Inner Regions and Interval Linearizations for Global Optimization

Gilles Trombettoni and Ignacio Araya and Bertrand Neveu and Gilles Chabert
INRIA, I3S, Université Nice–Sophia (France), UTFSM (Chile), Imagine LIGM Université Paris–Est (France), LINA, EMN (France)
Gilles.Trombettoni@inria.fr, iaraya@inf.utfsm.cl, neveub@certis.enpc.fr, Gilles.Chabert@emn.fr

Abstract
Researchers from interval analysis and constraint (logic) programming communities have studied intervals for their ability to manage infinite solution sets of numerical constraint systems. In particular, inner regions represent subsets of the search space in which all points are solutions. Our main contribution is the use of recent and new inner region extraction algorithms and safe linear relaxations respectively. More recently, the mathematical programming community has also contributed with a solver, called here IBBA+, that integrates constraint propagation and affine arithmetic (Ninin, Mesquine, and Rueher 1997) and extends this free C++ library. Our strategy significantly outperforms the best reliable global optimizers.

1 Introduction
Interval B&B algorithms are used to solve constrained global optimization problems in a reliable way, i.e., they provide an optimal solution and its cost with a bounded error or a proof of infeasibility. The story of interval B&B started with interval analysis (Moore 1966). Numerous pioneering ideas are detailed in books like (Moore 1966), (Hansen 2003), (Kearfott 1996), to name a few. In the middle of the nineties, Kearfott designed the GlobSol solver, and researchers from the constraint programming community designed the solvers Numerica (Van Hentenryck, Michel, and Deville 1997) and Icos (Lebbah, Michel, and Rueher 2007) that introduce interval constraint propagation algorithms and safe linear relaxations respectively. More recently, the mathematical programming community has also contributed with a solver, called here IBBA+, that integrates constraint propagation and affine arithmetic (Ninin, Mesquine, and Hansen 2011).

For ensuring reliability at a good performance, the interval paradigm is faced with two main difficulties.

Upper bounding in the feasible space. Local search is the most used approach for finding a feasible point (i.e., a solution satisfying the constraints) that improves the best value of the objective function $f$. However, to ensure reliability in presence of equality constraints, the exploration of a search space containing feasible and unfeasible points requires an additional iterative correction of the unfeasible points found during the local search and their certification with expensive interval analysis techniques (Lebbah, Michel, and Rueher 2007). This additional time-consuming iterative correction makes it simply impossible the competition, in terms of performance, with state-of-the-art non-reliable global optimizers like Baron (Tawarmalani and Sahinidis 2005).

In this paper, we propose a radically different approach where the search effort is spent only inside inner regions of the search space, i.e., regions in which all points are feasible. Several researchers from interval constraint programming have intensively studied inner boxes for paving the solution set (Collavizza, Delobel, and Rueher 1999; Benhamou and Goualard 2000). Inner boxes have also been used for minimizing the number of violated numerical inequality constraints (Normand et al. 2010). However, the paradigm remained not exploited in general global optimization under inequality and equality constraints.

Lower bounding with reliable convexification. All the existing solvers compute, at each node of the B&B, a convex, generally polyhedral, outer approximation of the solution set. The best solution of the obtained relaxation (of the initial problem) yields a lower bound of the cost that is needed to terminate the search. Most of the linear relaxations are sophisticated, rendering tedious the task of making them conservative. Indeed, the outer approximation must enclose the solution set in spite of floating-point calculation errors. Specific Reformulation-Linearization Techniques (Sherali and Adams 1999) are presented in (Kearfott 1996) and (Lebbah, Michel, and Rueher 2007) that add new variables equal to powers or products in the system, and define linear constraints between them. Ninin et al. use affine expressions

\[ f(x) = \sum_{i=1}^{n} c_i x_i^m + \sum_{i=1}^{n} d_i x_i^k \]

where $m, k \geq 2$ are integers, $c_i, d_i \in \mathbb{R}$, and $x_i \in \mathbb{R}$. This formulation requires Lagrangian machinery and Kuhn-Tucker theorem. This can lead to a huge aggregate function unadapted for interval computations (Hansen 2003).
arithmetic for computing a safe linearization of each operator (Ninin, Messine, and Hansen 2011). Instead, we propose in this paper a reliable linearization based on a first-order interval Taylor. The simplicity of this interval relaxation also leads to a dual version that can extract an inner polyhedral region inside the solution set and improve the upper bound.

1.1 Intervals and constrained global optimization

An interval \([x_i] = [x_i, x_i] \) defines the set of reals \(x_i\) s.t. \(x_i \leq x_i \leq x_i\). IR denotes the set of all intervals. The size or width of \([x_i]\) is \(w([x_i]) = x - x_i\). A box \([x]\) is the Cartesian product of intervals \([x_1] \times \ldots \times [x_n]\). Its width is defined by \(\max_i w([x_i])\). Mid([x]) denotes the middle of \([x]\). A numerical or continuous constrained global optimization problem is defined as follows.

**Definition 1 (Constrained global optimization)**

Consider a vector of variables \(x = [x_1, \ldots, x_n]\) varying in a box \([x]\), a real-valued function \(f : \mathbb{R}^n \to \mathbb{R}\), vector-valued functions \(g : \mathbb{R}^n \to \mathbb{R}^m\) and \(h : \mathbb{R}^n \to \mathbb{R}^p\). Given the system \(S = (f,g,h,x,][x]\), the constrained global optimization problem consists in finding:

\[
\min_{x \in [x]} f(x) \text{ subject to } g(x) \leq 0 \land h(x) = 0.
\]

\(f\) denotes the objective function; \(g\) and \(h\) are inequality and equality constraints respectively. \(x\) is said to be feasible if it satisfies the constraints.

Our interval optimizer extracts inner boxes and inner regions inside classical (outer) boxes.

**Definition 2**

Consider a system \(S = (f, g, h, x, [x]^{\text{out}})\) comprising only inequality constraints. An inner region \(x^{\text{in}}\) is a feasible subset of \([x]^{\text{out}}\), i.e., \(x^{\text{in}} \subset [x]^{\text{out}}\) and all points \(x \in x^{\text{in}}\) satisfy \(g(x) \leq 0\).

An inner box \([x]^{\text{in}}\) is an inner region which is a box.

Due to the inner linearizations achieved by our strategy, the considered inner regions are polytopes.

Interval arithmetic (Moore 1966) extends to \(\mathbb{R}\) elementary functions over \(\mathbb{R}\). For instance, the interval sum \((x_1 + x_2)_I\) and \((x_1 \times x_2)_I\) encloses the image of the sum function over its arguments, and this enclosing property basically defines what we call an interval extension.

**Definition 3 (Extension of a function to \(\mathbb{R}\))**

Consider a function \(f : \mathbb{R}^n \to \mathbb{R}\). \([f] : \mathbb{R}^n \to \mathbb{R}\) is said to be an extension of \(f\) to intervals if:

\[
\forall x \in \mathbb{R}^n \quad [f](x) = f(x) \quad (x \in [x])
\]

In our context, the expression of a function \(f\) is always a composition of elementary functions. The natural extension \([f]_N\) is then simply a composition of the corresponding interval operators.

The outer and inner interval linearizations proposed in this paper are related to the first-order interval Taylor extension (Moore 1966), defined as follows:

\[
[f]_N([x]) = f(\hat{x}) + \sum_i \left[\frac{\partial f}{\partial x_i}\right]_N ([x]) \ast (x_i - \hat{x}_i)
\]

where \(\hat{x}\) denotes any point in \([x]\), e.g., Mid([x]).

**Example.** Consider \(f(x_1, x_2) = 3x_1^2 + x_2^3 + x_1 \ast x_2\) in the box \([x] = [-1, 3] \times [-1, 5]\). The natural evaluation provides:

\[
[f]_N([x_1], [x_2]) = 3([-1, 3]^2 + [-1, 5]^3) + [-1, 3] \ast [-1, 5] = [0, 27] \times [0, 25] \times [5, -15, 15] = [-5, 67].
\]

The partial derivatives are:

\[
\frac{\partial f}{\partial x_1} \ast (x_1, x_2) = 6x_1 + x_2, \quad \frac{\partial f}{\partial x_2} \ast (x_1, x_2) = x_1 + 2x_2.
\]

The interval Taylor evaluation with \(\hat{x} = (1, 2)\) yields:

\[
[f]_T([x_1], [x_2]) = 9 \ast [-7, 23] \ast [-2, 2] \ast [-3, 13] \ast [-3, 3] = [-76, 94].
\]

1.2 Handling equations as inequality constraints

To handle equalities, in a first option followed by the interval community, one finds approximately a point that satisfies exactly the constraints. The solvers return a tiny box of width \(\epsilon_{\text{sol}}\) in which the existence of a real-valued point is (often) guaranteed by interval Newton methods. In a second option, one finds exactly a point that satisfies approximately the constraints. Equations are handled with a (tiny) admissible precision error \(\epsilon_{\text{eq}}\), i.e., a feasible floating-point \(x\) verifies \(h(x) \in [-\epsilon_{\text{eq}} + \epsilon_{\text{eq}}]\). All the constraints can thus be viewed as equalities: \(\{g(x) \leq 0, h(x) - \epsilon_{\text{eq}} \leq 0, -h(x) - \epsilon_{\text{eq}} \leq 0\}\). Ninin et al. were guided to this choice by their affine arithmetic, but we believe that this is a relevant approach for any global optimization solver. First, both policies are of equal status regarding reliability. Second, a precision error \(\epsilon_{\text{eq}}\) on the images of functions \(h\) better fits the original feasibility problem than a precision \(\epsilon_{\text{sol}}\) on the unknowns. Third, most of the equations defined by interval practitioners are already “thick” and do not require a relaxation with \(\epsilon_{\text{eq}}\). Indeed, constraints often have coefficients known with a bounded uncertainty (e.g., an imprecision on a measured distance) and sometimes contain irrational constants, like \(\pi\), that can be specified by tiny intervals. Finally, our experiments give an evidence that handling thick equalities can work efficiently in practice. The reason behind this good surprise is that this policy allows optimizers to extract inner regions in continua of solutions. Efficient inner region extraction and filtering algorithms can then focus the search in the tiny solution set defined by thick equations.

2 Description of our interval B&B

Our \textsc{IbexOpt} strategy follows the well-known branch and bound schema described in (Horst and Tuy 1966) to solve a constrained global optimization problem. Starting from an initial box, the algorithm splits it recursively until a solution minimizing the objective function is found. During the search, a current (generally non feasible) lower bound of the objective function is computed for each box in the list managed by the algorithm. We call \(lb\) (for lower bound) the minimum value of these lower bounds. Also, \(ub\) (for upper bound) is the cost of the current best feasible point found during the search. A termination occurs when \(ub - lb\) reaches a precision \(\epsilon_{\text{obj}}\), and the floating-point vector \(x_{ub}\) of cost \(ub\) is returned. Note that boxes the width of which does not exceed a precision \(\epsilon_{\text{sol}}\) are not put again in the list and their lower bound accounts in the computation of \(lb\).

At each iteration, the algorithm selects in the list the box \([x]\) with the lowest lower bound, thus following a best-first

\[\text{Following standard implementations, } \epsilon_{\text{obj}}\text{ is a percentage of } ub\text{ if } |ub| \geq 1; \epsilon_{\text{obj}}\text{ is an absolute distance if } |ub| \leq 1.\]
search. It chooses a branching variable \( x_i \in x \) heuristically, bisects \([x_i]\) and applies the main Contract&Bound procedure on the two sub-boxes. Note that the search tree, i.e., the “list” of boxes to be handled, is managed by a heap data structure to access to the minimum lower bound in constant time. More details about the overall schema can be found in (Ninin, Messine, and Hansen 2011).

Like for instance in Numerica (Van Hentenryck, Michel, and Deville 1997), the first task of our optimizer is to automatically introduce a new variable \( y \) in the input system \((f, g, h, x, [x])\). This variable is linked to the others by an additional constraint \( y = f(x) \). The domain \([y]\) is therefore an interval that encompasses the image of the objective function on \([x]\). It can be used as a simple way to propagate and retro-propagate contractions between \([x]\) and global bounds on the minimum. Hence, the extended box \([x] \times [y]\) defines the backtrackable state of the optimizer, and three variables shared by all nodes in the search tree are updated globally during the search: the current best candidate \( x_{ub} \), its cost \( ub \) \((f(x_{ub}) = ub)\) and the minimum of the lower bounds \( lb \).

### 2.1 A variant of the smear branching strategy

At each search node, a variant of the well-known smear function (Kearfott and Novoa III 1990) selects the next variable to be split. Given a system \((f, g, h, x, [x])\), the standard smear-based strategy selects the variable \( x_i \) in \( x \) with the greatest value \( \text{smearMax}(x_i) = \max_j \text{smear}(x_i, f_j) \) or \( \text{smearSum}(x_i) = \sum_j \text{smear}(x_i, f_j) \), according to two different variants, where \( f_j \) ranges over all the functions \((f \text{ and the components of } g, h)\).

\( \text{smear}(x_i, f_j) \) reflects an impact of the variable \( x_i \) on function \( f_j \). It depends on the partial derivative of \( f_j \) w.r.t. \( x_i \) and on the width of \([x_i]\). More precisely:

\[
\text{smear}(x_i, f_j) = \left[ \frac{\partial f_j}{\partial x_i} \right]_N \cdot \text{w}([x_i]).
\]

We propose a variant \( \text{smearRel}(x_i, f_j) \) of \( \text{smear}(x_i, f_j) \) that simply measures a relative impact falling in \([0, 1]\):

\[
\text{smearRel}(x_i, f_j) = \frac{\text{smear}(x_i, f_j)}{\sum_{x_k \in x} \text{smear}(x_k, f_j)}.
\]

Finally, our branching strategy \( \text{smearSumRel}(x_i) \) selects the variable \( x_i \) in \( x \) with the greatest impact:

\[
\text{smearSumRel}(x_i) = \sum_j \text{smearRel}(x_i, f_j).
\]

Although not always the best, this strategy appears to be more robust than its competitors on the tested benchmark.

### 2.2 The Contract&Bound procedure

The main algorithm Contract&Bound (see Algorithm 1) is called at each node of our B&B. The first line introduces in the current system the best cost \( ub \) ever found (like in any B&B). The procedure OuterContractLB filters the domains and improves the lower bound. InnerExtractUB extracts inner regions inside \([x]\), chooses a point \( x \) in those regions (if any), and potentially replaces \( x_{ub} \) by \( x \) and its cost \( ub \) by \( f(x) \). OuterContractLB calls two main procedures. First, the Mohc algorithm (Araya, Trombettoni, and Neveu 2010) contracts the box \([x] \times [y]\). It can thus also lower bound the objective function. This recent interval constraint propagation algorithm exploits monotonicity of functions. It uses an efficient Revise procedure that can optimally contract the box w.r.t. a single constraint (e.g., \( g_j(x) \leq 0 \)), when \( g_j(x) \) is monotonic w.r.t. every variable in the box, even if \( g_j(x) \) contains multiple occurrences of variables. Note that the smaller a box is, i.e., the deeper in the search tree, the likelier functions are monotonic w.r.t. variables. Second, OuterContractLB calls an interval linearization, called here OuterLinearization, for lower bounding the objective function, i.e., for increasing \( y \).

#### 2.3 Outer interval linearization

A safe polyhedral convexification is built upon a straightforward adaptation of a specific first order interval Taylor form of a nonlinear function (Lin and Stadtherr 2004). Consider a function \( f: \mathbb{R}^n \rightarrow \mathbb{R} \) defined on a domain \([x]\). For any variable \( x_i \in x \), let \([a_i]\) be \([df\big|_{x_i = \bar{x}}\big](x_i)\). The idea is to (lower) tighten \( f(x) \) with linear functions like:

\[
\forall x \in [x], f(x) + a_1 x_1 + \ldots + a_n x_n \leq f(x)
\]  

\[
\forall x \in [x], f(x) + a_1 x_1^r + \ldots + a_n x_n^r \leq f(x)
\]

where \( x_i^r = x_i - x_i^* \) and \( x_i^* = x_i - \bar{x} \). The first-order interval Taylor form can select any expansion point \( \bar{x} \) inside the box to achieve the linearization. Instead of the usual midpoint, a corner of the box is chosen here: \( x \) in form (1) or \( \bar{x} \) in form (2). If we consider an inequality \( g_j(x) \leq 0 \), Expression (1) or (2) defines a hyper-plane \( g_j^N(x) \) bounding the solution set from below: \( g_j^N(x) \leq g_j(x) \leq 0 \). Applying, for instance, the form (1) to the objective function \( f(x) \) and to the inequalities \( g_j(x) \leq 0 \) \((j = 1, \ldots, m)\), we can derive a linear program \( LP^{lb} \):

\[
LP^{lb} = \min \ f(x) + a_1 x_1 + \ldots + a_n x_n
\]

subject to:

\[
\forall j \ g_j(x) + a_1 x_1^r + \ldots + a_n x_n^r \leq 0
\]

\[
\forall i \ 0 \leq x_i^r, x_i^r \leq w([x_i])
\]

\[
\text{where : } x_i^r = x_i - x_i^*
\]

OuterLinearization calls a Simplex algorithm to solve \( LP^{lb} \) and returns infeasibility or the optimal value \( y^* \). Infeasibility means that \([x]\) contains no solution and can be discarded. Otherwise, if \( y^* \geq y \), then the best lower bound of the box is updated: \( y \leftarrow y^* \).

**Proposition 1** The interval linearizations (1) and (2) are correct and safe, i.e., they can be made robust to computation errors over floating point numbers.

Safety is ensured by the interval-based taylorisation (Neumaier 1990). The correction of relation (1) lies on the fact that any variable \( x_i \) is positive since its domain is \([0, d_i]\),

---

**Algorithm 1 Contract&Bound (in \( S, [x] \) out \( ub \))**

- \( \bar{y} \leftarrow ub - \epsilon_{obj} \)
- OuterContractLB \( (S, [x] \times [y]) \) \^ contraction */\(\)
- if \( [x] \times [y] = \emptyset \) then exit endif /* no solution */
- InnerExtractUB \( (S, [x], ub, x_{ub}) \) /* inner regions */
with \( d_i = w(x_i) = \pi_i - x_i \). Therefore, minimizing each term \( [a_i + x_i] \) for any point \( x_i \in [0, d_i] \) is obtained with \( a_i \). Symmetrically, relation (2) is correct since \( x_i^* \in [-d_i, 0] \), and the minimal value of a term is obtained with \( \pi_i \) (Lin and Studtherr 2004).

Note that, even though our linearizations are safe, the floating-point calculation errors made by the Simplex algorithm could make its output \( y^f \) unsafe. A cheap postprocessing step would then correct the results.

An improvement has been brought to this outer convexification for computing a tighter polytope. We lower tighten a function \( f(x) \) with expressions (1) and (2) simultaneously, using an expanded form:

1. \( f(x) + \sum_i a_i (x_i - x_i^*) = f(x) + \sum_i (a_i x_i - a_i x_i^*) = \sum_i a_i x_i + f(x) - \sum_i a_i x_i^* \)

2. \( f(x) + \sum_i \pi_i (x_i - \pi_i^*) = f(x) + \sum_i (\pi_i x_i - \pi_i \pi_i^*) = \sum_i \pi_i x_i + f(x) - \sum_i \pi_i \pi_i^* \)

### 2.4 Upper bounding with inner regions

The call to \( \text{OuterContractLB} \) is followed by a call to \( \text{InnerExtractUB} \) (see Algorithm 2). The procedure first calls an adaptation of a recent algorithm (Chabert and Beldiceanu 2010), named here \( \text{InHC4} \), for extracting an inner box from the outer box \( \{x\}^{\text{out}} \). For a single constraint, \( \text{InHC4} \) returns a box that is inner w.r.t. that constraint. The different boxes returned for all the constraints are intersected to obtain an inner box. Like \( \text{HC4} \) (Benhamou et al. 1999), the algorithm reasons on the syntactic tree of the constraints and uses projections for unary operators, with inward rounding however. Furthermore, in case of unions (e.g., in \( \land \) and \( \lor \) operators), one single interval is kept since holes contain inconsistent points, making the whole algorithm heuristic. For binary operators, the projections in the \text{backward phase} are different and also lead to heuristic choices. Details can be found in (Chabert and Beldiceanu 2010), Section 3.

If an inner box \( \{x\}^{\text{in}} \) is found by \( \text{InHC4} \), then \( \text{MonotonicityAnalysis} \) analyzes the monotonicity of the objective function \( f \) w.r.t. every variable \( x_i \). If the partial derivative \( [a_i] \) is \( \geq 0 \), then \( f \) is increasing and \( [x_i] \) is replaced by the degenerate interval \( [x_i, x_i^*] \) in \( \{x\}^{\text{in}} \) to minimize \( f(x) \) over \( \{x\}^{\text{in}} \). If \( [a_i] \leq 0 \), then \( f \) is decreasing and \( [x_i] \) is replaced by \( [x_i^*, x_i] \) in \( \{x\}^{\text{in}} \).

Next, we pick randomly a point \( x \) inside the box and replaces \( x_{ub} \) by \( x \) if \( x \) satisfies the constraints and improves the best cost \( ub \). Two different cases may occur. If an inner box has been extracted by \( \text{InHC4} \), then a point is selected inside \( \{x\}^{\text{in}} \). The feasibility does not need be checked since \( \{x\}^{\text{in}} \) contains only feasible points. If no inner box is available, and the constraints must be checked. Replacing this simple probing by a gradient descent did not improve the strategy.

---

**Algorithm 2 InnerExtractUB** (in: \( S, \{x\}^{\text{out}} \); in-out: \( ub, x_{ub} \))

```plaintext
\( \{x\}^{\text{in}} ← \text{InHC4}(S, \{x\}^{\text{out}}) /\* Inner box extraction */
if \( \{x\}^{\text{in}} \neq \emptyset \) then
\( \{x\}^{\text{in}} ← \text{MonotonicityAnalysis}(f, \{x\}^{\text{in}}) \)
x ← \text{RandomProbing}(\{x\}^{\text{in}})
else
x ← \text{RandomProbing}(\{x\}^{\text{out}})
end if
if cost \( = \|f\|_N([x, x]) \) /\* Cost evaluation */
if cost \( < ub \) and \( \{x\}^{\text{in}} \neq \emptyset \) or \( [g]_N([x, x]) \leq 0 \) then
ub ← cost; \( x_{ub} ← x \)
end if
end if
\( LP_{ub} ← \text{InnerLinearization}(S, \{x\}^{\text{out}}) \)
x' ← \text{Simplex}(LP_{ub})
if \( x' \neq \perp \) then
\( \text{cost} ← \|f\|_N([x', x']) \)
if \( \text{cost} < \text{ub} \) then
ub ← \text{cost}; \( x_{ub} ← x' \)
end if
end if
```

This is easy to understand in presence of equations since the inner boxes are tiny. This was more surprising for optimization problems under inequality constraints only. The last part of \( \text{InnerExtractUB} \) performs an \emph{inner} linearization of the system for extracting a polyhedral inner region.

### 2.5 Outer interval linearization

Symmetrically to the relation (1) used in outer linearization:

\begin{equation}
\forall x \in [x], \quad f(x) ≤ f^l(x) = f(x) + \sum_i a_i * (x_i - x_i^*). \tag{3}
\end{equation}

If an inequality \( f(x) ≤ 0 \) is handled, relation (3) enables us to build a hyper-plane \( f^l(x) \) s.t. \( f(x) \leq f^l(x) \leq 0 \). That is to say, the linear function \( f^l(x) \) can be used to define an inner region of \( [x] \). Applying this idea to the objective function \( f(x) \) and to the inequalities \( g_j(x) \leq 0 \), we can derive the linear program \( LP_{ub} \):

\begin{equation}
LP_{ub} = \min f(x) + \sum_i a_i * (x_i - x_i^*) \quad \text{subject to: } \forall j \quad g_j(x) + \sum_i a_{j,i} * (x_i - x_i^*) ≤ 0 \quad \forall i \quad x_i ≤ x_i^* \land x_i ≤ \pi_i \tag{4}
\end{equation}

A Simplex algorithm solves \( LP_{ub} \) and returns infeasibility or the optimal solution \( x' \) (see Algorithm 2). Infeasibility proves nothing because the linearized system is more constrained than the original system, so that one could still find solutions in the original one. If the Simplex algorithm returns an optimal solution of the inner approximation, then \( x' \) is also a solution of the original system, maybe not the optimal one. We evaluate the (original) objective function at the point \( x' \) and potentially update \( x_{ub} \) and \( ub \).

### 3 Experiments

We have implemented our strategy in the Interval-Based EXplorer \textbf{Ibex} (Chabert and Jaulin 2009). This free \texttt{C++} library has facilitated the implementation of our global optimizer by providing us a direct access to the \texttt{Mohec} algorithm, different branching strategies, automatic differentiation, etc. All parameters have been fixed to a given set of adequate values common to all the tested instances. The precision
has been fixed to $\epsilon_{obj} = 1.e-8$. Also, $\epsilon_{sol} = \epsilon_{eq}$. Finally, the admissible precision error $\epsilon_{eq}$ in thick equations $h(x) \in [\epsilon_{eq}, +\epsilon_{eq}]$ has been fixed to $\epsilon_{eq} = 1.e-8$ for all the experiments.

Tests have been performed on the benchmark of 74 systems used by our best competitor IBBA$^+$ (Ninin, Messine, and Hansen 2011). Table 1 presents a qualitative study analyzing which ingredients improve the performance.

Table 1: Qualitative study. The columns include the number of systems whose loss/gain in performance $CPU_{time(strategy)}$ is lower than $CPU_{time(strategy)}(ingredient\{\})$, caused by the removal of a single ingredient from our strategy, belongs to a given range (first line). The tested removals are: Mohc replaced by HC4 (Mohc/HC4); OuterLinearization (OuterLinear); InnerExtractUB replaced by a simple probing in the outer box (Inner/Probing); InnerLinearization (InnerLinear); InHC4; SmearSumRel replaced resp. by SmearMax (SSR/SM); Round Robin (SSR/RR); LF (SSR/LF); the largest-first heuristic selects the variable with the largest interval.

<table>
<thead>
<tr>
<th>Gain</th>
<th>0.02</th>
<th>0.1</th>
<th>0.5</th>
<th>2</th>
<th>10</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mohc/HC4</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>5</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>OuterLinear.</td>
<td>0</td>
<td>1</td>
<td>35</td>
<td>9</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Inner/Probing</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>24</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>InnerLinear.</td>
<td>0</td>
<td>0</td>
<td>62</td>
<td>7</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>InHC4</td>
<td>0</td>
<td>0</td>
<td>66</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SSR/SM</td>
<td>0</td>
<td>2</td>
<td>59</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>SSR/RR</td>
<td>0</td>
<td>1</td>
<td>42</td>
<td>13</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>SSR/LF</td>
<td>1</td>
<td>0</td>
<td>40</td>
<td>9</td>
<td>16</td>
<td>4</td>
</tr>
</tbody>
</table>

Interesting observations can be drawn. First, all five original devices proposed in our strategy appear to be useful in practice. Second, our simple outer linearization seems very helpful in the lower bounding phase. A future study should compare this convex interval taylorization with affine arithmetic and with the Quad RLT operator used by Icos. Third, extracting inner regions is also very useful in the upper bounding phase. Table 1 underlines that it is often sufficient to endow the strategy with InHC4 or InnerLinearization, although putting both devices is sometimes beneficial and never counter-productive on the selected benchmark.

We have also compared our strategy to the available and maintained reliable global optimizers: Globsol, Icos and IBBA$^+$, and with the non reliable deterministic solver Baron. Be aware that Baron does not guarantee its returned best solution that is sometimes not feasible and may have a too low cost.

Fig.1 shows the performance profiles of IbexOpt, Baron and our best reliable competitor IBBA$^+$. We give details on the 25 systems that are solved by IbexOpt in more than one second. Table 2 corresponds to the 12 systems solved by IbexOpt in less than 10 seconds. Table 3 includes the 13 systems solved in more than 10 seconds. Three systems ($ex6_2_5$, $ex6_2_7$ and $ex7_2_3$) are removed from this table because they are not solved by any solver, including Baron. The results for Globsol, IBBA$^+$, Icos and IbexOpt have been obtained on very similar computers (Intel X86, 3Ghz). Baron 9.0.7 has been run on the Neos server (see www.neos-server.org/neos/) also on a X86, thus making the comparison rather fair.

The figure and tables show that IbexOpt often outperforms its reliable competitors by one or several orders of magnitude. The performance profile illustrates that IbexOpt shows performances which are intermediary between IBBA$^+$ and Baron and that it can solve the same systems as Baron in 1000 seconds. The results obtained by Baron are impressive, although it should be noticed that several instances are solved during a pre-processing (the number of branching nodes is 1 in the tables).

Note that IbexOpt is better than Baron on 5 of the 25 difficult systems (see Tables 2 and 3), especially on the series $ex6_2_*$ having huge non polynomial objective functions. To our knowledge, no reliable solver could compete with Baron on trivial instances that Baron solve in seconds or more.

We have also tested a variant of our strategy where Mohc is replaced by 3BCID (Mohc) (Trombettoni and Chabert 2007). Although generally counterproductive in terms of performances, the variant is more robust and can solve the $ex7_2_3$ instance in 38 seconds with 6235 branching nodes, while Baron raises a memory overflow.

4 Conclusion

We have proposed a new framework for reliable global optimization that exploits inner regions in the upper bounding phase, thus avoiding the recourse to local search. Provided that equations can be defined with a tiny admissible precision error, the approach is also relevant for handling equality constraints. Our strategy is endowed with five significant devices. Three of them, i.e., Mohc, InHC4 and OuterLinearization have never been used in global optimization. Two of them, i.e., SmearSumRel and InnerLinearization, are new. All five have proven their efficiency in a sample of non trivial constrained global optimization systems. They confirm the relevance of inner region exploitation and polyhedral approximations based on convex interval taylorization.

Due to the number of novel ingredients, there is still significant space for improvement in the hope of reaching Baron performance in a long-term.

---

$^6$IBBA$^+$ corresponds to the most efficient strategy described in (Ninin, Messine, and Hansen 2011).
Table 2: Comparison on mean-difficult systems. The first two lines indicate the name of the competitor with the used precision $\epsilon_{obj}$ on the cost. Each entry contains generally the CPU time in second (first line of a multi-line) and the number of branching nodes (second line). A timeout of one hour (>3600) is shared by IBBA+, GlobSol and IbexOpt. It is 10 min (>600) for Icos, 1000 seconds for Baron (imposed by the Neos server). An empty entry indicates that the information is not available. In particular, GlobSol restricts itself to problems having less than 9 variables.

<table>
<thead>
<tr>
<th>System</th>
<th>$\epsilon_{obj}$</th>
<th>n</th>
<th>Baron 1.e-8</th>
<th>GlobSol 1.e-8</th>
<th>IBBA+ 1.e-8</th>
<th>Icos 1.e-3</th>
<th>IbexOpt 1.e-3</th>
<th>IbexOpt 1.e-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex2_1_7</td>
<td>0.33</td>
<td>20</td>
<td>16.75</td>
<td>1574</td>
<td>&gt;600</td>
<td>5.52</td>
<td>6.24</td>
<td></td>
</tr>
<tr>
<td>ex2_1_8</td>
<td>0.07</td>
<td>24</td>
<td>26.78</td>
<td>1916</td>
<td>&gt;600</td>
<td>5.78</td>
<td>6.50</td>
<td></td>
</tr>
<tr>
<td>ex3_1_1</td>
<td>0.51</td>
<td>8</td>
<td>116</td>
<td>131195</td>
<td>&gt;3600</td>
<td>0.48</td>
<td>1.31</td>
<td></td>
</tr>
<tr>
<td>ex6_1_4</td>
<td>0.25</td>
<td>6</td>
<td>2.70</td>
<td>1622</td>
<td>4.28</td>
<td>0.37</td>
<td>1.11</td>
<td></td>
</tr>
<tr>
<td>ex6_2_14</td>
<td>0.52</td>
<td>4</td>
<td>208</td>
<td>9510</td>
<td>&gt;600</td>
<td>0.77</td>
<td>1.59</td>
<td></td>
</tr>
<tr>
<td>ex7_2_1</td>
<td>0.05</td>
<td>7</td>
<td>24.72</td>
<td>8419</td>
<td>&gt;600</td>
<td>0.80</td>
<td>1.17</td>
<td></td>
</tr>
<tr>
<td>ex7_2_6</td>
<td>0.06</td>
<td>3</td>
<td>1.23</td>
<td>1319</td>
<td>2.68</td>
<td>0.02</td>
<td>5.35</td>
<td></td>
</tr>
<tr>
<td>ex7_3_4</td>
<td>0.93</td>
<td>12</td>
<td>&gt;3600</td>
<td>6986</td>
<td>&gt;600</td>
<td>1.27</td>
<td>3.11</td>
<td></td>
</tr>
<tr>
<td>ex14_2_1</td>
<td>0.03</td>
<td>5</td>
<td>36.73</td>
<td>16786</td>
<td>&gt;600</td>
<td>0.82</td>
<td>1.09</td>
<td></td>
</tr>
<tr>
<td>ex14_2_3</td>
<td>0.03</td>
<td>6</td>
<td>173</td>
<td>46673</td>
<td>&gt;600</td>
<td>2.57</td>
<td>2.92</td>
<td></td>
</tr>
<tr>
<td>ex14_2_4</td>
<td>0.03</td>
<td>5</td>
<td>127</td>
<td>30002</td>
<td>&gt;600</td>
<td>0.95</td>
<td>1.02</td>
<td></td>
</tr>
<tr>
<td>ex14_2_6</td>
<td>0.03</td>
<td>5</td>
<td>237</td>
<td>74630</td>
<td>&gt;600</td>
<td>1.20</td>
<td>1.29</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Comparison with competitors on difficult systems. In case of time out, the second line contains the precision obtained at this time.

<table>
<thead>
<tr>
<th>System</th>
<th>$\epsilon_{obj}$</th>
<th>n</th>
<th>Baron 1.e-8</th>
<th>GlobSol 1.e-8</th>
<th>IBBA+ 1.e-8</th>
<th>Icos 1.e-3</th>
<th>IbexOpt 1.e-3</th>
<th>IbexOpt 1.e-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex2_1_9</td>
<td>1.52</td>
<td>10</td>
<td>154</td>
<td>60007</td>
<td>59.9</td>
<td>13</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>ex6_1_1</td>
<td>7.64</td>
<td>8</td>
<td>3203</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>13</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>ex6_1_3</td>
<td>19.2</td>
<td>12</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>12811</td>
<td>14725</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_6</td>
<td>26</td>
<td>3</td>
<td>306</td>
<td>922664</td>
<td>&gt;600</td>
<td>16.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_8</td>
<td>19</td>
<td>3</td>
<td>220</td>
<td>265276</td>
<td>&gt;600</td>
<td>97554</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_9</td>
<td>170</td>
<td>4</td>
<td>465</td>
<td>203775</td>
<td>&gt;3600</td>
<td>1.e-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_10</td>
<td>&gt;1000</td>
<td>6</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>431</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_11</td>
<td>55</td>
<td>3</td>
<td>273</td>
<td>843457</td>
<td>224484</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_12</td>
<td>30</td>
<td>4</td>
<td>193</td>
<td>58231</td>
<td>4473</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex6_2_13</td>
<td>&gt;1000</td>
<td>6</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>1099</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex7_3_5</td>
<td>1.11</td>
<td>13</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>545676</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex14_1_7</td>
<td>1.27</td>
<td>10</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>177464</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex14_2_7</td>
<td>0.03</td>
<td>6</td>
<td>&gt;3600</td>
<td>&gt;600</td>
<td>84.73</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


References


