Polynomial-Time Solution of Initial Value Problems Using Polynomial Enclosures

Amin Farjudian

Division of Computer Science
University of Nottingham Ningbo, China

Amin.Farjudian@nottingham.edu.cn
www.cs.nott.ac.uk/˜avf/

Abstract. Domain theory has been used with great success in providing a semantic framework for Turing computability, over both discrete and continuous spaces. On the other hand, classical approximation theory provides a rich set of tools for computations over real functions with (mainly) polynomial and rational function approximations.
We present a semantic model for computations over real functions based on *polynomial enclosures*. As an important case study, we analyse the convergence and complexity of Picard's method of initial value problem solving in our framework. We obtain a geometric rate of convergence over Lipschitz fields and then, by using Chebyshev truncations, we modify Picard's algorithm into one which runs in *polynomial-time* over a set of polynomial-space representable fields, thus achieving a reduction in complexity which would be impossible in the step-function based domain models.

Keywords: computable analysis, computational complexity, initial value problem, Picard's method, approximation theory, Chebyshev series

1 Introduction

In classical mathematical analysis, for the most part one abstracts away from effective representations of objects. The constructive view of analysis [6] brings in a distinction between finitely representable objects and those that can only be approximated with finitely representable ones. In computable analysis [25] one works at an even lower level of abstraction, where claims of existence are required to be proven via procedures that are implementable on a Turing machine.

For instance, whereas in classical analysis it is true that any initial value problem (IVP) with a continuous field has a solution, in computable analysis proper this claim is only valid when a solution can be worked out using some algorithm which generates approximations to the solution to within any desired accuracy.

As such, in partial orders one finds a natural setting in which essential concepts such as approximation and convergence can be formulated [13]. As a special subclass of partial orders, domains [2] have been studied extensively as a semantic model of computation. Cartesian closed categories of domains with an interval domain object [11] provide a denotational semantic framework for computations over continuous spaces.

There is a canonical way of constructing function spaces in the category of domains via *step functions* [2, Chapter 4]. For real function spaces this approach is essentially equivalent to approximating functions via piece-wise constant enclosures. This is theoretically sufficient as with this construction many of the concepts from classical analysis can be reformulated in a domain theoretic setting [8, 9].

On the other hand, there is a long tradition in *approximation theory* [3, 21] with a very rich literature, in which computations over real functions are reduced to those over their relatively simpler polynomial (or even rational function) approximations. By the classic theorems of Jackson [15] and Bernstein [4] there is a tight link between the analytic properties of a function and how it can be approximated with polynomials.

At a practical level, almost all widely used maths software libraries provide some kind of functionality based on approximation theory. In fact there are some that have been written exclusively based on function approximations.[1] These libraries are mainly geared towards fast numerical computations based on the machine's floating-point unit. This means that a rigorous analysis of (floating-point) inaccuracies or the theoretical complexity of computations beyond the reach of the machine's resources is typically of a secondary concern.

Our aim is to present a semantic model *along the lines* of domain based ones, in which both the convergence and the complexity of computations can be studied. We will make sure that our framework is complete, i. e. all results can be approximated to within any desired accuracy. We pick IVP solving as an important case to study, and analyse the convergence and complexity of Picard's method as adapted to our framework.

2 Polynomial enclosures

In what follows, $\mathbb{N}_+$ denotes the set of positive natural numbers and $C[a,b]^n$ denotes the set of continuous functions from the n-dimensional cube $[a,b]^n$ to $\mathbb{R}$, where $a, b \in \mathbb{R}$, $n \in \mathbb{N}_+$ and $a \leq b$.

For $n \in \mathbb{N}_+$ and $f, g \in C[a,b]^n$ we define the function enclosure $[f,g]$ by

$$[f,g] \coloneqq \{h \in C[a,b]^n \mid \forall \bar{x} \in [a,b]^n : f(\bar{x}) \leq h(\bar{x}) \leq g(\bar{x})\}$$

If $\exists \bar{x}_0 \in [a,b]^n : g(\bar{x}_0) < f(\bar{x}_0)$ then $[f,g]$ will be empty. The functions f and g are called the *lower* and the *upper boundaries* of the enclosure $[f,g]$, respectively. For each point $t \in [a,b]^n$, the (local) *width* of the enclosure $[f,g]$ at t is defined as $\mathrm{w}_{[f,g]}(t) \coloneqq g(t) - f(t)$. The (global) width of the enclosure is defined as $\mathrm{w}([f,g]) \coloneqq \max\{\mathrm{w}_{[f,g]}(t) \mid t \in [a,b]^n\}$. Note that $\mathrm{w}([f,g])$ is well defined as $[a,b]^n$ is compact and both f and g are assumed to be continuous.

Definition 1 ($\Gamma(h)$: graph of a function enclosure). *Let $h = [f,g]$ be an enclosure where $f, g \in C[a,b]^n$. By the* graph *of h we mean the set of all* points *in $[a,b]^n \times \mathbb{R}$ lying between the graphs of its lower and upper boundaries, i. e.*

$$\Gamma(h) \coloneqq \{(\bar{x}, r) \in [a,b]^n \times \mathbb{R} \mid f(\bar{x}) \leq r \leq g(\bar{x})\}$$

[1] such as the free and open source MATLAB library *chebfun* available from `http://www2.maths.ox.ac.uk/chebfun`.

Definition 2 (<P, R>: centered rational polynomial enclosure). *Let $n \in \mathbb{N}_+$ and assume that P and R are polynomials with rational coefficients in n variables $X_1, \ldots, X_n$, i. e. $P, R \in \mathbb{Q}[X_1, \ldots, X_n]$. By the* centered rational polynomial enclosure *<P, R> we mean the* non-empty[2] *function enclosure $[f, g]$ in which $f, g \in C[a, b]^n$ and $\forall \bar{x} \in [a, b]^n : g(\bar{x}) = P(\bar{x}) + R(\bar{x}) \wedge f(\bar{x}) = P(\bar{x}) - R(\bar{x})$. We will refer to P and R as the* center *and the* radius *of the enclosure <P, R>, respectively.*

As we have imposed the non-emptiness condition, then $\forall \bar{x} \in [a, b]^n : R(\bar{x}) \geq 0$. The purpose of restricting the coefficients of P and R to rational numbers has been to ensure that all of our enclosures are finitely representable.

Note that the notations $[f, g]$ and $<P, R>$ for function enclosures are indeed interchangeable as we have $[f, g] = <(g + f)/2, (g - f)/2>$. In this paper there will be no decoupling of the boundaries of the enclosures in the sense that we will always add the error estimates to both the upper and the lower boundaries of an enclosure. Thus, we will opt for using the centered-enclosure notation as in Definition 2.

Remark 1. Throughout this paper, by a polynomial enclosure we will always mean a centered rational one.

For each $n \in \mathbb{N}_+$ we denote the set of all non-empty enclosures with boundaries in $C[a, b]^n$ by $\mathbb{FE}[a, b]^n$, i. e. $\mathbb{FE}[a, b]^n := \{[f, g] \mid f, g \in C[a, b]^n, \forall t \in [a, b]^n : f(t) \leq g(t)\}$. We define the order $\sqsubseteq$ over this set as follows: $\forall h_1, h_2 \in \mathbb{FE}[a, b]^n : h_1 \sqsubseteq h_2 \Leftrightarrow h_2 \subseteq h_1$. The pair $(\mathbb{FE}[a, b]^n, \sqsubseteq)$ is a partial order which we simply denote by $\mathbb{FE}[a, b]^n$. The pair $(\mathbb{PE}[a, b]^n, \sqsubseteq)$ in which $\mathbb{PE}[a, b]^n$ is the set of polynomial enclosures also forms a poset under the order inherited from $\mathbb{FE}[a, b]^n$, which we simply write as $\mathbb{PE}[a, b]^n$.

A sequence $\langle [f_i, g_i] \rangle_{i \in \mathbb{N}}$ of enclosures is said to converge to $[f, g]$ if $\forall k \in \mathbb{N} : [f_k, g_k] \sqsubseteq [f, g]$, $f = \lim_{i \to \infty} f_i$ and $g = \lim_{i \to \infty} g_i$, where the limits are taken with respect to the supremum norm, which is defined for each $f \in C[a, b]^n$ as $\| f \| = \sup\{f(\bar{x}) \mid \bar{x} \in [a, b]^n\}$. An element $h \in \mathbb{FE}[a, b]^n$ is said to be *maximal* if $\mathrm{w}(h) = 0$, in which case $h = [f, f]$ for some $f \in C[a, b]^n$. For simplicity, we will identify the maximal element $[f, f]$ with f. Using this convention one may talk about sequences of *function enclosures* in $\mathbb{FE}[a, b]^n$ that converge to *functions* in $C[a, b]^n$.

Note that neither $\mathbb{FE}[a, b]^n$ nor $\mathbb{PE}[a, b]^n$ is complete under the notion of convergence just defined. For instance, consider the sequence $h_i = [f_i, g_i]$ of enclosures in $\mathbb{PE}[0, 1]$ defined as $\forall i \in \mathbb{N}, x \in [0, 1] : f_i(x) = 0, g_i(x) = x^i$. This sequence forms a chain as $\forall i \in \mathbb{N} : h_i \sqsubseteq h_{i+1}$, but 'the limit' of $\langle g_i \rangle_{i \in \mathbb{N}}$ is the non-continuous function $\gamma : [0, 1] \to \mathbb{R}$ which satisfies $\gamma(x) = 0$ if $x \in [0, 1)$ and $\gamma(1) = 1$.

3 Solving initial value problems using an oracle machine

Let $m \in \mathbb{N}_+$ and consider the initial value problem (IVP)

$$\begin{cases} y'(t) = F(t, y(t)) \\ y(t_0) = y_0 \end{cases} \tag{1}$$

[2] It will be interesting to see (in our future work) what more we may achieve by removing the non-emptiness condition.

in which $F : \Omega \to \mathbb{R}^m$ will be referred to as the *field* of the IVP, and for which we seek a solution $y : [t_0, a] \to \mathbb{R}^m$ for a suitable $a \geq t_0$. We assume that $\Omega \subseteq \mathbb{R}^{m+1}$ is open and includes the initial point, i. e. $(t_0, y_0) \in \Omega$. Peano's existence theorem states that the mere continuity of the field F guarantees the *existence* of a solution [22]. Furthermore, if F is *Lipschitz continuous* in its second argument, i. e.

$$\exists L \in \mathbb{R} : \forall t \in \mathbb{R}, r_1, r_2 \in \mathbb{R}^m : \|F(t, r_1) - F(t, r_2)\|_{\sup} \leq L\|r_1 - r_2\|_{\sup}$$

then by Picard-Lindelöf theorem the IVP has a *unique* solution. The Lipschitz condition is sufficient but not necessary for the uniqueness [22]. A *necessary and sufficient condition* for the uniqueness of the solution was provided by Hiroshi Okamura, a generalisation of which can be found in [26].

If we integrate both sides of the differential equation in (1) and incorporate the initial condition, we obtain the following integral equation:

$$y(x) = y_0 + \int_{t_0}^{x} F(t, y(t)) \, \mathrm{d}t \tag{2}$$

Assume that for some suitable $a \geq t_0$ and $b > 0$ satisfying $[t_0, a] \times [-b, b]^n \subseteq \Omega$ the operator $Pic_F(g) = \lambda x. y_0 + \int_{t_0}^{x} F(t, g(t)) \, \mathrm{d}t$ is an endofunction over $[t_0, a] \times [-b, b]^n$. Then any fixed-point of Pic_F—if any does indeed exist—would be a solution to both the integral equation (2) and the IVP (1). In fact, Lipschitz continuity guarantees that the conditions for the Banach fixed-point theorem are satisfied,[3] so all one needs to do is to apply Pic_F repeatedly to obtain better approximations of the solution, a process famously known as *Picard's method of IVP solving*.

Assume that the field F and the initial point (t_0, y_0) are computable. Then under the Lipschitz assumption Picard's method yields a computable solution, whereas without the Lipschitz assumption all of the solutions could turn out to be non-computable [1].

Remark 2. Throughout this paper, the only notion of computability over real numbers that we will consider will be that of the Type-2 Theory of Effectivity (TTE) [25].

From now on, without loss of generality, we will focus on the one-dimensional case and will consider the following IVP:

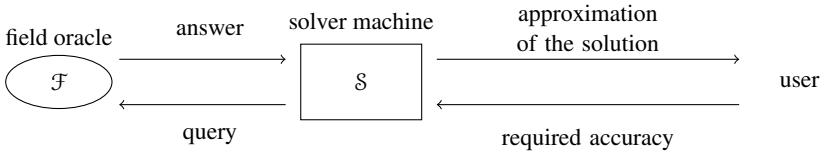
$$\begin{cases} y'(t) = F(t, y(t)) \\ y(0) = 0 \end{cases} \tag{3}$$

for which we seek a solution $y : [0, a] \to [-b, b]$, for suitable $a \geq 0$ and $b \geq 0$.[4]

We draw inspiration from Ko's work [17] and adopt an *oracle machine* model for a clear complexity analysis (as in Fig. 1). As we have simplified the initial condition to $(0, 0)$, the *solver* machine will only need:

1. a socket into which one plugs in a *field* oracle, i. e. an oracle which provides information about the field to the solver machine;
2. a pair of input and output tapes for interaction with the user (i. e. the outside world).

Fig. 1 An oracle machine for solving IVPs.



The user sends the required accuracy to the solver machine $\mathcal{S}$ in the form of a natural number k, and in return will be expecting a polynomial enclosure $<y_k, r_k>$ of the solution y whose width over $[0, a]$ is smaller than 2^{-k}. The solver machine in turn tries to obtain enough information about the field from its oracle $\mathcal{F}$ until it can provide the required approximation of the solution.

The advantage of using an oracle machine model is that we can abstract away from the cost of computing the field, and instead focus on the actual cost of IVP solving as performed by the solver machine. Nonetheless, the cost of handling queries and answers, and the cost of operations inside the solver machine *will* be taken into account.

Let us recount a rough sketch of how algorithms such as those of [10, 12]—where step-functions are used to approximate real functions—would work in this setting. The solver starts with the most conservative rectangle enclosing the graph of the solution, i. e. $([0, a], [-b, b])$. At iteration n, the solver possesses an enclosure of the graph of the solution made up of 2^n rectangles $\{B_i^n \mid 0 \le i < 2^n\}$, in which each rectangle B_i^n has got the horizontal side $[ai/2^n, a(i+1)/2^n]$. The solver machine sends each of these rectangles as queries to the field oracle.

The field oracle $\mathcal{F}$ possesses an 'encoding' of a domain theoretic extension IF of the field F. As such, in response to each query B_i^n it returns an interval enclosure $\mathit{IF}(B_i^n)$ of $F(B_i^n)$, i. e. $F(B_i^n) \subseteq \mathit{IF}(B_i^n)$.

Next, the solver performs a domain-theoretic integration on the step-function enclosure $\{([ai/2^n, a(i+1)/2^n], \mathit{IF}(B_i^n)) \mid 0 \le i < 2^n\}$, and works out an enclosure of this integral by 2^{n+1} rectangles $\{B_i^{n+1} \mid 0 \le i < 2^{n+1}\}$, which forms the next approximation of the solution y, and then the solver moves on to iteration $n + 1$.

This means that the mere number of queries and answers grows exponentially in n, which puts a heavy burden on the communications between the machine and its oracle. To address this problem, we will consider a different protocol for communications in which queries from the solver machine $\mathcal{S}$ to the field oracle $\mathcal{F}$ are of the form $(n, <P, R>)$, where $n \in \mathbb{N}$, and $<P, R>$ is a polynomial enclosure in $\mathbb{PE}[0, a]$, and answers from $\mathcal{F}$ to $\mathcal{S}$ come in the form of polynomial enclosures in $\mathbb{PE}[0, a]$, so do the approximations to the solution y as sent out to the user by $\mathcal{S}$.

A denotational semantic model of communication protocols (such as ours) over real numbers has been studied in a broader context in [19, 18]. Here we focus on the complexity theoretic implications of using one specific protocol.

[3] See e. g. [22, page 79].

[4] Generalising our results to the m-dimensional case and $y : [\alpha, \beta] \to [-b, b]^m$ will be straightforward.

3.1 Semantic behaviour of the field oracle $\mathcal{F}$

It is instructive to begin by viewing the input to $\mathcal{F}$ as a pair consisting of a number and a *function* enclosure. Assume that $\mathcal{F}$ receives a query $(n, [\alpha, \beta])$ on its input tape, where $n \in \mathbb{N}$ and $[\alpha, \beta] \in \mathbb{IE}[0, a]$. We expect the respective output to be an approximation of the image of $[\alpha, \beta]$ under the field F, to within 2^{-n} accuracy. So, consider the set $\Phi := \{(t, s) \in [0, a] \times \mathbb{R} \mid \exists z \in \mathbb{R} : \alpha(t) \leq z \leq \beta(t) \wedge s = F(t, z)\}$. It is easy to see that there are two functions $f, g \in C[0, a]$ for which we have $\Phi = \Gamma([f, g])$, i. e. the graph of $[f, g]$ (see Definition 1). We denote the enclosure $[f, g]$ as $F([\alpha, \beta])$, i. e.

$$F([\alpha, \beta]) := [f, g] \tag{4}$$

Next, we consider two polynomials ϕ and ψ that approximate f and g from below and above to within 2^{-n} accuracy, respectively, i. e. $[\phi, \psi] \sqsubseteq [f, g] \sqsubseteq [\phi + 2^{-n}, \psi - 2^{-n}]$. Remember that according to our protocols, the input to $\mathcal{F}$ is a pair $(n, <P, R>)$, in which $<P, R> \in \mathbb{IP}[0, a]$. Hence we define:

Definition 3 (Lipschitz field oracle). *The field oracle $\mathcal{F}$ is said to be* Lipschitz *with constant L if for each $n \in \mathbb{N}$ and $<P, R> \in \mathbb{IP}[0, a]$, the input query $(n, <P, R>)$ is responded with some $<\tilde{P}, \tilde{R}> \in \mathbb{IP}[0, a]$, in which $<\tilde{P}, \tilde{R}>$ is an enclosure of $F(<P, R>)$ (see (4) above) and $\forall t \in [0, a] : 0 \leq \tilde{R}(t) \leq 2^{-n} + LR(t)$.*

Even though we abstract away from what goes on inside the oracle $\mathcal{F}$, yet to demonstrate that Definition 3 is a plausible one, we have presented a way of obtaining Lipschitz oracles from analytic fields using polynomial approximations in Appendix A.

4 Convergence of Picard's method

Let us reformulate Picard's method in our setting. The idea is for the solver machine to work out a sequence $\{<y_n, r_n> \mid n \in \mathbb{N}\}$ of polynomial enclosures in such a way that they all enclose the final solution y, i. e. $\forall n \in \mathbb{N} : y \in <y_n, r_n>$:

base case: $y_0(x) = 0$, $r_0(x) = b$.

recursive step: Assume that the solver $\mathcal{S}$ has worked out the enclosure $<y_{n-1}, r_{n-1}>$. At this point $\mathcal{S}$ sends the query $(n, <y_{n-1}, r_{n-1}>)$ to the field oracle $\mathcal{F}$, and in return it receives the polynomial enclosure $<\tilde{P}_n, \tilde{R}_n>$. To obtain the next approximation $<y_n, r_n>$, all $\mathcal{S}$ needs to do is symbolic integration over polynomials, i. e.

$$y_n(t) = \int_0^t \tilde{P}_n(x)\,\mathrm{d}x, \quad r_n(t) = \int_0^t \tilde{R}_n(x)\,\mathrm{d}x \tag{5}$$

Theorem 1 (geometric convergence). *Consider the IVP (3) and the configuration of Fig. 1. Moreover, assume that the field oracle $\mathcal{F}$ is Lipschitz. Then the successive polynomial enclosures $<y_n, r_n>$ sent on the output tape by the solver machine converge to the solution y of the IVP* geometrically*, that is, for some constant $c > 1$, the width of $<y_n, r_n>$ decreases in $O(c^{-n})$.*

Proof. The main idea of the proof is similar to the usual one for the classical Picard-Lindelöf theorem. A detailed proof is given in Appendix B. □

5 Reducing the complexity to polynomial-time

One of the many ways with which the oracle $\mathcal{F}$ can work out a polynomial enclosure in response to any query from the solver machine $\mathcal{S}$ is via polynomial compositions (see Appendix A for a detailed account). Let $<\tilde{P}_n, \tilde{R}_n>$ be the answer that $\mathcal{F}$ sends to $\mathcal{S}$ in response to the query $(n, <y_{n-1}, r_{n-1}>)$ where $\tilde{P}_n(X) = P_n(X, y_{n-1}(X))$ and $P_n(X_1, X_2) \in \mathbb{Q}[X_1, X_2]$ is a polynomial approximation of the field F to within 2^{-n} accuracy.

Let us write q_n for the maximum degree of X_2 in $P_n(X_1, X_2)$, and d_n for the degree of the n^{th} polynomial approximation y_n of the solution y generated by the solver machine. It is straightforward to verify that the degrees of the polynomials $\{y_n \mid n \in \mathbb{N}\}$ satisfy $d_{n+1} \geq 1 + q_{n+1} d_n$ for all $n \in \mathbb{N}$, and $d_0 = 0$.

This should give us an idea of how the degrees of the approximants to the solution grow. For instance, even if $\forall n \in \mathbb{N} : q_n = 2$, then d_n grows at least exponentially. One way to address this inefficiency is to first approximate each of the polynomials $\tilde{P}_n$ and $\tilde{R}_n$ by lower degree polynomials $\tilde{y}_n$ and $\tilde{r}_n$, respectively, and then proceed to work out the next approximation to the solution $<y_n, r_n>$ accordingly.

There are various ways of approximating one polynomial with another of a lower degree. The approach which we consider here is based on using *truncated Chebyshev series*. The main reason for choosing this method over (say) the straightforward truncated Taylor series is that in numerical work, the Chebyshev basis is a much more well-behaved one compared with the monomial basis, a well-known fact spelled out in any standard text on function approximation [24, 20].

Another reason is that for any rational polynomial, the calculation of Chebyshev truncation involves only arithmetic over rational numbers, in contrast to methods such as the Carathéodory-Fejér [14] which involves computations of eigenvalues that are not necessarily rational.[5]

Assume that a polynomial p is written as $\sum_{i=0}^{n} a_i x^i$ in the monomial basis, and as $\sum_{i=0}^{n} b_i T_i(x)$ in the Chebyshev basis, in which $T_i(x) = \sum_{j=0}^{i} t_{ij} x^j$ is the i-th Chebyshev polynomial of the first kind ($i \geq 0$).[6] Going from the latter to the former is straightforward, and in the other direction, the coefficients b_i can be calculated as the entries of the vector B, which is the solution of the system of linear equations $S \times B = A$, in which $A = [a_0 \ldots a_n]^T$, $B = [b_0 \ldots b_n]^T$ and $S = [s_{i,j}]_{0 \leq i,j \leq n}$ is upper triangular with entries as follows:

$$s_{i,j} = \begin{cases} t_{ji} & \text{if } i \leq j \\ 0 & \text{if } i > j \end{cases} \tag{6}$$

For each $k \in \mathbb{N}$, the *Chebyshev-truncation* of p up to degree k is defined to be the polynomial $\mathrm{ct}(p, k) := \sum_{i=0}^{k'} b_i T_i(x)$ in which $k' = \min\{k, n\}$. As $\forall i \in \mathbb{N}, x \in [-1, 1] : |T_i(x)| \leq 1$ and as we will focus only on the values of x in $[0, 1]$, then we define the *Chebyshev-bound* of this truncation to be $\mathrm{cb}(p, k) := \sum_{i=k+1}^{n} |b_i|$.

For $\rho \geq 0$, let $C_\rho = \{\rho \mathrm{e}^{\mathrm{i}\theta} \mid 0 \leq \theta < 2\pi\}$ be the circle of radius ρ in the complex plain. We write E_ρ to denote the open region bounded by the ellipse which is obtained as the

[5] Nonetheless, we will be considering the Carathéodory-Fejér method in our future work due to its remarkable performance in practice.

[6] One can find a comprehensive treatment of Chebyshev polynomials in any standard book on approximation theory, e. g. [24, 21].

image of the circle C_ρ under the Joukowsky map $z \mapsto (z + z^{-1})/2$. The following is a corollary of Bernstein's theorem [5]:

Theorem 2. *Assume that $p \in \mathbb{R}[x]$ is a real polynomial and choose $\rho > 1$ and $C \geq 0$ in such a way that $\forall z \in E_\rho : |p(z)| \leq C$. Then the Chebyshev coefficients of p satisfy $b_k \leq 2C\rho^{-k}$. As a consequence $\forall n \in \mathbb{N} : \mathrm{cb}(p, n) \leq 2C\rho^{-n}/(\rho - 1)$.*

Now we focus on the solution of the equation (3) for values of $t \in [0, 1]$ and modify the algorithm of Section 4 to reduce the growth of the *size* of the polynomials y_n and r_n, i. e. both the degree and the bit size of coefficients. More precisely, after receiving the polynomial enclosure $<\tilde{P}_n, \tilde{R}_n>$ from the field oracle, the solver first works out the Chebyshev truncation of $\tilde{R}_n$ up to degree n to obtain the polynomial $\hat{r}_n \coloneqq \mathrm{ct}(\tilde{R}_n, n)$, and adds the bound $e(\hat{r}_n) \coloneqq \mathrm{cb}(\tilde{R}_n, n)$ to $\hat{r}_n$. Next it considers the Chebyshev truncation of $\tilde{P}_n$ up to degree n, which we write as $\hat{y}_n \coloneqq \mathrm{ct}(\tilde{P}_n, n)$, and adds the bound $e(\hat{y}_n) \coloneqq \mathrm{cb}(\tilde{P}_n, n)$ to $\hat{r}_n + e(\hat{r}_n)$. At this point our polynomials satisfy $\deg(\hat{y}_n) \leq n$ and $\deg(\hat{r}_n) \leq n$.

The final concern is that the sizes of the representations of the coefficients of $\hat{y}_n$ and $\hat{r}_n$ may have grown too large. So, assume that $\hat{y}_n(x) = \sum_{i=0}^{n} a_i x^i$ in monomial basis and for each $i \leq n$, let $p_i/2^n$ be the largest dyadic number with denominator 2^n which is not greater than $|a_i|$, and define $\mathring{a}_i$ as $p_i/2^n$ if $a_i \geq 0$, and $-p_i/2^n$ otherwise. We define $\tilde{y}_n(x) \coloneqq \sum_{i=0}^{n} \mathring{a}_i x^i$. It should be clear that, as $\forall i \leq n : |a_i - \mathring{a}_i| \leq 2^{-n}$, we will have $\|\tilde{y}_n - \hat{y}_n\| \leq (n + 1)2^{-n}$. Using the same method, we obtain $\tilde{r}_n$ from $\hat{r}_n$ by approximating its coefficients with appropriate dyadic numbers, with an added error of $(n + 1)2^{-n}$. So, the reduction of the sizes of the coefficients have added another $2(n + 1)2^{-n} = (n + 1)2^{-(n-1)}$ to our error estimates. Now we define:

$$y_n(t) = \int_0^t \tilde{y}_n(x)\,\mathrm{d}x, \quad r_n(t) = \int_0^t \left(\tilde{r}_n(x) + e(\hat{r}_n) + e(\hat{y}_n) + (n + 1)2^{-(n-1)}\right)\mathrm{d}x \tag{7}$$

In order to be able to use Theorem 2, we add the following assumption which states that *in the modified algorithm*:

$$\exists C \in \mathbb{N} : \forall n \in \mathbb{N} : \forall z \in E_5 : |\tilde{P}_n(z)| < C \wedge |\tilde{R}_n(z)| < C \tag{8}$$

We have chosen the elliptic region E_5 as it fully contains the circle C_2. This ensures that the sizes of the coefficients $\mathring{a}_i$ ($i \geq 0$) remain within the desired bounds. To see this, assume that the complex function f is analytic in a neighbourhood D of C_ρ. Then inside D, the function f is equal to its Taylor series about 0, i. e. $f(z) = \sum_{i=0}^{\infty} a_i z^i$. In particular, we have:

$$\forall i \in \mathbb{N} : a_i = \frac{1}{2\pi\mathrm{i}} \int_{C_\rho} \frac{f(z)}{z^{i+1}}\,\mathrm{d}z \tag{9}$$

Proposition 1. *Assume that $\rho > 1$ and $\exists C > 0 : \forall z \in C_\rho : |f(z)| < C$. Then $\forall k \in \mathbb{N} : |a_k| < C/\rho^k$.*

Proof. Straightforward from (9). □

Under assumption (8) and by using Proposition 1 (and the fact that C_2 is contained in E_5), the coefficients of the polynomial $\hat{y}_n(x) = \sum_{i=0}^{n} a_i x^i$ satisfy $\forall i : |a_i| \leq C$, which implies that each $\mathring{a}_i$ will be representable in $(n\lceil\log(C)\rceil + 1)$ bits, i. e. $O(n)$ bits. A similar

argument holds for the coefficients of $\hat{r}_n$ and $\tilde{r}_n$. As such we have reduced the degrees of our polynomials to $O(n)$, and by dyadic approximations, we have reduced the sizes of their representations to $O(n^2)$. Nonetheless, how much $\mathcal{F}$ blows up the sizes of its answers is not under our control. Therefore, we consider the field oracles for which, the size of the answer to a query is bounded by a polynomial function of the size of the query:

Definition 4 (p-representable field oracle). *Assume that the* size of the representation *of each finite object x is written as* $\mathrm{s}(x)$*. We say that the field oracle $\mathcal{F}$ is polynomial-space representable,* p-representable *for short, if there exists a polynomial $q \in \mathbb{N}[X]$ such that for each input query $(n, <P, R>)$, the answer polynomial enclosure $<\tilde{P}, \tilde{R}>$ satisfies* $\mathrm{s}(<\tilde{P}, \tilde{R}>) \leq q(m)$*, where* $m = \max\{n, \mathrm{s}(<P, R>)\}$.[7]

Remark 3. It is worth mentioning that according to Jackson-Bernstein Theorem [17, page 254], by assuming the field oracle to be p-representable, we are effectively excluding the fields that are not infinitely differentiable. Yet by another theorem of Bernstein [21, Theorem 5.2.1, page 148], we are effectively including all analytic fields.

We can prove that for all values of $\tau \leq \min(a, 1)$, the enclosures $<y_n, r_n>$ do indeed enclose the solution y over $[0, \tau]$, and in fact:

Theorem 3 (polynomial-time complexity). *For any Lipschitz p-representable field oracle $\mathcal{F}$ and under assumption (8), the modified algorithm solves the associated IVP in polynomial-time over* $[0, \tau]$*, in which* $\tau \leq \min(a, 1)$*.*

Proof (Sketch). The fact that our modified algorithm converges follows from (7) and (8) and Theorem 2.

To prove polynomial-time complexity, first note that the volume of data on the channels is capped as a result of the truncations on the one-hand, and the assumption that the field is p-representable on the other hand. Regarding the cost of computations inside the solver, symbolic integration in monomial basis can be done in polynomial-time, and the rewriting of polynomials in Chebyshev basis is also polynomial-time as the matrix S in (6) is upper triangular with non-zero diagonal entries. For the details of this proof, please see Appendix C. □

5.1 Analysis of our assumption

Admittedly, assumption (8) lacks the elegance that one would hope for. More importantly, it is not clear how restrictive this assumption is.

Note that for equation (3), Picard's method guarantees a lifetime of $r \leq \min(a, b/M)$ for the solution, in which M is the maximum norm of F over $[0, a] \times [-b, b]$. In Subsection 3.1, we required the field to return enclosures $<\tilde{P}, \tilde{R}>$ that were 'tight enough' in response to queries $(n, <P, R>)$. We considered the field and the polynomials as defined over real numbers. If we strengthen this requirement by interpreting the field and

[7] For a more detailed account of this definition, including the precise account of the representation of a polynomial, see Appendix C.

the polynomials involved over a certain subset of *complex* numbers, which in our case should contain E_5, we can guarantee polynomial-time complexity for our algorithm assuming $5 \leq \min(a, b/(2M))$.[8]

But this seems to leave out some very simple equations such as $y'(t) = 2y(t) + 2$. The problem is that the lifetime as guaranteed by Picard's method can be a gross underestimate of the real one. For instance, for the IVP $y'(t) = 2y(t) + 2$ with the initial condition $y(0) = 0$, the unique solution $y(t) = \exp(2t) - 1$ is defined over *the whole real line*, and it can be easily verified that our method converges in polynomial-time over this IVP.[9] Yet Picard's method only guarantees a lifetime of $r \leq 1/2$. Therefore, we will need to analyse the strength of assumption (8) using a different method.

Picard's method may be seen as a competition between the field oracle on the one hand, which may be highly expanding and thus may amplify tiny fluctuations, and the solver on the other hand, which smooths the results over by integration. Bournez et al. [7] have proven polynomial-time complexity of computing the Taylor coefficients of the solution under a cap on the growth of the field, which is called *poly-boundedness*. This condition is implied by p-representability of the field in our setting. Thus, in our future work, we will try to answer the following question:

Problem 1. Assume that the field F is extendable to a p-representable analytic function over a 'suitably large' compact subset of $\mathbb{C}^2$, and the field oracle's estimates are tight enough *over complex numbers*. Will it be true then that condition (8) is always satisfied and hence the IVP can be solved in polynomial-time using our modified method?

6 Summary

We have presented a denotational framework for analysing the semantics of computations over real functions based on polynomial enclosures. We focused on the case of Picard's method of IVP solving. An oracle machine model was considered to separate the actual procedure of IVP solving from the field application. In Theorem 1, we showed that Picard's method converges under an assumption which is equivalent to the classical Lipschitz condition on the field.

In general, IVPs cannot be solved in polynomial-time under the mere Lipschitz continuity condition on the field unless P=PSPACE [16]. On the other hand, from [23, 7] we know that under the stronger analyticity condition on the field, the Taylor coefficients of the solutions can be approximated in polynomial-time. But this does not provide any explicit bounds on the solution.

In contrast, in this paper, we have presented a semantic model which provides explicit error bounds at each stage in the form of polynomial enclosures of the result. In that respect, our model is comparable to the usual domain-models, with the difference that ours makes polynomial-time solution of IVPs possible, as opposed to the step-function models considered in [10, 12].

[8] We do not present a formal proof of this claim here as it can be verified easily with a bit of calculation.

[9] In fact, our implementation of the algorithm successfully converges over non-linear equations such as $y'(t) = \cos(t)y(t), y(0) = 1$ and $y''(t) - 0.2(1 - y(t)^2)y'(t) + y(t) - 0.1\cos(1.1t), y(0) = 1$, the latter being an instance of the forced Van der Pol's equation.

References

1. Aberth, O.: The failure in computable analysis of a classical existence theorem for differential equations. Proceedings of the American Mathematical Society 30(1), 151–156 (September 1971)
2. Abramsky, S., Jung, A.: Domain theory. In: Abramsky, S., Gabbay, D.M., Maibaum, T.S.E. (eds.) Handbook of Logic in Computer Science, vol. 3, pp. 1–168. Clarendon Press, Oxford (1994)
3. Achieser, N.I.: Theory of Approximation. Frederick Ungar Publishing Co. (1956)
4. Bernstein, S.N.: Sur les recherches récentes relatives à la meilleure approximation des fonctions continues par les polynômes. In: Proc. of 5th Inter. Math. Congress. vol. 1, pp. 256–266 (1912)
5. Bernstein, S.N.: Sur l'ordre de la meilleure approximation des fonctions continues par les polynômes de degré donné. Mem. Cl. Sci. Acad. Roy. Belg. 4, 1–103 (1912)
6. Bishop, E.: Foundations of Constructive Analysis. McGraw-Hill (1967)
7. Bournez, O., Graça, D.S., Pouly, A.: Solving analytic differential equations in polynomial time over unbounded domains. In: Proceedings of the 36th international conference on Mathematical foundations of computer science. pp. 170–181. MFCS'11, Springer (2011)
8. Edalat, A.: Dynamical systems, measures and fractals via domain theory. Information and Computation 120(1), 32–48 (July 1995)
9. Edalat, A., Lieutier, A.: Domain theory and differential calculus (functions of one variable). In: Proceedings of 17th Annual IEEE Symposium on Logic in Computer Science (LICS'02). pp. 277–286. Copenhagen, Denmark (2002)
10. Edalat, A., Pattinson, D.: A domain-theoretic account of Picard's theorem. LMS Journal of Computation and Mathematics 10, 83–118 (2007)
11. Escardó, M.H.: PCF extended with real numbers: a domain theoretic approach to higher order exact real number computation. Ph.D. thesis, Imperial College (1997)
12. Farjudian, A., Konečný, M.: Time complexity and convergence analysis of domain theoretic Picard method. In: Hodges, W., de Queiroz, R. (eds.) Proceedings of the 15th international workshop on Logic, Language, Information and Computation, WoLLIC '08, Edinburgh, Scotland. Lecture Notes in Artificial Intelligence, vol. 5110, pp. 149–163. Springer (2008)
13. Gierz, G., Hofmann, K.H., Keimel, K., Lawson, J.D., Mislove, M.W., Scott, D.S.: Continuous Lattices and Domains, Encycloedia of Mathematics and its Applications, vol. 93. Cambridge University Press (2003)
14. Gutknecht, M.H., Trefethen, L.N.: Real polynomial Chebyshev approximation by the Carathéodory-Fejér method. SIAM Journal on Numerical Analysis 19(2), 358–371 (April 1982)
15. Jackson, D.: Über die Genauigkeit der Annäherung stetiger Funktionen durch ganze rationale Funktionen gegebenen Grades und trigonometrische Summen gegebener Ordnung. Ph.D. thesis, Göttingen (1911)
16. Kawamura, A.: Lipschitz continuous ordinary differential equations are polynomial-space complete. In: CCC '09: 24th Annual IEEE Conference on Computational Complexity. pp. 149–160 (2009)
17. Ko, K.I.: Complexity Theory of Real Functions. Birkhäuser, Boston (1991)
18. Konečný, M., Farjudian, A.: Compositional semantics of dataflow networks with query-driven communication of exact values. Journal of Universal Computer Science 16(18), 2629–2656 (2010)
19. Konečný, M., Farjudian, A.: Semantics of query-driven communication of exact values. Journal of Universal Computer Science 16(18), 2597–2628 (2010)

20. Lorentz, G.G.: Approximation of Functions. AMS Chelsea Publishing (1986)
21. Mhaskar, H.N., Pai, D.V.: Fundamentals of Approximation Theory. Narosa (2007)
22. Miller, R., Michel, A.: Ordinary Differential Equations. Academic Press (1982)
23. Müller, N., Moiske, B.: Solving initial value problems in polynomial time. In: Proc. 22 JAIIO - PANEL '93, Part 2. pp. 283–293 (1993)
24. Rivlin, T.J.: Chebyshev Polynomials: from Approximation Theory to Algebra and Number Theory. Wiley, New York, 2nd edn. (1990)
25. Weihrauch, K.: Computable Analysis, An Introduction. Springer (2000)
26. Yoshizawa, T., Hayashi, K.: On the uniqueness of solutions of a system of ordinary differential equations. Memoirs of the College of Science, University of Kyoto. Ser. A, Mathematics 26(1), 19–29 (1950)

A Obtaining Lipschitz oracles from analytic fields

Assume that $F : [0,a] \times [-b,b] \to \mathbb{R}$ is the field of our IVP and assume that F is continuous and Lipschitz (in the classical sense) in its second argument with Lipschitz constant L'. As F is continuous, there exists a sequence $\{P_n \mid n \in \mathbb{N}\}$ of polynomials in $\mathbb{Q}[X_1, X_2]$ such that $\forall n \in \mathbb{N} : F \in <P_n, 2^{-n}>$ and $F = \lim_{n\to\infty} P_n$. Furthermore, if F is analytic, then we can choose the sequence of polynomials in such a way that $D_y(F) = \lim_{n\to\infty} D_y(P_n)$, where $D_y(F)$ is the derivative of F in the direction of the unit vector $(0,1)$. In fact, we can choose $\{P_n \mid n \in \mathbb{N}\}$ in such a way that

$$\forall n \in \mathbb{N} : \| D_y(P_n) \| < L' + 1 \tag{10}$$

If one defines $L \coloneqq L' + 1$, then there is a direct (*yet very inefficient*) way for the oracle $\mathcal{F}$ to generate its output as follows: on each input $(n, <P, R>)$, the oracle $\mathcal{F}$ simply substitutes in $P_n(X_1, X_2)$ the identity polynomial X for X_1, and the polynomial $P(X)$ for X_2, to obtain the univariate polynomial $\tilde{P}_n(X)$. Then it returns the polynomial enclosure $<\tilde{P}_n, 2^{-n} + LR>$. It should be clear that by the assumption of (10) and the fact that each P_n approximates F to within 2^{-n} accuracy, we will have $<\tilde{P}_n, 2^{-n} + LR> \sqsubseteq F(<P, R>)$.

B Proof of Theorem 1 regarding geometric convergence

We break down the proof of Theorem 1 into the following set of propositions:

Proposition 2. *Assume that the field oracle $\mathcal{F}$ is Lipschitz with constant L and let $y : [0,a] \to [-b,b]$ be a solution of the IVP. Then each $<y_n, r_n>$ encloses y.*

Proof. We prove the proposition by induction on n. Obviously $y \in <y_0, r_0>$. Now assume that $n > 0$ and $y \in <y_{n-1}, r_{n-1}>$. Let $[\phi_n, \psi_n] = F(<y_{n-1}, r_{n-1}>)$ and assume that $<\tilde{P}_n, \tilde{R}_n>$ is the answer sent by the field oracle $\mathcal{F}$ in response to the query $(n, <y_{n-1}, r_{n-1}>)$. As $\mathcal{F}$ is assumed to be Lipschitz, $<\tilde{P}_n, \tilde{R}_n> \sqsubseteq [\phi_n, \psi_n]$. By monotonicity of the integral operator and using (5) on page 6 we have $\forall t \in [0,a]$:

$$y_n(t) - r_n(t) \le \int_0^t \phi_n(x)\,\mathrm{d}x \le \int_0^t \psi_n(x)\,\mathrm{d}x \le y_n(t) + r_n(t) \tag{11}$$

On the other hand, $y \in \langle y_{n-1}, r_{n-1} \rangle$ implies $\lambda x.F(x, y(x)) \in [\phi_n, \psi_n]$, and as y is a solution of the IVP:

$$\lambda t.y(t) = \lambda t. \int_0^t F(x, y(x))\,\mathrm{d}x \in [\lambda t. \int_0^t \phi_n(x)\,\mathrm{d}x, \lambda t. \int_0^t \psi_n(x)\,\mathrm{d}x] \tag{12}$$

Combining (11) and (12) completes the proof. □

For each $n \in \mathbb{N}$, we define the n-th degree polynomial

$$e_n(t) = t \sum_{k=0}^{n-1} \epsilon_k t^k + \epsilon_n t^n \tag{13}$$

by assigning $\epsilon_n = L^n b/n!$ and

$$\forall k \in \{0, \ldots, n-1\} : \epsilon_k = \frac{L^k}{2^{n-k}(k+1)!}$$

Proposition 3. $\forall n \in \mathbb{N}, t \in [0, a] : 0 \leq r_n(t) \leq e_n(t)$

Proof. Straightforward induction using (5) on page 6, and using the fact that, as $\mathcal{F}$ is Lipschitz (Definition 3 on page 6), $\forall n \in \mathbb{N}_+, t \in [0, a] : 0 \leq \tilde{R}_n(t) \leq 2^{-n} + L r_{n-1}(t)$. □

Notice that in (13), we have split the coefficients of t^n into $\epsilon_{n-1} + \epsilon_n$. For instance, according to the format in (13) for r_2 we get:

$$0 \leq r_2(t) \leq \frac{1}{2^2 \times 1!}t + \frac{L}{2^1 \times 2!}t^2 + \frac{L^2 b}{2!}t^2$$

Proposition 4. *The maximum value of e_n over $[0, a]$ converges to zero geometrically, i. e. $\exists C > 0, n_0 \in \mathbb{N} : \forall n \geq n_0 : \| e_n \| \leq C/2^n$.*

Proof. First note that all of the coefficients of e_n are non-negative, hence e_n is non-decreasing and as a result it attains its maximum value at $t = a$, at which point according to (13)

$$e_n(a) = a \sum_{k=0}^{n-1} \epsilon_k a^k + \epsilon_n a^n = a \sum_{k=0}^{n-1} \frac{L^k a^k}{2^{n-k}(k+1)!} + \frac{L^n a^n}{n!}$$

Note that $(L^n a^n/n!)$ tends to 0 as $n \to \infty$ at a rate even faster than a geometric one. Moreover:

$$\begin{aligned} a \sum_{k=0}^{n-1} \frac{L^k a^k}{2^{n-k}(k+1)!} = a \sum_{k=0}^{n-1} \frac{(2La)^k}{2^n(k+1)!} &= \frac{1}{2L2^n} \sum_{k=1}^{n} \frac{(2La)^k}{k!} \\ &\leq \frac{1}{2L2^n} \sum_{k=1}^{\infty} \frac{(2La)^k}{k!} \leq \frac{\mathrm{e}^{2La} - 1}{L2^{n+1}} \end{aligned} \tag{14}$$

Hence $\lim_{n\to\infty} e_n(a) = 0$ and the rate of convergence is geometric. □

Corollary 1. *The widths of $\langle y_n, r_n \rangle$ converge to zero geometrically.*

Theorem 1 follows from Proposition 2 and Corollary 1.

C Proof of Theorem 3 regarding polynomial-time complexity

Let us first clarify what we mean by a p-representable field oracle. For this we need to present a method for representing univariate rational polynomials of the form $P(x) = \sum_{i=0}^{n} a_i x^i$. Such a polynomial can be fed into a Turing machine as the following sequence of symbols:

$$\#\#a_0\#a_1\#\ldots\#a_n\#\# \tag{15}$$

where # is used as a delimiter. If the *size of* each finite object x is denoted as $\mathrm{s}(x)$, then according to (15) the size of the polynomial P would be $\mathrm{s}(P) = (n+4)\,\mathrm{s}(\#) + \sum_{i=0}^{n} \mathrm{s}(a_i)$.

A polynomial enclosure $<P, R>$ can also be represented using the representations of P and R and an appropriate delimiter. For instance \$\$$P$\$$R$\$\$, in which case $\mathrm{s}(<P, R>) = \mathrm{s}(P) + \mathrm{s}(R) + 5\,\mathrm{s}(\$)$.

Definition 5 (p-representable). *We say that the field oracle $\mathcal{F}$ is polynomial-space representable,* p-representable *for short, if there exists a polynomial $q \in \mathbb{N}[X]$ such that for each input query $(n, <P, R>)$, the answer polynomial enclosure $<\tilde{P}, \tilde{R}>$ satisfies $\mathrm{s}(<\tilde{P}, \tilde{R}>) \leq q(m)$, where $m = \max\{n, \mathrm{s}(<P, R>)\}$.*

Now, assume that the field oracle $\mathcal{F}$ is p-representable. To prove polynomial complexity, we need to analyse the following contributing factors:

Communication of data: As we are capping the sizes of each y_n and r_n at stage n to at most $O(n^2)$, then the volume of data communicated between the solver on the one hand, and the field oracle and the outside world on the other hand remains within the polynomial bound. It is in fact very easy to see that this volume at each stage n is $O(n^{2\deg(q)+1})$, where q is the bounding polynomial as in Definition 5 above.

Chebyshev truncations: Note that the polynomial q puts a bound on the sizes of the answers of the field oracle $\mathcal{F}$, i. e. not just on the degree of the answers, but also on the sizes of the coefficients of the answer polynomials. Now assume that at stage n, the matrix S_n (as in (6) on page 7) is used to convert the representation of $\tilde{P}_n$ from monomial to Chebyshev basis, and assume that S_n has got dimension $\upsilon_n \times \upsilon_n$. Then obviously we have $\upsilon_n \in O(n^{2\deg(q)})$, and for each entry $s_{i,j}$ of S_n, we have $\mathrm{s}(s_{i,j}) \in O(n^{2\deg(q)})$. As S_n is upper triangular with all its diagonal entries non-zero, then at stage n the equation $S_n \times B = A$ can be solved with at most $O(\upsilon_n^2)$ arithmetic operations. The same argument holds for the Chebyshev truncation of $\tilde{R}_n$.

Dyadic approximations: In Section 5 we discussed the approximation of each of the coefficients a_i (for $0 \leq i \leq n$) of $\hat{y}_n$ by $\mathring{a}_i$ to obtain $\tilde{y}_n$. Let $i \leq n$ and for simplicity assume that $a_i \geq 0$, and let (the rational number) a_i be written as m_i/n_i. We want to find the biggest $p_i \in \mathbb{N}$ which is not bigger than $2^n m_i/n_i$. The easiest way to do this is by a simple integer division. As the sizes of the numerator and the denominator of $2^n m_i/n_i$ are at most $O(n^{2\deg(q)+1})$, then the division can be carried out in at most $O(n^{4\deg(q)+2})$ steps. This gives us p_i, and as a consequence, the required dyadic number $p_i/2^n$.

Therefore, as there are $n+1$ coefficients, the approximation of $\hat{y}_n$ by $\tilde{y}_n$ can be carried out in $O(n^{4\deg(q)+3})$. A similar argument holds for the approximation of $\hat{r}_n$ by $\tilde{r}_n$.

This completes the proof of Theorem 3. □