Theory Comput Syst (2012) 51:65–84
DOI 10.1007/s00224-011-9374-z

Representation Theorems for Analytic Machines and Computability of Analytic Functions

Tobias Gärtner · Günter Hotz

Published online: 30 November 2011

Abstract We present results concerning analytic machines, a model of real computation introduced by Hotz which extends the well-known Blum, Shub and Smale machines (BSS machines) by infinite converging computations. The well-known representation theorem for BSS machines elucidates the structure of the functions computable in the BSS model: the domain of such a function partitions into countably many semi-algebraic sets, and on each of those sets the function is a polynomial resp. rational function. In this paper, we study whether the representation theorem can, in the univariate case, be extended to analytic machines, i.e. whether functions computable by analytic machines can be represented by power series in some part of their domain. We show that this question can be answered in the negative over the real numbers but positive under certain restrictions for functions over the complex numbers. We then use the machine model to define computability of univariate complex analytic (i.e. holomorphic) functions and examine in particular the class of analytic functions which have analytically computable power series expansions. We show that this class is closed under the basic analytic operations composition, local inversion and analytic continuation.

Keywords Analytic machines · Analytic functions · Computable complex functions · Computability of analytic functions · Real computability

1 Introduction

In order to define computable functions on the real and complex numbers, several different proposals have been made. One prominent approach, introduced by Blum et

T. Gärtner (✉) · G. Hotz
Fachbereich für Informatik, Universität des Saarlandes, Saarbrücken, Germany
e-mail: gaertner@mpi-inf.mpg.de

G. Hotz
e-mail: hotz@cs.uni-sb.de

al. [3] defines a machine model that deals with real numbers as atoms. A computation on such a *BSS machine* is a finite sequence of arithmetic operations and conditional branches. Blum et al. and others have developed an elaborate complexity theory for these machines, and e.g. posed the $P \neq NP$ problem over the real numbers. In [1], a comprehensive description of the theory is presented. A main characteristic of the functions computable within this model is that those functions are, in a sense, piecewise defined rational functions. Elementary transcendental functions like the exponential function or trigonometric functions are not computable in this model.

Another successful approach, *recursive analysis*, defines computability over the real and complex numbers by allowing infinite computations on Turing machines, for instance by Grzegorczyk [7]. In the *Type 2 theory of effectivity (TTE)*, continuous structures like the real and complex numbers are mapped to the tape of the machine by means of representations, or naming systems, and the Turing machines perform infinite converging computations. An overview over the theory can be found in Weihrauch's book [19]. A main characteristic of the functions defined in this framework is that they are automatically continuous, and therefore discontinuous functions like the Heaviside function are not computable, at least when using the standard representations. A comprehensive overview over many different approaches to computability over the real and complex numbers, including many which have not been mentioned here, can be found in [21].

There are also approaches which try to connect the different approaches, like Brattka and Hertling's feasible real RAM. In this paper, we continue the work of Hotz et al. [4, 5, 9] and use the approach of the *analytic machines*. These machines are a kind of synthesis of BSS machines and recursive analysis, since they allow exact arithmetic operations with real numbers as atoms on the one hand and infinite converging computations on the other. One of the main results of the theory is Chadzelek's hierarchy theorem [4], which in particular states that analytic machines are not closed under composition. The model can be, however, restricted to perform rational arithmetic on rounded real inputs. A certain class, the *quasi-strongly* δ-$\mathbb{Q}$ *analytic functions* subsumes both BSS computable and Type 2 computable functions and is closed under composition. On the other hand, Ziegler [22] has shown that quasi-strongly δ-$\mathbb{Q}$-analytic functions can be computed by *nondeterministic* Type 2 machines, and that this class of functions can be defined within TTE by a suitable representation.

The topic of the present paper is the interrelation of analytic machines on the one hand and analytic functions in the classical mathematical sense of the word (i.e. functions representable by power series) on the other.

In the first part of this paper, we study to which extent functions that are computable by analytic machines are also representable by power series. The structure of the functions computable by BSS machines (which are analytic machines with finite running times) is elucidated by the well-known *representation theorem* for BSS computable functions (often called *path decomposition theorem* [1, Sect. 2.3, Theorem 1]): The domain of a BSS computable function decomposes into countably many semi-algebraic sets, and on each of these sets the function is a polynomial or rational function. Considering that a computation of a BSS machine is a sequence of arithmetic operations and branches, this theorem can be appreciated quite intuitively. The partition of the domain of a function is given by the different computation paths of the inputs. The theorem is of central importance for the computability and complexity theory of BSS machines, as it gives a mathematical tool for algebraic analysis of computations.

Since BSS computable functions are representable by polynomials (or rational functions) on parts of their domain and since analytic machines generalize the finite computations of BSS machines to infinite computations, the question arises to which extent functions computable by analytic machines are representable by the generalization of polynomials, namely power series, on parts of their domains. First, it is easily seen that in general, no direct analogue of the representation theorem will be possible, since by infinite calculations the domain of a function can be partitioned into uncountably many parts, and none of the parts is an open set. Since definition of represented power series really only makes sense on open sets (or at least sets with nonempty open interior), the first restriction one has to make is to assume that there is a computation path with a domain that actually has nonempty open interior.

It turns out that even then it is important to distinguish between functions defined over the real numbers on the one hand and the complex numbers on the other. Over the real numbers, we show that the representation theorem does not generalize to analytic machines, i.e. there are functions computable by analytic machines without branching operations that are nowhere representable by power series. On the other hand, we show that over the complex numbers the situation is different. Given a function computable by an analytic machine over the complex numbers (only being able to perform *complex arithmetic*), such that there are only finitely many branching operations on each computation path, we show that such a function is always representable by a power series on a part of its domain.

While the first part of the paper studies to which extent functions computable by analytic machines are representable by power series, the second part starts from the other end. We consider the functions of which we already know that they are representable by power series, i.e. the well known class of holomorphic or complex analytic functions. Complex function theory has been one of the most successful parts of mathematics and is, by many, also perceived as one of the most beautiful. The fact that each analytic function can be expressed by a power series also suggests that these functions have interesting computability properties, as a power series actually can be regarded as a specification of computation. We use the model of analytic machines in order to define computable analytic functions. We define the class of (analytically) computable analytic function and the class of coefficient computable functions, which are those functions whose power series expansion is computable by an analytic machine. We show that coefficient computable analytic functions are also computable. Furthermore, we show that the class of coefficient computable analytic functions is closed under composition, local inversion and analytic continuation. In view of the fact that analytic machines are not closed under composition, the closure properties of coefficient computable functions were not entirely expected.

The computability of analytic functions has already been studied in the framework of TTE, for example Ker-I Ko [10] and Müller [11] have studied polynomial time computable analytic functions, and Müller [12] has studied constructive aspects of analytic functions. More recently, Rettinger has studied continuations of analytic functions [16] and Riemann surfaces [15] in the framework of TTE. In [12], Müller shows that in the framework of TTE, the power series expansions of Type 2 computable analytic functions are computable, and conversely, that a function is computable if its power series is computable. He goes further and examines which information is necessary for these results to be constructive. Since in contrast to Type 2 machines, the convergence speed of analytic machines is not known, we needed to employ different methods than those used in [12].

2 Definition and Properties of Analytic Machines

The machine model underlying this work is a register machine model, which is similar to the model of Blum et al. (BSS machines). For finite computations, the models have the same computational power. In a sense, the analytic machines extend the BSS machines by infinite converging computations.

In the following section, we will briefly present the model of analytic machines. The definition has been adapted and shortened from [4]. For a more detailed description we refer the reader to this work.

2.1 Machine Model

Remark 2.1 The reader who is not interested in formal details and who is familiar with the BSS model of computation can easily skip the following definition or just review Figs. 1 and 2. The gist of the definition of the analytic machine is that it is a register machine model that has exact arithmetic operations and branching conditions over its base field, and that it allows for infinite converging computations.

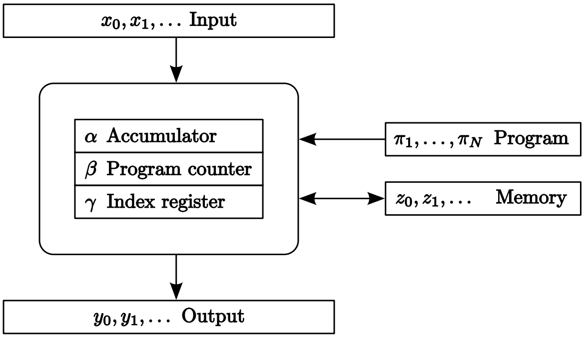
First, we give the definition of our mathematical machine that allows us to reason formally about computations and the functions computed by a machine. A mathematical machine over an alphabet A (not necessarily finite) is a tuple $\mathcal{M} = (K, K_\mathrm{i}, K_\mathrm{f}, K_\mathrm{t}, \Delta, A, \mathrm{in}, \mathrm{out})$. The set K is the *set of configurations* of $\mathcal{M}$, $K_\mathrm{i}, K_\mathrm{f}, K_\mathrm{t} \subset K$ are *initial*, *final* and *target configurations*. The function $\Delta : K \to K$ with $\Delta|_{K_\mathrm{f}} = id_{K_\mathrm{f}}$ is the *transition function*, and the functions $\mathrm{in} : A^* \to K_\mathrm{i}$ and $\mathrm{out} : K \to A^*$ are *input* and *output functions* over A.

A sequence $b = (k_i)_{i\in\mathbb{N}} \in K^{\mathbb{N}}$ with $k_0 = \mathrm{in}(x)$ and $k_{i+1} = \Delta(k_i)$ is called *computation of* $\mathcal{M}$ applied to x. The computation is called *finite* or *halting* if there is an $n \in \mathbb{N}$ such that $k_n \in K_\mathrm{f}$, and the minimal n with this property is called the *length* of the computation and $\mathrm{out}(k_n)$ its *result*.

Assuming a metric on A^*, we can also consider infinite converging computations. If for a computation $b = (k_i)_{i\in\mathbb{N}}$ a target configuration is assumed infinitely often, and the limit $\lim_{n\to\infty} \mathrm{out}(k_{i_n})$ for the subsequence $(k_{i_n})_{n\in\mathbb{N}} \subset K_\mathrm{t}$ of target configurations exists, we call the computation *analytic*. In this case, this limit is the *result of the analytic computation* b, and $\mathrm{out}(k_{i_n})$ is called the *n-th approximation*. If for an input $x \in A$ the resulting computation is neither finite nor analytic, then we say that $\mathcal{M}$ *does not converge* on this input, otherwise we say that the *machine converges* on this input.

The *domain* $\mathcal{D}_M$ of a machine $\mathcal{M}$ is the set of all $x \in A^*$ such that the computation of $\mathcal{M}$ with input x is analytic or finite, and its *halting set* $\mathcal{H}_M$ is the set of all $x \in A^*$ such that its computation with input x is finite.

Fig. 1 Register machine



1. assignments
   - $\alpha := x_i$, $\alpha := z_i$, $y_i := \alpha$, $z_i := \alpha$, $i \in \mathbb{N} \cup \{\gamma\}$
   - $\alpha := c$, $c \in \mathcal{R}$
   - $\gamma := 0$
2. arithmetics
   - $\alpha := \alpha + z_i$, $\alpha := \alpha \cdot z_i$
   - $\alpha := -\alpha$, $\alpha := \frac{1}{\alpha}$
   - $\gamma := \gamma + 1$, $\gamma := \gamma \dot{-} 1$
3. branches
   `goto` m
   `if` $\alpha > 0$ `then goto m else goto n` ($\mathcal{R}$ ordered)
   `if` $\alpha \neq 0$ `then goto m else goto n` ($\mathcal{R}$ not ordered)
   `if` $|\alpha| > r$ `then goto m else goto n`, $r \in \mathbb{Q}$ ($\mathcal{R} = \mathbb{C}$)
   `if` $|\alpha| < r$ `then goto m else goto n`, $r \in \mathbb{Q}$ ($\mathcal{R} = \mathbb{C}$)
4. special instructions
   `end`, `print`, `exception`

Fig. 2 Set of instructions

On its domain, the machine $\mathcal{M}$ now gives rise to the definition of a function $\Phi_M : \mathcal{D}_M \to A^*$, the *function defined by* $\mathcal{M}$. For inputs x such that the computation of $\mathcal{M}$ with input x is finite or analytic with result $y \in A^*$, we define $\Phi_M(x) = y$.

The mathematical machine will now be specialized to register machines over the ring $\mathcal{R}$. These machines are random access machines that operate with elements of $\mathcal{R}$ as atoms. For generality, the definition will be made for any ring $\mathcal{R}$ containing the integers (endowed with a metric in the case of analytic machines). The machines will, however, only be used for the fields of rational, real and complex numbers $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$. The basic notion of the machines is depicted in Figs. 1 and 2. A machine consists of a control unit with an accumulator α, a program counter β and an index register γ. There is an infinite input tape $x = (x_0, x_1, \ldots)$ that can be read, an infinite output tape $y = (y_0, y_1, \ldots)$ that can be written on, and an infinite memory $z = (z_0, z_1, \ldots)$. Furthermore, each machine has a finite program $\pi = (\pi_1, \ldots, \pi_N)$ with instructions from the instruction set $\Omega = \Omega_{\mathcal{R}}$ shown in Fig. 2. The set of configurations of a machine with program π is given by the contents of its registers and memory cells: $K=\{k=(\alpha,\beta,\gamma,\pi,x,y,z)\mid \alpha\in\mathcal{R},\beta,\gamma\in\mathbb{N},x,y,z\in\mathcal{R}^{\mathbb{N}}\}$. Initial, final and target configurations are given by $K_{\mathrm{i}}=\{k\in K\mid \alpha=\gamma=0,\beta=1,\ \forall j: y_j,z_j=0\}$, $K_{\mathrm{f}}=\{k\in K\mid \pi_\beta=\mathtt{end}\}$ and $K_{\mathrm{t}}=\{k\in K\mid \pi_\beta=\mathtt{print}\}$. The instruction `print` is used to identify target configurations, and `exception` stops the machine in a non-halting configuration.

With the definition of our machine model, we now define computable functions.

Definition 2.2 Let $D\subset\mathcal{R}^*$. We call a function $f:D\to\mathcal{R}^*$ *analytically $\mathcal{R}$-computable* if there exists an $\mathcal{R}$-machine $\mathcal{M}$ such that $D\subset\mathcal{D}_{\mathcal{M}}$ and $f=\Phi_{\mathcal{M}}|_D$. We call f *$\mathcal{R}$-computable* if there exists an $\mathcal{R}$-machine $\mathcal{M}$ such that $D\subset\mathcal{H}_{\mathcal{M}}$ and $f=\Phi_{\mathcal{M}}|_D$.

A set $A\subset\mathcal{R}^*$ is called *(analytically) decidable* if its *characteristic function* χ_A is (analytically) computable.

Remark 2.3 The use of real constants greatly influences the computational power of $\mathbb{R}$-machines, see, e.g., [13]. In Sect. 3 we will see another example of the power of general use of real constants.

For this reason, over the real and complex numbers, we usually restrict use of constants to rational ones if not explicitly mentioned otherwise. Formally, the constant assignment $\alpha:=c$ is therefore restricted to rational numbers c. If we consider computable functions with the use of (general or certain sets of irrational) constants we explicitly state this in the form 'f is computable with general constants' or 'f is computable with the set of constants C'.

Remark 2.4 In the complex case, BSS machines are usually not endowed with branchings on absolute value comparisons. Since we regard these operations as natural when dealing with complex numbers, we also consider machines that are able to do such absolute value comparisons.

The reader should, however, be aware that this does change, in the complex case, the computational power of the model compared to the classical BSS theory. For example, the complex unit disk $\{z:|z|\le 1\}$ is decidable by a $\mathbb{C}$-machine in our sense, whereas it is not decidable by a BSS machine (cmp. [1], Sect. 2.3). A detailed comparison of the change in computational power by introducing absolute value comparisons would thus be very interesting, but is beyond the scope of the present paper.

2.2 Properties of Computable Functions

For the finitely computable functions, there is Blum et al.'s well-known representation theorem for $\mathbb{R}$-computable functions (see, e.g. [1, Sect. 2.3, Theorem 1]).

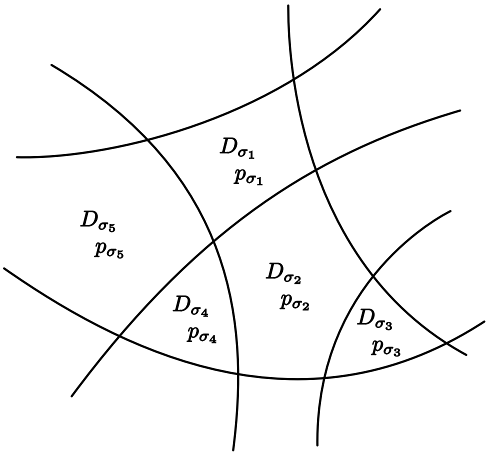
In order to state the theorem, we need to recall the notion of a semi-algebraic set: a *semi-algebraic* set is a finite union of subsets of a ring defined by systems of polynomial equations or inequalities.

Remark 2.5 In the case of the complex numbers $\mathbb{C}$, if we admit absolute value comparisons (which are not permitted for the original BSS machines), the term *semi-algebraic* is not precise: a set defined by an inequality $|f(z)|\le c$, f rational, is not a complex semi-algebraic set. Therefore, we call a set which is defined by finite unions of systems of polynomial equalities, inequalities or absolute value comparisons (i.e., $|r(x)| > c$, $|r(x)| < c$, $|r(x)| = c$, r rational function, $c \in \mathbb{Q}$) a *pseudo semi-algebraic set*.

Theorem 2.6 *Suppose $D \subset \mathbb{R}^n$ (or $\mathbb{C}^n$), and let f be an $\mathbb{R}$-computable (or $\mathbb{C}$-computable) function on D. Then D is the union of countably many semi-algebraic sets (or pseudo semi-algebraic sets in the complex case), and on each of those sets f is a rational function.*

Remark 2.7 If *the union of countably many (pseudo) semi-algebraic sets* in the above theorem is replaced by *an effective union of countably many (pseudo) semi-algebraic sets*, the theorem gives a necessary and sufficient characterization of the BSS-computable functions.

This theorem is a very useful tool for the analysis of computability and complexity of functions. In particular, while for Turing machines uncomputability is usually proved by reduction to the halting problem, for BSS machines showing that the properties of Theorem 2.6 are not fulfilled suffices for showing that a function is not computable. In this way, one easily can show that, e.g., the exponential function or the trigonometric functions (or, more general, all transcendental analytic functions) are not $\mathbb{R}$-computable or $\mathbb{C}$-computable.



The idea of the proof is quite simple: For a machine $\mathcal{M}$, consider the computation tree defined by the computations of $\mathcal{M}$ on different inputs. The nodes of the tree are labeled by the machine instruction given at the current state of the computation. At conditional branches, a node has two children, one for each outcome of the condition. Since, essentially, the only operations a machine can perform are arithmetical operations, the contents of the registers and memory of the machine are polynomials (resp. rational functions in case of divisions) of the input. The conditions of branching operations are therefore given by algebraic equalities or inequalities. Since each computation of a BSS machine is finite, the tree can have only countably many computation paths.

We denote computation paths by σ, and the computation path defined by an input x by $\sigma(x)$. The *domain D_σ of a computation path σ* is the set of all inputs x such that $\sigma = \sigma(x)$.

2.2.1 Undecidable Problems

For Turing machines, the well known halting problem is undecidable: There is no Turing machine that accepts as inputs the description of a Turing machine and decides if that given machine halts on a given input or not. This problem is, however, easily seen to be decidable by an analytic $\mathbb{R}$-machine.

The analogue of the halting problem for analytic machines is the *convergence problem*. That is, decide (analytically) whether a given analytic machine converges on its input or not. Formally, this problem is given by the function (where $\mathbf{M}_\mathbb{R}$ is a Gödel numbering of all analytic machines):

$$C : \mathbf{M}_\mathbb{R} \times \mathbb{R} \to \mathbb{R}, \qquad f(\mathcal{M}, x) = \begin{cases} 1 & \text{converges with input } x \\ 0 & \text{else} \end{cases}$$

By diagonalization, it is easy to see that this function is not analytically computable [4]. If we compose several analytic machines, however, the convergence problem becomes decidable. Intuitively, the term *composition* in the context of analytic machines means that an analytic machine uses the result of another analytic machine as input. Formally it is based on the computed functions: we say that a function f is *computable by i composed analytic machines* if there are i $\mathbb{R}$-analytically computable functions $f_1, \dots, f_i$ such that $f = f_i \circ \cdots \circ f_1$.

Chadzelek and Hotz [4] have shown that the convergence problem for analytic machines is computable by the composition of *three* analytic machines. Chadzelek has pointed out that it is possible to decide the problem with two analytic machines, but only if infinite intermediate results are used. The following theorem improves this result and shows that *two* machines suffice to decide the convergence problem, thus answering the open question of Chadzelek [5].

Theorem 2.8 *The convergence problem for analytic machines over the real numbers is decidable by two composed analytic machines.*

Proof We give the description of two machines $\mathcal{M}_1$ and $\mathcal{M}_2$ such that the composition of their computed functions decides the convergence problem.

Let $(a_n)_{n\in\mathbb{N}}$ be the sequence of outputs of target configurations of the input machine $\mathcal{M}$. The goal is to check whether (a_n) is a Cauchy sequence. Let $b_k := \sup_{n>m\geq k} |a_n - a_m|$. Then, the sequence (a_n) is convergent iff $\lim_{k\to\infty} b_k = 0$. In order to decide convergence of a sequence to zero, it is sufficient to consider the sequence formed of the numbers rounded to the next higher power of two. Therefore we define $r(x,k) := \max\{j \leq k : 2^{-j} \geq x\}$ for $x < 1$ and $r(x,k) := 0$ for $x > 1$, $k \in \mathbb{N}$ and set $\tilde{b}_k := r(b_k, k)$.

Then we have $\lim_{k\to\infty} b_k = 0$ iff $\lim_{k\to\infty} \tilde{b}_k = \infty$ (note that this also covers sequences that are ultimately constantly zero). The machine $\mathcal{M}_1$ now computes approximations $\tilde{b}_{k,j}$ of $\tilde{b}_k$. Let $b_{k,j} := \max\{|a_n - a_m| \mid j > n > m \geq k\}$ and $\tilde{b}_{k,j} = r(b_{k,j}, k)$. Then $\tilde{b}_{k,j}$ is non-increasing with j and therefore for each k there is a j such that $\tilde{b}_{k,j} = \tilde{b}_k$, i.e. the sequence $(\tilde{b}_{k,j})_j$ becomes stationary.

By parallel computation in rounds of k, $\mathcal{M}_1$ computes the approximation $\tilde{b}_{k,j}$ of $\tilde{b}_k$ for larger and larger j, and for each k, this approximation becomes stationary for sufficiently large j. The machine $\mathcal{M}_1$ stores the information about the $\tilde{b}_{k,j}$ in a single real number by a unary encoding. For $n \in \mathbb{N}$ let $\mathrm{u}(n)$ be the unary encoding of n. Then, in each round $\mathcal{M}_1$ outputs the real number $\mathcal{M}_1^{(j)} = 0.0\mathrm{u}(\tilde{b}_{1,j})0\mathrm{u}(\tilde{b}_{2,j})0\mathrm{u}(\tilde{b}_{3,j})0\ldots0\mathrm{u}(\tilde{b}_{j,j})$. Since for each k the computation of $\tilde{b}_{1,j}$ will become stationary, this sequence of outputs converges to a real number $\mathcal{M}_1(\mathcal{M}) = 0.0\mathrm{u}(\tilde{b}_1)0\mathrm{u}(\tilde{b}_2)0\mathrm{u}(\tilde{b}_3)0\ldots$.

Now, it follows that the original sequence (a_n) converges iff $\mathcal{M}_1(\mathcal{M})$ is an irrational number. Because if the original sequence converges, then $\tilde{b}_k$ is unbounded and the number computed by $\mathcal{M}_1$ has no periodic dual expansion. If, on the other hand, the original sequence is not convergent, then $\tilde{b}_k$ will become stationary (since it is nondecreasing with k and bounded) and the number computed by $\mathcal{M}_1$ has a periodic dual expansion.

The second machine $\mathcal{M}_2$ just has to decide whether its input is rational or not, which an analytic machine simply does by enumerating all rational numbers. □

The undecidability of the convergence problem for analytic machines and the decidability of this problem by two composed analytic machines imply that those functions cannot be closed under composition (this follows in fact already from Chadzelek's result that three composed analytic machines decide convergence):

Corollary 2.9 *The set of analytically computable functions is not closed under composition.*

3 Representation Theorems

We now come to the first main part of this work: Are there generalizations of Blum, Shub and Smale's representation theorem for analytic machines?

We begin with the following observation: An analytic machine can have infinitely many branching operations on a computation path, and therefore it is possible that the domain of every computation path has empty open interior. Consider, e.g., the characteristic function of the rationals (Dirichlet's function) $\chi_{\mathbb{Q}}$. This function is easily seen to be computable[1] by an analytic machine over $\mathbb{R}$, but it is discontinuous everywhere. Since the notion of a power series only makes sense on sets with nonempty open interior, in such cases there cannot be a representation by power series. Therefore, we

[1] Indeed, a machine that computes $\chi_{\mathbb{Q}}$ enumerates the rationals and compares its input to each of the enumerated numbers. If the input is equal to the current number of the enumeration, it outputs 1 and halts. Otherwise, it outputs 0 and proceeds with the enumeration.

restrict our consideration to computation paths σ whose domain D_σ has nonempty open interior, or in the simplest case, functions which are computable with a single computation path.

Remark 3.1 For an infinite computation, a machine with a finite program obviously has to branch infinitely often. However, if a branch does not depend on the input, i.e. is only dependent on *inner variables* like counters, the domain of the function is not split by such a branch. Therefore, we address those 'inner' branching operations that cannot split the domain of the computed function as *branches that do not depend on the input*. Formally, this means that the branching condition, which can be represented by a polynomial in the input (cmp. proof idea of Theorem 2.6), actually does not depend on the input, i.e. is a constant polynomial.

3.1 Representation over $\mathbb{R}$

In this paragraph we will show that over the real numbers, the representation theorem for finite $\mathbb{R}$-machines does not generalize to analytic machines. More precisely, we will show that there are functions over $\mathbb{R}$ which are nowhere representable by power series and which are yet computable by an analytic machine with a single computation path.

Proposition 3.2 *Suppose $f : [0,1] \to \mathbb{R}$ is a continuous function with the property $f(\mathbb{Q} \cap [0,1]) \subseteq \mathbb{Q}$, i.e. f maps rationals on rationals. Then f is analytically $\mathbb{R}$-computable with the use of general constants such that all inputs $x \in [0,1]$ have the same computation path.*

Proof If f is a function with the required property, then by the Weierstrass approximation theorem, f can be uniformly approximated on $[0,1]$ by the *Bernstein polynomials*

$$B_{n,f}(x) := \sum_{m=0}^{n} f\left(\frac{m}{n}\right)\binom{n}{m} x^m (1-x)^{n-m}. \tag{1}$$

Given the values $f(\frac{m}{n})$ of f on the rationals, arbitrary approximations of f can be computed without branching operations *that depend on the input*. The sequence $(f(\frac{m}{n}))_{m \le n}$ can be coded in a single real constant c_f with the help of computable pairing functions. The extraction of $f(\frac{m}{n})$ from the coded constant can be done without branching operations depending on the input. Considering the explicit presentation of the Bernstein Polynomials (1), we see that $B_{n,f}(x)$ can also be computed without branching operations depending on the input x (which is the required property of those polynomials). Therefore it is possible to compute f analytically with *a single computation path* for all inputs $x \in [0,1]$. □

Remark 3.3

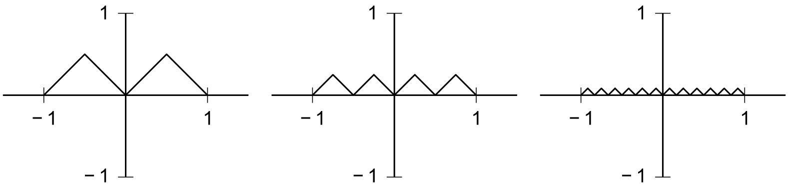
(a) This theorem nicely illustrates the power of general real constants for $\mathbb{R}$-machines.

(b) When considering the class of *strongly* analytic machines [4], i.e. machines that always give an output precision bound, it can be shown that this theorem also holds for functions computable by strongly analytic machines. Those machines are, however, not within the scope of this paper and therefore further details are omitted. The interested reader is pointed to [8, Proposition 5], where an extension of this theorem is proved for strongly analytic machines.

Example 3.4 We now give an example for a function with the properties of Proposition 3.2 which is nowhere representable by a power series. Moreover, we can show that this function can be computed with a single computation path even *without* the use of general real constants.

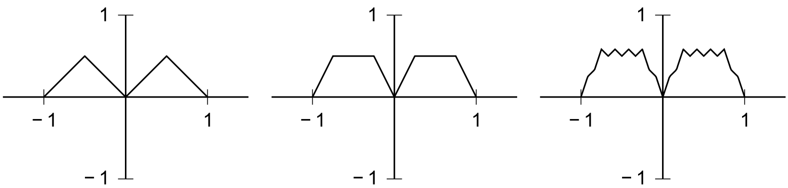
Let g be the periodic continuation of the function

$$x \mapsto \begin{cases} x: & 0 \le x \le \frac{1}{2} \\ 1-x: & \frac{1}{2} \le x \le 1 \end{cases}$$



and define the function G

$$G(x) := \sum_{k=1}^{\infty} \frac{1}{k!} g(k!x)$$



In the figure, the first three steps of the formation of G are shown, summands above and sums below.

It can be shown that G is a continuous but nowhere differentiable function. Hence G is nowhere representable by a power series.

The sum in the definition of G is *finite* for all rational inputs $\frac{p}{q}$. Therefore, the function G is *(finitely)* $\mathbb{R}$*-computable* on the rationals, and the encoding of the rational values in a real constant c_G is not necessary.

Combined with Proposition 3.2, we obtain that G is an example of a function that is computable by an analytic machine with a single computation path, and which is nowhere representable by a power series.

Remark 3.5 Note that Example 3.4 in conjunction with Proposition 3.2 would not work over $\mathbb{C}$: A sequence of complex polynomials which is uniformly convergent, converges necessarily to an analytic function. The Weierstrass approximation theorem can be used to uniformly approximate functions $\mathbb{C} \to \mathbb{C}$ if viewed as functions $\mathbb{R}^2 \to \mathbb{R}^2$, but in this case the approximating polynomials are not holomorphic polynomials (i.e. polynomials in one complex variable), but polynomials in the real variables given by real and imaginery parts of the argument. A $\mathbb{C}$-machine has, however, no access to real and imaginary parts of its input.

3.2 Representation over $\mathbb{C}$

Over the complex numbers, regularity of functions is often much easier obtained than over the real numbers. For example, it is well-known that over the reals, a function that is differentiable does not necessarily have a continuous (and, moreover, differentiable) derivative. In contrast, a function that is complex differentiable is already differentiable infinitely often and even analytic. Therefore, the question arises whether in the setting of analytic machines, there are also stronger regularity properties of functions computed by machines over $\mathbb{C}$. As before, we restrict our consideration to computation paths σ that have domains D_σ with nonempty open interior.

The following theorem shows that over the complex numbers, machines with such open domains of computation paths indeed admit a representation by power series on a subset of these domains.

Theorem 3.6 *Suppose $f : D \to \mathbb{C}$ is analytically computable by a $\mathbb{C}$-analytic machine $\mathcal{M}$. Suppose further that there is a computation path σ of $\mathcal{M}$, such that its domain D_σ has nonempty interior. Then there is an open and dense subset $D' \subset D_\sigma$ such that f is an analytic function on D'.*

Proof Consider the sequence of n-th approximations $f_n(z) := \mathcal{M}^{(n)}(z)$ of the machine for inputs $z \in D_\sigma$ for a computation path as in the theorem. By the representation theorem of Blum, Shub and Smale, each f_n is a rational function, and, therefore, a holomorphic function on the (by assumption nonempty) open interior of D_σ. The sequence (f_n) converges pointwise on D_σ to the function f. Osgood's theorem [14], a classical theorem from complex analysis, can be directly applied in this context. It implies that there is an open and dense subset $D' \subset D_\sigma$ such that the convergence of (f_n) is uniform on compact subsets of D'. From this it follows immediately that the limit f is an analytic function on D'.

The reader who is not familiar with Osgood's theorem can find modern proofs in [18] and [2]. □

The existence of a computation path that has a domain with nonempty open interior is, for example, fulfilled if each computation path of a machine $\mathcal{M}$ has only finitely many branching operations. Indeed, if this is the case, it follows that $\mathcal{M}$ has only countably many computation paths, and Baire's category theorem implies that at least one of the domains of these paths must have nonempty open interior. Therefore we have the following representation theorem:

Corollary 3.7 *Let $f : D \to \mathbb{C}$ be computable by a $\mathbb{C}$-analytic machine such that each computation path of $\mathcal{M}$ has only finitely many branching operations depending on the input. Then there is a nonempty open set on which f is representable by a power series.*

4 Computability of Analytic Functions

In the previous section we have discussed to which extent functions computable by analytic machines are representable by power series, i.e. are also analytic in the classical mathematical sense of the word. In this second part of the paper we now study computability properties of classical analytic functions. By an *analytic function* on a region of the complex plane $\mathbb{C}$ we mean a function that has a power series expansion in each point of the region, or equivalently, which is complex-differentiable in every point of D. Those functions are also called holomorphic functions, but in our context we prefer the term analytic because it stresses the power series aspect of the function.

We will define two classes of computable analytic functions, first those functions which are analytic and also analytically computable (i.e. computable by an analytic machine) and then those functions which have a power series expansion that is computable by an analytic machine, the functions which we call coefficient computable. We will then show properties of the functions in the respective classes and examine the relationship between those two classes. In particular, we will show that every coefficient computable analytic function is also computable, and that the class of coefficient computable analytic functions is closed under composition, local inversion and analytic continuation.

A part of this section has already been presented in [6]. The main new result here is Theorem 4.10.

Notation In this section, if not mentioned otherwise, D denotes a *region* in $\mathbb{C}$, i.e. a nonempty open connected subset of $\mathbb{C}$. For a complex number z_0 and $r > 0$, $U_r(z_0)$ is the open disk with center at z_0 and radius r and $\overline{U_r}(z_0)$ its closure.

4.1 Computable Analytic Functions

First, we show that the case of finitely computable functions or, equivalently, BSS-computable functions is not very interesting in the context of analytic functions:

Lemma 4.1 *Let D be a region in $\mathbb{C}$ and $f : D \to \mathbb{C}$ be analytic and (finitely) $\mathbb{C}$-computable. Then f is a rational function on D.*

Proof By Theorem 2.6, D is the countable union of (pseudo) semi-algebraic sets on which f is a rational function. By Baire's theorem it follows that at least one of those sets has nonempty interior, and therefore f equals a rational function on a nonempty open subset of D, and hence by the identity theorem on all of D. □

In light of Lemma 4.1, we will focus our attention on analytic functions which are not necessarily finitely computable. Therefore, in the following, we will freely omit the term ‘analytic’ when we talk about computability, i.e. instead of ‘analytically computable analytic functions’ we will just say ‘computable analytic functions’.

Definition 4.2 Let $D \subset \mathbb{C}$ be a region in the complex plane, and $f : D \to \mathbb{C}$ be a function. We call f a

- *computable analytic function* if f is holomorphic and analytically computable on D. We call f
- *coefficient computable in* $z_0 \in D$ if f is holomorphic on D and the power series expansion of f in z_0 is an analytically computable sequence, and we call f
- *uniformly coefficient computable on* D if the mapping $\mathbb{N} \times D \to \mathbb{C}$, $(k, z_0) \mapsto a_k(z_0)$ is analytically computable, where $a_k(z_0)$ is the k-th coefficient of the power series of f in z_0 and
- *(locally) coefficient computable on* D if f is holomorphic on D and each point $z_0 \in D$ has a neighborhood $U \ni z_0$, such that f is uniformly coefficient computable on U.

As indicated by the parentheses, we will usually leave out the term ‘locally’ for locally coefficient computable functions on a domain D.

Remark 4.3 In this section, we only consider machines without the use of general real (or complex) constants. In a real constant, countably many rational numbers can be encoded and therefore every analytic function is coefficient computable: the coefficients of the power series can simply be encoded in real constants (non-rational coefficients are encoded by their rational approximations; non-real complex numbers are stored as real and imaginery parts). Since every coefficient computable function is also analytically computable (this will be shown in Theorem 4.4), with the use of general real constants, every analytic function is computable by an analytic machine.

Now, we show that coefficient computability is the stronger notion, i.e. that each coefficient computable function is also computable.

Theorem 4.4 *Suppose* $f : D \to \mathbb{C}$ *is an analytic function on D with power series expansion* $f(z) = \sum_{k=0}^{\infty} a_k (z - z_0)^k$ *at a given* $z_0 \in D$*, such that the sequence* $(a_k)_{k\in\mathbb{N}}$ *is analytically computable*, *i.e.* f *is coefficient computable in* z_0. *Let be* $R > 0$ *such that* $\overline{U}_R(z_0) \subseteq D$. *Then* f *is analytically computable in the open disk* $U_R(z_0)$.

Proof Let $\mathcal{K}$ be the machine that (analytically) computes the coefficients of the power series of f in z_0. We give the description of a machine $\mathcal{M}$ which analytically computes f in $U_R(z_0)$. We denote the n-th approximation of $\mathcal{K}$’s computation of the coefficient a_k with $a_k^{(n)}$. Let $z \in U_R(z_0)$ be the input for $\mathcal{M}$. The basic idea would now be to approximate the value $f(z)$ by a diagonal approximation $\sum_{k=0}^{n} a_k^{(n)} (z - z_0)^k$. The problem is that there is no information about the speed of convergence of $a_k^{(n)}$ with n, which could also be different for each k. The trick is to use upper bounds on the coefficients given by basic theorems of function theory. We recall that by Cauchy’s inequalities (cf. e.g. [17]) for f there is a constant $M > 0$ such that $|a_k| \le \frac{M}{R^k}$ for all $k \in \mathbb{N}$. The machine $\mathcal{M}$ now has information about M and R stored as constants (w.l.o.g rational bounds can be chosen), and if a current approximation $a_k^{(n)}$ exceeds the bound $\frac{M}{R^k}$, it is replaced by this bound. Summarizing, $\mathcal{M}$ computes (as n-th approximation)

$$\mathcal{M}^{(n)}(z) = \sum_{k=0}^{n} b_k^{(n)}(z - z_0)^k \quad \text{with } b_k^{(n)} = \begin{cases} a_k^{(n)} & |a_k^{(n)}| \leq \frac{M}{R^k} \\ \frac{M}{R^k} & \text{else} \end{cases}$$

We assert that the sequence of outputs $\mathcal{M}^{(n)}(z)$ converges to $f(z)$. In order to see this, the difference $|f(z) - \mathcal{M}^{(n)}(z)| = |f(z) - \sum_{k=0}^{n} b_k^{(n)}(z - z_0)^k|$ is split into three parts, each of which becomes arbitrarily small for large n. The first part is $\sum_{k=0}^{n_0} |(a_k - b_k^{(n)})(z - z_0)^k|$ for a fixed n_0, and the second and third parts are $\sum_{k=n_0}^{n} |(a_k - b_k^{(n)})(z - z_0)^k|$ and $\sum_{k=n}^{\infty} |a_k(z - z_0)^k|$, respectively. By choosing n_0 large enough, the second and third parts can be made arbitrarily small. For n_0 then fixed, the first part can also be made arbitrarily small, since for large enough n the fixed number n_0 of the analytic computations of $a_0^{(n)}, \ldots, a_{n_0}^{(n)}$ can be made arbitrarily precise, and hence this can be done with their finite sum as well. □

Remark 4.5 Note that in Theorem 4.4 the prerequisite that $\overline{U}_R(z_0) \subseteq D$ for given fixed $R > 0$ is essential for the proof. Therefore the theorem (and the following ones which have the same prerequisite) is non-uniform in the sense that computability is given only on a fixed disk and not in the whole domain.

A refinement of the proof of Theorem 4.4 will lead us now to an even stronger result, namely that not only a function is computable if it is coefficient computable, but also all its derivatives. For the following, recall the notation $[k]_d := \frac{k!}{(k-d)!} = k \cdot (k-1) \cdots (k-d+1)$. We observe that if the sequence (a_k) is analytically computable, then so are all sequences $(a_{(d,k)})_k = ([k+d]_d a_{k+d})_k$. Furthermore, they are *uniformly* computable, i.e. the mapping $(d, k) \mapsto a_{(d,k)}$ is analytically computable. Now, we can show that

Theorem 4.6 *Suppose $f : D \to \mathbb{C}$ is an analytic function on D which is coefficient computable in a given $z_0 \in D$. Let further be $R > 0$ such that $\overline{U}_R(z_0) \subseteq D$.*

Then each derivative $f^{(d)}$ is analytically computable in the open disk $U_R(z_0)$, and furthermore, all derivatives are uniformly *computable, i.e. the mapping $(d, z) \mapsto f^{(d)}(z)$ is analytically computable on $\mathbb{N} \times U_R(z_0)$.*

Proof The proof is similar to the proof of Theorem 4.4. Using Cauchy's inequalities for the derivatives we get the estimate $|a_{d,k}| = (k+d)_d a_{k+d} \leq \frac{M}{R^{k+d}}(k+d)_d$. The approximation of $f(z)$ given by the machine is again split into three parts, and with a little more technical effort, each part can be shown to become arbitrarily small. □

Corollary 4.7 *Suppose $f : D \to \mathbb{C}$ is an analytic function on D which is coefficient computable in $z_0 \in D$. Let further be $R > 0$ such that $\overline{U}_R(z_0) \subseteq D$.*

Then f is uniformly coefficient computable on $U_R(z_0)$.

Proof By Theorem 4.6, the derivatives of f are uniformly computable on $U_R(z_0)$. Since the coefficients of the power series expansion of a function can be expressed by its derivatives, the power series expansion can be uniformly computed on $U_R(z_0)$. □

This shows that coefficient computability in a point is the basic property for the computability of analytic functions in our sense, since it implies the computability and even uniform coefficient computability in a disk around this point. We can actually go a little further:

4.1.1 Analytic Continuation

A very important notion in the realm of analytic functions is the notion of *analytic continuation*. We briefly recall the basic concept. Let $f : D \to \mathbb{C}$ be an analytic function, $z_0 \in D$ and $f(z) = \sum_{k=0}^{\infty} a_k(z - z_0)^k$ the power series expansion of f with convergence radius $r > 0$. Let $U_r(z_0)$ be the disk with center z_0 and radius r. Then, possibly the range of convergence $U := U_r(z_0)$ of the power series exceeds the boundary of D, the given domain of the function.[2] If we assume further that $D \cap U$ is connected, a well-defined analytic function g is defined on $D \cup U$ by $g(z) = f(z)$ if $z \in D$ and $\sum_{k=0}^{\infty} a_k(z - z_0)^k$ if $z \in U$. It is well-defined because $D \cap U$ is connected, since then f and the function defined by the power series have to be equal on all of $D \cap U$ because of the identity theorem (note that connectedness is sufficient for the application of this theorem; simple connectedness is not necessary). The function g is called *immediate analytic continuation* of f. This process can be iterated, and an extension of f on a much larger region G may be defined. If the requirement of the connectedness of $D \cap U$ is omitted, the resulting 'function' can have multiple values. Consequently developed, this leads to the theory of *Riemann surfaces*, but this is not the subject of this article. We confine ourselves to regions that are subsets of the complex plane.

A very interesting result about coefficient computable functions is that they are closed under analytic continuation:

Theorem 4.8 *Suppose that $f : D \to \mathbb{C}$ is coefficient computable and that f has an analytic continuation $g : G \to \mathbb{C}$ on a region $G \supset D$. Then g is coefficient computable on G.*

In particular, if a function $f : D \to \mathbb{C}$ is holomorphic on D and coefficient computable at a given $z_0 \in D$, it is coefficient computable on D.

Proof This is a consequence of Corollary 4.7. In order to show that f is coefficient computable in G, we have to show that it is uniformly coefficient computable in a neighborhood of each point $w \in G$. W.l.o.g, we can choose $w \in G$ with rational coordinates (for $w \in G$, choose a point w' with rational coordinates such that the open

[2]For example, consider the function f defined by the power series $f(z) := \sum_{k=0}^{\infty} (\frac{1}{2})^k z^k$, $|z| < 2$. If we now consider $z_0 = -1$, the power series expansion of f at z_0 is given by $\sum_{k=0}^{\infty} \frac{2}{3^{k+1}}(z + 1)^k$, which has a convergence radius of 3. Therefore, the radius of convergence of the expansion at -1 exceeds the boundary of the original domain $\{z : |z| < 2\}$.

disk of convergence of the power series of g at w' includes w), fix $z_0 \in D$ at which f is coefficient computable (e.g., a point with rational coordinates). Inductively, we obtain a sequence of points $z_1, z_2, \dots, z_n = w$ and functions $f = f_0, f_1, f_2, \dots, f_n$ such that f_{i+1} is an analytic continuation of f_i, z_{i+1} is in the disk of convergence of f_i at z_i and f_i is coefficient computable at z_i. In particular, f_n and thus g is coefficient computable at (and in a neighborhood of) $w = z_n$. □

This result is interesting because it relates an important notion of complex function theory to computability: In function theory, there is the notion of a *function germ*, which conveys that a holomorphic function is completely determined by the information of its power series expansion in a single point. Theorem 4.8 shows that a similar notion holds for coefficient computability: Coefficient computability of a function in a single point implies its computability in on its whole domain.

Remark 4.9 Note that in Theorem 4.8 coefficient computability in the whole domain D does not imply that there is a single machine that computes the coefficients of the function on the whole domain. In this sense, the result is also not uniform.

4.1.2 Computable Implies Coefficient Computable?

The question whether the converse of Theorem 4.4 is true as well, i.e. if a computable analytic function is also coefficient computable, is still open. However, the preceding results in conjunction with an application of a proof technique of Sect. 3 allows us to show that the converse holds at least in a restricted case:

Theorem 4.10 *Suppose $f : D \to \mathbb{C}$ is a computable analytic function on D such that a machine exists that has only finitely many branching operations depending on the input on each computation path σ. Then f is coefficient computable on D.*

Proof Let $\mathcal{M}$ be an analytic machine computing f. As shown in the proof of Corollary 3.7, there must exist a computation path σ such that the domain D_σ has nonempty open interior. Considering the n-th approximations f_n of f by M along σ, as in Theorem 3.6 with the help of Osgood's theorem we can infer that there is an open and dense subset $D' \subset D_\sigma$ such that the convergence is locally uniform on D'. We now fix a point $z_0 \in D'$ with rational real and imaginary parts. We show that f is coefficient computable in z_0 and thus by Theorem 4.8 coefficient computable on D. The machine $\mathcal{C}$ that computes the coefficients of the power series expansion of f at z_0 simulates the computation of the machine $\mathcal{M}$ on the input z_0 (which is given as a rational constant). It does, however, not compute the function value of the n-th approximation f_n, but the coefficients of the rational function f_n. This can be done by simulating the machine symbolically. Since the k-th derivative $f_n^{(k)}$ of the rational function f_n can be computed symbolically, the machine $\mathcal{C}$ can compute the value $f_n^{(k)}(z_0)$ for each k. By Weierstrass' convergence theorem, the local uniform convergence of f_n on D' implies the local uniform convergence of all derivatives, and therefore $f_n^{(k)}(z_0)$ converges to $f_n^{(k)}(z_0)$. With input k, the machine $\mathcal{C}$ outputs

$\frac{1}{k!} f_n^{(k)}(z_0)$ for increasing n, which converges to the k-th coefficient of the power series expansion of f in z_0. Thus, f is coefficient computable in z_0, and therefore by Theorem 4.8 (locally) coefficient computable on D. □

Question 4.11 *Does the converse of Theorem 4.4 hold in the general case, i.e. is each analytic function that is computable by an analytic machine also coefficient computable?*

4.2 Closure Properties of Coefficient Computable Functions

We now turn our attention to closure properties of the class of coefficient computable functions. After we have already shown that this class is closed under analytic continuation, we show now that it is also closed under the basic analytic operations composition and local inversion.

4.2.1 Composition

Since the class of generally analytically computable functions is not closed under composition (cf. Corollary 2.9), one could expect that this is also the case for the class of coefficient computable functions, which are defined by means of analytic machines. But with the theorems of the last section we can show that this class actually is closed under composition. The standard approach here would be to use a recursive description of the coefficients of the composition to compute these; since the precision of the approximations are, however, unknown, this approach is not successful. Instead, we directly express the coefficients in terms of the derivatives and apply Theorem 4.6.

Theorem 4.12 *Suppose $g : D \to \mathbb{C}$ and $f : g(D) \to \mathbb{C}$ are coefficient computable functions. Then the composition $f \circ g : D \to \mathbb{C}$ is also coefficient computable.*

Proof We fix a $z_0 \in D$ with rational coordinates and show that $g \circ f$ is coefficient computable in z_0. With Theorem 4.8, it then follows that $g \circ f$ is coefficient computable on D.

The general form of the n-th derivative of the composition $f \circ g$ at z_0 is given by *Faa di Bruno's Formula*:

$$(f \circ g)^{(n)}(z_0) = \sum_{(k_1,\dots,k_n)\in P_n} \frac{n!}{k_1! \cdot \dots \cdot k_n!} (f^{(k_1+\dots+k_n)})(z_1) \prod_{m=1}^{n} \left(\frac{g^{(m)}(z_0)}{m!} \right)^{k_m}$$

Here, the set $P_n = \{(k_1, \dots, k_n) \mid k_1 + 2k_2 + \dots + nk_n = n\}$ denotes the set of partitions of n.

In order to compute the coefficients of $f \circ g$ at z_0 observe that the coefficients of each derivative of f at $z_1 := g(z_0)$ and g at z_0 are uniformly computable, and that the n-th derivative of $f \circ g$ is a finite polynomial in those. □

4.2.2 Local Inversion

Similarly to case of composition, we can apply Theorem 4.6 to show that the local inversion of a coefficient computable function is again coefficient computable:

Theorem 4.13 *Suppose $f : D \to \mathbb{C}$ is coefficient computable and $f'(z_0) \neq 0$. Then the local inverse f^{-1} of f is coefficient computable in a neighborhood of $f(z_0)$.*

Proof W.l.o.g. we can assume z_0 has rational coordinates. Similarly to Theorem 4.12 one can show by induction, that the coefficients of f^{-1} at $z_1 := f(z_0)$ can be expressed as rational functions in the derivatives of f at z_0 and therefore in the coefficients of the power series of f in z_0. □

5 Conclusion and Further Questions

In this paper, we have studied to which extent the representation theorem for BSS machines can be generalized to analytic machines, i.e. whether functions computable by analytic machines are representable by power series. We have shown that over the real numbers, this is generally not possible, but over the complex numbers functions computable by analytic machines with 'not too many branching operations' are always representable by power series on some part of their domain.

Then, we have used the model of analytic machines to define two notions of computability for complex analytic functions. We have shown coefficient computability to be the stronger of the two notions. However, under the prerequisite of the existence of a computation path with a nonempty interior domain, computable analytic functions are also coefficient computable. We have further shown that, in contrast to the general analytically computable functions, the class of coefficient computable functions is closed under composition, and also under local inversion and analytic continuation.

1. Is the representation theorem for analytic $\mathbb{C}$-machines tight, i.e., is it necessary to restrict to the dense open subset D' of D in Theorem 3.6? With the help of Runge's theorem one can construct pointwise convergent sequences of holomorphic functions which indeed only converge uniformly on the dense open subset of the domain. It is not clear, however, whether these constructions can be emulated by an analytic machine.
2. The theorems in Sect. 4 concerning computability of analytic functions have covered the functions computable by general analytic machines. It would be interesting to see whether these results carry over to more restricted machine models, like the strongly analytic machines or quasi-strongly analytic machines.
3. Those theorems are of a non-uniform nature (cf. Remark 4.5). To which extent are more uniform results obtainable?
4. Moreover, those results have in a sense not been constructive, since in order to show computability, constants containing information about the regarded functions have been assumed in the constructed machines. Open questions remain whether any of these results can, at least in part, be gained constructively.

5. It could also be interesting to compare these results to the corresponding ones for Type 2 Turing machines from the viewpoint of relativization [20].

Acknowledgements We would like to thank the referees for their helpful suggestions, which have greatly improved the quality of the paper.

References

1. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer, New York (1998)
2. Beardon, A.F., Minda, D.: On the pointwise limit of complex analytic functions. Am. Math. Mon. **110**(4), 289–297 (2003)
3. Blum, L., Shub, M., Smale, S.: On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. Bull. Am. Math. Soc. **21**, 1–46 (1989)
4. Chadzelek, T., Hotz, G.: Analytic machines. Theor. Comput. Sci. **219**, 151–167 (1999)
5. Chadzelek, T.: Analytische Maschinen. Dissertation, Saarland University, 1998
6. Gärtner, T., Hotz, G.: Computability of analytic functions with analytic machines. In: Proc. CiE 2009. LNCS, vol. 5635, pp. 250–259. Springer, Berlin (2009)
7. Grzegorczyk, A.: On the definition of computable real continuous functions. Fundam. Math. **44**, 61–71 (1957)
8. Gärtner, T., Ziegler, M.: Real analytic machines and degrees. In: Proc. CCA 2010. Electronic Proceedings in Theoretical Computer Science, vol. 24 (2010)
9. Hotz, G., Vierke, G., Schieffer, B.: Analytic machines. Electron. Colloq. Comput. Complex. **2**, 25 (1995)
10. Ko, K.-I.: Complexity Theory of Real Functions. Birkhäuser, Boston (1991)
11. Müller, N.T.: Uniform computational complexity of Taylor series. In: Proc. 14th ICALP. LNCS, vol. 267, pp. 435–444 (1987)
12. Müller, N.T.: Constructive aspects of analytic functions. Informatikberichte **190**, 105–114 (1995)
13. Meer, K., Ziegler, M.: Uncomputability below the real halting problem. In: Proc. CiE 2006. LNCS, vol. 3988, pp. 368–377. Springer, Berlin (2006)
14. Osgood, W.F.: Note on the functions defined by infinite series whose terms are analytic functions of a complex variable. Ann. Math. **2**(3), 25–34 (1902)
15. Rettinger, R.: Computable Riemann surfaces. In: Proc. CiE 2007. LNCS, vol. 4497, pp. 638–647. Springer, Berlin (2007)
16. Rettinger, R.: Lower bounds on the continuation of holomorphic functions. Electron. Notes Theor. Comput. Sci. **221**, 207–217 (2008)
17. Rudin, W.: Real and Complex Analysis. McGraw-Hill, New York (1987)
18. Schiff, J.L.: Normal Families. Springer, Berlin (1993)
19. Weihrauch, K.: Computable Analysis. Springer, Berlin (2000)
20. Ziegler, M.: Computability and continuity on the real arithmetic hierarchy and the power of type-2 nondeterminism. In: Proc. CiE 2005. LNCS, vol. 3526, pp. 562–571. Springer, Berlin (2005)
21. Ziegler, M.: Real computability and hypercomputation. Technical report C-07013, KIAS, July 2007
22. Ziegler, M.: Real hypercomputation and continuity. Theory Comput. Syst. **41**, 177–206 (2007)