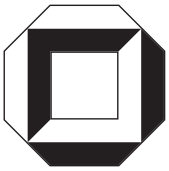

UNIVERSITÄT KARLSRUHE

A Second-Order Pruning Step for Verified Global Optimization

Marco Schnurr

Preprint Nr. 07/10

Institut für Wissenschaftliches Rechnen
und Mathematische Modellbildung

76128 Karlsruhe

Anschrift des Verfassers:

Dr. Marco Schnurr
Institut für Angewandte und Numerische Mathematik
Universität Karlsruhe
D-76128 Karlsruhe

A Second-Order Pruning Step for Verified Global Optimization*

Marco Schnurr

Institute for Applied and Numerical Mathematics, University of Karlsruhe, D-76128 Karlsruhe, e-mail: marco.schnurr@math.uni-karlsruhe.de

Abstract

We consider pruning steps used in a branch-and-bound algorithm for verified global optimization. A first-order pruning step was given by Ratz using automatic computation of a first-order slope tuple [21, 22]. In this paper, we introduce a second-order pruning step which is based on automatic computation of a second-order slope tuple. We add this second-order pruning step to the algorithm of Ratz. Furthermore, we compare the new algorithm with the algorithm of Ratz by considering some test problems for verified global optimization on a floating-point computer.

Keywords: global optimization, interval analysis, pruning step

MSC 2000: 65G20, 65K05, 90C56

1 Introduction

Let $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be continuous and $[x] \subseteq D$. Our aim is to find guaranteed two-sided bounds for the global minimum

$$f^* \quad := \quad \min_{x \in [x]} f(x)$$

and for all global minimizers $x^* \in [x]$. We require that the bounds satisfy a specified accuracy. For details, see section 5.

Common methods to address this problem are branch-and-bound algorithms using interval analysis. These algorithms are due to Hansen [8, 9], Ichida/Fujii [11] and Skelboe [34]. The approach is as follows. The interval $[x]$ is partitioned, and subintervals are discarded, once it is proven that they do not contain a global minimizer. The remaining subintervals are partitioned again until we have achieved the required accuracy. Interval analysis guarantees that no global minimizer is lost in the algorithm, even though the computation is performed on a floating-point computer.

Acceleration tools are crucial to obtaining acceptable computation times. If f is continuously differentiable, then a "monotonicity-test" discards intervals that do not contain a

*This paper contains some results from the author's dissertation [30].

zero of f'. Other tools, such as the "concavity test" or the interval newton method, use evaluations of f''. Furthermore, enclosures of the range of f' or f'' may provide better enclosures of the range of f than an interval arithmetic evaluation of f (see [1]). The "mean value form" is such an approach. Enclosures of f' and f'' can be obtained by automatic differentiation [20]. Introductions to global optimization methods using interval analysis can be found in [10] and [12].

Slope enclosures [15] can be used to compute enclosures of the range of f that are sharper than range enclosures provided by the mean value form. However, slope enclosures do not detect the monotonicity of a function. Therefore, other box-discarding techniques are required. Ratz [21, 23] introduces a pruning step that uses slope enclosures for eliminating subintervals of the current interval that do not contain a global minimizer. This method may also be used for nonsmooth functions. Ratz obtains a slope enclosure as an element of a slope tuple which may be computed by a technique analogous to automatic differentiation.

In this paper, we extend the method of Ratz [21, 23] by introducing a second-order pruning step. Furthermore, we include this second-order pruning step in a branch-and-bound algorithm for verified global optimization. We compare this approach with the algorithm of Ratz by considering some test problems. For the implementation we assume that f is locally Lipschitz continuous on $[x]$, that f is given by a function expression (cf. [1]), and that the interval arithmetic evaluation of f on $[x]$ exists. Then, by expanding techniques due to Shen/Wolfe [33] and Kolev [14], a second-order slope tuple can be computed [28, 30]. The source code of our program is freely available [26].

The paper is organized as follows. In sections 2 and 3, we introduce slope enclosures and explain the automatic computation of first-order and second-order slope tuples. Section 4 describes the componentwise computation of slope tuples which is used in our algorithm, and section 5 provides an introduction to global optimization using interval analysis. In section 6, we introduce a second-order pruning step for univariate functions. We apply this pruning step to global optimization of multivariate functions and state our algorithm in section 7. Finally, in section 8, we consider some examples and compare the new algorithm with the algorithm of Ratz.

Throughout this paper, $[x] = [\underline{x}, \overline{x}] = \left\{ x \in \mathbb{R}^n \middle| \, \underline{x}_i \leq x_i \leq \overline{x}_i \right\}$ with $\underline{x}, \overline{x} \in \mathbb{R}^n$ denotes an interval vector, and $\mathbb{IR}^n$ the set of all interval vectors $[x] \subseteq \mathbb{R}^n$. The midpoint of $[x]$ is denoted by $\operatorname{mid}[x] := \frac{1}{2}(\underline{x} + \overline{x})$. Furthermore, for an interval $[x] \in \mathbb{IR}$, we define the diameter $\operatorname{diam}[x] \in \mathbb{R}$ by $\operatorname{diam}[x] := \overline{x} - \underline{x}$, and the relative diameter $\operatorname{diam}_{\mathrm{rel}}[x] \in \mathbb{R}$ by

$$\operatorname{diam}_{\mathrm{rel}}[x] := \begin{cases} \dfrac{\operatorname{diam}[x]}{\min\{|x|, x \in [x]\}}, & \text{if } 0 \notin [x], \\ \operatorname{diam}[x], & \text{otherwise.} \end{cases}$$

2 Slope functions and slope enclosures

Slope enclosures provide enclosures of the function range which may be sharper than those obtained by using derivatives (see [15]). Furthermore, slope enclosures can be used for computational existence tests such as the Moore test [17] and tests based on Miranda's theorem [5, 25]. In this section, we give the definitions needed in the sequel.

Definition 2.1 Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous and $x_0 \in D$ be fixed. A function $\delta f : D \to \mathbb{R}$ satisfying

$$f(x) = f(x_0) + \delta f(x; x_0) \cdot (x - x_0), \quad x \in D, \tag{1}$$

is called a *(first-order) slope function of f with respect to x_0*.

An interval $\delta f([x]; x_0) \in \mathbb{IR}$ containing the range of $\delta f(x; x_0)$ on $[x] \subseteq D$, i.e. a $\delta f([x]; x_0) \in \mathbb{IR}$ with

$$\delta f([x]; x_0) \supseteq \{\delta f(x; x_0) \mid x \in [x]\},$$

is called a *(first-order) slope enclosure of f on $[x]$ with respect to x_0*.

If $x = x_0$, then (1) holds for arbitrary $\delta f(x_0; x_0) \in \mathbb{R}$. If f is differentiable in x_0, then we set $\delta f(x_0; x_0) := f'(x_0)$.

Let $\delta f([x]; x_0)$ be a first-order slope enclosure of f on $[x]$. Then,

$$f(x) \in f(x_0) + \delta f([x]; x_0) \cdot ([x] - x_0) \tag{2}$$

holds for all $x \in [x]$. Obviously, (2) may provide sharper enclosures of the range of f on $[x]$ than the mean value form [16].

For the continuous function $f : \mathbb{R} \to \mathbb{R}$,

$$f(x) = \begin{cases} \sqrt{x} & \text{for } x \geq 0, \\ 0 & \text{for } x < 0, \end{cases}$$

and $x_0 = 0$, $[x] = [-1, 1]$, a first-order slope enclosure $\delta f([x]; 0) \in \mathbb{IR}$ of f on $[x]$ with respect to x_0 does not exist. In order to give a sufficient existence statement we define the *limiting slope interval* (see [19]).

Definition 2.2 Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous on $[x] \subseteq D$, and let $x_0 \in [x]$. If

$$\liminf_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} \in \mathbb{R} \quad \text{and} \quad \limsup_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} \in \mathbb{R},$$

then the *limiting slope interval* $\delta f_{\lim}([x_0]) \in \mathbb{IR}$ is

$$\delta f_{\lim}([x_0]) := \left[\liminf_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}, \limsup_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} \right].$$

Obviously, the limiting slope interval $\delta f_{\lim}([x_0])$ exists, if f is Lipschitz continuous in some neighbourhood of x_0. If f is differentiable in x_0, we have

$$\delta f_{\lim}([x_0]) = \left[f'(x_0), f'(x_0) \right].$$

Lemma 2.3 Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous on $[x] \subseteq D$ and let $x_0 \in [x]$. We assume that $\delta f_{\lim}([x_0]) \in \mathbb{IR}$ exists. Then,

$$\delta f([x]; x_0) = \left[\inf_{\substack{x \in [x] \\ x \neq x_0}} \frac{f(x) - f(x_0)}{x - x_0}, \sup_{\substack{x \in [x] \\ x \neq x_0}} \frac{f(x) - f(x_0)}{x - x_0} \right]$$

is a first-order slope enclosure of f on $[x]$ with respect to x_0.

Proof: see [19]. □

Remark 2.4 Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be Lipschitz continuous in some neighbourhood of $x_0 \in D$. Muñoz and Kearfott [19] show the inclusion

$$\delta f_{\lim}([x_0]) \quad \subseteq \quad \partial f(x_0), \tag{3}$$

where $\partial f(x_0)$ is the generalized gradient [4]. Furthermore, they give a sufficient condition for equality in (3).

Definition 2.5 Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous, $[x] \subseteq D$, and $x_0 \in [x]$. Furthermore, we assume that $\delta f_{\lim}([x_0])$ exists. An interval $\delta_2 f([x]; x_0) \in \mathbb{IR}$ statisfying

$$f(x) \in f(x_0) + \delta f_{\lim}([x_0]) \cdot (x - x_0) + \delta_2 f([x]; x_0) \cdot (x - x_0)^2, \quad x \in [x], \tag{4}$$

is called a *second-order slope enclosure of f on $[x]$ with respect to x_0*.

3 Automatic computation of slope tuples

In the following sections, we assume that each function is given by a function expression in the sense of [1], i.e. the function expression consists of a finite number of operations $+, -, \cdot, /$ and a finite number of elementary functions. Furthermore, we assume that an interval arithmetic evaluation on the interval $[x]$ exists.

Definition 3.1 [21, 23] Let $u : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous, $[x] \subseteq D$, and $x_0 \in [x]$. A triple $\mathcal{U} = (U_x, U_{x_0}, \delta U)$ with $U_x, U_{x_0}, \delta U \in \mathbb{IR}$ satisfying

$$\begin{aligned} u(x) &\in U_x, \\ u(x_0) &\in U_{x_0}, \\ u(x) - u(x_0) &\in \delta U \cdot (x - x_0), \end{aligned}$$

for all $x \in [x]$ is called a *first-order slope tuple of u on $[x]$ with respect to x_0*.

Definition 3.2 Let $u : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous, $[x] \subseteq D$, and $x_0 \in [x]$. A *second-order slope tuple of u on $[x]$ with respect to x_0* is a 5-tuple $\mathcal{U} = (U_x, U_{x_0}, \delta U_{x_0}, \delta U, \delta_2 U)$ with $U_x, U_{x_0}, \delta U_{x_0}, \delta U, \delta_2 U \in \mathbb{IR}$, $U_{x_0} \subseteq U_x$, satisfying

$$\begin{aligned} u(x) &\in U_x, && (5)\\ u(x_0) &\in U_{x_0}, && (6)\\ \delta u_{\lim}([x_0]) &\subseteq \delta U_{x_0}, && (7)\\ u(x) - u(x_0) &\in \delta U \cdot (x - x_0), && (8)\\ u(x) - u(x_0) &\in \delta U_{x_0} \cdot (x - x_0) + \delta_2 U \cdot (x - x_0)^2 && (9) \end{aligned}$$

for all $x \in [x]$.

Automatic differentiation [20] is a technique to compute function and derivative values simultaneously without requiring an explicit formula for the derivative. By combining this

technique with interval analysis, we obtain enclosures of the function and the derivative range on some interval $[x]$.

By using an arithmetic for slope tuples analogous to automatic differentiation, first-order slope tuples can be computed without requiring an explicit formula of a slope function (see [15]). For approaches using nonsmooth elementary functions, such as $\varphi(x) = \operatorname{abs}(u(x))$, $\varphi(x) = \min(u(x), v(x))$, and functions given by two or more branches, see [12, 21, 24, 27, 31]. Furthermore, the enclosures may be sharpened by exploiting a unique point of inflection [29].

An arithmetic for the automatic computation of second-order slope tuples is given in [28, 30]. This extends results contained in [14, 33]. In [28, 30], the expression of the considered function may also contain nonsmooth elementary functions, such as $\varphi(x) = \operatorname{abs}(u(x))$, $\varphi(x) = \min(u(x), v(x))$, and functions given by two or more branches. A second-order slope tuple, more precisely relation (9), may provide a sharper enclosure of the function range than a first-order slope tuple. For details see [28, 30].

4 The componentwise computation of slope tuples

For $u : D \subseteq \mathbb{R}^n \to \mathbb{R}$ it is possible to perform the automatic computation of first-order and second-order slope tuples. For details see [21] and [28, 30], respectively. However, as explained by Ratz [21], a pruning step using such slope tuples would be very costly and not effective. Therefore, for multivariate functions, we use an approach called the "componentwise computation of slope tuples" [21]. The idea is to reduce the problem to the one-dimensional case. We briefly summarize this technique. It will be combined with a first-order and a second-order pruning step for univariate functions (see sections 5 and 6).

Definition 4.1 Let $u : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be continuous on $[x] \subseteq D$ and $i \in \{1, \ldots, n\}$ be fixed. We define the family $\mathcal{G}_i$ of functions by

$$\mathcal{G}_i \;:=\; \left\{ \begin{array}{l} g : [x]_i \subseteq \mathbb{R} \to \mathbb{R}, \quad g(t) := u(x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n) \\ \\ \text{where } x_j \in [x]_j \text{ is fixed for } j \in \{1, \ldots, n\},\ j \neq i. \end{array} \right\} \tag{10}$$

Each $g \in \mathcal{G}_i$ is a function of one variable. Thus, as described in section 3, for each $g \in \mathcal{G}_i$ the automatic computation of a slope tuple on the interval $[x]_i$ with respect to $(x_0)_i \in [x]_i$, $(x_0)_i \in \mathbb{R}$, can be performed.

Similar to Definition 3.2, we introduce a second-order slope tuple for the componentwise computation.

Definition 4.2 Let $u : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be continuous on $[x] \in \mathbb{IR}^n$, $[x] \subseteq D$. Furthermore, let $i \in \{1, \ldots, n\}$ and $(x_0)_i \in \mathbb{R}$, $(x_0)_i \in [x]_i$ be fixed. A *second-order slope tuple of u on $[x]$ with respect to the component i* is a 5-tuple $\mathcal{U} = (U_x, U_{x_0}, \delta U_{x_0}, \delta U, \delta_2 U)$, with $U_x, U_{x_0}, \delta U_{x_0}, \delta U, \delta_2 U \in \mathbb{IR}$, $U_{x_0} \subseteq U_x$, such that

$$\begin{array}{rcl} g(x_i) & \in & U_x, \\ g\big((x_0)_i\big) & \in & U_{x_0}, \\ \delta g_{\lim}\big([x_0]_i\big) & \subseteq & \delta U_{x_0}, \\ g(x_i) - g\big((x_0)_i\big) & \in & \delta U \cdot \big(x_i - (x_0)_i\big), \\ g(x_i) - g\big((x_0)_i\big) & \in & \delta U_{x_0} \cdot \big(x_i - (x_0)_i\big) + \delta_2 U \cdot \big(x_i - (x_0)_i\big)^2 \end{array}$$

holds for all $x_i \in [x]_i$ and all $g \in \mathcal{G}_i$. Here, $\mathcal{G}_i$ is defined as in (10).

The automatic computation of a second-order slope tuple $\mathcal{U}$ of u on $[x]$ with respect to the component i is analogous to the one-dimensional technique from section 3 (see [28, 30]).

Suppose $\mathcal{U}$ is a second-order slope tuple of u on $[x]$ with respect to the component i. Then, for all $x \in [x]$ we have

$$u\,(x) \;\in\; U_x, \tag{11}$$

$$u\,(x) \;\in\; U_{x_0} + \delta U \cdot \Big([x]_i - (x_0)_i \Big) \tag{12}$$

and

$$u\,(x) \;\in\; U_{x_0} + \delta U_{x_0} \cdot \Big([x]_i - (x_0)_i \Big) + \delta_2 U \cdot \Big([x]_i - (x_0)_i \Big)^2. \tag{13}$$

Therefore, (11)-(13) are enclosures of the range of u on $[x] \in \mathbb{IR}^n$.

Remark 4.3 We use a technique similar to the slope computation by Hansen [7, 10] in order to sharpen the enclosures (11)-(13). Let $u : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be continuous and $x_0 \in [x] \subseteq D$ be fixed. We have

$$\begin{aligned} u\Big(x_1, \ldots, x_n\Big) - u\Big((x_0)_1 , \ldots, (x_0)_n \Big) \\ = \;& u\Big(x_1, \ldots, x_n\Big) - u\Big((x_0)_1 , x_2, \ldots, x_n\Big) + u\Big((x_0)_1 , x_2, \ldots, x_n\Big) \\ & - u\Big((x_0)_1 , (x_0)_2 , x_3, \ldots, x_n\Big) + u\Big((x_0)_1 , (x_0)_2 , x_3, \ldots, x_n\Big) \\ & - + \cdots + u\Big((x_0)_1 , \ldots, (x_0)_{n-1} , x_n\Big) - u\Big((x_0)_1 , \ldots, (x_0)_n \Big). \end{aligned}$$

For each $i \in \{1, \ldots, n\}$, we consider the function

$$u_i : \Big((x_0)_1 , \ldots, (x_0)_{i-1} , [x]_i , [x]_{i+1} , \ldots, [x]_n \Big) \to \mathbb{R}$$

with

$$\begin{aligned} u_i\,(x) \;:=\; & u\Big((x_0)_1 , \ldots, (x_0)_{i-1} , x_i, x_{i+1}, \ldots, x_n\Big) \\ & \text{for } x \in \Big((x_0)_1 , \ldots, (x_0)_{i-1} , [x]_i , [x]_{i+1} , \ldots, [x]_n \Big). \end{aligned}$$

We now compute a second-order slope tuple $\mathcal{U}_i := (U_{x;i}, U_{x_0;i}, \delta U_{x_0;i}, \delta U_i, \delta_2 U_i)$ of u_i on $\big((x_0)_1 , \ldots, (x_0)_{i-1} , [x]_i , [x]_{i+1} , \ldots, [x]_n \big)$ with respect to the component i. Then, we have the enclosures

$$u\,(x) \;\in\; U_{x;1}, \tag{14}$$

$$u\,(x) \;\in\; U_{x_0;n} + \sum_{j=1}^{n} \delta U_j \cdot \Big([x]_j - (x_0)_j \Big), \tag{15}$$

$$u\,(x) \;\in\; U_{x_0;n} + \sum_{j=1}^{n} \delta U_{x_0;j} \cdot \Big([x]_j - (x_0)_j \Big) + \sum_{j=1}^{n} \delta_2 U_j \cdot \Big([x]_j - (x_0)_j \Big)^2 \tag{16}$$

for all $x \in [x]$.

5 Global optimization using interval analysis

Let $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be continuous and $[x] \subseteq D$. Our aim is to find guaranteed two-sided bounds for the global minimum

$$f^* \quad := \quad \min_{x \in [x]} f(x)$$

and for all global minimizers $x^* \in [x]$ so that the accuracy condition (17) is satisfied. This condition has also been used in [21]. Branch-and-bound algorithms using interval analysis are suitable for solving this problem. We continue by describing the general idea.

For the branch-and-bound algorithm we use a list $\mathcal{L}$ for intermediate and a list $\mathcal{Q}$ for final results. The elements of $\mathcal{L}$ and $\mathcal{Q}$ are pairs $\left([y], \underline{fy}\right)$ consisting of an interval vector $[y] = \left([y]_1, \ldots, [y]_n\right) \in \mathbb{IR}^n$ and a real number $\underline{fy}$ such that

$$\underline{fy} \leq \min_{y \in [y]} f(y).$$

Furthermore, we need a real number $\tilde{f}$ that is an upper bound of f^* in each step of the algorithm. We initialize the algorithm with an enclosure $f_{[x]}$ of the range of f on $[x]$ which may be obtained by an interval arithmetic evaluation of f (see [1]). Furthermore, we generate the pair $\left([x], \inf f_{[x]}\right)$ as the first element of $\mathcal{L}$. $\mathcal{Q}$ is initialized as an empty list. Moreover, we initialize $\tilde{f}$ by $\tilde{f} := \sup f_{[x]}$.

In the first step of the algorithm, we remove the first pair $\left([x], \inf f_{[x]}\right)$ from $\mathcal{L}$ and subdivide $[x]$. For each subinterval $[y] \subseteq [x]$ we compute an enclosure $f_{[y]}$ of the range of f on $[y]$ and generate the pair $\left([y], \inf f_{[y]}\right)$. If $\tilde{f} < \inf f_{[y]}$, then $[y]$ does not contain a global minimizer, and the pair $\left([y], \inf f_{[y]}\right)$ is discarded ("range check"). Furthermore, if $\tilde{f} > \sup f_{[y]}$, then $\tilde{f}$ is replaced by $\sup f_{[y]}$. If

$$\max_{j=1,\ldots,n} \operatorname{diam}_{\mathrm{rel}} [y]_j \leq \epsilon \quad \text{or} \quad \operatorname{diam}_{\mathrm{rel}} f_{[y]} \leq \epsilon, \tag{17}$$

then the pair $\left([y], \inf f_{[y]}\right)$ is stored in $\mathcal{Q}$, otherwise in $\mathcal{L}$. In the next step, we remove the next pair contained in $\mathcal{L}$ and proceed as before. The algorithm stops as soon as $\mathcal{L}$ is empty. Let $\left([q]_i, \inf f_{[q]_i}\right)$, $i = 1, \ldots, n$, be the pairs in $\mathcal{Q}$ after the algorithm has stopped. Then, all global minimizers of f are contained in the union of the $[q]_i$. Furthermore, we have $f^* \in \left[\min_i \left\{\inf f_{[q]_i}\right\}, \tilde{f}\right]$. Machine interval arithmetic on a floating-point computer guarantees these enclosures. For details of the algorithm see [10, 12].

It is crucial to apply some acceleration tools for the branch-and-bound algorithm above. There are some effective tools using derivatives such as the monotonicity test, the concavity test, and the interval newton step. Ratz [21, 23] introduces a first-order pruning step as an acceleration tool that also applies to nonsmooth functions: Checking a subinterval $[y] \subseteq [x]$, he gets the enclosure

$$f(x) \in \left[\underline{fx_0}, \overline{fx_0}\right] + \left[\underline{\delta f}, \overline{\delta f}\right] \cdot (x_i - (x_0)_i) \qquad \text{for all } x \in [y], \tag{18}$$

where $x_0 \in [y]$ is fixed, $\left[\underline{fx_0}, \overline{fx_0}\right] \in \mathbb{IR}$, and $\left[\underline{\delta f}, \overline{\delta f}\right] \in \mathbb{IR}$. (18) is obtained by the componentwise computation of a first-order slope tuple with respect to the component i. Hence, the graph of f on $[y]$ is bounded by hyperplanes that only depend on x_i.

By intersecting these hyperplanes with the level $\tilde{f}$, we obtain a subset of $[y]$ which does not contain a global minimizer of f. Computing these intersections is a one-dimensional problem, because the right hand side of (18) only depends on x_i.

We note that similar pruning steps can be carried out using enclosures of the derivative if f is continuously differentiable [35, 37, 38]. Then, the enclosure (18) does not depend on x_0, so that an "optimal" x_0 can be computed (cf. [2]). Compared to that, for Ratz' pruning step we are able to use (first-order) slope tuples. This may provide sharper enclosures of the range of f and applies to some nonsmooth functions as well.

In the next section, we introduce a second-order pruning step which may be combined with the componentwise computation of a second-order slope tuple.

6 A second-order pruning step

As explained above, by using the componentwise computation of slope tuples, the pruning step becomes a one-dimensional problem. Hence, we only consider univariate functions $f : D \subseteq \mathbb{R} \to \mathbb{R}$ in this section.

Let f be continuous on $[x] \subseteq D$, and let $c \in [y] = \left[\underline{y}, \overline{y}\right] \subseteq [x]$ be fixed. We assume that we have given an enclosure

$$f(x) - f(c) \in \left[\underline{\delta f c}, \overline{\delta f c}\right] \cdot (x - c) + \left[\underline{\delta_2 f}, \overline{\delta_2 f}\right] \cdot (x - c)^2 \quad \text{for all } x \in [y], \tag{19}$$

which may be obtained via automatic computation of a second-order slope tuple. Furthermore, let $\left[\underline{fc}, \overline{fc}\right]$ be an interval containing $f(c)$. Then, the range of f on $[y]$ is enclosed by the parabolas

$$f(x) \geq \underline{fc} + \overline{\delta f c} \cdot (x - c) + \underline{\delta_2 f} \cdot (x - c)^2 =: g_1(x) \qquad \text{for } \underline{y} \leq x \leq c, \tag{20}$$

$$f(x) \leq \overline{fc} + \underline{\delta f c} \cdot (x - c) + \overline{\delta_2 f} \cdot (x - c)^2 =: g_2(x) \qquad \text{for } \underline{y} \leq x \leq c, \tag{21}$$

$$f(x) \geq \underline{fc} + \underline{\delta f c} \cdot (x - c) + \underline{\delta_2 f} \cdot (x - c)^2 =: g_3(x) \qquad \text{for } c \leq x \leq \overline{y}, \tag{22}$$

$$f(x) \leq \overline{fc} + \overline{\delta f c} \cdot (x - c) + \overline{\delta_2 f} \cdot (x - c)^2 =: g_4(x) \qquad \text{for } c \leq x \leq \overline{y}, \tag{23}$$

(see Figure 1), as opposed to [21, 23], where the graph of f is enclosed by straight lines.

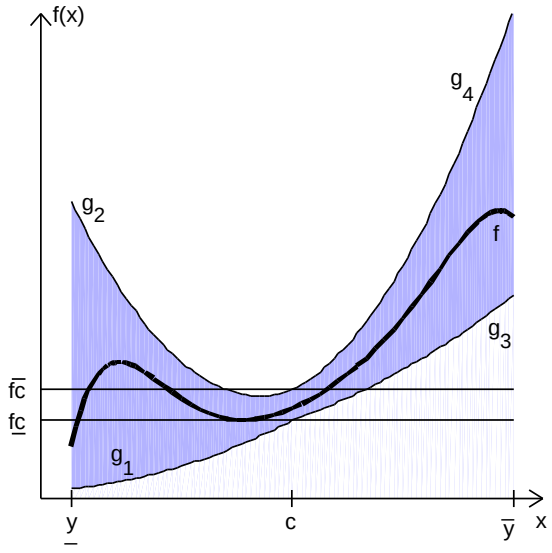

Figure 1: Enclosing the range of f by (20)-(23)

Let

$$\widetilde{f} \geq f^* = \min_{x\in[x]} f(x) \tag{24}$$

be an upper bound for the global minimum of f on $[x]$.

For the two quadratic equations

$$\widetilde{f} = \underline{fc} + \overline{\delta fc}\cdot(x-c) + \underline{\delta_2 f}\cdot(x-c)^2 \tag{25}$$

and

$$\widetilde{f} = \underline{fc} + \underline{\delta fc}\cdot(x-c) + \underline{\delta_2 f}\cdot(x-c)^2 \tag{26}$$

we define the discriminants

$$D_p := \left(\frac{\overline{\delta fc}}{2\,\underline{\delta_2 f}}\right)^2 - \frac{\left(\underline{fc}-\widetilde{f}\right)}{\underline{\delta_2 f}} = \left(\overline{\delta fc}^2 - 4\,\underline{\delta_2 f}\left(\underline{fc}-\widetilde{f}\right)\right)/\left(4\,\underline{\delta_2 f}^2\right) \tag{27}$$

and

$$D_q := \left(\frac{\underline{\delta fc}}{2\,\underline{\delta_2 f}}\right)^2 - \frac{\left(\underline{fc}-\widetilde{f}\right)}{\underline{\delta_2 f}} = \left(\underline{\delta fc}^2 - 4\,\underline{\delta_2 f}\left(\underline{fc}-\widetilde{f}\right)\right)/\left(4\,\underline{\delta_2 f}^2\right). \tag{28}$$

Now, we can state the second-order pruning step. First, we define *Assumption A* needed for the following theorems.

Assumption A: Let $f : D \subseteq \mathbb{R} \to \mathbb{R}$ be continuous on $[x]$, $c \in [y] = \left[\underline{y}, \overline{y}\right] \subseteq [x]$ and $f(c) \in \left[\underline{fc}, \overline{fc}\right]$. Furthermore, assume that the intervals $\left[\underline{\delta fc}, \overline{\delta fc}\right] \in \mathbb{IR}$ and $\left[\underline{\delta_2 f}, \overline{\delta_2 f}\right] \in \mathbb{IR}$ satisfy (19), and assume that $\widetilde{f} \in \mathbb{R}$ satisfies (24).

Theorem 6.1 Suppose that Assumption A holds. Furthermore, assume that $\underline{\delta_2 f} < 0$. Set

$$p := \left\{ \begin{array}{ll} \min\left\{c,\ c - \dfrac{\overline{\delta fc}}{2\,\underline{\delta_2 f}} - \sqrt{D_p},\ c - \dfrac{\overline{\delta fc}}{\underline{\delta_2 f}}\right\}, & \text{if } D_p > 0, \\ \min\left\{c,\ c - \dfrac{\overline{\delta fc}}{\underline{\delta_2 f}}\right\}, & \text{otherwise,} \end{array} \right\} \tag{29}$$

$$q := \left\{ \begin{array}{ll} \max\left\{c,\ c - \dfrac{\underline{\delta fc}}{2\,\underline{\delta_2 f}} + \sqrt{D_q},\ c - \dfrac{\underline{\delta fc}}{\underline{\delta_2 f}}\right\}, & \text{if } D_q > 0, \\ \max\left\{c,\ c - \dfrac{\underline{\delta fc}}{\underline{\delta_2 f}}\right\}, & \text{otherwise,} \end{array} \right\} \tag{30}$$

and

$$\mathcal{Z} := \left\{ \begin{array}{ll} \emptyset, & \text{if } p = q = c, \\ (p,q) \cap [y], & \text{otherwise.} \end{array} \right.$$

Then, we have

$$f(x) > f^*, \quad x \in \mathcal{Z}.$$

Proof: If $D_p > 0$, then the quadratic equation (25) has the solutions

$$p_1 := c - \frac{\overline{\delta fc}}{2\,\underline{\delta_2 f}} - \sqrt{D_p} \tag{31}$$

and

$$p_2 := c - \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} + \sqrt{D_p}. \tag{32}$$

Therefore, by $\underline{\delta_2 f} < 0$ we get

$$f(x) > \widetilde{f} \geq f^* \quad \text{for all } x \in (p_1, p_2) \cap \left[\underline{y}, c\right]. \tag{33}$$

In order to prove

$$\begin{cases} f(x) > f^* \text{ for all } x \in (p, c), & \text{if } p < c \text{ and } q \leq c, \\ f(x) > f^* \text{ for all } x \in (p, c], & \text{if } p < c \text{ and } q > c, \end{cases} \tag{34}$$

we distinguish four cases:

(i) Suppose $D_p > 0$, $p_2 > c$ and $\widetilde{f} \leq \underline{fc}$.

Then, we have

$$\sqrt{D_p} \geq \left| \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} \right|.$$

By (31) we get

$$\min\left\{p_1,\ c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}}\right\} = p_1 \leq c.$$

Therefore, using (33) we obtain

$$f(x) > f^* \quad \text{for all } x \in (p, p_2) \cap \left[\underline{y}, c\right]$$

with p from (29). This implies (34).

(ii) Suppose $D_p > 0$, $p_2 > c$ and $\widetilde{f} > \underline{fc}$.

Then, we have

$$\sqrt{D_p} < \left| \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} \right|,$$

and by (32) we have $\overline{\delta f c} > 0$. Therefore,

$$p = \min\left\{c,\ c - \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} - \sqrt{D_p},\ c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}}\right\} = c < p_1$$

holds. By (33) we obtain

$$f(x) > f^* \quad \text{for all } x \in (p, p_2) \cap \left[\underline{y}, c\right] = \emptyset.$$

(iii) Suppose $D_p > 0$ and $p_2 \leq c$.

Then, by (32) we have $\overline{\delta f c} < 0$. Because of $\underline{\delta_2 f} < 0$ we get

$$f(c) + \overline{\delta f c} \cdot (x - c) + \underline{\delta_2 f} \cdot (x - c)^2 > f(c)$$

for all

$$x \in \left(c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}},\ c\right) \cap \left[\underline{y}, c\right]. \tag{35}$$

Hence, by (19),

$$f(x) > f(c) \geq f^*$$

holds for all x from (35). Using

$$c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}} \; < p_2 \; \leq \; c,$$

we obtain

$$f(x) > f^* \quad \text{for all } x \in (p, c) \cap \left[\underline{y}, c\right] \tag{36}$$

from (33).

Because of $\overline{\delta f c} < 0$ we also have $\underline{\delta f c} < 0$. Therefore, if $q > c$ holds for q from (30), then we have

$$\sqrt{D_q} \; > \; \frac{\underline{\delta f c}}{2\,\underline{\delta_2 f}}. \tag{37}$$

Because (37) implies $\underline{fc} > \widetilde{f}$, we get

$$f(x) > f^* \quad \text{for all } x \in (p, c] \cap \left[\underline{y}, c\right], \quad \text{if } q > c. \tag{38}$$

From (36) and (38) we get (34).

(iv) Suppose $D_p \leq 0$.

If

$$\min\left\{c, \; c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}}\right\} = c$$

holds, then there is nothing to show. If

$$\min\left\{c, \; c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}}\right\} = c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}} < c,$$

then we have $\overline{\delta f c} < 0$. Analogously to (iii) we get (36) and (38) which gives (34).

Analogously to (i)-(iv), we proceed for the quadratic equation (26). Combining the results, we get

$$f(x) > f^*, \quad x \in (p, q) \cap [y],$$

if $p < c$ or $q > c$. □

Corollary 6.2 If the assumptions of Theorem 6.1 hold, then each $x^* \in [y]$ that is a global minimizer of f on $[x]$ is contained in

$$(-\infty, p] \; \cap \; \left[\underline{y}, c\right]$$

or in

$$[c, \overline{y}\,] \cap [q, \infty).$$

If $p < \underline{y}$ and $q > \overline{y}$, then $[y]$ cannot contain a global minimizer of f on $[x]$.

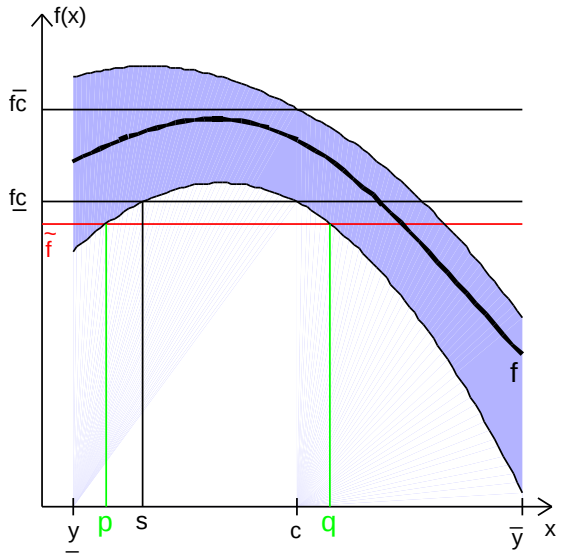

Figure 2: Illustration of Theorem 6.1

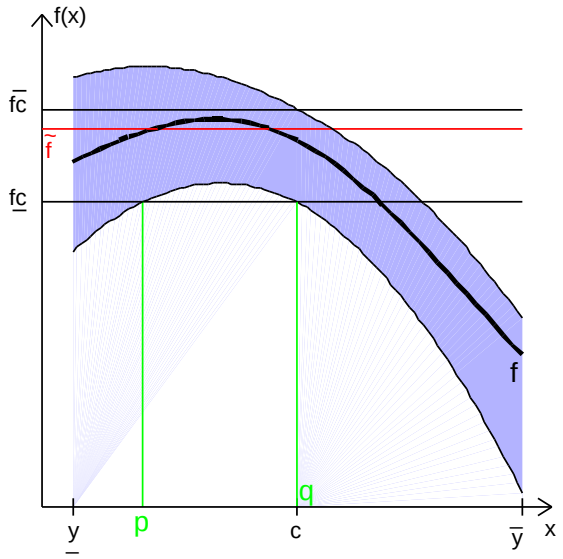

Figure 3: Theorem 6.1 in the case of $D_p > 0$, $p = c - \dfrac{\overline{\delta f c}}{\underline{\delta_2 f}}$, $D_q < 0$

Figure 2 illustrates Theorem 6.1 in the case of

$$D_p > 0, \quad p = c - \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} - \sqrt{D_p}, \quad D_q > 0, \quad q = c - \frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} + \sqrt{D_q}.$$

In the diagram we have

$$s := c - \frac{\overline{\delta f c}}{\underline{\delta_2 f}} > p.$$

Figure 3 illustrates Theorem 6.1 in the case of

$$D_p > 0, \quad p = c - \frac{\overline{\delta fc}}{\underline{\delta_2 f}}, \quad D_q < 0.$$

In both figures we have $f(x) > f^*$ for all $x \in (p, q)$. The other cases can be illustrated analogously.

Theorem 6.3 Suppose that Assumption A holds. Furthermore, assume that $\underline{\delta_2 f} = 0$. Set

$$p := \left\{ \begin{array}{ll} c + \left(\widetilde{f} - \underline{fc}\right) / \overline{\delta fc}, & \text{if } \overline{\delta fc} > 0 \text{ and } \widetilde{f} < \underline{fc}, \\ -\infty, & \text{if } \overline{\delta fc} < 0, \\ -\infty, & \text{if } \overline{\delta fc} = 0 \text{ and } \widetilde{f} < \underline{fc}, \\ c, & \text{if } \overline{\delta fc} \geq 0 \text{ and } \widetilde{f} \geq \underline{fc}, \end{array} \right\} \tag{39}$$

$$q := \left\{ \begin{array}{ll} c + \left(\widetilde{f} - \underline{fc}\right) / \underline{\delta fc}, & \text{if } \underline{\delta fc} < 0 \text{ and } \widetilde{f} < \underline{fc}, \\ +\infty, & \text{if } \underline{\delta fc} > 0, \\ +\infty, & \text{if } \underline{\delta fc} = 0 \text{ and } \widetilde{f} < \underline{fc}, \\ c, & \text{if } \underline{\delta fc} \leq 0 \text{ and } \widetilde{f} \geq \underline{fc}, \end{array} \right\} \tag{40}$$

and

$$\mathcal{Z} := \left\{ \begin{array}{ll} \emptyset, & \text{if } p = q = c, \\ (p, q) \cap [y], & \text{otherwise.} \end{array} \right.$$

Then, we have

$$f(x) > f^*, \quad x \in \mathcal{Z}. \tag{41}$$

Proof: Because of $\underline{\delta_2 f} = 0$,

$$f(x) \;\geq\; f(c) + \overline{\delta fc} \cdot (x - c) \;\geq\; \underline{fc} + \overline{\delta fc} \cdot (x - c) \tag{42}$$

holds for all $x \in \left[\underline{y}, c\right]$. We consider the four cases in (39):

(i) If $\overline{\delta fc} > 0$ and $\widetilde{f} < \underline{fc}$, then by (42) we have

$$f(x) \;\geq\; \underline{fc} + \overline{\delta fc} \cdot (x - c) \;>\; \widetilde{f} \;\geq\; f^*$$

for all $x \in \left(c + (\widetilde{f} - \underline{fc}) / \overline{\delta fc},\; c\right]$.

(ii) If $\overline{\delta fc} < 0$, then by (42) we obtain

$$f(x) > f(c) \geq f^*$$

for all $x \in \left[\underline{y}, c\right)$. If, additionally, $q > c$ holds, then by $\underline{\delta fc} \leq \overline{\delta fc} < 0$ and (40) we have $\underline{fc} > \widetilde{f}$. Thus, we get

$$f(x) \;>\; f^* \quad \text{for all } x \in \left[\underline{y}, c\right].$$

(iii) If $\overline{\delta fc} = 0$ and $\widetilde{f} < \underline{fc}$, then by (42) we have

$$f(x) \;\geq\; \underline{fc} \;>\; \widetilde{f} \;\geq\; f^*$$

for all $x \in \left[\underline{y}, c\right]$.

(iv) If $\overline{\delta fc} \geq 0$ and $\widetilde{f} \geq \underline{fc}$, then we have $p = c$.

Analogously, we proceed for the cases for q. Combining (i)-(iv) and all cases for q we obtain

$$f(x) > \widetilde{f} \geq f^*, \quad x \in (p, q) \cap [y], \quad \text{if } p < c \text{ or } q > c.$$

This proves (41). □

Remark 6.4 Corollary 6.2 also holds, if the assumptions of Theorem 6.3 instead of Theorem 6.1 are satisfied.

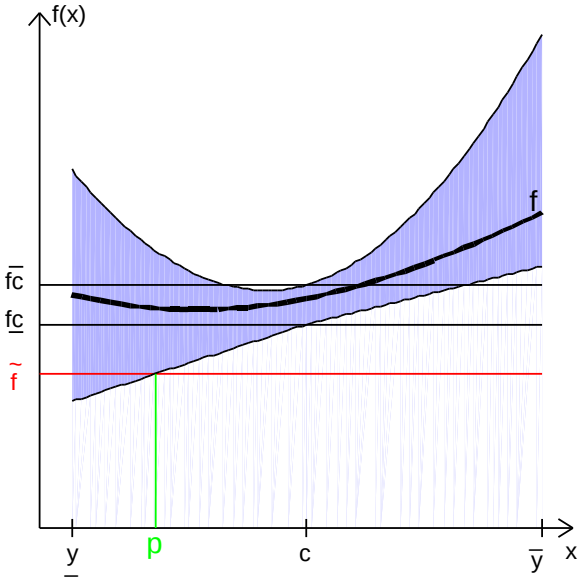

Figure 4: Illustration of Theorem 6.3

Figure 4 illustrates Theorem 6.3 in the case of

$$0 < \underline{\delta fc} \leq \overline{\delta fc} \text{ and } \widetilde{f} < \underline{fc}.$$

The search for global minimizers $x^* \in \left[\underline{y}, \overline{y}\right]$ of f on $[x]$ can be restricted to the interval $\left[\underline{y}, p\right]$.

Theorem 6.5 Suppose that Assumption A holds. Furthermore, assume that $\underline{\delta_2 f} > 0$. Set

$$p_1 := \begin{cases} c - \dfrac{\overline{\delta fc}}{2\,\underline{\delta_2 f}} - \sqrt{D_p}\,, & \text{if } D_p \geq 0\,, \\ \underline{y} - 1\,, & \text{if } D_p < 0\,, \end{cases}$$

$$p_2 := \begin{cases} c - \dfrac{\overline{\delta fc}}{2\,\underline{\delta_2 f}} + \sqrt{D_p}\,, & \text{if } D_p \geq 0\,, \\ \underline{y} - 1\,, & \text{if } D_p < 0\,, \end{cases}$$

$$q_1 \;:=\; \begin{cases} c - \dfrac{\underline{\delta f c}}{2\,\underline{\delta_2 f}} - \sqrt{D_q}\,, & \text{if } D_q \geq 0\,, \\ \overline{y} + 1\,, & \text{if } D_q < 0\,, \end{cases}$$

$$q_2 \;:=\; \begin{cases} c - \dfrac{\underline{\delta f c}}{2\,\underline{\delta_2 f}} + \sqrt{D_q}\,, & \text{if } D_q \geq 0\,, \\ \overline{y} + 1\,, & \text{if } D_q < 0\,, \end{cases}$$

and

$$\hat{\mathcal{Z}} := \Big([p_1, p_2] \cap \big[\, \underline{y}, c \,\big] \Big) \cup \Big([q_1, q_2] \cap [\, c, \overline{y}] \Big).$$

Then, we have

$$f\,(x) > f^*, \quad x \in [y] \setminus \hat{\mathcal{Z}}.$$

Proof: Let $x \leq c$. Then, we have

$$\begin{aligned} & \underline{fc} + \overline{\delta f c} \cdot (x - c) + \underline{\delta_2 f} \cdot (x - c)^2 \;>\; \widetilde{f} \\ \Leftrightarrow \quad & \Big(x - c + \overline{\delta f c} / \left(2\,\underline{\delta_2 f} \right) \Big)^2 \;>\; \left(\frac{\overline{\delta f c}}{2\,\underline{\delta_2 f}} \right)^2 + \frac{\left(\widetilde{f} - \underline{fc} \right)}{\underline{\delta_2 f}} \;=\; D_p\,. \end{aligned}$$

If $D_p < 0$, then by (20) we get

$$f\,(x) > \widetilde{f} \quad \text{for all } \; x \;\in\; \big[\, \underline{y}, c \,\big] \;=\; \big[\, \underline{y}, c \,\big] \setminus \emptyset \;=\; \big[\, \underline{y}, c \,\big] \setminus \big([p_1, p_2] \cap \big[\, \underline{y}, c \,\big] \big)\,.$$

If $D_p \geq 0$, then we have

$$\begin{aligned} & \underline{fc} + \overline{\delta f c} \cdot (x - c) + \underline{\delta_2 f} \cdot (x - c)^2 \;>\; \widetilde{f}, \quad x \in \big[\, \underline{y}, c \,\big] \\ \Leftrightarrow \quad & x \notin [p_1, p_2] \;\text{ and } x \in \big[\, \underline{y}, c \,\big] \\ \Leftrightarrow \quad & x \in \big[\, \underline{y}, c \,\big] \setminus \big([p_1, p_2] \cap \big[\, \underline{y}, c \,\big] \big)\,. \end{aligned}$$

Therefore, we have

$$f\,(x) > \widetilde{f} \qquad \text{for all } x \in \big[\, \underline{y}, c \,\big] \setminus \big([p_1, p_2] \cap \big[\, \underline{y}, c \,\big] \big) \tag{43}$$

both for $D_p < 0$ and $D_p \geq 0$.

By considering $x \in [\, c, \overline{y}\,]$ we get

$$f\,(x) > \widetilde{f} \qquad \text{for all } x \in [\, c, \overline{y}\,] \setminus ([q_1, q_2] \cap [\, c, \overline{y}\,]) \tag{44}$$

analogously. Combining (43) and (44) we obtain

$$f\,(x) > f^*, \quad x \in [y] \setminus \hat{\mathcal{Z}},$$

with

$$\hat{\mathcal{Z}} = \big(\, [p_1, p_2] \cap \big[\, \underline{y}, c \,\big] \,\big) \cup \big(\, [q_1, q_2] \cap [\, c, \overline{y}\,] \,\big).$$

□

Corollary 6.6 If the assumptions of Theorem 6.5 hold, then each $x^* \in [y]$ that is a global minimizer of f on $[x]$ is contained in

$$\left([p_1, p_2] \cap \left[\underline{y}, c\right]\right) \text{ or in } \left([q_1, q_2] \cap [c, \overline{y}]\right).$$

If

$$[p_1,p_2] \cap \left[\underline{y}, c\right] = \emptyset \tag{45}$$

and

$$[q_1, q_2] \cap [c, \overline{y}] = \emptyset, \tag{46}$$

then $[y]$ cannot contain a global minimizer of f on $[x]$. (45) and (46) hold, if $D_p < 0$ and $D_q < 0$.

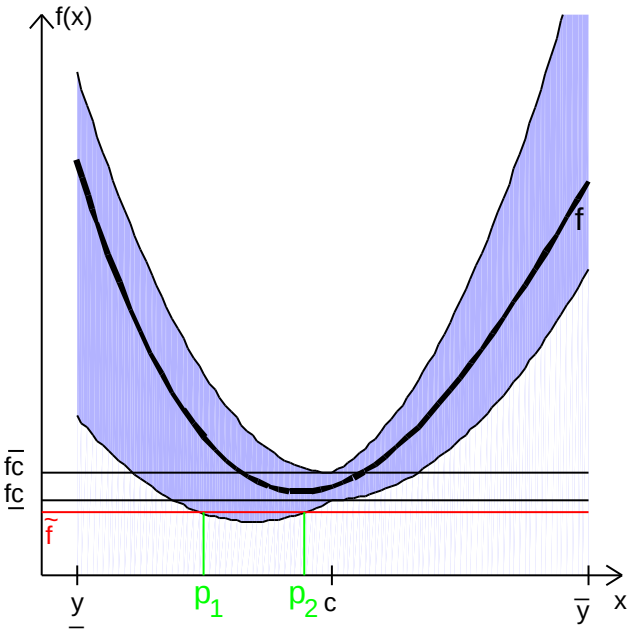

Figure 5: Illustration of Theorem 6.5

Figure 5 illustrates Theorem 6.5 for

$$D_p \geq 0 \quad \text{and} \quad D_q < 0.$$

In this case, the search for global minimizers $x^* \in \left[\underline{y}, \overline{y}\right]$ of f on $[x]$ can be restricted to $[p_1, p_2]$.

Figure 6 illustrates Theorem 6.5 for

$$D_p \geq 0,\ D_q \geq 0 \quad \text{and} \quad p_2 > c,\ q_1 < c.$$

In this case, the search for global minimizers $x^* \in \left[\underline{y}, \overline{y}\right]$ of f on $[x]$ can be restricted to $[p_1, q_2]$.

The other cases can be illustrated analogously.

Furthermore, by using (19) and the parabolas (20)-(23) we may update $\tilde{f}$ and compute a lower bound $\underline{fy}$ for the range of f on $[y]$. This is done by the following two theorems.

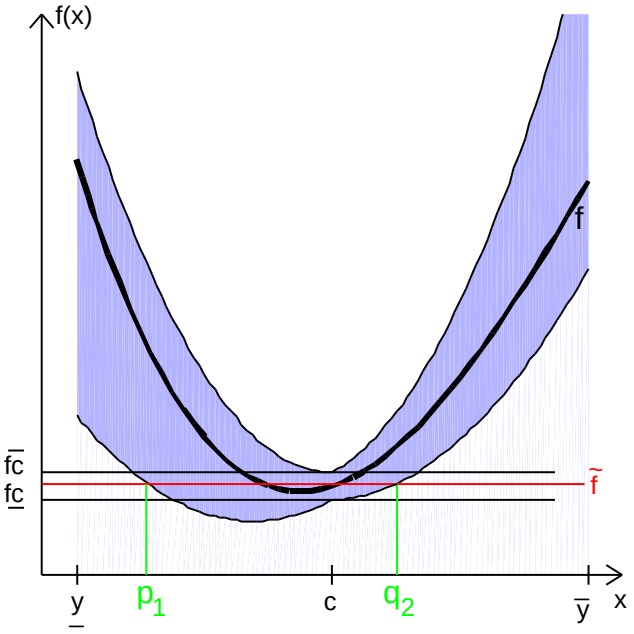

Figure 6: Illustration of Theorem 6.5

Theorem 6.7 Suppose that Assumption A holds. Moreover, we define

$$a_l := \overline{fc} + \underline{\delta fc} \cdot (\underline{y} - c) + \overline{\delta_2 f} \cdot (\underline{y} - c)^2,$$

$$a_r := \overline{fc} + \overline{\delta fc} \cdot (\overline{y} - c) + \overline{\delta_2 f} \cdot (\overline{y} - c)^2,$$

$$p_l := \begin{cases} \overline{fc} - \frac{1}{4} \left(\underline{\delta fc}\right)^2 / \overline{\delta_2 f}, & \text{if } \overline{\delta_2 f} > 0 \text{ and } c - \frac{1}{2} \underline{\delta fc} / \overline{\delta_2 f} \in [\underline{y}, c], \\ +\infty, & \text{otherwise,} \end{cases}$$

and

$$p_r := \begin{cases} \overline{fc} - \frac{1}{4} \left(\overline{\delta fc}\right)^2 / \overline{\delta_2 f}, & \text{if } \overline{\delta_2 f} > 0 \text{ and } c - \frac{1}{2} \overline{\delta fc} / \overline{\delta_2 f} \in [c, \overline{y}], \\ +\infty, & \text{otherwise.} \end{cases}$$

If $\overline{\delta_2 f} \leq 0$, then we have

$$f^* \leq \min \left\{ a_l, a_r, \overline{fc}, \widetilde{f} \right\}.$$

If $\overline{\delta_2 f} > 0$, then we have

$$f^* \leq \begin{cases} \min \left\{ p_l, a_l, \overline{fc}, \widetilde{f} \right\}, & \text{if } \underline{\delta fc} > 0, \\ \min \left\{ p_r, a_r, \overline{fc}, \widetilde{f} \right\}, & \text{if } \overline{\delta fc} < 0, \\ \min \left\{ \overline{fc}, \widetilde{f} \right\}, & \text{if } 0 \in [\underline{\delta fc}, \overline{\delta fc}]. \end{cases}$$

Proof: Because $f^* \leq f(x)$ holds for all $x \in [y]$, the claim follows by minimizing the right hand side of (21) and (23). □

Remark 6.8 Obviously, the upper bound of f^* in Theorem 6.7 is less than $\widetilde{f}$ or equal to $\widetilde{f}$. Therefore, in a global optimization algorithm we can update $\widetilde{f}$ by using Theorem 6.7.

Theorem 6.9 Let f be continuous on $[x] \subseteq D$, $c \in [y] = \left[\underline{y}, \overline{y}\right] \subseteq [x]$ and $f(c) \in \left[\underline{fc}, \overline{fc}\right]$. Furthermore, assume that $\left[\underline{\delta fc}, \overline{\delta fc}\right]$ and $\left[\underline{\delta_2 f}, \overline{\delta_2 f}\right]$ are intervals satisfying (19).

Moreover, we define

$$b_l := \underline{fc} + \overline{\delta fc} \cdot \left(\underline{y} - c\right) + \underline{\delta_2 f} \cdot \left(\underline{y} - c\right)^2,$$

$$b_r := \underline{fc} + \underline{\delta fc} \cdot \left(\overline{y} - c\right) + \underline{\delta_2 f} \cdot \left(\overline{y} - c\right)^2,$$

$$m_l := \begin{cases} \underline{fc} - \frac{1}{4}\left(\overline{\delta fc}\right)^2 / \underline{\delta_2 f}, & \text{if } \underline{\delta_2 f} > 0 \text{ and } c - \frac{1}{2}\,\overline{\delta fc} / \underline{\delta_2 f} \in \left[\underline{y}, c\right], \\ +\infty, & \text{otherwise,} \end{cases}$$

and

$$m_r := \begin{cases} \underline{fc} - \frac{1}{4}\left(\underline{\delta fc}\right)^2 / \underline{\delta_2 f}, & \text{if } \underline{\delta_2 f} > 0 \text{ and } c - \frac{1}{2}\,\underline{\delta fc} / \underline{\delta_2 f} \in \left[c, \overline{y}\right], \\ +\infty, & \text{otherwise.} \end{cases}$$

Then, for all $x \in [y]$ we have

$$f(x) \geq \begin{cases} \min\left\{b_l, b_r\right\}, & \text{if } \underline{\delta_2 f} \leq 0, \\ \min\left\{m_l, m_r, b_l, b_r\right\}, & \text{if } \underline{\delta_2 f} > 0. \end{cases}$$

Proof: The claim follows by minimizing the right hand side of (20) and (22). □

7 Algorithm

In this section, we state a branch-and-bound algorithm for global optimization of $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ on $[x] \subseteq D$. Let $\epsilon > 0$ be the parameter used for the accuracy condition (17). We initialize $\mathcal{L}$, $\mathcal{Q}$, and $\tilde{f}$ as described in section 5.

The lists are ordered in the following way: A pair $\left([y], \underline{fy}\right)$ is inserted into the list before all pairs $\left([z], \underline{fz}\right)$ with $\underline{fy} < \underline{fz}$ and after all pairs $\left([z], \underline{fz}\right)$ with $\underline{fy} \geq \underline{fz}$. Therefore, if $\underline{fy} > \tilde{f}$ holds for one pair $\left([y], \underline{fy}\right)$ of the list, then all subsequent pairs can be discarded.

While $\mathcal{L}$ is not empty, do the following steps:

1. Remove the first element $\left([y], \underline{fy}\right)$ of $\mathcal{L}$ and set $m := 1$.
2. Compute $t = (t_1, \ldots, t_n)$ with $t_i \in \{1, \ldots, n\}$ and $t_i \neq t_j$ for $i \neq j$ such that $\operatorname{diam}[y]_{t_k} \geq \operatorname{diam}[y]_{t_{k+1}}$ holds for $k = 1, \ldots, n-1$ (i.e. sorting by the diameter of the components $[y]_i$).
3. For $k = 1$ to n do steps 4 to 6.

4. Set $c = \operatorname{mid}[y]_{t_k}$. Compute a second-order slope tuple

$$\mathcal{F}_k = \left(f_{[y]}, \left[\underline{fc}, \overline{fc}\right], \left[\underline{\delta fc}, \overline{\delta fc}\right], \left[\underline{\delta f}, \overline{\delta f}\right], \left[\underline{\delta_2 f}, \overline{\delta_2 f}\right]\right)$$

of f on $[y]$ by componentwise computation with respect to component t_k. Use (11)-(13) to get an enclosure $\left[\underline{fy}, \overline{fy}\right]$ of the range of f on $[y]$ and use Theorem 6.9 for possibly increasing $\underline{fy}$. Update $\tilde{f}$ using Theorem 6.7. If $\underline{fy} \geq \tilde{f}$, then go to step 8, because $[y]$ cannot contain a global minimizer.

5. Carry out a first-order pruning step for $[y]_{t_k}$ (see [21, 23]). This gives the (possibly empty) intervals $\left[u^{(1)}\right]_{t_k} \subseteq \left[\underline{y}, c\right]$ and $\left[u^{(2)}\right]_{t_k} \subseteq \left[c, \overline{y}\right]$ with $x^*_{t_k} \in \left[u^{(1)}\right]_{t_k} \cup \left[u^{(2)}\right]_{t_k}$ for all global minimizers $x^* \in [y]$.

6. Use Theorems 6.1-6.5 for a second-order pruning step for $[y]_{t_k}$. Intersect the resulting intervals with the intervals $\left[u^{(1)}\right]_{t_k}$ and $\left[u^{(2)}\right]_{t_k}$ from step 5:
a) If all intersections are empty, then set $m := m - 1$ and go to step 8.
b) If there is exactly one intersection interval $\left[z^{(1)}\right]$, then set $[y]_{t_k} := \left[z^{(1)}\right]$.
c) If there are two intersection intervals $\left[z^{(1)}\right]$ and $\left[z^{(2)}\right]$, then set $\left[y^{(m)}\right] := [y]$, set the t_k-th component $\left[y^{(m)}\right]_{t_k} := \left[z^{(2)}\right]$ and set $m := m+1$. Finally, set $[y]_{t_k} := \left[z^{(1)}\right]$.

7. Set $\left[y^{(m)}\right] := [y]$.

8. In Step 6 at most one new interval vector $\left[y^{(i)}\right]$ is generated. Thus, in step 3-7 a total of m interval vectors $\left[y^{(i)}\right]$ is generated, where $0 \leq m \leq n+1$. By the properties of the pruning steps of first and second order, each global minimizer $x^* \in [y]$ is contained in a $\left[y^{(i)}\right]$, $i = 1, \ldots, m$. For all $\left[y^{(i)}\right]$, $i = 1, \ldots, m$, do steps 9-11.

9. Set $c = \operatorname{mid}\left[y^{(i)}\right]$ and compute an enclosure $\left[\underline{fy^{(i)}}, \overline{fy^{(i)}}\right]$ of the range of f on $\left[y^{(i)}\right]$ by intersecting (14)-(16). Generate the pair $\left(\left[y^{(i)}\right], \underline{fy^{(i)}}\right)$.

10. Use the enclosure $\left[\underline{fc}, \overline{fc}\right]$ of $f(c)$ obtained in step 9 for possibly updating $\tilde{f}$.

11. If $\tilde{f} < \underline{fy^{(i)}}$, then discard the pair $\left(\left[y^{(i)}\right], \underline{fy^{(i)}}\right)$. If

$$\max_{j=1,\ldots,n} \operatorname{diam}_{\mathrm{rel}} \left[y^{(i)}\right]_j \;\leq\; \epsilon$$

or if $\operatorname{diam}_{\mathrm{rel}}\left[\underline{fy^{(i)}}, \overline{fy^{(i)}}\right] \leq \epsilon$, then insert $\left(\left[y^{(i)}\right], \underline{fy^{(i)}}\right)$ into $\mathcal{Q}$, otherwise into $\mathcal{L}$.

12. Delete all pairs $\left([y], \underline{fy}\right)$ with $\underline{fy} > \tilde{f}$ from $\mathcal{L}$, because they do not contain a global minimizer of f.

After the termination of the algorithm, we have $f^* \in \left[\underline{fy}, \tilde{f}\right]$ for the first element $\left([y], \underline{fy}\right)$ of $\mathcal{Q}$. Furthermore, each global minimizer x^* of f on $[x]$ satisfies

$$x^* \;\in \bigcup_{([y],\underline{fy})\in\mathcal{Q}} [y].$$

For each $\left([y], \underline{fy}\right) \in \mathcal{Q}$ we have $\max_{j=1,\ldots,n} \operatorname{diam}_{\mathrm{rel}} [y]_j \leq \epsilon$ or $\operatorname{diam}_{\mathrm{rel}}\left[\underline{fy}, \tilde{f}\right] \leq \epsilon$. Note that the algorithm terminates on a floating-point computer, if the parameter $\epsilon > 0$ is greater than the machine accuracy.

Ratz				New Algorithm (Sect. 7)			
Ex.	STC 1	max LL	time [sec.]	Ex.	STC 2	max LL	time [sec.]
f1	2108	27	0.18	f1	1000	18	0.15
f2	925	19	0.06	f2	876	28	0.11
f3	14057	177	2.33	f3	20870	350	6.04
f4	4990	156	0.96	f4	12615	160	4.12
f5	135851	1557	57.39	f5	84183	797	32.65
f6	1748	49	0.34	f6	1276	50	0.40
f7	12437	105	2.54	f7	7588	99	2.37
f8	830	9	0.18	f8	1044	16	0.36
f9	5985	56	2.31	f9	2342	29	1.07
f10	433181	1047	509.59	f10	19226	87	16.11
f11	314	10	0.03	f11	252	8	0.05
f12	5764	66	1.24	f12	5284	56	1.82
f13	367455	11437	525.93	f13	348989	11083	520.86
f14	11808	271	1.39	f14	1204	15	0.20
f15	4116	75	0.32	f15	1778	37	0.31
f16	88398	1072	27.10	f16	36081	556	11.45
f17	306	6	0.04	f17	202	4	0.05
f18	2948	81	0.29	f18	2304	58	0.43
f19	5935	20	0.35	f19	6610	21	0.70
f20	25337	137	8.35	f20	3997	134	1.57
f21	436	15	0.06	f21	348	12	0.08
f22	8761	240	1.69	f22	5450	212	1.57
f23	238268	6559	184.00	f23	115487	3766	72.37
f24	2524	71	0.29	f24	1894	60	0.45
f25	24595	537	3.65	f25	17481	445	4.67

Table 1: Comparison of the new algorithm with the algorithm of Ratz

8 Examples

We compare the algorithm from section 7 with Ratz' program [21, 23]. For this purpose, we consider 25 test functions. The test functions are listed in the appendix together with the search interval $[x]$ and the parameter ϵ. Most of them can be found in [3], [18], [21], [32] and [36].

The following tables compare the algorithm of Ratz [21, 23] with the new algorithm with respect to the number of slope tuple computations of first (STC 1) and second order (STC 2), the maximal length of the list $\mathcal{L}$ (max LL), and the computation time in seconds. In [21, 23], the algorithm of Ratz was implemented in Pascal-XSC [6, 13], so we also implemented the new algorithm in this programming language. The computations were carried out on a PC with 2 Athlon MP 1800+ processors, 1 GB main memory and the operating system Suse Linux 9.3. The source code is freely available [26]. A current Pascal-XSC compiler is provided by the working group "Scientific Computing / Software Engineering" of the University of Wuppertal [39].

Table 1 shows that in most of the examples the new algorithm requires fewer slope tuple computations, and the maximal length of the working list $\mathcal{L}$ is less than in the algorithm

of Ratz. Because the computation of a second-order slope tuple is more costly than the computation of a first-order slope tuple, neither algorithm is generally better with respect to computation time. In some of the examples the new algorithm is faster, whereas in some of the examples it is slower than the algorithm of Ratz.

For some test functions, e.g. $f3$, $f4$, and $f10$, the computation times differ substantially. This can be explained as follows: Let $f_{[y]}$ be an enclosure of the range of f on some interval $[y] \in \mathbb{IR}^n$. If $0 \in f_{[y]}$, then we have

$$\mathrm{diam}_{\mathrm{rel}} f_{[y]} = \mathrm{diam} f_{[y]},$$

i.e. the relative diameter of $f_{[y]}$ is equal to its absolute diameter. Hence, depending on the current interval $[y]$, many subdivisions of $[y]$ may be needed until the accuracy condition (17) is satisfied. In fact, for the test problems $f3$, $f4$, and $f10$ the global minimum is $f^* = 0$, so that this problem arises.

The extent to which this effect results in higher computation times depends strongly on the search interval $[x]$. Function $f3$, i.e. the generalized function of Rosenbrock of dimension 5, illustrates this dependency. The results can be found in Table 2.

We observe that a slight variation of $[x]$ significantly changes the number of slope tuples that need to be computed and the computation time. In each case the unique global minimizer is $x^* = (1, \ldots, 1)^T$ with $f^* = 0$.

Finally, we consider the examples $f1$-$f25$ once again. We reduce the effect described above by increasing the function value by 1, i.e. we set $\widetilde{f}(x) := f(x) + 1$. The results are listed in Table 3. We note that for some functions, e.g. $\tilde{f}3, \tilde{f}4$, $\tilde{f}8$-$\tilde{f}10$, and $\tilde{f}14$, having $f^* = 0$ as the global minimum, the number of computed slope tuples and the computation time decreased significantly. For other functions the results are almost unchanged compared to f.

A similar effect can occur if $x^* = (0, \ldots, 0)$ is the global minimizer: Suppose that during the course of the algorithm we obtain an interval $[y]$ with $[y]_i = [a_i, b_i]$, $i = 1, \ldots, n$, for some small $a_i \leq 0$, $b_i > 0$. Suppose furthermore that in the next step of the algorithm we obtain two subintervals $\left[y^{(1)}\right]$ and $\left[y^{(2)}\right]$ such that $0 \in \left[y^{(1)}\right]_1$ and $0 \notin \left[y^{(2)}\right]_1$. Then, $\left[y^{(2)}\right]$ does not contain the global minimizer $x^* = (0, \ldots, 0)$. However, it may not be possible to discard $\left[y^{(2)}\right]$ because it is likely to be very close to x^*. Furthermore, the relative diameter of $\left[y^{(2)}\right]_1$ may be very large so that the first relation in (17) can only be satisfied for very small subintervals $[z] \subseteq \left[y^{(2)}\right]$. This effect can be observed for $\widetilde{f}19$.

In summary, in most of the examples, the new algorithm requires fewer slope tuple computations, and the maximal length of the working list $\mathcal{L}$ is less than in the algorithm of Ratz. Neither algorithm is generally better with respect to computation time. Nevertheless, for some of the examples the new algorithm is significantly faster than the algorithm of Ratz.

9 Conclusion

In this paper, we have introduced a second-order pruning step for verified global optimization on a floating-point computer. Using automatic computation of a second-order slope tuple, we added this second-order pruning step to an algorithm by Ratz. Furthermore, we compared our new algorithm with the algorithm of Ratz by considering some test problems. In most of the test problems, the new algorithm requires fewer slope tuple computations

1. $[x] \in \mathbb{IR}^5,\ [x]_i = [-5.12, 5.12]\,,\ i = 1, \ldots, 5,$
2. $[x] \in \mathbb{IR}^5,\ [x]_i = [-6, 6]\,,\ i = 1, \ldots, 5,$
3. $[x] \in \mathbb{IR}^5,\ [x]_i = [-2, 5, 2, 5]\,,\ i = 1, \ldots, 5,$
4. $[x] \in \mathbb{IR}^5,\ [x]_i = [-3, 4]\,,\ i = 1, \ldots, 5,$
5. $[x] \in \mathbb{IR}^5,\ [x]_i = [-1, 2]\,,\ i = 1, \ldots, 5,$
6. $[x] = \big([-2, 2]\,, [-1.5, 2.5]\,, [-3, 3]\,, [0, 3]\,, [-1, 1.5]\big)^T,$
7. $[x] = \big([-3, 6]\,, [-6, 2]\,, [-4, 3]\,, [-5, 3]\,, [-2, 6]\big)^T,$
8. $[x] = \big([-3, 6]\,, [-6, 2]\,, [-4, 3]\,, [-5, 1]\,, [-2, 6]\big)^T,$
9. $[x] = \big([-1.5, 3.2]\,, [0.12, 1.5]\,, [-2.1, 2.7]\,, [-2, 2]\,, [-1, 5.12]\big)^T,$
10. $[x] = \big([-1.5, 3.2]\,, [0.12, 1.5]\,, [-2.1, 2.7]\,, [-2, 2]\,, [-1.5, 5.12]\big)^T.$

Ratz				New Algorithm (Sect. 7)			
No.	STC 1	max LL	time [sec.]	No.	STC 2	max LL	time [sec.]
1	14057	177	2.33	1	20870	350	6.05
2	15045	144	2.48	2	12110	160	3.09
3	20603	258	3.70	3	16420	206	4.47
4	24459	290	4.36	4	35879	576	11.52
5	21376	412	3.90	5	17889	250	4.80
6	23761	285	3.99	6	24244	196	6.40
7	13116	152	1.86	7	11025	177	2.58
8	22974	351	4.10	8	7296	128	1.74
9	31433	509	5.89	9	74315	1561	29.75
10	253419	6869	183.80	10	21954	247	5.89

Table 2: The Rosenbrock function of dimension 5 for different search intervals $[x]$ and $\epsilon = 10^{-10}$.

Ratz				New Algorithm (Sect. 7)			
No.	STC 1	max LL	time [sec.]	No.	STC 2	max LL	time [sec.]
$\widetilde{\text{f1}}$	2004	27	0.17	$\widetilde{\text{f1}}$	938	18	0.15
$\widetilde{\text{f2}}$	642	19	0.04	$\widetilde{\text{f2}}$	668	28	0.08
$\widetilde{\text{f3}}$	11190	107	1.74	$\widetilde{\text{f3}}$	11296	137	2.77
$\widetilde{\text{f4}}$	6549	156	1.26	$\widetilde{\text{f4}}$	4047	160	1.20
$\widetilde{\text{f5}}$	135905	1557	59.27	$\widetilde{\text{f5}}$	84199	797	32.97
$\widetilde{\text{f6}}$	1748	49	0.34	$\widetilde{\text{f6}}$	1276	50	0.41
$\widetilde{\text{f7}}$	12437	105	2.54	$\widetilde{\text{f7}}$	7588	99	2.39
$\widetilde{\text{f8}}$	429	9	0.08	$\widetilde{\text{f8}}$	381	8	0,11
$\widetilde{\text{f9}}$	1480	21	0.43	$\widetilde{\text{f9}}$	1087	21	0.42
$\widetilde{\text{f10}}$	16932	89	9.67	$\widetilde{\text{f10}}$	11060	87	8.15
$\widetilde{\text{f11}}$	224	10	0.02	$\widetilde{\text{f11}}$	166	8	0.03
$\widetilde{\text{f12}}$	4030	66	0.80	$\widetilde{\text{f12}}$	5192	56	1.73
$\widetilde{\text{f13}}$	367447	11437	534.96	$\widetilde{\text{f13}}$	348967	11083	514.43
$\widetilde{\text{f14}}$	1256	19	0.10	$\widetilde{\text{f14}}$	838	15	0.13
$\widetilde{\text{f15}}$	3890	75	0.31	$\widetilde{\text{f15}}$	1670	37	0.29
$\widetilde{\text{f16}}$	88598	1072	28.14	$\widetilde{\text{f16}}$	36309	560	11.67
$\widetilde{\text{f17}}$	306	6	0.04	$\widetilde{\text{f17}}$	202	4	0.05
$\widetilde{\text{f18}}$	2948	81	0.30	$\widetilde{\text{f18}}$	2304	58	0.44
$\widetilde{\text{f19}}$	5930	20	0.37	$\widetilde{\text{f19}}$	55379	2060	28.12
$\widetilde{\text{f20}}$	25337	137	8.33	$\widetilde{\text{f20}}$	3997	134	1.59
$\widetilde{\text{f21}}$	436	15	0.06	$\widetilde{\text{f21}}$	348	12	0.08
$\widetilde{\text{f22}}$	8781	240	1.72	$\widetilde{\text{f22}}$	5450	212	1.59
$\widetilde{\text{f23}}$	238616	6559	188.290	$\widetilde{\text{f23}}$	115592	3766	72.45
$\widetilde{\text{f24}}$	2548	71	0.30	$\widetilde{\text{f24}}$	1906	60	0.46
$\widetilde{\text{f25}}$	24733	537	3.76	$\widetilde{\text{f25}}$	17535	445	4.76

Table 3: Comparison of the two algorithms $\widetilde{f}(x) := f(x) + 1$

than the algorithm of Ratz. Neither algorithm is generally better with respect to computation time, because the computation of a second-order slope tuple is more costly than the computation of a first-order slope tuple. The source code of the programs is freely available [26].

Acknowledgements

The author gratefully acknowledges the supervision of his dissertation by Prof. Dr. G. Alefeld. Furthermore, the author would like to thank Dr. J. Mayer for reading the paper and polishing the English.

Appendix

We consider the following 25 test problems:

1. Function of Branin : $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = ([-5, 10], [0, 15])^T$, $\epsilon = 10^{-12}$,
$$f(x) = \left(\frac{5}{\pi}x_1 - \frac{5.1}{4\pi^2}x_1^2 + x_2 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10.$$

2. Function of Rosenbrock: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-10, 50]^2$, $\epsilon = 10^{-12}$,
$$f(x) = 100\left(x_2 - x_1^2\right)^2 + (x_1 - 1)^2.$$

3. Generalized function of Rosenbrock of dimension 5: $f : \mathbb{R}^5 \to \mathbb{R}$, $[x] = [-5.12, 5.12]^5$, $\epsilon = 10^{-10}$,
$$f(x) = \sum_{i=1}^{4}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right).$$

4. Function G7 of Griewank: $f : \mathbb{R}^7 \to \mathbb{R}$, $[x] = [-50, 60]^7$, $\epsilon = 10^{-3}$,
$$f(x) = \sum_{i=1}^{7}\frac{x_i^2}{4000} - \prod_{i=1}^{7}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1.$$

5. Function L3 of Levy: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-10, 50]^2$, $\epsilon = 10^{-12}$,
$$f(x) = \sum_{i=1}^{5} i\cos\big((i-1)x_1 + i\big) \cdot \sum_{j=1}^{5} j\cos\big((j+1)x_2 + j\big).$$

6. Function L5 of Levy: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-10, 50]^2$, $\epsilon = 10^{-12}$,
$$\begin{aligned} f(x) \;=\; & \sum_{i=1}^{5} i\cos\big((i-1)x_1 + i\big) \cdot \sum_{j=1}^{5} j\cos\big((j+1)x_2 + j\big) \\ & + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2. \end{aligned}$$

7. A variant of function L5 of Levy: $f:\mathbb{R}^3\to\mathbb{R}$, $[x]=[-10,50]^3$, $\epsilon=10^{-12}$,

$$\begin{aligned} f(x) &= \sum_{i=1}^{5} i\cos\big((i-1)x_1+i\big)\cdot\sum_{j=1}^{5} j\cos\big((j+1)x_2+j\big) \\ &\quad +(x_1+1.42513)^2+(x_2+0.80032)^2+(x_3-1)^2\,. \end{aligned}$$

8. Function L8 of Levy: $f:\mathbb{R}^3\to\mathbb{R}$, $[x]=[-10,50]^3$, $\epsilon=10^{-12}$,

$$\left.\begin{aligned} f(x) &= \sum_{i=1}^{n-1}(y_i-1)^2\left(1+10\sin^2(\pi y_{i+1})\right)+\sin^2(\pi y_1)+(y_n-1)^2 \\ &\text{with } n=3 \text{ and } y_i=1+(x_i-1)/4, \qquad i=1,\ldots,n. \end{aligned}\right\} \tag{47}$$

9. Function L10 of Levy: $f:\mathbb{R}^5\to\mathbb{R}$, $[x]=[-10,50]^5$, $\epsilon=10^{-12}$ and (47) with $n=5$.

10. Function L12 of Levy: $f:\mathbb{R}^{10}\to\mathbb{R}$, $[x]=[-10,50]^{10}$, $\epsilon=10^{-8}$ and (47) with $n=10$.

11. Function L13 of Levy: $f:\mathbb{R}^2\to\mathbb{R}$, $[x]=[-10,50]^2$, $\epsilon=10^{-12}$,

$$\left.\begin{aligned} f(x) &= \sum_{i=1}^{n-1}(x_i-1)^2\left(1+\sin^2(3\pi x_{i+1})\right) \\ &\quad +(x_n-1)^2\left(1+\sin^2(2\pi x_n)\right)+\sin^2(3\pi x_1) \end{aligned}\right\} \tag{48}$$

with $n=2$.

12. Function L18 of Levy: $f:\mathbb{R}^7\to\mathbb{R}$, $[x]=[-10,50]^7$, $\epsilon=10^{-8}$ and (48) with $n=7$.

13. Function of Goldstein and Price: $f:\mathbb{R}^2\to\mathbb{R}$, $[x]=[-10,50]^2$, $\epsilon=10^{-12}$,

$$\begin{aligned} f(x) &= \Big(1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\Big) \\ &\quad \cdot\Big(30+(2x_1-3x_2)^2\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\Big)\,. \end{aligned}$$

14. Function SC32 of Schwefel: $f:\mathbb{R}^3\to\mathbb{R}$, $[x]=[-1.89,1.89]^3$, $\epsilon=10^{-12}$,

$$f(x) = \sum_{i=2}^{3}\Big(\left(x_1-x_i^2\right)^2+(x_i-1)^2\Big)\,.$$

15. Function R4 of Ratz: $f:\mathbb{R}^2\to\mathbb{R}$, $[x]=[-3,3]^2$, $\epsilon=10^{-12}$,

$$f(x) = \sin\left(x_1^2+2x_2^2\right)\cdot\exp\left(-x_1^2-x_2^2\right).$$

16. A variant of Shubert's test function from [21, Sect. 5.7.1]:
$f:\mathbb{R}^2\to\mathbb{R}$, $[x]=[-10,50]^2$, $\epsilon=10^{-12}$,

$$f(x) = \sum_{i=1}^{5} i\,\sin\big((i+1)x_1+i\big)\,\cos x_2.$$

17. Example 6.18 from [21]: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-10, 50]^2$, $\epsilon = 10^{-12}$,

$$\begin{aligned} f(x) &= |y_1 - 1| \left(1 + 10 \left|\sin(\pi y_2)\right|\right) + \left|\sin(\pi y_1)\right| + |y_2 - 1| \\ &\quad \text{with } y_i = 1 + (x_i - 1)/4, \qquad i = 1, 2. \end{aligned}$$

18. Example 6.19 from [21]: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = \left([-100, 100], [0.02, 100]\right)^T$, $\epsilon = 10^{-12}$,

$$f(x) = 10 \, |x_1 - 1| \left|\sin\left(\frac{1}{x_2}\right)\right| + (x_2 + 2) \cdot |x_1 - 1 + 2x_2| .$$

19. Example 6.20 from [21]: $f : \mathbb{R}^4 \to \mathbb{R}$, $[x] = [-4, 4]^4$, $\epsilon = 10^{-12}$,

$$f(x) = |x_1 + 10x_2| + 5 \, |x_3 - x_4| + |x_2 - 2x_3| + 10 \, |x_1 - x_4| .$$

20. Example 6.22 from [21]: $f : \mathbb{R}^9 \to \mathbb{R}$, $[x] = [-10, 50]^9$, $\epsilon = 10^{-12}$,

$$\begin{aligned} f(x) &= \sum_{i=1}^{8} |x_i - 1| \left(1 + \left|\sin(3\pi x_{i+1})\right|\right) \\ &\quad + |x_9 - 1| \left(1 + \left|\sin(2\pi x_9)\right|\right) + \left|\sin(3\pi x_1)\right| + 1. \end{aligned}$$

21. Example 6.26 from [21]: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [0, 10]^2$, $\epsilon = 10^{-12}$,

$$\begin{aligned} f(x) &= \min \Big\{ \, |\cos(2x_1)| + |\cos(2x_2)| - 3 \sin\left(\frac{\pi x_1}{10}\right) - 2 \sin\left(\frac{\pi x_2}{10}\right), \\ &\qquad 50 \, |x_1 - 1| + 50 \, |x_2 - 1| - 5 \Big\}. \end{aligned}$$

22. Function of Henriksen, Madsen, Dim2: $f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-10, 10]^2$, $\epsilon = 10^{-6}$,

$$f(x) = -\sum_{i=1}^{2} \sum_{j=1}^{5} j \, \sin\big((j+1) \, x_i + j\big).$$

23. Function of Henriksen, Madsen, Dim3: $f : \mathbb{R}^3 \to \mathbb{R}$, $[x] = [-10, 10]^3$, $\epsilon = 10^{-6}$,

$$f(x) = -\sum_{i=1}^{3} \sum_{j=1}^{5} j \, \sin\big((j+1) \, x_i + j\big).$$

24. Function from the SIAM 10x10-Digit-Challenge (see [3, p. 77]):
$f : \mathbb{R}^2 \to \mathbb{R}$, $[x] = [-1, 1]^2$, $\epsilon = 10^{-12}$,

$$\begin{aligned} f(x) &= \exp(\sin(50x_1)) + \sin(60 \exp x_2) + \sin(70 \sin x_1) \\ &\quad + \sin(\sin(80x_2)) - \sin(10 \, (x_1 + x_2)) + \left(x_1^2 + x_2^2\right)/4. \end{aligned}$$

25. A variant of example 24 (see [3, p. 99]): $f : \mathbb{R}^3 \to \mathbb{R}$, $[x] = [-1, 1]^3$, $\epsilon = 10^{-12}$,

$$\begin{aligned} f(x) &= \exp(\sin(50x_1)) + \sin(60 \exp x_2) \sin(60x_3) + \sin(70 \sin x_1) \cos(10x_3) \\ &\quad + \sin(\sin(80x_2)) - \sin(10 \, (x_1 + x_3)) + \left(x_1^2 + x_2^2 + x_3^2\right)/4. \end{aligned}$$

References

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

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

[3] F. Bornemann, D. Laurie, S. Wagon, and J. Waldvogel. *The SIAM 100-Digit Challenge: A Study in High-Accuracy Numerical Computing.* Society of Industrial and Applied Mathematics (SIAM), Philadelphia, 2004.

[4] F. H. Clarke. *Optimization and Nonsmooth Analysis.* John Wiley & Sons, New York, 1983.

[5] A. Frommer, B. Lang, and M. Schnurr. A comparison of the Moore and Miranda existence tests. *Computing*, 72:349–354, 2004.

[6] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz. *Numerical Toolbox for Verified Computing I.* Springer-Verlag, Berlin, 1993.

[7] E. R. Hansen. Interval forms of Newton's method. *Computing*, 20:153–163, 1978.

[8] E. R. Hansen. Global optimization using interval analysis – the one-dimensional case. *J. Optim. Theory Appl.*, 29:331–344, 1979.

[9] E. R. Hansen. Global optimization using interval analysis – the multi-dimensional case. *Numer. Math.*, 34:247–270, 1980.

[10] E. R. Hansen and G. W. Walster. *Global Optimization Using Interval Analysis: Second Edition, Revised and Expanded.* Marcel Dekker, New York, 2004.

[11] K. Ichida and Y. Fujii. An interval arithmetic method for global optimization. *Computing*, 23:85–97, 1979.

[12] R. B. Kearfott. *Rigorous Global Search: Continuous Problems.* Kluwer Academic Publishers, Dordrecht, 1996.

[13] R. Klatte, U. Kulisch, M. Neaga, D. Ratz, and Ch. Ullrich. *Pascal-XSC – Language Reference with Examples.* Springer, Berlin, 1992.

[14] L. Kolev. Use of interval slopes for the irrational part of factorable functions. *Reliab. Comput.*, 3:83–93, 1997.

[15] R. Krawczyk and A. Neumaier. Interval slopes for rational functions and associated centered forms. *SIAM J. Numer. Anal.*, 22:604–616, 1985.

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

[17] R. E. Moore. A test for existence of solutions to nonlinear systems. *SIAM J. Numer. Anal.*, 14(4):611–615, 1977.

[18] J. J. Moré, B. S. Garbow, and K. E. Hilstrom. Testing unconstrained optimization software. *ACM Trans. Math. Software*, 7:17–41, 1981.

[19] H. Muñoz and R. B. Kearfott. Slope intervals, generalized gradients, semigradients, slant derivatives, and csets. *Reliab. Comput.*, 10(3):163–193, 2004.

[20] L. B. Rall. *Automatic Differentiation: Techniques and Applications, Lecture Notes in Computer Science, Vol. 120.* Springer, Berlin, 1981.

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

[22] D. Ratz. A nonsmooth global optimization technique using slopes – the one-dimensional case. *J. Global Optim.*, 14:365–393, 1999.

[23] D. Ratz. A nonsmooth global optimization technique using slopes – the one-dimensional case. *J. Global Optim.*, 14:365–393, 1999.

[24] S. M. Rump. Expansion and estimation of the range of nonlinear functions. *Math. Comp.*, 65(216):1503–1512, 1996.

[25] U. Schäfer and M. Schnurr. A comparison of simple tests for accuracy of approximate solutions to nonlinear systems with uncertain data. *J. Ind. Manag. Optim.*, 2(4):425–434, 2006.

[26] M. Schnurr. Webpage for software download. `http://iamlasun8.mathematik.uni-karlsruhe.de/~ae26/software/`.

[27] M. Schnurr. Some supplements concerning automatic slope enclosures. *PAMM*, 6(1):691–692, 2006.

[28] M. Schnurr. *The Automatic Computation of Second-Order Slope Tuples for Some Nonsmooth Functions.* Preprint Nr. 07/09, Institut für Wissenschaftliches Rechnen und Mathematische Modellbildung, Universität Karlsruhe, Germany, 2007.

[29] M. Schnurr. *Computing Slope Enclosures by Exploiting a Unique Point of Inflection.* Preprint Nr. 07/08, Institut für Wissenschaftliches Rechnen und Mathematische Modellbildung, Universität Karlsruhe, Germany, 2007.

[30] M. Schnurr. *Steigungen höherer Ordnung zur verifizierten globalen Optimierung.* PhD thesis, Universität Karlsruhe, 2007. `http://digbib.ubka.uni-karlsruhe.de/volltexte/1000007229`.

[31] M. Schnurr and D. Ratz. Slope enclosures for functions given by two or more branches. Submitted for Publication.

[32] H. Schwefel. *Numerical Optimization of Computer Models.* Wiley, New York, 1981.

[33] Z. Shen and M. A. Wolfe. On interval enclosures using slope arithmetic. *Appl. Math. Comput.*, 39:89–105, 1990.

[34] S. Skelboe. Computation of rational interval functions. *BIT*, 4:87–95, 1974.

[35] D. G. Sotiropoulos and T. N. Grapsa. A branch-and-prune method for global optimization: The univariate case. In W. Krämer and J. W. v. Gudenberg, editors, *Scientific Computing, Validated Numerics, Interval Methods*, pages 215–226. Kluwer, Boston, 2001.

[36] A. Törn and A. Žilinskas. *Global Optimization, Lecture Notes in Computer Science, Vol. 350.* Springer, Berlin, 1989.

[37] T. Vinko, J.-L. Lagouanelle, and T. Csendes. A new inclusion function for global optimization: Kite – the one dimensional case. *J. Global Optim.*, 30:435–456, 2004.

[38] T. Vinko and D. Ratz. A multidimensional branch-and-prune method for interval global optimization. *Numer. Algorithms*, 37:391–399, 2004.

[39] XSC Website. Website on programming languages for scientific computing with validation.
`http://www.xsc.de` [December 2007].

IWRMM-Preprints seit 2006

Nr. 06/01 Willy Dörfler, Vincent Heuveline: Convergence of an adaptive *hp* finite element strategy in one dimension

Nr. 06/02 Vincent Heuveline, Hoang Nam-Dung: On two Numerical Approaches for the Boundary Control Stabilization of Semi-linear Parabolic Systems: A Comparison

Nr. 06/03 Andreas Rieder, Armin Lechleiter: Newton Regularizations for Impedance Tomography: A Numerical Study

Nr. 06/04 Götz Alefeld, Xiaojun Chen: A Regularized Projection Method for Complementarity Problems with Non-Lipschitzian Functions

Nr. 06/05 Ulrich Kulisch: Letters to the IEEE Computer Arithmetic Standards Revision Group

Nr. 06/06 Frank Strauss, Vincent Heuveline, Ben Schweizer: Existence and approximation results for shape optimization problems in rotordynamics

Nr. 06/07 Kai Sandfort, Joachim Ohser: Labeling of n-dimensional images with choosable adjacency of the pixels

Nr. 06/08 Jan Mayer: Symmetric Permutations for I-matrices to Delay and Avoid Small Pivots During Factorization

Nr. 06/09 Andreas Rieder, Arne Schneck: Optimality of the fully discrete filtered Backprojection Algorithm for Tomographic Inversion

Nr. 06/10 Patrizio Neff, Krzysztof Chelminski, Wolfgang Müller, Christian Wieners: A numerical solution method for an infinitesimal elasto-plastic Cosserat model

Nr. 06/11 Christian Wieners: Nonlinear solution methods for infinitesimal perfect plasticity

Nr. 07/01 Armin Lechleiter, Andreas Rieder: A Convergenze Analysis of the Newton-Type Regularization CG-Reginn with Application to Impedance Tomography

Nr. 07/02 Jan Lellmann, Jonathan Balzer, Andreas Rieder, Jürgen Beyerer: Shape from Specular Reflection Optical Flow

Nr. 07/03 Vincent Heuveline, Jan-Philipp Weiß: A Parallel Implementation of a Lattice Boltzmann Method on the Clearspeed Advance Accelerator Board

Nr. 07/04 Martin Sauter, Christian Wieners: Robust estimates for the approximation of the dynamic consolidation problem

Nr. 07/05 Jan Mayer: A Numerical Evaluation of Preprocessing and ILU-type Preconditioners for the Solution of Unsymmetric Sparse Linear Systems Using Iterative Methods

Nr. 07/06 Vincent Heuveline, Frank Strauss: Shape optimization towards stability in constrained hydrodynamic systems

Nr. 07/07 Götz Alefeld, Günter Mayer: New criteria for the feasibility of the Cholesky method with interval data

Nr. 07/08 Marco Schnurr: Computing Slope Enclosures by Exploiting a Unique Point of Inflection

Nr. 07/09 Marco Schnurr: The Automatic Computation of Second-Order Slope Tuples for Some Nonsmooth Functions

Nr. 07/10 Marco Schnurr: A Second-Order Pruning Step for Verified Global Optimization

Eine aktuelle Liste aller IWRMM-Preprints finden Sie auf:

www.mathematik.uni-karlsruhe.de/iwrmm/seite/preprints