Topological complexity of zero-finding

Erich Novak
Mathematisches Institut, Universität Erlangen-Nürnberg, Erlangen, Germany

Henryk Woźniakowski
Department of Computer Science, Columbia University, New York, USA
and Institute of Applied Mathematics, University of Warsaw, Warsaw, Poland

January, 1996
Modified: May 4, 1996

Abstract. The topological complexity of zero-finding is studied using a BSS machine over the reals with an information node. The topological complexity depends on the class of functions, the class of arithmetic operations, and on the error criterion.

For the root error criterion the following results are established. If only Hölder operations are permitted as arithmetic operations then the topological complexity is roughly $-\log_2 \varepsilon$ and bisection is optimal. This holds even for the small class of linear functions. On the other hand, for the class of all increasing functions, if we allow the sign function or division together with log and exp, then the topological complexity drops to zero.

For the residual error criterion, results can be totally different than for the root error criterion. For example, the topological complexity can be zero for the residual error criterion, and roughly $-\log_2 \varepsilon$ for the root error criterion.

1. Introduction

Our interest in the topological complexity is motivated by the recent work of Smale (1987) and Vassiliev (1992). In particular, they consider zero-finding for univariate polynomials of degree d with complex coefficients whose absolute value is at most one. They prove that the total number of comparison nodes in the computational graph is independent of an error parameter ε and is roughly d for small ε.

In this paper we deal with zero-finding for univariate functions defined on the interval $[0,1]$, and by the topological complexity we mean the depth of the computational graph.

We analyze different classes F of such functions that are subsets of the class

$$F^* = \{f \in C[0,1] \mid f(0) < 0, \text{ and } f(1) > 0\}. \tag{1}$$

The functions from F^* are equipped with the max norm, $\|f\| = \max_{t\in[0,1]} |f(t)|$.

We consider two definitions of ε-approximation which correspond to the *root* and *residual* error criteria. For the root error criterion, which was used in Smale (1987) and Vassiliev (1992), x is an ε-approximation of a zero of the function f if

$$|x - x^*| \le \varepsilon \quad \text{for some } x^* \text{ such that } f(x^*) = 0.$$

For the *residual* error criterion, x is an ε-approximation of a zero of the function f if

$$|f(x)| \le \varepsilon.$$

We assume that an ε-approximation is computed by a machine (algorithm, program) which uses finitely many information, arithmetic, and comparison operations. That is, we use machines over the reals as introduced by Blum, Shub and Smale (1989) which are additionally equipped with an information node. This information node permits the computation of function evaluations, $f(t)$ for $t \in [0,1]$.

We now comment on arithmetic operations. Usually the standard operations of addition, subtraction, multiplication and division are called arithmetic operations. As in many papers in this area, we understand arithmetic operations in a more general sense. Namely, by an *arithmetic operation* we mean an operation from a given set ARI. We consider various sets ARI. We always assume that

$$\mathrm{ARI}_{\min} = \{+,-,*\},$$

i.e., addition, subtraction and multiplication, is a subset of the set ARI. As we shall see, division plays a special role and the results depend on whether division belongs to the set ARI.

Since we are primarily interested in the topological complexity, we assume that the cost of each comparison is taken as unity, whereas information and arithmetic operations are free of charge. By

$$\mathrm{comp}^{\mathrm{TOP}}(F, \mathrm{ARI}, \varepsilon)$$

we denote the topological complexity, which is defined as the minimal number of comparisons necessary to compute an ε-approximation for any function from the class F by using finitely many information operations (function values), and finitely many arithmetic operations from the set ARI. Details of our model of computation are provided in Section 2. The reader who is not interested in these details may go directly to Section 3.

We prove that the topological complexity depends crucially on the class F of functions, on the choice of arithmetic operations ARI, and on the error criterion. Sometimes apparently innocent changes of F or ARI lead to completely different topological complexities.

Consider first the root error criterion. Then for all F and ARI studied in this paper we have

$$0 \le \mathrm{comp}^{\mathrm{TOP}}(F, \mathrm{ARI}, \varepsilon) \le \lceil -\log_2 \varepsilon - 1 \rceil.$$

Both bounds are sharp. For instance, the topological complexity is $\lceil -\log_2 \varepsilon - 1 \rceil$ for *all* classes F which contain the subclass of linear functions and for the set ARI which consists of all Hölder operations. (By a Hölder operation we mean a mapping defined over a bounded finitely dimensional domain which satisfies the Hölder condition, see Section 2 for details.) In this case, contrary to the case of Smale and Vassiliev, the topological complexity depends on ε and goes to infinity as ε approaches zero.

For the class of linear functions, $f(x) = ax + b$ and $a \ne 0$ the zero $-b/a$ can be computed exactly with one division. This example shows the significance of division. We also show that the inclusion of some other operations, such as signum, which are not Hölder may drastically change the topological complexity. The topological complexity is zero for the class of increasing functions and for the set ARI $= \{+, -, *, /, \exp, \log\}$.

Consider now the residual error criterion. Then the topological complexity is infinite for the class F^*. To obtain a finite topological complexity we must switch to a smaller class. We consider the class of Lipschitz and increasing functions and the set $\mathrm{ARI}_{\min}$. Then the topological complexity is zero. On the other hand, if we drop the assumption that functions are increasing then the topological complexity is positive even for the set ARI of all continuous operations.

We stress that for most classes F of functions and sets ARI studied in this paper, the total complexity, i.e., when all information, arithmetic and comparison operations cost unity, is insensitive to these changes and is always of order $-\log_2 \varepsilon$. Hence, the sensitivity of the topological complexity usually does not correspond to the sensitivity of the total complexity.

2. Real number oracle machines

We assume that the reader is familiar with the concept of a machine P (algorithm, program) over a ring presented by Blum, Shub and Smale (1989). Here we only sketch how computation is performed by such machines and we restrict ourselves only to machines over the reals. Input and output consist of finitely many real numbers. We have arithmetic instructions, a jump instruction, and several copy instructions. We now describe these instructions.

The standard arithmetic operations are the following: addition of a constant, addition of two numbers, multiplication with a constant, multiplication of two numbers, and division of two numbers. Division by 0 is equivalent to a non-terminating computation. The set of standard arithmetic operations without division is denoted by $\mathrm{ARI}_{\min}$ in this paper. Other instructions are sometimes allowed. For example, the evaluation of elementary functions such as sin, cos, exp are sometimes permitted. All such operations are of the form

$$r_k = \mathrm{op}\,(r_{k_1}, r_{k_2}, \dots, r_{k_j})$$

for some indices k, j and $k_1, k_2, \dots, k_j$. Here, $\mathrm{op} : D_{\mathrm{op}} \to \mathbb{R}$ with the domain $D_{\mathrm{op}} \subset \mathbb{R}^j$ for some j. The set of allowed arithmetic operations is denoted by ARI.

The machine P is able to perform backwards and forwards jumps in the list of instructions, depending whether $r_i \le r_j$ or not. We also have the usual copy instructions including indirect addressing, see also Novak (1995). It is clear that many problems of computational mathematics are computable by such machines. Examples include problems which are determined by finitely many parameters and whose solutions may be obtained by performing a finite number of arithmetic operations and comparisons. This holds for problems involving polynomials and matrices.

We now motivate the need of adding an *information* node to the real number machine P. We will do it by an example. Suppose we wish to compute an approximate solution of a nonlinear equation,

$$f(x) = 0,$$

for smooth functions $f : [0,1] \to \mathbb{R}$ from a given class F. We wish to use, for example, the secant method. That is, for given distinct initial approximations x_0 and x_1, we compute, say, k iterative steps,

$$x_{i+1} \;=\; x_i \;-\; \frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})}\, f(x_i), \quad i = 1, 2, \ldots, k.$$

Can we find a machine P which computes x_i? The answer depends on whether we can compute the values of $f(x_i)$ which, in turn, depends on the class F.

Suppose first that F is a class of real polynomials. Then $f \in F$ means that f is a polynomial of degree n, say, and it is fully determined by its coefficients a_i. Hence we can input $n+3$ real numbers, and the input vector $a = (a_0, a_1, \ldots, a_n, x_0, x_1)$ uniquely determines the function and the initiation of the secant method. We say that the input a supplies *complete* information. Knowing a we can compute $f(x_i)$ by, for example, Horner's rule which requires of order n additions and multiplications. Hence, since the problem is determined by finitely many parameters, we have a machine which computes k secant steps. We consider $f(x_i) = f(x_{i-1})$ equivalent to a non-terminating computation.

Suppose now that F is the class of continuous functions f such that $f(0)f(1) < 0$. In this case we cannot compute $f(x_i)$ since f is not necessarily polynomial or rational. The elements of the class F are *not* determined by finitely many parameters. Hence, there exists no BSS-machine for such a class.

How is this problem solved in computational practice? Typically, the user is asked to supply a subroutine which computes $f(x)$ for any x. This subroutine (oracle) is an additional computational device (an information node), and with this new node, secant steps can be carried out, and the sequence x_i can be computed. Observe that, in this case, the information about the function f is restricted to $f(x_i)$ for $i \leq k$. This information, which does not determine the function f uniquely, is *partial*.

We extend the concept of a machine over the reals to permit the use of *information* operations. These operations will supply partial information about the computational problem, and this partial information will be used to compute an approximate solution.

An *oracle* machine over the reals is a machine over the reals plus an information node. We extend the list of instructions by information operations. In this paper we use only the class

$$\mathrm{INFO}_{\mathrm{std}} = \{L : F \to \mathbb{R} \mid \text{there exists } t \in [0,1] \text{ such that } L(f) = f(t),\ \forall\, f \in F\},$$

that is, the information operation is of the form

$$r_k := I(x, f) = f(x)$$

for $f \in F^*$ and any $x \in [0,1]$.

The cost of computation by the machine P can be simply defined by counting how many operations of various types are performed by P. In computational mathematics, one usually counts the number of arithmetic operations, the number of comparisons, and the number of information operations (oracle calls). The cost of input, output as well as copy instructions is usually neglected. For simplicity, we also assume that input, output and copy instructions cost zero, although it is obvious how to proceed without this assumption. Assume that $\mathbf{c}_{\mathrm{AR}}$ denotes the cost of performing one arithmetic operation, whereas $\mathbf{c}_{\mathrm{BR}}$ denotes the cost of one comparison, i.e., the cost of performing one comparison of two real numbers. We assume that $\mathbf{c}_{\mathrm{IN}}$ is the cost of one information operation. Suppose that P terminates on input (a, f). Let the computation of $P(a, f)$ require n_{AR} arithmetic operations, n_{BR} comparisons, and n_{IN} information operations. The cost of computing $P(a, f)$ is

$$\mathrm{cost}\,(P, a, f) \;=\; \mathbf{c}_{\mathrm{AR}}\, n_{\mathrm{AR}} + \mathbf{c}_{\mathrm{BR}} n_{\mathrm{BR}} + \mathbf{c}_{\mathrm{IN}}\, n_{\mathrm{IN}}.$$

If P does not terminate on (a, f) then we formally set $\mathrm{cost}\,(P, a, f) \;=\; +\infty$. Sometimes we call $\mathbf{c}_{\mathrm{IN}} n_{\mathrm{IN}}$ the information cost and $\mathbf{c}_{\mathrm{AR}} n_{\mathrm{AR}} + \mathbf{c}_{\mathrm{BR}} n_{\mathrm{BR}}$ the computational cost.

Here we assumed, for simplicity, that the cost of all arithmetic operations from the set ARI is the same and equal to $\mathbf{c}_{\mathrm{AR}}$. Of course, this could be easily modified. Usually, it is much more expensive to compute an information operation than any other operation. For some practical problems, computation of $L(f) = f(t)$ may take hours even on a modern computer whereas arithmetic operations or comparisons can be done in a fraction of a second. That is the reason why we use different parameters for the cost of permissible operations. We also want to see how arithmetic, comparison and information operations affect the complexity of a problem. The global cost of P over F in the worst case setting is defined as

$$\mathrm{cost}(P, a) \;=\; \sup_{f \in F} \mathrm{cost}\,(P, a, f)\,.$$

Many problems of computational mathematics can be studied within the model of computation given by oracle machines over the reals. For a given problem we may usually find many different machines for its solution. Obviously, we would like to choose a machine that solves the problem with minimal cost. This naturally leads to the concept of complexity. Let F as well as the sets ARI and INFO of permissible arithmetic and information operations be fixed. Consider a map

$$S : F \to \mathrm{I\!R}.$$

We wish to compute an ε-approximation to $S(f)$ for elements $f \in F$. An *ε-approximation* is defined as an element $U(f)$ with *error* at most ε. Here we assume that $U : F \to \mathrm{I\!R}$ and that an error criterion is given, for example, by

$$e(U, f) = |S(f) - U(f)|.$$

The definition of the error of U depends on the setting. The worst case error of U is

$$e(U) = \sup_{f \in F} e(U, f).$$

We now discuss the computation of an ε-approximation $U(f)$. Clearly, to compute $U(f)$ means to find an oracle machine P for which U is a computable function. Observe that U

must depend on the error parameter ε, as well as on the problem parameters F, ARI, INFO, S, and the error criterion. By PAR we denote these problem parameters.

Assume that all the parameters are fixed. That is, the only input to the oracle machine P is f, and $P(f)$ is the output of computation. This means that P has all parameters ε and PAR built-in, $P = P_{\varepsilon,\mathrm{PAR}}$. The (total) *$\varepsilon$-complexity* is defined as the minimal cost of such oracle machines which solve the problem,

$$\mathrm{comp}(\varepsilon, \mathrm{PAR}) \;=\; \inf\{\mathrm{cost}(P): \;\; P = P_{\varepsilon,\mathrm{PAR}} \text{ is an } \varepsilon\text{-approximation}\}.$$

By convention we set $\inf \emptyset \;=\; +\infty$. That is, if there exists no oracle machine $P = P_{\varepsilon,\mathrm{PAR}}$ which is an ε-approximation then $\mathrm{comp}(\varepsilon, \mathrm{PAR}) = +\infty$. We stress that this is a "nonuniform" definition. The parameters ε, PAR determine the problem but are not part of the input. This nonuniform case is usually studied in information-based complexity, see Traub, Wasilkowski, Woźniakowski (1988). It is an interesting question whether, for a given problem with input parameters $a = (\varepsilon, \ldots)$, there exists an oracle machine (algorithm) P such that the cost of computing $P(a, f)$ is not much larger than the ε-complexity, uniformly for all a. See Novak (1995) for some examples.

This defines complexity in the worst case setting. Complexity depends, in particular, on $\mathbf{c}_{\mathrm{AR}}, \mathbf{c}_{\mathrm{BR}}, \mathbf{c}_{\mathrm{IN}}$ which are the cost of one arithmetic, comparison, and information operation. Since complexity minimizes the total cost, it minimizes the number of arithmetic, comparison and information operations weighted through $\mathbf{c}_{\mathrm{AR}}, \mathbf{c}_{\mathrm{BR}}$ and $\mathbf{c}_{\mathrm{IN}}$. For example, if $\mathbf{c}_{\mathrm{IN}}$ is much larger than $\mathbf{c}_{\mathrm{AR}}$ and $\mathbf{c}_{\mathrm{BR}}$ then we should perform less information operations at the expense of additional arithmetic and comparison operations. On the other hand, if $\mathbf{c}_{\mathrm{BR}}$ is much larger than $\mathbf{c}_{\mathrm{AR}}$ and $\mathbf{c}_{\mathrm{IN}}$ then we should eliminate as many comparisons as possible. In any case, depending on the values of $\mathbf{c}_{\mathrm{AR}}, \mathbf{c}_{\mathrm{BR}}$ and $\mathbf{c}_{\mathrm{IN}}$, we may perform a different number of arithmetic, comparison and information operations.

It is interesting to study how many operations of different types are necessary independently of the other operations. This can be achieved by taking two of $\mathbf{c}_{\mathrm{AR}}, \mathbf{c}_{\mathrm{BR}}, \mathbf{c}_{\mathrm{IN}}$ equal to zero and the third one equal to one. In this way we obtain different complexities. More precisely, we obtain:
the *arithmetic* complexity,

$$\mathrm{comp}^{\mathrm{ARI}}(\varepsilon, \mathrm{PAR}) \;=\; \mathrm{comp}(\varepsilon, \mathrm{PAR}) \quad \text{for } \mathbf{c}_{\mathrm{AR}} = 1,\ \mathbf{c}_{\mathrm{BR}} = 0,\ \mathbf{c}_{\mathrm{IN}} = 0,$$

the *topological* complexity,

$$\mathrm{comp}^{\mathrm{TOP}}(\varepsilon, \mathrm{PAR}) \;=\; \mathrm{comp}(\varepsilon, \mathrm{PAR}) \quad \text{for } \mathbf{c}_{\mathrm{AR}} = 0,\ \mathbf{c}_{\mathrm{BR}} = 1,\ \mathbf{c}_{\mathrm{IN}} = 0,$$

and the *information* complexity

$$\mathrm{comp}^{\mathrm{INFO}}(\varepsilon, \mathrm{PAR}) \;=\; \mathrm{comp}(\varepsilon, \mathrm{PAR}) \quad \text{for } \mathbf{c}_{\mathrm{AR}} = 0,\ \mathbf{c}_{\mathrm{BR}} = 0,\ \mathbf{c}_{\mathrm{IN}} = 1.$$

Clearly, the (total) complexity $\mathrm{comp}(\varepsilon, \mathrm{PAR})$ is bounded from below by

$$\mathrm{comp}(\varepsilon, \mathrm{PAR}) \geq \mathbf{c}_{\mathrm{AR}}\, \mathrm{comp}^{\mathrm{ARI}}(\varepsilon, \mathrm{PAR}) + \mathbf{c}_{\mathrm{BR}}\, \mathrm{comp}^{\mathrm{TOP}}(\varepsilon, \mathrm{PAR}) + \mathbf{c}_{\mathrm{IN}}\, \mathrm{comp}^{\mathrm{INFO}}(\varepsilon, \mathrm{PAR}).$$

It may happen that the total complexity goes to infinity as ε goes to zero, whereas one or even two of the other complexities are bounded or even zero.

Observe that we define the topological complexity here as the depth of the computational graph which corresponds to the maximal number of comparisons used in one particular computation. The total number of comparison nodes in the computation graph can be much bigger. This latter quantity is also called topological complexity, see Smale (1987).

This concludes the definition of the real number oracle machines, the model of computation, and various complexities in this model. We hope to pursue many aspects of real number oracle machines in the near future.

3. Root Error Criterion

In this section we study computation of an ε-approximation for the root error criterion. We assume that information is given by function values and consider different subsets of F^*, see (1), and various sets ARI.

F^* is the class of functions for which the well known bisection algorithm (machine) works. For $f \in F^*$ and a positive $\varepsilon \le 1/2$, the bisection algorithm $\mathrm{BIS}_\varepsilon(f)$ is given by:

$$
\begin{aligned}
& && k := \lceil -\log_2 \varepsilon - 1 \rceil, \\
&\text{INPUT:} && f \\
&\text{COMPUTE:} && a := 0,\ b := 1, \\
& && x := 0.5, \\
&\text{COMPUTE } k \text{ times:} && y = f(x), \\
& && \text{if } y \le 0 \text{ then } a := x \text{ else } b := x, \\
& && x := 0.5 * (a + b), \\
&\text{OUTPUT:} && \mathrm{BIS}_\varepsilon(f) := x.
\end{aligned}
$$

Clearly, x computed by bisection is an ε-approximation in the root error sense. Observe that k can be precomputed, i.e., it is a built-in parameter of the bisection machine BIS_ε. The bisection algorithm uses $k = \lceil -\log_2 \varepsilon - 1 \rceil$ function values and comparisons. The number of additions and multiplications is $2k$.

It is known, see Kung (1976), that at least k function values must be used by any algorithm which computes an ε-approximation for all functions from F^*. Hence, bisection is optimal with respect to the number of function values. Optimality of bisection is preserved also for subsets of F^* which consist of smooth functions, see Sikorski (1985). On the other hand, bisection is *not* optimal in an average case setting (with a Brownian bridge on smooth functions as the probability measure), see Novak, Ritter and Woźniakowski (1995).

Since $F \subset F^*$ and $\mathrm{ARI}_{\min} \subset \mathrm{ARI}$, the number of comparisons used by bisection provides an upper bound on the topological complexity, $\varepsilon \le 1/2$,

$$
(2) \qquad \mathrm{comp}^{\mathrm{TOP}}(F, \mathrm{ARI}, \varepsilon) \le \mathrm{comp}^{\mathrm{TOP}}(F^*, \mathrm{ARI}_{\min}, \varepsilon) \le \lceil -\log_2 \varepsilon - 1 \rceil.
$$

Is this bound sharp, or equivalently, is bisection also optimal with respect to the number of comparisons? The answer depends on F and ARI. The bound is sharp even for small subsets

F of F^* if we only allow arithmetic operations that are Hölder on bounded domains. More precisely, let

$$\mathrm{F_{lin}} \;=\; \{ f \in F^* :\; f(x) = ax + b, \forall\, x \in [0,1], \text{ for some } a, b \;\}.$$

That is, $\mathrm{F_{lin}}$ is a subclass of the class of linear functions. Observe that $f \in F^*$ implies that $f(0) = b < 0$ and $f(1) = a + b > 0$. This implies $a > 0$, however, $|a|$ can be arbitrarily small. The only zero of f is $x^*(f) = -b/a$. If the set ARI contains division then the exact solution may be computed without comparisons, and therefore the topological complexity is zero.

On the other hand, if division is *not* in ARI then, as we shall see, comparisons are needed and the bound (2) is sharp. This holds for the set $\mathrm{ARI_{hol}}$ which is defined as the set of all operations that are Hölder on bounded domains. That is, the operation op belongs to $\mathrm{ARI_{hol}}$ iff $\mathrm{op} : D_{\mathrm{op}} \to \mathbb{R}$, where the domain $D_{\mathrm{op}} \subset \mathbb{R}^j$ for some j, and for any bounded set $M \subset D_{\mathrm{op}}$ there are constants $\beta, \gamma > 0$ such that

$$|\mathrm{op}(x) - \mathrm{op}(y)| \le \gamma \cdot \|x - y\|^{\beta} \qquad \forall\, x, y \in M. \tag{3}$$

The set $\mathrm{ARI_{hol}}$ is very large. It obviously contains $\mathrm{ARI_{min}}$ as well as many of the standard functions. Note that the operation $\mathrm{op}(x,y) = x/y$ with the domain $D_{\mathrm{op}} \subset \mathbb{R}^2$ belongs to $\mathrm{ARI_{hol}}$ if there exists a positive δ such that the domain D_{op} satisfies the condition

$$(x,y) \,\in\, D_{\mathrm{op}} \quad \text{implies} \quad |y| \,\ge\, \delta.$$

Of course, the division $\mathrm{op}(x,y) = x/y$ with the domain $D_{\mathrm{op}} = \{(x,y) :\; y \neq 0\}$ does *not* belong to $\mathrm{ARI_{hol}}$. Similarly, the natural logarithm $\mathrm{op}(x) = \log(x)$ with the domain $D_{\mathrm{op}} = \{x :\; x > \delta\}$ belongs to $\mathrm{ARI_{hol}}$ if $\delta > 0$, but not for $\delta = 0$.

Proposition 1. If $\varepsilon \le 1/2$ then

$$\mathrm{comp}^{\mathrm{TOP}}(\mathrm{F_{lin}}, \mathrm{ARI_{hol}}, \varepsilon) \;=\; \lceil -\log_2 \varepsilon \,-\, 1 \rceil.$$

We need the following lemma.

Lemma 1. Let A be a bounded subset of $\mathrm{F_{lin}}$ and let $\mathrm{ARI} = \mathrm{ARI_{hol}}$. Assume that P is a machine which terminates for each f from A. Assume further that P does not use comparisons. Then the map $f \mapsto P(f)$, $f \in A$, is Hölder.

Proof: The machine P uses finitely many function values $f(t_i)$ where $f \in A$ and $t_i \in [0,1]$. Since A is bounded, all function values $f(t_i)$ are also bounded. The machine P uses finitely many arithmetic operations from $\mathrm{ARI_{hol}}$ which are all Hölder. Hence, P can be written as a composition of mappings that are Hölder over bounded domains. Therefore P is Hölder over A.

Proof of Proposition 1: Since bisection provides the needed upper bound on the number of comparisons, we only need to prove a matching lower bound. We clearly have

$$\{x \in [0,1] \mid\; f(x) = 0 \text{ for some } f \in \mathrm{F_{lin}}\} = (0,1).$$

Take an arbitrary $\delta \in (0, 1/2)$. For an arbitrary $x^* \in (0,1)$ define $f(x) = \delta(x - x^*)$. Then f belongs to $\mathrm{F}_{\mathrm{lin}}$ and

$$f(x^*) = 0 \quad \text{and} \quad \|f\| < \delta. \tag{4}$$

Due to (4), the zero-finding problem is unstable on $\mathrm{F}_{\mathrm{lin}}$, i.e., the map $f \mapsto x^*(f)$, where $x^*(f)$ is the zero of f, is not Hölder.

Assume that P is any machine with output $P(f)$ which is an ε-approximation of the zero of f for any $f \in \mathrm{F}_{\mathrm{lin}}$. We assume that P uses, for each $f \in \mathrm{F}_{\mathrm{lin}}$, at most n comparisons. Hence we can decompose $\mathrm{F}_{\mathrm{lin}}$,

$$\mathrm{F}_{\mathrm{lin}} = F^1 \cup F^2 \cup \cdots \cup F^{2^n},$$

in such a way that the machine P restricted to F^k does not use comparisons. Consider the set

$$F^\delta = \{f \in \mathrm{F}_{\mathrm{lin}} \mid \|f\| < \delta\}.$$

Applying Lemma 1 we conclude that $P : F^k \cap F^\delta \to \mathbb{R}$ is Hölder for each k. We denote the respective Hölder constant by γ_k and the exponent by β_k, see (3). Let $\gamma = \max_k \gamma_k < +\infty$ and $\beta = \min_k \beta_k > 0$. From (4) we have

$$\{x^*(f) \mid f \in F^\delta\} = (0,1).$$

For $f_1, f_2 \in F^k \cap F^\delta$ we have $\|f_1 - f_2\| \le 2\delta$, and hence

$$|P(f_1) - P(f_2)| \le \gamma \cdot (2\delta)^\beta.$$

This means that $P(F^\delta)$ is contained in the union of 2^n intervals of length $l = \gamma \cdot (2\delta)^\beta$. Since $\delta > 0$ is arbitrary, we can make l arbitrarily small. As a result we obtain that the error of P on the class $\mathrm{F}_{\mathrm{lin}} \cap F^\delta$ is at least $2^{-1}\,2^{-n}$. But the error of P is at most ε, and therefore the number of comparisons n is at least $\lceil -\log_2 \varepsilon - 1 \rceil$. This completes the proof.

Remark 1. Proposition 1 can be also formulated as follows. Assume that we want to approximate $-b/a$ for pairs $a, b \in \mathbb{R}$ such that $b < 0$ and $a + b > 0$. If we allow Hölder operations, i.e., no division, then comparisons are necessary and bisection is optimal. Hence, one division can be avoided at the expense of $\lceil -\log_2 \varepsilon - 1 \rceil$ comparisons to compute an ε-approximation of $-b/a$. This result can be compared with other results in complexity theory. For example, Strassen (1973) proved that for the evaluation of polynomials division can be avoided.

We stress that the assumptions $b < 0$ and $a + b > 0$ are essential. Indeed, let $b = -1$, and drop the assumption that $a - 1 > 0$. Suppose we want to approximate $1/a$ for, say, $a \in (0, 1]$ by using operations from $\mathrm{ARI}_{\mathrm{hol}}$. Let P_ε be an arbitrary machine which uses finitely many arithmetic operations from $\mathrm{ARI}_{\mathrm{hol}}$ and finitely many comparisons. That is, there exists a finite $k = k(\varepsilon)$ such that for all $a \in (0, 1]$ we perform at most k Hölder operations and comparisons to compute $P_\varepsilon(a)$. Clearly, $P_\varepsilon((0,1])$ is bounded. Since the mapping $a \mapsto 1/a$ is unbounded, we have

$$\sup_{a\in(0,1]} |1/a - P_\varepsilon(a)| \;=\; +\infty.$$

That is, we cannot compute an ε-approximation even for arbitrarily large ε. Similarly, if we switch to the relative error, we have

$$\sup_{a\in(0,1]} |1/a - P_\varepsilon(a)|/(1/a) \;\geq\; 1.$$

Hence, for the relative error, we cannot compute an ε-approximation for $\varepsilon < 1$. If, however, we restrict the domain of a to, say, $[\delta,1]$ for a positive δ, then the mapping $a \mapsto 1/a$ becomes Hölder, and $1/a$ can be approximated by arithmetic operations from $\mathrm{ARI}_{\min}$. In fact, $1/a$ can be approximated by Newton iteration $x_{i+1} = x_i(2 - ax_i)$ applied to the nonlinear equation $1/x - a = 0$ with the initial guess $x_0 = 1$. Then

$$k \;=\; 3\,\max\left\{0\,,\;\left\lceil \log_2\left(\log_2\frac{1}{\varepsilon} \,+\, \log_2\frac{\gamma}{\delta}\right) \;-\; \log_2\,\log_2\frac{1}{1-\delta}\right\rceil\right\}$$

multiplications and subtractions are enough to compute an ε-approximation with $\gamma = \delta$ for the relative error and $\gamma = 1$ otherwise. Note, however, that $k \to +\infty$ with $\delta \to 0$. In any case, comparisons are now not needed, and the topological complexity of this problem is zero.

We finally remark that in floating point arithmetic the division $1/a$ is computed by using the floating point representation of $a = m\,2^c$ where $1/2 \leq m < 1$ and c is an integer. Then $1/a = (1/m)\,2^{-c}$ and it is enough to compute $1/m$. This is usually done by Newton iteration as above with $\delta = 1/2$. We stress that the use of the floating point representation means that we have an access to the exponent part c of the number a. This, in turn, corresponds to the operation $\lceil \log_2 a \rceil$. Obviously, taking ceilings is a discontinuous operation, and both operations of taking ceilings and logarithms do not belong to $\mathrm{ARI}_{\mathrm{hol}}$.

We now show that several assumptions of Proposition 1 are essential. First we replace the set $\mathrm{ARI}_{\mathrm{hol}}$ by $\mathrm{ARI}_{\mathrm{sgn}} = \mathrm{ARI}_{\min} \cup \{\mathrm{sgn}\}$, where $\mathrm{sgn}(y) = 1$ for $y > 0$, $\mathrm{sgn}(0) = 0$, and $\mathrm{sgn}(y) = -1$ for $y < 0$. That is, we can now compute signs of the function values, $\mathrm{sgn}(f(x))$ for $x \in [0,1]$. How many comparisons do we now need? It turns out that no comparisons are needed. Indeed, it is enough to modify the bisection algorithm as follows.

$$\begin{array}{rl}
 & k \;:=\; \lceil -\log_2\,\varepsilon \;-\; 1\rceil, \\
\text{INPUT:} & f \\
\text{COMPUTE:} & a := 0,\;\; b := 1, \\
 & x := 0.5, \\
\text{COMPUTE } k \text{ times:} & y \;=\; \mathrm{sgn}(f(x)), \\
 & a := a \,+\, 0.25 * (1-y) * (b-a) \\
 & b := b \,-\, 0.25 * (1+y) * (b-a) \\
 & x := 0.5 * (a+b), \\
\text{OUTPUT:} & \mathrm{BIS}_\varepsilon(f) := x.
\end{array}$$

It is easy to check that this algorithm computes the same results as bisection. For instance, assume that $f(x) > 0$. Then $y = 1$ and the new a is equal to the old a whereas the new b is moved to the middle point of the old interval $[a, b]$ exactly as in the original bisection algorithm.

Since no comparison is used in this modification and we compute an ε-approximation, the topological complexity is zero. Observe that the total cost of the modified algorithm is comparable to the cost of bisection. In fact, the number of information operations is optimal and equals $\lceil -\log_2 \varepsilon - 1 \rceil$ whereas the number of arithmetic operations is still of order $-\log_2 \varepsilon$.

Hence, the signum function makes the topological complexity zero. Obviously, the signum function is discontinuous. The use of this particular discontinuous operation allows us to eliminate comparisons.

How about continuous operations which are *not* Hölder? Once more, Proposition 1 is not true for the class $\mathrm{F_{lin}}$ and the set ARI of continuous operations since division is continuous. But what will happen if we enlarge the class $\mathrm{F_{lin}}$? Again, Proposition 1 does not hold for the class of polynomials of bounded degree. This follows from the result of Renegar (1987) who proved that bisection is far from being optimal for polynomials of bounded degree assuming that the four standard arithmetic operations are used, i.e., $\mathrm{ARI_{min}}$ and division.

We now show that Proposition 1 does not hold and, in fact, the topological complexity is zero for the class of strictly increasing functions,

$$\mathrm{F_{inc}} \; = \; \{ f \in F^* | \; f \text{ is strictly increasing } \}$$

and for the set $\mathrm{ARI_{ext}}$ of the four standard arithmetic operations, the exponential function $\exp(x), \forall\, x \in \mathbb{R}$, and the natural logarithm $\log(x), \forall x > 0$,

$$\mathrm{ARI_{ext}} \; = \; \mathrm{ARI_{min}} \cup \{ \, / \, , \, \exp \, , \, \log \, \}.$$

Observe that all operations in $\mathrm{ARI_{ext}}$ are continuous.

Proposition 2. For every $\varepsilon > 0$,

$$\mathrm{comp}^{\mathrm{TOP}}(\mathrm{F_{inc}}, \mathrm{ARI_{ext}}, \varepsilon) \; = \; 0.$$

Proof: We present an algorithm (machine) P_ε which computes an ε-approximation and which does not use comparisons. In the computational nodes, the algorithm P_ε uses function values and operations from $\mathrm{ARI_{ext}}$. Let

$$n \; = \; \lceil 1 + 9/(4\varepsilon) \rceil .$$

The idea of the algorithm is to compute a suitable weighted mean of the equally spaced $x_i = (i-1)/(n-1)$, $i = 1, 2, \ldots, n$. The weights depend on the function values $f(x_i)$ such that the weighted mean approximates the zero of the function f. Hence, the weights are used now instead of comparisons to localize the position of the zero. It is convenient to assume that

$$f(0)^2 + f(1)^2 = 1. \tag{5}$$

This can be done without loss of generality since we can replace, in what follows, the $f(x_i)$ by $y_i = f(x_i)/(f(x_1)^2 + f(x_n)^2)$. The algorithm $P_\varepsilon(f)$ is defined as follows:

Step 1: Compute $f(x_1), \ldots, f(x_n)$
Step 2: Compute the numbers

$$d_{jk} := (f(x_j) - f(x_k))^2 > 0$$

for all $j, k = 1, \ldots, n$ with $j \neq k$.

It is important that each d_{jk} is *strictly* positive. This holds since $f \in F_{\text{inc}}$. We will see later that Proposition 2 does not hold for the slightly larger class of nondecreasing functions.

Step 3: Compute

$$w_{ijk} := \left(\frac{1}{f(x_i)^2 + d_{jk}}\right)^{2n^4/d_{jk}}$$

for $j \neq k$, and

$$w_i := \sum_{j \neq k} w_{ijk},$$

and finally,

$$P_\varepsilon(f) := \frac{\sum_{i=1}^n w_i x_i}{\sum_{i=1}^n w_i}.$$

We first note that all quantities are well defined since $d_{jk} > 0$ and division by 0 does not occur in the definition of w_{ijk}. Since we use ARI_{ext}, the computation of powers is possible using log and exp, i.e., by using $x^y = \exp(y \log(x))$. We stress that no comparison is used during the computation of $P_\varepsilon(f)$.

We now show that $P_\varepsilon(f)$ is an ε-approximation of the zero $x^* = x^*(f)$ for f from F_{inc}. There is an index i^* such that $x^* \in [x_{i^*}, x_{i^*+1}]$. Let $J = \{1, 2, \ldots, n\} \setminus \{i^*, i^*+1\}$ and

$$\tilde{y} := \min_{i=1,\ldots,n} |f(x_i)| \qquad \text{and} \qquad d := \min_{j \neq k} d_{jk}.$$

Since f is increasing (5) yields $\tilde{y} \leq 1$ and, for $n > 2$, also $d \leq 1$.

We know that $|f(x_{i^*})| = \tilde{y}$ or $|f(x_{i^*+1})| = \tilde{y}$. In the following we only discuss the first case, the second one is analogous. Hence we have

$$|f(x_{i^*})| = \tilde{y} \qquad \text{and} \qquad |f(x_i)| \geq \tilde{y} + d \quad \text{for all } i \in J.$$

There are distinct indices j, k such that

$$w := w_{i^*jk} = \left(\frac{1}{\tilde{y}^2 + d}\right)^{2n^4/d},$$

while for $i \in J$ and all pairs j, k, $j \neq k$,

$$w_{ijk} \leq \left(\frac{1}{(\tilde{y}+d)^2 + d} \right)^{2n^4/d}.$$

Hence we obtain

$$\frac{w}{w_{ijk}} \geq \left(\frac{(\tilde{y}+d)^2 + d}{\tilde{y}^2 + d} \right)^{2n^4/d}.$$

Using Bernoulli's inequality and the fact that $\tilde{y} \in [0,1]$, we obtain

$$\frac{w}{w_{ijk}} \geq 1 + \left(\frac{2\tilde{y}d + d^2}{\tilde{y}^2 + d} \right) \frac{2n^4}{d} = 1 + \frac{2\tilde{y} + d}{\tilde{y}^2 + d} 2n^4 \geq 2n^4.$$

Hence,

$$\frac{w}{w_{ijk}} \geq 2n^4,$$

again for all $i \in J$ and $j \neq k$. Every total weight w_i is the sum of $n(n-1)/2$ weights w_{ijk} and therefore

$$\frac{w_{i^*}}{w_i} \geq \frac{w}{w_i} \geq 4n^2, \quad \forall\, i \in J.$$

To estimate $|P(f,n) - x^*|$ we note that $|w_i x_i - w_i x^*| \leq w_i$ for all i, and $|w_i x_i - w_i x^*| \leq w_i/(n-1)$ for $i = i^*$ and $i = i^* + 1$. Since

$$w_i \leq c/(4n^2), \quad \text{for } c = \max\{w_{i^*}, w_{i^*+1}\},$$

we obtain

$$|P(f,n) - x^*| \leq \frac{\sum_{i=1}^n |w_i x_i - w_i x^*|}{\sum_{i=1}^n w_i} \leq \frac{c(n-2)/(4n^2) + 2c/(n-1)}{c} \leq \frac{9}{4(n-1)}.$$

Hence the described algorithm has a worst case error bounded by $9/(4n-4) \leq \varepsilon$. This completes the proof.

Comparing Propositions 1 and 2 we see that Proposition 1 holds for all subsets of F^* that contain the class $\mathrm{F}_{\mathrm{lin}}$ of linear functions, while Proposition 2 is only proved for the class $\mathrm{F}_{\mathrm{inc}}$ of strictly increasing functions. Actually, Proposition 2 is not true for the slightly larger class F_{nd} of nondecreasing functions,

$$\mathrm{F}_{\mathrm{nd}} = \{ f \in F^* \mid f \text{ is nondecreasing}, f(0) = -1, f(1) = 1 \}.$$

Proposition 3. If $\varepsilon < 1/2$ then

$$\mathrm{comp}^{\mathrm{TOP}}(\mathrm{F}_{\mathrm{nd}}, \mathrm{ARI}_{\mathrm{ext}}, \varepsilon) > 0.$$

Proof: The proof is similar to the proofs of Lemma 1 and Proposition 1. Observe that F_{nd} is bounded and closed. Let P be an arbitrary machine which terminates for all f from F_{nd} and which does not use comparisons. We show that P cannot compute an ε-approximation. As in Lemma 1, we show that the mapping $f \mapsto P(f)$ is Hölder on a subset of F_{nd} of functions that fulfill a Hölder condition. This is true even though we may use divisions and logarithms. Indeed, for any division, say $r_j = r_i/r_k$, consider the set M of possible values r_k. The set M is closed since F_{nd} is closed. The set M cannot contain 0 since $r_k = 0$ would cause the termination of computation and an ε-approximation would not be computed. Hence, the set M is bounded away from zero, and consequently division (and similarly logarithm) over this domain is Hölder. Applying now the proof of Proposition 1 (with $n = 0$), we conclude that the error of P is at least $1/2$ which is larger than ε.

Remark 3. Hertling (1996) in his recent paper showed the power of the set of arithmetic operations

$$\mathrm{ARI}_{\mathrm{abs}} \;=\; \{+,-,*,/,|\cdot|\}.$$

That is, we permit the four standard arithmetic operations and taking the absolute value. Then even for the class F^* we have

$$\mathrm{comp}^{\mathrm{TOP}}(F^*, \mathrm{ARI}_{\mathrm{abs}}, \varepsilon) \;=\; \left\lceil \log_2 \left(\left\lceil \log_2(\varepsilon^{-1}+2) \right\rceil \;-\; 1 \right) \right\rceil .$$

Hence, the topological complexity is roughly $\log_2 \log_2 \varepsilon^{-1}$ which is exponentially smaller than $\log_2 \varepsilon^{-1}$. This also shows that bisection is not optimal for the set $\mathrm{ARI}_{\mathrm{abs}}$.

Hertling (1996) also proved that adding additional continuous operations to the set $\mathrm{ARI}_{\mathrm{abs}}$ does not help. Namely, if $\mathrm{ARI}_{\mathrm{con}}$ denotes the set of all continuous operations then

$$\mathrm{comp}^{\mathrm{TOP}}(F^*, \mathrm{ARI}_{\mathrm{con}}, \varepsilon) \;=\; \mathrm{comp}^{\mathrm{TOP}}(F^*, \mathrm{ARI}_{\mathrm{abs}}, \varepsilon).$$

4. Residual Error Criterion

In this section we consider an ε-approximation in the residual error sense, i.e., for all f from F we want to compute x such that

$$|f(x)| \le \varepsilon.$$

First of all, observe that for the class F^* we cannot compute an ε-approximation even for large ε. Indeed, the class F^* contains functions with arbitrarily large values, and it can happen that for all points x_i for which the machine P computes $f(x_i)$ we get, for example, $f(x_i) = 1+\varepsilon$ if $x_i > 1/2$, and $f(x_i) = -1-\varepsilon$ if $x_i \le 1/2$. If x is an output of P then one can find a function f^* from F^* which is consistent with the computed values, $f^*(x_i) = f(x_i)$ and for which $|f^*(x)| > \varepsilon$. Since the output of P for the function f^* must be the same as for f, the machine P fails to compute an ε-approximation for f^*.

That is why we need to switch to a smaller class of functions to guarantee that the topological complexity is finite for the residual error criterion. We discuss the classes $\mathrm{F}_{\mathrm{Lip}}$ of Lipschitz functions, and $\mathrm{F}_{\mathrm{Linc}}$ of Lipschitz and increasing functions,

$$\begin{aligned} \mathrm{F}_{\mathrm{Lip}} &= \{ f \in F^* \mid f \text{ is Lipschitz with constant } L \}, \\ \mathrm{F}_{\mathrm{Linc}} &= \mathrm{F}_{\mathrm{Lip}} \cap \mathrm{F}_{\mathrm{inc}}. \end{aligned}$$

We prove that the topological complexity is zero for the class $\mathrm{F_{Linc}}$ even for the small set $\mathrm{ARI_{min}}$. However, for the class $\mathrm{F_{Lip}}$ and the set ARI of all continuous operations, the topological complexity is positive. Hence, we can approximate the zeros of Lipschitz and increasing functions from the class $\mathrm{F_{Linc}}$ without comparisons. However, this is not possible without assuming monotonicity of functions.

Proposition 4. For every $\varepsilon > 0$,

$$\mathrm{comp}^{\mathrm{TOP}}(\mathrm{F_{Linc}}, \mathrm{ARI_{min}}, \varepsilon) \;=\; 0.$$

Proof: We present a machine which computes an ε-approximation and uses only finitely many function values and arithmetic operations from $\mathrm{ARI_{min}}$. Contrary to bisection, the machine computes function values at points dependent only on ε, i.e., it uses nonadaptive information. Note that $f \in \mathrm{F_{Linc}}$ implies that $f(x) \in [-L, L]$, $\forall\, x \in [0,1]$.

In what follows, we consider two parameters: a positive δ and an integer n which will be then defined in terms of ε.

For any positive δ, (δ should be small and in any case $\delta < \min\{1, L\}$), we choose a polynomial p_δ whose graph over $[-L, L]$ resembles the graph of the sign function. More precisely we require that p_δ has the following properties:

$$\begin{array}{lll} a) & |p_\delta(x)| \le \;\; 1 & \text{on } [-L, L]; \\ b) & p_\delta(x) \le -1 + \delta & \text{on } [-L, -\delta]; \\ c) & p_\delta(x) \ge \;\; 1 - \delta & \text{on } [\delta, L]. \end{array}$$

For an integer n, consider the nonadaptive knots

$$x_i = \frac{i}{n+1}, \qquad i = 1, \dots, n.$$

Define

$$x \;=\; x_{n,\delta}(f) = \frac{n - \sum_{i=1}^{n} p_\delta(f(x_i))}{2n}. \tag{6}$$

Clearly, x belongs to $[0,1]$. We now show that $|f(x)|$ is small for small δ and large n.

Since f is increasing, there exists an integer $k \in [1, n]$ such that $f(x_i) < -\delta$ for $i = 1, \dots k$, and $f(x_i) \ge -\delta$ for the other i. Then one can easily check that

$$x \ge x_k^* \;:=\; \frac{k - k\delta/2}{n}.$$

Clearly, $f(x) \ge f(x_k^*)$ and $x_k^* < x_{k+1}$. Since $f(x_{k+1}) \ge -\delta$ and f is Lipschitz we obtain

$$f(x) \;\ge\; -\delta - L \cdot \left(\frac{k+1}{n+1} - \frac{k}{n} + \frac{k\delta}{2n} \right) \;\ge\; -\frac{\delta(2+L)}{2} \;-\; \frac{L}{n+1}.$$

To get an upper bound on $f(x)$ we proceed similarly and obtain

$$|f(x)| \le \frac{\delta(L+2)}{2} + \frac{L}{n+1}.$$

Hence, for $\delta \le \varepsilon/(L+2)$ and $n \ge \lceil 2L/\varepsilon - 1 \rceil$, the point x is an ε-approximation. Thus, we can define the output of the machine $P(f) = x$ with x given by (6). Observe that for given $L > 0$ and $\varepsilon > 0$ the values of δ, n, m, $1/(n+1)$, $1/(2n)$ as well as the coefficients of the polynomial p_δ can be precomputed, i.e., they can be built-in parameters of the machine P. The machine uses only function values and arithmetic operations from $\mathrm{ARI}_{\min}$. Since no comparison is used, the proof is complete.

We remark that the machine P from the proof uses roughly $2L/\varepsilon$ function values. We leave as an open problem if there exists a machine without comparison which uses essentially less function values.

We now show that the monotonicity assumption in the class $\mathrm{F}_{\mathrm{Linc}}$ of Proposition 4 is essential. Dropping the monotonicity assumption, we switch to the class $\mathrm{F}_{\mathrm{Lip}}$ of Lipschitz functions for which we have the following result.

Proposition 5. Let ARI be an arbitrary set consisting of continuous operations. If $\varepsilon < L/6$ then

$$\mathrm{comp}^{\mathrm{TOP}}(\mathrm{F}_{\mathrm{Lip}}, \mathrm{ARI}, \varepsilon) \; > \; 0.$$

Proof: In fact, we show Proposition 5 for a slightly smaller class

$$\mathrm{F}^0_{\mathrm{Lip}} \; = \; \mathrm{F}_{\mathrm{Lip}} \cap \{ f : \; f(0) \le -L/6, \; f(1) \ge L/6 \}.$$

Since $\mathrm{F}^0_{\mathrm{Lip}}$ is bounded and closed, a machine P without comparisons defines a continuous mapping. We show that P cannot solve the problem for $\varepsilon < L/6$.

Consider the piecewise linear function f^* consisting of three pieces with breakpoints $1/3$ and $2/3$, defined by $f^*(0) = -L/6$ and $f^*(1/3) = L/6$ and $f^*(2/3) = -L/6$ and $f^*(1) = L/6$. We consider two cases.

Case 1: $P(f^*) \le 2/3$.

Consider the affine linear function f_1 defined by $f_1(0) = -5L/6$ and $f_1(1) = L/6$. It is easy to see that any convex combination

$$f_\alpha = \alpha f^* + (1-\alpha) f_1, \quad \alpha \in [0,1],$$

is an element of $\mathrm{F}^0_{\mathrm{Lip}}$. Observe that $f_\alpha(2/3) = -L/6$ for each α. If $P(f_\alpha) \le 2/3$ for all α then the error is at least $L/6$ for a certain α. If $P(f_\alpha) > 2/3$ for some α then, by continuity of P, $P(f_\alpha) = 2/3$ for a certain α and, again, the maximal error of P is at least $L/6$.

Case 2: $P(f^*) \ge 1/3$. It is enough to replace f_1 by the affine linear function f_2 defined by $f_2(0) = -L/6$ and $f_2(1) = 5L/6$.

Hence, some comparisons are necessary for the class $\mathrm{F}_{\mathrm{Lip}}$ and $\varepsilon < L/6$. It would be interesting to find sharp estimates of the topological complexity in this case.

Remark 3. We remark that for the class $\mathrm{F}_{\mathrm{Linc}}$ we may also consider the root error criterion. Indeed, $\mathrm{F}_{\mathrm{Linc}}$ contains the class $\mathrm{F}_{\mathrm{lin}}$ of linear functions. Using the notation of the proof of Proposition 1 we conclude that $F^{\delta} \subset \mathrm{F}_{\mathrm{Linc}}$ for $\delta \in (0, L/2]$. Since Proposition 1 holds for F^{δ} it also holds for the class $\mathrm{F}_{\mathrm{Linc}}$. Proposition 1 states that for the root error criterion we must perform comparisons and the topological complexity is $\lceil -\log_2 \varepsilon - 1 \rceil$. For the residual complexity, the topological complexity is zero. This shows that the error criterion is crucial.

We have seen that for some classes it is easier to obtain an approximate zero in the residual sense than in the root sense. A similar phenomenon also occurs in the bit number model, see Weihrauch (1995).

Acknowledgments Our special thanks go to Steve Smale without whom this paper would not exist. We started this project during the Park City, Utah, conference in 1995, when Steve Smale proposed to extend the real number model of Blum, Shub and Smale to include information operations. Three of us spent many hours discussing this subject and the first written account of our discussions is given in Section 2. We also wanted to illustrate the extended model by some technical results. After the talk of V.A. Vassiliev in Park City, we decided to attack the problem of the topological complexity for classes of continuous functions. Steve Smale has been also very much involved in this project and, in particular, he noticed that the proof of Proposition 1 also holds for the tiny class of linear functions $\mathrm{F}_{\mathrm{lin}}$. It was his decision that only two of us be coauthors of this paper.

We are also grateful for useful comments from P. Hertling, L. Plaskota, and J. F. Traub.

References

1. L. Blum, M. Shub, and S. Smale (1989): On a theory of computation and complexity over the real numbers: NP completeness, recursive functions and universal machines. Bull. of the AMS **21**, 1–46.
2. P. Hertling (1996), Unstetigkeitsgrade - Topologische Komplexität, preliminary version, Report of Theoretische Informatik I, FernUniversität Hagen, D-58084 Hagen, Germany.
3. H. T. Kung (1976): The complexity of obtaining starting points for solving operator equations by Newton's method. In "Analytical Computational Complexity" (J. F. Traub, ed.), 35-75, Academic Press, New York.
4. E. Novak (1995): The real number model in numerical analysis. J. Complexity **11**, 57–73.
5. E. Novak, K. Ritter, H. Woźniakowski (1995): Average Case Optimality of a Hybrid Secant-Bisection Method. Math. Computation **64**, 1517-1539, 1995.
6. J. Renegar (1987): On the worst-case arithmetic complexity of approximating zeros of polynomials. J. Complexity **3**, 90–113.
7. K. Sikorski (1985): Optimal solution of nonlinear equations. J. Complexity **1**, 197-209.
8. S. Smale (1987): On the topology of algorithms. J. Complexity **3**, 81-89.
9. V. Strassen (1973): Vermeidung von Divisionen. J. Reine Angew. Math. **264**, 184–202.
10. J. F. Traub, G. W. Wasilkowski, H. Woźniakowski (1988): Information-based complexity. Academic Press.

11. V. A. Vassiliev (1992): Complements of discriminants of smooth maps: topology and applications. Transl. of Math. Monographs 98, 1992, revised 1994, Amer. Math. Soc., Providence, R.I.
12. K. Weihrauch (1995): A simple introduction to computable analysis. Informatik Bericht 171, Fernuniversität Hagen.