RAIRO Operations Research
RAIRO Oper. Res. **40** (2006) 285–302
DOI: 10.1051/ro:2006024

CONVEX QUADRATIC UNDERESTIMATION AND BRANCH AND BOUND FOR UNIVARIATE GLOBAL OPTIMIZATION WITH ONE NONCONVEX CONSTRAINT *

Hoai An Le Thi[1] and Mohand Ouanes[1, 2]

Abstract. The purpose of this paper is to demonstrate that, for globally minimize one dimensional nonconvex problems with both twice differentiable function and constraint, we can propose an efficient algorithm based on Branch and Bound techniques. The method is first displayed in the simple case with an interval constraint. The extension is displayed afterwards to the general case with an additional nonconvex twice differentiable constraint. A quadratic bounding function which is better than the well known linear underestimator is proposed while $w-$subdivision is added to support the branching procedure. Computational results on several and various types of functions show the efficiency of our algorithms and their superiority with respect to the existing methods.

Keywords. Global optimization, branch and bound, quadratic underestimation, $w-$subdivision.

1. Introduction

In recent years there has been a very active research in global optimization whose main tools and solution methods are developed *via* four basic approaches:

Received September 6, 2005. Accepted April 6, 2006.

* *This work is partially supported by Conseil Régional Lorraine via the project "Optimisation et Aide à la Décision dans les Systèmes d'Information".*

[1] Laboratoire de l'Informatique Théorique et Appliquée, UFR Scientifique MIM Université Paul Verlaine – Metz, Ile du Saulcy, 57045 Metz, France; lethi@univ-metz.fr

[2] Département de Mathématiques, Faculté des Sciences, Université de Tizi-Ouzou, Algeria.

outer approximation, cutting plane, branch and bound and interval analysis (see *e.g.* [2,8,13,14,19,20,24,25,28,30,33] and references therein) in combining with, in some cases, the reformulation techniques. Extensive surveys on global optimization exist in the literature (see *e.g*, [33] and recently [25]). In this work we focus on the univariate global optimization. Univariate global optimization problems attract attention of researchers not only because they arise in many real-life applications but also the methods for these problems are useful for the extension to the multivariable case.

The univariate global optimization problem with simple constraint is written in the form

$$(P) \quad \begin{cases} \alpha := \min f(s) \\ s \in S, \end{cases}$$

where $S := [a, b]$ is a bounded closed interval and f is a continuous function on S. Only the case where f is not concave function will be considered, because when f is concave the problem is trivial: $\min\{f(s) : s \in [a, b]\} = \min\{f(a), f(b)\}$.

With an additional nonconvex constraint, Problem (P) becomes the univariate nonconvex constrained global optimization problem which takes the form

$$(PC) \quad \begin{cases} \alpha_g := \min f(s) \\ s \in S \\ g(s) \leq 0, \end{cases}$$

where g is a nonconvex function on S.

Several methods have been studied in the literature for univariate global optimization problems. Let us mention the works for the Polynomial and Rational functions [15,34], the Hölder functions [12], the Lipschitz functions [18], and those in [4, 7, 16, 22, 29, 31–37]. As we know, the most part of existing algorithms solve Problem (P) while Problem (PC) is still difficult. In [34] Floudas and Visweswaran specialized their GOP algorithm developed in [8] (a decomposition based global optimization algorithm for solving constrained nonconvex non linear programming problems) to find efficiently the global minimum of univariate polynomial functions over an interval. In [15] the authors showed that the specialized version of GOP in [34] is equivalent to a version of an interval arithmetic algorithm which uses natural extension of the cord-slope form of Taylor's expansion for univariate polynomial functions, and studied other versions of the interval arithmetic algorithm corresponding to different ways to evaluate the bounds on the cord-slope. In [12] a piecewise convex lower-bounding function is used for solving (P) when f is a Hölder function. In [35] a class of differentiable functions with a Lipschitz constant for the first derivative is considered where the lower bounding function is constructed from smooth auxiliary functions and connecting points. In [36] a three-phase algorithm using an improved linear lower bounding function is investigated for Problem (P). The authors improved the linear lower bounding function (LLBF) introduced in their previous work and developed three phases: a basic way to remove a subregion not containing an ε-global solution by LLBF for the global phase, a local minimization algorithm in the local phase, and the

phase of checking the global solution. Numerical results reported there show that this is a promising approach, it is more efficient than several existing methods. In [7], Lipschitz univariate constrained global optimization problems where both the objective function and constraints can be multi-extremal are considered with Pijavskii's method [26].

The aim of this paper is to study branch and bound approaches for solving both Problems (P) and (PC). Throughout the paper we suppose that f and g are twice differentiable on S on which their second derivatives are bounded, *i.e.* there are positive numbers K and K_g such that $|f"(s)| \leq K$ and $|g"(s)| \leq K_g$ for all $s \in S$. Such a K and a K_g can be defined by several ways in practice: either it is possible to know *a priori* these values, or they are estimated in a way during the algorithm. In our approaches, K and K_g are assumed to be known.

Problems of this type have many applications in various fields [19] and particularly in the economy and finance. We introduce a quadratic lower bounding function which is better than the well known linear underestimator of f by the theory of approximation [3]. This bounding procedure suggests us an efficient $w-$subdivision in the branching. The algorithm has a finite convergence for obtaining an ε-solution.

There are several advantages of the proposed methods. Firstly, the lower bound is computed by a *very inexpensive way*: the quadratic lower bounding function is well determined from the constant K, and then all calculations are explicit. We do not need more information as in other methods such as the connecting points in [12] and [35]. Secondly, the minimizer of the quadratic underestimation can be used efficiently for the branching procedure. Thirdly, the algorithm is simple and easy to implement. It is composed of one phase, and not complicated as the three- phase algorithm [36]. We have compared our method with standard existing algorithms (Branch and Bound and/or Interval Arithmetic methods) for several classes of problems: the Polynomial and Rational functions [15, 34], the Hölder functions [12], the multi-extremal functions, [36, 37], and the Lipschitz functions with multi-extremal constraint [7, 26]. As it will be seen in the last section concerning numerical results, our algorithm is more efficient than the related standard methods. Both algorithms for solving (P) and (PC) work very well on a collection of test problems for univariate global optimization [7, 9, 12, 15, 35–37].

The structure of the paper is as follows. Section 2 discusses the branch and bound algorithm for (P). The algorithm for solving (PC) is developed in Section 3. Numerical examples and comparative computational results with several existing algorithms are presented in Section 4.

2. Solving simple constrained global optimization problems

We develop in this section a branch and bound algorithm for solving Problem (P). Before describing the algorithm and proving its convergence we have to specify the bounding and branching operations.

2.1. Bounding procedures

2.1.1. *An affine underestimating function*

Let $\{s_1, s_2, ..., s_m\}$ be a uniform discretization with mesh size h of $S = [a, b]$ where $s_1 = a$ and $s_m = b$. Let $\{w_1, w_2, ..., w_m\}$ be a finite sequence of functions defined as [3,5] ($m \geq 2$)

$$w_i(s) = \begin{cases} \frac{s - s_{i-1}}{s_i - s_{i-1}} \text{ if } s_{i-1} \leq s \leq s_i, \\ \frac{s_{i+1} - s}{s_{i+1} - s_i} \text{ if } s_i \leq s \leq s_{i+1}, \\ 0 \text{ otherwise,} \end{cases} \tag{1}$$

where $s_0 = s_1$ and $s_{m+1} = s_m$. We have [3,5]

$$\sum_{i=1}^{i=m} w_i(s) = 1, \forall s \in S \text{ and } w_i(s_j) = 0 \text{ if } i \neq j,\ 1, \text{ otherwise.}$$

Let $L_h f$ be the piecewise linear interpolant to f at points $s_1, ..., s_m$ [3,5]

$$L_h f(s) = \sum_{i=1}^{m} f(s_i)\, w_i(s). \tag{2}$$

The following well known error estimate whose proof is based on divided differences is necessary in the following.

Theorem 1. [3] *We have* $|L_h f(s) - f(s)| \leq \frac{1}{8} K h^2$, $\forall s \in S$.

Proof. See [3]. □

Remark 1. From this theorem we see that $L_h f(s)$ is an ε-piecewise linear estimation of f when $\frac{1}{8} K h^2 \leq \varepsilon$ that holds when $h = \frac{b-a}{m-1} \leq \sqrt{\frac{8\varepsilon}{K}}$, or $m \geq (b-a)\sqrt{\frac{K}{8\varepsilon}} + 1$. In other words, with an uniform discretization like above we need $m = \lfloor (b-a)\sqrt{\frac{K}{8\varepsilon}} \rfloor + 1$ function evaluations at m discretize points ($\lfloor x \rfloor$ denotes the integer part of x) for determining an ε-piecewise linear estimation of f.

At the current step k of the branch and bound algorithms we have to compute a lower bound of f on the interval $T^k := [a_k, b_k]$. Using Theorem 1 we get an affine minorization l_k of f on $[a_k, b_k]$ that is defined as, for any $s \in [a_k, b_k]$ ($h = b_k - a_k$)

$$\begin{aligned} l_k(s): &= L_h f(s) - \frac{1}{8} K h^2 = f(a_k)\frac{b_k - s}{b_k - a_k} + f(b_k)\frac{s - a_k}{b_k - a_k} - \frac{1}{8} K (b_k - a_k)^2 \\ &= \frac{1}{h}(f(b_k) - f(a_k))(s - b_k) + f(b_k) - \frac{1}{8} K h^2. \end{aligned}$$

2.1.2. *A convex quadratic underestimating function*

We can introduce another convex minorization of f on $[a_k, b_k]$ which is better than l_k :

Theorem 2. *We have:*

$$L_h f(s) - \frac{1}{8}Kh^2 \leq q_k(s) := L_h f(s) - \frac{1}{2}K(s-a_k)(b_k-s) \leq f(s), \quad \forall s \in [a_k, b_k]. \tag{3}$$

Proof.

$$E(s) := L_h f(s) - \frac{1}{8}Kh^2 - q_k(s) = -\frac{K}{8}(b_k-a_k)^2 + \frac{K}{2}(s-a_k)(b_k-s)$$

$$= \frac{K}{2}\left[-s^2 + (a_k+b_k)s - a_k b_k - \frac{1}{4}(b_k-a_k)\right].$$

The function E is concave on $[a_k, b_k]$, and its derivative is equal to zero at $s^* = \frac{1}{2}(a_k+b_k)$. Therefore, for any $s \in [a_k, b_k]$ we have

$$E(s) \leq \max\{E(s) : s \in [a_k, b_k]\} = E(s^*) = 0,$$

and the first inequality of (3) is verified.
To justify the second inequality, consider now the function ϕ defined on S by

$$\phi(s) := f(s) - q_k(s) = f(s) - L_h f(s) + \frac{1}{2}K(s-a_k)(b_k-s).$$

Clearly that $\phi''(s) = f''(s) - K \leq 0$ for all $s \in S$. Hence ϕ is concave function, and therefore for all $s \in [a_k, b_k]$ we have

$$\phi(s) \geq \min\{\phi(s) : s \in [a_k, b_k]\} = \phi(a_k) = \phi(b_k) = 0.$$

The second inequality of (3) is also proved. □

Remark 2. For each interval $\mathrm{T}^k = [a_k, b_k]$ we can use the above results with K_k instead of K, where K_k is a positive number such that $|f''(s)| \leq K_k$ for all $s \in [a_k, b_k]$. From numerical point of view, if such a K_k can be easily computed, then it is interesting to replace K by K_k on T^k, since the smaller K_k is, the better lower bound q_k will be.

A lower bound of f on T^k, denoted $LB(T^k)$, is then computed by solving the next convex program

$$LB(T^k) := \min\{q_k(s) : a_k \leq s \leq b_k\}. \tag{4}$$

Since q_k is a quadratic function in the form

$$q_k(s) := \frac{K}{2}s^2 + \left[\frac{f(b_k)-f(a_k)}{b_k-a_k} - \frac{K}{2}(b_k+a_k)\right]s + \frac{K}{2}a_k b_k + \frac{f(a_k)b_k - f(b_k)a_k}{b_k-a_k}, \tag{5}$$

the optimal solution s_k^* of (4) is explicitly defined as follows:

$$s_k^* = \begin{cases} \mu := \frac{1}{2}(a_k+b_k) - \frac{1}{K_h}(f(b_k)-f(a_k)) & \text{if } \mu \in [a_k, b_k], \\ a_k & \text{if } \mu < a_k, \\ b_k & \text{if } \mu > b_k. \end{cases} \tag{6}$$

Remark 3. If $s_k^* \notin]a_k, b_k[$, then

$$\min\{q_k(s) : a_k \le s \le b_k\} = \min\{f(s) : a_k \le s \le b_k\} = \min\{f(a_k), f(b_k)\}.$$

In such a case we have an exact evaluation on the interval T^k (see Fig. 2). Consequently T^k is deleted from the list of intervals which will be considered in the next steps of the branch and bound algorithms.

2.1.3. *Branching procedure: $w-$subdivision*

For dividing the interval T^k one can use its middle point (the normal subdivision). However the efficiency of the $w-$subdivision in [21, 27] suggests us to use it in this work. $w-$subdivision is a rectangular bisection procedure introduced by Falk-Soland [6] for separable nonconvex programming problems that can be summarized as follows. Let $q(x) := \sum_{i=1}^n q_i(x_i)$ be the separable lower bounding function of the separable objective function $f(x) := \sum_{i=1}^n f_i(x_i)$, and let T^k be the rectangle that is to be divided at iteration k. Choose an index i_k satisfying

$$i_k \in \arg\max_i \{f_i(x_i^*) - q_i(x_i^*)\}$$

and subdivide T^k into two subrectangles

$$T_1^k = \left\{x \in T^k : x_{i_k} \le x_i^*\right\}, T_2^k = \left\{x \in T^k : x_{i_k} \ge x_i^*\right\},$$

where x^* is a minimizer of q on T^k.

Using this idea for our algorithm we divide T^k *via* s_k^*, *i.e.* $T_1^k = T_1^k \cup T_2^k$, where $T_1^k = [a_k, s_k^*]$, $T_2^k = [s_k^*, b_k]$,and s_k^* is the point given in (6). This procedure seems to be very efficient: we often obtain the exact evaluation when computing lower bound.

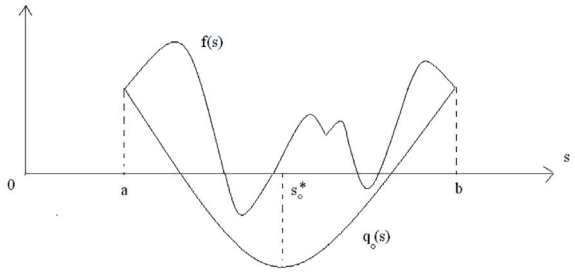

FIGURE 1. The lower bounding function of f, case 1.

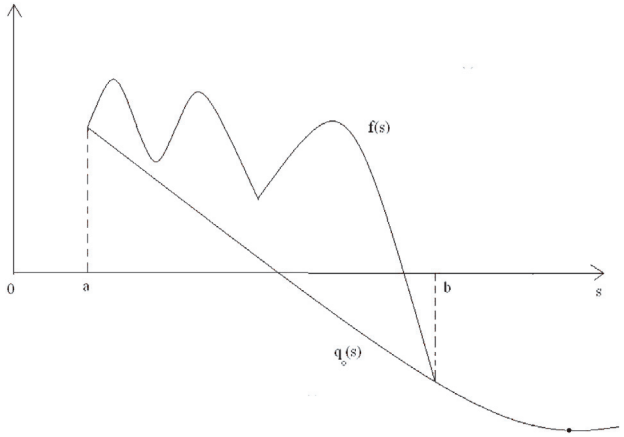

FIGURE 2. The lower bounding function of f, case 2.

2.2. The description of the branch and bound algorithm and its convergence

We can now describe the branch and bound algorithm for solving (P). Denote by, respectively, LB_k, UB_k and s^k the best lower bound, the best upper bound of α and the best solution to (P) at iteration k.

Algorithm BB

1. Initialization:
   a) Let ε be a given sufficiently small number, and let K be a constant verifying $K \geq |f''(s)|$ for all s in $[a,b]$.
   b) Set $k := 0, T^0 = [a,b], M := \{T^0\}$, $h_0 = a - b, s_0^* = \frac{1}{2}(a+b) - \frac{1}{K(b-a)}(f(b) - f(a))$.
   c) Set $UB_0 = \min\{f(a), f(b), f(s_0^*)\}$
   d) Set $LB_0 := LB(T^0) := q_0(s_0^*) = \frac{K}{2}s_0^{*2} + \left[\frac{f(b)-f(a)}{b-a} - \frac{K}{2}(b+a)\right] s_0^* + \frac{K}{2}ab + \frac{f(a)b-f(b)a}{b-a}$.
2. While $UB_k - LB_k > \varepsilon$ do

3. Let $T^k = [a_k, b_k] \in M$ be the interval such that $LB_k = LB(T^k)$. Let K_k be a constant verifying $K_k \geq |f"(s)|$ for all s in $[a_k, b_k]$.
4. Bisect T^k into two intervals by $w-$subdivision procedure: $T_1^k = [a_k, s_k^*] := \left[a_k^1, b_k^1\right]$, $T_2^k = [s_k^*, b_k] := \left[a_k^2, b_k^2\right]$.
5. For $i = 1, 2$ do

   5.1. set $s_{k,i}^* = \frac{1}{2}\left(a_k^i + b_k^i\right) - \frac{f\left(b_k^i\right) - f\left(a_k^i\right)}{K_k\left(b_k^i - a_k^i\right)}$.

   5.2. If $s_{k,i}^* \notin]a_k^i, b_k^i[$, then update $LB(T_i^k) = UB(T_i^k) = \min\left\{f(a_k^i), f(b_k^i)\right\}$.

   else set

$$\mathtt{LB}(\mathtt{T}_i^k) := \frac{K_k}{2}\mathtt{s}_{k,i}^{*2} + \left[\frac{f(b_k^i) - f(a_k^i)}{b_k^i - a_k^i} - \frac{K}{2}(b_k^i + a_k^i)\right]\mathtt{s}_{k,i}^* + \frac{K_k}{2}\mathtt{a}_k^i\mathtt{b}_k^i + \frac{f(a_k^i)b_k^i - f(b_k^i)a_k^i}{b_k^i - a_k^i}. \tag{7}$$

   5.3. To fit into M the intervals $T_i^k : M \leftarrow M \cup \{T_i^k : UB^k - LB(T_i^k) \geq \varepsilon, i = 1, 2\} \backslash \{T^k\}$.

   5.4. Update $UB_k = \min\left\{\mathtt{UB_k}, f(a_k^i), f(b_k^i), f(s_{k,i}^*)\right\}$.
6. Update $LB_k = \min\{LB(T) : T \in M\}$.
7. Delete from M all interval T such that $LB(T) > UB_k - \varepsilon$.
8. Set $k \leftarrow k + 1$.
9. End while
10. STOP: s^k is an ε - optimal solution to (P).

Theorem 3 (convergence of the algorithm)**.**
(i) For a given $\varepsilon > 0$, the algorithm terminates after a finite number of iterations and it provides an $\varepsilon-$optimal solution to problem (P).
(ii) With $\varepsilon = 0$ either the algorithm is finite or it generates a bounded sequence $\{s^k\}$, every accumulation point of which is a global optimal solution of (P), and

$$UB_k \searrow \alpha, \quad LB_k \nearrow \alpha.$$

Proof. From Remark 1 we see that the maximal number k of iterations at which $UB_k - LB_k \leq \varepsilon$ is $m = \lfloor (b-a)\sqrt{\frac{K}{8\varepsilon}} \rfloor + 1$. Thus the part (i) holds.
Assume now that the algorithm is infinite, then it must generate an infinite sequence $\{T^k\}$ of intervals whose lengths h decrease to zero, then the whole sequence $\{T^k\}$ shrinks to a singleton. First, from the description of the algorithm we see that, at each iteration k,

$$LB(T_i^k) \geq LB(T^k).$$

Indeed, if $s_{k,i}^* \notin]a_k^i, b_k^i[$, then $LB(T_i^k) = UB(T_i^k) = \min\left\{f(a_k^i), f(b_k^i)\right\} \geq LB(T^k) := \min\left\{f(s) : s \in \mathtt{T}^k\right\}$, and so T^k is deleted from the set M (see Rem. 3).

If $s^*_{k,i} \in]a^i_k, b^i_k[$ $(i = 1, 2)$, then from (5) it follows that

$$q_k(s) - q_{k,i}(s) = B^i s + C^i \tag{8}$$

with B^i and C^i being constants given by

$$\begin{aligned} B^i &:= \frac{f(b_k) - f(a_k)}{b_k - a_k} - \frac{K}{2}(b_k + a_k) - \frac{f(b^i_k) - f(a^i_k)}{b^i_k - a^i_k} + \frac{K}{2}(b^i_k + a^i_k), \\ C &:= \frac{K}{2}a_k b_k + \frac{f(a_k)b_k - f(b_k)a_k}{b_k - a_k} - \frac{K}{2}a^i_k b^i_k + \frac{f(a^i_k)b^i_k - f(b^i_k)a^i_k}{b^i_k - a^i_k}. \end{aligned}$$

Hence

$$\max\left\{q_k(s) - q_{k,i}(s) : s \in [a^i_k, b^i_k]\right\} = \max\left\{q_k(a^i_k) - q_{k,i}(a^i_k), q_k(b^i_k) - q_{k,i}(b^i_k)\right\} = 0 \tag{9}$$

because

$$\begin{aligned} q_k(a^1_k) = q_{k,1}(a^1_k), q_k(b^2_k) &= q_{k,2}(b^2_k), q_{k,1}(b^1_k) \\ &= f(b^1_k) < q_k(b^1_k), q_{k,2}(a^2_k) = f((a^2_k) < q_k(a^2_k). \end{aligned}$$

It means that $q_k(s) - q_{k,i}(s) \leq 0$, or $q_k(s) \leq q_{k,i}(s)$ for all $s \in \left[a^i_k, b^i_k\right]$. Consequently, $LB(T^k_i) \geq LB(T^k)$ and the sequence $\{LB_k\}$ is therfore non-decreasing and bounded from above by α.

In combining with the facts that the sequence $\{UB_k\}$ is non-increasing and bounded from below by α, and the whole sequence $\{T^k\}$ shrinks to a singleton we obtain

$$UB_k - LB_k \searrow 0, UB_k \searrow \alpha, \quad LB_k \nearrow \alpha.$$

Moreover, since $s^k \in T^0$ and $UB_k = f(s^k)$, any cluster point of the sequence $\{s^k\}$ belongs to T^0 and has the function value α, *i.e.*, solves problem (P). The theorem is proved.

3. Solving nonconvex constrained global optimization problems

Based on Algorithm BB for (P), we develop in this section a branch and bound algorithm for solving Problem (PC) whose feasible domain is denoted D_{pc}. We assume that the nonconvex constraint is essential, *i.e.* there exists $s \in [a, b]$ such that $g(s) > 0$ because if $g(s) \leq 0$ for all $s \in [a, b]$, then problems (PC) and (P) have the same feasible and so they are identical.

We compute a lower bound of the objective function of (PC) by relaxing both feasible set and objective function in order to get a problem of the type (4). More

precisely, on each sub-interval $[a_k, b_k]$ we consider the following problem ($h = b_k - a_k$)

$$(PC_h)\begin{cases} \beta_h := \min q_h(s) \\ L_h g(s) - \frac{1}{2}K_g(s-a_k)(b_k - s) \le 0 \\ a_k \le s \le b_k \end{cases}$$

whose feasible set is denoted by D_{pc}^h. Clearly, with $h_0 = b - a$ we have $D_{pc}^{h_0} \supset D_{pc}$, and for any k and h, $D_{pc}^h \supset D_{pc}^k := \{s \in [a_k, b_k] : g(s) \le 0\}$. Moreover

$$\lim_{h\to 0} q_h^g(s) \ = g(s), \lim_{h\to 0} q_h(s) \ = f(s)$$

that ensures the convergence of our algorithm.

We first remark that D_{pc}^k is a union of intervals while D_{pc}^h is exactly one interval (if they are not empty), because

$$q_h^g(s) := L_h g(s) - \frac{1}{2}K_g(s-a_k)(b - s_k)$$

is a convex function. Hence Problem (PC_h) takes the form (4), that is to say the minimization of a convex quadratic function on an interval.

The extremities a'_k and b'_k of the interval $D_{pc}^h := \{s \in [a_k,\ b_k] : q_h^g(s) \le 0\}$ can easily be computed as follows:

$$a'_k := \max\{a_k, s_1^g\}, b'_k := \min\{b_k, s_2^g\}$$
$$\text{with } s_1^g < s_2^g \text{ solutions of the equation } q_h^g(s) = 0. \quad (10)$$

The emptiness of $D_{pc}^h \cap D_{pc}$ can be detected by observing that: if the minimizer of $q_h^g(s)$ on $[a_k, b_k]$, denoted s_{kg}^*, is not in $[a_k, b_k]$, then

$$\min\{g(s) : s \in [a_k, b_k]\} = \min\{q_h^g(s) : s \in [a_k, b_k]\} = \min\{g(a_k), g(b_k)\}.$$

Therefore,

$$\text{if } s_{kg}^* \notin [a_k, b_k] \text{ and } \min\{g(a_k), g(b_k)\} > 0, \text{ then } D_{pc}^h \cap D_{pc} = \emptyset. \quad (11)$$

The algorithm for solving (PC) differ Algorithm **BB** mainly from the branching procedure:

- Subdivide the interval $T^k := [a_k\ , b_k]$ into $D_{pc}^{h_k^1}$ and $D_{pc}^{h_k^2}$ *via* s_{kf}^* as follows

$$D_{pc}^{h_k^1} := \left\{s \in \left[a_k\ , s_{kf}^*\right] : q_{h_k^1}^g(s) \le 0\right\}, D_{pc}^{h_k^2} := \left\{s \in \left[s_{kf}^*, b_k\right] : q_{h_k^2}^g(s) \le 0\right\}. \quad (12)$$

- Discard the subinterval $D_{pc}^{h_k^i}$ $(i = 1, 2)$ if its intersection with the feasible region D_{pc} is empty (using (11)). Otherwise, determine the extremities of $D_{pc}^{h_k^i}$ *via* (10).

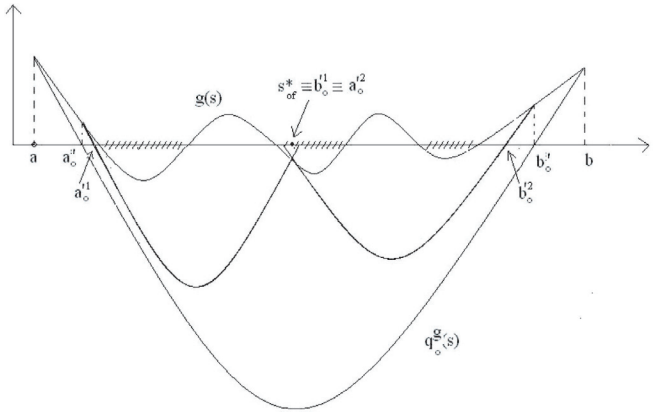

FIGURE 3. The feasible set of Problem (PC) after the first iteration is $[a_0'^1, b_0'^2]$.

The lower bounding procedure is not changed, namely solving the convex quadratic problem of the form

$$\min \{q_h(s) : a'_k \leq s \leq b'_k\}.$$

The upper bound can be improved when a feasible point of (PC) is detected by checking the condition $g(s) \leq 0$, with $s \in \left\{s^*_{kf}, s^*_{kg}, s^g_1, s^g_2\right\}$. Since $\lim_{h\to 0} q^g_h(s) = g(s)$, this condition must be satisfied at some iteration k of a branch and bound algorithm provided that the set D_{pc} is nonempty.

Our algorithm for solving Problem (PC), denoted Algorithm **BB2**, is the former Algorithm BB with the following modifications:

> Add in step 1a): let K_g `be a constant verifying` K$\geq |g"(s)|$ `for all s in` $[a, b]$.
> Modify the part concerning T^0 in step 1b): set $T^0 := \{s \in [a, b] : g(s) \leq 0\}$. Check the emptiness of T^0 by using (11). If $T^0 = \emptyset$, then STOP, the problem is infeasible, else determine the extremities a_0, b_0 of T^0 *via* (10).
> Step 1c): If $g(s^*_0) \leq 0$,then `set` $UB_0 := \min\{f(a_0), f(b_0), f(s^*_0)\}$, else `set` $UB_0 = \min\{f(a_0), f(b_0)\}$.
> Step 4: subdivide the interval $T^k := [a_k, b_k]$ into $D^{h^1_k}_{pc}$ and $D^{h^2_k}_{pc}$ *via* s^*_{kf} following (12). Check the emptiness of $D^{h^1_k}_{pc}$ (resp. $D^{h^2_k}_{pc}$) by using (11). If $D^{h^1_k}_{pc} \neq \emptyset$, (resp. $D^{h^2_k}_{pc} \neq \emptyset$), then determine its extremities a^1_k, b^1_k (resp. a^2_k, b^2_k) according to (10).
> Step 5.4. `Update` $UB_k = \min\left\{\text{UB}_\text{k}, f(a^i_k), f(b^i_k), f(s^*_{k,i})\right\}$. `For` $s \in \left\{s^*_{kf}, s^*_{kg}, s^g_1, s^g_2\right\}$, `if g(`$s$`)` ≤ 0, then `update` $UB_k = \min\{\text{UB}_\text{k}, f(s)\}$.

4. Illustrative examples and computational results

To illustrate the behavior of the algorithm, we perform the following numerical examples with $\varepsilon = 0.002$.

Example 1.

$$\min\left\{F_1(s) := \sin(s) + \sin\left(\frac{10s}{3}\right) + \ln(s) - 0.84s : 2.7 \leq s \leq 7.5\right\}.$$

Initialization (iteration $k = 0$): Set $K = 12.5$. Compute

$$s_0^* := \frac{1}{2}(a+b) - \frac{1}{K(b-a)}(f(b) - f(a)) = 5.150737.$$

Since $s_0^* \in [2.7, 7.5]$, we process the branching and bounding procedures with $T^0 = [2.7, 7.5]$, $LB_0 := q_0(s_0^*) = -37.973438$ and $UB_0 := \min\{f_2(a), f_2(b), f_2(s_0^*)\} = f_2(s_0^*) = -4.586929$. The algorithm terminates after 7 iterations.

Example 2.

$$\min\left\{F_2(s) := \frac{3}{4}\sin s + \frac{1}{4}\cos s : 0 \leq s \leq 1\right\}.$$

Initialization. Set $K = 1$. Set

$$s_0^* := \frac{1}{2}(a+b) - \frac{1}{K(b-a)}(f(b) - f(a)) = \frac{1}{2} - (f(1) - f(0)) = -0.016179.$$

Since $s_0^* \notin [0,1]$, $LB_0 = UB_0 = \min\{f(0), f(1)\} = f(0) = 0.25$. Hence the exact optimal solution is $s^0 = 0$ and the optimal value is 0.25.

Example 3.

$$\min\left\{F_3(s) := 3s^3 + 2s^2 - 1 : \sin(s) + s - 1 \leq 0, -1 \leq s \leq 1\right\}.$$

Initialization (iteration $k = 0$): Set $K = 8$, $K_g = 1$. Since $s_{0g}^* := -1$, the feasible set is not empty. The two solutions of the equation $LB_g(s) := 1.8415s - 1.5 + 0.5s^2 = 0$ are -4.3696 and 0.6865, the feasible set is then updated $T^0 \leftarrow [-1, 0.6865]$. On this interval we compute s_{0f}^*, LB_0, UB_0 and get $s_{0f}^* = -0.0906$, $LB_0 = -2.3078$, $UB_0 = 0.098$. Set $s^0 = -0.0906$.

By Table 1 we show how Algorithm BB2 works on this example. The algorithm terminates after 5 iterations at which $UB_5 - LB_5 = 0.0019 < \varepsilon$. From this table we see that in the step 4 of Algorithm BB2 the intervals T_i^k $(i = 1, 2)$ are often reduced: for example, $T_2^1 := [-0.0906, 0.6865]$ is reduced to $[-0.0906, 0.544]$ according to (10). This accelerates the convergence of the algorithm. We note also that the optimal solution is detected at iteration $k = 2$ when computing $s_{kg,i}^*$, $i = 1, 2$ $(s^2 = s_{2g,2}^*)$.

Example 4.

$$\min\left\{F_4(s) := \sin(s) + \sin\left(\frac{2s}{3}\right) : 3\cos(1+s) + (1+s)^2 - 400 \leq 0, 15 \leq x \leq 20\right\}.$$

TABLE 1. Illustrate Example 3.

k	T^k	s^*_{kf}	s^*_{kg}	T_1^k $LB(T_1^k)$	T_2^k $LB(T_2^k)$	LB_k	UB_k	s^k
1	$T^0 \downarrow$ $[-1, 0.6865]$	–0.0906	–1	$[-1, -0.0906]$ –0.3394	$[-0.0906, 0.6865]$ –0.4025	–0.4025	–0.1756	0.2361 ($s^*_{1f,2}$)
2	$T_2^1 \downarrow$ $[-0.0906, 0.544]$	0.2631	–0.0906	$[-.0906, 0.2631]$ –0.2013	$[0.2631, 0.5138]$ –0.2115	–0.3394	–0.1839	0.3577 ($s^*_{2g,2}$)
3	T_1^1	–0.4213	–1	$[-1, -0.4213]$ 0.3848	$[-0.4213, -0.0906]$ –0.0979	–0.2115	–0.1839	0.3577
4	$T_2^2 \downarrow$ $[0.2631, 0.3577]$	0.3213	0.2631	$[0.2631, 0.3213]$ –0.1852	$[0.3213, 0.3577]$ –0.1858	–0.2013	–0.1839	0.3577
5	$T_1^2 \downarrow$ $[-0.0906, 0.1829]$	0.1564	–0.0906	$[-0.0906, 0.1564]$ –0.1284	$[0.1564, 0.1829]$ –0.1570	–0.1858	–0.1839	0.3577

Initialization (iteration $k = 0$): Set $K = 1.5$, $K_g = 5$. Since $s^*_{0g} := 15$, the feasible set T^0 is not empty. By computing two solutions of the equation $LB_g(x) = 0$ we update the feasible set $T^0 \leftarrow [15, 19.175]$. On this interval we compute s^*_{0f}, LB_0, UB_0 and get $s^*_{0f} = 17.019$, $LB_0 = -2.9511$, $UB_0 = -1.9057$. Set $s^0 = 17.019$.

The algorithm terminates after 4 iterations at which $UB_4 - LB_4 = 0$, *i.e.* we get an exact optimal solution $s^* = 17.039$. We see that a very good approximation solution (almost exactly global minimizer) is obtained at the initialization step.

We report below the numerical results of our algorithms on four data sets of test problems:

- The first part contains 7 polynomial test functions from [15] (see also [9], chapter 4).
- the second part is composed of 6 twice differentiable functions chosen among Hölder test function studied in [12].
- The third part contains four multi-extremal functions considered in [36,37].
- The last part consists of 3 problems with polynomial objective function and one constraint studied in [7].

The algorithms has been implemented in C and run on a PC Inter Pentium M 1.40 GHz, 512 Mb of RAM. The following notations are used in the tables below:

- Numeva: the number of function evaluations;
- iter: the number of iterations;
- CPU: computing times in seconds.

4.1. POLYNOMIAL FUNCTIONS

In Table 2 we describe the expression of the objective function, the interval $[a, b]$ and the number of local minima (n_1) as well as that of global minima (n_2) given in [9, 15] for the first set of test problems.

The vector a in Problem 2 is defined by

$$a = (2.5, 1.666666666, 1.25, 1.0, 0.8333333, 0.714285714, 0.625, 0.555555555, 1.0, -43.6363636, 0.41666666, 0.384615384, 0.357142857, 0.3333333, 0.31250, 0.294117647, 0.277777777, 0.263157894, 0.25, 0.238095238, 0.227272727, 0.217391304, 0.208333333,$$

TABLE 2. Polynomial test problems used in [9, 15] .

Problem	Objective function f(s)	[a,b]	n_1	n_2
1	$\frac{1}{10} - s - \frac{79}{20}s^2 + \frac{71}{10}s^3 + \frac{3}{80}s^4 - \frac{52}{25}s^5 + \frac{1}{6}s^6$	[-2,11]	3	1
2	$\sum_{i=1}^{50} a_i s^i$	[1,2]	1	1
3	$0.0000089248s - 0.0218343s^2$ $+0.998266s^3 - 1.6995s^4 + 0.2s^5$	[0,10]	2	1
4	$4s^2 - 4s^3 + s^4$	[-5,5]	2	2
5	$1.75s^2 - 1.05s^4 + \frac{1}{6}s^6$	[-5,5]	3	1
6	$s^6 - 15s^4 + 27s^2 + 250$	[-5,5]	3	2
7	$s^4 - 3s^3 - 1.5s^2 + 10s$	[-5,5]	2	1

TABLE 3. Comparative computational results with GOP [8] and Interval Arithmetic methods [15], $\varepsilon = 10^{-12}$.

Pb	**BB**		NEF		NF		CF		QRF		CRF		GOP
	iter	CPU	iter	CPU	iter	CPU	iter	CPU	iter	CPU	iter	CPU	iter
1	59	$<10^{-12}$	44	1.00	28	0.52	27	1.58	28	0.76	35	1.00	26
2	31	$<10^{-12}$	59	11.96	49	7.32	43	95.06	52	11.72	52	10.60	45
3	15	$<10^{-12}$	37	0.68	27	0.42	27	1.18	30	0.68	39	1.10	38
4	24	$<10^{-12}$	77	1.16	48	0.60	47	1.42	73	1.58	46	1.04	62
5	33	$<10^{-12}$	35	0.72	21	0.34	31	1.76	49	1.28	17	0.56	43
6	21	$<10^{-12}$	87	1.96	54	0.96	53	2.92	75	2.04	73	2.06	
7	12	$<10^{-12}$	27	0.38	21	0.26	26	0.78	26	0.54	24	0.52	
Ave	26.43	$<10^{-12}$	52.29	2.55	35.43	1.49	36.29	14.96	47.57	2.66	40.86	2.41	42.80

$0.2, 0.192307692, 0.185085085, 0.178571428, 0.344827586, 0.6666666, -15.48387097,$ $0.15625, 0.1515151, 0.14705882, 0.14285712, 0.138888888, 0.135135135, 0.131578947,$ $0.128205128, 0.125, 0.121951219, 0.119047619, 0.116279069, 0.113636363, 0.1111111,$ $0.108695652, 0.106382978, 0.208333333, 0.408163265, 0.8$).

The number of iterations and computing times of Algorithm BB with the tolerance $\varepsilon = 10^{-12}$ are presented in Table 3. We also listed there the computational results of GOP [8] and Interval Arithmetic methods [15] (with the same tolerance) for the reference. The Interval Arithmetic methods [15] have been tested on a SUN 3/50-12 workstation with 1.5 mips central processor. It is worth noting that, according to the description of the algorithms, the number of performed operations in one iteration of Algorithm BB is smaller than that of Interval Arithmetic methods [15] and GOP [8]. Hence, from Table 3 we see that Algorithm BB is the best.

4.2. HÖLDER FUNCTIONS AND MULTI-EXTREMAL FUNCTIONS

The data (the objective function and the bounds on the variables) of the second and the third sets of test problems is described in the Table 4. In Table 5 we present the comparative computational results in terms of the number of function evaluations of Algorithm BB and 9 algorithms reported in [36, 37] for four test

TABLE 4. Test problems with Hölder functions (1.1 - 1.8) and multi-extremal functions (f_1 - f_4).

Problem	Objective function	[a,b]
1.1	$s^6 - 15s^4 + 27s^2 + 250$	$[-4, 4]$
1.2	$(s^2 - 5s + 6)/(s^2 + 1)$	$[-5, 5]$
1.5	$(3s - 1.4)\sin(18s)$	$[0, 1]$
1.6	$2(s-3)^2 + \exp(s^2/2)$	$[-3, 3]$
1.7	$-\sum_{k=1}^{5} k\sin[(k+1)s + k)]$	$[-10, 10]$
1.8	$\sum_{k=1}^{5} k\cos[(k+1)s + k]$	$[-10, 10]$
f_1	$\sin(s) + \sin\left(\frac{10s}{3}\right) + \ln(s) - 0.84s$	$[2.7, 7.5]$
f_2	$\sin(s) + \sin(\frac{2s}{3})$	$[3.1, 20.4]$
f_3	$(-\sum_{i=1}^{i=5} \sin((i+1)x + i)$	$[-10, 10]$
f_4	$(x + \sin(x))e^{-x^2}$	$[-10, 10]$

TABLE 5. Comparative computational results with the algorithms in [36], [37] for multi-extremal functions with $\varepsilon = 10^{-6}$.

Function	**BB**	S-LLB	N-LLB	O-LLB	Zil1	Zil2	Strong	Pijav	Brent	Batish
f_1	9	12	12	19	33	29	45	462	25	120
f_2	18	18	18	24	37	38	442	448	45	158
f_3	29	73	73	94	125	165	150	3817	161	816
f_4	34	13	19	35	35	34	98	376	229	83
Ave	22.50	29.00	30.50	43.00	57.50	66.5	158.75	1275.75	112.50	294.25

functions considered in [36, 37] (see [36] for more details). Likewise, in Table 6 (left) we report the number of function evaluations of Algorithm BB and the branch and bound method developed in [12] for six problems with Hölder functions. The numerical results of our Algorithm BB (the number of function evaluations, the number of iterations and computing times) with the precision $\varepsilon = 10^{-12}$ are listed in the right-hand side of the same table.

From the experimental results we see that our algorithm BB is efficient for all data sets. It is very fast: in all test problems the total processing time of our algorithm is smaller than 10^{-12} second. On the other hand, we observe that the number of intervals to be considered (the cardinality of the set M) at each iteration is small. Algorithm BB algorithm is much better than Algorithm HOL^1: Table 6 (left) indicates that the ratio of the number of function evaluations belongs to the interval $[2.4, 22.8]$. Note that the results of HOL^1 presented in Table 6 (left) correspond to the case $\alpha = 1$ which is the best case of this algorithm for different values of α (see [12]). Moreover, Algorithm BB is better than the best possible algorithm studied in [12].

For the four multi-extremal functions our algorithm is more efficient than the best algorithm presented in [36, 37], except for the last function.

TABLE 6. Comparative computational results with Algorithm HOL[1] in [12] for Hölder optimization (left) and Computational results of Algorithm BB with $\varepsilon = 10^{-12}$ for Hölder functions and multi-extremal functions (right).

Problem	ε	**BB**	HOL[1]
1.1	2×10^{-5}	58	749
1.2	7×10^{-8}	67	624
1.5	10^{-8}	6	137
1.6	3×10^{-7}	82	425
1.7	7×10^{-8}	81	193
1.8	2×10^{-7}	75	181

Problem	Numeva	iter	CPU
1.1	20	17	$<10^{-12}$
1.2	123	120	$<10^{-12}$
1.5	6	3	$<10^{-12}$
1.6	16	13	$<10^{-12}$
1.7	95	92	$<10^{-12}$
1.8	93	90	$<10^{-12}$
f_1	14	11	$<10^{-12}$
f_2	26	23	$<10^{-12}$
f_3	34	31	$<10^{-12}$
f_4	103	100	$<10^{-12}$

TABLE 7. Comparative computational results with Pijaavskii's method.

Problem	$f(s)$	$g(s)$	$[a, b]$	**BB2**	Pijaavskii's method
1	$-\frac{13}{6}x + \sin(\frac{13}{4}(2x+5)) - \frac{53}{12}$	$\exp(-\sin(3x))$ $-\frac{1}{10}(x-\frac{1}{2})^2 - 1$	$[-2.5, 1.5]$	7	83
2	$\frac{11x^2-10x+21}{2(x^2+1)}$	$\frac{1}{20} - \exp(-\frac{2}{5}(x+5)) \times$ $\sin(\frac{4}{5}\pi(x+5))$	$[-5, 5]$	23	953
3	$-\sum_{i=1}^{5}\cos(ix)$	$\frac{3}{2}(\cos(\frac{7}{20}(x+10))$ -$\sin(\frac{7}{4}(x+10)) + \frac{1}{2})$	$[-10, 10]$	14	119

4.3. LIPSCHITZ UNIVARIATE GLOBAL OPTIMIZATION WITH MULTI-EXTREMAL CONSTRAINTS

The three test problems with one constraint studied in [7] are described in Table 7. In the last two columns of this table we report the number of iterations of Algorithm BB2 and the number of iterations (given in [7]) of the method proposed by Pijavskii with a penalty technique (see [18,19,26]) for these problems. We see that Algorithm BB2 is much better than Pijaavskii's method: the ratio of the number of iteration is 11.85, 41.43, and 8.5 for Problems 1, 2 and 3 respectively. Note also that our algorithm found an exact solution ($\varepsilon = 0$) while the accuracy of Pijaavskii's method is $\varepsilon = 10^{-4}(b-a)$.

Conclusion. We have developed, in the first part of the paper, an efficient branch and bound method for globally minimizing a twice differentiable univariate function on a bounded interval. Our convex quadratic underestimator for twice continuously differentiable functions is good: it is close to the function when the interval is sufficiently small. Moreover it permits to use the $w-$subdivision, an interesting technique: the algorithm eliminates a considerable number of rectangles during

the bounding and branching procedures. Again, with these bounding and branching procedure our algorithm often find a very good approximation of the global minimizer in a few iterations. Computational results show the superiority of our algorithm compared with efficient existing algorithms.

Based on this scheme, we developed an algorithm which allows dropping infeasible subsets for nonconvex constrained Problem (PC). Both algorithms are very fast as all computations are explicit.

Therefore, we would like to extend these methods to multivariable global optimization problems. Work in this direction is currently in progress.

References

[1] W. Baritompa and A. Cutler, Accelerations for global optimization covering methods using second derivates. *J. Global Optim.* **4** (1994) 329–341.

[2] E. Baumann, Optimal centered forms. *BIT* **28** (1988) 80–87.

[3] C de Boor, *A practical Guide to Splines Applied Mathematical Sciences*. Springer-Verlag (1978).

[4] J. Calvin, and A. Ilinskas, On the convergence of the P-algorithm for one-dimensional global optimization of smooth functions. *J. Optim. Theory Appl.* **102** (1999) 479–495.

[5] P.G. Ciarlet, The Finite Element Method for Elliptic Problems. *Studies Math. Appl.* (1979).

[6] J.E. Falk and R.M. Soland, An algorithm for separable nonconvex programming problems. *Manage. Sci.* **15** (1969) 550–569.

[7] D. Famularo, Y.D. Sergeyev and P. Pugliese, *Test Problems for Lipschitz Univariate Global Optimization with Multiextremal Constraints*. Publication online.

[8] C. Floudas and V. Visweswaran, A Global Optimization Algorithm (GOP) for Certain Classes of Nonconvex NLPs-I. Theory. *Comput. Chemical Engin.* **14** (1990) 1394–1417.

[9] C.A. Floudas, P.M. Pardalos, C.S. Adjiman, W.R. Esposito, Z.H. Gümüs, S.T. Harding, J.L. Klepeis, C.A. Meyer and C.A. Schweiger, *Handbook of Test Problems in Local and Global Optimization*. Kluwer Academic Publishers (1999).

[10] V.P. Gergel, A Global Optimization Algorithm for Multivariate functions with Lipschitzian First Derivatives, *J. Global Optim.* **10** (1997) 257–281.

[11] K. Glashoff and S.A. Gustafson, *Linear Optimization and Approximation*. Amplitude Math. Sciences, Springer-Verlag (1983).

[12] E. Gourdin, B. Jaumard and R. Ellaia, Global Optimization of Hölder Functions. *J. Global Optim.* **8** (1996) 323–348.

[13] E. Hansen, Global optimization using interval analysis the multi-dimensional case. *Numer. Math.* **34** (1980) 247270.

[14] E. Hansen, Global Optimization using Interval Analysis. *Pure Appl. Math.* **165**, Marcel Dekker, New York (1992).

[15] P. Hansen, B. Jaumard and J. Xiong, Decomposition and interval arithmetic applied to minimization of polynomial and rational functions. *J. Global Optim.* **3** (1993) 421–437.

[16] P. Hansen, B. Jaumard and J. Xiong, Cord-slope form of Taylor's expansion in univariate global optimization. *J. Optim. Theory Appl.* **80** (1994) 441–464.

[17] P. Hansen, B. Jaumard and S.-H. Lu, Global Optimization of Univariate Functions by Sequential Polynomial Approximation. *Int. J. Comput. Math.* **28** (1989) 183–193.

[18] P. Hansen, B. Jaumard and S.-H. Lu, Global Optimization of Univariate Lipschitz Functions: 2. New Algorithms and Computational Comparison. *Math. Program.* **55** (1992) 273–292.

[19] R. Horst and P.M. Pardalos, *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht (1995).

[20] R. Horst and H. Tuy, *Global Optimization – Deterministic Approaches*. Springer-Verlag, Berlin (1993).

[21] H.A. Le Thi and T. Pham Dinh, A Branch-and-Bound method via D.C. Optimization Algorithm and Ellipsoidal technique for Box Constrained Nonconvex Quadratic Programming Problems. *J. Global Optim.* **13** (1998) 171–206.
[22] M. Locatelli and F. Schoen, An adaptive stochastic global optimization algorithm for one dimensional functions. *Ann. Oper. Res.* **58** (1995) 263–278.
[23] F. Messine and J. Lagouanelle, Enclosure methods for multivariate differentiable functions and application to global optimization. *J. Universal Comput. Sci.* **4** (1998) 589–603.
[24] R. Moore, *Interval Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, (1966).
[25] P.M. Pardalos and H.E. Romeijn, *Handbook of Global Optimization*, Volume 2. Nonconvex Optimization and Its Applications, Springer, Boston/Dordrecht/London (2002).
[26] S.A. Pijavskii, An Algorithm for Finding the Absolute Extremum of a Function. *USSR. Comput. Math. Math. Physics* **12** (1972) 57–67.
[27] T. Quynh Phong, L.T. Hoai An and P. Dinh Tao, On the global solution of linearly constrained indefinite quadratic minimization problems by decomposition branch and bound method. *RAIRO: Oper. res.* **30** (1996) 31–49.
[28] H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization.* Wiley, New York (1982).
[29] D. Ratz, A nonsmooth global optimization technique using slopes the one-dimensional case. *J. Global Optim.* **14** (1999) 365–393.
[30] A.H.G. Rinnooy and G.H. Timmer, Global Optimization: A Survey, in *Handbooks of Operations Research*, edited by G.L. Nemhauser *et al.* North-Holland, Elsevier Science Publishers (1989) 631–662.
[31] Ya.D. Sergeyev and V.A. Grishagin, A Parallel Method for Finding the Global Minimum of Univariate Functions. *J. Optim. Theory Appl.* **80** (1994) 513–536.
[32] Ya.D. Sergeyev, An One-Dimensional Deterministic Global Minimization Algorithm. *Russian J. Comput. Math. Math. Phys.* **35** 705–717.
[33] A. Törn and A. Zilinskas, *Global Optimization,* edited by G. Goos and J. Hartmanis. Springer, Berlin, *Lect. Notes Comput. Sci.* **350** (1989).
[34] V. Visweswaran and C.A. Floudas, Global Optimization of Problems with Polynomial Functions in One Variable, in *Recent Advances in Global Optimization*, edited by A. Floudas and P.M. Pardalos. Princeton, Princeton University Press, pp. 165–199.
[35] Ya.D. Sergeyev, Global one-dimensional optimization using smooth auxiliary functions. *Math. Program.* **81** (1998) 127–146.
[36] X. Wang and T.-S. Chang, An Improved Univariate Global Optimization Algorithm with Improved Linear Bounding Functions. *J. Global Optim.* **8** (1996) 393–411.
[37] A. Zilinskas, On Global One-Dimensional Optimization. *Engineering Cybernetics, Izvestija AN USSR* **4** (1976) 71–74.