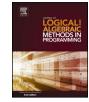


Contents lists available at ScienceDirect Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp



Models of computation for partial functions on the reals $\stackrel{\star}{\sim}$

CrossMark

Ming Quan Fu, Jeffery Zucker*

Department of Computing and Software, McMaster University, Hamilton, Ontario L8S 4K1, Canada

ARTICLE INFO

Article history: Received 20 November 2012 Received in revised form 3 November 2014 Accepted 3 November 2014 Available online 12 November 2014

Keywords: Generalized computability Computability on topological algebra Computability on the reals Grzegorczyk–Lacombe computability

ABSTRACT

We compare models of computation for partial functions $f: \mathbb{R} \to \mathbb{R}$. We consider four models: two concrete (Grzegorczyk–Lacombe and tracking computability), one abstract (approximability by a *While* program with "countable choice") and a new hybrid model: multipolynomial approximability. We show that these four models are equivalent, under the two assumptions:

- (1) the domain of f is the union of an *effective exhaustion*, i.e. a sequence of "stages", each of which is a finite union of disjoint rational open intervals, and
- (2) f is effectively locally uniformly continuous w.r.t. this exhaustion.

These assumptions seem to hold for all unary elementary functions of real analysis, many of which are, of course, partial. We make a conjecture with regard to this. © 2014 Elsevier Inc. All rights reserved.

1. Introduction

1.1. Background

In this paper we study computability of partial functions on the reals. We develop and compare a number of models of computability for such functions.

Previous work on comparing models of computability (for example, [19,23]) has tended to concentrate on computability models for total functions on \mathbb{R} . However, many of the well-known functions of elementary real analysis, which would certainly be considered as computable, are partial; for example, the rational, log and trigonometric functions. It is therefore essential that a study of models of computability on \mathbb{R} should include such functions in its considerations.

In fact, we will study partial functions $f: \mathbb{R} \to \mathbb{R}$, subject to certain assumptions on their domain, which seem to be satisfied by all functions of the kind listed above.

Now models of computation on \mathbb{R} can generally be divided into two classes: abstract and concrete. Abstract models of computation are independent of data representations. Concrete models, on the other hand, depend on a choice of data representation, usually constructed from the natural numbers \mathbb{N} , so that computation on an algebra is reduced to computation on \mathbb{N} [20,22,23].

¹ ' \rightarrow ' denotes a partial function.

http://dx.doi.org/10.1016/j.jlamp.2014.11.001 2352-2208/© 2014 Elsevier Inc. All rights reserved.

Research supported by a grant from the Natural Sciences and Engineering Research Council of Canada.
 Corresponding author.

E-mail addresses: fumq@mcmaster.ca (M.Q. Fu), zucker@mcmaster.ca (J. Zucker).

The abstract models that we consider are based on a high level *While* imperative programming language [21,22]. There are two familiar concrete models that we investigate: *Grzegorczyk–Lacombe* (GL) [7,8,12,15] and *tracking* computability [22,23].

We also consider another class of models: polynomial, or rather, in our case, *multipolynomial approximability*. This can be viewed as a hybrid model.

1.2. Comparison with case of total functions

The present paper can perhaps best be understood against the background of the paper [23]. We first summarize what was done in that paper. Five models of computation on \mathbb{R} were investigated for total functions:

- (i) Grzegorczyk-Lacombe (GL) computability,
- (ii) tracking computability,
- (iii) effective locally uniform (Q-)polynomial approximability,
- (iv) **WhileCC** approximability on a partial topological algebra \mathcal{R} ,
- (v) local uniform *While* approximability on a total topological algebra \mathcal{R}_t .

First, a brief explanation of these models. (More detailed descriptions will be given below.) Models (i) and (ii) are well known concrete models. Model (ii) uses a "tracking function" on \mathbb{N} according to a standard enumeration α of the rationals, and (hence) an enumeration $\overline{\alpha}$ of the computable reals. Models (iv) and (v) are abstract computation models, based on a *While* programming language. *WhileCC* is a nondeterministic extension of *While* which incorporates *countable choice*, i.e., nondeterministic choice of a natural number satisfying a given predicate. The algebras \mathcal{R} and \mathcal{R}_t are both topological algebras on \mathbb{R} : \mathcal{R} is a partial algebra, which includes partial equality, order and the inverse operation on the reals as basic functions, and \mathcal{R}_t is a total algebra, without these partial operations (and with the inverse operation restricted to naturals: see Remark 4.9.2).

In [23] all five computation models were shown to be equivalent, for functions $f : \mathbb{R}^m \to \mathbb{R}$ (m > 0) that are (a) *total* or, more generally, defined on a closed interval (or product of intervals, in the case m > 1), and (b) *effectively locally uniformly continuous*.

In the present paper, we attempt to generalize these results to the case that f need not be total. In fact we make two global assumptions on f:

- (a) **Domain exhaustion:** The domain U of f is the union of an *effective open exhaustion*, *i.e.*, an effective sequence of *stages* $(U_0, , U_1, U_2, ...)$, where² for $\ell = 0, 1, 2, ..., \overline{U_\ell} \subseteq U_{\ell+1}$ and U_ℓ is a finite union of rational open intervals $I_1^\ell, ..., I_{k_\ell}^\ell$ with disjoint closures, the *components* of the stage U_ℓ ; and
- (b) **Continuity:** The function f is effectively locally uniformly continuous with respect to this exhaustion.

So the "totality" assumption of [23] has been replaced by a more general "domain exhaustion" assumption.

These two assumptions appear to hold for all unary *elementary functions* on \mathbb{R} [9]. In Section 5.4 we present this as a conjecture.

A weaker version of the domain exhaustion assumption is considered in Section 5.2.

The important thing to note here is that dom(f) is (in general) no longer *connected* as a subspace of \mathbb{R} , as is guaranteed by the totality assumption in [23]. This invalidates, or at least complicates, some of the earlier arguments used in [23] to prove the equivalences listed above. We list three significant issues:

- (1) Polynomial approximability is no longer an appropriate computation model. Instead we consider (effectively locally uniform) multipolynomial approximability, in which each multipolynomial approximant q_{ℓ} is the union of a tuple of polynomials $(p_1^{\ell}, \ldots, p_{k_{\ell}}^{\ell})$, where **dom** $(p_i^{\ell}) = \overline{I_i^{\ell}}$, the closure of the *i*-th component of the stage U_{ℓ} ($i = 1, \ldots, k_{\ell}$).
- (2) Since *connectedness* of dom(f) is no longer assumed, the proof of equivalence of $While(\mathcal{R}_t)$ approximability with the other four models listed above fails. (Cf. [23, Lemma 3.2.18], where connectedness of dom(f) is crucial in the main step in the proof of $(v) \Rightarrow$ (iii).) In fact it is unknown whether equivalence of $While(\mathcal{R}_t)$ approximability with these other models still holds. In any case, this model (v) is left out of consideration here.
- (3) The analogue to the argument (iii) \Rightarrow (i) in the present paper extends the domains of the multipolynomials q_{ℓ} from the (the closure of) stage U_{ℓ} to U using linear interpolation.³ This only works if f is a function of one variable only, i.e., $f : \mathbb{R}^m \rightarrow \mathbb{R}$ only for m = 1 (unlike the case in [23]). More fundamentally, even the *definition* of "effective exhaustion" for m > 1 presents a challenge.⁴

² \overline{V} denotes the topological closure of a set *V*.

³ See proof of Equivalence Lemma 1 in Section 2.5.

⁴ This has been investigated in [6].

We must emphasize that the two global assumptions presented here are not intended as definitive characterizations of computable partial functions. They are intended rather as a useful generalization of the "totality" assumption of [23], which applies (apparently) to all elementary functions (cf. the conjecture in Section 5.4(2)). On the other hand, an investigation of the relationship of our models to (say) Weihrauch's Type Two computable functions on \mathbb{R} could well lead to a consideration of functions with domains more general than open sets, namely G_{δ} sets [25, p. 122, Ex. 18(b, c)].

1.3. Overview

Recall that we are considering partial functions $f: \mathbb{R} \to \mathbb{R}$, satisfying the domain exhaustion and continuity assumptions ((a) and (b) in Section 1.2 above). We shall prove, under these two global assumptions, the equivalence of the first four models (i)–(iv) listed in Section 1.2, with (in (iii)) *polynomial* approximability replaced by *multipolynomial* approximability. In other words, we shall prove the

Equivalence Theorem. Given a partial function $f: \mathbb{R} \to \mathbb{R}$, and an effective exhaustion (U_ℓ) of **dom**(f), suppose f is effectively locally uniformly continuous w.r.t. (U_ℓ) . Then the following are equivalent:

- (i) f is GL-computable w.r.t. (U_{ℓ}) ,
- (ii) *f* is tracking computable,
- (iii) f is effectively locally uniformly multipolynomially approximable w.r.t. (U_{ℓ}) ,
- (iv) f is **WhileCC** approximable on \mathcal{R} .

We will prove this Equivalence Theorem by means of three Equivalence Lemmas stating the equivalence of pairs of these models, as follows: Lemma 1: (i) \Leftrightarrow (iii), Lemma 2: (i) \Leftrightarrow (ii), and Lemma 3: (ii) \Leftrightarrow (iv), as we now discuss in more detail.

In Section 2 we present our first concrete computation model on \mathbb{R} : GL (Grzegorczyk–Lacombe) computability. Next we define the concept of multipolynomial approximability. This is a new model, not considered in [23], which enables us to generalize the equivalence results to functions on \mathbb{R} whose domains are not assumed to be connected⁵ (but satisfy at least the domain exhaustion assumption). We prove Equivalence Lemma 1: the equivalence between GL computability and multipolynomial approximability, and illustrate the multipolynomial approximability given by this proof for some well-known GL-computable partial functions on \mathbb{R} , using Maple 15.

In Section 3 we present our second concrete model: $\overline{\alpha}$ -tracking computability. We prove Equivalence Lemma 2, the equivalence between the two concrete models: GL and $\overline{\alpha}$ -tracking computability.

In Section 4, in preparation for our abstract model, we develop basic concepts connected with topological partial algebras, and in particular the topological partial algebra \mathcal{R} of reals. We then give the basic machinery for our abstract computation models on \mathbb{R} : the *While* programming language over \mathcal{R} . We also present the *WhileCC* language, which extends *While* with a nondeterministic "countable choice" command. We further consider an extensions of these languages with auxiliary real array variables, to form the languages *While*^{*} and *WhileCC*^{*}. These extensions are convenient for practical programming, but inessential from the viewpoint of theoretical computational power,⁶ and so we will write *While*^(*) and *WhileCC*^(*) for the language with or without these array variables. The appropriate notion of abstract computability turns out to be *WhileCC*^(*) approximability.

We then prove Equivalence Lemma 3: the equivalence of the abstract model (*WhileCC*^(*) approximability) with the concrete model ($\overline{\alpha}$ -computability). This was actually proved in [22] for a more general class of algebras. It can be viewed as a *completeness theorem* for our abstract model with respect to the concrete model.

By means of these three Equivalence Lemmas, we derive the Equivalence Theorem stated above.

In Section 5, we first (Section 5.1) summarize the results presented here, and discuss some related work. We then (Section 5.2) discuss a *weak version* of the domain exhaustion assumption, essentially the one used in [22,23]. With this version, it is possible to prove the equivalence of the models considered here *other than multipolynomial approximability*, even for functions $f: \mathbb{R}^m \to \mathbb{R}$ for m > 1. Next, in Section 5.3, we prove an invariance result for the global assumption (b) (continuity) with respect to assumption (a) (domain exhaustion).

Finally (Section 5.4) some ideas for future work are presented. The most interesting and important one we believe to be the extension of our equivalence theorem to functions of more than one argument, as discussed above (Section 1.2(3)).

Another interesting topic in Section 5.4 concerns our conjecture that all unary elementary functions satisfy the two global assumptions.

2. Exhaustions; GL computability; multipolynomial approximability; Equivalence Lemma 1

We define the concept of *exhaustion*, which is crucial for our work. Then we review one well-known concrete computation model: Grzegorczyk–Lacombe (GL) computability, and introduce a new model, *multipolynomial approximability*. Finally, we give the Equivalence Lemma that connects the above two models.

⁵ See point (1) in Section 1.2 above.

⁶ At least for computation on \mathcal{R} .

First, a note on codings.

2.1. Codings; computability of functions and predicates on \mathbb{Q}

We can define, in a standard way, surjective numerical *codings* of the sets \mathbb{N}^2 , \mathbb{N}^* , \mathbb{Z} and \mathbb{Q} . We write $\langle x, y \rangle$ for the code of a pair $(x, y) \in \mathbb{N}^2$, $\langle x_1, \ldots, x_n \rangle$ for the code of a tuple $(x_1, \ldots, x_n) \in \mathbb{N}^*$ $(n \ge 0)$, and more generally, $\lceil x \rceil$ for the code of an element x of \mathbb{Z} , \mathbb{Q} , etc.

A function or predicate of rationals $r_1, r_2, ...$ is called *computable* or *effective* or *decidable* if the corresponding function or predicate of codes $\lceil r_1 \rceil, \lceil r_2 \rceil, ...$ is computable (or recursive).

A predicate of rationals $r_1, r_2, ...$ is called *semicomputable* if the corresponding predicate of codes $\lceil r_1 \rceil, \lceil r_2 \rceil, ...$ is semicomputable (or r.e.).

2.2. Exhaustions; local approximability and continuity

Definition 2.2.1 (*Exhaustion*). Let U be an open subset of \mathbb{R} , and $(U_0, U_1, U_2, ...)$ a sequence of open subsets of \mathbb{R} , such that

- (1) $U = \bigcup_{\ell=0}^{\infty} U_{\ell}$,
- (2) for $\ell = 0, 1, 2, ..., U_{\ell}$ is a finite union of non-empty open finite intervals $I_1^{\ell}, I_2^{\ell}, ..., I_{k_{\ell}}^{\ell}$ ($k_{\ell} \ge 1$) whose closures are pairwise disjoint,⁷ and
- (3) $\overline{U_{\ell}} = \bigcup_{i=0}^{k_{\ell}} \overline{I_i^{\ell}} \subseteq U_{\ell+1}$ for $\ell = 1, 2, \dots$

Then the sequence (U_{ℓ}) is called an *exhaustion* of U, and for each ℓ , U_{ℓ} is a *stage* of the exhaustion, with *components* $I_{1}^{\ell}, \ldots, I_{k}^{\ell}$.

Definition 2.2.2 (*Effective exhaustion*). An exhaustion (U_{ℓ}) of U is called *effective* if for all ℓ , the components I_i^{ℓ} are *rational*, i.e., $I_i^{\ell} = (a_i^{\ell}, b_i^{\ell})$, where $a_i^{\ell}, b_i^{\ell} \in \mathbb{Q}$ $(i = 1, ..., k_{\ell})$, and $b_i^{\ell} < a_{i+1}^{\ell}$ $(i = 1, ..., k_{\ell} - 1)$, and the map

 $\ell \mapsto \langle k_{\ell}, \lceil a_1^{\ell} \rceil, \lceil b_1^{\ell} \rceil, \dots, \lceil a_{k_{\ell}}^{\ell} \rceil, \lceil b_{k_{\ell}}^{\ell} \rceil \rangle$

which delivers the sequence of stages

$$U_{\ell} = I_1^{\ell} \cup \cdots \cup I_{k_{\ell}}^{\ell}$$

is recursive.

Remark 2.2.3. From Definition 2.2.1(3) it follows that the components I_i^{ℓ} have the following covering property:

$$\forall \ell \; \forall i \in \{1, \ldots, k_\ell\} \; \exists j \in \{1, \ldots, k_{\ell+1}\} \quad \left(\overline{I_i^\ell} \subseteq I_j^{\ell+1}\right).$$

Now consider a partial function $f: \mathbb{R} \to \mathbb{R}$, with domain U, which is the union of an open exhaustion (U_{ℓ}) .

Definition 2.2.4 (Local uniform continuity). f is locally uniformly continuous w.r.t. (U_{ℓ}) if $\forall \ell \forall \varepsilon > 0 \exists \delta > 0 \forall x, y \in U_{\ell}$

$$|x-y| < \delta \implies |f(x)-f(y)| < \varepsilon.$$

This definition can be effectivized:

Definition 2.2.5 (*Effective local uniform continuity*). f is *effectively locally uniformly continuous w.r.t.* an effective exhaustion (U_{ℓ}) if there is a recursive function $M: \mathbb{N}^2 \to \mathbb{N}$ such that for all k, ℓ and all $x, y \in U_{\ell}$

$$|x-y| < 2^{-M(k,\ell)} \implies |f(x) - f(y)| < 2^{-k}.$$

Now consider a sequence of functions $f_n: \mathbb{R} \to \mathbb{R}$, all with the same domain $dom(f_n) = U$, the union of an effective exhaustion (U_ℓ) .

⁷ I.e., they don't even have any endpoints in common.

Definition 2.2.6 (*Effectively locally uniformly continuous sequence*). The sequence (f_n) is *effectively locally uniformly continuous* w.r.t. (U_ℓ) if there is a recursive function $M: \mathbb{N}^3 \to \mathbb{N}$ such that for all k, ℓ, n and all $x, y \in U_\ell$

 $|x-y| < 2^{-M(k,\ell,n)} \implies |f_n(x) - f_n(y)| < 2^{-k}.$

For the rest of this paper, we investigate the computability properties of functions $f: \mathbb{R} \to \mathbb{R}$, satisfying the following global assumptions on f:

Global Assumptions. The function $f: \mathbb{R} \rightarrow \mathbb{R}$ satisfies:

- (a) **Domain exhaustion:** The domain U of f is a union of an effective exhaustion (U_{ℓ}) (cf. Definition 2.2.2):
- (b) **Continuity:** f is effectively locally uniformly continuous w.r.t. (U_{ℓ}) .

2.3. GL-computability

The following six definitions are adapted from [15, Ch. 0], where the domains of f and f_n were assumed to be products of intervals: \mathbb{R}^m or $[0, 1]^m$ (m > 0).

We assume below that the functions f_n , like f, all have domain U.

Definition 2.3.1 (*Computable sequence of reals*). A sequence of real numbers (x_n) is *computable* iff there exists a computable double sequence of rationals (r_{nk}) such that for all n, k:

 $|x_n-r_{nk}|\leq 2^{-k}.$

Definition 2.3.2 (Sequential computability of function). f is sequentially computable on U if f maps every computable sequence of reals $x_n \in U$ into a computable sequence $(f(x_n))$ of reals.

Definition 2.3.3 (Sequential computability of sequence of functions). The sequence (f_n) of functions is sequentially computable on U if for any computable sequence (x_k) of reals in U, the double sequence $(f_n(x_k))$ of reals is computable, i.e. there exists a computable triple sequence (r_{nkj}) of rationals, such that for all n, k, j,

 $\left|r_{nkj}-f_n(x_k)\right|<2^{-j}.$

Definition 2.3.4 (*GL-computability*). f is GL-computable w.r.t. (U_{ℓ}) iff:

- (1) f is sequentially computable on U, and
- (2) f is effectively locally uniformly continuous w.r.t. (U_{ℓ}) .

Definition 2.3.5 (*GL-computable sequence*). The sequence (f_n) is GL-computable w.r.t. (U_ℓ) iff:

- (1) (f_n) is effectively locally uniformly continuous w.r.t. (U_ℓ) , and
- (2) (f_n) is sequentially computable on U.

Remark 2.3.6 (Definition of GL-computability).

- (1) Condition (1) in Definition 2.3.5 is subsumed under the global assumption (b).
- (2) Our definition is a modification of the original one [7,8,12,15] which assumes not only totality, but also *effective global uniform continuity* of *f*.

Definition 2.3.7 (*Effective local uniform convergence*). The sequence (f_n) converges to $f(f_n \to f)$ effectively locally uniformly w.r.t. (U_ℓ) iff there is a recursive function $M : \mathbb{N}^2 \to \mathbb{N}$ such that for all k, ℓ, n and all $x \in U_\ell$,

 $n \ge M(k, l) \implies |f_n(x) - f(x)| < 2^{-k}.$

Lemma 2.3.8 (*Closure of GL-computability under effective local uniform convergence).* If (f_n) is a *GL-computable sequence w.r.t.* (U_ℓ) , and $f_n \to f$ effectively locally uniformly w.r.t. (U_ℓ) , then f is *GL-computable w.r.t.* (U_ℓ) .

Proof. This adapts the proof in [15, Ch. 0, Thm 4] of the closure of GL-computability under effective uniform convergence on closed finite intervals. Note that by Definition 2.2.1, $\overline{U_{\ell}} \subseteq U_{\ell+1}$ for all ℓ , and so we can apply to the compact stages $\overline{U_{\ell}}$, the arguments given in the proof in [15] concerning effective uniform continuity and convergence on the compact domain of the functions there. \Box

2.4. Multipolynomial approximability

Note that by "polynomial" we will always mean Q-polynomial, i.e. polynomial with rational coefficients.

Definition 2.4.1 (*Multipolynomial*). Given a finite sequence of polynomials $(p_1, p_2, ..., p_k)$ and a sequence of open intervals $(I_1, I_2, ..., I_k)$ with disjoint closures, we define a (Q-)multipolynomial q(x) with domain $\bigcup_{i=1}^k \overline{I_i}$ as follows:

$$q(x) \simeq \begin{cases} p_1(x) & \text{if } x \in \overline{I_1} \\ p_2(x) & \text{if } x \in \overline{I_2} \\ \dots \\ p_k(x) & \text{if } x \in \overline{I_k} \\ \uparrow & \text{otherwise.} \end{cases}$$

(Here '~' means that both sides are either defined and equal, or undefined.) We denote this multipolynomial by

$$q = [p_1 \upharpoonright I_1, \ldots, p_k \upharpoonright I_k].$$

Definition 2.4.2 (*Effective sequence of multipolynomials*). Given an effective exhaustion (U_{ℓ}) of U, with $U_{\ell} = I_1^{\ell} \cup \cdots \cup I_{k_{\ell}}^{\ell}$, and an effective sequence of tuples of polynomials $(p_1^{\ell}, \ldots, p_{k_{\ell}}^{\ell})$ ($\ell = 0, 1, 2, \ldots$), the sequence (q_{ℓ}) , where

$$q_{\ell} = \left[p_1^{\ell} \upharpoonright \overline{I_1^{\ell}}, \dots, p_{k_{\ell}}^{\ell} \upharpoonright \overline{I_{k_{\ell}}^{\ell}}\right], \tag{2.1}$$

is called an effective sequence of multipolynomials.

Definition 2.4.3 (*Effective local multipolynomial approximability*). Given $f: \mathbb{R} \to \mathbb{R}$, and an effective exhaustion (U_{ℓ}) of U = dom(f), with $U_{\ell} = I_1^{\ell} \cup \cdots \cup I_{k_{\ell}}^{\ell}$, we say that the effective sequence of multipolynomials (q_{ℓ}) (as in (2.1)) converges to f $(q_{\ell} \to f)$ effectively locally uniformly w.r.t. (U_{ℓ}) if there is a recursive function $M: \mathbb{N}^2 \to \mathbb{N}$ such that for all k, ℓ, n , and all $x \in \overline{U_{\ell}}$:

 $n \ge M(k, \ell) \implies |q_n(x) - f(x)| < 2^{-k}.$

We also say: f is effectively locally multipolynomially approximable by (q_{ℓ}) w.r.t. (U_{ℓ}) .

Note the difference between effective local uniform approximation by a sequence of functions (f_n) and by a sequence of multipolynomials (q_ℓ) (Definitions 2.3.7 and 2.4.3): the domains of all the f_n are U, whereas the domain of each q_ℓ is \overline{U}_ℓ .

2.5. Equivalence between GL-computability and multipolynomial approximability

We present the first of our three Equivalence Lemmas.

Recall the global assumptions of domain exhaustion and continuity on $f: \mathbb{R} \to \mathbb{R}$. Let (U_ℓ) be an effective exhaustion of dom(f).

Equivalence Lemma 1 (GL-computability and multipolynomial approximability). The following are equivalent:

(i) f is effectively locally uniformly multipolynomially approximable w.r.t. (U_{ℓ}) ,

(ii) f is GL-computable w.r.t. (U_{ℓ}) .

Proof. We first prove (i) \Rightarrow (ii). So suppose *f* is effectively locally uniformly multipolynomially approximable by (q_{ℓ}) w.r.t. (U_{ℓ}) .

First we extend each q_{ℓ} to a function f_{ℓ} with domain *U*, by linear interpolation between the components: Suppose

$$q_{\ell} = \left[p_1^{\ell} \upharpoonright \overline{I_1^{\ell}}, \ldots, p_{k_{\ell}}^{\ell} \upharpoonright \overline{I_{k_{\ell}}^{\ell}}\right],$$

where $I_i^{\ell} = (a_i^{\ell}, b_i^{\ell})$ and $b_i^{\ell} < a_{i+1}^{\ell}$ for $i = 1, \dots, k_{\ell} - 1$. Define

$$f_{\ell}(x) \simeq \begin{cases} p_i^{\ell}(x) & \text{if } x \in \overline{I_i^{\ell}} \ (i = 0, \dots, k_{\ell} - 1) \\ \frac{(p_{i+1}^{\ell}(a_{i+1}) - p_i^{\ell}(b_i))(x - b_i)}{a_{i+1} - b_i} + p_i^{\ell}(b_i) & \text{if } x \in U, \ b_i^{\ell} < x < a_{i+1}^{\ell} \\ p_1^{\ell}(a_1^{\ell}) & \text{if } x \in U, \ x < a_1^{\ell} \\ p_{k_{\ell}}^{\ell}(b_{k_{\ell}}^{\ell}) & \text{if } x \in U, \ x > b_{k_{\ell}}^{\ell} \\ \uparrow & \text{if } x \notin U. \end{cases}$$

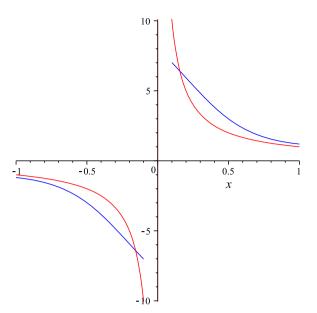


Fig. 1. Multipolynomial approximation for 1/x ($\ell = 5$).

Note the following:

- (1) Each f_{ℓ} is GL-computable, and in fact, (f_{ℓ}) is a GL-computable sequence of functions. This uses a "patching theorem" [15, p. 32, Thm. 2].
- (2) $f_{\ell} \to f$ effectively locally uniformly w.r.t. (U_{ℓ}) . To prove this, note that for any $x \in U$, there exists ℓ such that $x \in U_{\ell}$. Then $\forall n \geq \ell$, $f_n \upharpoonright U_{\ell} = q_n \upharpoonright U_{\ell}$, and so $f_n(x) = q_n(x)$.

Hence by Lemma 2.3.8, f is GL-computable w.r.t. (U_{ℓ}) .

Note that the above construction of the approximating functions f_{ℓ} by linear interpolation cannot be extended to an obvious way to functions $f : \mathbb{R}^m \to \mathbb{R}$ for m > 1. We return to this in Section 5.4.

Next we prove (ii) \Rightarrow (i). Here we adapt the proof of the Effective Weierstrass Theorem [15, Ch. 0, Sec. 7] as follows.

Suppose f is GL-computable w.r.t. (U_{ℓ}) . For each ℓ and each $i = 1, ..., k_{\ell}$, apply this theorem [15, p. 45] to the closed interval $I_i^{\overline{\ell}}$, to get a polynomial p_i^{ℓ} which approximates $f \upharpoonright \overline{I_i^{\ell}}$ uniformly on $\overline{I_i^{\ell}}$ by $\leq 2^{-\ell}$, i.e.,

$$\forall x \in \overline{I_i^\ell}, \quad \left| p_i^\ell(x) - f(x) \right| \le 2^{-\ell}.$$

Now define the sequence of multipolynomials (q_{ℓ}) as in (2.1). Then *f* is effectively locally uniformly multipolynomially approximable by (q_{ℓ}) w.r.t., (U_{ℓ}) , as desired. \Box

In the following two examples (illustrated using Maple 15), we adapt the construction of polynomial approximations given in [15, Ch. 0, Sec. 7] to the (non-connected) domains $\overline{U_{\ell}}$.

Example 2.5.1 (Multipolynomial approximations). Consider the functions f with domain U, where

(1) f(x) = 1/x, $U = \mathbb{R} \setminus \{0\}$. Let

$$U_{\ell} = \left(I_1^{\ell}, I_2^{\ell}\right),$$

where

$$I_1^{\ell} = \left(-\ell, -\frac{1}{\ell}\right), \qquad I_2^{\ell} = \left(\frac{1}{\ell}, \ell\right), \quad \ell = 1, 2, 3, \dots$$

In Fig. 1 the multipolynomial approximant q_{ℓ} for f is shown for $\ell = 10$.

(2) $f(x) = \tan(\pi x/2), U = \{x \in \mathbb{R} \mid x \neq 2k + 1, k \in \mathbb{Z}\}.$ Let

$$U_{\ell} = \bigcup_{i=-\ell}^{\ell} \left(2i - 1 + \frac{1}{\ell+1}, 2i + 1 - \frac{1}{\ell+1} \right), \quad \ell = 0, 1, 2, \dots$$

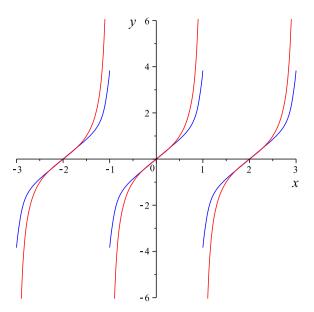


Fig. 2. Multipolynomial approx. for $tan(\pi x/2)$ ($\ell = 1$).

In Fig. 2 the multipolynomial approximant q_{ℓ} for f is shown for $\ell = 1$.

Remark 2.5.2.

- (1) It is clear that polynomial approximability as in e.g. [15] would be inappropriate here, where dom(f) is not an interval.
- (2) We use the polynomial approximation sequences (p_i^{ℓ}) as defined in [15] for $f \upharpoonright I_i^{\ell}$. Then, from these p_i^{ℓ} we define the multipolynomial sequences q_{ℓ} . Note however that the multipolynomials obtained by the formula in [15] have coefficients which are (computable, but) not rational. To obtain Q-multipolynomials as required by Equivalence Lemma 1, we must further approximate the polynomials p_i^{ℓ} obtained as above by Q-polynomials which approximate them uniformly
 - on $\overline{I_i^{\ell}}$ by $2^{-\ell}$. This is straightforward, and we omit details.
- (3) It would be interesting to investigate whether other polynomial sequences commonly used in approximation theory would give better results.

3. Tracking computability; Equivalence Lemma

In this section, we present our second concrete computation model: $(\overline{\alpha}$ -)*tracking computability* (or just $\overline{\alpha}$ -*computability*), and prove its equivalence to GL-computability.

This model arose [22–24] as a natural generalization, to not necessarily effective structures, of Mal'cev's numbering theory [13] (see Remark 3.1.7).

3.1. Tracking computability

We fix a standard enumeration of \mathbb{Q} , i.e., a bijection $\alpha : \mathbb{N} \approx \mathbb{Q}$, under which the field operations on \mathbb{Q} are all primitive recursive. In order to construct a satisfactory model of concrete computability on \mathbb{R} , we must first construct from α an enumeration $\overline{\alpha}$ of a more extensive subset of \mathbb{R} than \mathbb{Q} , namely the (α -)computational closure of \mathbb{Q} , as we now explain.

Definition 3.1.1 (*Computable reals*). First, we define the α -computational closure of \mathbb{Q} , i.e., the set \mathbb{R}_c of α -computable reals, where

 $\mathbb{Q} \subset \mathbb{R}_{c} \subset \mathbb{R},$

with an enumeration⁸

$$\overline{\alpha}: \Omega \twoheadrightarrow \mathbb{R}_{c}.$$

The set $\Omega \subset \mathbb{N}$ consists of codes for \mathbb{R}_c , i.e. pairs of numbers $c = \langle e, m \rangle$, where

⁸ '----- 'denotes a surjection.

(1) *e* is an index for a total recursive function $\{e\}$: $\mathbb{N} \to \mathbb{N}$ defining a Cauchy sequence

$$\alpha(\{e\}(0)), \, \alpha(\{e\}(1)), \, \alpha(\{e\}(2)), \dots, \tag{3.1}$$

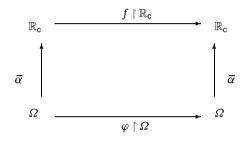
of elements of \mathbb{Q} , and

(2) m is an index for a computable modulus of convergence for this sequence:

$$\forall k, l \ge \{m\}(n) : \left| \alpha(\{e\}(k)) - \alpha(\{e\}(l)) \right| < 2^{-n}.$$
(3.2)

For any such code $c = \langle e, m \rangle \in \Omega$, $\overline{\alpha}(c)$ is defined as the limit in \mathbb{R} of the Cauchy sequence (3.1), and \mathbb{R}_c is the range of $\overline{\alpha}$.

Definition 3.1.2 ($\overline{\alpha}$ -tracking function). Given two functions $f : \mathbb{R} \to \mathbb{R}$ and $\varphi : \mathbb{N} \to \mathbb{N}$, we say that φ is an $\overline{\alpha}$ -tracking function for f if the following diagram commutes:



in the sense that for all $k \in \Omega$,

(i) $f(\overline{\alpha}(k)) \downarrow \Longrightarrow \varphi(k) \downarrow \land f(\overline{\alpha}(k)) = \overline{\alpha}(\varphi(k))$, and (ii) $f(\overline{\alpha}(k)) \uparrow \Longrightarrow \varphi(k) \uparrow$.

Definitions 3.1.3 ($\overline{\alpha}$ -computability and semicomputability).

- (a) The function $f: \mathbb{R} \to \mathbb{R}$ is $\overline{\alpha}$ -computable if it has a recursive $\overline{\alpha}$ -tracking function.
- (b) A subset of \mathbb{R} is $\overline{\alpha}$ -semicomputable if it is the domain of an $\overline{\alpha}$ -computable function.

Remark 3.1.4 ($\bar{\alpha}$ -semicomputable sets). It is easy to see that the union of an effective open exhaustion is $\bar{\alpha}$ -semicomputable. Hence also the domain of any function satisfying the domain exhaustion assumption (global assumption (a), Section 2.2) is $\bar{\alpha}$ -semicomputable.

Remark 3.1.5 (*Terminology for* α *-tracking function and* α *-computability*). In [22,23], the concept defined in Definition 3.1.2 was called a "*strict* $\overline{\alpha}$ -tracking function", whereas an " $\overline{\alpha}$ -tracking function" only had to satisfy condition (i). ("Strict $\overline{\alpha}$ -computability" and "strict $\overline{\alpha}$ -semicomputability" were defined accordingly.) However, these two concepts coincide for any function whose domain is $\overline{\alpha}$ -semicomputable [22, Lemma 10.2.4], and hence, by Remark 3.1.4, for any function satisfying our global assumptions.

Remark 3.1.6 (*Fast Cauchy sequences*). As explained in [22], we get an equivalent theory if we assume (by effectively taking subsequences) that the sequences (3.1) are *fast* Cauchy sequences, i.e., the modulus of convergence is always the identity function on \mathbb{N} , so that (3.2) becomes

 $\forall n, \forall k > n : \left| \alpha \left(\{e\}(k) \right) - \alpha \left(\{e\}(n) \right) \right| < 2^{-n}$

and so we can work with "e-codes" instead of "c-codes" as elements of Ω .

Remark 3.1.7 (*Mal'cev numberings*). From the perspective of Mal'cev's theory of numberings [13], α is a computable numbering, or effective listing, of the structure \mathbb{Q} . Moreover, we are extending Mal'cev's concept of numbering to the enumeration $\overline{\alpha}$ of the structure \mathbb{R}_c : it is a numbering, but not computable, as it is not an effective listing of \mathbb{R}_c , since the domain Ω of $\overline{\alpha}$ is not decidable. Its effectivity rests on the algorithmic nature of its operations.

3.2. Equivalence between $\overline{\alpha}$ - and GL-computability

Let U = dom(f), where f satisfies the global assumptions (Section 2.2).

Lemma 3.2.1. The predicate⁹ "**Nbd** $(r, 2^{-n}) \subseteq U_{\ell}$ " (for $r \in \mathbb{Q}$) is computable in r, n and ℓ (cf. Section 2.1).

This is clear.

Next we need some notation. Define

$$D = \{k \in \mathbb{N} \mid \alpha(k) \in U\},\$$
$$E = \{e \in \Omega \mid \overline{\alpha}(e) \in U\}.$$

Note that by Lemma 3.2.1, *D* is r.e.

Now suppose $x \in U \cap \mathbb{R}_c$. Then there exists $e \in E$ such that

 $\overline{\alpha}(e) = x.$

So putting

 $r_n = \alpha\big(\{e\}(n)\big),$

we see that the sequence (r_n) of rationals converges to x.

However there is no guarantee that $\forall n \ (r_n \in U)$. To ensure this, we can effectively change the index e to an index \hat{e} such that $\overline{\alpha}(\hat{e}) = \overline{\alpha}(e) = x$, and for all $n, \alpha(\{\hat{e}\}(n)) \in U$, i.e., $\hat{e}(n) \in D$, as follows.

Lemma 3.2.2. There is a partial recursive function $in_U : \mathbb{N} \to \mathbb{N}$ such that for all $e \in E$, putting $\hat{e} = in_U(e)$, we have $\hat{e} \in E$ and $\overline{\alpha}(\hat{e}) = \overline{\alpha}(e)$ and for all $n, \hat{e}(n) \in D$.

Proof. Let $e \in E$, $x = \overline{\alpha}(e)$. Then there exists *N* such that

$$Nbd(x, 2 * 2^{-N}) \subseteq U.$$

Hence

$$Nbd(r_N, 2^{-N}) \subseteq U \tag{3.4}$$

and so

$$\forall n > N, \quad r_n \in U. \tag{3.5}$$

Note that we can effectively find an N satisfying (3.4), and hence (3.5), by searching for some N and ℓ satisfying

 $Nbd(r_N, 2^{-N}) \subseteq U_\ell,$

using Lemma 3.2.1.

Now let \hat{e} be an index such that for all n,

$$\{\widehat{e}\}(n) = \lceil r_{N+n} \rceil$$

Note that \hat{e} can be found effectively from *e*. Hence there is a recursive function $\operatorname{in}_U : \mathbb{N} \to \mathbb{N}$ with $\operatorname{in}_U(e) = \hat{e}$, as desired. \Box

Note that the concept of a *computable sequence of reals* (Definition 2.3.1) can be reformulated in terms of $\overline{\alpha}$ -computability:

Lemma 3.2.3. A sequence (x_n) of reals is computable (according to Definition 2.3.1) if, and only if, there is a total recursive function $\psi : \mathbb{N} \to E$ such that for all $n, \overline{\alpha}(\psi(n)) = x_n$.

Proof. By the S-m-n Theorem [11,17]. \Box

We come to the Equivalence Lemma for our two concrete computation models. Recall our global assumptions (cf. Section 2.2) on $f: \mathbb{R} \to \mathbb{R}$. Let (U_ℓ) be an effective exhaustion of **dom**(*f*).

Equivalence Lemma 2 (*GL*- and $\overline{\alpha}$ -computability). The following are equivalent:

(ii) f is $\overline{\alpha}$ -computable.

(3.3)

⁽i) f is GL-computable w.r.t. (U_{ℓ}) .

⁹ Where *Nbd*(a, δ) is the open neighborhood of a with radius δ .

Proof. First we prove (ii) \Rightarrow (i). Suppose f is $\overline{\alpha}$ -computable. We must show f is GL-computable w.r.t. (U_{ℓ}) .

By the Continuity (global assumption (b)), we only have to show that f is sequentially computable on U (cf. Definitions 2.3.2 and 2.3.4).

Take any computable sequence (x_n) in U. We must show that the sequence $(f(x_n))$ is also computable. By Lemma 3.2.3, there is a recursive function $\psi \colon \mathbb{N} \to \mathbb{N}$, such that for all n

$$x_n = \overline{\alpha} \big(\psi(n) \big).$$

Let

$$S_1 = \overline{\alpha} \circ \psi : \mathbb{N} \to \mathbb{R}$$

Then for all *n*,

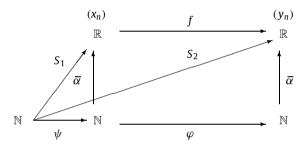
 $x_n = S_1(n).$

Since *f* is $\overline{\alpha}$ -computable, there is an $\overline{\alpha}$ -tracking function $\varphi \colon \mathbb{N} \to \mathbb{N}$ for *f*. Put

$$y_n = f(x_n),$$

and let

$$S_2 = \overline{\alpha} \circ \varphi \circ \psi : \mathbb{N} \to \mathbb{R}.$$



Then for all *n*,

$$y_n = S_2(n) = \overline{\alpha} \big((\varphi \circ \psi)(n) \big),$$

so by Lemma 3.2.3 (y_n) is also a computable sequence of reals. We have proved f is sequentially computable on U, and hence f is GL-computable w.r.t. (U_ℓ) . Next, we will prove (i) \Rightarrow (ii). The idea of the proof is that

(1) sequential computability of f determines its value on the rationals $\mathbb{Q} \cap U$, a dense subset of U, and

(2) effective local uniform continuity of f then determines its value on the computable reals $\mathbb{R}_{c} \cap U$.

So suppose f is GL-computable w.r.t. (U_{ℓ}) . We must show that f is $\overline{\alpha}$ -computable; i.e. we must construct an $\overline{\alpha}$ -tracking function φ for f.

By (3.3) and Lemma 3.2.1, D is r.e. Hence there is an effective enumeration or listing of D:

$$\rho:\mathbb{N}\approx D\subseteq\mathbb{N},$$

such that $\alpha \circ \rho$ is an effective enumeration of $\mathbb{Q} \cap U$. Putting

$$r_n =_{df} \alpha \big(\rho(n) \big) \in U$$

we have $\mathbb{Q} \cap U = ran(\alpha \circ \rho) = \{r_n \mid n \in \mathbb{N}\}.$ Also, the inverse function

$$\rho^{-1}: D \approx \mathbb{N}$$

is a partial recursive function from \mathbb{N} to \mathbb{N} . Note that for all $q \in \mathbb{Q} \cap U$,

$$q = \alpha(\lceil q \rceil) = r_{\rho^{-1}(\lceil q \rceil)}$$

(putting $n = \rho^{-1}(\lceil q \rceil)$ in (3.6)). Consider the computable double sequence (r_{nk}) of rationals where for all k,

 $r_{nk} = r_n = \alpha(\rho(n)).$

(3.7)

(3.6)

Suppose

 $f(r_n) = u_n. \tag{3.8}$

Since *f* is sequentially computable, there is computable double sequence (u_{nk}) of rationals such that $u_{nk} \rightarrow u_n$ (and by Remark 3.1.6 we may assume fast convergence).

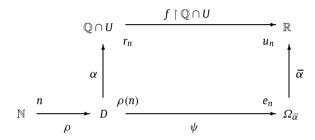
So we can find, effectively in *n*, and hence in $\rho(n)$, an index e_n for the sequence $(\lceil u_{n0} \rceil, \lceil u_{n1} \rceil, \lceil u_{n2} \rceil, ...)$. In other word, there is a recursive function ψ such that for all *n*

$$\psi(\rho(n)) = e_n.$$

Hence for all $n, u_n \in \mathbb{R}_c$ with

$$u_n = \overline{\alpha}(e_n) = \overline{\alpha}(\psi(\rho(n))). \tag{3.9}$$

So ψ (or rather $\psi \circ \rho$) is a kind of $(\alpha, \overline{\alpha})$ -tracking function for $f \upharpoonright \mathbb{Q} \cap U$:



Now suppose given a computable real $x \in \mathbb{R}_c \cap U$, with $x = \overline{\alpha}(e)$, $e \in \Omega$. Note that even though $x \in U$, there is no guarantee that for all n, $\{e\}(n) \in U$. We must therefore replace e by $\hat{e} = in_U(e)$, as in Lemma 3.2.2.

Then we have a sequence of rationals $\alpha \circ \{\widehat{e}\}$, such that for all *n*, putting

$$s_n = \alpha(\{\widehat{e}\}(n)) \in \mathbb{Q} \cap U,$$

we have

$$\lceil s_n \rceil = \{\widehat{e}\}(n) \in D \tag{3.10}$$

and

$$s_n \to x. \tag{3.11}$$

Next, putting

$$y_n = f(s_n) \tag{3.12}$$

we have

$$y_{n} = f(s_{n}) = f(r_{\rho^{-1}(\lceil s_{n} \rceil)}) \quad \text{by (3.7) putting } q = s_{n}$$

$$= u_{\rho^{-1}(\lceil s_{n} \rceil)} \qquad \text{by (3.8) with } n \leftarrow \rho^{-1}(\lceil s_{n} \rceil)$$

$$= \overline{\alpha} \left(\psi \left(\rho \left(\rho^{-1}(\lceil s_{n} \rceil) \right) \right) \right) \quad \text{by (3.9) with } n \leftarrow \rho^{-1}(\lceil s_{n} \rceil)$$

$$= \overline{\alpha} \left(\psi \left(\lceil s_{n} \rceil \right) \right)$$

$$= \overline{\alpha} \left(\psi \left(\lceil \hat{e} \rceil(n) \right) \right) \qquad \text{by (3.10).}$$

$$(3.13)$$

Hence $y_n \in \mathbb{R}_c$ for all *n*.

Next, since *f* is *effectively locally uniformly continuous* w.r.t. (U_ℓ) , by (3.12) (y_n) is an effective Cauchy sequence. (The proof is an effective version of the standard proof that local uniform continuity of a mapping preserves the Cauchy property of sequences.) Again (by taking subsequences) we may assume (y_n) is a fast Cauchy sequence.

By completeness of \mathbb{R} , (y_n) has a limit y. So by (3.11), (3.12) and continuity of f,

$$f(x) = y$$
.

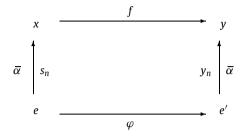
From (3.13) each y_n is a limit of an effective sequence of rationals

$$t_{nk} = \alpha \left(\left\{ \psi \left(\{ \hat{e} \}(n) \right) \right\}(k) \right).$$

Let e' be an index for the diagonal sequence (t_{kk}) . Then $e' \in \Omega$ and

$$\overline{\alpha}(e')=y.$$

The effective mapping from *e* to *e'* sketched above is a recursive $\overline{\alpha}$ -tracking function φ for *f*:



Hence *f* is $\overline{\alpha}$ -computable. \Box

This Equivalence Lemma, for total *f*, was stated without proof in [23].

In Section 5.2 we return to this Equivalence Lemma, to show how it holds for functions $f: \mathbb{R}^m \to \mathbb{R}$ for all $m \ge 1$ with a weaker domain exhaustion assumption.

4. Abstract models; the topological partial algebra ${\cal R}$

This section is devoted to abstract computation on \mathbb{R} . To prepare for this, we discuss abstract many-sorted algebras, and more particularly, topological partial algebras, illustrated by the topological partial algebra \mathcal{R} of reals. We then describe the *While* programming language, and its extensions, such as *WhileCC* (*While* with "countable choice"), and hence the concepts of *While* and *WhileCC* approximability for functions on \mathbb{R} . We then state the third Equivalence Lemma, on the equivalence of *WhileCC* approximability with $\bar{\alpha}$ -computability. From this follows the equivalence theorem for our four models of computation on \mathbb{R} , under our global assumptions.

4.1. Basic concepts: Signatures and algebras

Definition 4.1.1 (*Many-sorted signatures*). A many-sorted signature Σ is a pair $(Sort(\Sigma), Func(\Sigma))$ where

- (a) **Sort**(Σ) is a finite set of sorts, written s, s', \ldots , and
- (b) **Func**(Σ) is a finite set of (primitive or basic) function symbols F, each with a type of the form $s_1 \times \cdots \times s_m \to s$, written

 $F:s_1\times\cdots\times s_m\to s,$

where $m \ge 0$ is the arity of *F*. The case m = 0 corresponds to a *constant*; we then write $F: \rightarrow s$.

Definition 4.1.2 (*Product types over* Σ). A Σ -product type has the form $s_1 \times \cdots \times s_m$ ($m \ge 0$), where s_1, \ldots, s_m are Σ -sorts. Product types are written u, v, \ldots .

Definition 4.1.3 (Σ -algebras). A Σ -algebra A has, for each sort s of Σ , a non-empty set A_s , the *carrier of sort* s, and for each Σ -function symbol $F: u \to s$, where $u = s_1 \times \cdots \times s_m$, a (not necessary total) function $F^A: A^u \rightharpoonup A_s$, where

 $A^u =_{df} A_{s_1} \times \cdots \times A_{s_m}.$

The algebra A is total if F^A is total for each Σ -function symbol F. Otherwise it is partial.

We will write $\Sigma(A)$ for the signature of an algebra *A*.

4.2. Topological algebras

Definition 4.2.1 (*Continuity*). Given two topological spaces X and Y, a partial function $f : X \rightarrow Y$ is *continuous* if for every open $V \subseteq Y$,

 $f^{-1}[V] =_{df} \left\{ x \in X \mid x \in dom(f) \text{ and } f(x) \in V \right\}$

is open in X.

Definition 4.2.2 (*Topological partial algebra*). A topological partial algebra is a partial Σ -algebra with topologies on the carriers such that each of the basic Σ -functions is continuous, and the carriers \mathbb{B} and \mathbb{N} (if present) have the discrete topology.

Remark 4.2.3 (*Continuity of computable functions; the continuity principle*). The significance of the continuity of the basic functions of a topological algebra A is that it implies continuity of all **While** computable¹⁰ functions on A [20,21]. This is in accordance with the *Continuity Principle*, which can be expressed as

computability \implies *continuity*.

This principle is discussed in [20,22].

4.3. The algebra \mathcal{R} of reals

In this paper, we will work with the following algebra:

algebra	\mathcal{R}
carriers	R, B, N
functions	$0_{R}, 1_{R}: \rightarrow \mathbb{R}$,
	$plus_{R}, times_{R}: \ \mathbb{R}^2 \to \mathbb{R}$
	$inv_{R}: \ \mathbb{R} riangleq \mathbb{R}$,
	$0_{N}: o \mathbb{N}$,
	$suc_N: \mathbb{N} \to \mathbb{N}$
	tt, ff : $ ightarrow \mathbb{B}$,
	and, or : $\mathbb{B}^2 o \mathbb{B}$,
	not : $\mathbb{B} \to \mathbb{B}$,
	eq_N , $less_N$: $\mathbb{N}^2 \to \mathbb{B}$
	$eq_{R}, \ less_{R} \colon \ \mathbb{R}^2 \rightharpoonup \mathbb{B}$

The signature $\Sigma(\mathcal{R})$, with sorts real, bool, and nat, can be inferred from the above.

Remarks 4.3.1.

- (1) \mathcal{R} contains three carriers: \mathbb{R} , \mathbb{N} and \mathbb{B} , of sorts real, nat and bool respectively.
- (2) \mathcal{R} contains (as retracts) the field of reals, the naturals with 0 and successor, and the booleans with their standard operations, including equality and order on \mathbb{R} and \mathbb{N} .
- (3) \mathcal{R} is a partial algebra, with the following partial basic functions: inv_B, eq_B and less_B, where for $x, y \in \mathbb{R}$:

$$\begin{split} & \mathsf{inv}_{\mathsf{R}}(x) \simeq \begin{cases} 1/x & \text{if } x \neq 0 \\ \uparrow & \text{otherwise,} \end{cases} \\ & \mathsf{eq}_{\mathsf{R}}(x, y) \simeq \begin{cases} \uparrow & \text{if } x = y \\ \mathsf{ff} & \text{otherwise,} \end{cases} \\ & \mathsf{less}_{\mathsf{R}}(x, y) \simeq \begin{cases} \mathsf{tt} & \text{if } x < y \\ \mathsf{ff} & \text{if } x > y \\ \uparrow & \text{if } x = y. \end{cases} \end{split}$$

By contrast, the boolean functions on $\mathbb N\colon eq_N$ and $less_N,$ are total. The reasons for these semantic definitions will now be discussed.

Discussion 4.3.2 (*The topological partial algebra* \mathcal{R}). \mathcal{R} is a topological partial algebra when \mathbb{R} is given its usual topology, and \mathbb{B} and \mathbb{N} the discrete topology. This motivates our semantic definitions of the partial functions inv_R, eq_R and less_R in \mathcal{R} .

Note that the total versions of these functions are not continuous, as can easily be checked. By contrast, the total functions eq_N , $less_N$ on \mathcal{N} ar trivially continuous, because of the discrete topology on \mathbb{N} . Continuity of the basic functions of \mathcal{R} , and hence of all **While**-computable functions on \mathcal{R} , accords with the Continuity Principle (see Remark 4.2.3).

Note that \mathcal{R} is *standard* in the sense that it contains the booleans with their standard operations. This is clearly important for the purpose of programming on \mathcal{R} . \mathcal{R} is also N-*standard* in the sense that it contains the naturals (with 0 and successor). This is important for considerations of computation on \mathcal{R} [21–23].

¹⁰ See Section 4.5.

4.4. The algebra \mathcal{R}^*

 \mathcal{R}^* is formed from \mathcal{R} by adding the carrier \mathbb{R}^* (of sort real*) consisting of all *finite sequences* or *arrays* of reals, together with certain standard constants and operations for the empty array, for updating arrays, etc. [20,21,23].

The significance of arrays for computation is that they provide finite but unbounded memory. The reason for introducing the starred sort real* is the lack of effective coding of finite sequences from \mathbb{R} (unlike the case with \mathbb{N} and \mathbb{B}).

Although the use of \mathcal{R}^* is convenient for computational purposes, it does not affect the computational strength of abstract models on \mathbb{R} , as we will see (Section 4.8).

We will be using the topological partial algebras \mathcal{R} and \mathcal{R}^* in the rest of the paper.

4.5. The **While** programming language

Our abstract models of computation on \mathbb{R} are based on the *While* programming language and its extensions, applied to \mathcal{R} [20–23].

We review the *syntax* of the *While* language over a standard signature Σ . We use ' \equiv ' to denote syntactic identity between two expressions:

- Σ -variables: There are variables x^s , ... of each Σ -sort s.
- Σ -terms: The set of Σ -terms of sort *s*, denoted t^s, \ldots , is generated by

$$t^s ::= \mathbf{x}^s \mid F(t_1^{s_1}, \ldots, t_m^{s_m})$$

where *F* is a Σ -function symbol of type $s_1 \times \cdots \times s_m \rightarrow s$.

We also write t:s to indicate that t is a Σ -term of sort s. More generally, we write t:u if t is a tuple of Σ -terms of product type u. We write b, \ldots for boolean terms, i.e. terms of sort bool.

• *Σ*-statements *S*,... are generated by:

 $S ::= \text{skip} \mid x := t \mid S_1; S_2 \mid \text{if } b \text{ then } S_1 \text{ else } S_2 \text{ fi} \mid \text{while } b \text{ do } S_0 \text{ od}$

where x := t denotes simultaneous assignment, i.e. for some m > 0, $x \equiv (x_1, ..., x_m)$ and $t \equiv (t_1, ..., t_m)$ are variable and term tuples of the same product type, with the condition that $x_i \neq x_j$ for $i \neq j$.

• Σ -procedures P, \ldots have the form:

 $P \equiv \operatorname{proc} D$ begin S end

where the statement S is the body and D is the variable declaration of the form

 $D \equiv in a : u out b : v aux c : w$

where a, b and c are tuples of input variables, output variables and auxiliary variables respectively. We stipulate:

- (i) a, ${\rm b}$ and ${\rm c}$ each consist of distinct variables, and they are pairwise disjoint; and
- (ii) every variable in S must be declared in D (among a, b, c).
- If a : u and b : v, then P has type $u \to v$, written $P: u \to v$.

Now, for a standard Σ -algebra A, the *semantics* of the *While* language over A can be defined in a well-known way [20, 21,23] which will not be repeated here. In particular, the meaning of a *While*(Σ) procedure $P : u \rightarrow v$ is a partial function

 $P^A: A^u \rightharpoonup A^v.$

Then we define:

Definition 4.5.1 (While computable function).

- (a) A function $f: A^u \to A_s$ is said to be *computable* on A by a **While** procedure $P: u \to s$ if $f = P^A$.
- (b) *While*(*A*) is the class of functions *While* computable on *A*.

4.6. While approximability

From now on, we restrict our attention to the real algebra \mathcal{R} . We consider another paradigm of abstract computability related to the *While* language over \mathcal{R} .

Given a procedure P : nat × real → real, we write for any n,

 $P_n^{\mathcal{R}} =_{df} P^{\mathcal{R}}(n, \cdot) : \mathbb{R} \to \mathbb{R}.$

Now let $f: \mathbb{R} \to \mathbb{R}$ satisfy the two global assumptions, with (U_{ℓ}) an effective exhaustion of U = dom(f), and let $P: nat \times real \to real$ be a *While* procedure over \mathbb{R} .

Definition 4.6.1 (*Local uniform While approximability*). f is locally uniformly **While** approximable by P over \mathcal{R} if

- (1) for all *n*, $dom(P_n^{\mathcal{R}}) \supseteq U$, and
- (2) the sequence $(P_n^{\mathcal{R}} \upharpoonright U)$ converges to *f* effectively locally uniformly w.r.t. (U_ℓ) (cf. Definition 2.3.7).

4.7. While programming with countable choice

We extend the **While** language over \mathcal{R} to the **While**CC language by adding a new assignment statement:

 $x := choose z : P(z, \ldots)$

[22,23] where x and the 'choose' variable z have sort nat, and P(z,...) is a *semicomputable predicate* of z (and other variables), i.e., the halting set of a **WhileCC** procedure with z among its input variables.

Then 'choose z : P(z,...)' selects some value k such that $P(\bar{k},...)$ is true if any such k exists, and is undefined otherwise. In the abstract semantics [22], the meaning of 'choose z : P(z,...)' is the set of *all* such k's (hence "countable choice"). Any concrete model will select a particular k, according to the implementation.

The abstract semantics for **WhileCC** associates with a **WhileCC** procedure P : real \rightarrow real \rightarrow (many-valued) function:

$$P^{\mathcal{R}}: \mathbb{R} \to \mathcal{P}^+_{\omega}(\mathbb{R}^{\uparrow}),$$

where $\mathcal{P}^+_{\omega}(X)$ is the set of all countable *non-empty* subsets of *X*, and $\mathbb{R}^{\uparrow} = \mathbb{R} \cup \{\uparrow\}$, where ' \uparrow ' represents a divergent computation.

Next we consider WhileCC approximable computability or WhileCC approximability. Let

 $P: \mathsf{nat} \times \mathsf{real} \to \mathsf{real}$

be a WhileCC procedure. Again we write

 $P_n^{\mathcal{R}} =_{df} P^{\mathcal{R}}(n, .): \mathbb{R} \to \mathcal{P}_{\omega}^+(\mathbb{R}^{\uparrow}).$

Definition 4.7.1 (*WhileCC* approximability to a single-valued function). A function $f: \mathbb{R} \to \mathbb{R}$ is approximable by a *WhileCC* procedure P on \mathcal{R} iff for all $n \in \mathbb{N}$ and all $x \in \mathbb{R}$:

(i) $x \in dom(f) \implies \uparrow \notin P_n^{\mathcal{R}}(x) \subseteq Nbd(f(x), 2^{-n})$, and (ii) $x \notin dom(f) \implies P_n^{\mathcal{R}}(x) = \{\uparrow\}.$

Remark 4.7.2. The concept of *WhileCC* approximability, unlike that of local uniform *While* approximability (Definition 4.6.1), does not refer to the exhaustion (U_{ℓ}) of *dom*(f).

4.8. While* and WhileCC* computability

Recall the definition of the array algebra \mathcal{R}^* with signature Σ^* (Section 4.4).

A **While**^{*}(Σ) procedure is a **While**(Σ^*) procedure with the restriction that the array variables (i.e. variables of sort real^{*}) are used only as auxiliary variables, not for input or output.

The **While**^{*} language is clearly more convenient than **While** for writing programs over \mathcal{R} . However, it is not (in theory) stronger than **While** for defining functions on \mathcal{R} ; in fact (writing **While**^{*}(\mathcal{R}) for the set of functions **While**^{*} definable on \mathcal{R}):

 $While^*(\mathcal{R}) = While(\mathcal{R})$

by [21, §4.7], adapting the proof there to partial algebras. Similarly we can define the language

While $CC^*(\Sigma) = While CC(\Sigma^*)$

and again show that

```
While CC^*(\mathcal{R}) = While CC(\mathcal{R}).
```

Analogously, we can also define the concepts of $While^*$ and $WhileCC^*$ approximability on \mathbb{R} .

Hence in the rest of the paper, we will write *While*^(*) and *While*CC^(*) to refer to these languages either with or without arrays.

4.9. Equivalence of abstract and concrete computability

We come now to the third and last Equivalence Lemma, for abstract (**WhileCC**^(*)(\mathcal{R})) approximability and concrete ($\overline{\alpha}$ -tracking) computability. It can also be viewed as a *completeness* result for abstract (**WhileCC**^(*)(\mathcal{R})) vs. concrete (tracking) computability.

Recall our global assumptions on $f: \mathbb{R} \rightarrow \mathbb{R}$.

Equivalence Lemma 3 (Abstract and concrete computability). The following are equivalent:

(i) f is $\overline{\alpha}$ -computable

(ii) f is **WhileCC**^(*)(\mathcal{R}) approximable.

This was proved in [22,23] for complete separable metric algebras.¹¹ It can be seen that the conditions listed in [22, Thm. C^+] and [23, Thm. 4.2.13] are satisfied here. (See also the discussion in Section 5.2 below.)

Remark 4.9.1. Interestingly, the proof of this Equivalence Lemma requires the global (domain exhaustion and continuity) assumptions for f, even though the definitions of $\overline{\alpha}$ -computability and **WhileCC**^(*) approximability (unlike those of GL computability and multipolynomial approximability) do not mention them (cf. Remark 4.7.2).

In Section 5.2 we return to this Equivalence Lemma, to show how it holds for functions $f: \mathbb{R}^m \to \mathbb{R}$ for all $m \ge 1$ with a weaker domain exhaustion assumption.

Remark 4.9.2. In [23] it was proved that for *total* $f: \mathbb{R} \to \mathbb{R}$, local uniform *While* approximability of f over \mathcal{R}_t corresponds to $\overline{\alpha}$ -computability of f, where \mathcal{R}_t is the total topological algebra formed from \mathcal{R} by removing the partial operations eq_R and less_R, and replacing inv_R by the (weaker) inverse operation on \mathbb{N} :

$$\operatorname{inv}_{\mathsf{N}}: \mathbb{N} \to \mathbb{R},$$

where

$$\operatorname{inv}_{\mathsf{N}}(n) = \begin{cases} 1/n & \text{if } n \neq 0\\ 0_{\mathsf{R}} & \text{if } n = 0 \end{cases}$$

$$\tag{4.1}$$

which is total, but still continuous.

However we have been unable to prove (or disprove) such an Equivalence Lemma for *partial* functions f (satisfying the global assumptions).

5. Conclusion; future work

5.1. Conclusion: equivalence of all models

We have studied four models of computation of partial functions on the real numbers: two concrete, one abstract, and one based on multipolynomial approximation.

From the three Equivalence Lemmas (in Sections 2.5, 3.2 and 4.9) follows the Equivalence Theorem for these models (Section 1.3), for partial functions satisfying the two global assumptions of *domain exhaustion* and *continuity* (Section 1.2).

Over the past decade, as interest in computation on real, metric or other topological spaces, has grown, many models of computation have been proposed, developed and compared. To cite one among many examples, Stoltenberg-Hansen and Tucker [19] proved the equivalence of different types of concrete models for total functions, all based on effective representations by:

- algebraic domains [18],
- continuous domains [3,4],
- type 2 recursion [25],
- effective metric spaces [14], and
- computability structures [15].

One can also take a more radical approach to concrete models of computation. Such models, whatever their motivation, and however they are conceived and designed, rest upon *numberings*, which are mappings from \mathbb{N} onto some subset of the algebra *A*, typically dense in *A*. In these theories, the subsets are considered to be "finitistic" elements of *A*, precisely

 $^{^{11}\,}$ I.e. topological algebras where the topology is given by a metric.

because they are capable of being numbered. With this simple but fundamental viewpoint, we see that the equivalence between different models of computation can be expressed as *invariance* of computability between different numberings: if M_1 and M_2 are two models over A based on numberings α_1 and α_2 respectively, then equivalence between α_1 and α_2 implies that M_1 and M_2 have the same computable functions. Two interesting studies of the equivalence of numberings for familiar topological spaces are by Hertling [10] on the reals, and Blanck, Stoltenberg-Hansen and Tucker [2] on more general structures. A more abstract approach to invariance of computational models is given by Bauer and Blanck [1] using realizability theory.

5.2. A weaker domain exhaustion assumption

Can our domain exhaustion assumption be weakened? Consider the three Equivalence Lemmas, which together form the Equivalence Theorem:

- EL1 (Section 2.5) GL-comp. \iff multipoly approx.
- EL2 (Section 3.2) GL-comp. $\iff \overline{\alpha}$ -tracking comp.

EL3 (Section 4.9) $\overline{\alpha}$ -tracking comp. \iff *WhileCC*^{*} approx.

Note first that EL3 was presented and proved in [22,23] (as a "completeness theorem" for abstract vs concrete computation), in the much more general case of functions f on a complete separable metric algebra A, where

(a) the domain U of f is the union of an *open exhaustion* (U_{ℓ}) , i.e., an increasing sequence of open sets, which is (i) *semi-effective*, in the sense that the relation

 $\{(x, \ell) \in A \times \mathbb{N} \mid x \in U_\ell\}$

is While*-semicomputable;

(ii) *effectively open*, in the sense that there is a *While*-computable function $\gamma: A \times \mathbb{N} \to \mathbb{N}$ such that for all ℓ and all $a \in U_{\ell}$,

Nbd $(a, 2^{-\gamma(a,\ell)}) \subset U_{\ell}$

(b) f is effectively locally uniformly continuous w.r.t. this exhaustion.

Specializing again to the case $A = \mathcal{R}$: assumption (a) is easily seen to be *weaker* than the domain exhaustion assumption given in this paper (Definitions 2.2.1 and 2.2.2), since in the above definition,

- it is not assumed that each U_{ℓ} is a finite union of rational intervals;
- it is not assumed that $\overline{U_{\ell}} \subseteq U_{\ell+1}$, only that $U_{\ell} \subseteq U_{l+1}$;
- the condition (ii) of *effective openness* is derivable from the domain exhaustion assumption given in this paper. (That is the essence of Lemma 3.2.1, used in the proof of EL2.)

We will call (a) the **weak domain exhaustion assumption**, and the conjunction of (a) and (b) the **weak global assumptions**. Then from the results in [22,23], it follows that the Equivalence Lemma EL3 given here in Section 4.9 can be re-stated

using the weak global assumptions, and can also be generalized to functions $f: \mathbb{R}^m \to \mathbb{R}$ for m > 1. Next, considering EL2: note, in this regard, that the definition of GL-computability as given here (Section 2.3) can be

easily adapted to the case of functions $f: \mathbb{R}^m \to \mathbb{R}$ for m > 1.¹² Then the proof of this equivalence given in Section 3.2 can be easily generalized to the case of such functions satisfying the weak global assumptions above.

Combining these versions of EL2 and EL3, we arrive at the following "generalization" of the equivalence theorem: (cf. the Equivalence Theorem for all four computation models (i)–(iv) given in Section 1.3):

Generalized equivalence theorem. Suppose $f: \mathbb{R}^m \to \mathbb{R}$ (m > 0) has a domain with an effective exhaustion (U_ℓ) which satisfies the weak domain exhaustion assumption, and f is effectively locally uniformly continuous w.r.t. (U_ℓ) . Then the following are equivalent:

(i) f is GL-computable w.r.t. (U_{ℓ}) ,

(ii) f is tracking computable,

(iv) f is **WhileCC**^{*} approximable on \mathcal{R} .

This is called a "generalized" equivalence theorem because (a) it applies to functions of arity $m \ge 1$, and (b) the domain exhaustion assumption here is weaker than that used in this paper (as stated above).

¹² Such a definition is given in [15, Ch. 0, Sec. 3] in the case that the domain of f is a closed *m*-dimensional rectangle.

Conspicuously absent here, however, is the multipolynomial model:

(iii) *f* is effectively locally uniformly multipolynomially approximable w.r.t. (U_{ℓ}) .

To include this among the equivalences, we seem to need the "strong" effective exhaustion assumption (Definitions 2.2.1(2)), and even then it only works for m = 1 (Section 1.2, point (3)).

Remark 5.2.1 (*Further generalization of "effective exhaustion"*). It seems that the generalized equivalence theorem presented above can be generalized even further (still excluding the multipolynomial model), so as to apply to partial functions on *complete separable metric algebras*, as used in proving EL3 ("abstract/concrete completeness") in [22,23]. It only remains to give a generalized definition of GL-computability for such functions, and then generalize the proof of EL2 (Section 3.2) accordingly. This seems unproblematic.

Remark 5.2.2 (*Consistent generalization of "effective exhaustion"*). The generalization of the notion of "effective exhaustion" used above does *not* reduce to the notion developed in this paper when n = 1. It is strictly weaker, as indicated above. However, in order to include (for example) the multipolynomial approximability model in an equivalence theorem for functions $f: \mathbb{R}^m \to \mathbb{R}$ with m > 1, we want a notion of "effective exhaustion" in m dimensions that reduces, when m = 1, to the one developed in this paper (Definitions 2.2.1 and 2.2.2). To formulate the correct notion for this is a challenging problem.¹³

5.3. Invariance of continuity assumption

For two of our four models: GL-computability and multipolynomial approximability, the definition of computability of a function f with domain U depends on the choice of effective exhaustion (U_ℓ) of U, since the global assumption (b) (cf. Section 2.2) requires effective local uniform continuity of f with respect to this exhaustion.

It is therefore desirable to have an *invariance* result of the form that if global assumption (b) holds for *any* effective exhaustion of dom(f), then it holds for *all* effective exhaustions of dom(f). This is given by the following proposition.

Proposition 5.3.1 (Invariance of global assumption (b) w.r.t. (a)). Given $f : \mathbb{R} \to \mathbb{R}$ with domain U, let (U_{ℓ}) and (V_{ℓ}) be two effective exhaustions of U. If f is effectively locally uniformly continuous w.r.t. (U_{ℓ}) , then f is also effectively locally uniformly continuous w.r.t. (V_{ℓ}) .

Proof. Consider any stage V_{ℓ} of the exhaustion (V_{ℓ}) . Since $\overline{V_{\ell}} \subseteq U$, which is a union of expanding open sets U_m , by compactness of $\overline{V_{\ell}}$ there exists *m* such that

(5.1)

$$\overline{V_{\ell}} \subseteq U_m$$
.

By assumption, f is uniformly continuous on U_m , and so by (5.1) f is also uniformly continuous on V_ℓ .

Further, *m* in (5.1) can be effectively found from ℓ (as can be seen from Definition 2.2.2), *i.e.*, $m = g(\ell)$ for some total recursive *g*. Hence *f* is effectively locally uniformly continuous w.r.t. (V_ℓ), with a modulus *M*' obtainable from the modulus *M* for *f* w.r.t. (U_ℓ) (cf. Definition 2.2.6):

$$M'(k,\ell) = M(k,g(\ell)). \quad \Box$$

5.4. Future work

Three interesting problems left open by this paper are:

- (1) Generalizing the full Equivalence Theorem (including the effective multipolynomial model) to functions $f : \mathbb{R}^m \to \mathbb{R}$ for m > 1.
 - Two difficulties here are how to generalize to m > 1:
 - the proof in Equivalence Lemma 1 (Section 2.5) of the implication

multipolynomial approx. \implies GL computability

which, as it stands, requires extending the domain of the multipolynomial approximants to dom(f) by linear interpolation;

and more fundamentally:

• the *definition* of the concept of (effective) exhaustion (Definitions 2.2.1 and 2.2.2).¹⁴

¹³ We return to this in Section 5.4.

¹⁴ This has recently been accomplished in [6], where (briefly) the rational intervals forming the components of the stages of an exhaustion are replaced (for m > 1) by computable open basic semialgebraic sets.

(2) Investigating the

Conjecture. The two global assumptions are satisfied by all elementary functions on the reals.

Here the elementary functions on \mathbb{R} are functions defined by expressions built up from the variable *x* and constants for computable reals, by applying (repeatedly) the four field operations, *n*-th roots, the exponential and trigonometric functions, and their inverses.

This is a very interesting class of functions, investigated by, among others, Hardy [9].¹⁵ Richardson [16] proved, for a naturally defined subclass *C* of this class, the unsolvability of the identity problem (for *f*, *g* in *C*, does f = g?) and the integration problem (for *f* in *C*, is the integral of *f* in *C*?).

Note that the function $\sqrt[n]{x}$, for *n* even, is defined on the interval $[0, \infty)$, which is not open (and hence cannot be the union of an open exhaustion). This is easily remedied by defining its value to be 0 for x < 0, resulting in a total, effectively uniformly continuous function. Of course, many elementary functions (e.g. 1/x, $\log x$, $\tan x$) cannot be "totalized" in this way.

One can then easily prove that the domains of all elementary functions are open, by induction on the complexity of their defining expressions. However the above conjecture remains to be proved.

(3) Determining the status of the *While*^(*) approximability model:

whether (or under what conditions) it is equivalent to the four models on \mathbb{R} studied here. (See Remark 4.9.2.)

Acknowledgements

This article developed out of the first author's MSc thesis [5]. We are grateful to Jacques Carette, John Tucker and four anonymous referees for very helpful comments on earlier drafts.

References

- [1] A. Bauer, J. Blanck, Canonical effective subalgebras of classical algebras as constructive metric completions, J. Univers. Comput. Sci. 16 (18) (2010) 2496–2522.
- [2] J. Blanck, V. Stoltenberg-Hansen, J.V. Tucker, Stability of representations for effective partial algebras, Math. Log. Q. 57 (2011) 217–231.
- [3] A. Edalat, Domain theory and integration, Theor. Comput. Sci. 151 (1995) 163-193.
- [4] A. Edalat, Dynamical systems, measures, and fractals via domain theory, Inf. Comput. 120 (1995) 32-48.
- [5] Ming Quan Fu, Models of computability of partial functions on the reals, MSc thesis, Department of Computing & Software, McMaster University, 2007, Technical report CAS-08-01-JZ, January 2008.
- [6] Ming Quan Fu, Characterizations of semicomputable sets, and computable partial functions, on the real plane, PhD thesis, Department of Computing & Software, McMaster University, 2014, archived in DSpace at: http://hdl.handle.net/11375/16066.
- [7] A. Grzegorczyk, Computable functions, Fundam. Math. 42 (1955) 168–202.
- [8] A. Grzegorczyk, On the definitions of computable real continuous functions, Fundam. Math. 44 (1957) 61–71.
- [9] G.H. Hardy, The Integration of Functions of a Single Variable, Cambridge Tracts in Mathematics and Mathematical Physics, vol. 2, Cambridge University Press, 1905, available as an eBook at openlibrary.org.
- [10] R. Hertling, A real number structure that is effectively categorical, Math. Log. Q. 45 (2) (1999) 147-182.
- [11] S.C. Kleene, Introduction to Metamathematics, North Holland, 1952.
- [12] D. Lacombe, Extension de la notion de fonction récursive aux fonctions d'une ou plusieurs variables réelles, I, II, III, C. R. Acad. Sci. Paris 240 (1955) 2470–2480; C. R. Acad. Sci. Paris 241 (13–14) (1955) 151–153.
- [13] A.I. Mal'cev, Constructive algebras I, in: The Metamathematics of Algebraic Systems. A.I. Mal'cev, Collected Papers: 1936–1967, North Holland, 1971, pp. 148–212.
- [14] Y.N. Moschovakis, Recursive metric spaces, Fundam. Math. 55 (1964) 215–238.
- [15] M.B. Pour-El, J.I. Richards, Computability in Analysis and Physics, Springer-Verlag, 1989.
- [16] D. Richardson, Some unsolvable problems involving functions of a real variable, J. Symb. Log. 33 (1968) 514-520.
- [17] H. Rogers Jr., Theory of Recursive Functions and Effective Computability, McGraw-Hill, 1967.
- [18] V. Stoltenberg-Hansen, J.V. Tucker, Effective algebras, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, vol. 4, Oxford University Press, 1995, pp. 357–526.
- [19] V. Stoltenberg-Hansen, J.V. Tucker, Concrete models of computation for topological algebras, Theor. Comput. Sci. 219 (1999) 347–378.
- [20] J.V. Tucker, J.I. Zucker, Computation by 'while' programs on topological partial algebras, Theor. Comput. Sci. 219 (1999) 379-420.
- [21] J.V. Tucker, J.I. Zucker, Computable functions and semicomputable sets on many-sorted algebras, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, vol. 5, Oxford University Press, 2000, pp. 317–523.
- [22] J.V. Tucker, J.I. Zucker, Abstract versus concrete computation on metric partial algebras, ACM Trans. Comput. Log. 5 (2004) 611–668.
- [23] J.V. Tucker, J.I. Zucker, Computable total functions, algebraic specifications and dynamical systems, J. Log. Algebr. Program. 62 (2005) 71–108.
- [24] J.V. Tucker, J.I. Zucker, Abstract versus concrete computability: the case of countable algebras, in: V. Stoltenberg-Hansen, J. Väänänen (Eds.), Logic Colloquium '03, Proc. Annual European Summer Meeting, Association for Symbolic Logic, Helsinki, August 2003, in: Lecture Notes in Logic, vol. 24, Association for Symbolic Logic, 2006, pp. 377–408.
- [25] K. Weihrauch, Computable Analysis: An Introduction, Springer, 2000.

¹⁵ Hardy also included functions y = f(x) implicitly defined by polynomial equations in x and y.