Constructing Non-Computable Julia Sets

Mark Braverman[*]
Department of Computer Science
University of Toronto
mbraverm@cs.toronto.edu

Michael Yampolsky[†]
Department of Mathematics
University of Toronto
yampol@math.toronto.edu

ABSTRACT

While most polynomial Julia sets are computable, it has been recently shown [12] that there exist non-computable Julia sets. The proof was non-constructive, and indeed there were doubts as to whether specific examples of parameters with non-computable Julia sets could be constructed. It was also unknown whether the non-computability proof can be extended to the *filled* Julia sets. In this paper we give an answer to both of these questions, which were the main open problems concerning the computability of polynomial Julia sets.

We show how to construct a specific polynomial with a non-computable Julia set. In fact, in the case of Julia sets of quadratic polynomials we give a precise characterization of Julia sets with computable parameters. Moreover, assuming a widely believed conjecture in Complex Dynamics, we give a poly-time algorithm for computing a number c such that the Julia set J_{z^2+cz} is non-computable.

In contrast with these results, we show that the filled Julia set of a polynomial is always computable.

Categories and Subject Descriptors

F.2.1 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*Numerical Algorithms and Problems*; G.1.m [**Mathematics of Computing**]: Numerical Analysis—*Miscellaneous*

General Terms

Theory, Algorithms

Keywords

Julia sets, computability, dynamical systems, real computation

[*]Partially supported by an NSERC postgraduate scholarship

[†]Partially supported by an NSERC Discovery Grant

1. INTRODUCTION

Studying dynamical systems is key to understanding a wide range of phenomena ranging from planets' movement, to climate patterns to market dynamics. Various numerical tools have been developed to answer questions about specific dynamical systems, such as predicting the weather or planning the trajectory of a satellite. However, the theory of computation behind these problems is generally not developed. While we have vast knowledge about the computability and complexity of discrete problems, there is little known about the computability of even the most natural problems arising from dynamical systems. In the present paper we study the computability of Julia sets, which derive from iteration of polynomial mappings of $\mathbb{C}$.

Despite being the simplest non-linear examples of complex-analytic dynamical systems, quadratic polynomials $p(z) = z^2 + \alpha z$ have emerged as one of the central subjects in the theory of low-dimensional dynamical systems [17]. Their properties are in many ways archetypical, and the deceptively simple looking formula hides an enormous complexity of behavior. This complexity is found in a subset of the plane, called the Julia set of a polynomial. It manifests itself in the beautiful computer-generated images of such sets, and indeed, numerical experiments have been central to the development of the field [15, 24]. Julia sets are among the most drawn mathematical objects, as a brief Web search readily attests.

Prior to our work, the distinction between computationally "easy" and "hard" examples of Julia sets was made empirically, based on the community's ability to produce algorithms for different classes of such sets. This classification was not always correct. We were able to apply Theoretical Computer Science to give an answer to most of the problems surrounding the computability of Julia sets, and some of the questions about their computational complexity. The answers we have obtained are quite unexpected, and have revealed a deep connection between the computational hardness and the analytic properties of the dynamics.

A Turing Machine has a dynamical definition, and can even be embedded into a rather simple low-dimensional dynamics [1, 19, 20, 25]. In such examples, the unsolvability of the Halting problem results in the impossibility of certain predictions about the behavior of *individual* infinite trajectories. This is, however, a somewhat artificial difficulty: in practice it is often impossible to compute an infinite trajectory not because of some deep theoretical reasons, but simply due to the finite precision of computations. On the other hand, the structural properties of *all* trajectories, such as the shape of the *attractors* or *repellers* (and Julia sets are examples of the latter) is frequently sufficiently robust to allow numerical study. In particular, when the coefficients of the dynamical system can be generated algorithmically, one expects to be able to study its structure experimentally. As we will see, even in the case of quadratic polynomials, this need not be the case.

For a polynomial mapping $p : \mathbb{C} \to \mathbb{C}$ a *trajectory* or an *orbit* of a point $z \in \mathbb{C}$ is the infinite sequence of iterates $z_0 = z$, $z_1 = p(z)$, $z_2 = p(z_1) = p(p(z)), \ldots$ To define the Julia set J_p, note first that once the absolute value $|z|$ is very big, the absolute value $|p(z)|$ will be even bigger, and the orbit will escape to ∞. The set of points whose orbits do not escape to infinity is thus a planar compact, which is called the *filled Julia set* K_p. The *Julia set* is its boundary: $J_p = \partial K_p$. It can be shown to consist of all points z in $\mathbb{C}$ for which the behavior of the orbit is unstable: a small perturbation in z can lead to a significant change in the orbit. In Fig. 1 we see some examples of Julia sets and filled Julia sets for maps of the form $p(z) = z^2 + \alpha z$. The self-similarities which give each of the pictures its fractal shape are due to the invariance $p^{-1}(J_p) = J_p$.

A set S in $\mathbb{C}$ is said to be *computable* if one can draw it on a computer screen. The basic decision procedure involved is deciding whether to put a "pixel" on the screen or not. A pixel $P = B(c, r)$ is a ball with a rational center $c = q_1 + q_2 i \in \mathbb{C}$ and a rational radius r. The computation must output 1 if $B(c, r)$ intersects S. In this case the pixel is colored black. The computation must output 0 if the pixel is at least r-far from S, that is $B(c, 2 \cdot r) \cap S = \emptyset$, in which case the pixel is colored white. If none of these possibilities holds, either answer is acceptable.

The definition above is a standard definition for set computability, and turns out to be quite robust. It was first introduced in [21]. It is strongly related to other objects in Computable Analysis – an area that was originated by Banach and Mazur [2]. For example, under very broad conditions a continuous real function is computable if and only if its graph is computable (as a set) [6]. More on computability of Euclidean sets can be found in [7, 9, 29]. Set computability is discussed in greater detail in Section 2.

We remark that Julia sets have appeared in a context somewhat similar to ours in the book [5] where the authors studied their decidability in the *Blum-Shub-Smale (BSS)* model of real computation. However, due to the algebraic nature of the BSS model, not only all non-trivial Julia sets, but even the simplest sets with fractal structure (such as the middle-thirds Cantor set) are BSS-undecidable. As we will see, the Computable Analysis approach leads to a meaningful classification of Julia sets based on computability, which also carries a very concrete practical meaning.

From now on, we shall focus our attention on the one-complex-parameter family of polynomials of the form $p(z) = z^2 + \alpha z$. To simplify notation, we will write J_α and K_α for the corresponding Julia sets and filled Julia sets. In this context, computing J_α by a machine M means generating pictures of J_α with an arbitrarily high resolution given an access to arbitrary precision approximations of the parameter α. In other words, $M = M^\phi$ deciding whether to color a pixel, has an access to an oracle ϕ such that for any m, $\phi(m)$ is a rational point in $\mathbb{C}$ with $|\phi(m) - \alpha| < 2^{-m}$.

We are mostly concerned with negative results in this paper, and hence allow the machine M^ϕ to be *non-uniform*. The machine may be designed with the specific parameter α in mind. This definition makes the negative results the strongest possible in this context: even with α in mind, one cannot design a program that draws J_α. For positive results, non-uniform information is sometimes necessary. The non-uniform information is generally a finite amount of information that cannot be constructively extracted from the parameter α. We will see that in many cases no non-uniform information is needed at all.

The simplest class of Julia sets, both dynamically and computationally, is the class of *hyperbolic* Julia sets. These sets have been shown to be computable in [31]. A poly-time algorithm for computing some hyperbolic Julia sets was given in [27]. It has been later generalized in [8, 26] to all hyperbolic Julia sets. A related class are the parabolic Julia sets, which are also poly-time computable [10].

Another class of computable Julia sets are the filled Julia sets with empty interior, that is, the case when $J_\alpha = K_\alpha$ [3]. This is consistent with a more general fact we will prove in this paper that the filled Julia set K_α is always computable. Unlike the hyperbolic case, here little is known about whether the algorithms can be made efficient. In fact, there is an entire subfamily of these sets – called the Cremer Julia sets – for which no informative pictures have been produced to date.

The above list covers all cases but one – the most interesting from the computational point of view – the instance when J_α has a *Siegel disc*. The simplest case when it may occur is if $\alpha = e^{2\pi i \theta}$ with an irrational θ. Locally, near 0, the map $p_\alpha : z \mapsto z^2 + \alpha z$ behaves like a multiplication by α, which corresponds to a rotation by the angle θ. If there is an open region $S \subset K_\alpha$ around 0, then S is called a Siegel disc. In this case, the map p_α on S becomes a true rotation by θ after an analytic change of coordinates. An example of a computable Julia set with a Siegel disc can be seen in Fig. 1(b), which corresponds to $\theta = (\sqrt{5}+1)/2$ – the golden ratio.

Negative results

The case when J_α has a Siegel disc is the only case when it may not be computable. In [12] we have demonstrated that there are non-computable Julia sets with a Siegel disc. The proof used a diagonalization argument by constructing a parameter α that "fools" the countable set of machines $M_1^\phi, M_2^\phi, \ldots$ that may potentially compute J_α. In the proof a sequence $\alpha_1, \alpha_2, \ldots$ was constructed so that $\alpha_n \to \alpha$ and so that for each n, and for each $m \geq n$, α_m "fools" M_n. We were then able to deduce that J_α is uncomputable, by a suitable limiting argument.

The proof was non-constructive, as it used queries to the Halting Problem to obtain α_n from α_{n-1} while making sure that α_n (and all the following α's) "fools" M_n. Thus, while it has shown that it is not always possible to compute J_α given α, it left the possibility open that no such α is computable. This would mean that there is no "real" possibility of running into or producing an α for which J_α cannot be computed. In the present work we construct such an α. Moreover, we show how to construct an α such that computing J_α is *computationally equivalent* to solving the Halting Problem. Recall that a real number α is computable if there is a TM $M(n)$, which on an input n outputs a rational q_n with $|\alpha - q_n| < 2^{-n}$ [28]. We prove the following:

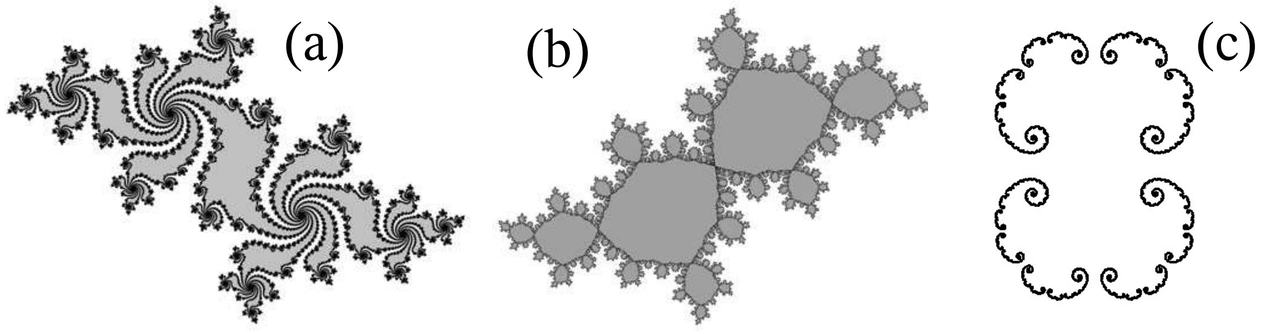

Figure 1: Examples of quadratic Julia sets J_p (black), and filled Julia sets K_p (gray); orbits that originate at white points escape to ∞; note that on picture (c) $K_p = J_p$, since K_p has empty interior

Theorem 1. *There exists a computable α such that given an oracle for J_α it is possible to solve the Halting Problem. In particular, J_α is non-computable.*

The simplest case when a Siegel disc occurs is when $|\alpha| = 1$, i.e. α has the form $\alpha = e^{2\pi i\theta}$. In this case we are actually able to characterize the non-computability of the possible Julia sets J_α for *computable* α's.

Theorem 2. *(1) Let $R(x, y)$ be any computable predicate on $\mathbb{N}^2$, and let $P(x) \equiv \exists y R(x, y)$. Then there is a computable $\alpha = e^{2\pi i\theta}$ such that J_α is computable relative to $P(x)$ and $P(x)$ is computable relative to J_α.*
(2) Let $\alpha = e^{2\pi i\theta}$ be computable. Then there is a computable predicate $R_\alpha(x, y)$ on $\mathbb{N}^2$ such that the predicate $P_\alpha(x) \equiv \exists y R_\alpha(x, y)$ is computable relative to J_α and vice versa: J_α is computable relative to $P_\alpha(x)$.

Here by "J_α is computable relative to $P(x)$" we mean that there is a TM which given an oracle access to the predicate $P(x)$ computes the Julia set J_α with any prescribed precision. The statement "$P(x)$ is computable relative to J_α" is interpreted in a similar fashion. Note that Theorem 1 is a special case of Theorem 2. It follows from part (1) of the theorem by taking $P(x)$ to be the Halting Problem predicate. Then there is a computable α such that $P(x)$ is computable relative to J_α, which means that given an oracle for J_α one can compute $P(x)$ and solve the Halting Problem.

The proof of Theorem 2 uses cutting-edge tools from Complex Dynamics, in particular [13], and is somewhat technical. In this presentation we isolate the underlying dynamics to make the exposition accessible.

We can further strengthen Theorem 2 by assuming a widely believed conjecture in Complex Dynamics, we call *the computability of υ conjecture*. It is a constructive version of the main result in [13]. It is significantly weaker than another conjecture in Complex Dynamics called the Marmi-Moussa-Yoccoz conjecture [22]. We prove:

Theorem 3. *Assuming the* computability of υ conjecture, *let $R(x, y)$ be any computable predicate on $\mathbb{N}^2$, and let $P(x) \equiv \exists y R(x, y)$. Then there is a* poly-time *computable $\alpha = e^{2\pi i\theta}$ such that J_α is computable relative to $P(x)$ and $P(x)$ is computable relative to J_α.*

This implies that there are *poly-time* computable α's with non-computable Julia sets. In other words, an n-bit approximation of α can be obtained in $poly(n)$ time, but J_α is non-computable.

Positive results

Our positive results deal with computability of *filled* Julia sets K_α. It is natural to ask whether the same techniques used to construct non-computable Julia sets can be used to construct non-computable filled Julia sets. In fact, this question was posed to us by John Milnor. Somewhat surprisingly, the answer to this question is negative: all filled Julia sets are computable.

Theorem 4. *For any α, K_α is computable. Moreover, in all non-parabolic cases, K_α can be computed by one of the two machines: M_c^ϕ (when it is connected), or M_d^ϕ (when it is disconnected).*

The result actually extends beyond the quadratic case, to all polynomial Julia sets. In the proof we give an algorithm for computing K_α. The key new element here is that we are able to deal with the case when K_α contains a Siegel disc – and so J_α may be non-computable.

2. COMPUTABILITY OF REAL FUNCTIONS AND SUBSETS OF $\mathbb{R}^2$

In this section we outline the basic definitions of computability of real functions and subsets of $\mathbb{R}^2$. These definitions are quite natural, and are standard within the context of Computable Analysis. They go back to the work of Banach and Mazur [2, 23]. A more contemporary exposition can be found in the books [18, 29]. The definitions we will be using here are summarized in [9].

First, consider a function $f : \mathbb{R} \to \mathbb{R}$. f is said to be computable, if there is a TM M, that given an access to arbitrarily good approximations of x can output an arbitrarily good approximation of $f(x)$. This is formalized using a notion of an *oracle*. Denote by $\mathbb{D} = \{\frac{m}{2^k} : m \in \mathbb{Z}, k \in \mathbb{N}\}$ – the set of the dyadic numbers. A function $\phi : \mathbb{N} \to \mathbb{D}$ is said to be an *oracle* for a number $x \in \mathbb{R}$ if for every n, $|\phi(n) - x| < 2^{-n}$. In other words, the oracle ϕ allows the machine to query a the number x with any desired precision 2^{-n}.

Definition 5. *The function $f : S \to \mathbb{R}$ is said to be computable on the set $S \subset \mathbb{R}$, if there is an oracle TM $M^\phi(n)$ such that for any $x \in S$ and for any oracle ϕ_x for x, $M^{\phi_x}(n)$ outputs a dyadic $d \in \mathbb{D}$ with $|f(x) - d| < 2^{-n}$.*

In particular, all simple functions, such as the functions available on a standard calculator, are computable. The domain also plays an important role: a constant function $f(x) = a$ is computable on $\mathbb{R}$ if and only if a is a computable number. On the other hand it is always computable on the singleton set $S = \{a\}$ – because it coincides with the computable function $f(x) = x$ on this set.

A set C in $\mathbb{R}^2$ is said to be computable if we can produce arbitrarily good pictures of it. A picture consists of a union of pixels – balls $P = B(q, 2^{-n})$ with centers $q \in \mathbb{D}^2$ on a suitable dyadic grid. For convenience, we assume here that C is closed and bounded. A pixel P should be colored black (included in the picture) if it intersects C. It should be colored white (excluded from the picture) if it is "far" from C. We say that it is "far" if it is at least one pixel radius away from C. The pictures on Fig. 1 are pictures of the corresponding Julia sets according to this definition. This definition is quite robust, and is equivalent to other natural definitions. More details may be found, for example, in [7]. Formally,

Definition 6. *A compact set $C \in \mathbb{R}^2$ is said to be computable, if there is a computable function $\psi : \mathbb{D}^2 \times \mathbb{N} \to \{0, 1\}$ (in the classical sense) satisfying*

$$\psi(q, n) = \begin{cases} 1 & \text{if } B(q, 2^{-n}) \cap C \neq \emptyset \\ 0 & \text{if } B(q, 2 \cdot 2^{-n}) \cap C = \emptyset \\ 0 \text{ or } 1 & \text{otherwise} \end{cases}$$

We are interested in questions of computability of Julia sets. The Julia set J_α depends on the parameter $\alpha \in \mathbb{C}$. As there are uncountably many parameters and only countably many machines, we will need an oracle for α as part of the input to the machine. Thus we are essentially trying to compute the set-valued function $\mathbb{J}$ that maps α to the set J_α.

Definition 7. *The function $\mathbb{J}$ is said to be computable on a set S of parameters α if there is an oracle machine $M^{\phi_1,\phi_2}(q, n)$ such that if (ϕ_1, ϕ_2) are oracles for the real and imaginary parts of α, then $M^{\phi_1,\phi_2}(q, n)$ computes J_α in the sense of Definition 6.*

If $\mathbb{J}$ is computable on a set S of parameters α, we say that the Julia set is *uniformly computable* on S. It can be shown, for example, that $\mathbb{J}$ is not uniformly computable on the entire complex plane $\mathbb{C}$ [12]. If $\mathbb{J}$ is computable on a singleton $S = \{\alpha\}$, we say that the Julia set J_α is *non-uniformly computable*, or just *computable*. This is the setting in which we prove our negative results: the machine for computing J_α has an oracle access to the parameter α, and is also specifically designed to work with this parameter – we are still able to produce parameters α for which this is impossible.

3. JULIA SETS AND SIEGEL DISCS

In this section we discuss the definitions as well as some facts about Julia sets and Siegel discs. We will only cite facts that are directly relevant to the proofs below. An excellent general reference about the mathematics behind Julia sets is [24]. For simplicity, we will only discuss Julia sets J_α of quadratic polynomials of the form $p_\alpha(z) = z^2 + \alpha z$. Using a linear change of coordinates any quadratic polynomial can be brought into this form. As mentioned in the Introduction,

Definition 8. *The* filled *Julia set K_α is the set of points z such that the orbit $z, p_\alpha(z), p_\alpha(p_\alpha(z)), \ldots$ does not escape to ∞. The Julia set J_α is the boundary of K_α: $J_\alpha = \partial K_\alpha$.*

Periodic orbits play a crucial role in the theory of Julia sets. Denote by $p^k_\alpha = p_\alpha \circ \ldots \circ p_\alpha$ the k-th iterate of p_α. If $z = p^k_\alpha(z)$ for some $k > 1$, then z is said to be a periodic point. The smallest such k is called the period of z, and $\{z, p_\alpha(z), \ldots, p^{k-1}_\alpha(z)\}$ is called a periodic orbit of period k. The function $p^k_\alpha(z)$ is a polynomial of degree 2^k. Hence the equation $p^k_\alpha(z) = z$ has a root for every k. It can be shown that there are infinitely many periodic orbits of p_α, of arbitrarily high period k.

For a periodic orbit $o = \{z_1, z_2, \ldots, z_k\}$ of period k, the *multiplier* of o is the derivative of $p^k_\alpha(z)$ at z_1:

$$m(o) = \left(p^k_\alpha(z_1)\right)' = (p_\alpha)'(z_1) \cdot (p_\alpha)'(z_2) \cdot \ldots \cdot (p_\alpha)'(z_k).$$

Note that this value does not depend on the choice of the point z_1 on the orbit o.

The value of $m(o)$ dictates the local behavior of iterations of p_α near the orbit o. If $|m(o)| < 1$, then if we start iterating a point z near some $z_i \in o$, then after k iterations $p^k_\alpha(z)$ will be even closer to z_i – approximately $|m(o)|$ times closer. Thus the orbits of points very close to o are attracted to o, and the orbit o is called *attracting*. Note that in particular, orbits of points near o do not escape to ∞, hence o is in the interior of K_α in this case.

If $|m(o)| > 1$, then the opposite happens, and the orbits of points in a small neighborhood of o always escape that neighborhood. In this case o is said to be *repelling*. The most interesting case is when $|m(o)| = 1$. In this case o is said to be *neutral*. We summarize some of the relevant properties of J_α and the periodic orbits of p_α in the following theorem. Details and proofs may be found in [24].

Theorem 9. *(1) There is at most one non-repelling periodic orbit;*
(2) all repelling periodic orbits are in J_α;
(3) repelling periodic orbits are dense in J_α:
$$J_\alpha = \overline{\{o \mid o \text{ is repelling}\}};$$
(4) if there is a non-repelling orbit, then J_α is connected.

Quadratic Julia sets J_α can be classified according to the status of their only non-repelling orbit. In the case when there is an attracting periodic orbit (Fig. 1(a)), or when the Julia set is not connected (Fig. 1(c)) the Julia set is said to be *hyperbolic*. Hyperbolic Julia sets are the easiest to analyze and are all poly-time computable [8, 26]. If there are no non-repelling orbits and J_α is not hyperbolic, then $K_\alpha = J_\alpha$ has empty interior. All Julia sets J_α such that $K_\alpha = J_\alpha$ has empty interior are computable [3], albeit with no running time guarantees.

It remains to consider the cases when p_α has a neutral periodic orbit o with multiplier $m(o) = e^{2\pi i\theta}$. When θ is rational, the Julia set is said to be *parabolic*. Parabolic Julia sets are also poly-time computable [10]. In the case when θ is irrational, two scenarios are possible, either there is an open neighborhood Δ of o in K_α, or there is not. If there is, J_α is called a *Siegel* Julia set. If there is not, it is called a *Cremer* Julia set. Cremer Julia sets have no interior, and thus are computable. All existing algorithms, however, have impractical running times for such parameters, and no useful pictures of Cremer Julia sets have been produced to date.

While studying Siegel discs and their computability properties, we shall focus on the simplest case, where $\alpha = e^{2\pi i\theta}$ for some $\theta \in \mathbb{R}$ is on the unit circle. In this case $p_\alpha(0) = 0^2 + \alpha \cdot 0 = 0$, and $|p'_\alpha(0)| = |\alpha| = 1$. Hence $\{0\}$ is the only non-repelling orbit for p_α, and its multiplier is α. In this case the Siegel disc is a topological disc Δ_θ around 0. The Riemann Mapping Theorem states that there is a unique analytic map $\phi_\theta : \mathcal{D} \to \Delta_\theta$ from the unit disc $\mathcal{D}$ to Δ_θ such that $\phi(0) = 0$ and $\phi'(0)$ is real and positive.

The map ϕ_θ conjugates the action of p_α to a simple rotation r_θ by an angle θ of the unit disc, $r_\theta : z \mapsto e^{2\pi i\theta} \cdot z$. That is, the following diagram commutative:

$$\begin{tikzcd} \mathcal{D} \arrow[r, "r_\theta"] \arrow[d, "\phi_\theta"'] & \mathcal{D} \arrow[d, "\phi_\theta"] \\ \Delta_\theta \arrow[r, "p_\alpha"] & \Delta_\theta \end{tikzcd}$$

The action of p_α on Δ_θ is just that of a distorted rotation.

By the uniqueness of the map ϕ, the value $\phi'(0) > 0$ is also unique and has a special meaning. It is called the *conformal radius* $r(\theta)$ of Δ_θ. Intuitively, it measures the size of Δ_θ. In particular, the radius of the largest circle around 0 inscribed in Δ_θ is a number between $\frac{1}{4}r(\theta)$ and $r(\theta)$.

The Riemann Mapping Theorem has a constructive proof [4, 16], which means that if we are given a boundary of a region Δ with an arbitrarily high precision, we can compute the Riemann Map $\phi : \mathcal{D} \to \Delta$, and in particular we can compute its conformal radius $r(\Delta) = \phi'(0)$. It can be shown that if we are given a precision ε picture of the Siegel Julia set J_α, we can compute $r(\theta)$ within an error of $O(\sqrt{\varepsilon})$. Thus, if one can produce images of J_α of arbitrarily high precision, then one can compute $r(\theta)$. In fact, it can be shown that the converse statement also holds. Proofs of both directions can be found in [3].

LEMMA 10. *For $\alpha = e^{2\pi i\theta}$, the Julia set J_α and the conformal radius $r(\theta)$ of the Siegel disc are computable relative to each other. That is, given access to α and $r(\theta)$ with an arbitrarily high precision one can compute J_α, while given access to J_α with an arbitrarily high precision one can compute $r(\theta)$.*

This reduces the need to work with the computability of a set to working on the computability of a single number. The function $r(\theta)$ has been studied extensively in Complex Dynamics. In particular, it turns out that it is strongly related to the *continued fraction expansion* of θ. Recall that the continued fraction expansion of $\theta \in (0,1)$ is an expression of the form $\theta = \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots}}}$, where $a_1, a_2, \ldots$ are positive integers. We abbreviate $\theta = [a_1, a_2, \ldots]$ to save space. The continued fraction expansion is unique and infinite when θ is irrational. It is finite when $\theta \in \mathbb{Q}$ (cf. [29], Example 4.1.14.3 for some computational properties of the continued fraction representation). We will also write $\theta_n = [a_n, a_{n+1}, \ldots]$, so that $\theta_n = 1/\{\theta_{n-1}\}$, where $\{\bullet\}$ is the *fractional part* function.

For an irrational $\theta \in [0,1) - \mathbb{Q}$, the *Yoccoz' Brjuno function* $\Phi(\theta)$ is defined by

$$\Phi(\theta) = \sum_{n=1}^{\infty} \theta_1\theta_2 \ldots \theta_{n-1} \log \frac{1}{\theta_n}.$$

Note that it is possible to make θ_n arbitrarily small by making a_n very big. In particular, for any rational q, $\Phi(\theta) \to \infty$ whenever $\theta \to q$. Moreover, it is not hard to see that there are irrational values of θ for which the sum above diverges, in which case we say that $\Phi(\theta) = \infty$. It was proved by Brjuno and Yoccoz that J_α has a Siegel disc if and only if $\Phi(\theta) < \infty$. If $\Phi(\theta) = \infty$ and θ is irrational, J_α has a Cremer point. This gives a sharp combinatorial criterion for testing whether there is a Siegel disc or a Cremer point. Much more recently, Buff and Cheritat [13] have shown a very sharp quantitative relation between the conformal radius of the Siegel disc $r(\theta)$ and $\Phi(\theta)$.

THEOREM 11. *[13] The function $\upsilon(\theta) \equiv \Phi(\theta) + \log r(\theta)$ extends from its domain of definition (where $\Phi(\theta) < \infty$ and $r(\theta) \neq 0$) to a continuous function on $[0,1]$. Moreover, there is an explicit algorithm for computing the values of $\upsilon(m/n)$ for a rational $m/n \in [0,1]$.*

Note that the theorem falls short of guaranteeing that the function υ is computable, since we do not know any effective bounds on its modulus of continuity. It is generally conjectured that υ is computable:

CONJECTURE 12. *The function $\upsilon(\theta)$ is computable of the $[0,1]$ interval.*

In fact, there is a stronger conjecture, called the Marmi-Moussa-Yoccoz Conjecture [22], that gives an explicit bound on the modulus of continuity of υ: whenever $|x-y| < f(\varepsilon) = \varepsilon^2$, $|\upsilon(x) - \upsilon(y)| < \varepsilon$.

Assuming computability of υ, Theorem 11 and Lemma 10 reduce the computational analysis of J_α to that of $\Phi(\alpha)$. Surprisingly, however, despite its simple analytic form, $\Phi(\theta)$ is not always computable. In fact, all the negative statements about computability of Julia sets in Theorems 1, 2 and 3 can also be shown to be true for the function Φ by similar arguments.

4. CHARACTERIZING SIEGEL DISCS WITH COMPUTABLE PARAMETERS

In this section we outline the proof of Theorem 2. The full details of the proof may be found in [11]. The proof is done by giving a precise characterization of the possible values of $r(\theta)$ for computable θ's. First, we need the notion of a right computable number.

DEFINITION 13. *A real number β is right-computable if there is a TM computing a non-increasing dyadic sequence $b_n \in \mathbb{D}$ such that $b_n \searrow \beta$.*

A right-computable number does not have to be computable, as there may be no recursive estimate on the rate of convergence of b_n to β. It is not hard to see that computing right-computable numbers is equivalent to evaluating predicates of the form $P(x) = \exists y R(x,y)$ for a recursive predicate $R(x,y)$ on $\mathbb{N} \times \mathbb{N}$.

LEMMA 14. *For any right-computable number β there is a recursive predicate $R(x,y)$ on $\mathbb{N} \times \mathbb{N}$ such that β is computable relative to $P(x) = \exists y R(x,y)$ and vice versa.*

Conversely, for any computable predicate $R(x,y)$ on $\mathbb{N}\times\mathbb{N}$ and for any interval $(a,b) \subset \mathbb{R}$, there is a right-computable number $\beta \in (a,b)$ such that the predicate $P(x) = \exists y R(x,y)$ is computable relative to β and vice-versa.

The proof is quite straightforward and can be found in the [11]. For example, let $P(x) = \exists y R(x,y)$ for a computable

$R(x,y)$. The number $\beta = 1 - \sum_{P(x)=1} 4^{-x}$ is right computable, since the computable sequence

$$\beta_n = 1 - \sum_{x \le n, \exists y \le n R(x,y)} 4^{-x}$$

is non-increasing and converges to β. It is easy to see that $P(x)$ is computable relative to β and vice-versa. The next lemma characterizes the conformal radii $r(\theta)$ for computable θ's.

LEMMA 15. *A number $r \in [0, \frac{1}{4}]$ is a conformal radius $r = r(\theta)$ for some computable θ if and only if r is right-computable.*

Lemmas 10, 15 and 14 together imply Theorem 2.

PROOF. **(of Theorem 2).** We will only prove the first part here. Note that the first part gives the interesting constructive non-computability results and implies Theorem 1. Let $R(x,y)$ be any computable predicate on $\mathbb{N}^2$, and let $P(x) \equiv \exists y R(x,y)$. By Lemma 14 there is a right-computable number $r \in [0, \frac{1}{4}]$ such that r is computable relative to $P(x)$ and vice-versa. By Lemma 15, $r = r(\theta)$ is the conformal radius of the Siegel disc of $J_{e^{2\pi i\theta}}$ for some computable θ. Finally, by Lemma 10 this implies that r is computable relative to J_α and vice-versa for $\alpha = e^{2\pi i\theta}$. α is computable, as it is given as a computable function of θ. Thus $P(x)$ is computable relative to J_α and vice-versa for a computable α. $\square$

In the remainder of the section we outline the proof of Lemma 15, which is the main technical contribution of the paper. The full proof can be found in [11]. We will only discuss the "r is right-computable $\Rightarrow$ it is the conformal radius $r = r(\theta)$ for some computable θ", this is the more difficult direction that gives us the main negative result. In the proof we explicitly compute θ given the sequence $r_n \searrow r$.

We compute a sequence of θ_n's such that $\theta_n \to \theta$, and $|\theta_n - \theta| < 2^{-n}$, thus computing θ. The elements of the sequence are given by their continued fraction expansion. The expansion is given by a finite sequence of integers $I_n = [a_1, \dots, a_{m_n}]$ followed by an infinite sequence of 1's. Thus $\theta_n = [a_1, a_2, \dots, a_{m_n}, 1, 1, \dots]$. The key ingredient of the proof is our ability to decrease $r(\theta)$ by any given controlled amount ε, while changing $\theta = [a_1, a_2, \dots]$ by an arbitrarily small δ. Thus we constructively obtain a sequence such that $|\theta_n - \theta| < 2^{-n}$, and $r(\theta_n) \searrow r$. The only thing we have to make sure is that $r(\theta) = r$ – that is, that $r(\lim \theta_n) = \lim r(\theta_n)$. The "controlled drop" is achieved by the following lemma.

LEMMA 16. *For any given initial segment $I = [a_0, a_1, \dots, a_n]$ and $m_0 > 0$, write $\omega = [a_0, a_1, \dots, a_n, 1, 1, 1, \dots]$. Then for any $\varepsilon > 0$, we can uniformly compute $m > m_0$, an integer t and an integer N such that if we write $\beta = [a_0, a_1, \dots, a_n, 1, 1, \dots, 1, N, 1, 1, \dots]$, where the N is located in the $n+m$-th position, we have*

$$r(\omega) - 2\varepsilon < r(\beta) < r(\omega) - \varepsilon, \qquad (1)$$

$$\Phi(\beta) > \Phi(\omega), \qquad (2)$$

and for any $\gamma = [a_0, a_1, \dots, a_n, 1, 1, \dots, 1, N, 1, \dots, 1, c_{n+m+t+1}, c_{n+m+t+2}, \dots]$,

$$\Phi(\gamma) > \Phi(\omega) - 2^{-n}. \qquad (3)$$

Condition (1) guarantees the drop, while conditions (2) and (3) are needed in order for the construction to work at the limit. The proof of the lemma can be found in [11]. To see why, in principle, such a drop is possible, recall that the function $\upsilon(\omega) = \Phi(\omega) + \log r(\omega)$ is continuous, hence if we keep β very close to ω, a controlled drop in $r(\omega)$ should correspond to a controlled increase in $\Phi(\omega)$. Recall that for $\beta = [b_1, b_2, \dots]$, with $\beta_k = [b_k, b_{k+1}, \dots]$, $\Phi(\beta) = \sum_{i=1}^{\infty} \beta_1 \beta_2 \dots \beta_{i-1} \log \frac{1}{\beta_i}$. Denote by β^N, the result of substituting the k-th entry of the continued fraction expansion b_k with N for some very big N, then $\beta_i \approx \beta_i^N$ except for $i = k$, we have

$$\Phi(\beta^N) = \sum_{i=1}^{\infty} \beta_1^N \beta_2^N \dots \beta_{i-1}^N \log \frac{1}{\beta_i^N} \approx$$
$$\sum_{i=1, i\neq k}^{\infty} \beta_1 \beta_2 \dots \beta_{i-1} \log \frac{1}{\beta_i} + \beta_1^N \beta_2^N \dots \beta_{k-1}^N \log \beta_k^N \approx$$
$$\sum_{i=1, i\neq k}^{\infty} \beta_1 \beta_2 \dots \beta_{i-1} \log \frac{1}{\beta_i} + \beta_1 \beta_2 \dots \beta_{k-1} \log N. \qquad (4)$$

Note that β_i are all smaller than 1, and in fact $\beta_i \beta_{i+1} < 1/2$ for all i, hence by making k big enough we can always make sure that we can select N to get just the right increase. A "drop" in $r(\omega)$ using a big N is illustrated on Fig. 2(a).

To complete the proof of Lemma 15, we begin with $\gamma_0 = [1, 1, \dots]$ for which $r(\gamma_0) > \frac{1}{4}$. We then perform a series of drops to obtain a sequence $\gamma_k = [I_k, 1, 1, \dots]$, where I_k is the initial segment of the continued fraction for γ_k, such that *(i)* I_{k+1} is the extension of I_k, *(ii)* $r_k + 2^{-k} < r(\gamma_k) < r_k + 2^{-k+1}$, and *(iii)* we can pass to the limit, for $\gamma = \lim \gamma_k$, $r(\gamma) = \lim r(\gamma_k)$. Then γ is computable, since to approximate it with precision 2^{-n} it suffices to know the first $2n$ terms of the continued fraction expansion, and $r(\gamma) = \lim r_k = r$.

A Non-Computable $J_{z^2+\alpha z}$ with a poly-time computable α

The construction above contains a single step for which we can make no estimates on how long it would take. The problem is that while we know that a β satisfying (1) from Lemma 16 exists, the only way to check that is to actually compute $r(\beta)$ – a process that could take a very long time. Assuming the Computability of υ Conjecture, $r(\theta)$ and $\Phi(\theta)$ are computable relative to each other. Hence in Lemma 16(1) it suffices to assure the correct drop in $\Phi(\theta)$. Unlike $r(\theta)$, $\Phi(\theta)$ has an explicit formula, which is relatively easy to evaluate.

If we have to deal with $\Phi(\theta)$ instead of $r(\theta)$ we can assure that the limit of the process in Lemma 16 is poly-time computable. During one step as in the lemma we first choose a very big m. Setting the next m terms of the continued fraction expansion of ω sets the next $\Theta(m)$ of the bits in its binary expansion. This gives us time *poly*(m) to evaluate the number N, which is possible. The details of the construction are outlined in [11].

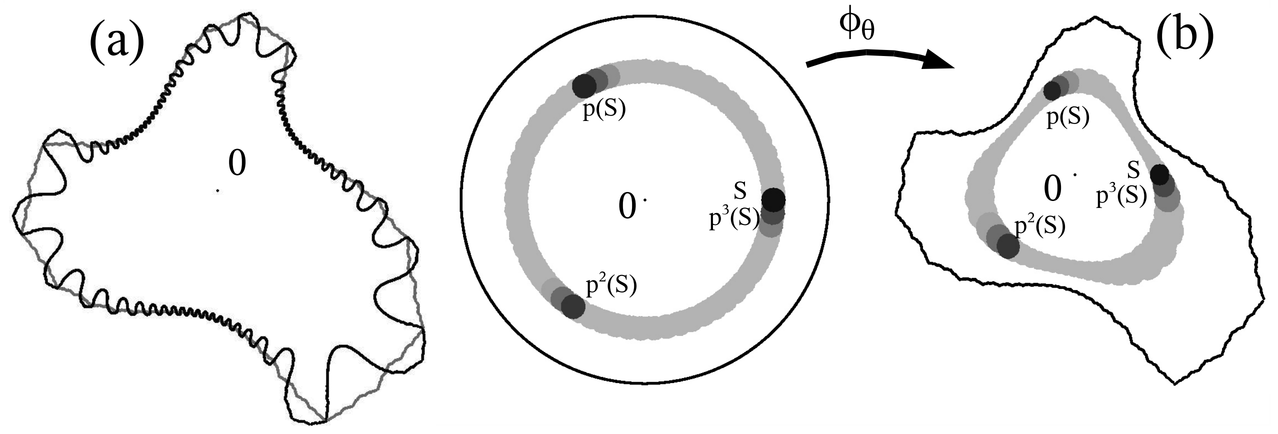

Figure 2: (a) a drop of the conformal radius $r(\omega)$ when switching from $\omega_2 = [3, 20, 1, 1 \ldots]$ (gray) to $\omega_3 = [3, 20, 200, 1, 1 \ldots]$ (black); (b) the first 70 images of S (black) separate 0 from the boundary of the Siegel disc with $\theta = [3, 20, 1, 1 \ldots]$. The action of p_α on the Siegel disc conjugates to a rotation by angle $2\pi\theta$ on the unit disc. The first few images of S are highlighted, note that the rotation is by slightly less then $1/3$ of a circle

5. FILLED JULIA SETS ARE COMPUTABLE

We have seen that quadratic Julia sets may not be computable in some cases. Surprisingly, the *filled* Julia set K_α is always computable. This has previously been known for all cases but the case when there is a Siegel disc [3] – i.e. the cases when J_α is also computable. It remained open whether K_α is always computable when there is a Siegel disc. Here we give an affirmative answer to this question.

An instructive example is the proof that K_α with no interior is computable. In this case $J_\alpha = K_\alpha$. According to Definition 6 the basic decision we should be able to make is given a dyadic point $d \in \mathbb{C}$ and a radius 2^{-n} to decide whether $B(d, 2^{-n})$ intersects K_α or if $B(d, 2 \cdot 2^{-n})$ is disjoint from K_α. One way of presenting such an algorithm is by giving two programs A and B such that at least one of them always terminates, and whenever a program terminates it outputs an acceptable answer.

Program A consists of an infinite loop that on iteration k computes all the periodic orbits of period k within an error of 2^{-n-2}. If d is $2 \cdot 2^{-n}$-close to one of the orbits, the program terminates and outputs 1. By Theorem 9, periodic orbits are dense in K_α and if d is $\frac{7}{4} \cdot 2^{-n}$-close to K_α, Program A will terminate. Since all the periodic orbits are in K_α, the program always outputs a valid answer when it terminates.

Program B consists of an infinite loop that on iteration k approximates the set $p_\alpha^k(B(d, \frac{3}{2} \cdot 2^{-n}))$. If the image escapes the ball $B(0, 4 + |\alpha|^2)$ (i.e. the image escapes to ∞), then the program terminates and outputs 0. If $B(d, \frac{3}{2} \cdot 2^{-n})$ is disjoint from K_α, its image will eventually escape to ∞, and the program will terminate with a valid answer. Here we use the fact that K_α has empty interior.

Thus in the case when K_α has empty interior, running programs A and B in parallel suffices to compute K_α. This is not the case, however when K_α has a Siegel disc. It is possible that a point d is far from the Julia set (hence Program A never terminates), but is inside K_α (hence program B never terminates). Thus we need a program C that terminates and outputs 1 when $B(d, 2^{-n})$ is completely inside K_α. Here we use the fact from Theorem 9 that J_α (and hence K_α) is connected in the Siegel case.

The Siegel disc has a periodic orbit o in the middle. The orbit o can be specified using finite amount of information. Let o' be any other periodic orbit. We know that o' is also in K_α. K_α is connected, hence any set A separating o from o' must intersect K_α.

Program C consists of an infinite loop that on iteration k approximates the set $A_k = \cup_{i=0}^k p_\alpha^i(B(d, \frac{3}{2} \cdot 2^{-n}))$. If A_k covers o or separates o from o' the program terminates and outputs 1. K_α is invariant under the action of p_α, hence if A_k separates o from o' it implies that $A_k \cap K_\alpha \neq \emptyset$, implying that $B(d, \frac{3}{2} \cdot 2^{-n}) \cap K_\alpha \neq \emptyset$ by the invariance of K_α under p_α, thus the algorithm outputs a valid answer.

It remains to see that Program C terminates whenever $B(d, 2^{-n})$ is inside K_α. Recall that the action of p_α on the Siegel disc Δ conjugates to a rotation by an irrational angle θ on a disc $\mathcal{D}$. The set $A = B(d, 2^{-n})$ in Δ corresponds to some open set B in $\mathcal{D}$, and a union of finitely many images of B under the rotation by angle θ will separate the boundary of $\mathcal{D}$ from its center. This corresponds to union of images of A separating o from points on the boundary of Δ, and in particular from o'. The situation is illustrated on Fig. 2(b). Thus Program C will terminate in this case.

6. CONCLUSION AND OPEN PROBLEMS

We have studied the quadratic Julia sets as an archetypical example of a global structure (a repeller in this case) arising in a non-linear dynamical system. A surprising finding is that even when the coefficients of the quadratic polynomial are computable, its Julia set may be non-computable. Thus, answering global questions even in well-understood dynamical systems may be impossible in practice.

Since quadratic polynomials acting in $\mathbb{C}$ may be viewed in many ways as basic examples of non-linear dynamical systems, it is reasonable to expect that more complicated systems would present even more examples of non-computability. Many exciting directions of study are open here. For instance, it should be straightforward to generalize our results to polynomials of higher degrees. On the other hand, it could prove very challenging to study the computability and complexity questions in a complex (or real) dimension higher than one. We note that these studies have a direct practical importance, as numerical modeling remains the main empirical tool in the study of dynamics.

Our study has led us to use analytic tools which only became available in dynamics in the last couple of years. Surprisingly also, the sharpness of our results suggests that nothing less would suffice. We could thus expect that the study of computability problems could lead to new developments in the analytic theory of dynamical systems. Conversely, our results suggest new avenues of study in computability theory itself. For instance, the Yoccoz' function Φ is an intriguing example of a "naturally occurring" non-computable real function. It's simple analytic form, and direct connection to number theory invites further study.

A direction in which there is much room for further development is studying the computational complexity of Julia sets. For example, we now know that all quadratic Cremer Julia sets are computable. However, no informative pictures of these sets have been produced to date. Still, there are no complexity lower bounds to rule out the chance that very efficient algorithms for these sets are waiting to be discovered (and indeed, the recent work [10] seems to suggest this possibility).

Acknowledgments

We would like to thank John Milnor for posing the question of computability of filled Julia sets to us. The first author is grateful to Stephen Cook for the many useful discussions on the subject.

7. REFERENCES

[1] E. Asarin, O. Maler, and A. Pnueli. Reachability analysis of dynamical systems with piecewise-constant derivatives. *Theor. Comp. Sci.*, 138:35–66, 1995.

[2] S. Banach and S. Mazur. Sur les fonctions calculables. *Ann. Polon. Math.*, 16, 1937.

[3] I. Binder, M. Braverman, and M. Yampolsky. Filled julia sets with empty interior are computable. *Journ. of FoCM, to appear*, 2007.

[4] E. Bishop and D. S. Bridges. *Constructive Analysis*. Springer-Verlag, Berlin, 1985.

[5] L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and Real Computation*. Springer-Verlag, New York, 1998.

[6] V. Brattka. Plottable real number functions. In *Marc Daumas and et al., editors, RNC'5 Real Numbers and Computers*, pages 13–30. INRIA, September 2003.

[7] V. Brattka and K. Weihrauch. Computability of subsets of euclidean space I: Closed and compact subsets. *Theoretical Computer Science*, 219:65–93, 1999.

[8] M. Braverman. Hyperbolic Julia sets are poly-time computable. *Electr. Notes Theor. Comput. Sci.*, 120:17–30, 2005.

[9] M. Braverman. On the complexity of real functions. In *Proceedings of 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA*, pages 155–164, 2005.

[10] M. Braverman. Parabolic julia sets are polynomial time computable. *Nonlinearity*, 19(6):1383–1401, 2006.

[11] M. Braverman and M. Yampolsky. Computability of Julia sets. e-print: http://www.arxiv.org/abs/math.DS/0610340, 2006.

[12] M. Braverman and M. Yampolsky. Non-computable Julia sets. *Journ. Amer. Math. Soc.*, 19(3):551–578, 2006.

[13] X. Buff and A. Chéritat. The Brjuno function continuously estimates the size of quadratic Siegel disks. *Annals of Math.*, 164(1):265–312, 2006.

[14] P. Collins. On the computability of reachable and invariant sets. In *IEEE Conf. on Decision and Control*, pages 4187–4192, 2005.

[15] A. Douady and J. H. Hubbard. Etude dynamique des polynômes complexes: I-II. Technical Report 84-02,85-04, Pub. Math. d'Orsay, 1984.

[16] P. Hertling. The effective Riemann mapping theorem. *Theor. Comp. Sci.*, 219(1-2):225–265, 1999.

[17] A. Katok and B. Hasselblatt. *Introduction to the modern theory of dynamical systems*. Cambridge University Press, Cambridge, 1995.

[18] K. Ko. *Complexity Theory of Real Functions*. Birkhäuser, Boston, 1991.

[19] P. Koiran, M. Cosnard, and M. Garzon. Computability with low-dimensional dynamical systems. *Theor. Comp. Sci.*, 132:113–128, 1994.

[20] P. Koiran and C. Moore. Closed-form analytic maps in one and two dimensions can simulate universal turing machines. *Theor. Comp. Sci.*, 210:217–223, 1999.

[21] D. Lacombe. Classes récursivement fermés et fonctions majorantes. *C. R. Acad. Sci. Paris*, 240:716–718, 1955.

[22] S. Marmi, P. Moussa, and J. C. Yoccoz. The Brjuno functions and their regularity properties. *Commun. Math. Phys.*, 186:265–293, 1997.

[23] S. Mazur. *Computable analysis*, volume 33. Rosprawy Matematyczne, Warsaw, 1963.

[24] J. Milnor. *Dynamics in one complex variable. Introductory lectures*. Princeton University Press, 3rd edition, 2006.

[25] C. Moore. Unpredictability and undecidability in dynamical systems. *Phys. Rev. Lett.*, 64:2354–2357, 1990.

[26] R. Rettinger. A fast algorithm for julia sets of hyperbolic rational functions. *Electr. Notes Theor. Comput. Sci.*, 120:145–157, 2005.

[27] R. Rettinger and K. Weihrauch. The computational complexity of some Julia sets. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 177–185, 2003.

[28] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings, London Mathematical Society*, pages 230–265, 1936.

[29] K. Weihrauch. *Computable Analysis*. Springer-Verlag, Berlin, 2000.

[30] A. Yao. Classical physics and the Church-Turing Thesis. Technical Report TR02-062, Electronic Colloquium on Computational Complexity (ECCC), 2002.

[31] N. Zhong. Recursively enumerable subsets of $\mathbb{R}^q$ in two computing models: Blum-Shub-Smale machine and Turing machine. *Theor. Comp. Sci.*, 197:79–94, 1998.