Reliable Computing **6**: 179–191, 2000.
© 2000 *Kluwer Academic Publishers. Printed in the Netherlands.*

Interval Branch and Bound Algorithm for Finding the First-Zero-Crossing-Point in One-Dimensional Functions

LEOCADIO G. CASADO
Department of Computer Architecture and Electronics, University of Almería, Spain, e-mail: leo@miro.ualm.es

INMACULADA F. GARCÍA
Department of Computer Architecture and Electronics, University of Almería, Spain, e-mail: inma@iron.ualm.es

and

YAROSLAV D. SERGEYEV
ISI-CNR c/o DEIS, Universitá della Calabria, 87036 Rende (CS), Italy, and University of Nizhni Novgorod, Nizhni Novgorod, Russia, e-mail: yaro@si.deis.unical.it

(Received: 16 February 1998; accepted: 4 May 1999)

Abstract. In this paper, new ideas have been incorporated to a basic interval branch-and-bound algorithm which solves the problem of finding zeros in one-dimensional functions. These new ideas are based on the combination of a new rejection criterion, a selection strategy and an easy-to-obtain precondition of the problem at hand. The methodology described here focuses on finding the first zero crossing point, allowing the search of other zero crossing points to be avoided. In addition, a heuristic subdivision criterion has been proposed that, compared to bisection rule, provides improvements in most of the forty problems that have been tested.

1. Introduction

The problem of finding the First Zero Crossing Point (FZCP) for one-dimensional functions, subject to bound constraints, arises in many applications in several fields. Examples of such fields and applications are digital signal processing techniques, i.e. in the non-parametric identification of special signal [18], in the wavelet transform [10], in image processing [3], in matrix computation for the eigenvalue problem [1], in the design of electrical and electronic simulation software tools for time-domain and switching networks analysis [2], etc.

There are different alternative ways to formulate the problem, for example: Given a real function $f : s \in S \subset \mathbb{R} \rightarrow \mathbb{R}$; $S = [a, b]$, $a \leq b$, where S is the search region, to determine:

$$s^* = \min\{s \mid f(s) = 0,\ s \in S\} \tag{1.1}$$

or

$$\begin{aligned} s_u &= \min\{s \mid f(s) \cdot f(a) \leq 0,\ s \in S\}, \quad \text{followed by} \\ s_l &= \max\{s \mid f(s) \cdot f(a) \geq 0,\ s < s_u\}. \end{aligned} \tag{1.2}$$

Without loss of generality, in this work we have used the first formulation and incorporated $f(a) > 0$ as a boundary condition for the value of the function at the left point of the search region S. If $f(a) < 0$ the objective function could be changed from f to $-f$.

This problem could be handled by different approaches. The first approach is based on the use of simple grid techniques which produce a dense mesh starting from the left margin of the interval and progressing by ε until the value of $f(s)$ becomes less than zero [2].

The second approach consists in using standard local techniques for finding roots of equations, in order to achieve a rapid convergence to the point s^*. The main drawback of these methods is that convergence is not assured if $f(s)$ is a multiextremal function on S [14]. Moreover, if the objective function $f(s)$ has several roots, different choices of the initial conditions can lead to different solutions for the equation $f(s) = 0$. A standard and fast local search technique is the well known Newton's method, which frequently provides a quadratic convergence. Nevertheless, Newton's method may also fail depending on the initial point. For instance, Newton's method does not converge for the function $f(x) = x^3 - x$ if the initial point is $x = \frac{1}{\sqrt{5}}$.

A third approach is the proposal of Daponte et al. [4], [5], [17]. It works on the framework of Global Optimization techniques based on the computation of the Lipschitzean derivatives. In [4], [5], [17] several efficient solutions have been proposed for solving this kind of problem. They handle several applications from the electrical engineering field. Examples of such applications are related to: (i) the design of a neural Analog to Digital converter; (ii) the computation of the cutoff frequency of electrical filters and (iii) the measurement of the phase angle between two functions with the same frequency.

The fourth approach to solve (1.2) consists of the use of existing Nonlinear Global Constrained Optimization algorithms based on the Branch and Bound strategy and Interval Arithmetic, as those proposed in [6], [8], [15].

Our algorithm is based on the third and fourth approaches; i.e. it uses Interval Arithmetic and Branch and Bound techniques [6], [8] and also incorporates the ideas described in [4], [5], [17]. All our solutions are as efficient as those proposed in [4], [17] and, in addition, this implementation of the algorithm does not have overhead in computing derivatives.

This paper has been organized as follows: Section 2 describes our algorithmic proposal to solve (1.1), called FZCP-IBB (First Zero Crossing Point Interval Branch and Bound). The FZCP-IBB consists of a suitable selection rule and a new rejection criterion added to the traditional one. A new branching rule, is also proposed. In addition, it will be shown by some graphical examples how the algorithm works. In

Section 3, results obtained from experiments made on a set of forty test functions and performance evaluation of FZCP-IBB are shown. Finally, in Section 4 some conclusions are presented.

2. FZCP-IBB Algorithm

Our algorithmic proposal for solving problem (1.1) falls into the general framework of Branch and Bound (B&B) algorithms. The B&B method is a heuristic tree search. Its principle lies in successive decompositions of the original problem in smaller disjoint subproblems until the optimal solution is found. The search avoids visiting some subproblems which are known not to contain an optimal solution. B&B methods can be characterized by four rules: *Branching, Selection, Bounding* and *Elimination* [7], [11]. To ensure that the algorithm stops, a *Termination rule* has to be incorporated.

In order to proceed, we need a few designations. Real numbers are denoted by $x, y, \dots$, and real bounded and closed one-dimensional intervals by $X, Y, \dots$, where $X = [\underline{x}, \overline{x}] = \{x : \underline{x} \leq x \leq \overline{x}\}$, and $\underline{x}$ and $\overline{x}$ are the lower and upper bounds of the interval X, respectively. The set of compact intervals is denoted by $\mathbb{I} := \{[a,b] : a \leq b,\ a,b \in \mathbb{R}\}$ and the width of an interval X is defined by $w(X) = \overline{x} - \underline{x}$. When we express a real number as an interval, we shall usually retain the simpler noninterval notation, i.e, x instead of $[x,x]$. A function $F : \mathbb{I} \to \mathbb{I}$ is an *inclusion function* of $f : \mathbb{R} \to \mathbb{R}$, if $f(X) \subseteq F(X)$, where $f(X)$ is the range of f over set X.

For the FZCP-IBB algorithm, the Branch and Bound rules are the following:

Branching rule: The initial search region is denoted by $S = [\underline{s}, \overline{s}]$. The *Branching rule* divides the selected interval $X \subseteq S$ using bisection method. The generated subintervals (we shall call these "active" intervals) are stored in a work list in a non-decreasing order based on $\underline{x}$, i.e, the subinterval with greater $\underline{x}$ is stored first.

Selection rule: The next interval to be processed is that on the head of the work list, which is organized as a LIFO (Last-In-First-Out) scheme. Therefore, the algorithm always works on the most left active interval, examining the intervals from left to right and verifying that there is not any Zero Crossing Point on the left of the selected interval (see traditional elimination rule below).

Bounding rule: The fundamental Theorem of Interval Arithmetic provides a natural and rigorous way to compute an *inclusion function*. In the present study the *inclusion function* of the objective function is available by natural interval extension [9], [12], [15].

Elimination rule: For the First Zero Crossing Point problem, traditional interval elimination rule rejects intervals $X \subseteq S$ such that $\underline{F}(X) > 0$ or $\overline{F}(X) < 0$, because for both cases $0 \notin f(X)$ is assured. As was shown in the introduction, the function is preconditioned in such a way that f has to be positive at $\underline{s}$; i.e. $\underline{F}(\underline{s}) > 0$. If $0 \in F(\underline{s})$, it can be concluded that the First Zero Crossing Point is on $\underline{s}$. For functions with $\overline{F}(\underline{s}) < 0$, F has to be changed to $-F$. With this precondition, a negative value of $\overline{F}(\overline{x})$, $X \subseteq S$, determines that the First Zero Crossing Point is on the left of $\overline{x}$. We will denote the most left evaluated point $\overline{x}$ with a negative value of $\overline{F}(\overline{x})$, by *NegPoint*. Therefore, new rejection ideas can be added to traditional ones by deleting all the intervals X stored at the work list which verify that $\underline{x} \geq$ *NegPoint* because they do not contain the First Zero Crossing Point. We denote the new rejection ideas by *NegPoint test*.

Termination rule: In our implementation, intervals $X : w(X) \leq \varepsilon$, which were selected and non rejected, are stored at the so-called final list instead of at the work list. The input parameter ε determines the desired accuracy of the problem solution. The algorithm finishes when the work list is empty. Another *Termination rule* could consist of placing into the final list those intervals X : $w(F(X)) \leq \delta$, or a combination of the above two. Readers can consult [15], where termination criteria and rounding errors are studied.

In our description of FZCP-IBB the following notations will be used:

S: The search region.

F: The natural interval extension of f.

L: Priority Working List.

Q: Final List containing all the intervals $X : w(X) \leq \varepsilon$ and $0 \in F(X)$.

ε: Termination criterion. If $w(X) \leq \varepsilon$, X is stored in Q.

Following is a detailed description of the algorithm:

```
1.  FZCP-IBB ($S = [\underline{s}, \overline{s}], F, \varepsilon, L, Q$)
2.    $NegPoint := \infty$
3.    $Q := \{\emptyset\}$                                     Final List
4.    $L := \{S\}$                                            Work List
5.    IF ( $\overline{F}(\underline{s}) < 0$ )                            Precondition
6.      $F$ is changed to $-F$
7.    ELSEIF ( $0 \in F(\underline{s})$ )
8.      $Q = Q + [\underline{s}, \underline{s}]$
9.      RETURN $Q$            The algorithm finishes with the solution $\underline{s}$
10.   WHILE ( $L \neq \{\emptyset\}$ )
11.     $X :=$ Head($L$)
12.     $L := L - \{X\}$
13.     $F_X :=$ F(X)
```

```
14.     IF ( $0 \notin F(X)$ )                         Traditional interval rejection rule
15.       Reject(X)
16.     ELSE
17.       IF ( $w(X) \leq \varepsilon$ )
18.         Q := Q + {X}                               X belongs to the final solution
19.       ELSE
20.         Subdivide($X, X_l, X_r$)
21.         $L := L + \{X_r\}$                  The right subinterval is stored in Head(L)
22.         IF ( $\overline{F}(\overline{x_l}) \leq 0$ AND $\overline{x_l} < NegPoint$ )
23.           $NegPoint := \overline{x_l}$
24.           Remove all $X_i \in L : \underline{x_i} \geq NegPoint$          NegPoint test
25.         $L := L + \{X_l\}$                   The left subinterval is stored in Head(L)
26.   RETURN $Q$
27. End pseudo-code
```

The main advantage of FZCP-IBB, in addition to its reliability, is that it has no free parameters except the desired accuracy (ε) of the solution.

The algorithm starts by initializing: *NegPoint* to infinity, the final list Q to the empty set and the working list L to the search region, (lines 2, 3 and 4, respectively) and making the necessary changes to satisfy the precondition imposed on the problem (line 5). The algorithm can finish if $0 \in F(\underline{s})$ (line 7). Otherwise, it finishes when there are no intervals in the work list L (line 10). The next interval to be processed is given by *Selection rule* (line 11). *Bounding rule* is applied in line 13. If the interval cannot be rejected by traditional interval *Rejection rule* (line 14) and neither satisfies the *Termination rule* (line 17), it is subdivided by the *Branching rule* (line 20). The generated subintervals (X_l and X_r) are then stored at the head of the work list by the order established by the *Selection rule* (lines 21 and 25). As a previous step to the storage of X_l in the work list, the value of *NegPoint* may be updated as a result of the evaluation of F at $\overline{x_l}$ (line 22). If the value of *NegPoint* is improved, the *NegPoint test* is applied (line 24).

In Figures 1 and 2, graphical examples of FZCP-IBB algorithm execution are shown. There, each box represents an interval $[\underline{x}, \overline{x}]$ and its upper and lower bounds $[\underline{F}(X), \overline{F}(X)]$. Three types of boxes can be distinguished among the evaluated intervals. Light grey boxes are those which were rejected by traditional rejection rule, white ones represent the set of subdivided intervals and dark grey boxes are the set of intervals at the final list Q which defines the solution to the First Zero Crossing Point problem. The number of interval evaluations (NEvali) and the stopping parameter ε, are shown at the top and bottom of every example.

The traditional *Rejection rule* of FZCP-IBB algorithm is able to eliminate a great amount of subintervals, mainly for functions with no zero crossing points in the search region. Therefore, fast convergence of FZCP-IBB can be obtained.

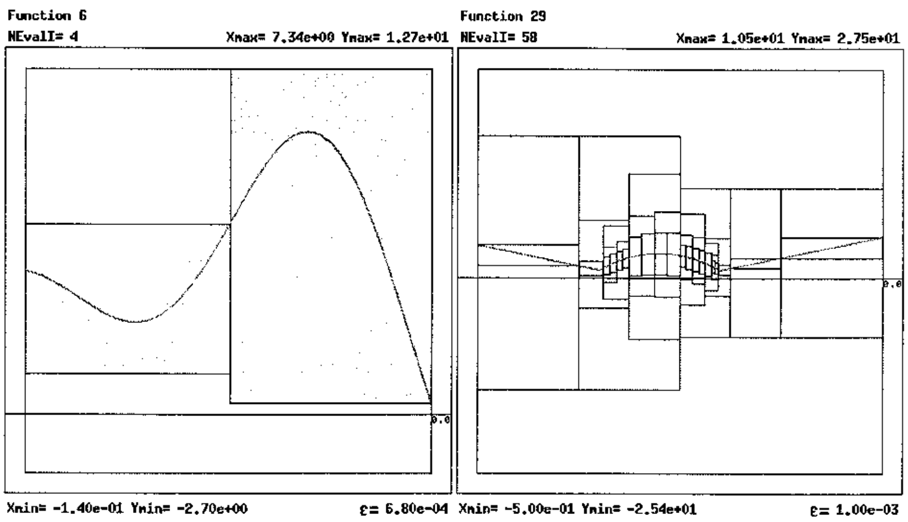

Figure 1. Graphical results of FZCP-IBB algorithm for problems 6 and 29 described in Table 1.

As an example of this behavior, the left picture of Figure 1 shows a function for which only four interval function evaluations were needed to determine that the function has no negative values. Nevertheless, due to the overestimation of the real range of the function given by the *inclusion function*, the traditional *Rejection rule* is applied only when intervals are small enough (see right picture of Figure 1). The performance of the method could be improved by applying better *inclusion functions* such as *centered forms* [8] [15], and *slope form* [13], [16].

In addition to the traditional *Rejection rule*, FZCP-IBB makes use of the values of *NegPoint* as a new rejection criterion, speeding up the convergence to the solution. Figure 2 shows the capability of the *NegPoint test* to eliminate intervals which do not contain the First Zero Crossing Point. Figure 2 is a graphical comparison of two FZCP-IBB algorithm executions, with and without the *NegPoint test.* In the left picture of Figure 2, intervals eliminated by *NegPoint test* (and not evaluated by Interval Arithmetic) have not been drawn. During the execution of FZCP-IBB, location of the *NegPoint* (at the right edge of boxes, X, with a negative value of $\overline{F}(\overline{x})$) was moving from right to left. If the *NegPoint test* (lines 22–24) is removed from the algorithm, all zero crossing points are determined (see right picture of Figure 2).

The proposed rejection and selection criteria work together providing a fast convergence by avoiding the search for all the zero crossing points of the objective function over L. Every time, the algorithm selects the most left interval in L. It examines intervals from left to right and any interval $X \in L$ is not evaluated unless the algorithm verifies that all the intervals on the left of X do not contain any zero crossing point. Working in this way, the method ensures that the found solution is

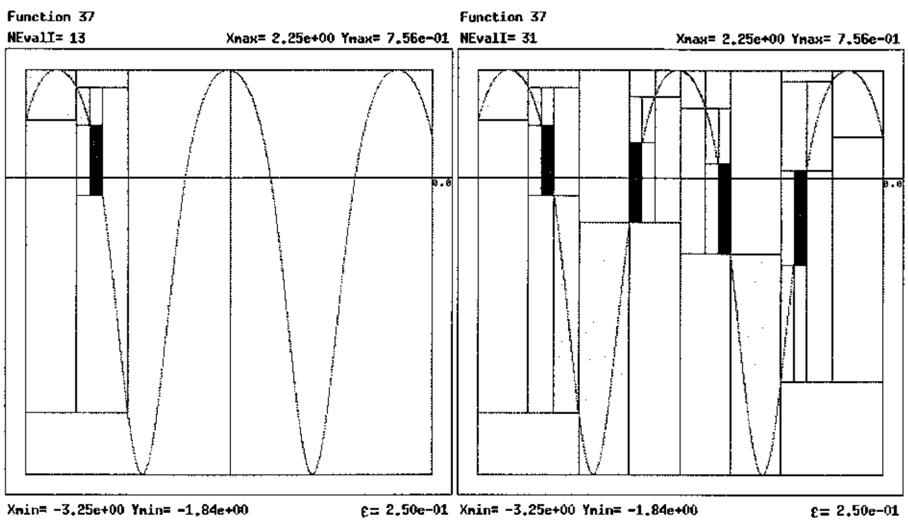

Figure 2. Graphical results of FZCP-IBB algorithm for problem 37 (see Table 1) with (left) and without (right) *NegPoint test.*

the First Zero Crossing Point of the function. It also tries to push the rejecting point, *NegPoint*, from right to left in order to eliminate the maximal sub-region from L and, therefore, to accelerate the search.

For differentiable functions, when high accuracy is needed, the convergence of FZCP-IBB algorithm could be accelerated by using the monotonicity test and the Newton's Method. In this way, FZCP-IBB algorithm will take advantage of the quadratic convergence of Newton's Method, which has to be applied only once; i.e. only on the interval $X \subseteq S$ where f is monotonic and $\overline{x} = NegPoint$.

In order to enforce the use of all available information and, therefore, to accelerate the search, a new adaptive non symmetric heuristic bisection method (AB) (instead of halving bisection, HB) is proposed. We have assumed that the location of the root in an interval is roughly proportional to the asymmetry in the inclusion function bounds about zero. As will be shown hereafter, the adaptive subdivision method reduces the number of interval function evaluations because it provides the possibility of rejecting wider intervals.

This AB method has been designed based on the following considerations:

- $\overline{F}(X) << w(F(X))$ means that most of the interval contains negative values of the function. Subdividing by the left part of the interval, a new value of *NegPoint* could be obtained and so a bigger subinterval can be rejected (the right generated subinterval, and also those intervals in the work list) than using a halving method.
- $\overline{F}(X) >> w(F(X))$ means that most of the interval may contain positive values of the function and subdividing by the left or right part of the interval, one (the wider is preferable) or both generated subintervals may be rejected by traditional

rejection rule. The interval will be divided by the left or right part depending on the value of $\overline{F}(\overline{x})$.

With AB method the generated subintervals are the following:

$$X_l = [\underline{x}, \underline{x} + \beta]$$

and

$$X_r = [\underline{x} + \beta, \overline{x}],$$

where

$$\beta = \begin{cases} 0.33 \times w(X) & \text{if } 0.00 \leq \dfrac{\overline{F}(X)}{w(F(X))} \leq 0.33, \\ w(X) \times \dfrac{\overline{F}(X)}{w(F(X))} & \text{if } 0.33 < \dfrac{\overline{F}(X)}{w(F(X))} \leq 0.66, \\ 0.66 \times w(X) & \text{if } 0.66 < \dfrac{\overline{F}(X)}{w(F(X))} \leq 1.00 \end{cases}$$

if $\overline{F}(X) - \overline{F}(\overline{x}) \leq \overline{F}(\overline{x}) - \underline{F}(X)$ and $0.66 < \dfrac{\overline{F}(X)}{w(F(X))} \leq 1.00$ then set $\beta = 1 - \beta$.

3. Numerical Results

In this section, it will be shown that the method proposed in this paper is able to obtain very accurate solutions of (1.1) by a few interval function evaluations. Also, the speed up yielded by the *NegPoint test* and the adaptive subdivision method will be evaluated.

A set of forty test functions, whose description is given in Table 1, were chosen for the algorithm performance evaluation. All the test functions are multiextremal, most of them have several zero crossing points and eleven functions have points of non-differentiability. Numerical results were obtained using two different values of the desired accuracy (ε) of the solution ($\varepsilon = w(S) \cdot 10^{-4}$ and $\varepsilon = w(S) \cdot 10^{-10}$).

In Tables 1 and 2 the following notation has been used for column headers:

N	Index of the function (for further references).
FZCP	First Zero-Crossing Point of the function.
S	Search region $[\underline{s}, \overline{s}]$.
NZCP	Number of Zero Crossing Point in S.
NL	Number of Local extrema (max. and min.).
GRID	Number of function evaluation using the Grid method with accuracy $\varepsilon = w(S) \cdot 10^{-4}$.

Table 1. Description of the test functions. The functions marked with $\surd$ have points of non-differentiability over $[\underline{s}, \overline{s}]$.

	N	Function f(x)	FZCP	S	NZCP	NL
	1	$-0.5x^2\ln(x)+5$	3.0117	[0.2,7]	1	2
	2	$-e^{-x}\sin(2\pi x)+1$	–	[0.2,7]	–	13
	3	$-\sqrt{x}\sin(x)+1$	1.1748	[0.2,7]	3	4
	4	$x\sin(x)+\sin(10x/3)+\ln(x)-0.84x+1.3$	2.9609	[0.2,7]	2	6
	5	$x+\sin(5x)$	0.8209	[0.2,7]	2	13
	6	$-x\sin(x)+5$	–	[0.2,7]	–	4
	7	$\sin(x)\cos(x)-1.5\sin^2(x)+1.2$	1.3408	[0.2,7]	4	7
	8	$2\cos(x)+\cos(2x)+5$	–	[0.2,7]	–	6
	9	$2\sin(x)e^{-x}$	3.1416	[0.2,7]	2	4
	10	$(3x-1.4)\sin(18x)+1.7$	1.2655	[0.2,7]	34	42
	11	$(x+1)^3/x^2-7.1$	1.3647	[0.2,7]	2	3
	12	$\begin{cases}\sin(5x)+2 & x\leq\pi\\ 5\sin(x)+2 & x>\pi\end{cases}$	3.5531	[0.2,7]	2	8
	13	$e^{\sin(3x)}$	–	[0.2,7]	–	9
	14	$\sum_{k=0}^{5} k\cos[(k+1)x+k]+12$	4.7831	[0.2,7]	2	15
	15	$2(x-3)^2-e^{x/2}+5$	3.2811	[0.2,7]	2	4
	16	$-e^{\sin(x)}+4$	–	[0.2,7]	–	4
	17	$\sqrt{x}\sin^2(x)$	3.1411	[0.2,7]	2	6
	18	$\cos(x)-\sin(5x)+1$	1.5708	[0.2,7]	6	13
	19	$-x-\sin(3x)+1.6$	1.9686	[0.2,7]	1	9
	20	$\cos(x)+2\cos(2x)e^{-x}$	1.1407	[0.2,7]	2	4
$\surd$	21	$x+\vert\sin(x)\cos(x)\vert$	–	[0.2,7]	–	10
$\surd$	22	$\vert\sin(x)+\cos(x)\vert+0.3$	–	[0.1,7]	–	6
	23	$-\sum_{k=1}^{5} k\sin[(k+1)x+k]+3$	−9.1405	[−10,10]	26	40
$\surd$	24	$\begin{cases}\cos(5x) & x\leq 3/2\pi\\ \cos(x) & \textit{otherwise}\end{cases}$	0.3142	[0,18]	12	13
$\surd$	25	$\begin{cases}\sin(x) & x\leq\pi\\ \sin(5x) & \textit{otherwise}\end{cases}$	−9.4248	[−10,10]	15	17
	26	$\ln(3x)\ln(2x)-1$	0.1472	[0.1,7]	2	3
	27	$\sum_{k=1}^{5} -\cos[(k+1)x]$	−9.8175	[−10,10]	38	41
$\surd$	28	$\begin{cases}\cos(5x) & 3/2\pi\leq x\leq 5/2\pi\\ \cos(x) & \textit{otherwise}\end{cases}$	1.5708	[0,4π]	8	9
$\surd$	29	$\begin{cases}-x+4 & x\leq 3\\ -8/9x^2+8x-15 & 3<x<6\\ x-5 & \textit{otherwise}\end{cases}$	–	[0,10]	–	5
$\surd$	30	$\begin{cases}x^2+0.5 & x\leq 1\\ 5\sin(2\pi x)+1.5 & 1<x\leq 3\\ x^2-8x+16.5 & \textit{otherwise}\end{cases}$	1.5485	[0,5]	4	8
	31	$\vert\sin(x)^3\cos(x)^3\vert+0.1$	–	[0,2π]	–	9
$\surd$	32	$\begin{cases}\sin(3x)+5 & x\leq 0\\ 7x^2-14x+5 & \textit{otherwise}\end{cases}$	0.4655	[−6,2]	2	10
	33	$-e^{-x}\sin(2\pi x)+0.5$	0.0924	[0,4]	2	10
	34	$(x^2-5x+6)/(x^2+1)-0.5$	1.2583	[−5,5]	1	4

Table 1. (*continued*)

	N	Function f(x)	FZCP	S	NZCP	NL
✓	35	$x\|\sin(x)\|+6$	−8.6593	[−10,10]	2	14
✓	36	$\|x\sin(x)\|-1.5$	−9.5820	[−10,10]	14	15
	37	$-e^{\sin(3x)}+1$	−2.0944	[−3,2]	4	7
	38	$-x+\sin(3x)+1$	1.0354	[0,6.5]	1	8
✓	39	$\begin{cases} \sin(x) & \sin(x) > \cos(x) \\ \cos(x) & \textit{otherwise} \end{cases}$	−9.4248	[−10,10]	7	14
	40	$(x+\sin(x))e^{-x^2}+0.8$	−0.8002	[−5,5]	2	4

TR-HB Only traditional *Rejection rule* is applied. *Branching rule* is halving bisection.

NP-HB Traditional *Rejection rule* and *NegPoint test* are applied. *Branching rule* is halving bisection.

NP-AB Traditional *Rejection rule* and *NegPoint test* are applied. *Branching rule* is adaptive non symmetric bisection.

In Table 2, results of the execution of the GRID method and several versions of FZCP-IBB algorithm described in this paper are shown. Numerical results demonstrate that for functions with no zero crossing points over the search interval, the GRID technique does not efficiently work because it has to scan the whole interval (10,000 function evaluations, for $\varepsilon = w(S) \cdot 10^{-4}$), while all the versions of FZCP-IBB only need a few interval function evaluations. In fact, for the subset of functions without zero crossing points (2, 6, 8, 13, 16, 21, 22, 29, and 31, see Table 2), the problem (1.1) was solved in very few iterations. For these kinds of functions, only the traditional rejection criterion does useful work. For instance, for function $N = 29$, TR-HB outperforms NP-HB and NP-AB because TR-HB does not need to evaluate $F(\overline{x})$.

For functions with several zero crossing points, *NegPoint test* incorporated to traditional rejection rule make the algorithm work significantly faster because only the First Zero Crossing Point is determined. For the set of test functions checked in this work, on average more than 90% of the work is saved with respect to the GRID method.

It is interesting to note cases where functions never reach negative values but touch the zero line in one or more points (for instance, function $N = 17$). Our results show that the GRID method was unable to find the solution while FZCP-IBB solved it in a few iterations. Similar results were obtained for the following modification made to function $N = 17$: $\sqrt{x}\sin^2(x) - 10^{-8}$, for which the GRID method does not work while **TR-HB**, **NP-HB** and **NP-AB** algorithms need only 55, 73 and 31 iterations respectively to reach the solution.

Comparing numerical results of **TR-HB** and **NP-HB**, it can be said that only for functions 1, 17, 19, 29 and 38, **TR-HB** outperforms **NP-HB**. However, for these

Table 2. Numerical results for GRID and the FZCP-IBB algorithms. Functions marked with $\surd$ have points of non-differentiability over $[\underline{s},\overline{s}]$. GRID ($\varepsilon = w(S)\cdot 10^{-10}$) $\geq 10^6$.

			$\varepsilon = w(S)\cdot 10^{-4}$			$\varepsilon = w(S)\cdot 10^{-10}$		
	N	GRID	TR-HB	NP-HB	NP-AB	TR-HB	NP-HB	NP-AB
	1	4,135	29	37	25	69	85	55
	2	10,000	1	1	1	1	1	1
	3	1,434	83	34	27	223	89	64
	4	4,061	125	56	45	321	108	84
	5	914	65	50	41	189	124	109
	6	10,000	3	4	4	3	4	4
	7	1,678	127	36	34	341	87	78
	8	10,000	1	1	1	1	1	1
	9	4,326	55	39	23	135	84	69
	10	1,567	663	46	47	2,031	95	87
	11	1,713	603	190	201	1,845	626	661
	12	4,931	55	39	29	135	89	67
	13	10,000	1	1	1	1	1	1
	14	6,740	77	68	60	181	141	118
	15	4,531	239	40	46	691	132	129
	16	10,000	1	1	1	1	1	1
	17	10,000	55	82	61	135	202	154
	18	2,016	185	36	28	521	85	69
	19	2,601	61	82	70	101	133	97
	20	1,384	65	44	33	171	129	91
$\surd$	21	10,000	1	1	1	1	1	1
$\surd$	22	10,000	1	1	1	1	1	1
	23	430	1,179	89	75	3,427	237	208
$\surd$	24	175	263	34	26	743	84	65
$\surd$	25	288	341	36	23	981	85	68
	26	69	51	35	26	131	85	65
	27	92	2,667	99	98	11,615	338	351
$\surd$	28	1,250	267	63	43	747	163	106
$\surd$	29	10,000	39	58	64	39	58	64
$\surd$	30	3,097	157	36	29	317	85	76
	31	10,000	1	1	1	1	1	1
$\surd$	32	8,082	157	56	53	469	164	145
	33	231	53	35	32	139	92	78
	34	6,259	65	61	47	165	161	129
$\surd$	35	671	53	36	31	143	96	73
$\surd$	36	210	339	34	29	967	86	67
	37	1,812	103	37	27	263	89	64
	38	1,593	33	34	23	73	87	51
$\surd$	39	288	167	36	23	447	85	61
	40	4,200	175	89	82	609	272	269
			$\varepsilon = w(S)\cdot 10^{-4}$			$\varepsilon = w(S)\cdot 10^{-10}$		
	Av.	**4,270**	**215.2**	**44.0**	**37.8**	**709.4**	**112.2**	**97.1**

functions the number of zero crossing points is one or none. It is also the case that these are the less difficult problems. On average the computational burden of **NP-HB** is 80% (85%) less than **TR-HB** for $\varepsilon = w(S) \cdot 10^{-4}$ ($\varepsilon = w(S) \cdot 10^{-10}$).

On the other hand, adaptive bisection branching ideas (**NP-AB** version of FZCP-IBB algorithm) permit the acceleration of the search with respect to the halving bisection (**NP-HB**). The speed up for the set of test functions moves in the interval $[-5\%, 35\%]$, but on average only 13% of improvement was obtained approximately for both values of ε.

4. Conclusions

In this paper, a traditional interval algorithm for solving the zero crossing points problem has been modified by incorporating new rejection and selection techniques that enable the detection of only the First Zero Crossing Point.

The algorithm with a suitable selection rule has been changed by introducing an additional rejection criterion and a precondition imposed to the problem at hand. If this new rejection criterion is not applied, all the zero crossing points of the objective function will be found with the desired accuracy and with the corresponding increase of the computational cost. Also, a heuristic adaptive branching rule has been proposed providing some improvements for most of the test functions.

Finally we would like to highlight that, for differentiable multiextremal functions, the speed up convergence of the algorithm proposed in this paper can be improved by using the monotonicity test to determine the strictly necessary subinterval where the Newton's Method should be applied.

Acknowledgements

The authors thank A. Molinaro for kindly placing at their disposal the test functions and the corresponding results for the grid technique. We would also like to thank the anonymous reviewers for their constructive criticism.

This work was supported by the Ministerio de Educación y Cultura of Spain (CICYT TIC96-1125-C03-03) and by the Consejería de Educación de la Junta de Andalucía (07/FSC/MDM).

References

1. Baoxin Li, Law, A. G., Raafat, H., Nguyen, P. H. and Yan, Y.-F.: Eigenvalues of Tridiagonal Matrices: An Alternative to Givens' Methods, *Computers Math. Applic.* **19** (4) (1990), pp. 89–94.
2. Bedrosian, D. and Vlach, J.: Time-Domain Analysis of Networks with Internally Controlled Switches, *IEEE Transactions on Circuits Syst.* **CAS-39** (3) (1992), pp. 199–212.
3. Clark, J. J.: Authenticathing Edges Produced by Zero-Crossing Algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI* **11** (1989), pp. 43–57.
4. Daponte, P., Grimaldi, D., Molinaro, A., and Sergeyev, Ya. D.: An Algorithm for Finding the Zero Crossing of Time Signals with Lipschitzeans Derivatives, *Measurements* **16** (1995), pp. 37–49.
5. Daponte, P., Grimaldi, D., Molinaro, A., and Sergeyev, Ya. D.: Fast Detection of the First Zero-Crossing in a Meassurement Signal Set, *Measurements* **19** (1) (1996), pp. 29–39.

6. Hansen, E.: *Global Optimization Using Interval Analysis* (*Pure and Applied Mathematics* **165**), Marcel Dekker, New York, 1992.
7. Ibaraki, T.: Theoretical Comparisons of Search Strategies in Branch and Bound Algorithms, *Int. J. Comput. Inform. Sci.* **5** (1976), pp. 315–344.
8. Kearfott, R. B.: *Rigorous Global Search: Continuous Problems*, Kluwer Academic Publishers, 1996.
9. Knüppel, O.: *BIAS—Basic Interval Arithmetic Subroutines*, Technical Report 93.3, University of Hamburg, 1993.
10. Mallat, S.: Zero Crossing of a Wavelet Transform, *IEEE Transactions on Information Theory* **17** (4) (1991), pp. 1019–1033.
11. Mitten, L. G.: Branch and Bound Methods: General Formulation and Properties, *Operation Reseach* **18** (1970), pp. 24–34.
12. Moore, R. E.: *Interval Analysis*, Prentice-Hall, 1966.
13. Neumaier, A.: *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.
14. Press, W. H., Flanner, Y. B. P., Teukolsky, S. A., and Vetterling, W. T.: *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, 1986.
15. Ratschek, H. and Rokne, J.: *New Computer Methods for Global Optimization*, Ellis Horwood Lmt. (John Willey & Sons), 1988.
16. Ratz, D.: A Nonsmooth Global Optimization Technique Using Slopes—The One-Dimensional Case, *Journal of Global Optimization*, accepted for publication.
17. Sergeyev, Ya. D., Daponte, P., Grimaldi, D., and Molinaro, A.: Two Methods for Solving Optimization Problems Arising in Electronic Measurement and Electrical Engenieering, *SIAM J. Optimization*, to appear.
18. Zhu, Q., Kam, M., and Yeager, R.: Non-Parametric Identification of QAM Costellations in Noise, in: *IEEE Conference on Acoustic, Speech and Signal Processing*, volume 4, 1993, pp. 184–187.