Computing 65, 263–269 (2000)

Computing

A New Multisection Technique in Interval Methods for Global Optimization

L. G. Casado and **I. García,** Almería, and **T. Csendes,** Szeged

Received May 31, 1999; revised January 20, 2000

Abstract

A new multisection technique in interval methods for global optimization is investigated, and numerical tests demonstrate that the efficiency of the underlying global optimization method can be improved substantially. The heuristic rule is based on experiences that suggest the subdivision of the current subinterval into a larger number of pieces only if it is located in the neighbourhood of a minimizer point. An estimator of the proximity of a subinterval to the region of attraction to a minimizer point is utilized. According to the numerical study made, the new multisection strategies seem to be indispensable, and can improve both the computational and the memory complexity substantially.

AMS Subject Classifications: 65K05, 90C30.

Key Words: Global optimization, branch & bound, multisection.

1. Introduction

This paper investigates new multisection variants of a branch-and-bound algorithm [5, 8] for solving the box constrained global optimization problem [4, 8]:

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

where the n-dimensional interval $X \subseteq \mathbb{R}^n$ is the search region, and $f(x) : X \subset \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective function. The global minimum value of f is denoted by f^*, and the set of global minimizer points of f on X by X^*.

Herein real numbers are denoted by $x, y, \ldots$, and real bounded and closed interval vectors by $X = [\underline{X}, \overline{X}], Y = [\underline{Y}, \overline{Y}], \ldots$, where $\underline{X}_i = \min X_i$ and $\overline{X}_i = \max X_i$, for $i = 1, 2, \ldots, n$. The set of compact intervals is denoted by $\mathbb{I} := \{[a, b] | a \leq b, a, b \in \mathbb{R}\}$ and the set of n-dimensional interval vectors (also called boxes) by $\mathbb{I}^n$. A function $F : \mathbb{I}^n \rightarrow \mathbb{I}$ is called *inclusion function* of f in $X \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$, if $x \in X$ implies $f(x) \in F(X)$. In other words, $f(X) \subseteq F(X)$, where $f(X)$ is the range of the function f on X. It is assumed in the present study that the inclusion function of the objective function is available (possibly given by interval arithmetic [5, 8]).

The width of the interval X is defined by $w(X) = \overline{X} - \underline{X}$, 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 $m(X) = (\underline{X} + \overline{X})/2$, if $X \in \mathbb{I}$, and $m(X) = (m(X_1), m(X_2), \ldots, m(X_n))^T$, if $X \in \mathbb{I}^n$. F is said to be an *inclusion isotone function* over X if $\forall Y, Z \in \mathbb{I}^n(X)$ $Y \subseteq Z$ implies $F(Y) \subseteq F(Z)$. F is called an order α (or α-*convergent*) inclusion function of f over X if $\forall Y \in \mathbb{I}^n(X)$ $w(F(Y)) - w(f(Y)) \leq Cw(Y)^{\alpha}$, where C and α are some positive constants. We study the basic algorithm described in [3] but only the Cut-Off tests is used as the accelerating devices (Algorithm 1).

Algorithm 1. Basic B&B algorithm with midpoint and CutOff tests for Global Optimization. The notation "+" is used for entering and "−" for discarding elements in the tree.

proc *GlobalOptimize*$(X, f, F, \varepsilon, T, Q) \equiv$	
$\quad Q = \{\}$	Final Tree
$\quad T := (X, \underline{F}_X)$	Work Tree
$\quad \tilde{f} = \bar{F}_X$	Upper bound for f^*
$\quad$ **while**$(T \neq \{\})$	
$\qquad (X, \underline{F}_X) := \{(X, \underline{F}_X) \in T \mid \underline{F}_X = \min\{\underline{F}_{X_i}\}, \forall (X_i, \underline{F}_{X_i}) \in T\}$	
$\qquad T := T - (X, \underline{F}_X)$	Remove $(X, \underline{F}_X)$ from T
$\qquad \tilde{f} = \min\{\tilde{f}, \bar{f}(X)\}$	Improve upper bound for f^*
$\qquad$ *Subdivide*$(X, U^1, \ldots, U^s)$	
$\qquad$ **for** $i := 1$ **to** s	
$\qquad\quad F_U := F(U^i)$	
$\qquad\quad$ **if** $(\tilde{f} < \underline{F}_U)$	
$\qquad\qquad$ **then next$_i$**	
$\qquad\quad$ **if** $(w(U^i) < \varepsilon)$	
$\qquad\qquad$ **then** $Q := Q + (U^i, \underline{F}_U)$	Store U^i and $\underline{F}_U$in Q
$\qquad\qquad$ **else** $T := T + (U^i, \underline{F}_U)$	Store U^i and $\underline{F}_U$in T
$\quad T :=$ *CutOffTest*$(T, \tilde{f})$	$\forall (X, \underline{F}_X) \in T, \tilde{f} < \underline{F}_X \Rightarrow$ remove $(X, \underline{F}_X)$ from T
end	

2. Multisection in Global Optimization

Now we investigate the following ways of box subdivision. The first one (a) is the traditional bisection by the widest interval component [8]. According to new studies [1, 3, 5, 7], multisection may improve the efficiency of such branch-and-bound techniques. The second way of subdivision, (b) provides 2^n subintervals by halving all interval components. The subdivision techniques (c) and (d) result in 3^n and 4^n subintervals, respectively. The new subdivision method will use three parameters, P_1, P_2 and $p\tilde{f}$ and depending on the relative value of these parameters the current box will be subdivided using methods (a), (b) or (d). We have considered as an indicator parameter

$$p\tilde{f}(X) = \frac{\tilde{f} - \underline{F}(X)}{\overline{F}(X) - \underline{F}(X)} \in [0, 1]$$

which gives us the relative position of the value $\tilde{f}$ (see line 8 in Algorithm 1) in the $F(X)$ interval. According to our experiments, X subintervals with bigger $p\tilde{f}(X)$ values are closer to minimizer points. In [2] this parameter has been successfully used as a predictor of the computational cost of a particular subinterval.

P_1 and P_2 $(0 < P_1 < P_2 < 1)$ are two values considered as thresholds. Depending on the values of P_1, P_2 and $p\tilde{f}(X)$, X will be divided by a different strategy. Boxes with $p\tilde{f} < P_1$ will be subdivided according to rule (a); boxes with $P_1 \leq p\tilde{f} < P_2$ will be cut according to (b); boxes with $p\tilde{f} \geq P_2$ according to (d). Now we investigate to what extent $p\tilde{f}$, this easy to obtain information can be utilized.

Theorem 1. There exist optimization problems (1) for which the inclusion function $F(X)$ of $f(x)$ is isotone and α-convergent, and the following statements are true:

1. For an arbitrary large number $N(>0)$ of consecutive actual intervals of Algorithm 1 have the properties that: neither of these processed intervals contains a global minimizer point, and the related $p\tilde{f}$ values are larger than a preset $P_2 < 1$,
2. there exists a subsequence of the actual intervals converging to a global minimizer point, for which $p\tilde{f} < P_1$ (for a fixed $0 < P_1$).

Proof: Consider an arbitrary optimization problem (1) that has two separate global minimizer points, x^*, x' $(f(x^*) = f^* = f(x'),\ x^* \neq x'$ and $x^*,\ x' \in X)$, and which has also a nonoptimal point: $\hat{x} \in X : f(\hat{x}) > f(x^*)$. A suitable inclusion function will be constructed on the basis of the naive interval arithmetic that provides an isotone and α-convergent inclusion function $F(X)$ (with $\alpha = 1$) [8].

1. Consider a point $\hat{x} \in X$ with $f(\hat{x}) > f^*$. For the set $\{X_i\}_{i=0}^{N-1}$ of actual intervals generated by subdivision for which $\hat{x} \in X_i$, we define the inclusion function as:

   $$G(Z) = [\underline{F}(Z) - Dw(Z)^{\alpha}, \hat{f}],$$

   where $\hat{f}$ is a value greater than or equal to $\overline{F}(X)$. It is easy to see that for a given suitable $\hat{f}$, and for any $N > 0$ one can find a large enough D in such a way that the resulting inclusion function values will imply the first statement of Theorem 1. Thus for the first N iterations the current best upper bound on the global minimum will be $\tilde{f} = \hat{f}$, and the lower bound of the inclusion function $G(X)$ will be minimal for the first N members of the interval sequence $\{X_i\}$. This is why the first N actual intervals will be $\{X_i\}_{i=0}^{N-1}$ (selected in Step 6 of Algorithm 1). By construction, the $p\tilde{f}$ values will be equal to one, and hence greater than $P_2 < 1$.

The inclusion isotonicity property obviously holds for the set $\{X_i\}_{i=0}^{N-1}$, and for all intervals related to these ones both the isotonicity and the α-convergence can be established (for most cases e.g. simply by switching back to the inclusion function F).

2. We define the inclusion function for intervals containing one of the global minimizer points x' as

$$G(Z) = [\underline{f}(Z) - Cw(Z)^{\alpha}, \overline{F}(Z)],$$

where $\underline{f}(Z)$ is the lower bound of the range of f on the argument interval, i.e. f^*. Also, the values of C and α are identical with those valid for the α-convergence of the underlying $F(X)$ inclusion function. For intervals containing the minimizer point x^* the inclusion function is given as

$$G(Z) = [\underline{f}(Z) - Cw(Z)^{\alpha}, E],$$

where the upper bound E is determined in such a way that the $p\tilde{f}$ value be smaller than the preset $P_1 > 0$ parameter for a fixed $\delta > 1$ real number close to one:

$$E = \underline{G}(Z) + \delta \frac{\tilde{f} - \underline{G}(Z)}{P_1}.$$

The sequence of actual intervals have N subintervals that belong to the subsequence containing the point $\hat{x}$ according to the first part of the proof. Then, after a possible transition period, all the actual intervals will contain either x', or x^* due to the α-convergence of the inclusion function. In this phase, the actual intervals will contain in rotation x' and x^*, since the interval selection rule chooses that subinterval which has the smallest lower bound on the global minimum value, and this lower bound is a function of the width of the given subinterval. It is why $\tilde{f}$ will converge to f^*, and without this construction for intervals containing x^* it could not be established.

The constructed inclusion function will be obviously inclusion isotone. The α-convergence property holds for the interval sequence converging to x' with the same C and α parameters as for $F(X)$. For the interval sequence converging to x^*, the width of the inclusion function value will be $\delta(\tilde{f} - \underline{G}(Z))/P_1$, which is δ/P_1 times more than the neighbouring interval in the sequence of actual intervals containing x', i.e. the α-convergence is proven with $\delta C/P_1$ and with the same α parameter.

Since only those interval sequences converging to x^* and x' are infinite, the α-convergence property can be obviously proven for all cases. Now fixing the inclusion functions defined in step 2, the earlier inclusion function values for those intervals containing x^* or x' but not discussed in step 2 of the proof can be enlarged to meet the inclusion isotonicity property. Finally, the $\hat{f}$ value is defined as the largest upper bound for the subintervals. □

The consequence of Theorem 1 is that even with an isotone and α-convergent inclusion function, the generated sequence of investigated subintervals can be distracted for an arbitrarily long time (N iterations) from the global minimizer points.

3. Numerical Results

The numerical tests were carried out on a Pentium II PC (233 Mhz, 64 Mbyte RAM) running under Linux operating system. The programs were coded in C. The interval arithmetic was implemented via the BIAS routines [6]. We used seven standard global optimization test problems: the Six-Hump-Camel-Back, Goldstein-Price, Hartman-3, Levy-3, Shekel-5, Shekel-7 and Shekel-10 with the usual search regions [4, 8]. The stopping criterion was that for all remained boxes $w(X) < \varepsilon$. Each time an actual box X is selected for subdivision from the list, we evaluate $f(m(X))$ to improve $\tilde{f}$. The number of such real function evaluations is the same as the number of iterations.

The two new strategies were: *PA*: P_1 and P_2 are optimized for all functions (same P_1 and P_2) parameter for all problems); and *P*: P_1 and P_2 are optimized specially for each single function. The optimal P_1 and P_2 parameters (constant during a B&B run) were determined by a simple stochastic global optimization algorithm SASS (Single Agent Stochastic Search) [9]. The objective function used was $f(P_1, P_2) = 3.5 * IE(P_1, P_2) + fE(P_1, P_2)$, where $f(P_1, P_2)$ is calculated by the number of interval and real function evaluations of the B&B Algorithm 1 using the *P* subdivision scheme.

The numerical results on the test set are comprised in Table 1. Data are displayed for $\varepsilon = 10^{-1}$, 10^{-2} and 10^{-3} to enable implications on the efficiency on different complexity problems. Here the average values and the relative efficiency figures compared to those achieved by the algorithm variant (a) are listed. The consequences are:

Regarding IE values, the multisection procedures (c) and (d) are definitely worse than the basic (a) method, while (b) resulted in better efficiency. The latter improvement grows with decreasing ε (with more difficult to solve problems). The improvement of the (b) multisection method ($s = 4$ in the algorithm) is in accordance with the results on similar multisection algorithms [7]. Both new variants (*PA* and *P*) performed even better, and the customized *P* was always the best (up to 39% better than (a)).

The number of real function evaluations improved with all the studied new algorithm variants. The *P* method does not give the smallest *fE* for $\varepsilon = 10^{-1}$ and 10^{-3}. This is in accordance with the merit function of the optimization gave the P_1 and P_2 values: the compound computational cost of $3.5 * \text{IE} + \text{fE}$ was minimized. The improvements for the *PA* and *P* methods were between 52 and 81%, and between 69 and 84%, respectively. With decreasing stopping criterion parameter ε the saving is improving.

The required *CPU* time, in contrast to *IE* and *fE*, can also describe the anticipated overhead. Its value correlated the most with the *IE* value in the present study. Thus the algorithm variant *P* was again always the best. The improvements for *PA* and *P* were between 14 and 38%, and between 24 and 45%, respectively.

Regarding memory complexity (*ML*), the important question is whether a given problem is to be solved within the available memory capacity. Although swapping

Table 1. Summary of the numerical results

		CPU		*IE*		*fE*		*ML*	
ε	subd.	aver.	/(a) %	aver.	/(a) %	aver.	/(a) %	aver.	/(a) %
10^{-1}	(a)	0.0784		1294		646		182	
	(b)	0.0661	84	1090	84	250	39	210	115
	(c)	0.1867	238	2644	204	252	39	256	141
	(d)	0.1657	211	2639	204	129	20	349	192
	PA	0.0676	86	1027	79	310	48	211	116
	P	0.0599	76	943	73	199	31	184	101
10^{-2}	(a)	0.4147		10572		5286		1011	
	(b)	0.3143	76	8038	76	1786	34	867	86
	(c)	0.6866	166	17102	162	1396	26	998	99
	(d)	0.9677	233	26380	250	1199	23	1053	104
	PA	0.3033	73	7595	72	1563	30	841	83
	P	0.2734	66	6696	63	1141	22	652	64
10^{-3}	(a)	5.5214		104666		52332		9455	
	(b)	3.6571	66	75839	72	13578	26	7625	81
	(c)	12.5427	227	256041	245	13268	25	10074	107
	(d)	7.2420	131	159114	152	5693	11	4284	45
	PA	3.4066	62	68367	65	9823	19	5009	53
	P	3.0634	55	64183	61	8512	16	4444	47

The averages (abbreviated as aver.) of the data on the test runs are summarized together with the relative values compared to the average values obtained by the (a) subdivision type. The notation used for column headers: *CPU*: execution time in seconds; *IE*: number of Interval function Evaluations (as $F(X)$); *fE*: number of real function Evaluations (as $f(x)$); *ML*: Maximum number of nodes in the working List necessary for the actual solution

can be a solution in some cases, its price, in enlarged CPU time, can be very high. The new multisection strategies decreased the memory requirements for $\varepsilon = 10^{-2}$ and $\varepsilon = 10^{-3}$, and the improvements for *PA* and *P* were 17 and 47%, and 36 and 53%, respectively.

Summing up the results of the numerical tests, it can be concluded that if the customizing of the P_1 and P_2 algorithm parameters is possible, then the *P* method is the best choice. Otherwise the P_1 and P_2 values obtained for the *PA* algorithm variant, can be used as good choices. As in the case of subdivision direction selection [1, 3], we expect that our results on the improved efficiency for the underlying optimization algorithm remain valid for a wider class of interval optimization procedures with different sets of accelerating devices and other algorithmic changes e.g. those discussed in [1, 3, 7].

Acknowledgements

This work was supported by the Ministry of Education of Spain (CICYT TIC96-1125-C03-03 and CICYT TIC96-1259E), by the Consejeria de Educación de la Junta de Andalucia (07/FSC/MDM), and by the Grants FKFP 0739/97, OTKA T 016413 and T 017241.

References

[1] Berner, S.: New results on verified global optimization. Computing *57*, 323–343 (1996).
[2] Casado, L. G., García, I.: New load balancing criterion for parallel interval global optimization algorithm. Proceedings of 16th IASTED International Conference. Garmisch-Partenkirchen, Germany, 321–323 (1998).

[3] Csendes, T., Ratz, D.: Subdivision direction selection in interval methods for global optimization. SIAM J. Numer. Anal. *34*, 922–938 (1997).
[4] Hansen, E.: Global optimization using interval analysis. New York: Marcel Decker, 1992.
[5] Kearfott, R. B.: Rigorous global search: continuous problems. Dordrecht: Kluwer, 1996.
[6] Knüppel, O.: BIAS – basic interval arithmetic subroutines. Technical Report 93.3, University of Hamburg 1993.
[7] Markót, M.Cs., Csendes, T., Csallner, A.E.: Multisection in interval branch-and-bound methods for global optimization II. Numerical tests. J. Global Optim. (in press).
[8] Ratschek, H., Rokne, J.: New computer methods for global optimization. Chichester: Ellis Horwood, 1988.
[9] Solis, F. J., Wets, J. B.: Minimization by random search techniques Math. Oper. Res. *6*, 19–50 (1981).

L. G. Casado, I. García
Department of Computer Architecture and Electronics
Almería University ES-04120
Almería, Spain
e-mail: leo@miro.ualm.es

T. Csendes
Institute of Informatics University of Szeged
H-6701 Szeged, Hungary
e-mail: csendes@inf.u-szeged.hu