Computability and dynamical systems

J. Buescu, D. S. Graça, and N. Zhong

Abstract In this paper we explore results that establish a link between dynamical systems and computability theory (not numerical analysis). In the last few decades, computers have increasingly been used as simulation tools for gaining insight into dynamical behavior. However, due to the presence of errors inherent in such numerical simulations, with few exceptions, computers have not been used for the nobler task of proving mathematical results. Nevertheless, there have been some recent developments in the latter direction. Here we introduce some of the ideas and techniques used so far, and suggest some lines of research for further work on this fascinating topic.

1 Introduction – From Numerics to Dynamics to Computation

In the last century significant developments have been made in the fields of dynamical systems and the theory of computation. Actually, the latter only appeared in the 1930s with the groundbreaking work of Turing, Church and others. These two areas have mostly evolved separately, with very sporadic interactions throughout most of the 20th century. However, with the advent of fast digital computers and their extensive use as simulation tools, this gap has been narrowing, and some work has been done to establish bridges across it. This paper focuses on this research.

J. Buescu
DM/FCUL, University of Lisbon, Portugal and CMAF, Lisbon, Portugal, e-mail: jbuescu@ptmat.fc.ul.pt

D. S. Graça
DM/Faculdade de Ciências e Tecnologia, Universidade do Algarve, Faro, Portugal & SQIG/Instituto de Telecomunicações, Lisbon, Portugal, e-mail: dgraca@ualg.pt

N. Zhong
DMS, University of Cincinnati, Cincinnati, OH 45221-0025, U.S.A., e-mail: ning.zhong@uc.edu

Dynamical systems theory is of interest to computer scientists for a number of reasons. We could point out that computers are used to control continuous processes in everyday life or that silicon is reaching its limits, and new paradigms of computation are now sought (e.g. quantum computation [19]), many of them involving dynamical systems.

However, in this paper we are interested in presenting what the theory of computation has to offer to the dynamical systems community.

The modern theory of dynamical systems began with Poincaré in the late 19th century, reached a high level of development in the Russian school by the middle of the 20th century, and was further developed by western mathematicians and scientists beginning in the 1960s. This development entailed the convergence of two very strong but quite distinct currents: a modeling (numerical) approach and an analytical approach.

On the modeling side, the increasing availability of computational power allowed the numerical study of mathematical models for systems of definite interest in problems of physics, engineering or mathematical sciences in general, showing that these low-dimensional deterministic systems apparently exhibited, in a persistent fashion, a strong form of chaotic behavior. The first and foremost example is of course that of the Lorenz attractor [36], whose display of sensitive dependence on initial conditions led Lorenz himself to coin the term "butterfly effect" to describe this form of chaos. It is far from the only one; soon other model systems were shown to exhibit the same kind of deterministic, low-dimensional chaotic behavior characterized by sensitive dependence on initial conditions. Thus, for instance, the Duffing equation [20], the (nonautonomous) van der Pol system [42] or the Rössler system [47] which arise as (differential) equations of motion for specific physical systems and also discrete time diffeomorphisms or maps, like the Hénon map or the logistic equation, which may be seen as arising directly or indirectly from a Poincaré section of the flow of a differential equation.

On the analytical side, hyperbolic dynamical systems theory began in the Russian school (especially in Anosov's work) and was further developed from the 1960s onward by the Smale school, with the purpose of giving a solid mathematical foundation to the fact that deterministic low-dimensional systems may exhibit persistent chaotic behavior, as evidenced by the wealth of specific examples referred to above. Thus arose the motivation for the main theoretical thrusts in what is nowadays called uniformly hyperbolic dynamical systems theory, leading from Anosov diffeomorphisms to the general theory of hyperbolic systems, whose invariant sets have the structure of a uniform invariant splitting into stable and unstable directions (see Smale [51]). This theory is extremely rich and allowed for the construction and study of very specific instances: the Arnold cat map, the Smale horseshoe and the corresponding symbolic dynamics derived from the associated Markov partitions.

Hyperbolic systems were conceived as an attempt to construct a rigorous theory describing persistent chaotic behavior. There were good grounds to believe that hyperbolic systems coupled with the dynamical equivalence relation of topological conjugacy (corresponding to structural stability) were the appropriate setting for a rigorous theory of chaotic phenomena, since this was the adequate generalization

of what was known for two-dimensional systems, namely Peixoto's theorem [41], which states that all planar vector fields have structurally stable perturbations, and one can thus disregard systems which are structurally unstable.

However, further research progressively revealed a vast gap between chaotic behavior as computationally observed in "strange attractors" and the dynamics of hyperbolic systems. Smale himself [50] delivered the first blow when he showed that, in dimension 3 or higher, structurally stable systems are not dense. Thus, even though achieving a complete characterization of hyperbolic systems and their properties was a major accomplishment in dynamical systems, hyperbolicity is too strong a property to characterize a generic set of differential equations or diffeomorphisms.

In particular, the strange attractors arising from the Lorenz system, the Hénon map, the Duffing equation and other computationally well-studied systems, although persistently chaotic, are not hyperbolic and thus fall outside the scope of hyperbolic theory. Indeed, it could have been the case that the Lorenz attractor, in spite of all the numerical studies, did not exist as a (persistent, structurally unstable, chaotic) strange attractor; hyperbolic dynamics simply does not provide an answer. The existence of the Lorenz attractor was, in fact, listed by Steven Smale as one of several challenging problems for the 21st century [52].

The way to bridge this gap, within the purely analytical approach, is to extend hyperbolic theory to more general systems. One way to achieve this goal is to allow for *partially hyperbolic* systems, where we require that the flow or map admits an invariant splitting but, instead of requiring uniform rates of expansion and contraction, we allow some directions to have mixed expansive, contractive or neutral behavior in different parts of the system. This approach originated in the works of Pugh-Shub and Mañé in the 1970s.

Yet another way to extend the theory is to use concepts from ergodic theory, where we drop the uniform hyperbolicity requirement and replace it by asymptotic expansion/contraction rates in directions which may depend measurably on the initial point. Such systems are referred to as *non-uniformly hyperbolic*, and the focus of the theory is to construct physical (SRB) invariant measures and more generally equilibrium states, and to study their ergodic properties. The equivalence relation corresponding to structural stability is known as *stochastic stability*.

From the computational point of view much work has also been done in order to bridge this gap. In this approach we need to construct rigorous theoretical methods which allow us to transcend conjectures suggested by more or less precise numerical experiments and prove mathematical results in the most rigorous sense of the term. Paradigmatic in this approach are breakthroughs such as Lanford's computer-assisted proof of the Feigenbaum conjectures [34] and, more recently, W. Tucker's proof that the Lorenz attractor exists [56].

In both cases rigorous computational methods went for beyond educated numerical experiments; they provided deep theoretical insights into the mathematical structure underlying the corresponding dynamical phenomena. In the first case, it substantiated the renormalization interpretation of universality in C^1-unimodal maps: it proved that there is a fixed point of a renormalization operator in a suitable map space with a one-dimensional unstable manifold, the corresponding eigenvalue be-

ing the Feigenbaum constant. In the second case, Tucker's work finally provided a proof of the long-standing conjecture that the dynamics of the ordinary differential equations of Lorenz is that of the geometric Lorenz attractor of Williams, Guckenheimer, and Yorke or, in short, that the Lorenz system does indeed contain a persistent strange attractor.

To develop such an approach one must in general leave the realm of plain numerical simulation and look for general statements on computability (in the sense of the theory of computation) of the objects and concepts of dynamical systems theory. Although this is a fairly recent field of research, some promising results have already been achieved. The purpose of this paper is to give an overview of the methods used and results obtained, as well as to point out directions for possible future research.

2 Computable Analysis

In the study of differential equations and dynamical systems, scientific computation is playing an ever larger role because most equations cannot be solved explicitly but only approximately by numerical methods. Thus it becomes of central importance to know whether or not the problem being solved is computable. In particular, if a solution is non-computable, then no numerical algorithm computing the solution can always provide approximations with arbitrarily desired precision.

Computability over discrete spaces has been well studied since the 1930s. Although there are several markedly different models which formalize the notion of computability, such as Turing machines, lambda calculus, recursive functions, etc., they all generate the same class of computable functions. This formal notion of computability and the Turing machine model have been accepted by the scientific community as the standard model of computation. Indeed, as the Church-Turing thesis asserts, any intuitively and reasonably computable function is computable by a Turing machine. We refer the reader to [49] for more details on basic results about the theory of computation.

The Turing machine, however, cannot be directly applied to compute real functions because it can only have as input and output a "finite number of bits." To circumvent this, several extensions of the Turing machine model have been proposed. One such extension is the BSS model [5], [4]. In the BSS model, a real number can be directly stored on a single cell, so that exact computations over real numbers can be carried out in finite time using infinite-precision arithmetic. Even though this model is algebraically elegant, it has certain weaknesses as a model for scientific computation. For example, the non-computability results obtained in this model do not correspond to computing practice in the real number setting (see [9] for more details), which is undesirable, since identifying non-computable parameters, functions and sets is one of the main objectives in the computability study of continuous structures [10], [44], and [59].

Another extension is the Type-2 Turing machine or oracle Turing machine model, which has been developed since the 1950s by many authors. For recent develop-

ments and more details about this model, the reader is referred to [43], [31] and [58]. In this model, computations for functions $f : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ between Baire spaces are explicitly defined via Type-2 machines. Roughly speaking, this means that for any input sequence a in $\mathbb{N}$ on a read-only input tape, the machine computes (in the discrete sense) and writes the sequence $f(a)$ on a one-way output tape. The idea is that the machine keeps reading digits from the input and doing computations (as any computer does) to get partial results written on the output tape. Since the input tape has an infinite number of digits, the computation may require an infinite number of steps to describe the exact output. Because it is desirable to get useful results in finite time, one requires the output tape to be one-way, i.e., the machine cannot change what it has already written on the tape, thus ensuring that one has partially correct results at any given moment (the longer one waits, the more accurate the results are). Computations of real functions $f : A \to B$, $A, B \subseteq \mathbb{R}$, can then be performed by encoding real numbers by sequences of rational numbers and employing a Type-2 machine to compute rational approximations of $f(x)$ with arbitrary precision from a suitable rational approximation of x. The Type-2 Turing machine is used in computable analysis as the model of computation. In this note, we use the computable analysis approach.

In the following, we present the precise definitions for encoding real numbers as well as computable real numbers and computable functions.

Definition 1. 1. A sequence $\{r_n\}$ of rational numbers is called a ρ-name of a real number x if there are three functions a, b and c from $\mathbb{N}$ to $\mathbb{N}$ such that for all $n \in \mathbb{N}$, $r_n = (-1)^{a(n)} \frac{b(n)}{c(n)+1}$ and

$$|r_n - x| \leq \frac{1}{2^n}. \tag{1}$$

2. A double sequence $\{r_{n,k}\}_{n,k\in\mathbb{N}}$ of rational numbers is called a ρ-name for a sequence $\{x_n\}_{n\in\mathbb{N}}$ of real numbers if there are three functions a, b, c from $\mathbb{N}^2$ to $\mathbb{N}$ such that, for all $k, n \in \mathbb{N}$, $r_{n,k} = (-1)^{a(k,n)} \frac{b(k,n)}{c(k,n)+1}$ and

$$\left|r_{n,k} - x_n\right| \leq \frac{1}{2^k}.$$

3. A real number x (a sequence $\{x_n\}_{n\in\mathbb{N}}$ of real numbers) is called computable if it has a computable ρ-name, i.e. there is a Type-2 machine that generates the ρ-name without input.

The notion of ρ-name extends in an obvious way to l-vectors. Thus a sequence $\{(r_{1n}, r_{2n}, \ldots, r_{ln})\}_{n\in\mathbb{N}}$ of rational vectors is called a ρ-name of $(x_1, x_2, \ldots, x_l) \in \mathbb{R}^l$ if $\{r_{jn}\}_{n\in\mathbb{N}}$ is a ρ-name of x_j, $1 \leq j \leq l$. It is easy to see from the definition that a ρ-name of a real number x is simply a code of x by rational numbers.

Next we present a notion of computability for open and closed subsets of $\mathbb{R}^l$ (cf. [58], Definition 5.1.15). We implicitly use ρ-names. For instance, to obtain names of open subsets of $\mathbb{R}^l$, we note that the set of rational balls $B(a, r) = \{x \in \mathbb{R}^l :$

$|x-a| < r\}$, where $a \in \mathbb{Q}^l$ and $r \in \mathbb{Q}$, is a subbase for the standard topology over $\mathbb{R}^l$. Depending on the ρ-names used, we obtain different notions of computability. We omit further details for lack of space.

Definition 2. 1. An open set $E \subseteq \mathbb{R}^l$ is called recursively enumerable (r.e. for short) open if there are computable sequences $\{a_n\}$ and $\{r_n\}$, $a_n \in E$ and $r_n \in \mathbb{Q}$, such that

$$E = \cup_{n=0}^{\infty} B(a_n, r_n).$$

Without loss of generality one can also assume that for any $n \in \mathbb{N}$, the closure of $B(a_n, r_n)$, denoted as $\overline{B(a_n, r_n)}$, is contained in E.

2. A closed subset $K \subseteq \mathbb{R}^l$ is called r.e. closed if there exist computable sequences $\{b_n\}$ and $\{s_n\}$, $b_n \in \mathbb{Q}^l$ and $s_n \in \mathbb{Q}$, such that $\{B(b_n, s_n)\}_{n\in\mathbb{N}}$ lists all rational open balls intersecting K.
3. An open set $E \subseteq \mathbb{R}^l$ is called computable (or recursive) if E is r.e. open and its complement E^c is r.e. closed. Similarly, a closed set $K \subseteq \mathbb{R}^l$ is called computable (or recursive) if K is r.e. closed and its complement K^c is r.e. open.

Roughly speaking, an open subset U of $\mathbb{R}^2$ is r.e. if there is a computer program that sketches the image of U by plotting rational open balls on a screen, which will eventually fill up U (but may take infinite time to do so). We may not know how well these balls are filling up U in any finite time if U is merely r.e. On the other hand, if U is recursive, then there is a program that plots the balls filling U up to precision 2^{-k} (in terms of Hausdorff distance) on input k [58].

Definition 3. Let A, B be sets, where ρ-names can be defined for elements of A and B. A function $f : A \to B$ is computable if there is a Type-2 machine such that on any ρ-name of $x \in A$, the machine computes as output a ρ-name of $f(x) \in B$.

When dealing with open sets in $\mathbb{R}^l$, we identify a special case of computability, which we call semi-computability. Let $\mathscr{O}(\mathbb{R}^l) = \{O | O \subseteq \mathbb{R}^l$ is open in the standard topology$\}$.

Definition 4. A function $f : A \to \mathscr{O}(\mathbb{R}^l)$ is called semi-computable if there is a Type-2 machine such that on any ρ-name of $x \in A$, the machine computes as output two sequences $\{a_n\}$ and $\{r_n\}$, $a_n \in \mathbb{R}^l$ and $r_n \in \mathbb{Q}$ such that

$$f(x) = \cup_{n=0}^{\infty} B(a_n, r_n).$$

Without loss of generality one can also assume that for any $n \in \mathbb{N}$, the closure of $B(a_n, r_n)$ is contained in $f(x)$.

We call this function semi-computable because we can tell in a finite time if a point belongs to $f(x)$, but we have to wait an infinite time to know that it does not belong to $f(x)$.

3 Description of results

In this section we describe some recent results concerning computability of continuous dynamical systems. We consider two types of results: (i) computability of important parameters and sets appearing in dynamical systems, and (ii) using dynamical systems as computing models.

In the line of (i), our first result concerns a very basic question – the computability of a single trajectory of a dynamical system defined by a (vector) ODE

$$y' = f(y). \tag{2}$$

This may seem like a trivial question – just use a standard numerical algorithm. However, these methods usually require a Lipschitz constant to ensure uniqueness of solutions, which is essential for computation. Since the behavior of a trajectory over time is in general unknown beforehand, one may not have a knowledge of a Lipschitz constant that can be used to compute the entire trajectory (actually it often happens that no such "global" Lipschitz constant exists).

This problem is studied by several authors. In [22], we show that if f is computable and effectively locally Lipschitz (meaning that we can locally compute Lipschitz constants), then we can compute the entire trajectory. This result is extended in [17]. There it is shown that if the solution is unique, then the solution must be computable over its lifespan (the maximal interval on which the solution exists), under the classical conditions ensuring existence of a solution to (2) for a given initial point. The idea is to generate all possible "tubes" which cover the solution, and then check if this cover is valid within the desired accuracy. The proof is constructive, although terribly inefficient in practice. Nevertheless, it solves the problem of computing a given trajectory for (2).

The result above is not surprising, since the Picard iteration scheme used in the classical existence proof is constructive. However, the issue remains as to whether or not one can compute the lifespan. In [22] we provide a negative answer, showing that even if f is analytic and computable, the lifespan is in general non-computable (i.e. not recursive). However, if f is computable, the lifespan is r.e. The non-computability of the lifespan suggests limitations concerning numerical methods for solving ODE problems, because numerical methods often assume the existence of some time interval where the solution is defined, and this assumption is crucial in error analysis. In the case where the lifespan is non-computable, one may have to settle for a numerical algorithm computing only a local solution.

We have also shown in [22] that the problem of determining whether or not the lifespan is bounded cannot be decided by a Turing machine, even if f is computable and analytic. This result is extended in [24] to the case where f is computable and polynomial. The result is further refined in [46], where it is shown that the set of all initial data generating solutions with lifespans longer than k, $k \in \mathbb{N}$, is in general not computable. The set is however r.e. if f is computable.

Next we describe some results related to the dynamics of a given system. In [61], it is shown that the domain of attraction of a computable and asymptotically stable

hyperbolic equilibrium point of the nonlinear system (2) is in general not recursive, though it is r.e. This tells us that the domain of attraction can be approximated from the inside on the one hand, but on the other hand there is no algorithm determining how far such an approximation is from filling up this domain. When restricted to planar systems, more can be said. For example, in [25], we show that the operator $\mathscr{F}$ is strictly semi-computable if we consider only structurally stable systems; on the other hand, $\mathscr{F}$ fails to be semi-computable if all C^1 systems are permitted, where $\mathscr{F}$ is the operator that takes two inputs, the description of the flow and a cover of an attractor, and outputs the domain of attraction for the given attractor. In [25] we also demonstrate how to decide whether or not there are limit cycles, and furthermore how to compute hyperbolic ones when given a compact set without an equilibrium point (equilibrium points are computable from f). As a consequence, all kinds of hyperbolic attractors in the plane can be computed, though their domains of attraction cannot.

We now turn to the issue (ii) of using dynamical systems as computing models. We have shown in [23] that the evolution of a given Turing machine can be embedded in the dynamics defined by polynomial differential equations, with some degree of robustness to perturbations. In other words, polynomial differential equations can simulate Turing machines. In [7] the following variation of the above result is given: for any given compact set $[a,b] \subseteq \mathbb{R}$, a function $f : [a,b] \to \mathbb{R}$ is computable if and only if it is computable by the "limit dynamics" of polynomial differential equations, i.e., there is a (vector) polynomial p such that given an initial point $x \in [a,b]$, the solution to the initial-value problem $y' = p(t,y)$, $y(0) = (x, y_{2,0}, \ldots, y_{n,0})$, with $y_{2,0}, \ldots, y_{n,0} \in \mathbb{R}$ independent of x, is composed of two components, which we suppose without loss of generality to be y_1, y_2, satisfying

$$|y_1(t) - f(x)| \leq y_2(t)$$

and $y_2(t) \to 0$ as $t \to \infty$ (i.e., y_1 converges towards $f(x)$ with error bounded by y_2).

There are interesting results by other authors, usually more related to control theory. Control theory is an interdisciplinary branch of engineering and mathematics that studies how to manipulate the parameters affecting the behavior of a system to produce the desired or optimal outcome. Some good introductions to control theory for mathematicians can be found in [60], [53].

Numerous interesting techniques and results have been obtained over the years by the control theory community. However they have not found their way into the dynamical systems community. In our opinion, this has various causes, ranging from lack of interaction between the two communities and, to some degree, because control theory is more application-oriented. For instance, many results focus on *hybrid systems* (see e.g. [13]), defined as differential equations with discontinuous right hand sides or having (some) discrete variables.

A topic of interest for control theory is *stability* [60]. Usually this notion is related to Lyapunov stability. In [3], the authors consider a particular class of discrete-time dynamical systems, defined by continuous piecewise affine functions. They show that the stability problem (in their version, "Is the system globally asymptotically

stable?") is non-computable, which establishes fundamental limitations for the computation of this kind of problem. Other notions of stability can be considered, e.g. shadowing or robustness, as done in [28]. The author focuses on the *reachability problem*: given some initial point in the state space, does the flow reach some region or point A? In [28] it is shown that the shadowing property is not enough to decide reachability by a computer, while robustness is sufficient.

The previous results rely on the paper [12]. There Collins shows that, in general, the reachable set of some initial region can be semi-computed (technically, lower-computed), but can only be computed under some special conditions. Another interesting algorithm to study the reachability problem is given in [15].

The reachability problem has been one of the most studied problems in the literature, and is interesting for dynamical systems since it has obvious resemblance to the problem of computing the domain of attraction of a given attractor. As a matter of fact, our results about computability of domains of attraction presented in [25] are based on some of these techniques and provide a good example of how control theory may be of use in dynamical systems.

Most results about the reachability problem give rise to undecidability (i.e., cannot be solved by an algorithm) as it is usually easy to encode the evolution of a given Turing machine in the dynamics of the system, e.g. [40], [8], [1], [6], [33], [23] and to show that the reachability problem is equivalent to the Halting Problem, the foremost undecidable problem in the theory of computation, cf. [2].

Despite this undecidability, these results use creative ways to analyze the dynamics of the system. Moreover, they depend critically on the use of exact computations. If some robustness to errors is allowed (in a weaker form than that required by structural stability), then usually the reachability problem is decidable as was mentioned in [28], but previously seen in other classes [38], [39], [23]. This fact was used in [25]. The idea is to cover some region with a grid of points (more precisely, small squares) and follow the individual evolution of each point to get an estimate of the domain of attraction. By using a larger grid (in absolute size) and a thinner mesh size, in the limit one can show rigourously that we compute the domain of attraction, even though exact computation takes "infinite time." Nevertheless, at each point of the computation, we have an estimate of this domain, with the error converging to 0 with time.

Another important area of study in control theory is *controllability* [60]. In controllability the aim is to investigate the possibility of forcing the system into a particular state by using an appropriate control signal. This topic has been partially studied by some members of the dynamical systems community, in control of chaos, which is based on the fact that any chaotic attractor contains an infinite number of unstable periodic orbits, and that one can use small perturbations to stabilize the trajectory into one of these periodic orbits [48].

The literature about computability and controllability focuses essentially on the computation of classes of "controllers" which allow the control of specific classes of systems: hybrid systems [37], [57], [16] and discrete-time semicontinuous systems [14].

Concerning complex dynamical systems, there is an exciting result by Braverman and Yampolsky [10]. They show that there is no algorithm which computes the Julia set J_c of the quadratic polynomial $f_c(z) = z^2 + c$ from the parameter c, using elaborate arguments involving Julia sets with Siegel disks. This shows that there are limitations when doing accurate computations of those pretty images of Julia sets usually presented to the public.

Other results of interest are those using shifts. In [40], Moore uses generalized shifts to show that basins of attraction, chaotic behavior or even periodicity are non-computable. This kind of result brings to the field of computability questions traditionally related to dynamical systems [29]. In particular, these include deriving necessary conditions for universality [18], computability of entropy [32], [54], [27], [26] [55], and understanding the "edge of chaos [35]."

Also along this line, some work has been done concerning computability of dynamical systems seen from a statistical perspective [30], [21]. We believe this is an interesting and promising topic of research.

4 Further work

The computability theory of continuous dynamical systems is still in an early stage of development, despite notable progress in recent years. Many important fundamental problems have not yet been studied. In general, the problems fall into two categories – computability and computational complexity.

As for computability, one topic of broad scope is to detect non-computable parameters and invariant sets of classical importance and ask further for the fine structure via the theory of degree of unsolvability. Examples are attractors/repellors and their basins in natural families of dynamical systems such as the Hénon attractor, the Rössler attractor, and the Lorenz attractor. Another interesting problem is to identify the analytic/geometric properties that are critical to ensure computability of an object under consideration. For example, in [61] we showed that there exists a C^∞ and polynomial-time computable function f defined on $\mathbb{R}^2$ such that the origin $(0,0)$ is the only sink of $dx/dt = f(x(t))$, and the domain of attraction of $(0,0)$ is not computable. However, the issue remains as to whether or not the domain of attraction of a computable polynomial system in the plane is computable.

It could also be interesting to investigate the computability of the dynamical systems used to model the motion of charged particles in modern particle accelerators. These devices (the LHC at CERN, the Tevatron at Fermilab, and many others) are among the most complex machines ever constructed, and numerous numerical codes are used in their design and operation; these numerical algorithms are correspondingly complex. Yet, the computability theory is still lacking.

When it comes to computational complexity, so far as we know, the only major problems which have been investigated are local solutions of the initial value problems for certain ordinary differential equations [31] and Julia sets [10], [45]. There are many processes and sets arising from dynamical systems which have been

proved to be computable but yet their computational complexity remains unknown. One such example is the Smale horseshoe. It can be shown that the horseshoes are computable, uniformly from the horseshoe maps [11]. Nevertheless, the difficulty of the computation is not yet known.

Acknowledgments. J. Buescu was partially supported by *Fundação para a Ciência e a Tecnologia*, Financiamento Base 2009 - ISFL/1/209. D. Graça was partially supported by *Fundação para a Ciência e a Tecnologia* and EU FEDER POCTI/POCI via SQIG - Instituto de Telecomunicações. DG was also attributed a Taft Research Collaboration grant which made possible a research visit to U. Cincinnati. N. Zhong was partially supported by the 2009 Taft Summer Research Fellowship.

References

1. Asarin, E., Maler, O.: Achilles and the tortoise climbing up the arithmetical hierarchy. J. Comput. System Sci. **57**(3), 389–398 (1998)
2. Asarin, E., Maler, O., Pnueli, A.: Reachability analysis of dynamical systems having piecewise-constant derivatives. Theoret. Comput. Sci. **138**, 35–65 (1995)
3. Blondel, V.D., Bournez, O., Koiran, P., Tsitsiklis, J.N.: The stability of saturated linear dynamical systems is undecidable. J. Comput. System Sci. **62**, 442–462 (2001)
4. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer (1998)
5. Blum, L., Shub, M., Smale, S.: On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. Bull. Amer. Math. Soc. **21**(1), 1–46 (1989)
6. Bournez, O.: Achilles and the Tortoise climbing up the hyper-arithmetical hierarchy. Theoret. Comput. Sci. **210**(1), 21–71 (1999)
7. Bournez, O., Campagnolo, M.L., Graça, D.S., Hainry, E.: Polynomial differential equations compute all real computable functions on computable compact intervals. J. Complexity **23**(3), 317–335 (2007)
8. Branicky, M.S.: Universal computation and other capabilities of hybrid and continuous dynamical systems. Theoret. Comput. Sci. **138**(1), 67–100 (1995)
9. Braverman, M., Cook, S.: Computing over the reals: foundations for scientific computing. Notices Amer. Math. Soc. **53**(3), 318–329 (2006)
10. Braverman, M., Yampolsky, M.: Non-computable Julia sets. J. Amer. Math. Soc. **19**(3), 551–0578 (2006)
11. Buescu, J., Graça, D., Zhong, N.: Computability and horseshoes. preprint (2009)
12. Collins, P.: Continuity and computability of reachable sets. Theor. Comput. Sci. **341**, 162–195 (2005)
13. Collins, P.: Chaotic dynamics in hybrid systems. Nonlinear Dyn. Syst. Theory **8**(2), 169–194 (2008)
14. Collins, P.: Computability of controllers for discrete-time semicontinuous systems. In: Proc. 18th International Symposium on the Mathematical Theory of Networks and Systems (2008)
15. Collins, P.: The reach-and-evolve algorithm for reachability analysis of nonlinear dynamical systems. Electron. Notes Theor. Comput. Sci. **223**, 87–102 (2008)
16. Collins, P.: Controllability and falsification of hybrid systems. In: Proc. European Control Conference, p. To appear (2009)
17. Collins, P., Graça, D.S.: Effective computability of solutions of differential inclusions — the ten thousand monkeys approach. Journal of Universal Computer Science **15**(6), 1162–1185 (2009)

18. Delvenne, J.C., Kurka, P., Blondel, V.: Decidability and universality in symbolic dynamical systems. Fund. Inform. **74**(4), 463–490 (2006)
19. Deutsch, D.: Quantum theory, the Church-Turing principle and the universal quantum computer. Proc. R. Soc. Lond. Ser. A **A400**, 97–117 (1985)
20. Duffing, G.: Erzwungene Schwingungen bei Veränderlicher Eigenfrequenz. Vieweg Braunschweig (1918)
21. Galatolo, S., Hoyrup, M., Rojas, C.: Effective symbolic dynamics, random points, statistical behavior, complexity and entropy. Inform. and Comput. (to appear)
22. Graça, D., Zhong, N., Buescu, J.: Computability, noncomputability and undecidability of maximal intervals of IVPs. Trans. Amer. Math. Soc. **361**(6), 2913–2927 (2009)
23. Graça, D.S., Campagnolo, M.L., Buescu, J.: Computability with polynomial differential equations. Adv. Appl. Math. **40**(3), 330–349 (2008)
24. Graça, D.S., Campagnolo, M.L., Buescu, J.: Computational bounds on polynomial differential equations. Appl. Math. Comput. **215**(4), 1375–1385 (2009)
25. Graça, D.S., Zhong, N.: Computing domains of attraction for planar dynamics. In: C.S. Calude, J.F. Costa, N. Dershowitz, E. Freire, G. Rozenberg (eds.) 8th International Conference on Unconventional Computation (UC 2009), LNCS 5715, pp. 179–190. Springer (2009)
26. Hertling, P., Spandl, C.: Computability theoretic properties of the entropy of gap shifts. Fundam. Inf. **83**, 141–157 (2008)
27. Hertling, P., Spandl, C.: Shifts with decidable language and noncombustible entropy. Discrete Math. Theor. Comput. Sci. **10**, 75–94 (2008)
28. Hoyrup, M.: Dynamical systems: stability and simulability. Math. Structures Comput. Sci. **17**, 247–259 (2007)
29. Hoyrupa, M., Kolçaka, A., Longo, G.: Computability and the morphological complexity of some dynamics on continuous domains. Theoret. Comput. Sci. **398**, 170–182 (2008)
30. Hoyrupa, M., Rojas, C.: Computability of probability measures and martin-löf randomness over metric spaces. Inform. and Comput. **207**, 830–847 (2009)
31. Ko, K.I.: Computational Complexity of Real Functions. Birkhäuser (1991)
32. Koiran, P.: The topological entropy of iterated piecewise affine maps is uncomputable. Discrete Math. Theor. Comput. Sci. **4**(2), 351–356 (2001)
33. Koiran, P., Moore, C.: Closed-form analytic maps in one and two dimensions can simulate universal Turing machines. Theoret. Comput. Sci. **210**(1), 217–223 (1999)
34. Lanford, O.E.: A computer-assisted proof of the feigenbaum conjectures. Bull. AMS **6**, 427–434 (1982)
35. Legenstein, R., Maass, W.: What makes a computational system dynamically powerful? In: S. Haykin, J.C. Principe, T. Sejnowski, J. Mcwhirter (eds.) New Directions in Statistical Signal Processing: From Systems to Brain, pp. 127–154. MIT Press (2007)
36. Lorenz, E.N.: Deterministic non-periodic flow. J. Atmos. Sci. **20**, 130–141 (1963)
37. Lygeros, J., Tomlin, C., Sastry, S.: Controllers for reachability specifications for hybrid systems. Automatica **35**, 349–370 (1999)
38. Maass, W., Orponen, P.: On the effect of analog noise in discrete-time analog computations. Neural Comput. **10**(5), 1071–1095 (1998)
39. Maass, W., Sontag, E.: Analog neural nets with gaussian or other common noise distributions cannot recognize arbitrary regular languages. Neural Comp. **11**, 771–782 (1999)
40. Moore, C.: Unpredictability and undecidability in dynamical systems. Phys. Rev. Lett. **64**(20), 2354–2357 (1990)
41. Peixoto, M.: Structural stability on two-dimensional manifolds. Topology **1**, 101–121 (1962)
42. van der Pol, B.: A theory of the amplitude of free and forced triode vibrations. Radio Review **1**, 701–710, 754–762 (1920)
43. Pour-El, M.B., Richards, J.I.: Computability in Analysis and Physics. Springer (1989)
44. Pour-El, M.B., Zhong, N.: The wave equation with computable initial data whose unique solution is nowhere computable. Math. Logic Quarterly **43**, 499–509 (1997)
45. Rettinger, R., Weihrauch, K.: The computational complexity of some julia sets. In: Proc. 35th annual ACM symposium on Theory of computing, pp. 177– 185. ACM (2003)

46. Rettinger, R., Weihrauch, K., Zhong, N.: Topological complexity of blowup problems. Journal of Universal Computer Science **15**(6), 1301–1316 (2009)
47. Rössler, O.E.: An equation for continuous chaos. Phys. Lett. A **57**(5), 397–398 (1976)
48. Shinbrot, T., Grebogi, C., Yorke, J.A., Ott, E.: Using small perturbations to control chaos. Nature **363**, 411–417 (1993)
49. Sipser, M.: Introduction to the Theory of Computation, 2nd edn. Course Technology (2005)
50. Smale, S.: Structurally stable systems are not dense. Amer. J. Math. **88**, 491–496 (1966)
51. Smale, S.: Differentiable dynamical systems. Bull. Amer. Math. Soc. **73**, 747–817 (1967)
52. Smale, S.: Mathematical problems for the next century. Math. Intelligencer **20**, 7–15 (1998)
53. Sontag, E.D.: Mathematical Control Theory, 2nd edn. Springer (1998)
54. Spandl, C.: Computing the topological entropy of shifts. Math. Log. Quart. **53**(4-5), 493–510 (2007)
55. Spandl, C.: Computability of topological pressure for shifts of finite type with applications in statistical physics. Electr. Notes Theor. Comput. Sci. **202**, 385–401 (2008)
56. Tucker, W.: A rigorous ode solver and smale's 14th problem. Found. Comput. Math. **2**(1), 53–117 (2002)
57. Vidal, R., Schaffert, S., Shakernia, O., Lygeros, J., Sastry, S.: Decidable and semi-decidable controller synthesis for classes of discrete time hybrid systems. In: Proc. 40th IEEE Conference on Decision and Control, pp. 1243–1248 (2001)
58. Weihrauch, K.: Computable Analysis: an Introduction. Springer (2000)
59. Weihrauch, K., Zhong, N.: Is wave propagation computable or can wave computers beat the Turing machine? Proc. London Math. Soc. **85**(3), 312–332 (2002)
60. Zabczyk, J.: Mathematical Control Theory: An Introduction. Birkhäuser (1992)
61. Zhong, N.: Computational unsolvability of domain of attractions of nonlinear systems. Proc. Amer. Math. Soc. **137**, 2773–2783 (2009)