Journal of Global Optimization **7**: 183–207, 1995.

On the Selection of Subdivision Directions in Interval Branch-and-Bound Methods for Global Optimization*

DIETMAR RATZ dietmar.ratz@math.uni-karlsruhe.de
Institut für Angewandte Mathematik, Universität Karlsruhe, D-76128 Karlsruhe, Germany

and

TIBOR CSENDES csendes@inf.u-szeged.hu
Departm. of Applied Informatics, József Attila University, H-6720 Szeged, Árpád tér 2, Hungary

(Received: 31 January 1995; accepted 28 April 1995)

Abstract. This paper investigates the influence of the interval subdivision selection rule on the convergence of interval branch-and-bound algorithms for global optimization. For the class of rules that allows convergence, we study the effects of the rules on a model algorithm with special list ordering. Four different rules are investigated in theory and in practice. A wide spectrum of test problems is used for numerical tests indicating that there are substantial differences between the rules with respect to the required CPU time, the number of function and derivative evaluations, and the necessary storage space. Two rules can provide considerable improvements in efficiency for our model algorithm.

Keywords: Global optimization, interval arithmetic, branch-and-bound, interval subdivision

1. Introduction

The investigated class of interval branch-and-bound methods for global optimization [7], [8], [19] addresses the problem of finding guaranteed and reliable solutions of global optimization problems

$$\min_{x \in X} f(x), \tag{1}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and $X \subseteq \mathbb{R}^n$ is an n-dimensional interval vector. We do not require a special problem structure, but we assume inclusion functions of the objective function and its gradient to be available [1]. These inclusion functions are utilized to compute bounds for f on an interval vector (and therefore on a continuum of points, including those points that are not finitely representable). So valleys, no matter how narrow, are enclosed with certainty.

* The work has been supported by the Grants OTKA 2879/1991, and MKM 414/1994.

The basic idea of such interval branch-and-bound algorithms is to apply several interval techniques to reject large regions in which the optimum can be guaranteed not to lie. For this reason, the original interval vector X gets subdivided, and subregions which cannot contain a global minimizer of f are discarded, while the other subregions get subdivided again until the desired accuracy (width) of the interval vectors is achieved. In this context, our special interest lies in the choice of the direction for the interval subdivision steps, and the present paper investigates the possible improvements of this choice for interval branch-and-bound methods for global optimization. The generality of the problem class and the modest requirement of the existence of the inclusion functions stress the importance of any improvement in efficiency.

The global minimum value of f on X is denoted by f^*, and the set of global minimizer points of f on X by X^*. That is,

$$f^* = \min_{x \in X} f(x) \qquad \text{and} \qquad X^* = \{x^* \mid f(x^*) = f^*\}.$$

We denote real numbers by $x, y, \ldots$ and real bounded and closed interval vectors by $X = [\underline{X}, \overline{X}], Y = [\underline{Y}, \overline{Y}], \ldots$, where $\min X = \underline{X}$, $\max X = \overline{X}$, $\min Y = \underline{Y}$, $\max Y = \overline{Y}$, etc.

The set of compact intervals is denoted by $\mathbb{I} := \{[a, b] \mid a \leq b,\ a, b \in \mathbb{R}\}$ and the set of n-dimensional interval vectors (also called boxes) by $\mathbb{I}^n$. For real vectors and interval vectors the notations

$$x = (x_i), \quad x_i \in \mathbb{R}, \qquad \text{and} \qquad X = (X_i), \quad X_i \in \mathbb{I}$$

are used.

The width of the interval X is defined by $w(X) = \max X - \min X$ if $X \in \mathbb{I}$, and $w(X) = \max_{i=1}^n w(X_i)$, if $X \in \mathbb{I}^n$. The midpoint of the interval X is defined by $m(X) = (\min X + \max X)/2$ if $X \in \mathbb{I}$, and $m(X) = (m(X_i))$, if $X \in \mathbb{I}^n$.

We call a function $F : \mathbb{I}^n \to \mathbb{I}$ an *inclusion function* of $f : \mathbb{R}^n \to \mathbb{R}$ in X, if $x \in X$ implies $f(x) \in F(X)$. In other words, $f_{\mathrm{rg}}(X) \subseteq F(X)$, where $f_{\mathrm{rg}}(X)$ is the range of the function f on X. The inclusion function of the gradient of f is denoted by $\boldsymbol{\nabla} F$.

There are several ways to build an inclusion function for a given optimization problem (e.g. by using the Lipschitz constant). Interval arithmetic [1], [7], [8], [19] is a convenient tool for constructing inclusion functions. This can be done for almost all functions specified by a finite algorithm (i.e. not only for given expressions). Moreover, applying so-called automatic differentiation or differentiation arithmetic in connection with interval arithmetic [7], we are also able to compute the inclusion function for the gradient.

Automatic differentiation combines the advantages of symbolic and numerical differentiation and handles numbers instead of symbolic formulas. The computation of the gradient is done automatically together with the computation of the function value. The main advantage of this process is that only the algorithm or formula for the function is required. No explicit formulas for the gradient are required.

It is assumed in the following that the inclusion functions have the *isotonicity* property, i.e. $X \subseteq Y$ implies $F(X) \subseteq F(Y)$, and that

$$w(F(X^i)) \to 0 \text{ as } w(X^i) \to 0, \quad \text{for all } F. \tag{2}$$

2. The Model Algorithm

Interval subdivision methods for global optimization (c.f. [3], [5], [7], [8], [15], [19], [20]) usually start from an initial box $X \in \mathbb{I}^n$, subdivide X and store the subboxes in a list L, and discard subboxes which are guaranteed not to contain a global minimizer, until the desired accuracy (width) of the interval vectors in the list is achieved. To do so, several special steps and tests are applied (cut-off test, monotonicity test, concavity test, interval Newton-like step, or local search procedures).

Our model algorithm has the most important common features of such interval subdivision methods for global optimization, but it includes no local search procedure (c.f. [3]), no concavity test, and no Newton-like steps, since the latter require the inclusion of the Hessian. On the other hand, the cut-off and monotonicity tests are applied, because their usage does not require additional information on the problem (see below). It would not make sense to skip these tests. Although, cross-effects of the direction selection rules and the skipped steps are possible, the investigation of their numerical implication is the subject of an other study.

ALGORITHM 2.1. $\mathsf{GlobalOptimize}\,(f, X, \varepsilon, Y, F^*, L)$

1. $Y := X$; $L := \{\,\}$; $\widetilde{f} := f(m(X))$; $\qquad$ $\{$Upper bound for $f^*\}$
2. $k :=$ $\mathsf{OptimalComponent}\,(Y)$; $\qquad$ $\{$Direction Selection$\}$
3. $\mathsf{Bisection}\,(Y, k, U^1, U^2)$;
4. **for** $i := 1$ **to** 2 **do**
5. $\quad F_U := F(U^i)$; **if** $\widetilde{f} < \underline{F_U}$ **then next**$_i$;
6. $\quad$ **if** $\mathsf{MonotonicityTest}\,(\nabla F(U^i))$ **then next**$_i$;
7. $\quad L := L + (U^i, \underline{F_U})$; $\qquad$ $\{$Store U^i and $\underline{F_U}$ in $L\}$
8. **if** $L = \{\,\}$ **then return**;
9. $(Y, \underline{F_Y}) :=$ $\mathsf{Head}\,(L)$; $L := L - (Y, \underline{F_Y})$;
10. $\widetilde{f} := \min\{\widetilde{f}, f(m(Y))\}$; $\qquad$ $\{$Improve upper bound for $f^*\}$
11. $L :=$ $\mathsf{CutOffTest}\,(L, \widetilde{f})$;
12. **if** $w(F(Y)) > \varepsilon$ **then goto** 2;
13. $F^* := [\underline{F_Y}, \widetilde{f}]$; $\qquad$ $\{$Best possible inclusion of $f^*\}$
14. **return** Y, F^*, L.

We use the notation "+" for entering and "−" for discarding elements in the list L. $\mathsf{Head}\,(L)$ delivers the first element of L. For abbreviation, we write $\underline{F_Y}$ instead of $\min F(Y)$.

We call the interval vector Y, which is first set in Step 1 and updated in Step 9, the *leading box*, and the leading box of the m-th iteration is denoted by Y^m.

In contrast to the model algorithm used in [6], we used a simplified version of the algorithm from [7] and [21]. It incorporates the cut-off and monotonicity tests according to the following sub-algorithms.

ALGORITHM 2.2. CutOffTest $(L, \widetilde{f})$

for all $(Y, \underline{F_Y}) \in L$ **do**
    if $\widetilde{f} < \underline{F_Y}$ **then** $L := L - (Y, \underline{F_Y})$;
return L;

ALGORITHM 2.3. MonotonicityTest (G)

for $i := 1$ **to** n **do**
    if $0 \notin G_i$ **then return** *true*;
return *false*;

Since we do not do anything special to handle boundary points, the monotonicity test may discard subboxes containing global minimizer points if they lie on the boundary of X. Thus, we assume in the following that there exists a stationary point $x^* \in X$ for which $f(x^*) = f^*$ which makes sense, for the aim of our study is investigating the impact of the direction selection rules on the convergence of Algorithm 2.1.

Our model algorithm uses a special ordering of the subdivided boxes Y in the pending list L. The boxes Y are stored as pairs $(Y, \underline{F_Y})$ sorted in nondecreasing order with respect to the $\underline{F_Y}$ as a first ordering criterion and in decreasing order with respect to the age of the boxes as a second ordering criterion. Therefore, a newly computed pair is stored in the list L according to the following ordering rule (c.f. [21]):

$$\left.\begin{array}{ll} \bullet \text{ either} & \underline{F_W} \le \underline{F_Y} < \underline{F_Z} \text{ holds,} \\ \bullet \text{ or} & \underline{F_Y} < \underline{F_Z} \text{ holds, and } (Y, \underline{F_Y}) \text{ is the first element of the list,} \\ \bullet \text{ or} & \underline{F_W} \le \underline{F_Y} \text{ holds, and } (Y, \underline{F_Y}) \text{ is the last element of the list,} \\ \bullet \text{ or} & (Y, \underline{F_Y}) \text{ is the only element of the list,} \end{array}\right\} \quad (3)$$

where $(W, \underline{F_W})$ is the predecessor and $(Z, \underline{F_Z})$ is the successor of $(Y, \underline{F_Y})$ in L.

That is, the second components of the list elements may not decrease, and a new pair is entered behind all other pairs with the same second component. Since the first element of the list has the smallest second component, we can directly use the corresponding box to compute $f(m(Y))$ for the improvement of $\widetilde{f}$ in performing the cut-off test. Due to this special ordering, we can also save some work when deleting elements in the cut-off test, because we can delete the whole rest of the list when we have reached the first element to be deleted.

3. Subdivision Direction Selection Rules

The main target of this paper is Step 2 of Algorithm 2.1. There, we can apply different rules trying to find an optimal component (coordinate direction) to bisect the box Y. We call these rules *interval subdivision direction selection rules* and we investigated four different rules. In OptimalComponent, each of the rules selects a direction k by using a merit function:

$$k := \min\left\{ j \mid j \in \mathcal{J} \quad \text{and} \quad D(j) = \max_{i=1}^{n} D(i) \right\}, \tag{4}$$

where $\mathcal{J} = \{1, 2, \ldots, n\}$ and $D(i)$ is determined by the given rule. The usual definition of such rules does not specify a certain coordinate direction if the maximum is achieved several times. Thus, we take the smallest one.

Rule A

The interval-width-oriented rule [15], [19], [23] chooses the coordinate direction with

$$D(i) := w(X_i). \tag{5}$$

This rule was justified by the idea of subdividing the original interval vector X in a uniform way. It has also been used for generating subdivision directions in other optimization procedures (e.g. [10]). Algorithm 2.1 with Rule A is convergent both with and without the monotonicity test (e.g. in [5] and [19]). This rule allows a relatively simple analysis of the convergence speed (as in [19], Chapter 3, Theorem 6).

Rule B

The rule of Hansen and Walster [8] selects the coordinate direction by using (4) with

$$D(i) := w(G_i(X)) \cdot w(X_i), \tag{6}$$

where $G(X) = \nabla F(X)$. It is a heuristical direction selection rule which aims to find the component with the largest value of

$$\begin{aligned} W_i \;=\; & \max_{t \in X_i} f\left(m(X_1), \ldots, m(X_{i-1}), t, m(X_{i+1}), \ldots, m(X_n)\right) \\ & - \min_{t \in X_i} f\left(m(X_1), \ldots, m(X_{i-1}), t, m(X_{i+1}), \ldots, m(X_n)\right). \end{aligned}$$

The factor W_i, that is assumed to reflect how much f varies as x_i varies over X_i, is then approximated by $w(G_i(X)) \cdot w(X_i)$.

Rule C

The rule of Ratz [21] can be formulated with (4) and

$$D(i) := w\left(G_i(X) \cdot (X_i - m(X_i))\right), \tag{7}$$

where again $G(X) = \nabla F(X)$. The underlying idea was to minimize the width of the inclusion

$$\begin{aligned} w(F(X)) &= w(F(X) - f(m(X))) \\ &\approx w(\nabla F(X) \cdot (X - m(X))) \\ &= w\left(\sum_{i=1}^{n} \frac{\partial F}{\partial x_i}(X) \cdot (X_i - m(X_i))\right) \\ &= \sum_{i=1}^{n} w\left(\frac{\partial F}{\partial x_i}(X) \cdot (X_i - m(X_i))\right). \end{aligned}$$

Obviously, the component i is to be chosen for which $w(\frac{\partial F}{\partial x_i}(X) \cdot (X_i - m(X_i)))$ is the largest. The important difference between (6) and (7) is that in rule C the width of the multiplied intervals is maximized and not the multiplied widths of the respective intervals, which deliver different values in general (due to the subdistributive law).

In [6] we remarked that the right hand side of (7) can be written as

$$\max\{|\min G_i(X)|, |\max G_i(X)|\} w(X_i)$$

and that Rules B and C give the same merit function value if and only if either $\min G_i(X) = 0$ or $\max G_i(X) = 0$. We also mentioned the relation of rule C to Lipschitzian partition methods for global optimization [17], [18] and to the "*maximum smear*" function (used as a direction selection merit function solving systems of nonlinear equations [11]).

Rule D

The fourth rule uses a relative width of the intervals and is defined by (4) and

$$D(i) := \begin{cases} w(X_i) & \text{if } 0 \in X_i, \\ w(X_i)/\min\{|x_i| \mid x_i \in X_i\} & \text{otherwise.} \end{cases} \tag{8}$$

It is derivative-free like Rule A, and it reflects the machine representation of the intervals. Consider the case when the width of one component interval is greater than all other component widths, but the minimum and maximum values of this interval are nearly adjoining machine numbers. In this case the subdivision of the other components is more important than the subdivision of the "large" component.

Figures 1 and 2 show the distributions of subboxes for the discussed direction selection rules A, B, C, and D, respectively, when solving the *Branin* problem and

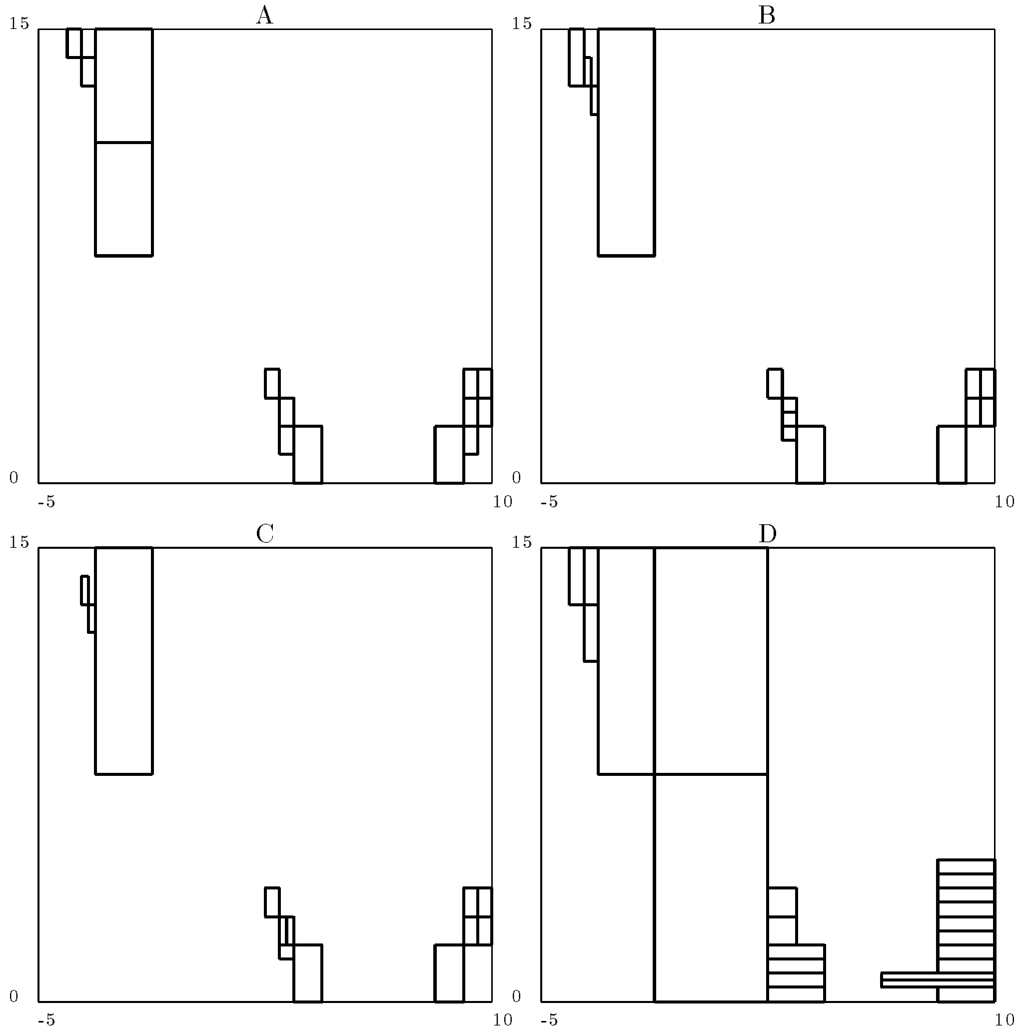

Figure 1. Remaining subintervals after 50 iteration steps of Algorithm 2.1 with the direction selection rules A, B, C, and D for the Branin problem

the *Six-Hump-Camel-Back* problem (see the Appendix for their definition) with Algorithm 2.1.

Figure 1 shows the situations after 50 iterations of the model algorithm for the Branin problem. The numbers of subboxes are 14, 14, 13, and 23, respectively. Rule A tends to form square-like boxes, while the others produce elongated subboxes.

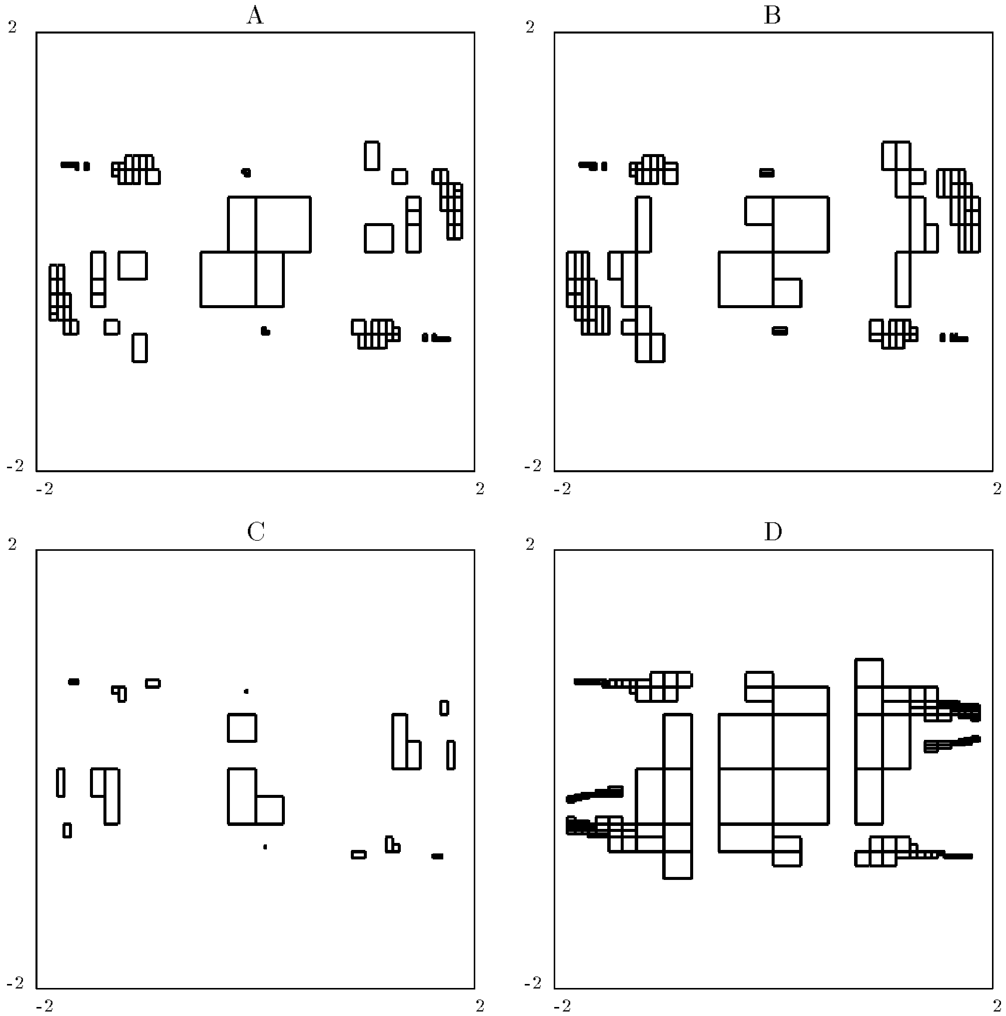

Figure 2. Remaining subintervals after 500 iteration steps of Algorithm 2.1 with the direction selection rules A, B, C, and D for the Six-Hump-Camel-Back problem

The greatest volume decrease is due to Rule C, the least volume decrease is due to Rule D.

Figure 2 shows the situations after 500 iterations of the model algorithm for the Six-Hump-Camel-Back problem. The numbers of subboxes are 94, 100, 31, and 167, respectively. Here, Rules A and D produced square-like boxes, while Rules B and C produced elongated subboxes. Again, the greatest volume decrease is due to Rule C, the least volume decrease is due to Rule D.

EXAMPLE 3.1 Let us consider the function $f(x) = x_1 \cdot x_2 \cdot x_3$ to demonstrate the influence of the different direction selection rules on the width of the interval function evaluation of f. We use

$$X = \begin{pmatrix} [0,1] \\ [-10,20] \\ [1000,2000] \end{pmatrix}$$

with

$$G = \nabla F(X) = \begin{pmatrix} [-20000,40000] \\ [0,2000] \\ [-10,20] \end{pmatrix} \quad \text{and} \quad c = m(X) = \begin{pmatrix} 0.5 \\ 5 \\ 1500 \end{pmatrix}.$$

Applying Rule A, we get

$$\begin{aligned} D(1) &= w(X_1) = 1 \\ D(2) &= w(X_2) = 30 \\ D(3) &= w(X_3) = 1000 \end{aligned}$$

and choose $k := 3$. Thus, we bisect in

$$U^1 = \begin{pmatrix} [0,1] \\ [-10,20] \\ [1000,1500] \end{pmatrix} \quad \text{and} \quad U^2 = \begin{pmatrix} [0,1] \\ [-10,20] \\ [1500,2000] \end{pmatrix},$$

and we get $F(U^1) = [-15000, 30000]$ and $F(U^2) = [-20000, 40000]$.

Applying Rule B, we get

$$\begin{aligned} D(1) &= w(G_1) \cdot w(X_1) = 60000 \cdot 1 = 60000 \\ D(2) &= w(G_2) \cdot w(X_2) = 2000 \cdot 30 = 60000 \\ D(3) &= w(G_3) \cdot w(X_3) = 30 \cdot 1000 = 30000 \end{aligned}$$

and choose $k := 1$. Thus, we bisect in

$$U^1 = \begin{pmatrix} [0,0.5] \\ [-10,20] \\ [1000,2000] \end{pmatrix} \quad \text{and} \quad U^2 = \begin{pmatrix} [0.5,1] \\ [-10,20] \\ [1000,2000] \end{pmatrix},$$

and we get $F(U^1) = [-10000, 20000]$ and $F(U^2) = [-20000, 40000]$.

Applying Rule C, we get

$$\begin{aligned} D(1) &= w(G_1 \cdot (X_1 - c_1)) = w([-20000, 20000]) = 40000 \\ D(2) &= w(G_2 \cdot (X_2 - c_2)) = w([-30000, 30000]) = 60000 \\ D(3) &= w(G_3 \cdot (X_3 - c_3)) = w([-10000, 10000]) = 20000 \end{aligned}$$

and choose $k := 2$. Thus, we bisect in

$$U^1 = \begin{pmatrix} [0,1] \\ [-10,5] \\ [1000,2000] \end{pmatrix} \quad \text{and} \quad U^2 = \begin{pmatrix} [0,1] \\ [5,20] \\ [1000,2000] \end{pmatrix},$$

and we get $F(U^1) = [-20000, 10000]$ and $F(U^2) = [0, 40000]$.

Applying Rule D, we get

$$\begin{aligned} D(1) &= w(X_1) = 1 \\ D(2) &= w(X_2) = 30 \\ D(3) &= w(X_3)/1000 = 1 \end{aligned}$$

and choose $k := 2$. Thus, we bisect as for Rule C getting the same boxes and interval function evaluations as above.

Assuming now that the upper bound $\tilde{f} = -10000$ for the global minimum value f^* is already known, we can discard U^2 in the cut-off test if we use Rule C or D.

4. Convergence and the direction selection rules

In the following, we summarize the main definitions and theoretical results given in [6] for the somewhat more general model algorithm used in the present paper, and we investigate the relations between the subdivision selection rules and convergence properties of Algorithm 2.1. The difference between the algorithms is that in [6] the list ordering was not specified for elements with equal lower bound on the objective function values. With the special ordering studied in the present paper, we can prove stronger convergence statements.

For our theoretical study, we defined the sequence of interval vectors that can be produced by the model algorithm, and we specified a property (*balanced*) of the subdivision direction selection rules that can ensure convergence for the model algorithm.

DEFINITION 4.1 We call an infinite sequence of interval vectors $(Y^s)_{s=0}^{\infty}$ an *infinite subdivision sequence of* Y, if $Y^0 = Y$ and if for each nonnegative integer s the box Y^{s+1} is given as $Y_j^{s+1} = Y_j^s$ for $j = 1, \ldots, k-1, k+1, \ldots, n$, and either $Y_k^{s+1} = [\min Y_k^s, m(Y_k^s)]$, or $Y_k^{s+1} = [m(Y_k^s), \max Y_k^s]$, where k is the direction selected by the given rule with Y^s, $F(Y^s)$ and $\nabla F(Y^s)$.

If we assume that the box Y is not discarded by the monotonicity test and $\varepsilon = 0$, it is easy to see, that the set of leading boxes $(Y^s)_{s=0}^{\infty}$ contains at least one infinite subdivision sequence. It may contain infinite subdivision sequences but also finite sequences of subboxes that end with a box Y, the subdivision of which resulted in such subboxes, that either $0 \notin \nabla F(U)$ or $f^* \geq F(U)$ holds for these. The latter finite sequences do not affect the convergence of the procedure.

DEFINITION 4.2 We call a direction selection rule *balanced*, if for all interval vectors X, for all isotone inclusion functions $F(X)$ and $\nabla F(X)$ having property (2), and for each infinite subdivision sequence of X that is a subsequence of the leading boxes $(Y^s)_{s=0}^{\infty}$, the sequence of directions generated by the given rule contains each k of the possible directions $1, 2, \ldots, n$ for which $w(X_k) > 0$ infinitely many times.

The rules fitting Definition 4.2 do not necessarily deliver the directions in a uniform way, but each direction is chosen again after a finite number of iteration steps.

We denote the set of accumulation points of the sequence $(Y^s)_{s=0}^{\infty}$ by A. Recall that the inclusion functions $F(X)$ and $\nabla F(X)$ are assumed to be isotone and to satisfy (2). For the sake of convergence investigation, we set the stopping criterion parameter ε to zero and we assume that $w(X) > 0$ (otherwise the solution requires no search and thus no subdivision). Recall also, that we assume that there exists a stationary point $x^* \in X^*$ for which $f(x^*) = f^*$.

The following three theorems and two corollaries have been proven in [6] for a general model algorithm that did not assume that the pairs in the list L with equal second element are ordered according to their age. Hence, these results hold also for Algorithm 2.1.

THEOREM 4.3 *Algorithm 2.1 converges in the sense that* $\lim_{s\to\infty} w(Y^s) = 0$*, if and only if the interval subdivision direction selection rule is balanced.*

THEOREM 4.4 *Assume that the interval subdivision direction selection rule is balanced. Then Algorithm 2.1 converges to global minimizer points in the sense that* $\lim_{s\to\infty} F(Y^s) = f^*$*,* $A \neq \emptyset$ *and* $A \subseteq X^*$*.*

One direction of the assertions of Theorem 4.3 and Theorem 4.4 are generalizations of some convergence results in [19] for the model algorithm with the studied class of direction selection rules. Notice that the opposite direction of all the statements in Theorem 4.4 is not true. For example, $A \neq \emptyset$ also holds if the direction selection rule is not balanced.

THEOREM 4.5 *Assume that Algorithm 2.1 converges for a given problem (1) to global minimizer points in the sense that* $\lim_{s\to\infty} F(Y^s) = f^*$*, and thus* $A \subseteq X^*$*. Then either the algorithm proceeds on the problem like an algorithm with a balanced direction selection rule, or there exists a box* $\hat{X} \subseteq X$ *such that* $f(x) = f^*$ *for all* $x \in \hat{X}$*, and* $w(\hat{X}_i) > 0$*,* $i \in \mathcal{J}$ *for all coordinate directions that are selected only a finite number of times.*

The essential meaning of Theorem 4.5 is that with the exception of problems for which a box $\hat{X}$ as defined above exists, the direction selection rule must be balanced to ensure convergence to global minimizer points.

COROLLARY 4.6 *The subdivision direction selection Rules A and D are balanced, and thus Algorithm 2.1 converges to global minimizer points with each of these rules.*

COROLLARY 4.7 *Either subdivision direction selection Rules B and C choose each direction* $i \in \mathcal{J}$ *for which* $w(X_i) > 0$ *an infinite number of times, and thus Algorithm 2.1 converges to global minimizer points with each of these rules, or the algorithm converges to a subbox of* X *with a positive width that contains only global minimizer points.*

The next definition and theorem point out the consequences of the new list ordering on the convergence properties.

DEFINITION 4.8 We call a global minimizer point x' of problem (1) *hidden*, if there exists a subbox $X' \subseteq X$ with positive volume ($w(X'_i) > 0$, $i = 1, 2, \ldots, n$) for which $x' \in X'$ and $\min F(X') = f^*$, while there exists an other global minimizer point x'' of the same problem such that $\min F(X'') < f^*$ holds for each subbox $X'' \subseteq X$ with positive volume that contains x''. Global minimizer points that do not fulfill the criteria of a hidden minimizer are called *non-hidden*.

THEOREM 4.9 *If Algorithm 2.1 with direction selection Rules A, B, C, or D converges to a global minimizer point* $x^* \in X^*$ *in the sense that it is an accumulation point* ($x^* \in A$) *of the sequence of leading boxes* $(Y^s)_{s=0}^{\infty}$, *then it converges to all non-hidden global minimizer points in the same sense.*

Proof: If we solve the global optimization problem (1) with given inclusion functions F and ∇F, then Algorithm 2.1 produces an infinite sequence of leading boxes $(Y^s)_{s=0}^{\infty}$ with a subsequence of $(Y^{s_l})_{l=0}^{\infty}$ such that $\lim_{l=0}^{\infty} Y^{s_l} = x^* \in X^*$. Consider an arbitrary non-hidden global minimizer point $x' \in X^*$, and assume that x' is not an accumulation point of the sequence $(Y^s)_{s=0}^{\infty}$. Then x' is contained only in a finite number of subboxes of $(Y^s)_{s=0}^{\infty}$, let Y^l be the last one of these subboxes.

As x' is a non-hidden global minimizer point, $\min F(Y^l) \leq \min F(Y^s)$ must hold for an infinite number of indices $s \geq l$. This implies that, due to the list ordering applied in Algorithm 2.1, Y^l is selected for further subdivision — and this is a contradiction. ∎

A direct consequence of Theorems 4.4 and 4.9 and Corollaries 4.6 and 4.7 is

COROLLARY 4.10 *1. If the interval subdivision direction selection rule is balanced, then the set of accumulation points* A *of the sequence* $(Y^s)_{s=0}^{\infty}$ *generated by Algorithm 2.1 contains all non-hidden global minimizer points of the given optimization problem (1).*

2. Algorithm 2.1 both with subdivision direction selection Rules A and D converges to all non-hidden global minimizer points of problem (1).

3. Either Algorithm 2.1 both with subdivision direction selection Rules B and C converges to all non-hidden global minimizer points of problem (1), or the algorithm converges to a set of subboxes of X *with positive width that contain only non-hidden global minimizer points.*

EXAMPLE 4.11 We want to find global minimizer points of problem (1) with $f(x) = x_1^2 x_2^4$ on the interval vector $X = [0,1]^2$. We use the range functions as inclusion functions, so $F(X) = f_{\mathrm{rg}}(X) = X_1^2 X_2^4$, and $G_1(X) = \frac{\partial F}{\partial x_1}(X) = 2X_1 X_2^4$, and $G_2(X) = \frac{\partial F}{\partial x_2}(X) = 4X_1^2 X_2^3$. The set of global minimizer points is $X^* = [0,1] \times [0,0] \cup [0,0] \times [0,1]$. Since $\min F(Y) = 0 = f^*$ for each subbox $Y \subseteq X$, all the global minimizer points are non-hidden. According to Theorem 4.9, X^* is equal to the set of accumulation points A of the sequence of leading boxes $(Y^s)_{s=0}^{\infty}$.

Using the direction selection Rule C with Algorithm 2.1, we get

$$D(1) = w(G_1(Y^0)(Y_1^0 - m(Y_1^0))) = w([0.0, 2.0][-0.5, 0.5]) = w([-1.0, 1.0]) = 2$$
$$D(2) = w(G_2(Y^0)(Y_2^0 - m(Y_2^0))) = w([0.0, 4.0][-0.5, 0.5]) = w([-2.0, 2.0]) = 4$$

for $Y^0 = X$. The second coordinate is selected for subdivision. Consider now a subbox of the form $[0,1] \times [0,d]$ (where $d > 0$). For this box $D(1) = 2d^4$ and $D(2) = 4d^4$, and thus always the direction $k = 2$ is chosen. Hence, the subsequence $[0,1] \times [0,d]$ $(d = 1, 0.5, 0.25, \ldots)$ of $(Y^s)_{s=0}^{\infty}$ converges to the interval vector $Y^* = [0.0, 1.0] \times [0.0, 0.0]$ without a single subdivision in the first coordinate (c.f. Corollary 4.7). According to the comment after the definition of Rule C, the merit function values and thus the selected directions are the same with Rule B, i.e. the same result interval vector is obtained by applying Rule B.

EXAMPLE 4.12 Our algorithm with Rule B may become non-convergent if we remove the monotonicity test. Consider the function $f(x) = x_1 + x_2^2$. With $\frac{\partial F}{\partial x_1} = [1,1]$, Rule B chooses always $k = 2$ for all interval vectors with $w(X_i) > 0$, $i = 1, 2$. The list ordering does not play any role in the direction selection, and hence Algorithm 2.1 shows the same behavior as the model algorithm of [6]. Therefore, $\lim_{s\to\infty} \min F(Y^s) \neq \lim_{s\to\infty} \max F(Y^s)$, where Y^s is again the actual box Y in the s-th iteration. Although the probability to have this phenomenon in real-life problems is small, it is nonetheless noteworthy that this behavior differs from that of Rule A.

5. Numerical experiences

We list the functions f and starting interval vectors X used in our tests in the Appendix to supply a complete documentation of the problems input data. The first group of functions from S5 to RB is the group of standard test functions taken from [24]. We also used these functions in [6]. The rest of the functions are from [14] and from [22] with the exception of the Griewank functions (Griew) which are also taken from [24]. The last group of functions (R4 – R8) is new.

We carried out the numerical tests on a HP 9000/730 using an implementation of Algorithm 2.1 in PASCAL–XSC [12] Version 2.03. The program is a modification (simplification) of the code given in [7]. The inclusion functions were produced by natural interval extensions, i.e. they were all isotone and they fulfill condition (2). The gradients were calculated by automatic differentiation, thus no numerical or symbolic derivatives were used. In contrast to our earlier study [6], now the gradient was calculated in a single step, and thus the monotonicity test could not save the computation of certain components of the gradient. All the numerical results of the subsequent sections were obtained with $\varepsilon = 10^{-2}$.

Tables 1 to 4 contain the efficiency measures provided solving the test problems. The first column gives the problem name, and the second column gives the dimension of the problem. The efficiency measures for Rules B, C, and D are also expressed as percentages of the respective value for Rule A. In the second last lines the computational efforts are given which are necessary to solve the whole set of test problems. The percentages in these lines show how much effort is needed with the actual rule compared to the value obtained by Rule A. This is the expected ratio of improvement (if less than 100%) solving a large set of problems similar to the studied one. The average of percentages values (denoted by AoP) reflect the relative computational burden one can anticipate for a single problem if the actual rule is used instead of Rule A, according to the statistical information provided by the set of test problems.

5.1. Comparing the Standard Time Units

Table 1 summarizes the CPU times required for the solution of the global optimization test problems. The CPU times are expressed in standard time units to allow a fair comparison with results obtained on other computer platforms. The standard time unit (1000 real evaluations of the Shekel-5 function) was 0.18 sec on the computational environment described above.

The STU values given in Table 1 are substantially smaller than those in our earlier study [6], this is in part due to the better interval arithmetic implementation. According to the STU values, Rules B and C are better choices than Rule A or D. On the basis of the numerical tests made, we can expect 16% or 22% improvements in the computation time if we use Rules B or C instead of Rule A. Rule D causes about 19% increase. Completing a large set of problems similar to the studied set, Rule B require 62% less, Rule C 63% less, and Rule D 25% less CPU time.

5.2. Comparing the Number of Function Evaluations

Table 2 gives the number of objective function evaluations (NFE) necessary to solve the test problems. For practical applications, this measure together with the number of gradient evaluations is more important than the required CPU time, since the functions involved are usually more complex than those of the test problems (see e.g. [13], [20]). According to the present test results, 15% and 19% improvement can be expected if Rules B and C are used instead of Rule A, and Rule D causes 19% higher number of function evaluations. The sum of the numbers of function evaluations (and also that of the gradient evaluations) must be interpreted with care, because the individual complexities of the test problems are different. When a

Table 1. Standard time units required by the four methods for the solution of global optimization test problems

Function	Dim.	Rule A	Rule B (B/A)	Rule C (C/A)	Rule D (D/A)
S5	4	0.51	0.51 (100%)	0.51 (100%)	0.51 (100%)
S7	4	0.72	0.73 (101%)	0.70 (97%)	0.74 (103%)
S10	4	1.02	1.03 (101%)	0.98 (96%)	1.01 (99%)
H3	3	5.32	2.45 (46%)	2.01 (38%)	11.18 (210%)
H6	6	32.31	21.30 (66%)	18.23 (56%)	78.92 (244%)
GP	2	942.91	813.83 (86%)	839.19 (89%)	2630.72 (279%)
SHCB	2	2.42	2.54 (105%)	2.23 (92%)	3.13 (129%)
BR	2	1.37	1.26 (92%)	1.21 (88%)	2.75 (201%)
RB	2	0.06	0.04 (67%)	0.04 (67%)	0.07 (117%)
THCB	2	0.88	0.71 (80%)	0.68 (68%)	1.30 (147%)
L3	2	123.15	83.88 (68%)	83.10 (67%)	120.30 (98%)
L5	2	23.67	17.24 (73%)	17.10 (72%)	23.12 (98%)
L8	3	1.35	1.35 (100%)	1.35 (100%)	1.35 (100%)
L9	4	2.37	2.37 (100%)	2.37 (100%)	2.37 (100%)
L10	5	3.74	3.74 (100%)	3.74 (100%)	3.74 (100%)
L11	8	9.96	9.96 (100%)	9.96 (100%)	9.96 (100%)
L12	10	15.55	15.55 (100%)	15.55 (100%)	15.55 (100%)
L13	2	1.96	0.92 (47%)	0.92 (47%)	0.96 (100%)
L14	3	1.87	1.55 (83%)	1.55 (83%)	1.87 (100%)
L15	4	2.94	3.02 (103%)	2.87 (97%)	2.96 (101%)
L16	5	4.16	3.79 (91%)	3.63 (87%)	4.18 (101%)
L18	7	7.84	7.19 (92%)	7.19 (92%)	7.86 (101%)
Schw2.1	2	0.91	1.19 (130%)	1.19 (130%)	0.69 (76%)
Schw3.1	3	0.10	0.10 (100%)	0.10 (100%)	0.10 (100%)
Schw3.1p	3	0.10	0.10 (100%)	0.10 (100%)	0.10 (100%)
Schw2.5	2	0.11	0.12 (109%)	0.10 (91%)	0.14 (127%)
Schw2.7	3	4123.40	244.59 (6%)	239.01 (6%)	778.51 (19%)
Schw2.10	4	9.11	3.01 (33%)	3.09 (34%)	9.11 (100%)
Schw2.14	4	2.59	2.27 (88%)	2.24 (86%)	7.05 (272%)
Schw2.18	2	0.68	0.64 (94%)	0.64 (94%)	0.71 (104%)
Schw3.2	3	0.15	0.10 (67%)	0.10 (67%)	0.13 (87%)
Schw3.7	30	0.42	0.41 (98%)	0.38 (91%)	0.36 (86%)
Griew5	5	87.41	87.41 (100%)	87.41 (100%)	87.41 (100%)
Griew7	7	1046.80	1043.21 (99%)	1031.06 (98%)	1048.91 (101%)
R4	2	16.79	9.05 (54%)	8.21 (49%)	59.27 (353%)
R5	3	9.65	6.69 (69%)	3.64 (38%)	3.63 (38%)
R6	5	23.26	18.61 (80%)	11.26 (48%)	13.18 (57%)
R7	7	61.03	41.54 (68%)	23.93 (39%)	32.29 (53%)
R8	9	113.72	71.35 (63%)	44.31 (39%)	55.74 (49%)
Sum		6682.31	2525.35 (38%)	2471.88 (37%)	5021.88 (75%)
AoP			(84%)	(78%)	(119%)

Table 2. Number of function evaluations required by the four methods for the solution of global optimization test problems

Function	Dim.	Rule A	Rule B (B/A)	Rule C (C/A)	Rule D (D/A)
S5	4	87	87 (100%)	87 (100%)	87 (100%)
S7	4	93	93 (100%)	89 (96%)	93 (100%)
S10	4	95	97 (102%)	93 (98%)	95 (100%)
H3	3	419	197 (47%)	175 (42%)	877 (209%)
H6	6	1631	1077 (66%)	915 (56%)	3807 (233%)
GP	2	87217	76475 (88%)	79755 (91%)	263395 (302%)
SHCB	2	1283	1271 (99%)	1135 (88%)	1635 (127%)
BR	2	255	231 (91%)	223 (87%)	597 (234%)
RB	2	77	49 (64%)	49 (64%)	89 (115%)
THCB	2	731	591 (81%)	547 (74%)	1051 (144%)
L3	2	2805	1977 (71%)	1969 (70%)	2945 (105%)
L5	2	781	549 (70%)	549 (70%)	741 (95%)
L8	3	43	43 (100%)	43 (100%)	43 (100%)
L9	4	57	57 (100%)	57 (100%)	57 (100%)
L10	5	71	71 (100%)	71 (100%)	71 (100%)
L11	8	115	115 (100%)	115 (100%)	115 (100%)
L12	10	143	143 (100%)	143 (100%)	143 (100%)
L13	2	43	39 (90%)	39 (90%)	43 (100%)
L14	3	63	57 (90%)	57 (90%)	63 (100%)
L15	4	83	77 (93%)	75 (93%)	83 (100%)
L16	5	93	85 (91%)	83 (91%)	93 (100%)
L18	7	129	117 (91%)	117 (91%)	129 (100%)
Schw2.1	2	603	717 (119%)	717 (119%)	433 (72%)
Schw3.1	3	59	59 (100%)	59 (100%)	59 (100%)
Schw3.1p	3	59	59 (100%)	59 (100%)	59 (100%)
Schw2.5	2	137	137 (100%)	127 (93%)	159 (116%)
Schw2.7	3	29989	2051 (7%)	1999 (7%)	5533 (18%)
Schw2.10	4	605	247 (41%)	249 (41%)	625 (103%)
Schw2.14	4	745	687 (92%)	667 (90%)	1531 (206%)
Schw2.18	2	803	803 (100%)	803 (100%)	851 (106%)
Schw3.2	3	111	69 (62%)	69 (62%)	95 (86%)
Schw3.7	30	3	3 (100%)	3 (100%)	3 (100%)
Griew5	5	4095	4095 (100%)	4095 (100%)	4095 (100%)
Griew7	7	23039	23039 (100%)	23039 (100%)	23039 (100%)
R4	2	903	503 (56%)	479 (53%)	2919 (323%)
R5	3	259	185 (71%)	113 (44%)	111 (43%)
R6	5	283	313 (111%)	201 (71%)	231 (82%)
R7	7	699	489 (70%)	297 (42%)	389 (56%)
R8	9	1015	653 (64%)	421 (41%)	523 (52%)
Sum		159721	117607 (74%)	119783 (75%)	316907 (198%)
AoP			(85%)	(81%)	(119%)

similar set of problems is to be solved, the expected improvements are 26% for Rule B, 25% for Rule C, while Rule D means about twice as much function evaluations.

5.3. Comparing the Number of Gradient Evaluations

Table 3 provides the number of gradient evaluations (NGE). As mentioned earlier, the gradients are calculated in a single step, and not componentwise as in our previous study [6]. Thus the NFE is an upper bound on the NGE values. The remarkable stability in the NGE/NFE ratios found in the earlier paper is now even stronger, and the number of cases where NGE equals NFE is larger than in [6]. This fact is mainly due to the single step evaluation that does not allow skipping the calculation of some gradient components, and can also be caused to a smaller extent by the use of automatic differentiation that may result in less tight inclusions of the gradients than with the hand-coded routines. The range of the NGE/NFE values is between 70% and 100%.

According to the test results, 14% and 19% improvements can be expected if Rule B or Rule C is used instead of Rule A, while Rule D causes 18% higher number of gradient evaluations. When a similar set of problems is to be solved, the anticipated improvements are as high as 25% for Rule B and 24% for Rule C, while Rule D means about twice as much gradient evaluations.

5.4. Comparing the Space Complexity

Table 4 shows the minimal lengths of the list L necessary to solve the test problems with the studied direction selection rules. The joint space complexity of the whole set of test problems is the maximal value of the corresponding column. According to the test results, a list of length 8197 is enough to solve the set of test problems with Rule A, while the necessary list lengths for the other rules were 6729, 6740, and 19327, respectively. The latter ones represent –18%, –18% and +136% differences. The average of the percentages for the new rules were 89%, 86% and 116%, respectively.

5.5. Summary

Two dominant behaviors can be recognized by studying the numerical results: for about half of the test problems Rule B, and especially Rule C ensure much more efficient solution than Rule A, while Rule D is the worst in this sense. The improvements showed in Tables 1 to 4 are even stronger for this first subset of problems. For a smaller set of problems the differences due to the direction selection rules are moderate, just few percents. The few remaining test problems show various other patterns. It is remarkable that usually the same behavior characterized each problem in different tables.

Table 3. Number of gradient evaluations required by the four methods for the solution of global optimization test problems

Function	Dim.	Rule A	Rule B (B/A)	Rule C (C/A)	Rule D (D/A)
S5	4	87	87 (100%)	87 (100%)	87 (100%)
S7	4	93	93 (100%)	89 (96%)	93 (100%)
S10	4	95	97 (102%)	93 (98%)	95 (100%)
H3	3	407	189 (47%)	163 (40%)	861 (212%)
H6	6	1464	997 (68%)	850 (58%)	3215 (220%)
GP	2	73553	65473 (89%)	67834 (92%)	223601 (304%)
SHCB	2	1137	1107 (97%)	999 (88%)	1527 (134%)
BR	2	255	231 (91%)	223 (87%)	500 (196%)
RB	2	77	49 (64%)	49 (64%)	89 (115%)
THCB	2	707	535 (76%)	503 (71%)	1025 (145%)
L3	2	2452	1765 (72%)	1761 (72%)	2384 (97%)
L5	2	548	437 (80%)	437 (80%)	555 (101%)
L8	3	43	43 (100%)	43 (100%)	43 (100%)
L9	4	57	57 (100%)	57 (100%)	57 (100%)
L10	5	71	71 (100%)	71 (100%)	71 (100%)
L11	8	115	115 (100%)	115 (100%)	115 (100%)
L12	10	143	143 (100%)	143 (100%)	143 (100%)
L13	2	43	39 (90%)	39 (90%)	43 (100%)
L14	3	63	57 (90%)	57 (90%)	63 (100%)
L15	4	83	77 (93%)	75 (93%)	83 (100%)
L16	5	93	85 (91%)	83 (91%)	93 (100%)
L18	7	129	117 (91%)	117 (91%)	129 (100%)
Schw2.1	2	515	625 (121%)	625 (121%)	424 (82%)
Schw3.1	3	59	59 (100%)	59 (100%)	59 (100%)
Schw3.1p	3	59	59 (100%)	59 (100%)	59 (100%)
Schw2.5	2	137	137 (100%)	127 (93%)	158 (115%)
Schw2.7	3	25829	1534 (6%)	1496 (6%)	4722 (18%)
Schw2.10	4	605	242 (40%)	244 (40%)	605 (100%)
Schw2.14	4	742	687 (93%)	667 (90%)	1460 (197%)
Schw2.18	2	803	803 (100%)	803 (100%)	851 (106%)
Schw3.2	3	111	69 (62%)	69 (62%)	95 (86%)
Schw3.7	30	3	3 (100%)	3 (100%)	3 (100%)
Griew5	5	4095	4095 (100%)	4095 (100%)	4095 (100%)
Griew7	7	23039	23039 (100%)	23039 (100%)	23039 (100%)
R4	2	843	463 (56%)	443 (53%)	2851 (338%)
R5	3	259	185 (71%)	113 (44%)	111 (43%)
R6	5	283	313 (111%)	201 (71%)	231 (82%)
R7	7	699	489 (70%)	297 (42%)	389 (56%)
R8	9	1015	653 (64%)	421 (41%)	523 (52%)
Sum		140811	105319 (75%)	106649 (76%)	274547 (195%)
AoP			(86%)	(81%)	(118%)

Table 4. Space complexity of the four methods for the solution of global optimization test problems in terms of the necessary length of the list

Function	Dim.	Rule A	Rule B (B/A)	Rule C (C/A)	Rule D (D/A)
S5	4	14	15 (107%)	15 (107%)	14 (100%)
S7	4	17	18 (106%)	17 (100%)	17 (100%)
S10	4	18	20 (111%)	18 (100%)	18 (100%)
H3	3	27	17 (63%)	16 (59%)	42 (156%)
H6	6	183	99 (54%)	88 (48%)	372 (203%)
GP	2	6878	6729 (98%)	6740 (98%)	19327 (281%)
SHCB	2	164	158 (96%)	142 (87%)	174 (106%)
BR	2	25	22 (88%)	20 (80%)	51 (204%)
RB	2	16	15 (94%)	15 (94%)	18 (113%)
THCB	2	58	52 (90%)	48 (83%)	74 (128%)
L3	2	821	531 (65%)	526 (64%)	708 (86%)
L5	2	167	141 (84%)	140 (84%)	166 (99%)
L8	3	10	10 (100%)	10 (100%)	10 (100%)
L9	4	13	13 (100%)	13 (100%)	13 (100%)
L10	5	16	16 (100%)	16 (100%)	16 (100%)
L11	8	25	25 (100%)	25 (100%)	25 (100%)
L12	10	31	31 (100%)	31 (100%)	31 (100%)
L13	2	10	11 (110%)	10 (100%)	10 (100%)
L14	3	14	15 (107%)	14 (100%)	14 (100%)
L15	4	18	20 (111%)	21 (117%)	18 (100%)
L16	5	23	24 (104%)	25 (109%)	23 (100%)
L18	7	34	31 (91%)	30 (88%)	34 (100%)
Schw2.1	2	101	118 (117%)	118 (117%)	92 (91%)
Schw3.1	3	7	7 (100%)	9 (129%)	7 (100%)
Schw3.1p	3	7	7 (100%)	7 (100%)	7 (100%)
Schw2.5	2	15	15 (100%)	15 (100%)	19 (127%)
Schw2.7	3	8197	385 (5%)	377 (5%)	1172 (14%)
Schw2.10	4	267	96 (36%)	96 (36%)	249 (93%)
Schw2.14	4	96	71 (74%)	67 (70%)	272 (283%)
Schw2.18	2	24	24 (100%)	24 (100%)	28 (117%)
Schw3.2	3	17	14 (82%)	12 (71%)	14 (82%)
Schw3.7	30	2	2 (100%)	2 (100%)	2 (100%)
Griew5	5	704	704 (100%)	704 (100%)	704 (100%)
Griew7	7	5505	5632 (102%)	5249 (95%)	5505 (100%)
R4	2	116	76 (66%)	72 (62%)	264 (228%)
R5	3	34	27 (79%)	19 (56%)	20 (59%)
R6	5	52	37 (71%)	29 (56%)	36 (69%)
R7	7	62	47 (76%)	39 (63%)	50 (81%)
R8	9	72	57 (79%)	49 (68%)	64 (89%)
Maximum		8197	6729 (82%)	6740 (82%)	19327 (236%)
AoP			(89%)	(86%)	(116%)

The trends of the present test results are close to those reported in [6], where the algorithm used a different list ordering, componentwise calculated and hand-coded gradient inclusion functions. The few larger differences in the efficiency figures can be explained by the algorithmic changes.

Summarizing the consequences of the numerical tests, we can conclude that Rule C is the best choice in terms of most of the efficiency measures, closely followed by Rule B. Although Rule D was worse than Rule A for many of the test problems, for some cases (e.g. Schwefel No. 2.1 or R5) it was nonetheless the best rule from many points of view. The numerical experiences indicate that with the recognition of the problem type, a substantial amount of computational effort can be saved by using the proper one of the new direction selection rules. For some test problems, the right direction selection rule could cause dramatic improvements in terms of computation time or space complexity (which is of vital importance in some application fields). It should be stressed that the discussed algorithmic changes do not require additional information on the problems, and they provide the efficiency improvements on a very wide problem class.

Appendix

Problem descriptions

In the following, we list the functions f and starting interval vectors X used in our tests, the abbreviated and full names of the corresponding problems, and the dimensionality of the problems.

S5, S7, S10: Shekel ($x \in I\!R^4$):

$$f_{\mathrm{S}m}(x) = -\sum_{i=1}^{m} \frac{1}{(x-A_i)(x-A_i)^T + c_i},$$

where $A \in I\!R^{m\times 4}$, $c \in I\!R^m$, and $m = 5$, $m = 7$, and $m = 10$, respectively. We use $X_i = [0, 10]$, $i = 1, \ldots, 4$.

H3, H6: Hartman ($x \in I\!R^3$ and $x \in I\!R^6$, respectively):

$$f_{\mathrm{H}n}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{n} A_{ij}(x_j - P_{ij})^2\right).$$

for $n = 3$ and $n = 6$, where $A, P \in I\!R^{4\times n}$ and $c \in I\!R^n$. We use $X_i = [0, 1]$, $i = 1, \ldots, n$.

GP: Goldstein and Price ($x \in I\!R^2$):

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

We use $X_i = [-2, 2]$, $i = 1, \ldots, 2$.

SHCB: Six-hump camel-back function, Branin ($x \in I\!R^2$):

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4.$$

We use $X_i = [-2, 2]$, $i = 1, \ldots, 2$.

BR: Branin ($x \in I\!R^2$):

$$f(x) = \left(\frac{5}{\pi}x_1 - \frac{5.1}{4\pi^2}x_1^2 + x_2 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10.$$

We use $X_1 = [-5, 10]$ and $X_2 = [0, 15]$.

RB: Rosenbrock ($x \in I\!R^2$):

$$f(x) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2.$$

We use $X_i = [-1.2, 1.2]$, $i = 1, \ldots, 2$.

THCB: Three-hump camel-back function ($x \in I\!R^2$):

$$f(x) = 12x_1^2 - 6.3x_1^4 + x_1^6 + 6x_2(x_2 - x_1).$$

We use $X_i = [-3, 3]$, $i = 1, \ldots, 2$.

L3: Levy ($x \in I\!R^2$):

$$f(x) = \sum_{i=1}^{5} i\cos((i-1)x_1 + i)\sum_{j=1}^{5} j\cos((j+1)x_2 + j).$$

We use $X_i = [-10, 10]$, $i = 1, \ldots, 2$.

L5: Levy ($x \in I\!R^2$):

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

We use $X_i = [-10, 10]$, $i = 1, \ldots, 2$.

L8, L9, L10, L11, L12: Levy ($x \in \mathbb{R}^n$, $n = 3, 4, 5, 8, 10$, respectively):

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

We use $X_i = [-10, 10]$, $i = 1, \ldots, n$.

L13, L14, L15, L16, L18: Levy ($x \in \mathbb{R}^n$, $n = 2, 3, 4, 5, 7$, respectively):

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

We use $X_i = [-10, 10]$, $i = 1, \ldots, n$ for $n = 2, 3, 4$ and $X_i = [-5, 5]$, $i = 1, \ldots, n$ for $n = 5, 7$.

Schw2.1: Beale ($x \in \mathbb{R}^2$):

$$f(x) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2.$$

We use $X_1 = [-1.5, 7.5]$ and $X_2 = [-4, 5]$.

Schw3.1: Schwefel ($x \in \mathbb{R}^3$):

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

We use $X_i = [-10, 10]$, $i = 1, \ldots, 3$.

Schw3.1p: Schwefel ($x \in \mathbb{R}^3$):

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

with $P = [0.999, 1.001]$. We use $X_i = [-10, 10]$, $i = 1, \ldots, 3$.

Schw2.5: Booth ($x \in \mathbb{R}^2$):

$$f(x) = (x_1 + 2x_1 - 7)^2 + (2x_1 + x_2 - 5)^2.$$

We use $X_i = [-5, 5]$, $i = 1, \ldots, 2$.

Schw2.7: Box 3D ($x \in \mathbb{R}^3$):

$$f(x) = \sum_{k=1}^{10} \left(\exp(\frac{-kx_1}{10}) - \exp(\frac{-kx_2}{10}) - (\exp(\frac{-k}{10}) - \exp(-k))x_3 \right)^2 .$$

We use $X_i = [-10, 30]$, $i = 1, \ldots, 3$.

Schw2.10: Kowalik ($x \in \mathbb{R}^4$):

$$f(x) = \sum_{i=1}^{11} \left(a_i - x_1 \frac{b_i^2 + b_i x_2}{b_i^2 + b_i x_3 + x_4} \right) .$$

We use $X_i = [0, 0.42]$, $i = 1, \ldots, 4$.

Schw2.14: Powell ($x \in \mathbb{R}^4$):

$$f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4 .$$

We use $X_i = [-4, 5]$, $i = 1, \ldots, 4$.

Schw2.18: Matyas ($x \in \mathbb{R}^2$):

$$f(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2 .$$

We use $X_i = [-30, 30]$, $i = 1, \ldots, 2$.

Schw3.2: Schwefel ($x \in \mathbb{R}^3$):

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

We use $X_i = [-1.89, 1.89]$, $i = 1, \ldots, 3$.

Schw3.7: Schwefel ($x \in \mathbb{R}^{30}$):

$$f(x) = \sum_{i=1}^{30} x_i^{10} .$$

We use $X_i = [-0.184, 0.184]$, $i = 1, \ldots, 30$.

Griew5: Griewank ($x \in \mathbb{R}^5$):

$$f(x) = \sum_{i=1}^{5} \frac{x_i^2}{400} - \prod_{i=1}^{5} \cos(\frac{x_i}{\sqrt{i}}) + 1 .$$

We use $X_i = [-600, 600]$, $i = 1, \ldots, 5$.

Griew7: Griewank ($x \in I\!R^7$):

$$f(x) = \sum_{i=1}^{7} \frac{x_i^2}{4000} - \prod_{i=1}^{7} \cos(\frac{x_i}{\sqrt{i}}) + 1.$$

We use $X_i = [-600, 500]$, $i = 1, \ldots, 7$.

R4: Ratz ($x \in I\!R^2$):

$$f(x) = \sin(x_1^2 + 2x_2^2)\exp(-x_1^2 - x_2^2)$$

We use $X_i = [-3, 3]$, $i = 1, \ldots, 2$.

R5, R6, R7, R8: Ratz ($x \in I\!R^n$, $n = 3, 5, 7, 9$, respectively):

$$f(x) = \left(\sin^2\left(\pi\frac{x_1+3}{4}\right)\sum_{i=1}^{n-1}\left(\frac{x_i-1}{4}\right)^2\left(1+20\sin^2\left(\pi\frac{x_{i+1}+3}{4}\right)\right)\right)^2$$

We use $X_i = [-10, 10]$, $i = 1, \ldots, n$.

References

1. Alefeld G. and Herzberger J. (1983), *Introduction to Interval Computations*, Academic Press, New York.
2. Benson, H. P. (1982), *On the Convergence of two Branch-and-Bound Algorithms for Nonconvex Programming Problems*, J. Optim. Theory and Appl., **36**, 129–134.
3. Caprani, O., Godthaab, B., and Madsen, K. (1993), *Use of a Real-Valued Local Minimum in Parallel Interval Global Optimization*, Interval Computations, **3**, 71–82.
4. Csendes, T. (1988), *Nonlinear Parameter Estimation by Global Optimization — Efficiency and Reliability*, Acta Cybernetica, **8**, 361–370.
5. Csendes, T. and Pintér, J. (1993), *The Impact of Accelerating Tools on the Interval Subdivision Algorithm for Global Optimization*, European J. of Operational Research, **65**, 314–320.
6. Csendes, T. and Ratz, D. (1995), *Subdivision Direction Selection in Interval Methods for Global Optimization*, submitted for publication.
7. Hammer, R., Hocks, M., Kulisch, U., and Ratz, D. (1993), *Numerical Toolbox for Verified Computing I*, Springer-Verlag, Berlin.
8. Hansen, E. (1992), *Global Optimization Using Interval Analysis*, Marcel Dekker, New York.
9. Jansson, C. and Knüppel, O. (1992), *A Global Minimization Method: the Multi-Dimensional Case*, Report 92.1, Technische Universität Hamburg-Harburg.
10. Jones, D. R., C. D. Perttunen and B. E. Stuckman (1994), *Lipschitzian Optimization without the Lipschitz Constant*, J. of Optim. Theory and Appl., **79**, 157–181.
11. Kearfott, R. B. and Novoa, M. (1990), *INTBIS, a Portable Interval Newton/Bisection Package*, ACM T. on Mathematical Software, **16**, 152–157.
12. Klatte, R., Kulisch, U., Neaga, M., Ratz, D., and Ullrich, Ch. (1992), *PASCAL-XSC – Language Reference with Examples*, Springer-Verlag, New York.
13. Kristinsdottir, B. P., Zabinsky, Z. B., Csendes, T., and Tuttle, M. E. (1993), *Methodologies for Tolerance Intervals*, Interval Computations, **3**, 133–147.

14. Levy, A. V., Montalvo, A., Gomez, S., and Calderon, A. (1981), *Topics in Global Optimization*, Lecture Notes in Mathematics, No. 909, Springer-Verlag, Berlin.
15. Moore, R. E. (1966), *Interval Analysis*, Prentice Hall, Engelwood Cliffs.
16. Neumaier, A. (1990), *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge.
17. Pintér, J. (1986), *Extended Univariate Algorithms for n-dimensional Global Optimization*, Computing **36**, 91–103.
18. Pintér, J. (1992), *Convergence Qualification of Adaptive Partition Algorithms in Global Optimization*, Mathematical Programming **56**, 343–360.
19. Ratschek, H. and Rokne, J. (1988), *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester.
20. Ratschek, H. and Rokne, J. (1993), *Experiments Using Interval Analysis for Solving a Circuit Design Problem*, J. Global Optimization **3**, 501–518.
21. Ratz, D. (1992), *Automatische Ergebnisverifikation bei globalen Optimierungsproblemen*, Dissertation, Universität Karlsruhe.
22. Schwefel, H. (1991), *Numerical Optimization of Computer Models*, Wiley, New York.
23. Skelboe, S. (1974), *Computation of Rational Interval Functions*, BIT **4**, 87–95.
24. Törn, A. and Žilinskas, A. (1989), *Global Optimization*, Lecture Notes in Computer Science, No. 350, Springer-Verlag, Berlin.