Journal of Global Optimization **19:** 61–81, 2001.

Using Interval Analysis for Solving Planar Single-Facility Location Problems: New Discarding Tests ⋆

JOSÉ FERNÁNDEZ⋆⋆ and BLAS PELEGRÍN
Departamento de Estadística e Investigación Operativa, Universidad de Murcia, 30100 Espinardo, Murcia, Spain E-mail: josefdez@um.es

(Received 6 September 1999; accepted in revised form 26 July 2000)

Abstract. Interval analysis is a powerful tool which allows the design of branch-and-bound methods able to solve many global optimization problems. The key to the speed of those methods is the use of several tests to discard boxes or parts of boxes in which no optimal point may occur. In this paper we present three new discarding tests for two-dimensional problems which are specially suitable for planar single-facility location problems. The usefulness of the new tests is shown by a computational study.

Key words: continuous location, global optimization, interval analysis, discarding tests, computational study

1. Introduction

Location science deals with the location of one or more facilities in a way that optimizes a certain objective such as minimizing transportation cost, minimizing the undesirable effects produced by the facility, capturing the largest market share, etc. In planar single-facility location problems only one facility is to be located and the set of possible locations is a plane or a region of the plane. Location problems are usually hard to solve, that is why they are sometimes used as test problems to show the efficiency of global optimization methods. For more details related with locational aspects, the interested reader is referred to [7, 24].

General methods able to solve different location models are useful for several reasons: they allow to solve location models for which there exist no *ad hoc* algorithms. Furthermore, they allow the locationer to focus his/her attention on the design of realistic models, and not to worry about the methods for solving them. Despite the importance of the topic, relatively few research work dealing with it can be found in the literature. One of the few papers presenting a general method for

⋆ This paper is a revised version of a paper presented at the VIIIth International Symposium on Locational Decisions (ISOLDE VIII) held at Coimbra and Estoril (Portugal) in 1999.

⋆⋆ Correspondence to: José Fernández Hernández, Facultad de Matemáticas, Universidad de Murcia, 30100 Espinardo, Murcia, Spain.

solving continuous location models is [17]. In that paper, Hansen et al. presented their Big Square Small Square (BSSS) method, a branch-and-bound algorithm able to solve most of constrained planar minisum single-facility location problems with l_p norms and continuous non-decreasing cost functions. Later, Plastria [25] modified the BSSS method with the aim of accelerating the calculations, minimizing the information to be stored and determining a region of *near-optimality*. His method, known as Generalized Big Square Small Square method (GBSSS), is applicable to any constrained planar single-facility location problem with distances measured by mixed norms and any box-wise optimizable continuous function of the distances as objective function. Recently, another general method has been presented by Fernández et al. [13]. This method shares some features with both BSSS and GBSSS and uses interval analysis tools. The method presented in [13] is not able to solve any constrained planar single-facility location model, but it also allows to handle problems with perturbed data [14].

The key to the speed of global optimization algorithms based on interval analysis is their use of several tests to discard boxes or parts of boxes in which no optimal point may occur. The aim of this paper is to present three new discarding tests for two-dimensional problems which are specially suitable for planar single-facility location problems. The use of the new tests makes the interval algorithm presented in [13] more efficient and quicker.

The paper is organized as follows. In Section 2, we introduce the notation, present a prototype interval global optimization algorithm and discuss the general discarding tests commonly used in the interval analysis literature. In Section 3 we describe the new discarding tests: the best corner test, the monotonicity-border test and the reduction test. A computational study which shows the usefulness of the new tests is presented in Section 4: we apply the interval algorithm in [13] with the new tests to the obnoxious facility location model given in [13]. Finally, the last section contains the conclusions and an outline of the directions for future research.

2. Interval Analysis and Global Optimization

In this section, we briefly summarize the fundamental concepts of interval analysis which are needed for this paper. For more details, the interested reader is referred to [16, 27, 29].

Following the notation suggested by Kearfott [19] as an standard, boldface will denote intervals, lower case will be used for scalar quantities or vectors (vectors are then distinguished from components by use of subscripts), and upper case for vectors or matrices. Brackets '[·]' will delimit intervals, while parentheses '(·)' vectors and matrices. Underlines will denote lower bounds of intervals and overlines give upper bounds of intervals. For example, we may have the interval vector $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T$, where $\boldsymbol{x}_i = [\underline{x_i}, \overline{x_i}]$. The *width* of an interval $\boldsymbol{x}$ is denoted by $w(\boldsymbol{x}) = \overline{x} - \underline{x}$ and its relative width by $w_{\text{relat}}(\boldsymbol{x}) = w(\boldsymbol{x})/\min\{|x| : x \in \boldsymbol{x}\}$ if

$0 \notin \boldsymbol{x}$ and $w(\boldsymbol{x})$ otherwise. The width of an interval vector $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T$ is to be understood as $w(\boldsymbol{x}) = \max\{w(\boldsymbol{x}_i) : i = 1, \ldots, n\}$. The set of intervals will be denoted by $\mathbb{I}$, and the set of n-dimensional interval vectors, also called *boxes*, by $\mathbb{I}^n$.

The *interval arithmetic operations* are defined by

$$\boldsymbol{x} * \boldsymbol{y} = \{x * y : x \in \boldsymbol{x}, y \in \boldsymbol{y}\} \text{ for } \boldsymbol{x}, \boldsymbol{y} \in \mathbb{I}, \tag{1}$$

where the symbol $*$ stands for $+$, $-$, $\cdot$ and $/$, and where $\boldsymbol{x}/\boldsymbol{y}$ is only defined if $0 \notin \boldsymbol{y}$. Definition (1) is equivalent to simple constructive rules (see [16, 27, 29]). The algebraic properties of (1) are different from those of real arithmetic operations (for instance, the subtraction and division in $\mathbb{I}$ are not the inverse operations of addition and multiplication, respectively), but the main properties from the operational point of view still hold, as for instance the so-called subdistributive law,

$$\boldsymbol{x} \cdot (\boldsymbol{y} + \boldsymbol{z}) \subseteq \boldsymbol{x} \cdot \boldsymbol{y} + \boldsymbol{x} \cdot \boldsymbol{z} \text{ for } \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{I},$$

and the *inclusion isotonicity*,

$$\boldsymbol{x} \subseteq \boldsymbol{y}, \boldsymbol{z} \subseteq \boldsymbol{t} \Longrightarrow \boldsymbol{x} * \boldsymbol{z} \subseteq \boldsymbol{y} * \boldsymbol{t} \text{ (if } \boldsymbol{y} * \boldsymbol{t} \text{ is defined) for } \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{t} \in \mathbb{I}.$$

The inclusion isotonicity is implicitly used in the construction of *inclusion functions*, which are the main interval arithmetic tool applied to optimization methods.

DEFINITION 1. *A function* $\boldsymbol{f} : \mathbb{I}^n \to \mathbb{I}$ *is said to be an inclusion function of* $f : \mathbb{R}^n \to \mathbb{R}$ *provided*

$$\{f(x) : x \in \boldsymbol{x}\} \subseteq \boldsymbol{f}(\boldsymbol{x})$$

for all boxes $\boldsymbol{x} \subset \mathbb{I}^n$ *within the domain of* $\boldsymbol{f}$.

Observe that if $\boldsymbol{f}$ is an inclusion function for f then we can directly obtain lower bounds and upper bounds of f over any box $\boldsymbol{x}$ within the domain of f just by taking $\underline{\boldsymbol{f}}(\boldsymbol{x})$ and $\overline{\boldsymbol{f}}(\boldsymbol{x})$, respectively.

For a function h predeclared in some programming language (like sin, exp, etc.), it is not too difficult to obtain a *predeclared* inclusion function $\boldsymbol{h}$, since the monotonicity intervals of predeclared functions are well known and then we can take $\boldsymbol{h}(\boldsymbol{x}) = \{h(x) : x \in \boldsymbol{x}\}$ for any $\boldsymbol{x} \in \mathbb{I}$ in the domain of h. For a general function $f(x)$, $x \in \mathbb{R}^n$, the easiest method to obtain an inclusion function is the *natural interval extension*, which is obtained by replacing each occurrence of the variable x with a box including it, $\boldsymbol{x}$, each occurrence of a predeclared function h by its corresponding inclusion function $\boldsymbol{h}$, and the real arithmetic operators by the corresponding interval operators.

The prototype interval algorithm for solving the general global optimization problem

$$\begin{cases} \min & f(x) \\ \text{s.t.} & g_j(x) \leqslant 0, i = 1, \ldots, r \\ & x \in \boldsymbol{X}_0 \end{cases} \tag{2}$$

where $\boldsymbol{X}_0$ is an initial box containing the region to which the search for optimal points can be restricted, is as follows.

Prototype algorithm

1. $\boldsymbol{Y} \leftarrow \boldsymbol{X}_0$, initialize the lists $\mathcal{L}_{\mathcal{W}} \leftarrow \emptyset$, $\mathcal{L}_{\mathcal{S}} \leftarrow \emptyset$.
2. Choose coordinate directions to split $\boldsymbol{Y}$.
3. Split $\boldsymbol{Y}$ normal to the chosen directions, cutting the box a given number of times in each direction. Let $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_s$ be the subboxes obtained.
4. For $i = 1$ to s do
   4.1. Delete $\boldsymbol{Y}_i$ if it can be proven that $\boldsymbol{Y}_i$ contains no solution or diminish $\boldsymbol{Y}_i$ if it can be proven that a part of $\boldsymbol{Y}_i$ contains no solution.
   4.2. If $\boldsymbol{Y}_i$ is not deleted, then store it (as whole or diminished) into the working list $\mathcal{L}_{\mathcal{W}}$.
5. If $\mathcal{L}_{\mathcal{W}} = \emptyset$ then STOP.
6. Choose a box from $\mathcal{L}_{\mathcal{W}}$ and remove it from that list. Let $\boldsymbol{Y}$ denote the chosen box.
7. If $\boldsymbol{Y}$ satisfies any of the stopping criteria then enter $\boldsymbol{Y}$ in the solution list $\mathcal{L}_{\mathcal{S}}$ and go to Step 5 else go to Step 2.

Different methods can be derived depending on how the Steps 2, 3, 4, 6 and 7 are completed. In recent papers (see [5, 6, 31, 32]) Csendes and Ratz have studied rules for the selection of subdivision directions and they have empirically proved that the choice of some rules may reduce the CPU time required for solving different problems by around 20% as compared with the classical rule which selects the direction of maximum width. Berner [1] and Casado, García and Csendes [2] have studied different box-splitting strategies and they have proved that multisection may reduce by around 20% the computational time as compared with bisection. Nevertheless, these steps are not decisive in two-dimensional problems and no significant reduction can be obtained with them (see [8]). On the other hand, Berner [1] has proved that, in some sense, the most suitable box to choose from the working list is the one with the smallest lower bound $\underline{\boldsymbol{f}}(\boldsymbol{Y})$.

In this paper we deal with the most important of the steps of the algorithm, Step 4.1, i.e., with the tests for verifying that a box or a part of a box contains no optimal point. We can find in the literature several discarding tests which have proved their usefulness in different global optimization problems. We next discuss the most successful ones.

- *Midpoint test:* Every time a box $\boldsymbol{Y}$ is chosen from $\mathcal{L}_{\mathcal{W}}$, and provided that its midpoint c is certainly feasible, we compute $\overline{\boldsymbol{f}}(c)$ and update the best upper bound of the inclusion function of the objective function for a point interval, $\tilde{f}$. The midpoint test discards boxes whose objective function value at any of its points is greater than the current $\tilde{f}$, i.e., a box $\boldsymbol{Z}$ is removed if $\underline{\boldsymbol{f}}(\boldsymbol{Z}) > \tilde{f}$. Also, a new box $\boldsymbol{Z}$ must only be entered in $\mathcal{L}_{\mathcal{W}}$ if $\tilde{f} \geqslant \underline{\boldsymbol{f}}(\boldsymbol{Z})$ is satisfied.

The test remains valid if an arbitrary certainly feasible point $c' \in \boldsymbol{Y}$ is used instead of the midpoint of $\boldsymbol{Y}$.

- *Feasibility test:* Let us suppose the feasible set S is defined as in (2), by $S = \{x \in \boldsymbol{X}_0 : g_j(x) \leqslant 0, j = 1, \ldots, r\}$. We say that a box $\boldsymbol{Y}$ *certainly satisfies* the constraint $g_j(x) \leqslant 0$ if $\overline{\boldsymbol{g}_j}(\boldsymbol{Y}) \leqslant 0$ and that $\boldsymbol{Y}$ *does certainly not satisfy* it if $\underline{\boldsymbol{g}_j}(\boldsymbol{Y}) > 0$. A box $\boldsymbol{Y} \subseteq \boldsymbol{X}_0$ is said *certainly feasible* if it certainly satisfies all the constraints, *certainly infeasible* if it does certainly not satisfy at least one of the constraints, and *undetermined* otherwise. A box $\boldsymbol{Y} \subset \boldsymbol{X}_0$ is said *certainly strictly feasible* if $\overline{\boldsymbol{g}_j}(\boldsymbol{Y}) < 0$, $j = 1, \ldots, r$. The feasibility test discards boxes certainly infeasible. Notice that to apply this test we just need an inclusion function $\boldsymbol{g}_j$ for each of the functions g_j defining the constraints. The application of this test also provides information about the feasibility of a box.
- *Monotonicity test:* This test is used to figure out whether the objective function f is strictly monotone in a certainly feasible box $\boldsymbol{Y} \subseteq \boldsymbol{X}_0$. Then, $\boldsymbol{Y}$ cannot contain a global minimizer in its interior. Thus, a global minimizer can only lie on a side of $\boldsymbol{Y}$, if this side is part of a side of $\boldsymbol{X}_0$. This can be done with the use of interval analysis: if $\nabla \boldsymbol{f} = (\nabla \boldsymbol{f}_1, \ldots, \nabla \boldsymbol{f}_n)^T$ is an inclusion function for the gradient of the objective function ∇f, and for a certainly feasible box $\boldsymbol{Y} \subseteq \boldsymbol{X}_0$ we have that $0 \notin \nabla \boldsymbol{f}_i(\boldsymbol{Y})$ for any i, then $\boldsymbol{Y}$ can be deleted or reduced to one of its sides with respect to the i-th component.
- Other discarding tests, such as the *concavity test* (see [16, 29], which discards or reduces to one of its sides certainly feasible boxes in which the objective function cannot be convex at any of its points), or the *interval Newton step* (see [16, 29], similar to the usual real Newton step) require the objective function to be twice differentiable in the box $\boldsymbol{Y}$ to which they are applied and from the computational point of view, their usefulness is not sure when applied to location problems (see [8, 13]).

We want to point out that, except for the interval Newton step, none of the previous mentioned tests tries to discard or reduce undetermined boxes making use of the fact that the boxes are undetermined. The midpoint test can be applied to both certainly feasible boxes and undetermined boxes; it just evaluates the inclusion function $\boldsymbol{f}$ at the current box $\boldsymbol{Y}$ and removes the box if $\underline{\boldsymbol{f}}(\boldsymbol{Y}) > \tilde{f}$, but it does not make use of the fact that the box is undetermined. The feasibility test discards infeasible boxes and gives information about the feasibility of box, but it does not make use of the fact that the box is undetermined, either. The monotonicity test and the concavity test only work over feasible boxes. Only the interval Newton step when applied to an undetermined box makes use of this fact: it applies one step of an interval Newton algorithm to the system of equations defined by the Fritz John or the Karush–Kuhn–Tucker conditions. Unfortunately, from the computational point of view it is not clear how useful the interval Newton step is when applied to location problems (see [8, 13]).

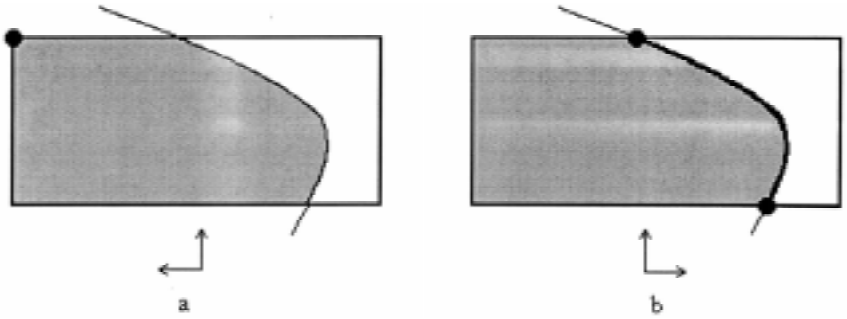

Figure 1. In gray, the part of the feasible set contained in the undetermined box. The arrows indicate the decreasing directions of the objective function. Optimal solution(s) can only lie in the thick lines or points.

The lack of tests operating over undetermined boxes is not surprising since most of the papers dealing with applications of interval analysis to Global Optimization consider unconstrained problems. Some exceptions are [3, 4, 8, 13, 15, 18, 20–22, 28].

Nevertheless, interval global optimization algorithms may have to process many undetermined boxes when they are applied to constrained problems. This is the case for location problems, where you may have to process a number of boxes on the border of the feasible set, specially when the optimal solution of the problem is on the border. The finding of new discarding tests operating over undetermined boxes is then a crucial step for accelerating interval constrained global optimization algorithms.

3. New discarding tests

The new discarding tests we present in this section are designed for two-dimensional problems (in particular, they can be applied to single-facility location problems) but they can be generalized for n-dimensional problems. They operate over undetermined boxes and require the objective function to be differentiable and monotonous at the boxes to which they are applied.

Before explaining them, notice that if $\boldsymbol{Y} \in \mathbb{I}^2$ is undetermined and f is monotonous at $\boldsymbol{Y}$, then the best feasible solution over the box, in case $\boldsymbol{Y}$ has feasible points, is in $(\partial S \cap \boldsymbol{Y}) \cup (\partial \boldsymbol{Y} \cap S)$.

3.1. BEST CORNER TEST

Let us suppose that f is differentiable at the box $\boldsymbol{Y}$ and that

$$0 \notin \nabla \boldsymbol{f}_i(\boldsymbol{Y}), \quad i = 1, 2, \tag{3}$$

where $\nabla \boldsymbol{f} = (\nabla \boldsymbol{f}_1, \nabla \boldsymbol{f}_2)^T$ is an inclusion function for the gradient of the objective function, ∇f. (3) implies that f is monotonous in both variables. Then, the

optimal point over $\boldsymbol{Y}$ is either at a corner of $\boldsymbol{Y}$ (in case the best corner of $\boldsymbol{Y}$ is a feasible point, see Figure 1a) or in $\partial S \cap \boldsymbol{Y}$ (see Figure 1b).

Given an undetermined box $\boldsymbol{Y}$ satisfying (3), the best corner test studies whether its best corner is feasible. If so, the box is reduced to that corner. Finding the best corner of a box $\boldsymbol{Y}$ satisfying (3) is not difficult, since we know the monotonicity of f in each of its components. For instance, if $\nabla \boldsymbol{f}_1(\boldsymbol{Y}) > 0$ and $\overline{\nabla \boldsymbol{f}_2}(\boldsymbol{Y}) < 0$ then the upper-left corner is the best one (see Figure 1a).

The generalization of this test to the n-dimensional case is straightforward. When the best corner of a box $\boldsymbol{Y}$ satisfying (3) is not feasible, we still know that the best point of $\boldsymbol{Y}$ is in $\partial S \cap \boldsymbol{Y}$. This fact could be used to find optimal points on that one-dimensional set, although in general this may be as difficult as solving the original problem. Only in some special cases, as the one described next, can the mentioned search be done efficiently.

3.2. MONOTONICITY-BORDER TEST

Let us suppose that f is differentiable at the undetermined box $\boldsymbol{Y}$ and that:

c1. $\boldsymbol{Y}$ satisfies (3), i.e., $0 \notin \nabla \boldsymbol{f}_i(\boldsymbol{Y}),\ i = 1, 2,$

c2. the best corner of $\boldsymbol{Y}$ is not feasible, and

c3. $\boldsymbol{Y}$ certainly satisfies all the constraints except one linear constraint.

Condition *c3* may seem to be rather restrictive, but for some kind of problems it is fulfilled. This is the case for continuous location problems. Depending on the geographical region where the new facility is to be located, the feasible set of a location problem may have a very general shape. In order to be able to formulate it as a set of analytical constraints, the usual method is to approximate the region by (non-convex) polygons (that may contain polygonal holes) and then to decompose them into easy-to-describe sets. The decomposition into convex polygons is one of the best options since convex polygons are not easy-to-describe, but they also have good optimization properties (see [8–11]). In fact, many location models suppose that the geographical region is given by a union of convex polygons. Although at the beginning of the algorithm the boxes may not certainly satisfy some constraints, as the algorithm goes on and the boxes become smaller many undetermined boxes will satisfy *c3*.

If $\boldsymbol{Y}$ is a box satisfying conditions *c1*, *c2* and *c3*, then the line defining the linear constraint that $\boldsymbol{Y}$ does not certainly satisfy mostly cuts the border of $\boldsymbol{Y}$ at two different points a and b. As explained before, the best feasible point over $\boldsymbol{Y}$ is on the segment ab (see Figure 2).

The monotonicity of f along segment ab (in the direction from a to b) is given by the directional derivative of f along the vector $\vec{v} = b - a$, $D_{\vec{v}} f$. On the other hand, if we denote by $\mathrm{d}f(x)$ the differential function of f at x, then we know that

$$D_{\vec{v}} f(x) = \lim_{h \downarrow 0} \frac{f(x + h\vec{v}) - f(x)}{h} = \mathrm{d}f(x)(\vec{v}) = \nabla f(x)^T \cdot \vec{v}.$$

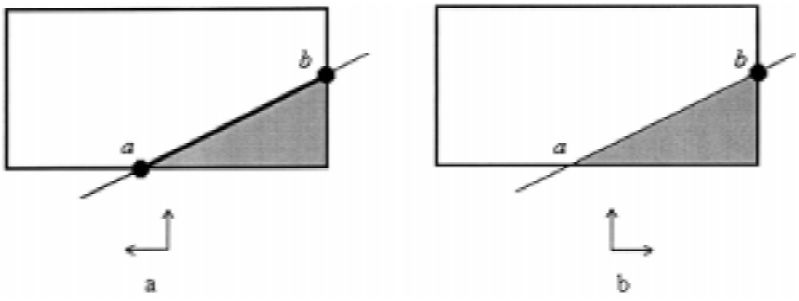

Figure 2. Monotonicity-border test. The arrows indicate the decreasing directions of the objective function. Optimal solution(s) can only lie in the thick lines or points.

Then, if $\boldsymbol{Z} \in \mathbb{I}^2$ is a box containing both a and b, the interval $\boldsymbol{t}$ defined by

$$\boldsymbol{t} = \nabla \boldsymbol{f}(\boldsymbol{Z})^T \cdot \vec{v}$$

is an inclusion for the directional derivative of f along segment ab in the direction given by the vector $\vec{v} = b - a$. Although the smaller the box $\boldsymbol{Z}$, the smaller the inclusion, in practice it is better to use the box $\boldsymbol{Y}$ if we have already calculated $\nabla \boldsymbol{f}(\boldsymbol{Y})$, since in this way we can avoid the calculation of $\nabla \boldsymbol{f}(\boldsymbol{Z})$. Also, an inclusion of $\vec{v}$ must be used to keep the reliability.

- If $\underline{\boldsymbol{t}} > 0$, f is an increasing function along segment ab (from a to b), so a is the best feasible point of $\boldsymbol{Y}$.
- If $\overline{\boldsymbol{t}} < 0$, f is a decreasing function along segment ab, so b is the best feasible point of $\boldsymbol{Y}$.
- If $0 \in \boldsymbol{t}$, nothing can be said about where the best point of $\boldsymbol{Y}$ lies.

The monotonicity-border test checks whether one of the first two cases described above occurs. If so, $\boldsymbol{Y}$ is reduced to the corresponding end-point of ab. In the third case, we still know that the best point of $\boldsymbol{Y}$ is on the segment ab. This could be used to carry out a line search for optimal points, using one of the available methods for minimizing functions of a single variable. Notice that we should use methods able to obtain global solutions.

The monotonicity-border test allows to detect situations as in Figure 2b, but it may also solve situations as in Figure 2a when f is monotonous along the segment ab, maybe because the slope of f along one of the axis is much greater than along the other axis. Notice that what is important is the slope along the segment.

The generalization of this test to the n-dimensional case can be done as follows. Let us suppose that f is differentiable in the undetermined box $\boldsymbol{Y} \subseteq \boldsymbol{X}_0 \subseteq \mathbb{R}^n$ and that $\boldsymbol{Y}$ satisfies conditions *c1* (which, for the n-dimensional case, is: $0 \notin \nabla \boldsymbol{f}_i(\boldsymbol{Y}),\ i = 1, \ldots, n$), *c2* and *c3*. Then the hyperplane π defined by the linear constraint that $\boldsymbol{Y}$ does not certainly satisfy mostly cuts the border of $\boldsymbol{Y}$ at s different points, $p_1, \ldots, p_s$, with $s \geqslant n$. A vectorial base of the vectorial subspace giving the direction of the linear subspace defined by $\pi \cap \boldsymbol{Y}$ is composed by at most h of the s vectors $\vec{v}_1 = p_1 - p_2, \vec{v}_2 = p_2 - p_3, \ldots, \vec{v}_s = p_s - p_1$, with $h \leqslant n - 1$.

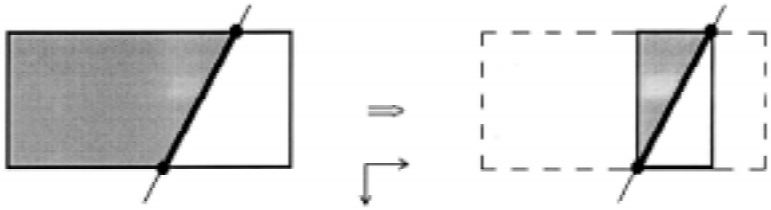

Figure 3. Reduction test. The arrows indicate the decreasing directions of the objective function. Optimal solution(s) can only lie in the thick segment, so we only need to consider the smallest box containing it.

For simplicity of notation, let us suppose that the base is formed by $\vec{v}_1, \ldots, \vec{v}_h$. Let $\boldsymbol{Z} \in \mathbb{R}^{h+1}$ denote the smallest box containing the points $p_1, \ldots, p_{h+1}$. If $0 \notin \nabla \boldsymbol{f}(\boldsymbol{Z})^T \cdot \vec{v}_i, i = 1, \ldots, h$, then f is monotonous on $\boldsymbol{Z}$ along each of the directions given by vectors $\vec{v}_i, i = 1, \ldots, h$. So, one of the points $p_1, \ldots, p_{h+1}$ is the best feasible point of $\boldsymbol{Y}$. The monotonicity-border test checks whether the condition $0 \notin \nabla \boldsymbol{f}(\boldsymbol{Z})^T \cdot \vec{v}_i$ holds for $i = 1, \ldots, h$. If it is the case, then $\boldsymbol{Y}$ is reduced to the corresponding p_i.

3.3. REDUCTION TEST

Let us suppose that f is differentiable at the undetermined box $\boldsymbol{Y}$ and that $\boldsymbol{Y}$ satisfies conditions *c1*, *c2* and *c3*. As explained before, the best point of $\boldsymbol{Y}$ is on the segment ab. The reduction test reduces $\boldsymbol{Y}$ to the smallest box containing ab (see Figure 3). Although very simple, the reduction test may be in practice very useful, since it may significantly reduce the size of a box. This in turn allows to obtain better inclusions and other discarding tests to be more efficient. The generalization of this test to the n-dimensional case is straightforward. In that case, the box $\boldsymbol{Y}$ is reduced to the smallest box containing the intersection points between the hyperplane defining the linear constraint which the box does not certainly satisfy and the sides of the box.

This test can be applied either after the monotonicity-border test when that test does not reduce the box to one of the end-points of segment ab, or alone, without using the monotonicity-border test. Since both tests require the same information, the use of the monotonicity-border test before the reduction test is recommended.

As for the order of application of discarding tests, we recommend the following order:

1. Midpoint test,
2. Feasibility test,
   a) If the box is certainly feasible:
      3. Monotonicity test,
      4. Concavity test,
      5. Newton step (applied to $\nabla f(x) = 0$).
   b) If the box is undetermined:

3. Best corner test,
4. Monotonicity-border test,
5. Reduction test,
6. Newton step (applied to the Fritz John conditions).

4. Computational study

Usually, papers dealing with applications of interval analysis to Global Optimization of constrained problems do not present computational experiments, except just one or two examples to show how the algorithm or a given step works. Some exceptions are [3, 4, 8, 13, 15, 18]. In [20–22], Kearfott presents some results for equality constrained problems, but his aim was to analyze different methods able to prove the existence of feasible points in a given box.

In this section, first we give a detailed numerical example to show how the discarding tests work. Then we present some computational experiments to show the efficiency of the new tests. Especially, we have used them in an interval algorithm for solving the obnoxious facility location model presented in [13]. Its aim is to locate, within a geographical region, a facility considered undesirable by the inhabitants of the region, but which is not noxious for the health of the people, in the sense that its effects do not endanger peoples' lives, at least directly or in a sudden way. The objective is to minimize the global repulsion of the inhabitants of the region to the location of the facility while taking into account environmental concerns which make some areas unsuitable for the location of the facility.

The repulsion of the inhabitants at the city a_i when the facility is located at the point x is modeled by the function

$$\mathrm{rp}(a_i, x) = \frac{1}{1 + \exp(\alpha_i + \beta_i \cdot d_i(a_i, x))},$$

with $\beta_i > 0$, where $d_i(a_i, x)$ is a measure of the distance between a_i and x, and α_i and β_i are two parameters given a priori for every existing city $a_i, i = 1, \ldots, m$. The lower the value of α_i, the higher the repulsion of the inhabitants to the location of the facility near their city or its outlying areas, and the higher the value of β_i, the faster is the change in opinion from considering a distance non-acceptable to acceptable.

The objective of the model is to minimize the function

$$f(x) = \sum_{i=1}^{m} \omega_i \cdot \mathrm{rp}(a_i, x)$$

where ω_i is a positive weight proportional to the importance of the city a_i, usually related to its number of inhabitants.

Furthermore, around each city there is a forbidden area where the location is not allowed. This is to avoid the possibility of the facility being located in one of

Table 1. Parameters associated to each city

City	a_{1i}	a_{2i}	α_i	β_i	ω_i	r_i^2
1	0.96	5.23	0.14	3.01	5.4	0.072
2	4.58	9.61	−2.06	4.71	7.5	0.140
3	3.43	0.24	−0.32	4.62	7.5	0.141
4	0.29	9.45	−4.92	3.20	9.2	0.212
5	9.30	5.69	−4.67	1.03	7.4	0.137
6	1.86	8.41	−4.58	2.18	4.0	0.040
7	3.74	8.09	−0.93	1.92	2.1	0.011
8	6.06	5.74	−1.67	3.96	8.3	0.171
9	4.40	8.24	−4.23	1.21	4.4	0.048
10	1.42	5.48	−0.32	3.84	9.5	0.227

Table 2. Coordinates of the vertices $v_i = (v_{1i}, v_{2i})$ of the convex polygon

Vertex	1	2	3	4	5	6	7	8	9	10
v_{1i}	7	10	10	9	5	3	−1	−2	−1	1
v_{2i}	9	8	6	2	0	0	1	2	5	10

the cities or too close to them, since the minisum objective may cause this kind of result. Protected areas where the location is not allowed, and/or other areas where the location is not possible, are taken into account in the model by considering them as forbidden areas. The forbidden areas can have any shape, the only requirement being to be able to specify the final feasible set S through a set of analytical constraints, $g_j(x) \leqslant 0$, $j = 1, \ldots, r$.

4.1. A NUMERICAL EXAMPLE

We next illustrate the steps of the algorithm with a small numerical example consisting of ten cities. Their coordinates $a_i = (a_{1i}, a_{2i})$ and the values of the parameters α_i, β_i and ω_i are given in Table 1. The protected area around each city consists of a circle centered at the city and radius r_i, as given in Table 1. The l_{2b}-norm, given by $l_{2b}(x) = \sqrt{b_1(x_1)^2 + b_2(x_2)^2}$, with parameters $b_1 = 1.98$ and $b_2 = 1.49$, is used as distance function for all the cities. The border of the feasible set is given by a convex polygon of 10 edges. The coordinates of the vertices of that polygon are given in Table 2. In all, 20 constraints form the feasible set (see Figure 4). The initial box was $\boldsymbol{X}_0 = ([-2.0, 11.0], [-1.0, 10.5])$.

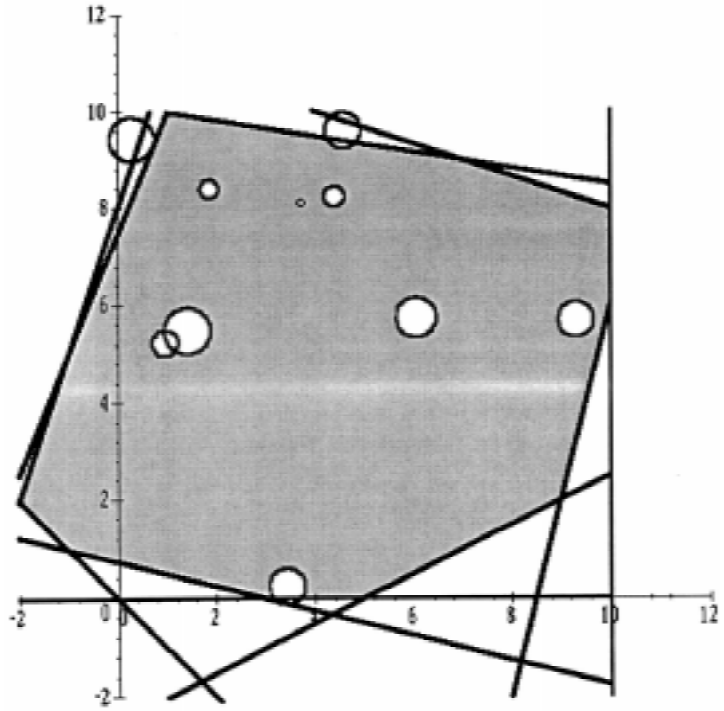

Figure 4. Feasible set (in grey) of the numerical example. The cities are located at the center of the circles.

We have used the interval algorithm which is obtained from the prototype algorithm using the bisection normal to the coordinate direction of maximum width as subdivision method, the box $\boldsymbol{Y}$ with the smallest lower bound $\underline{\boldsymbol{f}}(\boldsymbol{Y})$ as the next box out of $\mathcal{L}_w$, $w(\boldsymbol{Y}) < \epsilon_X$ and $w_{\text{relat}}([\underline{\boldsymbol{f}}(\boldsymbol{Y}), \min\{\tilde{f}, \overline{\boldsymbol{f}}(\boldsymbol{Y})\}) < \epsilon_f$ as criteria to send a box to $\mathcal{L}_s$, with $\epsilon_X = \epsilon_f = 0.0001$, and the midpoint test, the feasibility test, the monotonicity test and the three new discarding tests described in Section 3 as discarding tests. We will refer to this algorithm as AlgNDT. The implementation of the new tests was done as follows. If $\boldsymbol{Y}$ is an undetermined box, AlgNDT investigates whether $\boldsymbol{Y}$ certainly satisfies all the constraints except one linear constraint. If it is the case, an inclusion for the gradient of the objective function over that box is calculated, $\nabla \boldsymbol{f}(\boldsymbol{Y})$, using automatic differentiation [26], and the best corner test, the monotonicity-border test and the reduction test are applied to the box, in that order. Otherwise, the box is added to $\mathcal{L}_w$. Although it is possible to apply the best corner test to boxes on which more constraints are hurt, we decided to apply it in this way for two reasons: first, we are sure that f is differentiable at that boxes (the only points where it is not are at the existing cities a_i, but these points are enclosed by forbidden regions); second, if the best corner test does not reduce the box $\boldsymbol{Y}$ to one of its corners, we still can use the calculated inclusion $\nabla \boldsymbol{f}(\boldsymbol{Y})$ to try to reduce

the box with one of the other two tests. We just describe the iterations in which a given test first removes or reduces a box.

In the first iteration, the initial box $\boldsymbol{X}_0$ is bisected normal to direction x_1, yielding boxes $\boldsymbol{Y}_1 = ([-2.0, 4.5], [-1.0, 10.5])$ and $\boldsymbol{Y}_2 = ([4.5, 11.0], [-1.0, 10.5])$. Both boxes are undetermined and none of them can be deleted or diminished by any of the tests, so they are sent to $\mathcal{L}_W$. In Step 6, $\boldsymbol{Y}_1$ is chosen as the next box $\boldsymbol{Y}$ to work with, since $\underline{\boldsymbol{f}}(\boldsymbol{Y}_1) = 0.000015841827 < \underline{\boldsymbol{f}}(\boldsymbol{Y}_2) = 0.014330285759$. Its midpoint c is certainly feasible and allows us to update $\tilde{f}$, which now is set to $\tilde{f} = \overline{\boldsymbol{f}}(c) = 1.057474197846$. The algorithm goes to Step 2 with the current box $\boldsymbol{Y}$.

In iteration 2, the two boxes into which $\boldsymbol{Y}$ is split are undetermined and cannot be deleted, so they are sent to $\mathcal{L}_W$. The subbox $([-2.0, 4.5], [-1.0, 4.75])$ is chosen as the next box $\boldsymbol{Y}$ to work with. Its midpoint is certainly feasible and allows us to update $\tilde{f}$, which now is set to 0.008671288888. With this new $\tilde{f}$, the midpoint test can remove the box $([4.5, 11.0], [-1.0, 10.5])$ from $\mathcal{L}_W$.

In iteration 6, the box to work with is $\boldsymbol{Y} = ([-2.0, -0.375], [-1.0, 1.875])$. The first box into which it is split, $\boldsymbol{Y}_1 = ([-2.0, -0.375], [-1.0, 0.4375])$, is removed by the feasibility test, since it does not certainly satisfy the linear constraint defined by the vertices v_6 and v_7 of the convex polygon.

In iteration 9, the second subbox $\boldsymbol{Y}_2 = ([-0.375, 1.25], [0.4375, 1.875])$ into which the current box $\boldsymbol{Y}$ is split satisfies all the constraints except one linear constraint (the one defined by the vertices v_6 and v_7), but it does not satisfy condition *c1*, so we do not apply the new discarding tests.

It is in iteration 11, when we first find a subbox, $\boldsymbol{Y}_2 = ([-1.59375, -1.1875], [1.15625, 1.875])$, satisfying all the constraints except one linear constraint (the one defined by the vertices v_7 and v_8) and also satisfying condition *c1*. Its best corner is not certainly feasible, and the monotonicity-border test cannot reduce the box to a point. Finally, the reduction test reduces the box to $([-1.59375, -1.1875], [1.1875, 1.59375])$.

The second subbox generated in iteration 18, $\boldsymbol{Y}_2 = ([-0.375, 1.25], [1.875, 4.75])$, is not entered in $\mathcal{L}_W$ since $\tilde{f} = 0.000283560783 < \underline{\boldsymbol{f}}(\boldsymbol{Y}_2) = 0.001507769701$, i.e., it is removed by the midpoint test.

The second subbox, $\boldsymbol{Y}_2 = ([-1.796875, -1.59375], [1.875, 2.234375])$, generated in iteration 24 is certainly strictly feasible and it is removed by the monotonicity test ($\nabla \boldsymbol{f}(\boldsymbol{Y}) = ([0.000184344345, 0.001144042639], [0.000120153351, 0.000823112810])$).

In iteration 25, the subbox $\boldsymbol{Y}_2 = ([-2.0, -1.796875], [2.0546875, 2.234375])$ satisfies all the constraints except one linear constraint (the one defined by the vertices v_8 and v_9) and it also satisfies *c1*. Its best corner is not certainly feasible, but the monotonicity-border test reduces the box to the point $(-1.981770833333, 2.0546875)$ (the inclusion for the directional derivative of f along the segment of the line defined by the linear constraint intersected by $\boldsymbol{Y}_2$ is $\boldsymbol{t} = [0.000581929348, 0.002257540398]$).

In iteration 38, the subbox $\boldsymbol{Y}_2 = ([-1.9873046875, -1.974609375], [1.9873046875, 2.009765625])$ is reduced to its best corner, the point $(-1.9873046875, 1.9873046875)$.

The algorithm stopped at iteration 54. It generated 108 boxes: 6 of them were removed by the feasibility test, 44 of them by the midpoint test and 1 of them by the monotonicity test, 1 box was reduced to a point by the best corner test and 9 by the monotonicity-border test, and 14 boxes were reduced by the reduction test. The best feasible point found by the algorithm was $(-1.99993896484375, 1.99993896484375)$ and the upper bound for the objective function was $\tilde{f} = 0.000224097581142$. Two undetermined boxes were given as result,

$$([-2.0, -1.999900817871094], [1.99993896484375, 2.000026702880859])$$

and

$$([-1.99993896484375, -1.99993896484375], [1.99993896484375, 1.99993896484375]).$$

4.2. COMPUTATIONAL EXPERIMENTS

For our experiments we have generated 300 problems according to the location model described above which have a convex polygon as border. Specifically, 100 problems have a convex polygon of 5 edges as border, for other 100 problems the convex polygons have 10 edges and in the last 100 problems the number of edges of the polygons is 20. Each of these three groups of 100 problems is formed by 5 subgroups of 20 problems, the number of cities considered in each subgroup being $m = 5$, 10, 15, 20 and 25, respectively. Each of the subgroups is formed by 10 problems in which the only non-linear constraints considered are the ones around the cities, and 10 problems which in addition also have other 20 non-linear constraints (which define the protected areas).

The non-linear constraint defining the forbidden area around a city a_i is a circle centered at a_i and with radius $\omega_i/20$. The other non-linear constraints were randomly chosen from 50 preset ones, i.e., 25 circles, 15 ellipses and 10 parabolas. The convex polygons were randomly chosen from 6 preset ones, all of them inside the initial box $\boldsymbol{X}_0$, which always was $([-2.0, 11.0], [-1.0, 10.5])$. The l_{2b}-norm was used as distance function (see [12]); b_1 and b_2 are the parameters of the distance function considered in the study. The parameters of the model were drawn randomly from uniform distributions defined on the following intervals: $b_1, b_2 \in [1.0, 2.5]$ (the distance function was the same for every city), $\alpha_i \in [-5, 1]$, $\beta_i \in [1, 6]$ and $\omega_i \in [1, 10]$. The coordinates of the cities were drawn from a uniform distribution on the rectangle $([-1.0, 10.5], [-0.5, 10.0])$.

Every problem was solved twice. First, we used an interval algorithm which works as AlgNDT but without the new discarding tests. We will refer to this algorithm as AlgGDT. Then, we solved the problems with AlgNDT.

The algorithms have been programmed in Pascal-XSC [23], and run under MS-DOS on a PC with an Intel Pentium processor with a CPU speed of 133 MHz. The Standard Time Unit (see [30]), i.e., the computation time needed to evaluate the Shekel-5 function 1000 times at the point (4,4,4,4) for that machine is 0.4394 seconds if we use a function with variables of type '`real`', and 2.9428 seconds if we use the corresponding natural interval extension (with variables of type '`interval`').

The results obtained with AlgGDT and AlgNDT when $\epsilon_X = \epsilon_f = 0.01$ are summarized in Tables 3 and 4, respectively. The given values correspond to the averages of the 20 problems of each of the subgroups with the same number of linear constraints (lc) and the same number of cities (m). The studied variables are the CPU time in seconds (time), number of boxes generated by the algorithm (bg), number of certainly feasible boxes in the solution list ($\mathcal{L}_{\mathcal{S}_\mathcal{F}}$), number of undetermined boxes in the solution list ($\mathcal{L}_{\mathcal{S}_\mathcal{U}}$), number of boxes discarded by the feasibility test (ft), by the midpoint test (mpt) and by the monotonicity test (mt), number of boxes reduced to a point by the best corner test (bct) and by the monotonicity-border test (mbt), number of boxes reduced by the reduction test (rt) and maximum number of boxes stored in $\mathcal{L}_\mathcal{W}$ at any time (ml).

AlgNDT obtained better results for each of the problems. Considering all the 300 problems, the relative reduction in time obtained by AlgNDT as compared with AlgGDT attains a considerable 21.21% and the reduction in the number of boxes generated by the algorithm is 43.47%. As we can see in Table 4, the best test at discarding boxes in AlgNDT is the midpoint test, followed by the feasibility test. Nevertheless, the 13.6 boxes that as average are reduced to a point by the monotonicity-border test and the 23.7 boxes whose size is reduced by the reduction test allow to reduce the number of boxes to be processed from 381.9 to 215.9, that is, 166 boxes. It follows that the area discarded by the new tests is difficult to reject by the general discarding tests. On the other hand, the best corner test does not seem to be efficient for this kind of problems, as shown by the small number of boxes it reduces to a point. Notice that we have only applied the best corner test to a box $\boldsymbol{Y}$ if it satisfies condition *c3*; since the only linear constraints in a given problem are the ones defining the exterior border, which was a convex polygon, the corner outside the feasible set is usually the best corner (from the objective function, the more far the point from the existing cities, the less the repulsion of the inhabitants). Nevertheless, applying the best corner test is not computationally costly: the most time-consuming step, obtaining the inclusion for the gradient, is also used in the monotonicity-border test and in the reduction test. In fact, all the undetermined boxes to which we applied the new tests were modified by one of them, which implies that no unnecessary work is done when applying the new tests. Concerning the maximum number of boxes stored in $\mathcal{L}_\mathcal{W}$ at any time, which can be thought as a measure of the memory complexity of the algorithms, we see that the algorithm which uses the new tests obtains a considerable relative reduction of 37.16% as compared with AlgGDT.

Table 3. Computational results using AlgGDT ($\epsilon_X = \epsilon_f = 0.01$)

lc	m	time	bg	$\mathcal{L}_{\mathcal{S}_\mathcal{F}}$	$\mathcal{L}_{\mathcal{S}_\mathcal{U}}$	ft	mpt	mt	ml
5	5	3.92	270.5	0.0	14.8	58.3	57.7	5.3	22.5
	10	11.37	392.6	0.0	27.1	77.1	84.6	8.4	33.2
	15	21.11	425.8	2.2	25.4	70.4	102.3	13.6	38.4
	20	27.86	472.8	0.0	33.7	85.3	108.3	10.0	41.1
	25	36.09	465.4	1.8	27.1	81.0	111.1	12.6	42.6
global average		20.07	405.4	0.8	25.7	74.4	92.8	10.0	35.6
10	5	3.97	260.9	0.0	14.8	45.9	63.9	6.7	21.6
	10	9.81	331.2	0.0	26.1	60.3	71.6	8.6	29.7
	15	11.89	271.8	0.0	17.6	49.6	63.7	6.0	21.6
	20	22.08	374.4	0.0	23.2	68.7	87.5	8.8	32.5
	25	27.17	366.0	0.0	29.1	60.6	84.9	9.4	33.3
global average		14.98	320.9	0.0	22.2	57.0	74.3	7.9	27.8
20	5	5.82	384.5	0.0	25.4	71.7	89.6	6.5	31.3
	10	13.00	447.6	0.0	32.8	87.2	94.8	9.9	38.3
	15	21.90	505.0	0.0	39.3	96.7	110.0	7.4	45.6
	20	20.30	356.2	0.0	22.3	69.0	82.2	5.5	29.5
	25	29.70	404.1	0.0	29.8	69.6	92.1	11.5	33.3
global average		18.15	419.5	0.0	29.9	78.8	93.8	8.2	36.0
all problems		17.73	381.9	0.3	25.9	70.1	87.0	8.7	33.1

So as to study the efficiency of the new tests when we increase the accuracy, we then solved the 300 problems considering $\epsilon_X = \epsilon_f = 0.001$ and then $\epsilon_X = \epsilon_f = 0.0001$. The results are summarized in Table 5. As we can see, the increase of accuracy clearly shows the usefulness of the new tests. When the required precisions are set to 0.001, the use of new tests allows 50.82% reduction in time, 67.05% in the number of boxes generated, and 68.16% in the maximum number of boxes stored in $\mathcal{L}_\mathcal{W}$ at any time. When the tolerances are set to 0.0001, the reductions are 74.76%, 83.40% and 86.81%, respectively. Notice also that whereas the average times for solving the problems with AlgGDT for the tolerances 0.01, 0.001 and 0.0001 are 17.73, 39.22 and 99.80, respectively, which implies an increase of more than twice the previous time for each unit of precision (exponential growth), the corresponding average times when using the new discarding tests are 13.97, 19.29 and 25.19, which implies a linear increase of around 5 seconds for each unit of precision.

Table 4. Computational results using AlgNDT ($\epsilon_X = \epsilon_f = 0.01$)

lc	m	time	bg	$\mathcal{L}_{\mathcal{S}_\mathcal{F}}$	$\mathcal{L}_{\mathcal{S}_\mathcal{U}}$	ft	mpt	mt	bct	mbt	rt	ml
5	5	3.15	169.3	0.0	5.6	32.6	40.0	2.1	0.4	8.9	14.0	14.9
	10	8.85	210.6	0.0	6.7	35.0	50.1	2.9	0.2	14.7	23.9	19.2
	15	18.19	280.5	2.2	7.6	38.2	75.7	10.1	0.1	9.7	19.2	28.6
	20	21.88	253.8	0.0	9.9	34.7	64.2	4.3	0.2	17.8	31.3	25.4
	25	30.47	287.0	1.8	10.3	39.4	75.0	9.0	0.3	11.3	32.6	33.15
global average		16.50	240.2	0.8	8.0	36.0	61.0	5.7	0.3	12.5	24.2	24.3
10	5	3.16	149.2	0.0	5.0	21.6	39.5	3.1	0.7	9.6	13.6	13.3
	10	7.09	160.2	0.0	6.5	22.1	38.2	3.6	0.8	13.7	17.1	14.7
	15	9.17	144.6	0.0	5.1	19.4	37.7	2.5	0.0	11.8	18.1	14.0
	20	16.40	189.7	0.0	6.2	25.4	48.4	4.9	0.3	14.1	23.1	19.2
	25	22.18	202.5	0.0	10.7	25.5	49.9	4.6	0.1	15.7	24.7	22.2
global average		11.60	169.2	0.0	6.7	22.8	42.8	3.8	0.4	13.0	19.3	16.7
20	5	4.79	244.6	0.0	10.7	40.1	58.5	3.8	1.2	11.9	26.0	21.4
	10	10.07	256.5	0.0	9.2	44.4	58.0	4.3	0.1	16.3	27.6	21.1
	15	15.37	252.8	0.0	13.5	38.7	56.2	3.7	0.1	17.7	40.6	26.8
	20	14.95	205.8	0.0	6.3	35.1	48.9	2.8	0.1	13.3	18.2	16.9
	25	23.89	231.9	0.0	7.8	31.4	57.6	6.2	0.1	17.2	25.4	20.7
global average		13.81	238.3	0.0	9.5	38.0	55.9	4.2	0.3	15.3	27.6	21.4
all problems		13.97	215.9	0.3	8.1	32.3	53.2	4.6	0.3	13.6	23.7	20.8

Table 5. Computational results for different required precision

$\epsilon_X = \epsilon_f$	algorithm	time	bg	$\mathcal{L}_{\mathcal{S}_\mathcal{F}}$	$\mathcal{L}_{\mathcal{S}_\mathcal{U}}$	ml
0.01	AlgGDT	17.73	381.9	0.3	25.9	33.1
	AlgNDT	13.97	215.9	0.3	8.1	20.8
0.001	AlgGDT	39.22	827.3	0.3	63.3	71.3
	AlgNDT	19.29	272.6	0.3	7.8	22.7
0.0001	AlgGDT	99.80	2015.7	0.3	168.6	178.2
	AlgNDT	25.19	334.7	0.3	8.3	23.5

Table 6. Computational results with AlgNDT for different required precision

$\epsilon_X = \epsilon_f$	time	bg	$\mathcal{L}_{\mathcal{S}_\mathcal{F}}$	$\mathcal{L}_{\mathcal{S}_\mathcal{U}}$	ft	mpt	mt	bct	mbt	rt	ml
0.01	13.97	215.9	0.3	8.1	32.3	53.2	4.6	0.3	13.6	23.7	20.8
0.001	19.29	272.6	0.3	7.8	36.6	61.4	5.8	0.6	31.7	39.7	22.7
0.0001	25.19	334.7	0.3	8.3	41.4	68.2	8.7	0.8	50.3	55.4	23.5

From Table 6, which give details on the number of boxes discarded or reduced by the different tests used in AlgNDT for the different tolerances, we can see that the smaller the ϵ, the higher the efficiency of the new tests. It justifies the improvements in time and boxes generated mentioned above. For instance, when $\epsilon_X = \epsilon_f = 0.01$, the feasibility test is better than the reduction test, which in turn is better than the monotonicity-border test; all of them are far behind the midpoint test. When $\epsilon_X = \epsilon_f = 0.001$, the monotonicity-border test and the reduction test are competitive with the feasibility test, although still less efficient than the midpoint test. When the required precision is 0.0001 the two new discarding tests are better than the feasibility test, and the difference in the efficiency with the midpoint test is smaller.

Finally, notice also that the number of boxes in the solution list is smaller when we use AlgNDT (see Tables 3 and 4 and Table 5). This means that the use of the new tests allows the interval algorithm to obtain better approximations of the solution set. The increase of accuracy makes this fact more obvious: whereas the number of final boxes with AlgGDT increases more than twice for ten times less ϵ, the number of final boxes with AlgNDT remains closely constant (see Table 5).

5. Summary, conclusions and future research

In [13], Fernández et al. presented a general method able to solve most of continuous location models in the literature. Their method is based on the application of interval analysis tools to Global Optimization. The key to the speed of global optimization algorithms based on Interval Analysis is their use of tests to discard boxes or parts of boxes in which no optimal point may occur. In this paper we have presented three new discarding tests which may speed up the convergence of the algorithm in [13] considerably. The algorithm and the new tests can also be used to solve other constrained two-dimensional problems.

The new tests, unlike most of the discarding tests in the literature, operate over undetermined boxes. The best corner test may reduce the box to one of its corners by studying the monotonicity of the objective function in each of its variables. The monotonicity-border test studies the monotonicity of the objective function along a segment of a line (defining a linear constraint of the feasible set) intersected by the box, and may reduce the box to one of the end-points of the intersected segment.

The reduction test reduces a box containing a segment as described before to the smallest box containing the segment. These new tests can be generalized to the n-dimensional case.

Through a computational study, it has been shown that the new tests are very efficient at discarding parts of the feasible set and that they succeed in accelerating the convergence of the algorithm substantially. Although these conclusions were obtained for a particular obnoxious facility location model, they may be valid for other continuous location or two-dimensional problems. The tests may also prove to be efficient in two-dimensional constrained problems for which the global maximum value and the global minimum value are not too different, since in that case the interval algorithm will probably have to split most of the feasible set in small boxes, and in particular, many undetermined boxes will have to be checked.

A direction for future research is the finding of new discarding tests for two-dimensional problems. In particular, a test for boxes which satisfy all the constraints except two linear constraints may be specially useful for location problems; such kind of boxes happens for instance at the intersection of two lines defining linear constraints. In fact, when the two lines intersect at a point P, we can use that vertex of the feasible set to split the box into four subboxes: a feasible one, an infeasible one, and two other boxes which satisfy all the constraints except one linear constraint. The finding of tests for boxes which satisfy all the constraints except one, not necessarily linear, can also be very useful. For the n-dimensional case, the design of discarding tests operating over undetermined boxes is still a nearly unexplored area, which deserve the attention in the future.

Acknowledgments

This work has been supported by the Fundación Séneca under grant PB/11/FS/97.

References

1. Berner, S. (1996), New Results on Verified Global Optimization, *Computing* 57: 323–343.
2. Casado, L.G., García, I. and Csendes, T., A New Multisection Technique in Interval Methods for Global Optimization, *Computing* (to appear).
3. Csallner, A.E. (1993), Global optimization in separation network synthesis, *Hungarian Journal of Industrial Chemistry* 21: 303–308.
4. Csendes, T. (1998), Optimization methods for process network synthesis – a case study. In Christer Carlsson and Inger Eriksson (eds.), *Global multiple criteria optimization and information systems quality*, Abo Academy, Turku, pp. 113–132.
5. Csendes, T. and Ratz, D. (1996), A review of subdivision direction selection in interval methods for global optimization, *ZAMM* 76: 319–322.
6. Csendes, T. and Ratz, D. (1997), Subdivision direction selection in interval methods for global optimization, *SIAM Journal on Numerical Analysis* 34: 922–938.
7. Drezner, Z. (1995), *Facility Location: a Survey of Applications and Methods*, Springer, Berlin.
8. Fernández, J. (1999), *Nuevas técnicas para el diseño y resolución de problemas de localización continua (New techniques for design and solution of continuous location models)*, Ph.D.

Thesis, Dpto. Estadística e Investigación Operativa, Universidad de Murcia, Murcia, Spain (in Spanish).
9. Fernández, J., Cánovas, L. and Pelegrín, B. (1997), DECOPOL-codes for decomposing a polygon into covex subpolygons, *European Journal of Operational Research* 102: 242–243.
10. Fernández, J., Cánovas, L. and Pelegrín, B. (2000), Algorithms for the decomposition of a polygon into convex polygons, *European Journal of Operational Research* 121: 330–342.
11. Fernández, J., Cánovas, L. and Pelegrín, B. (2000), Decomposition of a polygon with holes into convex polygons, Working paper 2000/2, Department of Statistics and Operations Research, University of Murcia, Murcia, Spain.
12. Fernández, J., Fernández, P. and Pelegrín, B. (1997), Estimating Actual Distances by Norm Functions: a Comparison between the $l_{k,p,\theta}$-norm and the $l_{b_1,b_2,\theta}$-norm and a Study about the Selection of the Data Set, *Computers and Operations Research* (to appear).
13. Fernández, J., Fernández, P. and Pelegrín, B. (2000), A continuous location model for siting a non-noxious undesirable facility within a geographical region, *European Journal of Operational Research* 121: 259–274.
14. Fernández, J. and Pelegrín, B. (2000), Sensitivity Analysis in Continuous Location Models via Interval Analysis, *Studies in Locational Analysis* 14: 121–136
15. Goos, A. and Ratz, D. (1997), Praktische Realisierung und Test eines Verifikationsverfahrens zur Loesung globaler Optimierungsprobleme mit Ungleichungsnebenbedingungen, *Forschungsschwerpunkt Computerarithmetik, Intervallrechnung und Numerische Algorithmen mit Ergebnisverifikation*, technical report (available at http://www.uni-karlsruhe.de/~iam/html/reports.html).
16. Hansen, E. (1992), *Global Optimization Using Interval Analysis*, Marcel Dekker, New York.
17. Hansen, P., Peeters, D., Richard, D. and Thisse, J.F. (1985), The minisum and minimax location problems revisited, *Operations Research* 33: 1251–1265.
18. Hua, J.Z., Brennecke, J.F. and Stadtherr, M.A. (1998), Enhanced interval analysis for phase stability: cubic equation of state models, *Industrial Engineering Chemical Research* 37: 1519–1527.
19. Kearfott, R.B. (1996), *Rigorous Global Search: Continuous Problems*, Kluwer, Dordrecht.
20. Kearfott, R.B. (1996), Test results for an interval branch and bound algorithm for equality-constrained optimization, in: *State of the art in global optimization*, Kluwer, Dordrecht, 181–199.
21. Kearfott, R.B. (1996), On Verifying Feasibility in Equality Constrained Optimization Problems, technical report (available at http://interval.louisiana.edu/preprints.html).
22. Kearfott, R.B. (1998), On Proving Existence of Feasible Points in Equality Constrained Optimization Problems, *Mathematical Programming* 83: 89–100.
23. Klatte, R., Kulisch, U., Neaga, M., Ratz, D. and Ullrich, C. (1992), *PASCAL-XSC – Language Reference with Examples*, Springer, Heidelberg.
24. Love, R.F., Morris, J.G. and Wesolowsky, G.O. (1988), *Facilities Location: Models and Methods*, North Holland, New York.
25. Plastria, F. (1992), GBSSS: The Generalized Big Square Small Square Method for Planar Single-Facility Location, *European Journal of Operational Research* 62: 163–174.
26. Rall, L.B. (1981), *Automatic Differentiation, Techniques and Applications*, Lecture Notes in Computer Science, No. 120, Springer, Berlin.
27. Ratschek, H. and Rokne, J. (1988), *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester.
28. Ratschek, H. and Rokne, J. (1993), Experiments using interval analysis for solving a circuit design problem, *Journal of Global Optimization* 3: 501–518.
29. Ratschek, H. and Voller, R.L. (1991), What can Interval Analysis do for Global Optimization?, *Journal of Global Optimization* 1: 111-130.

30. Ratz, D. (1994), Box-splitting strategies for the interval Gauss-Seidel step in a global optimization method, *Computing* 53: 337–353.
31. Ratz, D. (1996), On Branching Rules in Second-Order Branch-and-Bound Methods for Global Optimization, in: Alefeld, Frommer and Lang (eds.), *Scientific Computing and Validated Numerics*, Akademie-Verlag, pp. 221–227.
32. Ratz D. and Csendes T. (1995), On the Selection of Subdivision Directions in Interval Branch-and-Bound Methods for Global Optimization, *Journal of Global Optimization* 7: 183–207.