Yokohama Publishers
ISSN 1880-5221 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 2, No. 2, pp. 241–259, May 2006

SOLVING DISCRETE MINIMAX PROBLEMS USING INTERVAL ARITHMETIC

D.G. Sotiropoulos

Abstract: We present an interval algorithm for solving discrete minimax problems where the constituent minimax functions are continuously differentiable functions of one real variable. Our approach is based on smoothing the max-type function by exploiting the Jaynes's maximum entropy [*Phys. Rev.*, 106:620–630, 1957]. The algorithm works within the branch-and-bound framework and uses first order information of the entropic objective function by means of an interval evaluation. First order information aids threefold. Firstly, to check monotonicity. Secondly, to apply mean value form for bounding the range of the function, and finally, to prune the search interval using the current upper bound of the global minimum. Our numerical results show that the proposed algorithm guarantees computationally rigorous bounds for the global minimum and all global minimizers.

Key words: *minimax problem, maximum entropy, interval arithmetic, linear pruning steps, branch-and-prune*

Mathematics Subject Classification: *90C47, 65G20*

1 Introduction

There are many applications in engineering where a not necessarily differentiable objective function has to be optimized. The discrete minimax problem is such an example. The purpose of this paper is to describe a reliable method for solving the minimax optimization problem

$$\min_{x\in X} \max_{1\le i\le m} \{f_i(x)\}, \tag{1}$$

where $f_i : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$ $(i = 1, \ldots, m)$ are continuously differentiable functions, $\mathcal{D}$ is the closure of a nonempty bounded open subset of $\mathbb{R}$, and $X \subseteq \mathcal{D}$ is a search interval representing bound constraints for x. Our aim is to find the global minimum f^* and the set $X^* = \{x^* \in X : f(x^*) = f^*\}$ of all global minimizers of the objective function $f(x) = \max\{f_1(x), \ldots, f_m(x)\}$.

The objective function $f(x)$ has discontinuous first derivatives at points where two or more functions $f_i(x)$ are equal to $f(x)$ even if each $f_i(x)$ $(i = 1, \ldots, m)$ has continuous first derivatives. Solving an optimization problem such as (1) requires, in general, the comparison of a continuum of values and the choice of the optimum value. Since interval arithmetic is a means to handle continua, it provides competitive methods for solving optimization problems. A thorough introduction to the whole area of interval arithmetic can be found in [1, 4, 9].

Interval methods for global optimization combine interval arithmetic with the so-called branch-and-bound principle. These methods subdivide the search region in subregions (branches) and use bounds for the objective function to exclude from consideration subregions where a global minimizer cannot lie. Interval arithmetic provides the possibility to compute such rigorous bounds automatically. Moreover, when the objective function is continuously differentiable, interval enclosures of the derivative along with a mean value form can be used to improve the enclosure of the function range.

In this paper, we exploit Jaynes's maximum entropy [8] for transforming problem (1) into a smooth optimization problem that can be solved efficiently. The proposed interval algorithm utilizes first-order information by means of an interval derivative evaluation. When first-order information is available, monotonicity test is the only accelerating device that one can apply in order to discard subregions where the function is strictly monotone. Our algorithm along with a monotonicity test uses a new accelerating device that composes *inner* and *outer pruning steps* to eliminate parts of the search interval where the global minimum does not exist. The numerical results indicate that the use of the pruning steps leads to more efficient interval algorithms for global optimization.

The paper is organized as follows. The smooth approximation is described in Section 2, while in Section 3 we present the way that we can construct inclusion functions without underflow or overflow problems. We next describe, in Section 4, an accelerating device that utilizes derivative bounds and aims to eliminate parts of the search interval. Finally, in Section 5, we describe the proposed algorithm, while in Section 6 numerical results for a test set are also reported.

2 The Maximum Entropy Function

Although the term "entropy" first appeared in the literature of Physics by 1865, it gained wide publicity by the work of Shannon's Information Theory in 1948. Later, Jaynes* [8] proposed the principle of Maximum Entropy (MaxEnt) which provides a method for solving problems when available information is in the form of moment constraints. The entropy functional introduced by Jaynes along with interval analysis tools [14] is nowadays a possible basis of a theoretical framework appropriate to deal with non-smooth objective functions [7, 13, 16].

Definition 1. *Let p be a positive real number and let $f_i : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$ $(i = 1, \ldots, m)$ be given functions. The maximum entropy function $f_p : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$ of $f_1, \ldots, f_m$ is defined by*

$$f_p(x) = \frac{1}{p} \cdot \ln \left(\sum_{i=1}^{m} \exp\left(p\, f_i(x)\right) \right) \tag{2}$$

The following lemma states that it is possible to approximate the objective function $f(x)$ with arbitrary accuracy using (2) as a smoothing function.

Lemma 1. *For any $\varepsilon > 0$, $\exists\ p_\varepsilon > 0$ such that $|f_p(x) - f(x)| < \varepsilon$, $\forall\ p > p_\varepsilon$.*

*Unpublished works of Prof. Edwin Thompson Jaynes (1922–1998) is available at: `http://bayes.wustl.edu/etj/etj.html`

Proof.

$$\begin{aligned} f_p(x) - f(x) &= \frac{1}{p} \ln\left(\sum_{i=1}^{m} \exp\left(p\, f_i(x)\right)\right) - \frac{1}{p} \ln\left(\exp\left(p\, f(x)\right)\right) \\ &= \frac{1}{p} \ln\left(\sum_{i=1}^{m} \exp\left(p\left(f_i(x) - f(x)\right)\right)\right). \end{aligned} \tag{3}$$

By definition of $f(x)$, we have that $f_i(x) - f(x) \leq 0$, $i = 1, 2, \ldots, m$ and there is at least one $j \in \{1, 2, \ldots, m\}$ such that $f_j(x) - f(x) = 0$. Therefore, the summation of the exponential terms in (3) ranges between 1 and m. Hence, we obtain that

$$0 \leq f_p(x) - f(x) \leq \frac{\ln(m)}{p} \tag{4}$$

and our assertion follows with $p_\varepsilon = \ln(m)/\varepsilon$. □

Lemma 2. *For any $p > 0$, we have that*

$$f_p(x) - \frac{\ln(m)}{p} \leq f(x) \leq f_p(x). \tag{5}$$

Corollary 1. *The maximum entropy function $f_p(x)$ decreases monotonically as p increases and, for any $x \in \mathbb{R}$, $f_p(x) \to f(x)$, as $p \to \infty$.*

The derivative of the function $f_p(x)$ is given by

$$f_p'(x) = \sum_{i=1}^{m} \alpha_i\, f_i'(x) \tag{6}$$

where

$$\alpha_i = \exp\left(p\, f_i(x)\right) \left(\sum_{\ell=1}^{m} \exp\left(p\, f_\ell(x)\right)\right)^{-1}. \tag{7}$$

It is evident that the entropy multiplier vector $\alpha = (\alpha_1, \ldots, \alpha_m)$ belongs in the unit simplex

$$\Sigma_m \triangleq \left\{ \alpha \in \mathbb{R}^m \mid \alpha_i \geq 0,\, i \in \{1, \ldots, m\},\, \sum_{i=1}^{m} \alpha_i = 1 \right\}. \tag{8}$$

However, the terms $\exp(p\, f_i(x))$ become quite large when p approaches ∞. Thus, to prevent overflow, special care must be taken in computing α_i. For this reason we redefine α_i as

$$\alpha_i = \exp\left(p\left(f_i(x) - f(x)\right)\right) \left(\sum_{\ell=1}^{m} \exp\left(p\left(f_\ell(x) - f(x)\right)\right)\right)^{-1}. \tag{9}$$

We denote by $\mathcal{I}(x) = \left\{ i \in \{1, \ldots, m\} : f_i(x) = \max_{j \in \{1, \ldots, m\}} f_j(x) \right\}$ the active index set. Then, for any fixed $x \in \mathbb{R}$, it holds that

$$\exp\left(p\left(f_i(x) - f(x)\right)\right) \equiv 1, \quad \text{for all} \quad i \in \mathcal{I}(x), \tag{10}$$

and

$$\lim_{p \to \infty} \exp\left(p\left(f_i(x) - f(x)\right)\right) \equiv 0, \quad \text{for all} \quad i \notin \mathcal{I}(x). \tag{11}$$

The following theorem states a first-order optimality condition for the minimax problem (1) in terms of the entropy multipliers α_i.

Theorem 1. *Suppose that the functions $f_i : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$, $i \in \{1, \ldots, m\}$, are continuously differentiable. If x^* is a local minimizer of $f(x)$ then the associated multiplier $\alpha^* \in \Sigma_m$ defined in (9), satisfies the conditions*

$$\sum_{i=1}^{m} \alpha_i^* f_i'(x^*) = 0 \tag{12}$$

and

$$\sum_{i=1}^{m} \alpha_i^* \left(f(x^*) - f_i(x^*)\right) = 0. \tag{13}$$

Proof. Condition (12) immediately follows from the fact that x^* is also a local minimizer of $f_p(x)$, in view of Lemma 1. For the proof of condition (13), note that $f(x^*) - f_i(x^*) = 0$ for all $i \in \mathcal{I}(x^*)$, while $a_i^* = 0$ for all $i \notin \mathcal{I}(x^*)$. □

Remark. Obviously, we can associate with problem (1) the Lagrangian function $L : \mathbb{R} \times \Sigma_m \to \mathbb{R}$, defined by

$$L(x, \mu) = \sum_{i=1}^{m} \mu_i f_i(x).$$

We notice that if x^* is a local minimizer then, in view of (13), $L(x^*, \alpha^*) = f(x^*)$. Moreover, $L(x, \alpha^*) = f(x)$ for all $x \in \mathbb{R}$, because $\alpha_i^* = 0$ for all $i \notin \mathcal{I}(x^*)$ and $\sum_{i=1}^{m} \alpha_i^* = 1$. Hence x^* must also be a local minimizer of the restriction of $L(\cdot, \alpha^*)$ to $\mathbb{R}$.

3 MaxEnt Inclusion Functions

The main interval arithmetic tool applied to optimization problems is the concept of an inclusion function; see, for example [1, 9, 10]. In this section, we discuss the way we obtain infallible bounds for the functions $f_p(x)$ and $f_p'(x)$. We first give some notations: The set of compact intervals is denoted by $\mathbb{IR}$. If $X = [\underline{x}, \overline{x}]$ is a given interval, the lower bound $\underline{x}$ is referred as $\inf X$ while the upper bound $\overline{x}$ as $\sup X$. The *midpoint* $m(X)$ and the *width* $w(X)$ are defined by $m(X) = (\underline{x} + \overline{x})/2$ and $w(X) = (\overline{x} - \underline{x})$, respectively.

An *inclusion function* F_p of the given function f_p is an interval function (an expression that can be evaluated according to the rules of interval arithmetic) that encloses the range of f_p on all intervals $Y \subseteq X$. The inclusion function F_p is called *inclusion isotone*, if $F_p(Y) \subseteq F_p(X)$ for every $Y \subseteq X$. The inclusion function of the derivative of f_p is denoted by F_p'. Inclusion functions can be produced in a number of ways such as natural extension, mean value forms, and Taylor expansion. Each of these forms have slightly different properties and convergence order. For a more thorough discussion on these issues, see [11].

3.1 Range Bounds of f_p

Replacing the variable x in the definition of f_p by the domain X and evaluating this expression according to the rules of interval arithmetic [1, 9, 10], it is possible to compute validated range bounds. The expression which arises is called *natural interval extension* of $f_p(x)$ to X. Since both $\exp : \mathbb{R} \to \mathbb{R}$ and $\ln : (0, \infty) \to \mathbb{R}$ are monotonic increasing, relation

(2) implies that

$$\begin{aligned} F_p(X) &= \frac{1}{p}\ln\left(\sum_{i=1}^{m}\exp\left(p\cdot F_i(X)\right)\right) \\ &= \left[\frac{1}{p}\ln\left(\sum_{i=1}^{m}\exp\left(p\,u_i\right)\right),\ \frac{1}{p}\ln\left(\sum_{i=1}^{m}\exp\left(p\,v_i\right)\right)\right] \end{aligned} \tag{14}$$

where $u_i = \inf F_i(X)$, $v_i = \sup F_i(X)$ and $F_i : \mathbb{IR} \to \mathbb{IR}$ is an interval extension of $f_i : \mathbb{R} \to \mathbb{R}$, $i \in \{1, \ldots, m\}$. However, the interval extension given by (14) cannot be used in practice since an attempt to compute the $\inf F_p(X)$ or $\sup F_p(X)$ might lead to underflow or overflow problems when p approaches the ∞. We overcome this difficulty by exploiting relation (3) (see also [13, 16]).

If we set $u_j = \max_{1\le i\le m}\{u_i\}$ and $v_k = \max_{1\le i\le m}\{v_i\}$, by utilizing (3) we define the functionals:

$$\phi(u) = u_j + \frac{1}{p}\ln\left(1 + \sum_{i\neq j}^{m}\exp\left(p\left(u_i - u_j\right)\right)\right) \tag{15}$$

and

$$\phi(v) = v_k + \frac{1}{p}\ln\left(1 + \sum_{i\neq k}^{m}\exp\left(p\left(v_i - v_k\right)\right)\right). \tag{16}$$

Therefore, we can rewrite (14) in terms of $\phi(u)$ and $\phi(v)$ as

$$F_p(X) = [\phi(u), \phi(v)], \tag{17}$$

or,

$$F_p(X) = [u_j, v_k] + \frac{1}{p}\cdot[\widetilde{e}_j, \widetilde{e}_k], \tag{18}$$

where $\widetilde{e}_j = \ln\left(1 + \sum_{i\neq j}^{m}\exp\left(p\left(u_i - u_j\right)\right)\right)$, $\widetilde{e}_k = \ln\left(1 + \sum_{i\neq k}^{m}\exp\left(p\left(v_i - v_k\right)\right)\right)$, and $0 \le \widetilde{e}_j \le \widetilde{e}_k \le \ln(m)$. Notice that for sufficiently large p, the term $1/p\,[\widetilde{e}_j, \widetilde{e}_k]$ is negligible.

Now, if $p\left(u_i - u_j\right) < \ln(\texttt{MinMachineNumber}) < 0$, then this term is omitted, when $\phi(u)$ is computed since an attempt to evaluate $\exp(p\left(u_i - u_j\right))$ would lead to underflow. Similarly, if the evaluation of $\exp\left(p\left(v_i - v_k\right)\right)$ would cause underflow, then this term is replaced with the smallest positive machine number `MinMachineNumber`, when $\phi(v)$ is computed to obtain $\sup F_p(X)$.

3.2 Range Bounds of f_p Using Mean Value Form

Since f_p is continuously differentiable, an interval extension of f_p written in the mean value form [9] offers a second order approximation to the range of f_p over an interval X. Assume at the moment that an interval $F_p'(X)$ that encloses the range of f_p' on the interval X is available. Then the interval extension $F_{MVF} : \mathbb{IR} \times \mathbb{R} \to \mathbb{IR}$ of f_p on X defined by

$$F_{MVF}(X, c) = f_p(c) + F_p'(X)\cdot(X - c) \tag{19}$$

is the mean value form of f_p on X with center c. The selection of the center c is a crucial task that permits a more general definition of the mean value form and leads to different inclusions. A common approach is to set $c = m(X)$. An extensive discussion on these

issues have been presented in [15]. Compared with a natural interval extension F_p of f_p, the mean value form F_{MVF} assures tighter enclosures when the interval X is narrow. On the contrary, F_{MVF} may drastically overestimate the range of f_p when the width of X is large. For this reason, it is a common practice to use the intersection $F_{MVF}(X,c) \cap F_p(X)$ for a better estimation of the range.

3.3 Range Bounds of f_p'

In this section, we illustrate the construction of an interval extension $F_p' : \mathbb{IR} \to \mathbb{IR}$ of f_p'. We consider the natural interval extension of $f_p'(x)$ to X. Recalling relation (6), we have that

$$F_p'(X) = \sum_{i=1}^{m} A_i \cdot F_i'(X), \tag{20}$$

where $F_i' : \mathbb{IR} \to \mathbb{IR}$ is an interval extension of $f_i' : \mathbb{R} \to \mathbb{R}$, $i \in \{1, \ldots, m\}$. Our task is now to compute intervals A_i such that $A_i \subseteq [0,1]$. From (7) it is easy to show, after some algebraic manipulations, that multipliers α_i can be written in the following form

$$\alpha_i = \left(1 + \sum_{\ell \neq i} \exp\left(p \ (f_\ell(x) - f_i(x)) \right) \right)^{-1} \tag{21}$$

Considering the natural interval extension of α_i we obtain

$$A_i = \left(1 + \sum_{\ell \neq i} \exp\left(p \ (F_\ell(X) - F_i(X)) \right) \right)^{-1} \tag{22}$$

Since $\exp : \mathbb{R} \to \mathbb{R}$ is monotone increasing, finally, we obtain the following interval

$$A_i = \left[1 + \sum_{\ell \neq i} u_\ell, \ 1 + \sum_{\ell \neq i} v_\ell \right]^{-1} = \left[1 + \sum_{\ell \neq i} v_\ell, \ 1 + \sum_{\ell \neq i} u_\ell \right], \tag{23}$$

where $u_\ell = \exp\left(p \cdot \inf\left(F_\ell(X) - F_i(X) \right) \right)$ and $v_\ell = \exp\left(p \cdot \sup\left(F_\ell(X) - F_i(X) \right) \right)$.

Here, we encounter both underflow and overflow problems during the computation of the interval multipliers A_i. Therefore, we have to examine the following four cases.

(*i*) If $p \sup\left(F_\ell(X) - F_i(X) \right) < \ln(\texttt{MinMachineNumber}) < 0$, then the term v_ℓ is replaced with the smallest positive machine number, `MinMachineNumber`, since an attempt to evaluate $\exp\left(p \cdot \sup\left(F_\ell(X) - F_i(X) \right) \right)$ would lead to underflow.

(*ii*) If $p \sup\left(F_\ell(X) - F_i(X) \right) > \ln(\texttt{MaxMachineNumber}) > 0$, then the $\inf(A_i)$ is replaced with zero since an attempt to evaluate $\exp\left(p \cdot \sup\left(F_\ell(X) - F_i(X) \right) \right)$ would lead to overflow.

(*iii*) If $p \inf\left(F_\ell(X) - F_i(X) \right) < \ln(\texttt{MinMachineNumber}) < 0$, then the term u_ℓ is replaced with zero.

(*iv*) If $p \inf\left(F_\ell(X) - F_i(X) \right) > \ln(\texttt{MaxMachineNumber}) > 0$, then the $\sup(A_i)$ is replaced with the smallest positive machine number `MinMachineNumber`.

4 Pruning Steps Using Derivative Bounds

In this section we describe a pruning technique which is based on first-order information and serves as an accelerating device in the main part of our algorithm. We next present the theoretical aspects as well as algorithmic formulations for the pruning (inner and outer) steps. As we show, inner and outer pruning steps arise from the solution of linear interval inequalities obtained from first-order Taylor expansions. In order to simplify our notations, in what follows we omit the subscript p from the entropic function f_p.

Let $\widetilde{f}$ denotes the smallest already known upper bound of the global minimum f^* and let $Y \subseteq X$ be the current subinterval. The upper bound $\widetilde{f}$ can be used in an attempt to prune Y. Our aim is to find an interval enclosure $\widetilde{Y}$ of the set of points $\widetilde{y}$ such that $f(\widetilde{y}) \leq \widetilde{f}$. Taking into account that $f \in C^1$, we can expand f about a center $c \in Y$, i.e., $f(\widetilde{y}) = f(c) + (\widetilde{y} - c) \cdot f'(\xi)$, where ξ is a point between c and $\widetilde{y}$. Since $c \in Y$ and $\widetilde{y} \in Y$, then $\xi \in Y$. Thus,

$$f(\widetilde{y}) \in f(c) + F'(Y) \cdot (\widetilde{y} - c) \leq \widetilde{f}.$$

By setting $z = \widetilde{y} - c$, $f_c = f(c)$, and $D = [\underline{d}, \overline{d}] = F'(Y)$, we form the following linear interval inequality

$$\left(f_c - \widetilde{f}\right) + D \cdot z \leq 0, \tag{24}$$

where we have retained the simpler noninterval notation for the degenerate interval $[0, 0]$. Keeping in mind the set-valued properties of $\mathbb{IR}$, we can use $\subseteq$ as a partial ordering and we may extend the relation $\leq$ to mean $[\underline{a}, \overline{a}] < [\underline{b}, \overline{b}]$ if and only if $\overline{a} < \underline{b}$. By straightforward but tedious analysis, the solution set Z of the inequality (24) is determined as follows (cf. [4, 5]):

$$Z = \begin{cases} [-\infty,\ (\widetilde{f} - f_c)/\overline{d}] \cup & \\ [(\widetilde{f} - f_c)/\underline{d},\ +\infty], & \text{if } \widetilde{f} < f_c,\ \underline{d} < 0 < \overline{d}, \quad (25a) \\ [-\infty,\ (\widetilde{f} - f_c)/\overline{d}], & \text{if } \widetilde{f} < f_c,\ \underline{d} \geq 0 \text{ and } \overline{d} > 0, \quad (25b) \\ [(\widetilde{f} - f_c)/\underline{d},\ +\infty], & \text{if } \widetilde{f} < f_c,\ \underline{d} < 0 \text{ and } \overline{d} \leq 0, \quad (25c) \\ \emptyset, & \text{if } \widetilde{f} < f_c,\ \underline{d} = \overline{d} = 0, \quad (25d) \\ [-\infty, +\infty], & \text{if } \widetilde{f} \geq f_c,\ \underline{d} \leq 0 \leq \overline{d}, \quad (25e) \\ [-\infty,\ (\widetilde{f} - f_c)/\underline{d}], & \text{if } \widetilde{f} \geq f_c, \underline{d} > 0, \quad (25f) \\ [(\widetilde{f} - f_c)/\overline{d},\ +\infty], & \text{if } \widetilde{f} \geq f_c,\ \overline{d} < 0. \quad (25g) \end{cases}$$

Recall that $z = \widetilde{y} - c$. Therefore, $\widetilde{Y} = c + Z$ is the set of points $\widetilde{y} \in Y$ that satisfy inequality (24). Since we are only interested in points $\widetilde{y} \in Y$, we compute the interval enclosure $\widetilde{Y}$ as

$$\widetilde{Y} = (c + Z) \cap Y. \tag{26}$$

The last two cases of (25) is no of interest since the function is strictly monotone in the entire subinterval and thus it cannot contain any stationary point in its interior. We next deal with the rest of them: If Z is the union of two intervals, say Z_1 and Z_2 (case (25a)), then the interval enclosure $\widetilde{Y}$ is composed by the intervals $\widetilde{Y}_i = (c + Z_i) \cap Y$, $i = 1, 2$. In different case, where Z is a single interval, the desired solution of the inequality (24) is a single interval $\widetilde{Y}$ and as a consequence, the current subinterval Y is pruned either from the right (case (25b)) or from the left (case (25c)). Case (25d) is a special one that occurs only when f is constant ($\underline{d} = \overline{d} = 0$) within Y. Since $\widetilde{f} < f_c$, interval Y can not contain any global minimizer and can be discarded by hand.

4.1 Inner Pruning Step

In this subsection we algorithmically formulate the *inner pruning step*. This step utilizes derivative bounds and an extra function value at a point $c \in Y \subseteq X$. For the case of interest where $0 \in D = [\underline{d}, \overline{d}]$, if we set

$$p = c + (\widetilde{f} - f_c)/\overline{d} \quad \text{and} \quad q = c + (\widetilde{f} - f_c)/\underline{d}$$

then, when $\widetilde{f} < f_c$, relation (26) in connection with (25) takes the following form:

$$\widetilde{Y} = \begin{cases} [\underline{y}, p] \cup [q, \overline{y}], & \text{if } \underline{d} < 0 < \overline{d}, & \text{(27a)} \\ [\underline{y}, p], & \text{if } \underline{d} \geq 0 \text{ and } \overline{d} > 0, & \text{(27b)} \\ [q, \overline{y}], & \text{if } \underline{d} < 0 \text{ and } \overline{d} \leq 0, & \text{(27c)} \\ \emptyset, & \text{if } \underline{d} = \overline{d} = 0. & \text{(27d)} \end{cases}$$

When $\widetilde{f} \geq f_c$ and $\underline{d} \leq 0 \leq \overline{d}$, the interval enclosure $\widetilde{Y}$ coincides with $\widetilde{Y} = [\underline{y}, \overline{y}]$, and hence, no pruning is possible.

In case (27a), the interval enclosure $\widetilde{Y}$ is composed by intervals $Y_1 = [\underline{y}, p]$ and $Y_2 = [q, \overline{y}]$. Since $\widetilde{f} - f_c < 0$, $p < c < q$ and point p express the leftmost point such that $\widetilde{f} < f(p)$, while q is the rightmost point such that $\widetilde{f} < f(q)$. Thus, the global minimum cannot lie in the gap interval (p, q) and therefore interval (p, q) can be discarded with guarantee (see Figure 1). In case (27b) and (27c), the interval enclosure $\widetilde{Y}$ is the single interval Y_1 and Y_2, respectively; gap interval $(p, \overline{y}]$ (resp. $[\underline{y}, q)$) can also be discarded. When (27d) is the case, then the whole interval is discarded, since $\widetilde{Y} = \emptyset$. The properties of interval $\widetilde{Y}$ are summarized in the next theorem:

Theorem 2. *Let $f : \mathcal{D} \to \mathbb{R}$ be a C^1 function, $Y \in \mathbb{IR}$, $c \in Y \subseteq X \subseteq \mathcal{D} \subseteq \mathbb{R}$. Moreover, let $f_c = f(c)$, $0 \in D = F'(Y)$, and $\widetilde{f} \geq \min_{x \in X} f(x)$. Then, interval $\widetilde{Y}$ in (27) has the following properties:*

1. *$\widetilde{Y} \subseteq Y$.*
2. *Every global optimizer x^* of f in X with $x^* \in Y$ satisfies $x^* \in \widetilde{Y}$.*
3. *If $\widetilde{Y} = \emptyset$ then there exists no global minimizer of f in Y.*

Proof. Property 1 follows immediately from the definition of $\widetilde{Y}$ in (27). The proof of Property 2 is implied by the above discussion, while Property 3 is a consequence of Property 2. □

Based on cases (27a)-(27d), where a pruning is possible under the necessary condition $\widetilde{f} < f_c$, we can now formulate the detailed steps of the inner pruning algorithm. Algorithm 1 takes as input the subinterval $Y = [\underline{y}, \overline{y}] \subseteq X$, $c \in Y$, $f_c = f(c)$, the derivative enclosure $D = [\underline{d}, \overline{d}]$ over Y, and the current upper bound $\widetilde{f}$, and returns the pruned (possibly empty) subset $\widetilde{Y} = Y_1 \cup Y_2$ of Y.

In Figure 1 we illustrate the inner pruning step by drawing two lines from point $(c, f(c))$ and intersecting them with the $\widetilde{f}$-axis. The first line with the largest (positive) slope $\overline{d}$ of f in Y intersects the $\widetilde{f}$-axis at point p while the line with the smallest slope $\underline{d}$ intersects the $\widetilde{f}$-axis at point q. Notice that, in geometrical sense, inner pruning step is equivalent to one step of the extended interval Newton's method [3, 4] applied to equation $f(y) - \widetilde{f} = 0$, $y \in Y$.

Algorithm 1 The inner pruning algorithm.

InnerPrune$(Y, c, f_c, D, \widetilde{f}, Y_1, Y_2)$
1: **if** $\overline{d} > 0$ **then**
2: $p := c + (\widetilde{f} - f_c)/\overline{d}$;
3: **if** $p \geq \underline{y}$ **then**
4: $Y_1 := [\underline{y}, p]$;
5: **else**
6: $Y_1 := \emptyset$;
7: **if** $\underline{d} < 0$ **then**
8: $q := c + (\widetilde{f} - f_c)/\underline{d}$;
9: **if** $q \leq \overline{y}$ **then**
10: $Y_2 := [q, \overline{y}]$;
11: **else**
12: $Y_2 := \emptyset$;
13: **return** Y_1, Y_2;

4.2 Outer Pruning Step

Outer pruning step utilizes already known information in an attempt to contract the bounds of the current subinterval $Y \subseteq X$. Let us assume, at the moment, that the function values $f(\underline{y})$ and $f(\overline{y})$ at the two boundary points are available and $0 \in D = F'(Y)$. Expanding f about the endpoints $\underline{y}$ and $\overline{y}$, we obtain two interval inequalities similar to (24):

$$(f(\underline{y}) - \widetilde{f}) + D \cdot z_{\scriptscriptstyle L} \leq 0 \quad \text{and} \quad (f(\overline{y}) - \widetilde{f}) + D \cdot z_{\scriptscriptstyle R} \leq 0 \tag{28}$$

where $z_{\scriptscriptstyle L} = \widetilde{y}_{\scriptscriptstyle L} - \underline{y}$ and $z_{\scriptscriptstyle R} = \widetilde{y}_{\scriptscriptstyle R} - \overline{y}$. In a recent paper, Casado et al. [2] made the key observation that it is possible to prune the current interval without additional cost if we know only a lower bound for the value of f at the boundary points, while the exact function values themselves are not necessary. We exploit this observation in developing the outer pruning step. If $\underline{f_{\scriptscriptstyle L}}$ and $\underline{f_{\scriptscriptstyle R}}$ are the lower bounds of $f(\underline{y})$ and $f(\overline{y})$, respectively, then inequalities in (28) are equivalent to

$$(\underline{f_{\scriptscriptstyle L}} - \widetilde{f}) + D \cdot z_{\scriptscriptstyle L} \leq 0 \quad \text{and} \quad (\underline{f_{\scriptscriptstyle R}} - \widetilde{f}) + D \cdot z_{\scriptscriptstyle R} \leq 0. \tag{29}$$

Our aim is to find the set $\widetilde{Y}$ of points $\widetilde{y}$ satisfying both inequalities of (29). Thus, we request

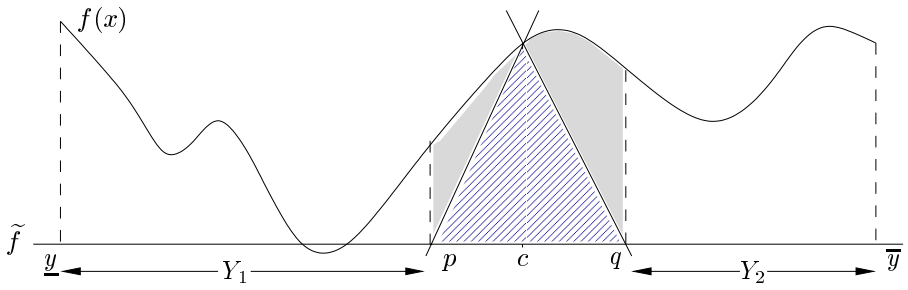

Figure 1: Geometric interpretation of the inner pruning step when $\underline{d} < 0 < \overline{d}$.

$\widetilde{Y} = \widetilde{Y}_L \cap \widetilde{Y}_R$ where, according to (25) and (26),

$$\begin{aligned}
\widetilde{Y}_L &= \left((\underline{y} + Z_L) \cap Y\right) \\
&= \left([-\infty, \underline{y} + (\widetilde{f} - \underline{f_L})/\overline{d}] \cup [\underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}, +\infty]\right) \cap [\underline{y}, \overline{y}] \\
&= \left([-\infty, \underline{y} + (\widetilde{f} - \underline{f_L})/\overline{d}] \cap [\underline{y}, \overline{y}]\right) \cup \left([\underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}, \ +\infty] \cap [\underline{y}, \overline{y}]\right) \\
&= \emptyset \cup [\underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}, \overline{y}] \\
&= [\underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}, \overline{y}], \quad \text{and} \qquad (30)
\end{aligned}$$

$$\widetilde{Y}_R = \left((\overline{y} + Z_R) \cap Y\right) = [\underline{y}, \overline{y} + (\widetilde{f} - \underline{f_R})/\overline{d}]. \qquad (31)$$

The properties of interval $\widetilde{Y}$ are summarized in the next theorem:

Theorem 3. *Let $f : \mathcal{D} \to \mathbb{R}$ be a C^1 function, $Y = [\underline{y}, \overline{y}] \subseteq X \subseteq \mathbb{R}$ and $0 \in F'(Y) = [\underline{d}, \overline{d}]$. Let also $\underline{f_L}$ and $\underline{f_R}$ be the lower bound of $f(\underline{y})$ and $f(\overline{y})$, respectively, and $\widetilde{f} \geq \min_{x \in X} f(x)$ be the current upper bound such that $\widetilde{f} \leq \min\{\underline{f_L}, \underline{f_R}\}$. Then, the interval*

$$\widetilde{Y} = [r, s] = \left[\underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}, \ \overline{y} + (\widetilde{f} - \underline{f_R})/\overline{d}\right] \qquad (32)$$

has the following properties:
1. *$\widetilde{Y} \subseteq Y$.*
2. *Every global optimizer x^* of f in X with $x^* \in Y$ satisfies $x^* \in \widetilde{Y}$.*
3. *If $\widetilde{Y} = \emptyset$, then there exists no global optimizer of f in Y.*

Proof. Property 1 immediately follows from the definition of interval $\widetilde{Y}$ in (32). For the proof of Property 2, assume that there exists an $x^* \in Y \cap [\underline{y}, r)$ such that $f(x^*) = f^*$. Then, there will be a $d_* \in [\underline{d}, \overline{d}]$ satisfying $f(x^*) = f(\underline{y}) + d_*(x^* - \underline{y})$. Thus,

$$\begin{aligned}
f(x^*) &= f(\underline{y}) + d_* \cdot (x^* - \underline{y}) \geq f(\underline{y}) + \underline{d} \cdot (x^* - \underline{y}) \\
&> \underline{f_L} + \underline{d} \cdot (x^* - \underline{y}) > \underline{f_L} + \underline{d} \cdot (r - \underline{y}) = \underline{f_L} + (\widetilde{f} - \underline{f_L}) = \widetilde{f},
\end{aligned}$$

which contradicts with $\widetilde{f} \geq \min_{x \in X} f(x)$. Therefore $x^* \notin \mathcal{Z}$. The case $x^* \notin \mathcal{Z} = Y \cap (s, \overline{y}]$ can be processed similarly. Finally, Property 3 follows from Properties 1 and 2. □

We present now the algorithmic formulation of the outer pruning step described above. Algorithm 2 takes as input the subinterval $Y = [\underline{y}, \overline{y}] \subseteq X$, the derivative enclosure $D = [\underline{d}, \overline{d}]$ over Y, the current upper bound $\widetilde{f}$, and the lower bounds $\underline{f_L}$ (left) and $\underline{f_R}$ (right) of $f(\underline{y})$ and $f(\overline{y})$, respectively, and returns the outwardly pruned (possibly empty) interval Y.

In Figure 2 the geometric interpretation of the outer pruning step is illustrated by using sharp lower bounds for $f(\underline{y})$ and $f(\overline{y})$. Point s is determined as the intersection of the line $h(y) = f(\overline{y}) + \overline{d} \cdot (y - \overline{y})$ with the current upper bound $\widetilde{f}$ for the global minimum. Similarly, point r is given by the line $g(y) = f(\underline{y}) + \underline{d} \cdot (y - \underline{y})$ with the $\widetilde{f}$-axis.

It is obvious that, given an upper bound $\widetilde{f}$, the sharper the value of $\underline{f_L}$ and $\underline{f_R}$ is, the more the current search interval is contracted. The sharpest value of $\underline{f_L}$ and $\underline{f_R}$ is $f(\underline{y})$ and $f(\overline{y})$, respectively. However, this choice would cost two extra function evaluations. We avoid any extra computational effort by utilizing the already known information. We compute the function values only for the starting interval X. After the application of Algorithm 2 at

Algorithm 2 The outer pruning algorithm.

OuterPrune$(Y, \underline{f_L}, \underline{f_R}, D, \widetilde{f})$

1: **if** $\widetilde{f} < \underline{f_R}$ **then** { interval Y can be pruned from the right }
2: **if** $\overline{d} > 0$ **then**
3: $s := \overline{y} + (\widetilde{f} - \underline{f_R})/\overline{d}$;
4: **if** $s \geq \overline{y}$ **then**
5: $Y := [\underline{y}, s]$;
6: **else**
7: $Y := \emptyset$;
8: **if** $\widetilde{f} < \underline{f_L}$ **then** { interval Y can be pruned from the left }
9: **if** $\underline{d} < 0$ **then**
10: $r := \underline{y} + (\widetilde{f} - \underline{f_L})/\underline{d}$;
11: **if** $r \leq \underline{y}$ **then**
12: $Y := [r, \overline{y}]$;
13: **else**
14: $Y := \emptyset$;
15: **return** Y;

the starting interval X as OuterPrune$(X, f(\underline{x}), f(\overline{x}), F'(X), \widetilde{f})$, the value of $\widetilde{f}$ acts as a lower bound, say $\widehat{f}$, for the value of f at the endpoints of the outwardly pruned interval $Y \subseteq X$ (see Figure 2).

4.3 The Pruning Steps Algorithm

Algorithm 1 can be considerably enhanced when combined with Algorithm 2. The combined algorithm (Algorithm 3) aims to discard portions of the search space in which the value of the objective function is always greater than the best known upper bound $\widetilde{f}$ and thus to accelerate the optimization process. We next give a detailed description of the proposed method.

Algorithm 3 takes as input the subinterval $Y = [\underline{y}, \overline{y}]$, the common lower bound $\widehat{f}$ for the value of f at the endpoints of Y, the center $c \in Y$, $f_c = f(c)$, the derivative bounds $D = [\underline{d}, \overline{d}]$, and the current upper bound $\widetilde{f}$, and returns at most two subintervals Y_1 and Y_2 as well as the value of $\widehat{f}$ for the generated subintervals.

Steps 3–8 handle the case where an inner pruning is possible when the condition $\widetilde{f} <$

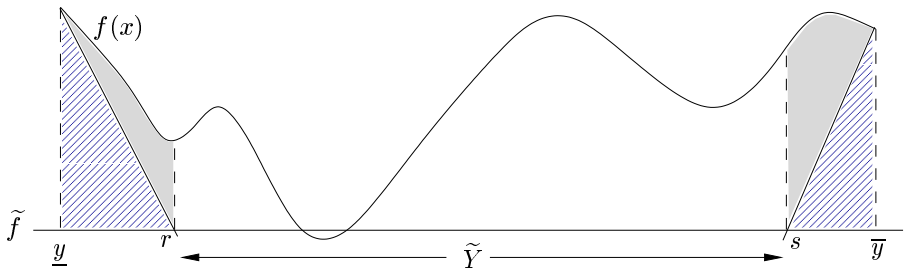

Figure 2: Geometric interpretation of the outer pruning step.

Algorithm 3 The pruning algorithm.

Prune$(Y, \widehat{f}, c, f_c, D, \widetilde{f}, Y_1, Y_2)$

1: $Y_1 := \emptyset$; $Y_2 := \emptyset$; { initialize subintervals Y_1 and Y_2 }
2: **if** $\widetilde{f} < f_c$ **then**
3: InnerPrune$(Y, c, f_c, D, \widetilde{f}, Y_1, Y_2)$; { a inner prune is possible }
4: **if** $Y_1 \neq \emptyset$ **then**
5: OuterPrune$(Y_1, \widehat{f}, \widetilde{f}, D, \widetilde{f})$; { possible outer pruning in Y_1 from the left }
6: **if** $Y_2 \neq \emptyset$ **then**
7: OuterPrune$(Y_2, \widetilde{f}, \widehat{f}, D, \widetilde{f})$; { possible outer pruning in Y_2 from the right }
8: $\widehat{f} := \widetilde{f}$; { set the lower bound for the created subinterval(s) }
9: **else**
10: $Y_1 := [\underline{y}, c]$; $Y_2 := [c, \overline{y}]$; { subdivide the interval at point c }
11: OuterPrune$(Y_1, \widehat{f}, f_c, D, f_c)$; { outer pruning from the left for Y_1 }
12: OuterPrune$(Y_2, f_c, \widehat{f}, D, f_c)$; { outer pruning from the right for Y_2 }
13: $\widehat{f} := f_c$; { set the lower bound for subintervals Y_1 and Y_2 }
14: **return** $Y_1, Y_2, \widehat{f}$;

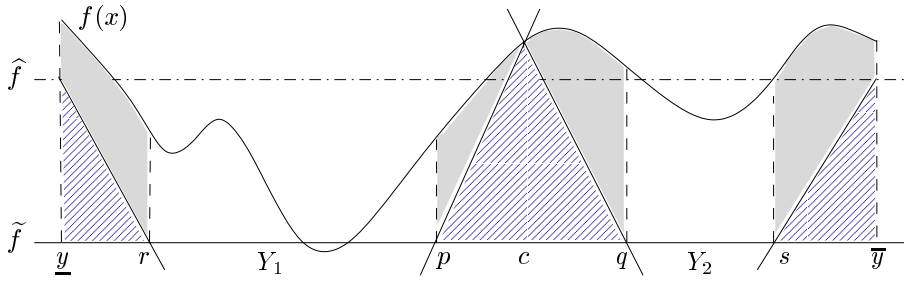

Figure 3: Geometric interpretation of pruning steps in case where $\widetilde{f} < f_c$.

f_c holds. After the application of the inner pruning step, an outer pruning step may be performed at the produced subintervals Y_1 and Y_2 when $\widetilde{f} < \widehat{f}$. This can be done by applying Algorithm 2 to each nonempty subinterval, taking into account that: (i) the function value of f at the right endpoint of Y_1 and the left endpoint of Y_2 is bounded below from $\widetilde{f}$, and (ii) the function value of f at the left endpoint of Y_1 and right endpoint of Y_2 inherit the lower bound $\widehat{f}$ of their ancestor Y (see Figure 3). After the application of the outer pruning steps the endpoints of Y_1 and Y_2 are bounded below by $\widetilde{f}$ (Step 8). A geometrical interpretation of Steps 3–8 is given in Figure 3 where inner and outer pruning steps are combined. Notice that no splitting is necessary since the branching process follows immediately from the inner pruning step.

When inner pruning is not possible (Steps 10–13), Algorithm 3 forms two subintervals, Y_1 and Y_2, by subdividing Y at point c and performs an outer pruning step at the left side of Y_1 and the right side of Y_2 when $\widetilde{f} < \widehat{f}$ (see Figure 4). By this way, a sharp lower bound for the function value at the common endpoint of Y_1 and Y_2 is obtained. It must be pointed out that it is crucial to subdivide at point c where the function value f_c is known; subdividing at any different point would require an extra function evaluation.

Algorithm 3 acts as a new accelerating device that utilizes already known information:

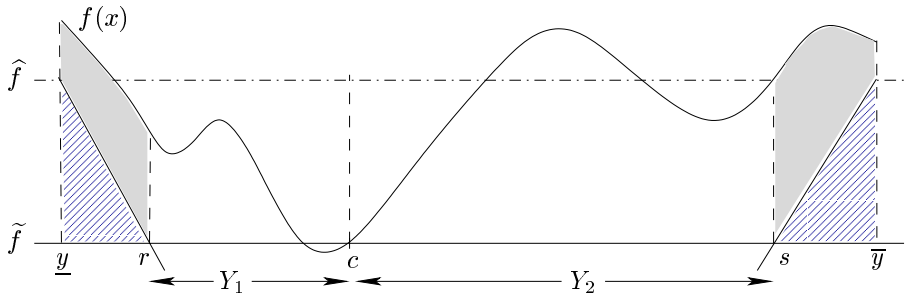

Figure 4: Geometric interpretation of pruning steps in case where $\widetilde{f} \geq f_c$.

the interval derivative $F'(Y)$ and a function value $f(c)$ used in the mean value form to obtain a lower bound for f over Y. The following theorem summarizes the properties of Algorithm 3.

Theorem 4. *Let $f : \mathcal{D} \to \mathbb{R}$ be a C^1 function, $c \in Y \subseteq X \subseteq \mathbb{R}$. Let also $f_c = f(c)$, $D = F'(Y)$, and $\widehat{f} > \widetilde{f} \geq \min_{x \in X} f(x)$, then Algorithm 3 applied as*

$$\mathsf{Prune}(Y, \widehat{f}, c, f_c, D, \widetilde{f}, Y_1, Y_2)$$

has the following properties:

1. $Y_1 \cup Y_2 \subseteq Y$.
2. *Every global optimizer x^* of f in X with $x^* \in Y$ satisfies $x^* \in Y_1 \cup Y_2$.*
3. *If $Y_1 \cup Y_2 = \emptyset$, then there exists no global optimizer of f in Y.*

Proof. It follows from the definition of Y_1 and Y_2 and Theorems 2 and 3. □

5 The Branch-and-Prune Algorithm

The proposed algorithm is based on the branch-and-bound principle. Within the branch and bound framework it is useful to conceptualize the search process in terms of a tree where each node is associated with an interval. The root of the tree is the initial search region X while each node is a subinterval of X having at most two descendants. Generally speaking, the search tree is expanded incrementally by iterating the following steps: (*i*) The initial search space is subdivided into smaller subintervals, (*ii*) The objective function (and possibly its derivatives) is bounded over the subintervals, and (*iii*) Subintervals that definitely cannot contain a global minimizer are pruned (using the calculated bounds).

The efficiency of a branch-and-bound scheme lies in its ability to enumerate most of the branches implicitly. This in turn depends on the bounding efficiency and the pruning tests. In first-order interval methods three criteria are commonly used to ensure that a subinterval Y contains no global minimizer. Then, the corresponding node is pruned and the computation moves to another branch of the tree. We next briefly describe the interval techniques that accelerate the search process:

Function range test: An interval Y is pruned when its lower bound $\inf F(Y)$ is greater than the current upper bound $\widetilde{f}$. When range test fails to prune Y, Y is stored in a list with candidate intervals for further investigation.

Cut-off test: When $\widetilde{f}$ is improved, function range test is applied for all candidate intervals in the list. Obviously, the better the improvement of $\widetilde{f}$ is, the more effective the cut-off test

is.

Monotonicity test: It determines whether the objective function f is strictly monotone in an entire interval Y. If $0 \notin F'(Y)$ then Y is pruned.

The above set of accelerating devices can be augmented by adding the *pruning steps* described in the previous section. As already described, Algorithm 3 is responsible for the branching process when incorporated in a branch-and-bound scheme. When all previous tests fail, Algorithm 3 not only generates the offsprings of the parent node but also it discards parts of the interval where the global minimum does not lie. By this way the search space is reduced and the whole process is further accelerated.

Notice, however, that Algorithm 3 increases the computational effort since it requires the function value $f(c)$ at some point c. This extra cost can be avoided if one exploits information gained by the bounding process. Recall that we have to calculate the function value at the center c when we use the optimal mean value form for bounding the range of f. Since $f(c)$ is then known, we supply c (and $f(c)$) as input to Algorithm 3. Moreover, a function evaluation at some point c aids to improve the upper bound $\widetilde{f}$.

The efficiency of a branch-and-prune method heavily depends on the way the search tree is traversed. In our scheme search is performed according to the best-first strategy where the interval with the smallest value of the inclusion function is selected. Candidate subintervals Y_i are stored in a *working list* $\mathcal{L}$. Elements of $\mathcal{L}$ are sorted in nondecreasing order with respect to their lower bound $\inf F(Y_i)$ and in decreasing order with respect to their age (cf. [3]). The rationale behind this rule is that since we aim to find the global minimum, we should concentrate on the most promising interval, the one with the lowest lower bound.

5.1 Description of the Algorithm

We now give a detailed algorithmic formulation of the proposed global optimization method. All ideas described in the previous sections were incorporated in Algorithm 4. The algorithm takes as input the objective entropic function f_p, the initial search interval X, and the tolerance ϵ, and returns an interval F^* containing the global minimum f^*, along with the result list $\mathcal{L}^*$ of intervals containing all global minimizers.

Initially, the working list $\mathcal{L}$ and the result list $\mathcal{L}^*$ are empty and the upper bound $\widetilde{f}$ is initialized as the minimum value among $f_p(\underline{x})$, $f_p(\overline{x})$, and $f_p(c)$ where c is the midpoint of X. In Steps 4–7, we separately treat the boundary points and add them into the result list $\mathcal{L}^*$ if they are candidates for minimizers. In Step 8, Algorithm 2 is called to outer prune the initial interval X using the values $f_p(\underline{x})$ and $f_p(\overline{x})$ and a common lower bound $\widehat{f}$ for the function values of the endpoints of the pruned interval is set in Step 9. We next bound the range of f_p over X by computing the intersection of the natural extension $F_p(X)$ with the mean value form $F_{MVF}(X, c)$ of f_p. Notice that each member of $\mathcal{L}$ is a structure that contains all necessary information associated with the current interval.

Steps 13–28 are applied to each candidate interval Y until the working list $\mathcal{L}$ is empty. The algorithm takes (and removes) the first element from $\mathcal{L}$ and prunes it according to the procedure described in Section 4.3. For each nonempty subinterval Y_i returned by the pruning step a monotonicity test is applied and when the test fails, the function is evaluated at the midpoint. If the upper bound $\widetilde{f}$ is updated, a cut-off test is applied to $\mathcal{L}$ (Steps 19–21). After bounding the range of f_p over Y_i, a range test is applied in Step 23 and Y_i is added either to the result list $\mathcal{L}^*$ or to the working list $\mathcal{L}$ according to the termination criterion (Steps 24–27).

When no candidate intervals are contained in the working list, the algorithm terminates

by returning an enclosure for the global minimum F^* as well as all elements of the result list $\mathcal{L}^*$. The next theorem establishes the correctness of Algorithm 4.

Theorem 5. *Let $f_p : \mathcal{D} \to \mathbb{R}$, $X \subseteq \mathcal{D} \subseteq \mathbb{R}$, and $\epsilon > 0$. Then Algorithm 4 has the following properties:*

1. $f^* \in F^*$.
2. $X^* \subseteq \bigcup_{(Y, \inf F_Y) \in \mathcal{L}^*} Y$.

Algorithm 4 The branch-and-prune algorithm.

GlobalOptimize$(f_p, X, \epsilon, F^*, \mathcal{L}^*)$

$\mathcal{L} = \{\}$; $\quad \mathcal{L}^\star = \{\}$;
$c = \mathrm{mid}(X)$;
$\widetilde{f} := \min\{\sup F_p(\underline{x}), \sup F_p(\overline{x}), \sup F_p(c)\}$;
if $\widetilde{f} \geq \sup F_p(\underline{x})$ **then**
  $\mathcal{L}^\star := \mathcal{L}^\star \uplus ([\underline{x}, \underline{x}], \inf F_p(\underline{x}))$;
if $\widetilde{f} \geq \sup F_p(\overline{x})$ **then**
  $\mathcal{L}^\star := \mathcal{L}^\star \uplus ([\overline{x}, \overline{x}], \inf F_p(\overline{x}))$;
OuterPrune$(X, \inf F_p(\underline{x}), \inf F_p(\overline{x}), F_p'(X), \widetilde{f})$;
$\widehat{f} := \widetilde{f}$;
$F_X := (F_p(c) + F_p'(X)\,(X - c)) \cap F_p(X)$;
$\mathcal{L} := \mathcal{L} \uplus (X, \inf F_X, F_p'(X), c, F_p(c), \widehat{f})$;
while $\mathcal{L} \neq \{\}$ **do**
  $(Y, \inf F_Y, F_p'(Y), c, F_p(c), \widehat{f}) :=$ PopHead$(\mathcal{L})$;
  Prune$(Y, c, F_p(c), F_p'(Y), \widetilde{f}, \widehat{f}, Y_1, Y_2)$;
  for $i := 1$ to 2 **do**
    if $Y_i = \emptyset$ **then next** i;
    if $0 \notin F_p'(Y_i)$ **then next** i;
    $c = \mathrm{mid}(Y_i)$;
    if $\sup F_p(c) < \widetilde{f}$ **then**
      $\widetilde{f} := \sup F_p(c)$;
      CutOffTest$(\mathcal{L}, \widetilde{f})$;
    $F_Y := (F_p(c) + F_p'(Y_i)\,(Y_i - c)) \cap F_p(Y_i)$;
    if $\inf F_Y \leq \widetilde{f}$ **then**
      if $w(Y_i) \leq \epsilon$ **then**
        $\mathcal{L}^\star := \mathcal{L}^\star \uplus (Y_i, \inf F_Y)$;
      else
        $\mathcal{L} := \mathcal{L} \uplus (Y, \inf F_Y, F_p'(Y_i), c, F_p(c), \widehat{f})$;
  end for
end while
$(Y, \inf F_Y) :=$Head$(\mathcal{L}^\star)$; $\quad F^\star = [\inf F_Y, \widetilde{f}]$;
CutOffTest$(\mathcal{L}^\star, \widetilde{f})$;
return $F^\star$, $\mathcal{L}^\star$;

6 Numerical Results

Algorithm 4 was implemented and tested on 10 test functions given in [13, 16]. The implementation was carried out in C++ using the C-XSC-2.0 library [6]. It should be emphasized that interval arithmetic was used to evaluate $f_p(c)$ in order to bound all rounding errors. Moreover, special care of correct rounding had been taken while computing the pruning points in Algorithms 1 and 2 (for a thorough discussion on these issues see [12]). Numerical results were obtained with $p = 10^{+30}$ and $\epsilon = 10^{-8}$.

	Function eval.			Derivative eval.			Bisections			List length		
No.	M	P	P/M	M	P	P/M	M	P	P/M	M	P	P/M
1	240	18	8%	157	9	6%	78	2	3%	2	1	50%
2	254	12	5%	167	5	3%	83	1	1%	2	2	100%
3	135	19	14%	87	9	10%	43	4	9%	1	1	100%
4	145	29	20%	93	17	18%	46	6	13%	2	1	50%
5	154	32	21%	87	19	22%	43	0	0%	3	3	100%
6	610	92	15%	323	53	16%	161	1	1%	6	4	67%
7	389	101	26%	253	60	24%	126	0	0%	3	3	100%
8	181	80	44%	113	48	42%	56	19	34%	3	2	67%
9	750	206	27%	487	127	26%	243	3	1%	8	8	100%
10	1348	333	25%	879	207	24%	439	8	2%	13	10	77%
$\sum$	4206	922	22%	2646	554	21%	1318	44	3%	43	35	81%
Ø			20%			19%			6%			81%

Table 1: Comparison results between the traditional method (M) and the branch-and-prune method (P).

The proposed algorithm (P), which has been described in detail in Algorithm 4 is compared with a simpler version (M) in which no pruning steps are used. Specifically, algorithm (M), uses a monotonicity test, a cut-off test, and bisection as subdivision rule. In both algorithms, the bounds were obtained from the intersection of the mean value form with the natural interval extension of the entropic function f_p, where the center of the form is the midpoint of the interval.

Tables 1 summarize numerical comparison between the proposed algorithm (P) and algorithm (M). For each test function we report the number of entropic function evaluations, the number of entropic derivative evaluations, the number of bisections, and the maximum list length. The last two rows of the table give the total and the average values for the complete test set. According to Table 1, the proposed method (P) always outperforms the traditional method. On average, we had 80% improvement in the number of function evaluations, 81% in the number of derivative evaluations, and 19% improvement in the list length. The 94% improvement in the number of bisections reveals the fact that for the branching process our algorithm mostly utilizes inner pruning rather than bisection. Bisections mainly takes place when the global minimum has been reached and the algorithm tries to isolate the global minimizer within an interval in such a way that termination criterion to be fulfilled.

7 Conclusions

In this work we have presented an interval algorithm for computing verified enclosures for the global minimum and all global minimizers of discrete minimax problems in $\mathbb{R}$. The algorithm uses first-order information by applying a smoothing technique and utilizes a pruning step that accelerates the search process. Moreover, the pruning step is responsible for the branching process and eliminates large parts of the search space where the global minimum does not exist. Our numerical results show that the proposed method exhibits rich potentials and always outperforms the traditional method which uses only a monotonicity test. Our future work in this area will be on extending the proposed method for multidimensional problems.

Acknowledgement

The author would like to thank the editors and the anonymous referees whose suggestions help to improve the presentation of this article.

Appendix: Test Functions, Search Intervals, and Solutions

The test set consists of one-dimensional problems presented in [13, 16].

1. $f(x) = \max\{x/4,\ x(1-x)\}, \quad X = [1/2, 3/2]$
   $X^* = \{3/4\}, \quad f^* = 3/16.$

2. $f(x) = \max\{|x|,\ |x-1|\}, \quad X = [-1, 2],$
   $X^* = \{1/2\}, \quad f^* = 1/2.$

3. $f(x) = \max\{|x|,\ |x-1|,\ -|x-2|\}, \quad X = [-3, 3],$
   $X^* = \{1\}, \quad f^* = 1.$

4. $f(x) = \max\{\sin(x/10),\ \cos(x/10)\}, \quad X = [0, 20\pi + 1],$
   $X^* = \{50\pi/4\}, \quad f^* = -1/\sqrt{2}.$

5. $f(x) = \max\{x(2-x),\ (x-1)(3-x)\}, \quad X = [5/4, 5/2],$
   $X^* = \{3/2,\ 5/2\}, \quad f^* = 3/4.$

6. $f(x) = \max\{-2x + 3x^2 - x^3,\ 2x - 3x^2 + x^3\}, \quad X = [0, 2],$
   $X^* = \{0,\ 1,\ 2\}, \quad f^* = 0.$

7. $f(x) = \max\{f_1(x),\ f_2(x),\ f_3(x)\}, \quad X = [0, 6],$
   $f_1(x) = x(x-2)(x-4)(x-6)(x-8),$
   $f_2(x) = x(2-x)(x-4)(x-6)(x-8),$
   $f_3(x) = (x-2)(x-7).$
   $X^* = \{2,\ 4,\ 6\}, \quad f^* = 0.$

8. $f(x) = \max\{f_1(x),\, f_2(x),\, f_3(x),\, f_4(x)\}, \quad X = [0,6],$
$f_1(x) = x(x-1)(x-2)(x-3),$
$f_2(x) = (1/2-x)(x-3/2)(x-5/2),$
$f_3(x) = (x-1)(x-4),$
$f_4(x) = (x-2)(x-5).$
$X^* = \{2.785...\}, \quad f^* = -0.838....$

9. $f(x) = \max\{\sin(10x),\, \cos(10x)\}, \quad X = [-2,2],$
$X^* = \{\frac{(8k-3)\pi}{40} \mid k = -2,-1,0,1,2,3\}, \quad f^* = -1/\sqrt{2}.$

10. $f(x) = \max\{\sin(x),\, \cos(x)\}, \quad X = [0,20\pi],$
$X^* = \{\frac{(8k+5)\pi}{4} \mid k = 0,\ldots,9\}, \quad f^* = -1/\sqrt{2}.$

References

[1] G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, London, 1983.

[2] L.G. Casado, J.A. Martínez, I. García, and Ya.D. Sergeyev, New interval analysis support functions using gradient information in a global minimization algorithm, *Journal of Global Optimization* 25(4) (2003) 345–362.

[3] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz, *C++ Toolbox for Verified Computing I, Basic Numerical Problems: Theory, Algorithms, and Programs*, Springer-Verlag, 1995.

[4] E. Hansen, *Global Optimization using Interval Analysis*, Marcel Dekker, New York, 1992.

[5] E. Hansen and S. Sengupta, Global constained optimization using interval analysis, in *Interval Mathematics 1980*, K. Nickel (ed.), Springer-Verlag, Berlin, 1980, pp. 25–47.

[6] W. Hofschuster and W. Krämer, C-XSC 2.0 - A C++ library for extended scientific computing, in *Numerical Software with Result Verification: International Dagstuhl Seminar, Dagstuhl Castle, Germany, January 19-24, 2003* G. Goos, J. Hartmanis, and J. van Leeuwen (eds.) Vol. 2991, Lecture Notes in Computer Science, Springer, 2004, pp. 15–35.

[7] Z.Y. Huang, An interval entropy penalty method for nonlinear global optimization, *Reliable Computing* 4 (1998) 15–25.

[8] E.T. Jaynes, Information theory and statistical mechanics, *Phys. Rev.* 106 (1957) 620–630.

[9] R.E. Moore, *Interval Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1966.

[10] A. Neumaier, *Interval Methods for systems of equations*, Cambridge University Press, 1990.

[11] H. Ratschek and J. Rokne, *Computer Methods for the Range of Functions*, Ellis Horwood, Chichester, 1984.

[12] D. Ratz, *Automatic Slope Computation and its Application in Nonsmooth Global Optimization*, Shaker Verlag, Aachen, 1998.

[13] Z. Shen, Z.Y. Huang, and M.A. Wolfe, An interval maximum entropy method for a discrete minimax problem, *Applied Mathematics and Computation* 87 (1997) 49–68.

[14] Z. Shen, A. Neumaier, and M.C. Eiermann, Solving minimax problems by interval methods, *BIT* 30 (1990) 742–751.

[15] D.G. Sotiropoulos and T.N. Grapsa, Optimal centers in branch-and-prune algorithms for univariate global optimization, *Applied Mathematics and Computation* 169 (2005) 247–277.

[16] M.A. Wolfe, On discrete minimax problems in $\mathbb{R}$ using interval arithmetic, *Reliable Computing* 5 (1999) 371–383.

Manuscript received 7 January 2005
revised 19 November 2005, 27 February 2006
accepted for publication 27 February 2006

D.G. Sotiropoulos
Ionian University, Computer Science Department, Plateia Tsirigoti 7, GR-491 00, Corfu, Greece
E-mail address: dgs@ionio.gr