An Interval Branch-and-Prune Algorithm for Discrete Minimax Problems

D.G. Sotiropoulos* and **T.N. Grapsa**

University of Patras, Department of Mathematics, GR-265 04 Rio, Patras, Greece.

We present an interval branch-and-prune method for a discrete minimax problem 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]. Our algorithm works within the branch and bound framework and incorporates a new accelerating device that composes inner and outer pruning steps. Pruning is achieved by using first order information of the entropic objective function by means of an interval evaluation. Our algorithm was implemented and tested on a complete test set and numerical results are presented.

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. The 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 $f(x) = \max\{f_1(x), \ldots, f_m(x)\}$.

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 a recent paper an interval branch-and-prune algorithm for computing verified enclosures for the global minimum and all global minimizers has been developed for the one-dimensional case [2]. In relation to this previous work, the present study is specialized to the minimax problem (1). Our approach is based on smoothing the max-type function by exploiting the Jaynes's maximum entropy [1] for transforming problem (1) into the problem of minimizing a continuously differentiable function.

* Corresponding author. E-mail: dgs@math.upatras.gr, Phone: +30 2610 997332, Fax: +30 2610 992965.

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 [1] proposed the Principle of Maximum Entropy which has proved to be useful for obtaining convergent upper bounds on non-smooth objective functions.

Let p be a positive integer and let $f_i : X \to \mathbb{R}$ $(i = 1, \ldots, m)$ be given functions. The maximum entropy function $f_p : X \to \mathbb{R}$ of $f_1, \ldots, f_m$ is defined as

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

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

$$f_p'(x) = \sum_{i=1}^{m} \alpha_i(x) \cdot f_i'(x) \quad \text{where} \quad \alpha_i(x) = \exp\left(p \cdot f_i(x)\right) / \sum_{j=1}^{m} \exp\left(p \cdot f_j(x)\right) \tag{3}$$

Since $\exp(p \cdot f_i(x))$ becomes quite large when p approaches to ∞, to prevent overflow, special care must be taken in computing $f_p(x)$ and $\alpha_i(x)$ (see [3]). It can be easily proved that $f(x) \leq f_p(x) \leq f(x) + \ln(m)/p$; hence, $f_p(x)$ decreases monotonically as p increases and, for any $x \in \mathbb{R}$, $f_p(x) \to f(x)$, as $p \to \infty$. This property has been exploited by several authors [5, 4, 3].

3 Interval extension of the entropic function

Let $\mathbb{IR} = \{[a, b] \mid a \leq b,\ a, b \in \mathbb{R}\}$ be the set of compact intervals. If $X = [\underline{x}, \overline{x}]$ is a given interval, the lower bound $\underline{x}$ is referred as $\inf X$ and the upper bound $\overline{x}$ as $\sup X$.

We call a function $F\colon \mathbb{IR} \to \mathbb{IR}$ an *inclusion function* or an *interval extension* of $f\colon \mathbb{R} \to \mathbb{R}$ in X, if $x \in X$ implies $f(x) \in F(X)$. In other words, $f(X) = f_{\mathrm{rg}}(X) = \{f(x) \mid x \in X\} \subseteq F(X)$ if $f_{\mathrm{rg}}(X) \subseteq F(X)$, where $f(X)$ or $f_{\mathrm{rg}}(X)$ both denote the range of the function f on X. The inclusion function of the derivative of f is denoted by F'. Interval extensions can be computed via interval arithmetic.

An interval extension of $f_p(x)$ may be computed as follows: Since both $\exp : \mathbb{R} \to \mathbb{R}$ and $\ln : (0, \infty) \to \mathbb{R}$ are monotonic increasing, relation (2) implies that

$$F_p(X) = \frac{1}{p} \ln \left(\sum_{i=1}^{m} \exp\left(p \cdot F_i(X)\right) \right) = \left[\frac{1}{p} \ln \left(\sum_{i=1}^{m} \exp\left(p\, u_i\right) \right),\ \frac{1}{p} \ln \left(\sum_{i=1}^{m} \exp\left(p\, v_i\right) \right) \right] \tag{4}$$

where $u_i = \inf F_i(X)$ and $v_i = \sup F_i(X)$ in which $F_i : \mathbb{IR} \to \mathbb{IR}$ is an interval extension of $f_i : \mathbb{R} \to \mathbb{R}$. By similar way, we can construct an interval extension $F_p' : \mathbb{IR} \to \mathbb{IR}$ of f_p' using relation (3).

4 The branch-and-prune algorithm

The main advantage of the above smoothing technique is that it convert a minimax problem into a smooth optimization problem that can be solved by any interval branch-and-bound method. Recently the notion of linear (inner and outer) pruning steps has been introduced to develop an efficient branch-and-prune optimization algorithm [2]. Pruning steps act as a accelerating device that utilizes already known information to eliminate regions which cannot contain global minimum points.

We next give a detailed algorithmic formulation of the proposed method. The algorithm takes as input the entropic objective function f_p, the initial search interval X, and the tolerance ϵ, and returns an interval $F^\star$ containing the global minimum f^*, along with the result list $\mathcal{L}^\star$ of intervals containing all global minimizers.

Initially, the working list $\mathcal{L}$ and the result list $\mathcal{L}^\star$ are empty. An upper bound $\widetilde{f}$ is initialized as the minimum value among $f_p(\underline{x})$, $f_p(\overline{x})$, and $f_p(c)$. In Steps 4–7, we separately treat the boundary points and add them into the result list if they are candidates for minimizers. In Step 8, the initial interval X is pruned outwardly 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 as the intersection of the natural extension $F_p(X)$ with the mean value form of f_p, where the center c is the midpoint of X. Steps 13–28 are applied to each candidate interval

Algorithm 1 The branch-and-prune algorithm.

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

1: $\mathcal{L} = \{\}$; $\mathcal{L}^\star = \{\}$;
2: $c = \mathrm{mid}(X)$;
3: $\widetilde{f} := \min\{\sup F_p(\underline{x}), \sup F_p(\overline{x}), \sup F_p(c)\}$;
4: **if** $\widetilde{f} \geq \sup F_p(\underline{x})$ **then**
5: $\quad \mathcal{L}^\star := \mathcal{L}^\star \uplus ([\underline{x}, \underline{x}], \inf F_p(\underline{x}))$;
6: **if** $\widetilde{f} \geq \sup F_p(\overline{x})$ **then**
7: $\quad \mathcal{L}^\star := \mathcal{L}^\star \uplus ([\overline{x}, \overline{x}], \inf F_p(\overline{x}))$;
8: OuterPrune($X, \inf F_p(\underline{x}), \inf F_p(\overline{x}), F_p'(X), \widetilde{f}$);
9: $\widehat{f} := \widetilde{f}$;
10: $F_X := (F_p(c) + F_p'(X)\,(X - c)) \cap F_p(X)$;
11: $\mathcal{L} := \mathcal{L} \uplus (X, \inf F_X, F_p'(X), c, F_p(c), \widehat{f})$;
12: **while** $\mathcal{L} \neq \{\}$ **do**
13: $\quad (Y, \inf F_Y, F_p'(Y), c, F_p(c), \widehat{f}) :=$ PopHead($\mathcal{L}$);
14: $\quad$ Prune($Y, c, F_p(c), F_p'(Y), \widetilde{f}, \widehat{f}, Y_1, Y_2$);
15: $\quad$ **for** $i := 1$ to 2 **do**
16: $\quad\quad$ **if** $Y_i = \emptyset$ **then next** i;
17: $\quad\quad$ **if** $0 \notin F_p'(Y_i)$ **then next** i;
18: $\quad\quad c = \mathrm{mid}(Y_i)$;
19: $\quad\quad$ **if** $\sup F_p(c) < \widetilde{f}$ **then**
20: $\quad\quad\quad \widetilde{f} := \sup F_p(c)$;
21: $\quad\quad\quad$ CutOffTest($\mathcal{L}, \widetilde{f}$);
22: $\quad\quad F_Y := (F_p(c) + F_p'(Y_i)\,(Y_i - c)) \cap F_p(Y_i)$;
23: $\quad\quad$ **if** $\inf F_Y \leq \widetilde{f}$ **then**
24: $\quad\quad\quad$ **if** $d_{\mathrm{rel}}(F_Y) \leq \epsilon$ **or** $d_{\mathrm{rel}}(Y_i) \leq \epsilon$ **then**
25: $\quad\quad\quad\quad \mathcal{L}^\star := \mathcal{L}^\star \uplus (Y_i, \inf F_Y)$;
26: $\quad\quad\quad$ **else**
27: $\quad\quad\quad\quad \mathcal{L} := \mathcal{L} \uplus (Y, \inf F_Y, F_p'(Y_i), c, F_p(c), \widehat{f})$;
28: $\quad$ **end for**
29: **end while**
30: $(Y, \inf F_Y)$:=Head($\mathcal{L}^\star$); $F^\star = [\inf F_Y, \widetilde{f}]$;
31: CutOffTest($\mathcal{L}^\star, \widetilde{f}$);
32: **return** $F^\star$, $\mathcal{L}^\star$;

Y until the working list $\mathcal{L}$ is empty. The algorithm takes (and removes) the first element from $\mathcal{L}$ and prune it. For each nonempty subinterval Y_i returned by the pruning step monotonicity test is applied and when test fails, a center c is selected as the midpoint of Y_i. When the upper bound $\widetilde{f}$ is updated, cut-off test is applied to $\mathcal{L}$ (Steps 19–21). After bounding the range of f_p over Y_i, range test is applied in Step 23 and Y_i is added either to the result list $\mathcal{L}^\star$ 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 $F^\star$ as well as all elements of the result list $\mathcal{L}^\star$.

5 Numerical examples

The algorithm was implemented in C++ using the C-XSC-2.0 library. The inclusion functions were implemented via the C-XSC routines while the derivatives are computed with forward automatic differentiation. To test the algorithm performance, a test set of 20 problems collected in [5, 3] along with some new test cases were used. Our numerical results show that the algorithm always returns guaranteed bounds for f^* and x^*.

Example. We minimize the function $f(x) = \max\{\sin(10x), \cos(10x)\}$ taken from [5]. The global minimum is $f^* = -1/\sqrt{2}$, while the set of global minimizers is $X^* = \{\frac{(8k-3)\pi}{40} \mid k = -2, -1, 0, 1, 2, 3\}$.
Applying our algorithm using the *maximum entropy function* and *pruning steps*, we get

```
Search interval         :[-2,2]
Tolerance (relative)    :  1E-8
No. of function eval.   :   182
No. of derivative eval. :   111
No. of bisections       :     3
Necessary list length   :     8

1.[ 1.649336139646693E+000, 1.649336147168703E+000]   4.560628264212071E-009
2.[-1.492256510728526E+000,-1.492256510161631E+000]   3.798900639630000E-010
3.[-2.356194490324211E-001,-2.356194487040641E-001]   1.393589956321010E-009
4.[-8.639379797503177E-001,-8.639379797176424E-001]   3.782122866392218E-011
5.[ 1.021017612406508E+000, 1.021017612429809E+000]   2.281996333939525E-011
6.[ 3.926990816888396E-001, 3.926990817101174E-001]   5.418322276484709E-011

  [-7.071068058500655E-001,-7.071067811170894E-001]   3.497771010131201E-008
encloses the global minimum value!

  6 interval enclosure(s)
```

References

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

[2] D.G. Sotiropoulos and T.N. Grapsa. Optimal Centers in Branch-and-Prune Algorithms for Global Optimization. Submmited for publication, 2004. `http://www.math.upatras.gr/~dgs/pub.htm`.

[3] M.A. Wolfe. On discrete minimax problems in R using interval arithmetic. *Reliable Computing*, 5:371–383, 1999.

[4] Huang Zhen Yu. An interval entropy penalty method for nonlinear global optimization. *Reliable Computing*, 4:15–25, 1998.

[5] Shen Zuhe, Huang Zhen Yu, and M.A. Wolfe. An interval maximum entropy method for a discrete minimax problem. *Applied Mathematics and Computation*, 87:49–68, 1997.