than 0.5% of the constraints contain 15 or more variables in bilinear terms. In a branch-and-bound application, one can either skip these "long" nonseparable constraints when performing the convex relaxation, or accept the higher cost for processing them.

In fact, most problems possess – if at all – only very few "long" nonseparable quadratic constraints. Among the 1006 problems from the full COCONUT Environment test set, this only happened for two problems `chandheq` and `eigenc2`. The problem `chandheq` of dimension 100 has 100 constraints each with bilinear terms containing all 100 variables. `eigenc2` of dimension 462 has 210 constraints each with bilinear terms containing 42 variables. For these problems the present methods would be somewhat (but not excessively) slow.

The remaining 1004 "typical" problems contain 5 or fewer constraints with 40 or more nonseparable quadratic variables. For these problems we calculated in Tables 1 and 2 some additional statistics on dimensions and constraints.

7.3 **Selection of the test set**

In this section we present the results of testing the different filtering methods from this paper on the Coconut Environment test set. We assess and compare the use-

fulness of the filtering methods by creating a scenario which is typical for subproblems treated during the branch and bound process and by applying them after the quadratic constraint propagation has already reached its contraction limit.

7.3.1 Test settings and their justification

In order to create a suitable environment for testing the following considerations have been made.

- We selected from `Lib` 1-3 of the COCONUT Environment test set [20] all constrained quadratic optimization problems and constraint satisfaction problems with no more than 300 variables and no more than 300 constraints. For each of the resulting 219 problems.... Though the test set selected is limited in dimensionality, the statistics of Subsection 7.2 shows that its constraint sizes are typical of general problems.
- Each objective function (if present) was replaced by a constraint $f(x) \leq \overline{f}$, where (for problems formulated in the form (3))

$$\overline{f} := \sup \mathbf{f}^* + e_r|\mathbf{f}^*|, \quad \mathbf{f}^* := (x^*)^T \mathbf{A}^0 x^* + 2\mathbf{a}^{0T} x^* + d \tag{58}$$

  and x^* is the approximate reference solution x^* provided by the TEST ENVIRONMENT [1]; it is the best approximate global minimizer among those found by a number of solvers applied in the past to each problem. In view of the discussion in Subsection 1.4, this simulates the stage of a branch and bound process where a reasonable approximation of the global optimizer is already available but potentially large boxes are not yet eliminated. The relative error factor $e_r = 0.01$ was chosen such that only four problems for which the reference solutions was very poor, were found infeasible by subsequent quadratic constraint propagation as described below. By construction, the remaining 215 problems are expected to have a small (and possibly empty) feasible region only.
- The behavior on unbounded domains depends a lot on the details of bounds and constraints; therefore we make no general claims. The toy examples show that the new methods can sometimes be very efficient in this case. But finite bounds cannot be obtained unless some single constraint intersects the unbounded box in a bounded domain. In the original problem formulations of the selected problems this was not the case often enough to make these interesting test problems. Note that it is possible to reformulate every constraint satisfaction problem with linear and quadratic constraints only to an equivalent CSP in which all variables are bounded; see [17].
- For all test problems selected, we added to the bound constraints in the original formulation uniform bounds of $[-1000, 1000]$ for each variable. This means that our test set tests the quality of our new filtering methods not on the box obtained at the root node but at one in a more typical-sized box during the branch and bound. This should give a more useful performance indicator than if tested on unbounded boxes.
- The box resulting as described is subjected iteratively to (fairly cheap) quadratic constraint propagation (as described in [2], which is more powerful than simple constraint propagation) until the the variable bounds remain fixed. For quadratic problems, this typically contracts quite strongly in the first few iterations and then becomes ineffective. The resulting box is chosen as the trial box for our

new methods. This guarantees that the older methods would do nothing on the initial box and would have to be split, or expensive methods such as those based on linear programming would have to be applied. Therefore any filtering gain achieved on this box delays the need to split or to apply an expensive method, and hence translates immediately into a corresponding gain in efficiency when the method is added to a branch and bound code.

The test procedure for each test problem was the following:

Pretest phase:

1. The test problem is read, and simplified.
2. The Test Environment solution x^* is used to introduce an upper bound on the objective as described in (58).
3. Each infinite upper or lower bound on a variable is replaced by the artificial upper bound of 1000 or lower bound of -1000.
4. Quadratic constraint propagation (see [2]) is used to reduce the variable bounds as much as possible. If the problem is found infeasible (which happened in 4 cases), the Test Environment solution was poor, and the problem is skipped.

Testing phase:

1. Algorithm 4 is applied to the test problem, using all possible quadratic relaxation methods independently. Since all methods perform the same operations for numerically convex constraints, such constraints are processed only by the `EHull` method. The other quadratic relaxation methods are only used (and the results measured) for constraints where `EHull` fails.
2. Several method parameters are logged, including the maximal gain

$$g_{\max} := \max_i \left(1 - \frac{\operatorname{wid} \mathbf{x}_i^{\text{after}}}{\operatorname{wid} \mathbf{x}_i^{\text{before}}}\right).$$

7.3.2 Test results

The method `eHull` was applicable to 124 of the 215 feasible problems and yielded a good reduction (maximal gain ≥ 0.2) in 32 cases. The other methods were applicable to 46 of the remaining 91 feasible problems and yielded a good reduction (maximal gain ≥ 0.2) in 5 cases for `QRelIDChol`, 10 cases for `QRelMDChol`, 11 cases for `QRelGDCholMod` and 16 cases for `QRelGDChol`.

The first part of Table 3 contains parameters describing the test set: the number of problems, the total number of variables and constraints, the number of artificial variable bounds introduced and the names of the infeasible (and therefore untested) problems. This part also contains an approximate classification of the constraints by convexity: a constraint is classified (approximately) as convex if $\underline{A}$ is numerically positive definite, as concave if $\overline{A}$ is numerically negative definite, and indefinite otherwise.

The remainder of Table 3 gives a cumulative summary of the test results. For each quadratic relaxation method the number of problems and constraints for which it applicable, the total number of possible pivots (which is the sum of the number of rows (or columns) of all A for every applicable constraint), the total number of

```
Test result summary for the Test Environment problems
Problems:219
  Total variable number:4718
    Artificial variable bounds introduced: 1371
  Total constraint number: 3227
    Linear constraints: 1201
    Bilinear constraints: 881
    Quadratic constraints: 1145
      Convex: 481
      Concave: 61
      Indefinite: 603
  Infeasible: [makela3, makela4, minmaxrb, polak4]

EHull:
  Applicable for 124 problems
  Applicable for 533 constraints
  Number of possible pivots: 2512
  Number of accepted or unmodified pivots: 2512
  Pivot acceptance factor: 1
  Mean max gain: 0.204
  Problems with gain over 0.2: 32

QRelIncDirChol:
  Applicable for 46 problems
  Applicable for 607 constraints
  Number of possible pivots: 12975
  Number of accepted or unmodified pivots: 1549
  Pivot acceptance factor: 0.12
  Mean max gain: 0.104
  Problems with gain over 0.2: 5

QRelModDirChol:
  Applicable for 46 problems
  Applicable for 607 constraints
  Number of possible pivots: 12975
  Number of accepted or unmodified pivots: 235
  Pivot acceptance factor: 0.02
  Mean max gain: 0.2
  Problems with gain over 0.2: 10

QRelGenDirCholMod:
  Applicable for 45 problems
  Applicable for 507 constraints
  Number of possible pivots: 2975
  Number of accepted or unmodified pivots: 1449
  Pivot acceptance factor: 0.49
  Mean max gain: 0.229
  Problems with full reduction: [bt13]
  Problems with gain over 0.2: 11

QRelGenDirChol:
  Applicable for 46 problems
  Applicable for 607 constraints
  Number of possible pivots: 12975
  Number of accepted or unmodified pivots: 1549
  Pivot acceptance factor: 0.12
  Mean max gain: 0.318
  Problems with full reduction: [bt13, matrix2]
  Problems with gain over 0.2: 16
```

Table 3 Test results for the COCONUT Environment Testset.

accepted and unmodified (and therefore positive) pivots, the pivot acceptance factor (the number of accepted pivots divided by the number of possible pivots), the mean maximum gain over all applicable problems and the number of problems where the gain was over 0.2.

Figure 4 shows the performance of the different quadratic relaxation methods: for each of them the maximal gain on each applicable problem were sorted in descending order, and plotted against the problem number. Since the figure shows that the average performance of the `QRelGDChol` method clearly surpasses the performance of the other methods on the global scale, it is also interesting to find out if there are some problems where any of the other methods are better than `QRelGDChol`. Therefore the performance of the quadratic relaxation methods (except `EHull`) were compared on individual problems: the maximal gain on each applicable problem was sorted by the maximal gain of `QRelGDChol` and plotted against the problem number in Figure 5. The figure shows that the `QRelGDChol` method always outperforms all other methods, and that the `QRelGDCholMod` method comes second.

7.4 **Conclusions**

Based on the results of previous research on convex quadratic problems (method `EHull`) in this paper different rigorous filtering methods applicable to nonconvex

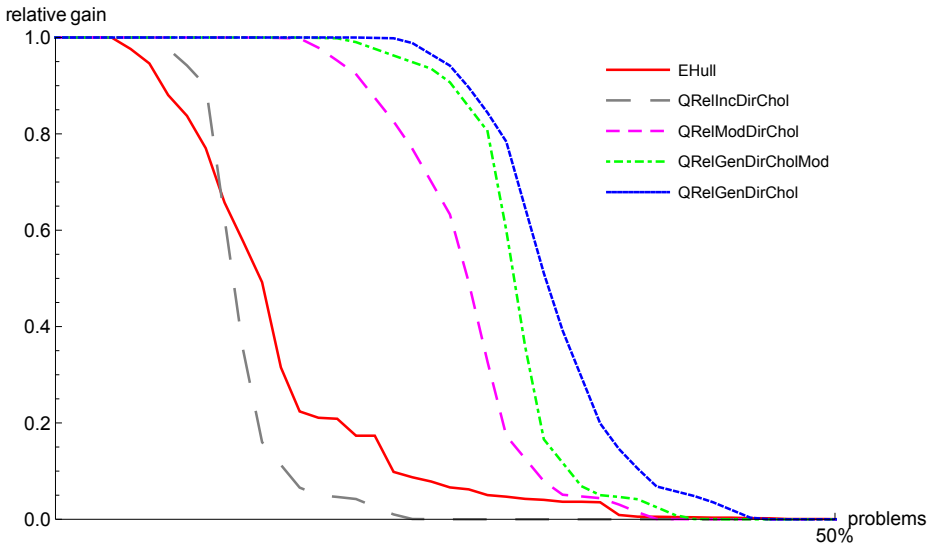

Fig. 4 Performance profiles for each quadratic relaxation method on the COCONUT Environment Testset. The maximal gains for each method are sorted in descending order and plotted against the percentage of problems (a total of 124 for `EHull` and a total of 46 for the other methods).

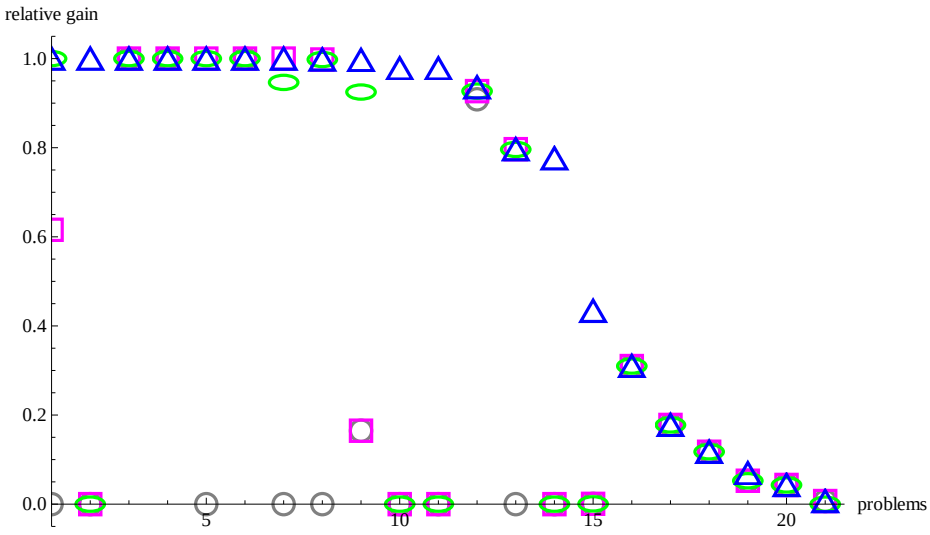

Fig. 5 Performance profiles for each quadratic relaxation method (except `EHull`) on the COCONUT Environment Testset on individual problems. For each method the maximal gains were sorted by the maximal gain of `QRelGDChol` and plotted against the problem number. Only the 21 test problems with nonzero gain are plotted.

quadratic optimization or constraint satisfaction problems were presented. The new methods use a novel and generalized version of the directed Cholesky factorization.

The tests in the paper show that the application range and efficiency of `EHull` has been greatly extended and improved by the new methods. The tests were all performed after quadratic constraint propagation stopped to improve the bounds, and especially the `QRelGDChol` method has been observed to yield a significant bound reduction for a substantial fraction of the constraints not tractable by `EHull`. Actually the performance profiles show that the `QRelGDChol` (although it is more tedious to implement) yields for each test problem as least as good reduction as the best of the other methods.

The different methods have been incorporated in the QuadFilter algorithm, for which the computational cost is roughly proportional to $\sum_{i\in Q} |K_i|^3$, where Q is the index set of all nonseparable quadratic constraints and K_i is the index set of the variables that occur in bilinear terms of the ith constraint.

Since for most high dimensional problems none or only very few constraints contain a high-dimensional bilinear part, the QuadFilter method is expected to be usually fast even in very high dimensions. The method remains applicable even when a constraint contains a large number p of variables in bilinear terms; it just gets slower, at a rate of $O(p^3)$.

The behavior on unbounded domains depends a lot on the details of bounds and constraints; therefore we make no general claims. The toy examples show that the new methods can sometimes be very efficient in this case. But finite bounds cannot be obtained unless some single constraint intersects the unbounded box in a bounded domain. To increase the efficiency further, one would have to consider the simultaneous exploitation of multiple constraints (see Domes & Neumaier [4]).

References

1. F. Domes, M. Fuchs, H. Schichl, and A. Neumaier. The Optimization Test Environment. *Optimization and Engineering*, 15:443–468, 2014. URL: `http://www.mat.univie.ac.at/~dferi/testenv.html`.
2. F. Domes and A. Neumaier. Constraint propagation on quadratic constraints. *Constraints*, 15:404–429, 2010. URL: `http://www.mat.univie.ac.at/~dferi/research/Propag.pdf`.
3. F. Domes and A. Neumaier. Rigorous enclosures of ellipsoids and directed Cholesky factorizations. *SIAM Journal on Matrix Analysis and Applications*, 32:262–285, 2011. URL: `http://www.mat.univie.ac.at/~dferi/research/Cholesky.pdf`.
4. F. Domes and A. Neumaier. Constraint aggregation in global optimization. *Mathematical Programming*, pages 1–27, 2014. Online First. URL: `http://www.mat.univie.ac.at/~dferi/research/Aggregate.pdf`.
5. F. Domes and A. Neumaier. Directed modified Cholesky factorization and ellipsoid relaxations. Technical report, University of Vienna, 2014. URL: `http://www.mat.univie.ac.at/~dferi/research/Modchol.pdf`.
6. F. Domes and A. Neumaier. JGloptLab – a rigorous global optimization software. in preparation, 2014. URL: `http://www.mat.univie.ac.at/~dferi/publications.html`.
7. F. Domes and A. Neumaier. Rigorous verification of feasibility. *Journal of Global Optimization*, pages 1–24, 2014. online first. URL: `http://www.mat.univie.ac.at/~dferi/research/Feas_csp.pdf`.
8. A. Frommer and B. Hashemi. Verified stability analysis using the lyapunov matrix equation. *Electronic Transactions on Numerical Analysis*, 40:187–203, 2013.
9. E. R. Hansen. *Global Optimization Using Interval Analysis*. Marcel Dekker Inc., New York, 1992.
10. R. B. Kearfott. On proving existence of feasible points in equality constrained optimization problems. *Mathematical Programming*, 83(1–3):89–100, 1995. URL: `http://interval.louisiana.edu/preprints/constrai.pdf`.

11. R.B. Kearfott, M.T. Nakao, A. Neumaier, S.M. Rump, S.P. Shary, and P. van Hentenryck. Standardized notation in interval analysis. In *Proc. XIII Baikal International School-seminar "Optimization methods and their applications"*, volume 4, pages 106–113, Irkutsk: Institute of Energy Systems, Baikal, 2005.
12. R. Misener and C. A. Floudas. GloMIQO: Global mixed-integer quadratic optimizer. *Journal of Global Optimization*, 57(1):3–50, 2013. URL: `http://dx.doi.org/10.1007/s10898-012-9874-7`, `doi:10.1007/s10898-012-9874-7`.
13. R. Misener and C. A. Floudas. ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59(2-3):503–526, 2014. URL: `http://dx.doi.org/10.1007/s10898-014-0166-2`.
14. A. Neumaier, O. Shcherbina, W. Huyer, and T. Vinkó. A comparison of complete global optimization solvers. *Mathematical Programming B*, 103:335–356, 2005.
15. H. Schichl and M. C. Markót. Algorithmic differentiation techniques for global optimization in the COCONUT environment. *Optimization Methods and Software*, 27(2):359–372, 2012. URL: `http://www.mat.univie.ac.at/~herman/papers/griewank.pdf`.
16. H. Schichl, M. C. Markót, A. Neumaier, Xuan-Ha Vu, and C. Keil. The COCONUT Environment, 2000-2010. Software. URL: `http://www.mat.univie.ac.at/coconut-environment`.
17. H. Schichl, A. Neumaier, M. Markót, and F. Domes. On solving mixed-integer constraint satisfaction problems with unbounded variables. In Carla Gomes and Meinolf Sellmann, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 7874 of *Lecture Notes in Computer Science*, pages 216–233. Springer Berlin Heidelberg, 2013. URL: `http://www.mat.univie.ac.at/~dferi/research/cpaior2013.pdf`.
18. R. B. Schnabel and E. Eskow. A new modified Cholesky factorization. *SIAM Journal on Scientific and Statistical Computing*, 11(6):1136–1158, 1990.
19. R. B. Schnabel and E. Eskow. A revised modified Cholesky factorization algorithm. *SIAM Journal on Optimization*, 9(4):1135–1148, 1999.
20. O. Shcherbina, A. Neumaier, D. Sam-Haroud, Xuan-Ha Vu, and Tuan-Viet Nguyen. Benchmarking global optimization and constraint satisfaction codes. In Ch. Bliek, Ch. Jermann, and A. Neumaier, editors, *Global Optimization and Constraint Satisfaction*, pages 211–222. Springer, 2003. URL: `http://www.mat.univie.ac.at/~neum/ms/bench.pdf`.
21. A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1):25–57, May 2006. URL: `https://projects.coin-or.org/Ipopt`, `doi:10.1007/s10107-004-0559-y`.