Deciding Robust Instances of an Escape Problem for Dynamical Systems in Euclidean Space

Eike Neumann
Swansea University, UK

Abstract

We study the problem of deciding whether a point escapes a closed subset of $\mathbb{R}^d$ under the iteration of a continuous map $f\colon \mathbb{R}^d \to \mathbb{R}^d$ in the bit-model of real computation. We give a sound partial decision method for this problem which is complete in the sense that its halting set contains the halting set of all sound partial decision methods for the problem. Equivalently, our decision method terminates on all problem instances whose answer is robust under all sufficiently small perturbations of the function. We further show that the halting set of our algorithm is dense in the set of all problem instances. While our algorithm applies to general continuous functions, we demonstrate that it also yields complete decision methods for much more rigid function families: affine linear systems and quadratic complex polynomials. In the latter case, completeness is subject to the density of hyperbolicity conjecture in complex dynamics. This in particular yields an alternative proof of Hertling's (2004) conditional answer to a question raised by Penrose (1989) regarding the computability of the Mandelbrot set.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Dynamical Systems, Computability in Analysis, Non-Linear Functions

Digital Object Identifier 10.4230/LIPIcs.MFCS.2025.79

Related Version *Full Version*: https://arxiv.org/abs/2506.21481 [26]

Funding This work was supported by Swansea University.

1 Introduction

A large number of problems in a diverse range of areas, such as program verification [31], numerical analysis [40, 35], economics [4], probabilistic systems [1, 6], and biology [45, 24] reduce to a problem of the following type: Given a set $A \subseteq X$, a function $f\colon X \to X$, and a point $x_0 \in A$, decide whether x_0 escapes A under finitely many iterations of f.

Unsurprisingly, the study of automated methods for the solution of such problems has received much attention [41, 43, 29, 28, 30, 27, 21, 36, 37, 19, 16, 5, 20].

Existing work on automated decision methods largely focusses on linear systems. In fact, the scope of automated decision methods for non-linear systems appears to be quite limited: the above problem is undecidable [42] already when $f\colon \mathbb{R}^d \to \mathbb{R}^d$ is a piecewise linear function and A is a polyhedron in $\mathbb{R}^d$.

All of the aforementioned results assume that the system is specified exactly, by means of rational or algebraic numbers. In application areas such as engineering and the natural sciences, it may be unrealistic to assume that the system under consideration is known to perfect accuracy. Rather, one should assume that the system is known only up to some (presumably small) error with respect to a given distance function. In this context, one is arguably less interested in deciding the problem for a single given instance, but to exhibit a neighbourhood of the given instance for which the answer to the problem remains constant – provided that such a neighbourhood exists. If the given instance lies on a "decision boundary", *i.e.*, the answer to the problem is sensitive to arbitrarily small perturbations of the instance, not much appears to be gained by solving the problem for that specific instance. In some

50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025).
Editors: Paweł Gawrychowski, Filip Mazowiecki, and Michał Skrzypczak; Article No. 79; pp. 79:1–79:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

sense, this can also be viewed as an opportunity to increase the scope of automated decision methods, since the aforementioned undecidability results usually occur due to instances that lie on such decision boundaries.

This has led to various problems of this type being studied in various, generally non-equivalent, formalisations of “robust decidability” [25, 2, 38, 14, 3].

The aim of this paper is to demonstrate by means of a case study that *computable analysis* constitutes a very suitable framework for the study of robust decidability questions. Computable analysis is the study of computation on data that is not specified by a complete finite description, but rather given as an infinite sequence of increasingly accurate approximations. For example, a real number $x \in \mathbb{R}$ can be encoded by an infinite sequence $(I_n)_n$ of nested intervals with rational endpoints such that $\{x\} = \bigcap_{n\in\mathbb{N}} I_n$. An algorithm for computing a real number $x \in \mathbb{R}$ takes as input a non-negative integer $n \in \mathbb{N}$ and outputs an interval I_n with rational endpoints. The algorithm is required to ensure that the sequence $(I_n)_n$ is nested and satisfies $\{x\} = \bigcap_{n\in\mathbb{N}} I_n$. An algorithm for computing a real function $f\colon \mathbb{R} \to \mathbb{R}$ takes as input an infinite sequence of intervals $(I_m)_m$ which is guaranteed to encode a real number $x \in \mathbb{R}$, and returns as output an infinite sequence of intervals $(J_n)_n$. The algorithm is required to ensure that for all input sequences $(I_m)_m$ encoding a real number x, the output sequence $(J_n)_n$ encodes the real number $f(x)$. The infinite sequence $(I_m)_m$ is presented to the algorithm as a “black box”: the algorithm can query the black box for any integer $m \in \mathbb{N}$ to receive the interval I_m. Formally, this idea can be realised by using oracle Turing machines with an “input oracle” [22] or, equivalently, by using Turing machines with an infinite read-only tape that contains the entire infinite input sequence [44]. It is important to observe that in the definition above, an algorithm for computing a real function operates on *all* real numbers, not just on computable ones. This should be contrasted with *Markov computability* [23, 18], where algorithms operate only on computable objects which are presented as indexes of Turing machines that compute the objects.

One may object that the above model of computation is still unrealistic: Our initial motivation – to study systems that are not known exactly – suggests to consider decision methods for objects that are known only to *some fixed finite precision.* While in computable analysis objects are not given by an infinitely precise finite description, they can still be approximated to *arbitrary*, rather than fixed, finite precision. However, an algorithm that computes a function from one space to another can be automatically lifted to one that computes the range of the function over any given compact subset of the space [32, Proposition 11], automatically yielding an effective method for operating on objects that are known only to some fixed precision. See also [25, Section 2] for a discussion of this in the context of decision problems.

We study the following very general instantiation of the decision problem mentioned above: Given a continuous function $f\colon \mathbb{R}^d \to \mathbb{R}^d$, a closed set $A \subseteq \mathbb{R}^d$, and a point $x_0 \in A$, determine whether the point escapes A under iteration of f. To make this precise, we introduce standard encodings of points, sets, and functions, which we discuss in detail in Section 2.3. Here, the point escapes if and only if there exists $n \in \mathbb{N}$ such that $f^n(x_0) \notin A$ – *i.e.*, the point is said to escape if it leaves the set after some number of function applications, even if it re-enters the set after further applications of f. We will refer to this problem as the *Point Escape Problem.*

Since the inputs to an algorithm are given as infinite sequences of increasingly accurate approximations, there is no hope to obtain a correct algorithm that halts on every input for *any* non-trivial problem of the above type: every computable function must be continuous, and every continuous function from a connected space to a two-point space is constant. Since no *total* algorithm exists for deciding the problem in question, the next best alternative is to ask for a partial algorithm that halts on as many problem instances as possible. Accordingly, we will call a partial algorithm for deciding a problem *complete* if its halting set contains the halting set of all correct partial algorithms deciding the same problem. This is equivalent to asking that the algorithm halt on all problem instances whose answer is stable under small perturbations. See Section 2.2 for a more formal discussion and [25, Proposition 2.1] for a proof of this equivalence. Observe that for discrete spaces such as the natural numbers, an algorithm is complete in this sense if and only if it halts on all inputs, so that the above definition of completeness is a generalisation of the usual definition in classical computability.

Our algorithm for solving the Point Escape Problem is rather straightforward: We keep track of an overapproximation O of a finite initial segment $\{x_0, f(x_0), \dots, f^N(x_0)\}$ of the orbit, as well as of overapproximations Q_i of the individual points $\{f^i(x_0)\}$ in the orbit. If we witness that Q_i is separated from the closed set A, then we conclude that x_0 escapes under f. If we witness that O is included in the interior of A and we can find an overapproximation O' of the image $f(O)$ with $O' \subseteq O$, then we conclude that the point x_0 must be trapped. If the overapproximations we have computed do not allow us to draw either conclusion, we compute more accurate overapproximations to longer initial segments of the orbit, until we are able to make a decision.

Our main contribution is to show that this simple algorithm is complete in the above sense. The key idea is to show that the search for an invariant set for f is guaranteed to terminate provided that f has a *robust invariant* whose interior contains $f(x_0)$. A compact set $V \subseteq \mathbb{R}^d$ is called a robust invariant for f if $f(V) \subseteq V^\circ$, where V° denotes the interior of V. We show that if x_0 is trapped but $f(x_0)$ is not contained in a robust invariant, then x_0 escapes under arbitrarily small perturbations of f and A.

We will further show that if x_0 is trapped but $f(x_0)$ is not contained in a robust invariant and $A \neq \mathbb{R}^d$, then x_0 escapes A under arbitrarily small perturbations of f alone, with A being fixed. This implies that our algorithm, which takes x_0 and A as inputs, is complete for all fixed x_0 and $A \subsetneq \mathbb{R}^d$, when only the function f is given as an input. Thus, a bespoke algorithm for fixed special sets or initial points, say $x_0 = 0$ or $A = [0,1]^d$ will not halt on more functions than our general algorithm.

Since a complete algorithm is only required to halt on robust instances, one is lead to the problem of determining the "size" of the robust instances. In the worst case, a problem may not have any robust instances at all, so that a complete decision method is given by the algorithm that never halts. We show that the halting set of our algorithm is "large", in the sense that the algorithm halts on a dense set of inputs and that the set of trapped problem instances is the closure of its interior.

Finally, which problem instances are robust depends on the class of functions and how these functions are represented (by our aforementioned result it surprisingly does not essentially depend on how the sets or initial points are represented). We allow arbitrary continuous functions as inputs, given by the weakest representation that makes function evaluation uniformly computable.

In general, if the problem is restricted to a smaller class of functions, potentially with a representation that induces a stronger topology, then previously non-robust instances may become robust, so that our algorithm will in general fail to be complete for restrictions of the Point Escape Problem to smaller function classes.

Indeed, The topology induced by our representation of continuous functions is the topology of uniform convergence on compact sets. In this topology, every neighbourhood of a function f contains functions g that agree with f on some compact set, but differ arbitrarily from f

outside this compact set. For this reason, our algorithm will fail to halt on very simple-looking problem instances where the given set is unbounded. Consider for example the problem instance (A, f, x_0) consisting of the set $A = \{x \in \mathbb{R} \mid x \geq 0\}$, the function $f(x) = 2x$, and the initial point $x_0 = 1$. Since f does not map any compact set into its interior, our algorithm will fail to halt on this problem instance.

However, we will show that systems such as the above can be effectively treated by our algorithm after first applying a suitable compactification. More specifically, we show that the problem of deciding whether a point escapes a polyhedron (which is in general unbounded) under an affine linear map reduces to the Point Escape Problem for continuous functions in such a way that robust problem instances get mapped to robust problem instances. We conjecture that this reduction extends to more general non-linear systems that are sufficiently well behaved "at infinity".

To discuss the applicability of our algorithm to a class of much more rigid non-linear systems, we consider the problem of deciding whether the origin in the complex plane has an unbounded orbit under a quadratic polynomial, represented by a single complex parameter. The existence of a complete algorithm for this problem is closely related to the computability of the Mandelbrot set [17]. We give a reduction of this problem to the Point Escape Problem that maps robust instances to robust instances if and only if the hyperbolicity conjecture holds true.

The rest of the paper is structured as follows: in Section 2 we review the relevant background material from computable analysis, discuss completeness of decision methods over continuous data, and formally introduce our representations of points, sets, and functions. Section 3 contains a formal description of our main algorithm and our main results: that the algorithm is correct, complete, and generically terminating. In Sections 4 and 5 we describe our applications to linear and quadratic systems respectively.

2 Preliminaries

2.1 Computable Analysis

We review some basic definitions and results from computable analysis. Good introductions to the subject are given in [44, 11, 22, 12, 32]. The key idea for computing with first-order objects such as real numbers and real functions is to encode them via integer sequences. Let $\mathbb{N}^{\mathbb{N}}$ denote the space of all sequences $(p_n)_n$ of non-negative integers. We make $\mathbb{N}^{\mathbb{N}}$ into a topological space by endowing it with the product topology, which is generated by the distance function $d((p_n)_n, (q_n)_n) = 2^{-\inf\{n \mid p_n \neq q_n\}}$. A sequence $(p_n)_n$ is called *computable* if there exists a Turing machine which takes as input a natural number n and returns as output the natural number p_n.

An oracle Turing machine *computes* a partial function $f\colon \subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ if for all oracles Φ with $(\Phi(n))_n \in \operatorname{dom} f$ and all natural numbers N, the machine halts in finite time on input N with oracle Φ and outputs the number q_N, where $(q_m)_m = f((\Phi(n))_n)$. Observe that the oracle Φ is not a fixed function, but part of the input. Further, observe that we do not constrain the machine's behaviour for oracle-inputs outside the domain of f. A partial function $f\colon \subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ is called *computable* if it is computed by some Turing machine.

A *represented space* (X, δ_X) is a set X together with a partial surjective map $\delta_X\colon \subseteq \mathbb{N}^{\mathbb{N}} \to X$. We will usually write just X for the represented space (X, δ_X) when δ_X is implicit or inferable from the context. Let X be a represented space, and let $x \in X$. A point $p \in \mathbb{N}^{\mathbb{N}}$ is called a *name* of x if $p \in \operatorname{dom} \delta_X$ and $\delta_X(p) = x$. The point x is called *computable* if it has a computable name.

Let $f\colon X \to Y$ be a function between represented spaces. A partial function $F\colon \subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ is called a *realiser* of f if $\operatorname{dom} F \supseteq \operatorname{dom} \delta_X$ and $\delta_Y \circ F = f \circ \delta_X$. The function f is called *computable* if it has a computable realiser. It is called *continuous* if it has a continuous realiser (with respect to the relative topology induced by the product topology on $\mathbb{N}^{\mathbb{N}}$).

If X is a set and $\delta_X\colon \subseteq \mathbb{N}^{\mathbb{N}} \to X$ and $\delta'_X\colon \subseteq \mathbb{N}^{\mathbb{N}} \to X$ are representations, we say that δ_X and δ'_X are *computably equivalent* or just *equivalent* if the identity $\mathrm{id}_X\colon X \to X$ on X is computable as a map $\mathrm{id}_X\colon (X, \delta_X) \to (X, \delta'_X)$ and as a map $\mathrm{id}_X\colon (X, \delta'_X) \to (X, \delta_X)$ between represented spaces. By replacing "computable" with "continuous" in the above definition, we obtain the definition of *topological equivalence*.

Any representation $\delta_X\colon \subseteq \mathbb{N}^{\mathbb{N}} \to X$ induces a topology on X, namely the final topology, where $U \subseteq X$ is open if and only if $\delta_X^{-1}(U)$ is an open subset of $\operatorname{dom} \delta_X$ (with respect to the relative topology induced by the product topology on $\mathbb{N}^{\mathbb{N}}$). We will call this topology *the topology of the represented space* X. For a function $f\colon X \to Y$ between represented spaces, we have two a priori distinct notions of continuity available: continuity in the sense of having a continuous realiser, and *topological continuity* in the sense of being continuous with respect to the topologies on X and Y. Any continuous function is topologically continuous, but the converse is false in general. A representation $\delta_X\colon \subseteq \mathbb{N}^{\mathbb{N}} \to X$ is called *admissible* if all topologically continuous functions $f\colon Y \to X$, where Y is a represented space, are continuous (cf. [32, Theorem 36]). All representations we consider in this paper will be admissible. Beyond this, we will only consider representations that are *open maps*, *i.e.* representations where every prefix of a name defines an open subset of the represented space. Intuitively, this means that all names contain the same amount of information.

2.2 Complete Decision Methods

An algorithm over discrete data is called *complete* if it halts on all inputs. A set is said to be *decidable* if there exists a complete algorithm that correctly determines whether a given input belongs to the set. If we apply the same definition to continuous data, we often end up with a trivial notion: if X is a represented space such as $\mathbb{R}^d$ or $C(\mathbb{R}^d, \mathbb{R}^d)$ whose associated topology is connected, then the only decidable subsets of X in in the above sense are the empty set and X itself, for a decision method defines a computable function of type $X \to \{0, 1\}$ and this function must be continuous.

To obtain a more meaningful notion, the definition of "complete algorithm" must be extended in a different way. Let X be a represented space. Let $A \subseteq X$. A *(partial) decision method* for A is an algorithm that takes as input the name of a point $x \in X$ and either runs forever or halts in finite time. Upon halting, the algorithm is required to output the integer 1 if $x \in A$ and the integer 0 if $x \notin A$. The *halting set* of a decision method is the set of all $x \in X$ such that the algorithm halts on all names of x (for open representations, a decision method that halts on *some* name of x automatically extends to a decision method that halts on all names of x). Letting ∂A denote the boundary of A, it is easy to see that the halting set of a decision method is contained in $X \setminus \partial A$. A decision method for A is *complete* if its halting set is equal to $X \setminus \partial A$. A subset $A \subseteq X$ which admits a complete decision method is called *maximally partially decidable* or simply *decidable*. Observe that for any space X which carries the discrete topology, such as $X = \mathbb{N}$, any subset of X has empty boundary, so that our definition of decidability agrees with the usual definition. For spaces such as $X = \mathbb{R}^d$, this definition of "decidability" yields a considerably richer structure than the trivial one induced by the naïve direct generalisation. For the spaces we are interested in, completeness can be characterised as a kind of optimality: If X admits an open representation, then a decision method for $A \subseteq X$ is complete if and only if its halting set contains the halting set of every decision method for A. This definition is further discussed and motivated in [25].

We will call a problem instance $x \in X$ a *robust instance* if $x \notin \partial A$, and a *boundary instance* otherwise. Thus, an algorithm is complete if and only if it halts on all robust instances. Observe that if we witness an algorithm halting on a given problem instance $x \in X$, then we can identify a prefix $p \in \mathbb{N}^*$ of the given name of x such that the algorithm halts on all names that extend p. When the representation of X is open, this prefix corresponds to an open neighbourhood of x where the answer to the problem is constant.

2.3 Encoding Points, Sets, and Functions

We describe representations of points in $\mathbb{R}^d$, continuous functions $\mathbb{R}^d \to \mathbb{R}^d$, and closed and compact subsets of $\mathbb{R}^d$. Our representations of points and functions are up to equivalence the usual standard representations of computable analysis that can be found in the literature [44]. Our representations of sets are equivalent to a join of certain standard representations.

Before we introduce our representations, let us introduce some standard notation and terminology. Throughout, we will work with the supremum norm $||x||_\infty = \max\{|x_i| \mid i \in \{1,\dots,d\}\}$ and its induced metric $d(x,y) = ||x-y||_\infty$ on $\mathbb{R}^d$. For $x \in \mathbb{R}^d$ and $r > 0$, we let $B(x,r) = \{y \in \mathbb{R}^d \mid ||x-y||_\infty < r\}$ and $\overline{B}(x,r) = \{y \in \mathbb{R}^d \mid ||x-y||_\infty \leq r\}$ denote the open and closed ball of radius r about x respectively. In $\mathbb{R}^d$, the closed ball of radius r about x is the closure of the open ball of radius r about x. For a set $S \subseteq \mathbb{R}^d$, we let $B(S,r) = \{y \in \mathbb{R}^d \mid \exists x \in S.\, ||x-y||_\infty < r\} = \bigcup_{x\in S} B(x,r)$ and $\overline{B}(S,r) = \{y \in \mathbb{R}^d \mid \exists x \in S.\, ||x-y||_\infty \leq r\} = \bigcup_{x\in S} \overline{B}(x,r)$. Observe that $B(S,r)$ is always an open set, while $\overline{B}(S,r)$ is not necessarily a closed set. In particular, $\overline{B}(S,r)$ is not necessarily equal to the closure of $B(S,r)$. This is the case, however, if the set S is closed. For a proof, see [26, Proposition 23].

For continuous maps $f,g\colon \mathbb{R}^d \to \mathbb{R}^d$ we let $||f-g||_\infty = \sup\{||f(x)-g(x)||_\infty \mid x \in \mathbb{R}^d\} \in [0,+\infty]$. For a compact set $K \subseteq \mathbb{R}^d$ let $||f-g||_{\infty,K} = \sup\{||f(x)-g(x)||_\infty \mid x \in K\} \in [0,+\infty)$. The set K is allowed to be empty in this definition. We define a distance function on the space $C(\mathbb{R}^d,\mathbb{R}^d)$ of all continuous maps by

$$d(f,g) = \sum_{n\in\mathbb{N}} 2^{-n} \min\left\{1, ||f-g||_{\infty,\overline{B}(0,2^n)}\right\}. \tag{1}$$

We choose this metric since its induced topology is the same as the topology induced by the weakest representation of the space of continuous functions that renders evaluation computable – see Proposition 4.

In order to formulate our main algorithm, it will be convenient to work with subdivisions of $\mathbb{R}^d$ into dyadic rational cubes of uniform size. To facilitate this, we choose representations based on such subdivisions.

Let $\mathcal{Q}^d_n = \left\{\left[\frac{x_1}{2^n}, \frac{x_1+1}{2^n}\right] \times \dots \times \left[\frac{x_d}{2^n}, \frac{x_d+1}{2^n}\right] \mid x_1,\dots,x_d \in \mathbb{Z}\right\}$ denote the set of all d-dimensional dyadic rational cubes of side-length 2^{-n}. The sets $\mathcal{Q}^d_n$ subdivide $\mathbb{R}^d$ into a cubical mesh of mesh-width 2^{-n}, with the interiors of distinct cubes being disjoint. Let $\mathcal{Q}^d_* = \bigcup_{n\in\mathbb{N}} \mathcal{Q}^d_n$.

For a subset $A \subseteq \mathcal{Q}^d_*$ of $\mathcal{Q}^d_*$, we write $|A| = \bigcup_{Q\in A} Q \subseteq \mathbb{R}^d$. For a set $A \subseteq \mathbb{R}^d$, we write $\mathcal{Q}^d_n(A) = \{Q \in \mathcal{Q}^d_n \mid Q \cap A \neq \emptyset\}$. For sets A and B, we write $A \subseteq_{\text{fin}} B$ to indicate that A is a finite subset of B.

For a set $A \subseteq \mathbb{R}^d$, we let $\operatorname{diam} A = \sup\{d(x,y) \mid x,y \in A\}$ denote its *diameter*. We write A° for the *interior* of A, $\overline{A}$ for the *closure* of A, and ∂A for the *boundary* of A. For sets $A, B \subseteq \mathbb{R}^d$, we write $A \Subset B$ if $\overline{A} \subseteq B^\circ$. It is clear that for finite subsets A, B of $\mathcal{Q}^d_*$ the relations $|A| \subseteq |B|$ and $|A| \Subset |B|$ are decidable.

We will now introduce our representations of points, sets, and functions. Elements of $\mathbb{Q}^d_*$ and finite sets of elements of $\mathbb{Q}^d_*$ can be coded by natural numbers. We will fix appropriate encodings, without making them explicit.

▶ **Definition 1.** *Define a representation $\rho^d \colon \subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{R}^d$ as follows:*

Up to coding, a ρ^d-name of a point $x \in \mathbb{R}^d$ is a sequence $(X_n)_n$ of finite sets $X_n \subseteq_{\mathit{fin}} \mathbb{Q}^d_n$ satisfying $x \in |X_n|^\circ$ for all n, $|X_{n+1}| \subseteq |X_n|$ for all n, and $\operatorname{diam}|X_n| \to 0$ as $n \to \infty$.

▶ **Proposition 2.** *The representation ρ^d is open and admissible and induces the standard Euclidean topology on $\mathbb{R}^d$, i.e., the topology generated by the norm $||\cdot||_\infty$.*

Proof. See [26, Proposition 26]. ◀

A code of a multi-valued function $F\colon \mathbb{Q}^d_n\left([-2^n,2^n]^d\right) \rightrightarrows \mathbb{Q}^d_n$ with finite values is an integer that encodes a list of the form $\langle (Q_0, \langle R_{0,0}, \dots, R_{0,N_0}\rangle), \dots, (Q_M, \langle R_{M,0}, \dots, R_{M,N_M}\rangle)\rangle$ where the sequence $(Q_i)_i$ contains every element of the finite set $\mathbb{Q}^d_n\left([-2^n,2^n]^d\right)$ exactly once, and for all i, j we have $R_{i,j} \in \mathbb{Q}^d_n$. Such a list encodes the function that sends the cube Q_i to the finite set $\{R_{i,0}, \dots, R_{i,N_i}\}$.

▶ **Definition 3.** *Define a representation $\left[\rho^d \to \rho^d\right] \colon \subseteq \mathbb{N}^{\mathbb{N}} \to C(\mathbb{R}^d, \mathbb{R}^d)$ as follows:*

Up to coding, a $\left[\rho^d \to \rho^d\right]$-name of a continuous function $f\colon \mathbb{R}^d \to \mathbb{R}^d$ is a sequence $(F_n)_n$ of set-valued maps $F_n\colon \mathbb{Q}^d_n\left([-2^n,2^n]^d\right) \rightrightarrows \mathbb{Q}^d_n$ with finite values satisfying the following requirements:

1. *$f(Q) \subseteq |F_n(Q)|^\circ$ for all $Q \in \mathbb{Q}^d_n\left([-2^n,2^n]^d\right)$.*
2. *If $Q \in \mathbb{Q}^d_n\left([-2^n,2^n]^d\right)$ and $Q' \in \mathbb{Q}^d_{n+1}\left(\left[-2^{n+1},2^{n+1}\right]^d\right)$ with $Q' \subseteq Q$, then $|F_{n+1}(Q')| \subseteq |F_n(Q)|$.*
3. *If $(Q_n)_{n\geq m}$, $m \in \mathbb{N}$, is a sequence of cubes with $Q_n \in \mathbb{Q}^d_n\left([-2^n,2^n]^d\right)$, $Q_{n+1} \subseteq Q_n$, and $\operatorname{diam} Q_n \to 0$ as $n \to \infty$, then $\operatorname{diam}|F_n(Q_n)| \to 0$ as $n \to \infty$.*

▶ **Proposition 4.** *The representation $\left[\rho^d \to \rho^d\right]$ is open and admissible and induces the compact-open topology on $C\left(\mathbb{R}^d, \mathbb{R}^d\right)$. This topology further coincides with the topology induced by the metric (1).*

Proof. See [26, Proposition 27]. ◀

For a set-valued function $G\colon X \rightrightarrows Y$ and a set $A \subseteq X$, we write $G(A) = \bigcup_{x\in A} G(x) \subseteq Y$.

The encoding of continuous functions makes function evaluation on points uniformly computable: if $(X_n)_n$ is a name of a point $x \in \mathbb{R}^d$ and $(F_n)_n$ is a name of a function $f\colon \mathbb{R}^d \to \mathbb{R}^d$ then $(F_n(X_n))_n$ is a name of the point $f(x)$ – potentially up to shifting the sequence to make F_n defined on X_n for all n. Further, if δ is any representation of $C(\mathbb{R}^d, \mathbb{R}^d)$ that renders evaluation computable, then there exists an algorithm which translates a δ-name of f to a name $(F_n)_n$ of f with respect to the above representation. For a proof see [44, 39].

Continuous functions on compact sets are uniformly continuous. It is a folklore result in computable analysis that this fact is effectively witnessed by the above encodings (cf. also [22, Theorem 2.24]):

▶ **Lemma 5.** *Let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function. Let K be a compact set. Then for all n there exists m such that for all $Q \in \mathbb{Q}^d_m(K)$ we have $\operatorname{diam}|F_n(Q)| < 2^{-n}$.*

Proof. By definition, for all $x \in K$ there exists $Q_x \in \mathbb{Q}^d_{m_x}(K)$ with $\operatorname{diam}|F_{m_x}(Q_x)| < 2^{-n}$. By monotonicity, if $\mathbb{Q}^d_m \ni Q \subseteq Q_x^\circ$, then $\operatorname{diam}|F_m(Q)| \leq \operatorname{diam}|F_{m_x}(Q_x)| < 2^{-n}$.

By compactness of K, the cover $\bigcup_{x\in K} Q_x^\circ$ must have a finite subcover $Q_{x_1}^\circ, \dots, Q_{x_s}^\circ$. Now, there exists a number $m \in \mathbb{N}$ such that every $Q \in \mathcal{Q}_m^d(K)$ is contained in some $Q_{x_i}^\circ$ with $i \in \{1, \dots, s\}$. The claim follows. ◀

A closed set will be represented by a sequence of lists of cubes that exhaust its complement, together with a sequence of lists of cubes that exhaust its interior.

▶ **Definition 6.** *Define a representation $\alpha^d\colon \subseteq \mathbb{N}^\mathbb{N} \to \mathcal{A}(\mathbb{R}^d)$ of the set of closed subsets of $\mathbb{R}^d$ as follows: Up to coding, a α^d-name of a closed set $A \subseteq \mathbb{R}^d$ is a sequence of pairs $(I_n^A, E_n^A)_n$ of finite sets $I_n^A, E_n^A \subseteq_{\text{fin}} \mathcal{Q}_n^d$, satisfying the following properties:*

1. $|I_n^A| \Subset A$ *for all* n.
2. $A^\circ = \bigcup_{n\in\mathbb{N}} |I_n^A|^\circ$.
3. $|E_n^A| \Subset \mathbb{R}^d \setminus A$ *for all* n.
4. $\mathbb{R}^d \setminus A = \bigcup_{n\in\mathbb{N}} |E_n^A|^\circ$.
5. *If* $Q \in I_n^A$ *and* $Q' \in \mathcal{Q}_{n+1}^d$ *with* $Q' \subseteq Q$, *then* $Q' \in I_{n+1}^A$.
6. *If* $Q \in E_n^A$ *and* $Q' \in \mathcal{Q}_{n+1}^d$ *with* $Q' \subseteq Q$, *then* $Q' \in E_{n+1}^A$.

▶ **Proposition 7.** *The representation α^d is open and admissible.*

Proof. See [26, Proposition 28]. ◀

Observe that if $(X_n)_n$ is a name of a point $x \in \mathbb{R}^d$ and if $(I_n^A, E_n^A)_n$ is a name of a closed set $A \subseteq \mathbb{R}^d$, then $x \in A^\circ$ if and only if for all large n we have $X_n \Subset I_n^A$ and $x \in \mathbb{R}^d \setminus A$ if and only if for all large n we have $X_n \Subset E_n^A$.

This property extends to arbitrary compact sets: If $(I_n^A, E_n^A)_n$ is a name of a closed set, and K is a compact set which is contained in A° or $\mathbb{R}^d \setminus A$, then this containment is witnessed by a finite initial segment of the name $(I_n^A, E_n^A)_n$.

▶ **Proposition 8.** *Let $K \subseteq \mathbb{R}^d$ be a compact set. Let $(I_n^A, E_n^A)_n$ be a name of a closed set A. Then $K \subseteq A^\circ$ if and only if for all large $n \in \mathbb{N}$, the set K is contained in $|I_n^A|^\circ$. We have $K \subseteq \mathbb{R}^d \setminus A$ if and only if for all large $n \in \mathbb{N}$, the set K is contained in $|E_n^A|^\circ$.*

Proof. We show the claim for $K \subseteq A^\circ$. The second claim is proved analogously. If $K \subseteq |I_n^A|^\circ$ for some n then $K \subseteq A^\circ$ by definition. Now, assume that $K \subseteq A^\circ$. Since $|I_n^A|^\circ \subseteq |I_{n+1}^A|^\circ$ for all n, it suffices to show that there exists $n \in \mathbb{N}$ such that $K \subseteq |I_n^A|^\circ$. We have $K \subseteq A^\circ = \bigcup_{n\in\mathbb{N}} |I_n^A|^\circ$ Since K is compact, the open cover on the right-hand side must have a finite subcover. The claim follows. ◀

3 The Point Escape Problem

We will now present our algorithm for maximally partially deciding whether a point escapes a closed set under the iteration of a continuous function. More precisely, we consider the following decision problem:

▶ **Definition 9.** *The* Point Escape Problem *asks to decide for a given $d \in \mathbb{N}$, a continuous function $f\colon \mathbb{R}^d \to \mathbb{R}^d$, given via a $[\rho^d \to \rho^d]$-name, a closed set $A \subseteq \mathbb{R}^d$, given via an α^d-name, and a given point $x_0 \in A$, given via a ρ^d-name, whether there exists a natural number $n \in \mathbb{N}$ with $f^n(x) \notin A$.*

An instance (f, A, x_0) of the Point Escape Problem where $f^n(x) \notin A$ for some $n \in \mathbb{N}$ is called an escaping instance*. An instance (f, A, x_0) where $f^n(x) \in A$ for all n is called a* trapped instance*.*

Algorithm 1 Point Escape Algorithm.

procedure POINT-ESCAPE($(F_n)_n, ((I_n^A, E_n^A)_n)_n, (X_n)_n$)
  for $n = 0, 1, \ldots$ **do**
    case POINT-ESCAPE-STAGE($n, F_n, I_n^A, E_n^A, X_n$) **of**
      `Escapes` $\Rightarrow$ **return** `Escapes`.
      `Trapped` $\Rightarrow$ **return** `Trapped`.
      `Unknown` $\Rightarrow$ **continue**.
  end for
end procedure

Algorithm 2 Point Escape Algorithm – n^{th} Stage.

procedure POINT-ESCAPE-STAGE($n \in \mathbb{N}, F\colon \mathcal{Q}_n^d\left([-2^n, 2^n]^d\right) \rightrightarrows \mathcal{Q}_n^d, I^A \subseteq_{\text{fin}} \mathcal{Q}_n^d, E^A \subseteq_{\text{fin}} \mathcal{Q}_n^d, X \subseteq_{\text{fin}} \mathcal{Q}_n^d$)
  Let $C_n = \mathcal{Q}_n^d \setminus \mathcal{Q}_n^d\left([-2^n, 2^n]^d\right)$.
  if $|X| \cap |C_n| \neq \emptyset$ **then** — If x_0 is not certainly contained in $[-2^n, 2^n]^d$: proceed to the next stage.
    return `Unknown`.
  end if
  Let $Q_{n,1} = F(X)$. — The set $Q_{n,i}$ will contain the point $f^i(x_0)$.
  Let $P_{n,1} = \left\{Q \in Q_{n,1} \mid Q \not\Subset \left|E_n^A\right|\right\}$. — The set $P_{n,1}$ will contain the point x_0.
  Let $O_{n,1} = Q_{n,1}$. — The set $O_{n,i}$ will contain the set $\{f(x_0), f^2(x_0), \ldots, f^i(x_0)\}$.
  if $Q_{n,1} \cap C_n \neq \emptyset$ **then** — If F_n cannot be applied to $Q_{n,1}$: proceed to the next stage.
    return `Unknown`.
  end if
  for $i = 1, 2, \ldots$ **do**
    Compute $Q_{n,i+1} = F(Q_{n,i})$. — Compute an inclusion of the next point on the orbit.
    Compute $P_{n,i+1} = \left\{Q \in F(P_{n,i}) \mid Q \not\Subset \left|E_n^A\right|\right\}$. — The set $P_{n,i+1}$ contains $f(P_{n,i}) \cap A$.
    Compute $O_{n,i+1} = O_{n,i} \cup Q_{n,i+1}$. — Compute the next inclusion of the whole orbit.
    if $P_{n,i+1} = \emptyset$ **then** — If $P_{n,i}$ is empty: the point must have escaped.
      return `Escapes`.
    else if $Q_{n,i+1} \cap C_n \neq \emptyset$ **then** — If F_n cannot be applied to $Q_{n,i}$: proceed to the next stage.
      return `Unknown`.
    else if $O_{n,i} = O_{n,i+1}$ **then** — If the orbit-inclusion is invariant under F_n:
      if $O_{n,i} \Subset I_n^A$ **then** — If the orbit is certainly contained in A: the instance must be trapped.
        return `Trapped`.
      else — Otherwise, no further progress can be made: proceed to the next stage.
        return `Unknown`.
      end if
    end if
  end for
end procedure

Algorithm 1 above takes as input a name $(F_n)_n$ of a continuous function $f\colon \mathbb{R}^d \to \mathbb{R}^d$, a name $(I_n^A, E_n^A)_n$ of a closed set $A \subseteq \mathbb{R}^d$, and a name $(X_n)_n$ of a point $x_0 \in A$. It either runs forever or halts within finitely many steps. Upon halting, it reports whether there exists $n \in \mathbb{N}$ such that $f^n(x_0) \notin A$. The algorithm proceeds in stages, working on a subdivision of $[-2^n, 2^n]^d$ into cubes of side-length 2^{-n} in each stage. The stages are described in Algorithm 2. At each stage, Algorithm 2 computes an overapproximation of the orbit of x_0 to determine whether x_0 escapes. At the same time, it searches for an invariant set under F_n which is contained in A°, yielding an invariant set under f which is contained in A°.

It is straightforward to verify that Algorithm 1 is correct, *i.e.*, that it produces the correct answer upon halting.

▶ **Proposition 10.** *Let $A \subseteq \mathbb{R}^d$ be a closed set, let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function, and let $x_0 \in A$ be a point. Let $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$ be names of A, f, and x_0 respectively. Assume that Algorithm 1 is given the inputs $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$.*

If the algorithm eventually halts, it correctly reports whether x_0 is trapped or whether it escapes.

Proof. See [26, Proposition 10]. ◀

3.1 Completeness

We will now show that our algorithm is *complete*, *i.e.* that it halts on all problem instances for which the answer is robust under small perturbations.

It is easy to see that the algorithm halts on all problem instances where the point escapes. We will show that if the point is trapped, termination of the algorithm is guaranteed by the existence of a *robust invariant*. Conversely, the absence of a robust invariant entails that the point escapes under arbitrarily small perturbations of the function.

▶ **Definition 11.** *Let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function. A* robust invariant *for f is a compact set $V \subseteq \mathbb{R}^d$ such that $f(V) \subseteq V^\circ$.*

We first observe the following sufficient conditions for termination:

▶ **Proposition 12.** *Let $A \subseteq \mathbb{R}^d$ be a closed set, let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function, and let $x_0 \in A$ be a point. Let $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$ be names of A, f, and x_0 respectively. Assume that Algorithm 1 is given the inputs $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$.*

1. *If x_0 escapes A under f, then the algorithm eventually halts.*
2. *If $f(x_0)$ is contained in the interior of a robust invariant V for f with $V \subseteq A^\circ$, then the algorithm eventually halts.*

Proof. See [26, Proposition 12]. ◀

The core of the completeness proof is the following

▶ **Lemma 13.** *Let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous map. Let $K \subseteq \mathbb{R}^d$ be a compact set. Let $x_0 \in K^\circ$.*

Assume that x_0 is not contained in the interior of any robust invariant of f in K. Then for all $\varepsilon > 0$ there exists a map $\widetilde{f}\colon \mathbb{R}^d \to \mathbb{R}^d$ with $\left|\left|\widetilde{f} - f\right|\right|_\infty < \varepsilon$ such that x_0 escapes K under $\widetilde{f}$.

Proof Sketch. We will only sketch the proof idea. The full proof is given in [26, Appendix B].

Let $\varepsilon > 0$. Consider the set $E_0 = \mathbb{R}^d \setminus K^\circ$. Since K is compact and f is continuous, there exists $0 < \delta < \varepsilon/4$ such that $x_0 \notin \overline{B}(E_0, \delta)$ and such that $x, y \in K$, $d(x, y) \leq \delta$ implies $d(f(x), f(y)) < \varepsilon/4$.

Consider the compact set $K_0 = K \setminus B(E_0, \delta)$. Then by construction, $K_0 \Subset K$ and x_0 is contained in the interior of the complement of K_0. By assumption, K_0 cannot be a robust invariant of f. Hence, the set $E_1 = \left\{x \in K \setminus B(E_0, \delta) \mid f(x) \in \overline{B}(E_0, \delta)\right\}$ must be non-empty. If $x_0 \notin \overline{B}(E_1, \delta)$, we can repeat the same argument, applied to $K_1 = K \setminus (B(E_0, \delta) \cup B(E_1, \delta))$ to obtain the non-empty closed set $E_2 = \left\{x \in K \setminus (B(E_0, \delta) \cup B(E_1, \delta)) \mid f(x) \in \overline{B}(E_1, \delta)\right\}$.

Proceeding by induction, we obtain a sequence of non-empty closed subsets $E_1, E_2, \dots$ of K such that E_i is disjoint from $\bigcup_{j=0}^{i-1} B(E_j, \delta)$ and for $x \in K \setminus \bigcup_{j=0}^{i} B(E_j, \delta)$ we have $f(x) \in \overline{B}(E_i, \delta)$ if and only if $x \in E_{i+1}$. A straightforward compactness argument shows that the sequence must be finite, so that there exist $N \geq 1$ with $x_0 \in \overline{B}(E_N, \delta)$ and $x_0 \notin \overline{B}(E_i, \delta)$ for $i < N$.

Now, since x_0 is δ-close to an element of E_N, which gets mapped δ-close to some element of E_{N-1} under f, we can modify f on a small neighbourhood of x_0 such that x_0 gets mapped to a point $x_1 \in E_{N-1}$. We then proceed to modify f on a small neighbourhood of x_1 to ensure that x_1 maps to a point $x_2 \in E_{N-2}$. By induction, we obtain a sequence of points $x_1, \dots, x_N$ and a perturbation $\widetilde{f}$ of f by at most ε, such that $\widetilde{f}(x_i) = x_{i+1}$ and $x_N \notin K$. ◀

Lemma 13 applies only to compact sets. It admits the following extension to arbitrary closed sets:

▶ **Lemma 14.** *Let $f\colon \mathbb{R}^d \to \mathbb{R}^d$, let $A \subseteq \mathbb{R}^d$ be a closed set, and let $x_0 \in A$.*

If there is no robust invariant V for f with $f(x_0) \in V^\circ \subseteq V \subseteq A^\circ$, then there exist sequences $(f_n)_n$ and $(A_n)_n$ with $f_n \to f$ and $A_n \to A$ such that $x_0 \in A$ and x_0 escapes A_n under f_n.

Further, if $A \neq \mathbb{R}^d$, then we may take $A_n = A$ for all n.

Proof. See [26, Lemma 14]. ◀

Our main theorem now follows immediately from Proposition 10, Proposition 12, and Lemma 14:

▶ **Theorem 15.** *Let $A \subseteq \mathbb{R}^d$ be a closed set, let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function, and let $x_0 \in A$ be a point. Let $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$ be names of A, f, and x respectively. Assume that Algorithm 1 is given the inputs $(I_n^A, E_n^A)_n$, $(F_n)_n$, and $(X_n)_n$.*

1. *If the algorithm eventually halts, it correctly reports whether x_0 is trapped or whether it escapes.*
2. *If x_0 escapes A under f, then the algorithm eventually halts.*
3. *If $f(x_0)$ is contained in the interior of a robust invariant V of f with $I \subseteq A^\circ$, then the algorithm eventually halts.*
4. *If the algorithm does not halt, then there exist sequences $(f_{0,n})_n$, $(f_{1,n})_n$, $(A_n)_n$ with $f_{i,n} \to f_n$ for $i = 0, 1$ and $A_n \to A$, such that x_0 escapes $A_{0,n}$ under $f_{0,n}$ and is trapped in $A_{1,n}$ under $f_{1,n}$ for all n. Moreover, if $A \neq \mathbb{R}^d$, then we can choose $A_{0,n} = A_{1,n} = A$ for all n.*

In particular, Algorithm 1 is a complete decision method for the Point Escape Problem.

Theorem 15 establishes more than just the completeness of the algorithm: if A is not equal to all of $\mathbb{R}^d$ and the algorithm fails to halt on a problem instance (f, A, x_0), then the answer to the problem is unstable under small perturbations of the function f alone. This implies that Algorithm 1 yields a complete decision method for the Special Point Escape Problem where A and x_0 are fixed, and only f is given as an input. Of course, this result does not extend to the case where $A = \mathbb{R}^d$ since every point is robustly trapped in $\mathbb{R}^d$, but not every function has a robust invariant.

3.2 Generic Termination

A complete algorithm for a decision problem need not be able to solve the problem on a "large" set of instances. For example, the set $\mathbb{Q} \subseteq \mathbb{R}$ is maximally partially decided by the algorithm that never halts. Given a complete algorithm, one is thus led to the problem of characterising the "size" of its halting set.

All escaping instances of the Point Escape Problem are robustly escaping – in particular, Algorithm 1 halts on a dense set of inputs – which means that for every input (f, A, x_0) there exists a sequence of inputs $(f_n, A_n, x_{0,n})$ with $f_n \to f$, $A_n \to A$, $x_{0,n} \to x_0$ such that the algorithm halts on input $(f_n, A_n, x_{0,n})$ for all n. Convergence is with respect to the final topology induced by the representation. It is of course not the case that all trapped instances are robustly trapped. However, we will show that the robustly trapped instances are typical among the trapped ones: every boundary instance (*i.e.* every trapped instance that is not robustly trapped) can be perturbed into a robustly trapped instance under arbitrarily small perturbations. In this sense, both the robustly escaping instances and the robustly trapped instances constitute "large" sets.

▶ **Lemma 16.** *Let (f, K, x_0) be a boundary instance of the Point Escape Problem, where K is a compact set. Then there exist sequences $(f_n)_n$ and $(K_n)_n$ with $f_n \to f$, $K_n \to K$ such that (f_n, K_n, x_0) is a robust trapped instance of the Point Escape Problem for all n.*

Proof Sketch. Since Algorithm 1 halts on all escaping instances, the instance (f, K, x_0) must be trapped.

Let $\varepsilon > 0$. Consider the compact set $K_\varepsilon = \bigcup_{y \in K} \overline{B}(y, \varepsilon)$. Then the orbit $\{f^n(x_0) \mid n \in \mathbb{N}\}$ is contained in the interior of K_ε.

By compactness, K_ε admits a cover of the form $\bigcup_{i=0}^{N} B(y_i, \varepsilon)$ with $y_i \in K_\varepsilon$. The orbit $\{f^n(x_0) \mid n \in \mathbb{N}\}$ is contained in K_ε, so that, by the pigeonhole principle, there exist minimal $j < k \leq N+1$ such that $f^j(x_0)$ and $f^k(x_0)$ are contained in the same ball $B(y_i, \varepsilon)$. The points $f^j(x_0), f^{j+1}(x_0), \ldots, f^{k-1}(x_0)$ are uniformly bounded away from the boundary of K_ε by some $\delta > 0$. We may further assume that the balls $\overline{B}(f^i(x_0), \delta)$ are disjoint for $i \in \{j, \ldots, k\}$ and that δ is so small that $d(x, y) < \delta$ implies $d(f(x), f(y)) < \varepsilon$. For $j \leq i < k-1$ we replace f on $B(f^i(x_0), \delta/2)$ by the constant function with value $f^{i+1}(x_0)$. On $B(f^{k-1}(x_0), \delta/2)$, we replace f by the constant function with value $f^j(x_0)$. To make the function continuous again, we use the annuli $B(f^i(x_0), \delta) \setminus B(f^i(x_0), \delta/2)$ to connect f with the modified function via linear interpolation. Then the set $\bigcup_{i=j}^{k-1} \overline{B}(f^i(x_0), \delta/2)$ is a robust invariant for the perturbed function which is contained in the interior of K_ε and contains $f^j(x_0)$ in its interior. From this, we easily obtain a robust invariant in the interior of K_ε that contains x_0 in its interior.

The full proof is given in [26, Appendix C]. ◀

▶ **Corollary 17.** *Let (f, A, x_0) be a boundary instance of the Point Escape Problem. Then there exist sequences $(f_n)_n$ and $(A_n)_n$ with $f_n \to f$, $A_n \to A$ such that (f_n, A_n, x_0) is a robust trapped instance of the Point Escape Problem for all n.*

Proof. The instance (f, A, x_0) must be a trapped instance of the Point Escape Problem. We show that there exists a sequence of functions $(f_n)_n$ and a sequence of compact sets $(K_n)_n$ with $f_n \to f$ and $K_n \to A$ such that (f_n, K_n, x_0) is a (not necessarily robust) trapped instance of the Point Escape Problem. Together with Lemma 16 this yields the claim.

We distinguish two cases: If the orbit of x_0 under f is bounded, say, contained in the ball $\overline{B}(0, N)$ for some N, we let $f_n = f$ and $K_n = \overline{B}(0, N + 2^n)$ for all n. It is clear that x_0 is trapped in K_n under f for all n. It is easy to see that $K_n \to A$ in the topology induced by our representation of closed sets.

Now assume that the orbit of x_0 under f is unbounded. Fix some $N > ||x||_\infty$. Let $x_j = f^{(j)}(x_0)$. Let $n \in \mathbb{N}$. Let m be the smallest index such that $x_0, \ldots, x_m$ are contained in $\overline{B}(0, N + 2^n)$, but $x_{m+1} \notin \overline{B}(0, N + 2^n)$. Let $\delta = ||x_{m+1}||_\infty - N - 2^n > 0$. Let $K_n = A \cap \overline{B}(0, N + 2^n + \delta)$. Let

$$f_n(x) = \begin{cases} f(x) & \text{if } x \in \overline{B}(0, N+2^n), \\ \frac{||x||_\infty - N - 2^n}{\delta} x_0 + \left(1 - \frac{||x||_\infty - N - 2^n}{\delta}\right) f(x) & \text{if } x \in \overline{B}(0, N+2^n+\delta) \setminus \overline{B}(0, N+2^n), \\ x_0 & \text{if } x \in \mathbb{R}^d \setminus \overline{B}(0, N+2^n+\delta). \end{cases}$$

Observe that by construction we have $f_n(x_{m+1}) = x_0$, so that the orbit of x_0 under f_n is equal to $\{x_0, \dots, x_{m+1}\} \subseteq K_n$. It is easy to see (for example by using Proposition 4) that $f_n \to f$ in the topology induced by our representation of functions and that $K_n \to K$ in the topology induced by our representation of closed sets. ◀

4 Linear Systems

As mentioned in the introduction, Algorithm 1 may fail to terminate on very simple-looking problem instances when A is not a compact set. The main reason for this is that our representation of continuous functions induces the topology of uniform convergence on compact sets, or equivalently, the topology induced by the metric (1). In order for two functions $f, g\colon \mathbb{R}^d \to \mathbb{R}^d$ to have distance at most 2^{-n}, it suffices that f and g agree on the cube $[-2^{n+1}, 2^{n+1}]^d$, potentially admitting function values of arbitrarily large distance outside of this cube. Thus, every function f has "arbitrarily small" perturbations that differ from f arbitrarily outside a certain bounded set.

Revisiting the example from the introduction, consider the set $A = \{x \in \mathbb{R} \mid x \geq 0\}$, the initial point $x_0 = 1 \in A$, and the map $f\colon \mathbb{R} \to \mathbb{R}$, $f(x) = 2x$. It is clear that the point x_0 is trapped in A under iteration of f. However, it cannot be robustly trapped, since f does not admit any robust invariants. More explicitly, with respect to the metric (1), f is the limit of the sequence

$$f_n(x) = \min\left\{2x, \left(1 - 2^{n+1}\right)x + \left(1 + 2^{n+1}\right)(2^n - 1)\right\}.$$

We have $f_n^n(1) = 2^n$ and $f_n(2^n) = -1$, so that x_0 escapes A under each f_n.

While a naïve direct application of Algorithm 1 fails to verify that the system is trapped, we can exploit the fact that the particular f and A given above are well behaved "near infinity" to compute a compactified version of the system that is amenable to analysis via Algorithm 1.

Consider the map $\Phi\colon \mathbb{R} \to (-1,1)$, $\Phi(x) = \frac{x}{1+|x|}$. This map has a continuous inverse, explicitly given by $\Phi^{-1}\colon (-1,1) \to \mathbb{R}$, $\Phi^{-1}(x) = \frac{x}{1-|x|}$. We have $\Phi \circ f \circ \Phi^{-1}(x) = \frac{2x}{1+|x|}$. The expression on the right-hand side of this equality defines a total function $\overline{f}(x)\colon \mathbb{R} \to \mathbb{R}$. Observe that a point x is trapped in A under f if and only if $\Phi(x)$ is trapped in A under $\overline{f}$. The latter can be verified by Algorithm 1 for all $x > 0$, since $\overline{f}$ admits the robust invariant $\left[\frac{\Phi(x)}{2}, 2\right]$ with $\Phi(x) \in \left(\frac{\Phi(x)}{2}, 2\right) \subseteq \left[\frac{\Phi(x)}{2}, 2\right] \subseteq A^\circ$.

To further illustrate how the scope of Algorithm 1 can be extended using this compactification technique, we generalise the above observation to give a solution for the problem of deciding whether a point escapes a polyhedron (which is in general unbounded) under the iteration of an affine linear map:

▶ **Definition 18.** *The* Linear Escape Problem *asks to determine for a given non-singular matrix $A \in \mathbb{R}^{d \times d}$, a given vector $b \in \mathbb{R}^d$, a given polyhedron $P \subseteq \mathbb{R}^d$, and a given point $x_0 \in \mathbb{R}^d$ whether x_0 escapes P under the map $f_{A,b}(x) = Ax + b$.*

Here, A is given by a $\rho^{d \times d}$-name of its entries, and P is given by a finite list of affine half-spaces $H_1, \dots, H_m$ with $P = \bigcap_{i=1}^m H_i$, where each H_i is given by a ρ^d-name of a normal $0 \neq N_i \in \mathbb{R}^d$ and a ρ^1-name of a distance $D_i \in \mathbb{R}$, such that $H_i = \left\{x \in \mathbb{R}^d \mid N_i x \leq D_i\right\}$.

We will give a reduction of the Linear Escape Problem to the Point Escape Problem that sends robust instances to robust instances based on the compactification idea discussed above.

Before we describe the reduction, we classify the robust instances of the Linear Escape Problem. Since all escaping instances are robust, we focus only on trapped instances.

For a matrix A, let $\sigma(A) \subseteq_{\text{fin}} \mathbb{C}$ denote the set of its eigenvalues and $r(A) = \max\{|\lambda| \mid \lambda \in \sigma(A)\}$ denote the largest value among the absolute values of its eigenvalues.

For a polyhedron $P = \bigcap_{i=1}^{m} H_i$ with $H_i = \{x \in \mathbb{R}^d \mid N_i x \leq D\}$, we let $P_0 = \bigcap_{i=1}^{m} H_i^0$, where $H_i^0 = \{x \in \mathbb{R}^d \mid N_i x \leq 0\}$.

▶ **Lemma 19.** *Let (A, b, P, x_0) be a trapped instance of the Linear Escape Problem. Then (A, b, P, x_0) is robust if and only if the orbit $\left(f_{A,b}^n(x_0)\right)_n$ is contained in the interior of P and one of the three following conditions is met:*

1. *$r(A) < 1$ and the unique fixed point of the map $f_{A,b}(x) = Ax + b$ is contained in P°.*
2. *A has a simple real eigenvalue $\rho > 1$ with $\rho > |\lambda|$ for all other eigenvalues and there exists an eigenvector v for ρ with $v \in P_0^\circ$ and $x_0 = \alpha v + w$ where $\alpha > 0$ and w is a linear combination of eigenvalues and generalised eigenvalues for eigenvalues $\lambda \neq \rho$ (in the sense of the real Jordan normal form).*
3. *A has the simple real eigenvalue $\rho = 1$ with $\rho > |\lambda|$ for all other eigenvalues, there exists an eigenvector v for ρ with $v \in P_0^\circ$ such that $x_0 = \alpha v + w$, $b = \beta v + u$, where $\alpha, \beta > 0$, and w and u are linear combinations of eigenvalues and generalised eigenvalues for eigenvalues $\lambda \neq \rho$ (in the sense of the real Jordan normal form).*

Proof. The proof is very similar to that of [25, Proposition 5.1, Proposition 6.2.]. It is given in [26, Appendix D]. ◀

We now describe the reduction to the Point Escape Problem.

Consider the continuous map $\Phi\colon \mathbb{R}^d \to B(0,1)$, $\Phi(x) = \frac{x}{1+||x||}$. This map is invertible with inverse $\Phi^{-1}\colon B(0,1) \to \mathbb{R}^d$, $\Phi^{-1}(x) = \frac{x}{1-||x||}$. The map $\Phi \circ f_{A,b} \circ \Phi^{-1}\colon B(0,1) \to B(0,1)$ extends (uniformly computably) to the continuous map

$$\overline{f}_{A,b}\colon \mathbb{R}^d \to \mathbb{R}^d,\ \overline{f}_{A,b}(x) = \frac{Ax + (1 - \min\{1, ||x||\})\, b}{||Ax + (1 - \min\{1, ||x||\})\, b|| + 1 - \min\{1, ||x||\}}.$$

For a polyhedron $P = \bigcap_{i=1}^{m} H_i$ with $H_i = \{x \in \mathbb{R}^d \mid Nx \leq D\}$, we define $\widehat{P} = \bigcap_{i=1}^{m} \widehat{H}_i$, where $\widehat{H_i} = \{x \in \mathbb{R}^d \mid Nx \leq D\,(1 - \min\{1, ||x||\})\}$. Observe that $\Phi(P) = \widehat{P} \cap B(0,1)$.

▶ **Proposition 20.** *The point x_0 is trapped in P under $f_{A,b}$ if and only if the point $\frac{x_0}{1+||x_0||}$ is trapped in $\widehat{P}$ under $\overline{f}_{A,b}$.*

Proof. Since Φ is bijective we have $x \in P$ if and only if $\Phi(x) \in \Phi(P) = \widehat{P} \cap B(0,1)$. On $B(0,1)$, the map $\overline{f}_{A,b}$ agrees with $\Phi \circ f_{A,b} \circ \Phi^{-1}$, which maps $B(0,1)$ to $B(0,1)$. It follows that we have $\overline{f}_{A,b}^n\,(\Phi\,(x_0)) = \Phi \circ f_{A,b}^n(x_0)$, which yields the claim. ◀

We now obtain our desired reduction:

▶ **Proposition 21.** *The instance (A, b, P, x_0) is a robust instance of the Linear Point Escape Problem if and only if the instance $(\overline{f}_{A,b}, \widehat{P}, \Phi(x_0))$ is a robust instance of the Point Escape Problem.*

Proof. If the instance escapes, the claim is obvious. Thus, assume that the instance is trapped. We show that $\overline{f}_{A,b}$ has a robust invariant in the interior of $\widehat{P}$.

Consider the classification of robust instances given in Lemma 19.

In the first case, it follows from the proof of Lemma 19 that the point $p = \Phi((I - A)^{-1}b)$ is a fixed point of $\overline{f}_{A,b}$ such that every point in $B(0,1)$ converges to $\Phi((I - A)^{-1}b)$ under the iteration of $\overline{f}_{A,b}$. Moreover, for a sufficiently small $\varepsilon > 0$ and sufficiently large N, the ball $\overline{B}(p, \varepsilon)$ is mapped by $\overline{f}^N_{A,b}$ inside the ball $B(p, \varepsilon/2)$.

In the second and third case, it follows from the proof of Lemma 19 that the point $p = \frac{v}{||v||}$ is a fixed point of $\overline{f}_{A,b}$ with $\overline{f}^n_{A,b}(\Phi(x_0)) \to p$ as $n \to \infty$. Moreover, for a sufficiently small $\varepsilon > 0$ and sufficiently large N, the ball $\overline{B}(p, \varepsilon)$ is mapped by $\overline{f}^N_{A,b}$ inside the ball $B(p, \varepsilon/2)$.

In either case, for sufficiently small $\varepsilon > 0$, the ball $\overline{B}(p, \varepsilon)$ is a robust invariant of $\overline{f}^N_{A,b}$ for some N which is contained in the interior of $\widehat{P}$, such that all $x \in \overline{B}(p, \varepsilon)$ remain in the interior of $\widehat{P}$ under iteration of $\overline{f}_{A,b}$, $\overline{f}^k_{A,b}(x_0) \in B(p, \varepsilon)$ for some k, and $\overline{f}^j_{A,b}(x_0) \in \widehat{P}^\circ$ for all $j \leq k$.

It now follows from [26, Proposition 31, Proposition 32] that $\overline{f}_{A,b}(x_0)$ is contained in the interior of a robust invariant of $\overline{f}_{A,b}$ which is contained in the interior of $\widehat{P}$. Theorem 15 yields that the instance is robust. ◀

5 The Quadratic Family

As a final application of Algorithm 1, we provide a complete decision method for the Mandelbrot set subject to the hyperbolicity conjecture. The *Mandelbrot set* $\mathcal{M}$ is the set of all parameters $c \in \mathbb{C}$ such that the orbit of the origin $0 \in \mathbb{C}$ under the map $f_c(z) = z^2 + c$ is bounded. The Mandelbrot set is a compact connected subset of $\mathbb{C}$. Despite $\mathcal{M}$ being best known for its numerous computer-generated depictions [33], it is unknown whether there exists any rigorous algorithm for computing accurate images of $\mathcal{M}$ to a given resolution. See [13] for a good discussion of this. This has lead Penrose [34] to go as far as conjecturing that the Mandelbrot set might be an uncomputable subset of $\mathbb{C}$, however without specifying a computational model. Shortly after Penrose made his conjecture, Blum and Smale [8] showed that the Mandelbrot set is undecidable in the BSS-model of real computation [7]. However, Brattka [10] soon raised the objection that BSS-decidability does not adequately capture the intuitive notion of computability that Penrose had in mind: In fact, Penrose explicitly mentions the closed epigraph of the exponential function as an example of an intuitively computable subset of the complex plane. However, all BSS-decidable subsets of $\mathbb{C}$ must be semi-algebraic, so that the epigraph of the exponential function is BSS-undecidable. Computable analysis offers an alternative definition of computability: a subset $X \subseteq \mathbb{C}$ of the complex plane is called computable if its *distance function* $d_X \colon \mathbb{C} \to \mathbb{R}$, $d_X(c) = \inf \{|c - z| \mid z \in X\}$ is computable. It is quite easy to see that the distance function of a compact set is computable if and only if the set can be plotted at any given resolution – see for example [13] for details.

Hertling [17] showed that $\mathcal{M}$ is computable in this sense subject to the *hyperbolicity conjecture*, which we describe below. He even showed that under this conjecture the *signed distance function* $d^s_{\mathcal{M}}(c) = (-1)^{\chi_{\mathcal{M}}(c)} d_{\mathcal{M}}(c)$ is computable. This is stronger than computability, as there exist sets whose distance function is computable but whose signed distance function is not. It follows from Hertling's work that the signed distance function of $\mathcal{M}$ is computable if and only if $\mathcal{M}$ is maximally partially decidable.

The problem of maximally partially deciding the Mandelbrot set reduces to the Point Escape Problem as follows: map $c = a + ib \in \mathbb{C}$ to the Point Escape Problem instance $\left(g_c, \overline{B}(0,3), 0\right)$, where $g_c(x,y) = \left(x^2 - y^2 + a, 2xy + b\right)$. Observe that $g_c(x,y) = (\mathrm{Re}\left(f_c\left(x+iy\right)\right), \mathrm{Im}\left(f_c\left(x+iy\right)\right))$.

We will discuss how much of the Mandelbrot set our algorithm for the Point Escape Problem is able to compute under this reduction. To state our main result we require one more definition. A point $c \in \mathcal{M}^\circ$ in the interior of $\mathcal{M}$ is called *hyperbolic* if f_c has an *attracting cycle*, *i.e.* there exist $z_0 \in \mathbb{C}$ and $n \in \mathbb{N}$ with $f_c^n(z_0) = z_0$, $f_c^j(z_0) \neq f_c^k(z_0)$ for $j, k < n$, and $\left|\frac{d}{dz}(f_c^n)(z_0)\right| < 1$.

The *hyperbolicity conjecture* states that every point in the interior of $\mathcal{M}$ is hyperbolic. We obtain the following result:

▶ **Theorem 22.** *The decision problem for the Mandelbrot set reduces to the General Escape Problem via the reduction above. This reduction maps a parameter $c \in \mathbb{C}$ to a robust instance of the General Escape Problem if and only if c belongs to $\mathbb{C} \setminus \mathcal{M}$ or to a hyperbolic component of the interior of $\mathcal{M}$. In particular, the reduction maps robust instances to robust instances if and only if the hyperbolicity conjecture holds true.*

Proof. Consider the map $c \mapsto \left(g_c, \overline{B}(0,3), 0\right)$ defined above. It is easy to see that this is really a reduction. It is obvious that – under this reduction – Algorithm 1 halts on all parameters outside $\mathcal{M}$ and that Algorithm 1 does not halt on any boundary point.

Now, let $c \in \mathcal{M}^\circ$. Assume first that c is hyperbolic. Then c has an attracting cycle $\{z_0, z_1, \ldots, z_{n-1}\}$ as described above. A celebrated theorem by Fatou [9, Theorem 1] asserts that the origin is attracted by this attracting cycle, *i.e.* for all ε there exists N such that $\left|f_c^N(0) - z_0\right| < \varepsilon$.

We show that the attracting cycle induces a robust invariant. Since $|\frac{d}{dz}(f_c^n)(z_0)| < 1$, there exists $\delta > 0$ such that $|\frac{d}{dz}(f_c^n)(z_0)| < 1 - \delta$. Further, observe that by the chain rule we have $\frac{d}{dz}(f_c^n)(z_0) = \frac{d}{dz}(f_c^n)(f^k(z_0))$ for all k. By continuity, there exists a number $\varepsilon > 0$ such that $|\frac{d}{dz}(f_c^n)(z)| < 1 - \delta$ for all z with $|z - f^k(z_0)| < \varepsilon$ for some $k \in \mathbb{N}$. Consider the set $K = \bigcup_{j=0}^{n-1} \overline{B}(f_c^k(z_0), \varepsilon)$. It follows from the uniform bound on the first derivative that on K, the function f_c^n is Lipschitz-continuous with Lipschitz constant $1 - \delta$. Let $z \in K$ with $|z - f_c^k(z_0)| < \varepsilon$. Then we obtain:

$$\left|f_c^n\left(f_c(z)\right) - f_c^k(z_0)\right| = \left|f_c^n\left(f_c(z)\right) - f_c^n\left(f_c^k(z_0)\right)\right| \leq |1-\delta|\left|f_c^n(z) - f_c^k(z_0)\right| \leq |1-\delta|\,\varepsilon < \varepsilon.$$

So K is a robust invariant for f_c^n. Since 0 is attracted by the attracting cycle, we have $f^N(0) \in K^\circ$ for sufficiently large N.

It now follows from [26, Proposition 31, Proposition 32] that there exists a robust invariant for f_c whose interior contains 0. This invariant must be contained in the closed disk of radius 2, for any orbit of f_c that contains an element of absolute value greater than 2 must be unbounded. In particular, it must be contained in the open disk of radius 3. Hence, Algorithm 1 halts on input $(g_c, \overline{B}(0,3), 0)$.

Suppose on the other hand that c belongs to a non-hyperbolic component of the interior of $\mathcal{M}$. Let $K_c = \{z \in \mathbb{C} \mid f_c^n(z) \text{ is bounded}\}$ denote the *filled-in Julia set* of c. It can be shown (see [15]) that since c belongs to a non-hyperbolic component of the interior of $\mathcal{M}$, we must have $c \in K_c = \partial K_c$. This implies that for all $\varepsilon > 0$ there exists z_ε with $|z_\varepsilon - c| < \varepsilon$ and $z_\varepsilon \notin K_c$. Since the orbit $f_c^n(z_\varepsilon)$ is unbounded and disjoint from K_c with $0 \in K_c$, there exists a number $\delta > 0$ with $|f_c^n(z_\varepsilon)| > \delta$. Thus, if we define $f_\varepsilon(z)$ to be equal to $f_c(z)$ if $|z| \geq \delta$ and to otherwise be defined by $f_\varepsilon(z) = z_\varepsilon\left(1 - \frac{|z|}{\delta}\right) + f_c(z)\frac{|z|}{\delta}$, then the orbit of 0 under f_ε is unbounded, and $||f_\varepsilon - f_c||_\infty < \varepsilon$. This shows that Algorithm 1 cannot halt on input $\left(g_c, \overline{B}(0,3), 0\right)$. ◀

6 Conclusion and Future Work

We have studied the problem of deciding for a given set, initial point, and function whether the initial point escapes the set under iteration of the function. We have allowed arbitrary continuous functions as inputs, represented by the weakest representation that makes function evaluation uniformly computable. We have given an algorithm that certifies that the initial point is trapped by searching for a robust invariant. We have shown that this algorithm is complete in the sense that no sound algorithm can detect further trapped instances. This settles the very natural question of how much is decidable about the escape problem when we are given just enough information about our system to compute the orbit of a given point. In general, computable analysis allows us to ask and answer questions of this form: if we are given a certain (limited) amount of information about a system, how much can we say about the system's behaviour? This leads to various natural directions for future work: On the one hand, it would be very interesting to study more general systems, for example by replacing $\mathbb{R}^d$ with a locally compact metric space X. On the other hand, it would be interesting to study more special systems which are presented with additional information, such as polynomials or other subclasses of analytic maps which are "well-behaved at infinity" such as Pfaffian functions. We have provided partial evidence that, somewhat surprisingly, our algorithm for general systems yields maximal partial algorithms for rather rigid special systems, such as linear systems and – conjecturally – one-dimensional complex quadratic systems. It would be very interesting to determine whether this reduction extends, for example, to higher-degree multivariate polynomials.

References

1. Manindra Agrawal, S. Akshay, Blaise Genest, and P.S. Thiagarajan. Approximate Verification of the Symbolic Dynamics of Markov Chains. In *2012 27th Annual IEEE Symposium on Logic in Computer Science*, pages 55–64, 2012. doi:10.1109/LICS.2012.17.
2. S. Akshay, Hugo Bazille, Blaise Genest, and Mihir Vahanwala. On Robustness for the Skolem and Positivity Problems. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPIcs*, pages 5:1–5:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.STACS.2022.5.
3. Christel Baier, Florian Funke, Simon Jantsch, Toghrul Karimov, Engel Lefaucheux, Joël Ouaknine, Amaury Pouly, David Purser, and Markus A. Whiteland. Reachability in Dynamical Systems with Rounding. In Nitin Saxena and Sunil Simon, editors, *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2020, December 14-18, 2020, BITS Pilani, K K Birla Goa Campus, Goa, India (Virtual Conference)*, volume 182 of *LIPIcs*, pages 36:1–36:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.FSTTCS.2020.36.
4. William J. Baumol. *Economic Dynamics*. Prentice Hall, 1970.
5. Jason P. Bell and Stefan Gerhold. On the positivity set of a linear recurrence. *Israel Journal of Mathematics*, 157:333–345, 2007. doi:10.1007/s11856-006-0015-1.
6. Vincent D. Blondel, Emmanuel Jeandel, Pascal Koiran, and Natacha Portier. Decidable and Undecidable Problems about Quantum Automata. *SIAM Journal on Computing*, 34(6):1464–1473, 2005. doi:10.1137/S0097539703425861.
7. Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation*. Springer, 1998.
8. Lenore Blum and Steve Smale. The Gödel Incompleteness Theorem and Decidability over a Ring. In Morris W. Hirsch, Jerrold E. Marsden, and Michael Shub, editors, *From Topology to*

Computation: Proceedings of the Smalefest, pages 321–339, New York, NY, 1993. Springer US. doi:10.1007/978-1-4612-2740-3_32.

9 Bodil Branner. The Mandelbrot Set. In Robert L. Devaney and Linda Keen, editors, *Chaos and Fractals: The Mathematics Behind the Computer Graphics*, pages 75–106. American Mathematical Society, 1989.

10 Vasco Brattka. The emperor's new recursiveness: The epigraph of the exponential function in two models of computability. In Masami Ito and Teruo Imaoka, editors, *Words, Languages & Combinatorics III*, pages 63–72, 2003.

11 Vasco Brattka and Peter Hertling, editors. *Handbook of Computability and Complexity in Analysis*, Theory and Applications of Computability, Cham, 2021. Springer. doi:10.1007/978-3-030-59234-9.

12 Vasco Brattka and Gero Presser. Computability on subsets of metric spaces. *Theoretical Computer Science*, 305(1 – 3):43–76, 2003. doi:10.1016/S0304-3975(02)00693-X.

13 Mark Braverman and Michael Yampolsky. *Computability of Julia Sets*. Springer, 2009.

14 Julian D'Costa, Toghrul Karimov, Rupak Majumdar, Joël Ouaknine, Mahmoud Salamati, Sadegh Soudjani, and James Worrell. The Pseudo-Skolem Problem is Decidable. In Filippo Bonchi and Simon J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, volume 202 of *LIPIcs*, pages 34:1–34:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.34.

15 Adrien Douady and John H. Hubbard. *Étude dynamique des polynômes complexes*. Publications mathématiques d'Orsay, 1984.

16 Stefan Gerhold and Manuel Kauers. A procedure for proving special function inequalities involving a discrete parameter. In Manuel Kauers, editor, *Symbolic and Algebraic Computation, International Symposium ISSAC 2005, Beijing, China, July 24-27, 2005, Proceedings*, pages 156–162. ACM, 2005. doi:10.1145/1073884.1073907.

17 Peter Hertling. Is the Mandelbrot set computable? *Mathematical Logic Quarterly*, 51(1):5–18, 2005. doi:10.1002/malq.200310124.

18 Mathieu Hoyrup and Cristóbal Rojas. On the Information Carried by Programs About the Objects they Compute. *Theor. Comp. Sys.*, 61(4):1214–1236, November 2017. doi:10.1007/s00224-016-9726-9.

19 Manuel Kauers and Veronika Pillwein. When can we detect that a P-finite sequence is positive? In Wolfram Koepf, editor, *Symbolic and Algebraic Computation, International Symposium, ISSAC 2010, Munich, Germany, July 25-28, 2010, Proceedings*, pages 195–201. ACM, 2010. doi:10.1145/1837934.1837974.

20 George Kenison. The Threshold Problem for Hypergeometric Sequences with Quadratic Parameters. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*, volume 297 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 145:1–145:20, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2024.145.

21 George Kenison, Oleksiy Klurman, Engel Lefaucheux, Florian Luca, Pieter Moree, Joël Ouaknine, Markus A. Whiteland, and James Worrell. On Positivity and Minimality for Second-Order Holonomic Sequences. *CoRR*, abs/2007.12282, 2020. arXiv:2007.12282.

22 Ker-I Ko. *Complexity theory of real functions*. Progress in Theoretical Computer Science. Birkhäuser Boston, Inc., Boston, MA, 1991. doi:10.1007/978-1-4684-6802-1.

23 Boris A. Kushner. Markov's constructive analysis; a participant's view. *Theoretical Computer Science*, 219(1):267–285, 1999. doi:10.1016/S0304-3975(98)00291-6.

24 Stephen Melczer and Marc Mezzarobba. Sequence positivity through numeric analytic continuation: uniqueness of the Canham model for biomembranes. *Comb. Theory*, 2(2):Paper No. 4, 20, 2022. doi:10.5070/C62257847.

25 Eike Neumann. Decision problems for linear recurrences involving arbitrary real numbers. *Logical Methods in Computer Science*, Volume 17, Issue 3, August 2021. doi:10.46298/lmcs-17(3:16)2021.

26 Eike Neumann. Deciding Robust Instances of an Escape Problem for Dynamical Systems in Euclidean Space, 2025. doi:10.48550/arXiv.2506.21481.

27 Eike Neumann, Joël Ouaknine, and James Worrell. Decision Problems for Second-Order Holonomic Recurrences. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 99:1–99:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.99.

28 Joël Ouaknine and James Worrell. On the Positivity Problem for Simple Linear Recurrence Sequences. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 318–329. Springer, 2014. doi:10.1007/978-3-662-43951-7_27.

29 Joël Ouaknine and James Worrell. Positivity Problems for Low-Order Linear Recurrence Sequences. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 366–379. SIAM, 2014. doi:10.1137/1.9781611973402.27.

30 Joël Ouaknine and James Worrell. Ultimate Positivity is Decidable for Simple Linear Recurrence Sequences. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 330–341. Springer, 2014. doi:10.1007/978-3-662-43951-7_28.

31 Joël Ouaknine and James Worrell. On linear recurrence sequences and loop termination. *ACM SIGLOG News*, 2(2):4–13, 2015. doi:10.1145/2766189.2766191.

32 Arno Pauly. On the topological aspects of the theory of represented spaces. *Computability*, 5(2):159–180, 2016. doi:10.3233/COM-150049.

33 Heinz-Otto Peitgen and Peter Richter. *The Beauty of Fractals*. Springer-Verlag, Heidelberg, 1986.

34 Roger Penrose. *The Emperor's New Mind*. Oxford University Press, 1989.

35 Veronika Pillwein. Positivity of certain sums over Jacobi kernel polynomials. *Advances in Applied Mathematics*, 41(3), 2008. doi:10.1016/J.AAM.2007.12.001.

36 Veronika Pillwein. Termination Conditions for Positivity Proving Procedures. In *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*, ISSAC '13, pages 315–322, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2465506.2465945.

37 Veronika Pillwein and Miriam Schussler. An efficient procedure deciding positivity for a class of holonomic sequences. *ACM Communications in Computer Algebra*, 49(3):90–93, 2015. Extended abstract of the poster presentation at ISSAC 2015.

38 Stefan Ratschan. Safety verification of non-linear hybrid systems is quasi-decidable. *Formal Methods in System Design*, 44(1):71–90, 2014. doi:10.1007/s10703-013-0196-2.

39 Matthias Schröder. *Admissible Representations for Continuous Computations*. PhD thesis, FernUniversität Hagen, 2002.

40 G. Szegö. Über gewisse Potenzreihen mit lauter positiven Koeffizienten. *Mathematische Zeitschrift*, 37:674–688, 1933. URL: http://eudml.org/doc/168480.

41 R. Tijdeman, M. Mignotte, and T.N. Shorey. The distance between terms of an algebraic recurrence sequence. *Journal für die reine und angewandte Mathematik*, 349:63–76, 1984. URL: http://eudml.org/doc/152622.

42 Ashish Tiwari. Termination of Linear Programs. In *CAV'04*, volume 3114 of *LNCS*. Springer, 2004. doi:10.1007/978-3-540-27813-9_6.

43 N.K. Vereshchagin. Occurrence of zero in a linear recursive sequence. *Mat. Zametki*, 38(2):177–189, 1985.

44 Klaus Weihrauch. *Computable Analysis*. Springer, 2000.

45 Thomas Yu and Jingmin Chen. Uniqueness of Clifford torus with prescribed isoperimetric ratio. *Proceedings of the American Mathematical Society*, 150, 2022. doi:10.1090/proc/15750.