Computational Complexity of Real Powering and Improved Solving Linear Differential Equations

Ivan Koswara, Svetlana Selivanova(✉), and Martin Ziegler

School of Computing, KAIST, Daejeon, Republic of Korea
sseliv@kaist.ac.kr

Abstract. We re-consider the problem of solving systems of differential equations approximately up to guaranteed absolute error $1/2^n$ from the rigorous perspective of sequential and parallel time (i.e. Boolean circuit depth, equivalently: Turing machine space) complexity. While solutions to general smooth ODEs are known "PSPACE-complete" [Kawamura'10], we show that (i) The Cauchy problem for linear ODEs can be solved in NC^2, that is, within polylogarithmic parallel time $\mathcal{O}(\log^2 n)$ by Boolean circuits of polynomial size. (ii) The Cauchy problem for linear analytic PDEs, having a unique solution by the Cauchy-Kovalevskaya theorem, can be also solved in polylogarithmic parallel time, thus generalizing the case of analytic ODEs [Bournez/Graça/Pouly'11]. (iii) Well-posed Cauchy and boundary-value problems for linear PDEs in classes of continuously differentiable functions are solvable in the counting complexity class $\#\mathsf{P}^{\#\mathsf{P}}$: improving over common numerical approaches yielding exponential sequential time or parallel polynomial time. Our results build on efficient algorithms and their analyses for real polynomial, matrix and operator powering which do not occur in the discrete case and may be of independent interest.

1 Introduction and Brief Summary of Main Results

Recursive Analysis provides a rigorous algorithmic foundation to Numerics, that is, to operations on continuous data by means of approximations [18]. It has a long history of thorough investigations regarding ordinary (ODEs) and partial differential equations (PDEs) with respect to computability and real complexity. For example, it has been established that (a) ODEs with a polynomial/analytic right-hand side can be solved in (appropriately parameterized) sequential polynomial time [3]; while (b) general non-linear ODEs are optimally solved by Euler's Method using an amount of memory polynomial in the output precision

Supported by the National Research Foundation of Korea (grant NRF-2017R1E1A1A 03071032) and by the Korean Ministry of Science and ICT (grant NRF-2016K1A3A 7A03950702). Dedicated to the memory of Ker-I Ko. We thank Gleb Pogudin for seminal discussions that led to Theorem 7.

R. van Bevern and G. Kucherov (Eds.): CSR 2019, LNCS 11532, pp. 215–227, 2019.
https://doi.org/10.1007/978-3-030-19955-5_19

parameter n [9], equivalently: in polynomial parallel time [2]; and (c) solving Poisson's linear PDEs corresponds to the complexity class $\#\mathsf{P}$ [11].

The main definitions and notions of real complexity theory (the framework in which we work) are summarized in Sect. 2. The discrete complexity hierarchy[1] [17, Corollary 2.34] translates to the real setting (see Definition 4) as

$$\begin{gathered}\mathbb{R}\mathsf{NC}^1 \subseteq \mathbb{R}\mathsf{SPACE}(\log n) \subseteq \mathbb{R}\mathsf{NC}^2 \subseteq \mathbb{R}\mathsf{PTIMESPACE}(\log^2 n) \subseteq \mathbb{R}\mathsf{NC}^4 \\ \text{and more generally } \mathbb{R}\mathsf{NC}^i \subseteq \mathbb{R}\mathsf{PTIMESPACE}(\log^i n) \subseteq \mathbb{R}\mathsf{NC}^{2i} \ldots \subseteq \\ \mathbb{R}\mathsf{PTIME} \subseteq \mathbb{R}\#\mathsf{P} \subseteq \mathbb{R}\#\mathsf{P}^{\#\mathsf{P}} \subseteq \mathbb{R}\mathsf{PSPACE} = \mathbb{R}\mathsf{PAR} \subseteq \mathbb{R}\mathsf{EXP}\ .\end{gathered} \tag{1}$$

The present work continues the above complexity investigations for linear ODEs and PDEs with the following contributions (see rigorous formulations later in this section and ideas of proofs in Sect. 4):

(a) Theorem 1 establishes linear systems of ODEs computable in polylogarithmic parallel time (=depth), specifically in $\mathbb{R}\mathsf{NC}^2$, the real counterpart to NC^2.

(b) Also linear *analytic* PDEs can be solved in polylogarithmic parallel time according to Theorem 3, generalizing the case of analytic ODEs.

(c) By Theorem 2, a large class of continuously differentiable linear PDEs with boundary conditions, classically treated with numerical difference schemes, can be solved with computational complexity $\mathbb{R}\mathsf{PSPACE}$;

(d) For many cases with periodic boundary conditions this can be improved to $\mathbb{R}\#\mathsf{P}^{\#\mathsf{P}}$ (probably even $\mathbb{R}\#\mathsf{P}$, which will be a subject of future investigation).

In all these cases, the output consists of (approximations up to error 2^{-n} to) the real value $u(t)$ or $u(t,x)$ of the solution at given time/space t and x. Our results are obtained by applying matrix/operator exponentials, known but rarely used in classical Numerics. They in turn rely on efficient recursive algorithms for powering polynomials, matrices and operators developed in this paper (see Sect. 3), which may be of independent interest:

(e) Given $A \in [-1;1]^{\mathrm{poly}(n)\times\mathrm{poly}(n)}$ with bounded powers $\|A^k\| \leq 1$, its power $A^{\mathrm{poly}(n)}$ can be computed in polylogarithmic depth $\mathbb{R}\mathsf{NC}$; see Proposition 6(c). This result is used to prove Theorem 1.

(f) If the entries of $A \in [-1;1]^{2^n\times 2^n}$ are computable in polynomial time, then the entries of A^{2^n} are computable in $\mathbb{R}\mathsf{PSPACE}$, and this is optimal in general; see Proposition 6(g). This result is used to prove Theorem 2, general case of a difference scheme approximating the considered system of PDEs.

(g) For circulant matrices A of constant bandwidth with polynomial-time computable entries and bounded powers this result can be improved: the entries of the matrix power A^{2^n} are computable in $\mathbb{R}\#\mathsf{P}$, see Theorem 7, where also powering of polynomials as an important auxiliary tool is considered. These results are used to prove Theorem 2, particular case of a difference scheme.

[1] Rigorously speaking, $\#\mathsf{P}$ is a class of integer functions rather than decision problems and should here be read as $\mathsf{PTIME}^{\#\mathsf{P}}$.

(h) Theorem 8 and Example 9 establish complexity of powering and exponentiation of (e.g. differential) operators in Banach spaces. These results are used to prove Theorem 3.

Note that the hypothesis on uniformly bounded powers of the matrices in (e) and (g) makes the problems trivial over integers[2], yet interesting and new in the real case under consideration. The matrices corresponding to convergent difference schemes for PDEs meet this hypothesis because of the stability property required for a difference scheme to be convergent. Our work can be regarded as instances and confirmation of [1].

When translating the classical discrete parallel/depth complexity to the real setting, we restrict to 2^ℓ-Lipschitz functions f. A family C_n of Boolean circuits is required to approximate $y = f(x)$ up to error 2^{-n} for every $x \in \mathrm{dom}(f)$, the latter given by approximations up to error $2^{-\ell-n}$. Space complexity of f is formalized as the number of working tape cells used by a Turing machine, not counting input nor output tape, to similarly approximate $y = f(x)$ up to error 2^{-n}: see Sect. 2 for details. We thus follow the general paradigm of measuring computational cost (depth=parallel/sequential time, memory) over the reals, other than in the discrete setting with finite inputs, in dependence on n reflecting roughly the number of correct bits of the output approximation attained in the worst-case over all (uncountably many) continuous arguments. 'Binary' error bounds 2^{-n}, rather than 'unary' $1/n$, capture roughly n correct binary digits and yield stronger statements.

We consider Cauchy (i.e. initial-value) problems for autonomous linear evolutionary differential equations in a general form

$$\tfrac{\partial}{\partial t}\vec{u}(t) \;=\; A\vec{u}(t), \quad t \in [0;1], \quad \vec{u}(0) = \vec{\varphi}. \tag{2}$$

Here A is a matrix in the case of ODEs and a more general operator (including partial differentiation for PDEs) for other cases. Similarly, $\vec{\varphi}$ is an initial real-valued vector for ODEs and an initial function in a more general setting. Let us now formally state our main contributions regarding differential equations:

Theorem 1. *Given $A \in [-1;1]^{d\times d}$ and $\vec{v} \in [-1;1]^d$ and $t \in [0;1]$, the solution*

$$\vec{u}(t) = \exp(tA)\vec{v} := \sum_k \frac{t^k}{k!} A^k \vec{v} \tag{3}$$

to the system of linear ordinary differential Eq. (2) is computable by Boolean circuits of depth $\mathcal{O}\big((\log d + \log n)^2\big)$ and size polynomial in $d + n$.

Recall that circuit depth is synonymous for parallel time. Theorem 1 thus formally captures the intuition that solving ODEs in the linear case is easier than in the analytic case [3], and much easier than in the general C^1 smooth case proven "PSPACE-complete" [9]. It is no loss of generality to impose unit bounds on $A, \vec{v}, t$: the general case is covered by rescaling.

[2] Cmp. Geoff Robinson's answer on https://math.stackexchange.com/q/59693.

Our next result is concerned with finitely-often differentiable solutions to Eq. (2) with $A = \sum_{|\alpha|\leq k} b_\alpha(x) D_x^\alpha$ and $\vec{\varphi} = \vec{\varphi}(x)$. They can be reduced [4], by adding extra variables, to first-order linear systems of PDEs with

$$A = \sum_{j=1}^{m} B_j(x) \frac{\partial}{\partial x_j}, \tag{4}$$

where $B_j(x)$ are matrices of a suitable dimension.

Theorem 2. *For $m \in \mathbb{N}$ and a convex open bounded $\Omega \subseteq \mathbb{R}^m$ consider the initial-value problem (IVP) (2) with the operator (4), and the boundary-value problem (BVP) with additionally given linear boundary conditions $\mathcal{L}u(t,x)|_{\partial\Omega} = 0, (t,x) \in [0,1] \times \partial\Omega$. Suppose the given IVP and BVP be* well posed *in that the classical solution $\vec{u} : [0;1] \times \overline{\Omega} \to \mathbb{R}$ (i) exists, (ii) is unique, and (iii) depends continuously on the data $\varphi(x)$ and $B_i(x)$ (for the BVP also on coefficients of $\mathcal{L}$). More precisely we assume that $u(t,x) \in C^2$ and its C^2-norm is bounded linearly by C^2-norms of the data (in functional spaces guaranteeing all the required properties). Moreover suppose that the given IVP and BVP admit a (iv) stable and (v) approximating with at least the first order of accuracy* difference scheme *(A_n) in the sense of Definition 10.*

(a) If the difference scheme (meaning its matrix) A_n and the initial condition φ are (vi) computable in depth $s(n) \geq \log(n)$ in the sense of Definition 5, then evaluating the solutions $u : [0;1] \times \overline{\Omega} \ni (t,x) \mapsto u(t,x)$ of both IVP and BVP is feasible in depth $\mathcal{O}\big(s(2^n) + n \cdot \log n\big)$.

(b) If A_n and φ are (vi') computable in polynomial sequential time and A_n is additionally circulant of constant bandwidth, then the solution function u belongs to the real complexity class $\mathbb{R}\#\mathsf{P}^{\#\mathsf{P}}$.

This second result establishes polynomial parallel time (equivalently: polynomial space or depth) complexity for the considered PDEs in the binary output precision parameter n and further down to the second level of the Counting Hierarchy. As a main tool we modify the classical difference schemes approach: standard step-by-step iterations would yield only exponential sequential time; we replace them with efficient matrix powering according to Proposition 6 and Theorem 7 below. It complements work like [14] measuring bit-cost in dependence on $N = 2^{\mathcal{O}(n)}$, the size of the grid under consideration, and implicitly supposing the difference scheme and initial data computable in logarithmic depth $s(n) = \mathcal{O}(\log n)$: where we consider the output precision parameter n, and allow for more involved difference schemes with $s(n) = \mathcal{O}(\log^2 n)$, say. Theorem 2 also complements rigorous cost analyses considering approximations up to *output* error $1/n$, or up to fixed error and in dependence on the length of the (algebraic) *input* [15,16]; and it vastly generalizes previous works on the computational complexity of Poisson's PDE [11].

Between linear ODEs and finitely-often continuously differentiable PDEs are analytic PDEs, captured by the Kovalevskaya Theorem; and their computational complexity also turns out to lie between the aforementioned two:

Theorem 3 (Polynomial-Time/Polylogarithmic-Space Kovalevskaya). *Let $f_1, \ldots, f_e : [-1;1]^d \to \mathbb{R}^{e\times e}$ and $v : [-1;1]^d \to \mathbb{R}^e$ denote real functions analytic on some open neighborhood of $[-1;1]^d$ and consider the system of linear partial differential equations*

$$\partial_t \vec{u}(\vec{x},t) = f_1(\vec{x})\partial_1 \vec{u} \ + \cdots + \ f_e(\vec{x})\partial_e \vec{u} \qquad \vec{u}(\vec{x},0) \equiv v \ . \tag{5}$$

If $f_1, \ldots f_e$ are computable in sequential polynomial time, then the unique real analytic local solution $\vec{u} : [-\varepsilon; +\varepsilon]^{d+1} \ni (\vec{x},t) \mapsto \vec{u}(\vec{x},t) \in \mathbb{R}^e$ to Eq. (5) is again computable in sequential polynomial time.
If $f_1, \ldots f_e$ are computable in polylogarithmic depth, then so is $\vec{u}$.

We emphasize that the constructive proof of Kovalevskaya's Theorem [4, §4.6.3] expresses the solution's j-th coefficient as a multivariate integer polynomial p_j in the initial condition's and right-hand side's coefficients; however as p_j's total degree and number of variables grows with j, its number of terms explodes exponentially. Symbolic-numerical approaches employ Janet, Riquier or Gröbner bases whose worst-case complexity however is also exponential [13].

2 Real Complexity Theory

We consider the computational worst-case cost of computing continuous real functions on a compact domain, formalized as follows:

Definition 4. *Equip $\mathbb{R}^d$ with the maximum norm and fix 2^ℓ-Lipschitz $f : \Omega \subseteq [-2^k; 2^k]^d \to [-2^m; 2^m]$, $k, \ell, m \in \mathbb{N}$.*

(a) Consider a Turing Machine $\mathcal{M}$ with read-only *input tape,* one-way *output tape, and working tape(s). $\mathcal{M}$ is said to* compute *f (w.r.t. μ) if, given $\mathsf{0}^n\mathsf{1}\,\mathrm{bin}(\vec{a}) \in \{\mathsf{0},\mathsf{1}\}^{n+1+d\cdot\mathcal{O}(k+\ell+n)}$ for $\vec{a} \in \{-2^{k+\ell+n}, \ldots, 0, \ldots + 2^{k+\ell+n}\}^d$ with $|\vec{x} - \vec{a}/2^{\ell+n}| \le 2^{-\ell-n}$ for some $\vec{x} \in \Omega$, $\mathcal{M}$ outputs $\mathrm{bin}(b)$ for some $b \in \{-2^{m+n}, \ldots, 0, \ldots, 2^{m+n}\}$ with $|f(\vec{x}) - b/2^n| \le 2^{-n}$ and stops. Here $\mathrm{bin}(\vec{a}) \in \{\mathsf{0},\mathsf{1}\}^*$ denotes some binary encoding of integer vectors.*

(b) Fix $s, t : \mathbb{N} \to \mathbb{N}$ with $t(n) \ge n$ and $s(n) \ge \log_2(n)$. The computation from (a) runs in time *t and* space *s if $\mathcal{M}$ stops after at most $t\big(n + m + d \cdot \mathcal{O}(n + k + \ell)\big)$ steps and uses at most $s\big(n + m + d \cdot \mathcal{O}(n + k + \ell)\big)$ cells on its work tape, not counting input nor output tape usage and regardless of $\vec{a}$ as above. In this case write $f \in \mathbb{R}\mathsf{TIME}(t) \cap \mathbb{R}\mathsf{SPACE}(s)$. Polynomial time is abbreviated $\mathbb{R}\mathsf{PTIME} = \bigcup_i \mathbb{R}\mathsf{TIME}\big(\mathcal{O}(n^i)\big)$, polynomial space means $\mathbb{R}\mathsf{PSPACE} = \bigcup_i \mathbb{R}\mathsf{SPACE}\big(\mathcal{O}(n^i)\big)$, and $\mathbb{R}\mathsf{PTIMESPACE}(\log^i n) := \mathbb{R}\mathsf{PTIME} \cap \mathbb{R}\mathsf{SPACE}(\log^i n)$.*

(c) Consider a Boolean circuit C_n having $\mathcal{O}(n + m)$ binary outputs and $d \cdot \mathcal{O}(k + \ell + n)$ binary inputs. Such a sequence (C_n) computes *f if C_n, on every (possibly padded) input $\mathrm{bin}(\vec{a})$ with $|\vec{x} - \vec{a}/2^{n+\ell}| \le 2^{-k-\ell}$ for some $\vec{x} \in \Omega$, it outputs some $\mathrm{bin}(b)$ such that $|f(\vec{x}) - b/2^n| \le 2^{-n}$.*

(d) *We say that f is computable in* depth t*, written $f \in \mathbb{R}\mathsf{DEPTH}(t)$, if there exists a sequence (C_n) of Boolean circuits over basis binary NAND, say, of depth at most $t\big(n + m + d \cdot \mathcal{O}(n + k + \ell)\big)$ computing f. $\mathbb{R}\mathsf{NC}^i$ abbreviates $\mathbb{R}\mathsf{DEPTH}\big(\mathcal{O}(\log^i n)\big)$ with the additional requirements that (i) the circuits be logspace uniform and (ii) their size (#gates) grows at most polynomially in $n+m+d\cdot\mathcal{O}(n+k+\ell)$. Similarly, $\mathbb{R}\mathsf{PAR}$ abbreviates $\bigcup_i \mathbb{R}\mathsf{DEPTH}\big(\mathcal{O}(n^i)\big)$ with the additional requirement that the (possibly exponentially large) circuits C_n be polynomial-time uniform in that a polynomial-time Turing machine can, given $n \in \mathbb{N}$ in unary and $I < J \in \mathbb{N}$ in binary with respect to some fixed topological order, report whether in C_n the output of gate #I is connected to gate #J.*

(e) *We say f belongs to $\mathbb{R}\#\mathsf{P}$ if it can be computed by a Turing machine $\mathcal{M}^\varphi$ in polynomial time (b) given oracle access to some counting problem φ in the discrete complexity class $\#\mathsf{P}$.*
Similarly for $\mathbb{R}\#\mathsf{P}^{\#\mathsf{P}}$.

We follow the classical conception of real numbers as 'streams' of approximations, both for input and output [6]: the alternative approach based on oracles [8,12] involves a *stack* of query tapes to ensure closure under composition.

Common numerical difference schemes are matrices A_m whose dimension D_m grows exponentially with the precision parameter m and therefore cannot be output entirely within polynomial time; instead Definition 5 requires any desired entry $(A_m)_{I,J}$ to admit efficient approximations, where indices I, J are given in binary such that their length remains polynomial in m:

Definition 5. *(a)* Computing *a vector $\vec{v} \in \mathbb{R}^D$ means to output, given $J \in \{0, \ldots D-1\}$ in binary and $n \in \mathbb{N}$ in unary, some $a \in \mathbb{Z}$ in binary such that $|v_J - a/2^n| \le 2^{-n}$; similarly for matrices $A \in \mathbb{R}^{D\times D}$, considered as vectors in $\mathbb{R}^{\mathcal{O}(D\cdot D)}$ via the pairing function $(I,J) \mapsto (I+J)\cdot(I+J+1)/2+J$.*

(b) *For in both arguments monotonically non-decreasing $t(n,m)$ and $s(n,m) \le t(n,m)$, a sequence $\vec{v}_m \in \mathbb{R}^{D_m}$ of vectors is computed in sequential time t and space s if its entries have binary length at most polynomial in its dimension $\sup_J |v_{m,J}| \le 2^{\mathrm{poly}(D_m)}$ and it takes a Turing machine at most $t(n,m)$ steps and $s(n,m)$ tape cells to output, given $J \in \{0, \ldots D_m - 1\}$ in binary and $m \in \mathbb{N}$ in unary, some $a \in \mathbb{Z}$ in binary with $|v_{m,J} - a/2^n| \le 2^{-n}$; similarly for sequences of matrices $A_m \in \mathbb{R}^{D_m \times D_m}$.*

(c) *Polynomial sequential time/space and poly/logarithmic space means polynomial and poly/logarithmic in $n+m$, respectively. $\mathbb{R}\#\mathsf{P}$ for (sequences of) vectors consists of those $\vec{v}_m \in \mathbb{R}^{D_m}$ computable in polynomial time by a Turing machine with a $\#\mathsf{P}$ oracle; similarly for (sequences of) matrices.*

(d) *A sequence $\vec{v}_m \in \mathbb{R}^{D_m}$ of vectors is computed in* depth $s(n,m)$ *if its entries have binary length at most polynomial in D_m: $\sup_J |v_{m,J}| \le 2^{\mathrm{poly}(D_m)}$, and a family of Boolean circuits $C_{n,m}$ of depth $s(n,m)$ can output, given $J \in \{0, \ldots D_m - 1\}$ in binary, some $a \in \mathbb{Z}$ in binary with $|v_{m,J} - a/2^n| \le 2^{-n}$.*

(e) *Polylogarithmic depth means Boolean circuits $C_{n,m}$ in (d) of depth polylogarithmic in $n+m$; $\mathbb{R}\mathsf{NC}^i$ abbreviates $\mathbb{R}\mathsf{DEPTH}\big(\mathcal{O}(\log^i n)\big)$ with additional requirements (i) and (ii) as in Definition 4(f); similarly for $\mathbb{R}\mathsf{PAR}$.*

Note that already reading J in (b) takes time of order $\log D_m \leq t(n,m)$, and space of order $s(n,m) \geq \operatorname{loglog} D_m$. Similarly, a circuit of depth $s(n,m)$ can access at most $2^{s(n,m)} \geq \log D_m$ input gates.

3 Efficient Real Polynomial/Matrix/Operator Powering

A major ingredient to our contributions are new efficient algorithms for real polynomial/matrix/operator powering, analyzed in dependence on various parameters that do not occur/make sense in the classical discrete setting:

Proposition 6. *(a) For $D, K \in \mathbb{N}$ and $A, A', B, B' \in [-2^L; 2^L]^{D\times D}$, it holds $|A^K|_\infty \leq 2^{KL} \cdot D^{K-1}$ and $|A \cdot B - A' \cdot B'|_\infty \leq D \cdot 2^{L+1} \cdot \max\big\{|A - A'|_\infty, |B - B'|_\infty\big\}$, where $|B|_\infty := \max_{I,J} |B_{I,J}|$.*

(b) For $D, K \in \mathbb{N}$, matrix powering $[-2^L; 2^L]^{D\times D} \ni A \mapsto A^K$ is computable by circuits C_n of depth $\mathcal{O}\big(\log(K) \cdot (\log n + \log D + \log K + \log L)\big)$ and size polynomial in $n + D + K + L$.

(c) Refining (b), if $|A^k|_\infty \leq 2^L$ holds for all $k \leq K$, the depth of circuits computing $\mathbb{R}^{D\times D} \ni A \mapsto A^K$ can be reduced to $\mathcal{O}\big((\log n + \log D + \mathbf{loglog} K + \log L) \cdot \log K\big)$.

(d) Suppose sequences $\vec{u}_m, \vec{v}_m \in \big[-2^{L_m}; +2^{L_m}\big]^{D_m}$ are computable in depth $s(n,m) \geq \log(n) + \log(m)$ and have inner product $\vec{u}_m^\perp \cdot \vec{v}_m \in \big[-2^{L_m}; +2^{L_m}\big]$. Then said inner product is computable in depth $\mathcal{O}\big(\log(D_m) + s(n + L_m + \log D_m), m\big)$.

(e) Suppose sequences $\vec{v}_m, \vec{w}_m$ are computable in polynomial sequential time. Then their inner product $\vec{u}_m^\perp \cdot \vec{v}_m \in \mathbb{R}$ is in $\mathbb{R}\#\mathsf{P}$; and this is optimal in general.

(f) Suppose $A_m \in \mathbb{R}^{D_m \times D_m}$ are computable in depth $s(n,m) \geq \log(n) + \log(m)$ and satisfy $A_m^k \in \big[-2^{L_m}; 2^{L_m}\big]^{D_m\times D_m}$ for all $k \leq K_m$. Then the powers $A_m^{K_m} \in \mathbb{R}^{D_m\times D_m}$ are computable in depth

$$\mathcal{O}\Big(\log(K_m) \cdot (\log n + \log L_m + \log D_m + \operatorname{loglog} K_m) \\ + s\big(n + (L_m + \log D_m) \cdot \log K_m, m\big)\Big)$$

(g) If A_m is computable in polynomial time and same for $\mathbb{N} \ni K_m < 2^{\mathrm{poly}(m)}$, then the powers $A_m^{K_m}$ are computable in $\mathbb{R}\mathsf{PSPACE}$; and this is optimal in general.

Proof (Sketch). Claims (b), (c) and (f) are based on repeated squaring, each matrix multiplication being D^2 inner products, realized as prefix sums (carry look-ahead) and known logarithmic-time integer multiplication (d). Regarding (g), encode PSPACE-complete reachability as matrix powering. □

Theorem 7. *For a uni-variate polynomial $p = \sum_{j=0}^d p_j X^j \in \mathbb{R}[X]$, let $|p| := \sum_j |p_j|$ denote its norm. Fix $d \in \mathbb{N}$.*

(a) *Given* $a, b \in \mathbb{R}$ *with* $|a| + |b| \leq 1$ *and* $J \leq K \in \mathbb{N}$, $\binom{K}{J} \cdot a^J \cdot b^{K-J}$ *can be approximated to* 2^{-n} *in time polynomial in* n *and the binary length of* K.

(b) *Given* $a_1, \ldots, a_d \in \mathbb{R}$ *with* $\sum_j |a_j| \leq 1$ *and given* $J_1, \ldots J_d, K \in \mathbb{N}$ *with* $\sum_j J_j = K$, $\binom{K}{J_1, \ldots J_d} \cdot a_1^{J_1} \cdots a_d^{J_d}$ *can be approximated up to error* 2^{-n} *in time polynomial in* n *and the binary length of* K. *Here* $\binom{K}{J_1, \ldots J_d} = \frac{K!}{J_1! \cdots J_d!}$ *denotes the* multi*nomial coefficient.*

(c) *For* $p = \sum_{j=0}^{d} p_j X^j \in \mathbb{R}[X]$ *with* $|p| \leq 1$ *and polynomial-time computable coefficients, the coefficient vector*

$$\sum_{\substack{J_0+J_1+\ldots+J_d=2^n \\ J_1+2J_2+\cdots+dJ_d=J}} p_0^{J_0} \cdots p_d^{J_d} \cdot \binom{2^n}{J_0, \ldots J_d}, \qquad J \leq d \cdot 2^n \tag{6}$$

of p^{2^n} *belongs to* $\mathbb{R}\#\mathsf{P}$.

(d) *Let* $C_m \in \mathbb{R}^{2^m \times 2^m}$ *denote a circulant matrix of bandwidth* d *with polynomial-time computable entries* $c_{-d}, \ldots c_{+d} \in \mathbb{R}$ *such that* $|p_{-d}| + \ldots + |p_{+d}| \leq 1$. *Then the matrix power* $C_m^{2^m}$ *belongs to* $\mathbb{R}\#\mathsf{P}$.

We omit the proof of Theorem 7 because of space constraints. Item (d) is proved by means of Items (a)–(c). Items (a) and (c) use the Gaussian Distribution as approximation. Note that, again, Items (c) and (d) are trivial over integers; considering the real setting makes them meaningful and crucial for cost analyses of difference schemes.

Our next tool is about efficient operator powering. It generalizes Proposition 6(b) to compact subsets of some infinite-dimensional vector space.

Theorem 8. *Fix a Banach space* $\mathcal{B}$ *with norm* $\|\cdot\|$ *and linear map* $A : \mathcal{B} \to \mathcal{B}$. *And fix an increasing sequence* $\mathcal{V}_d \subseteq \mathcal{V}_{d+1} \subseteq \mathcal{B}$ *of non-empty compact convex symmetric subsets such that*

(i) $A^K : \mathcal{V}_d \to \mathcal{V}_{\mathrm{poly}(d+K)}$ *is well-defined for all* $d, K \in \mathbb{N}$ *and*
(ii) *satisfies* $\|A^K v\| \leq \mathcal{O}(1)^d \cdot d^K \cdot K!$ *for all* $v \in \mathcal{V}_d$
(iii) *and is computable in sequential time polynomial in* $n + d + K$
(iii') *or is computable in polylogarithmic depth* $\mathrm{poly}(\log n + \log d + \log K)$.

$$u(t) \;:=\; \exp(tA)v \;=\; \sum\nolimits_K t^K \cdot A^K v / K! \;\in\; \mathcal{B}$$

is well-defined for all $\vec{v} \in \mathcal{V}_d$ *and all* $|t| < 1/d$ *and satisfies* $u_t = Au$. *Moreover* $\mathcal{V}_d \times [0; 1/2d] \ni (v, t) \mapsto u(t)$ *is computable in sequential time polynomial in* $n+d$ *(iii) or (iii') in depth* $\mathrm{poly}(\log n + \log d)$.

Proof (Sketch). Under hypothesis (ii), the series $u(t) = \sum_K t^K \cdot A^K v / K!$ permits differentiation under the sum for $|t| \leq 1/2d$ and hence solves $u_t = Au$. Moreover the first n terms of the series approximate u up to error 2^{-n}. □

Notice that a naïve hypothesis (i') $A : \mathcal{V}_d \to \mathcal{V}_{2d}$ would only imply $A^K : \mathcal{V}_d \to \mathcal{V}_{2^K d}$: blowing up exponentially in K. The stronger hypothesis (i), as well as (ii) to (iii'), are for instance satisfied for $\mathcal{V}_d := [-2^d; 2^d]^D$ by every $A \in [-2^d; 2^d]^{D\times D}$; recall Proposition 6(a). A more involved case revolves around univariate analytic functions.

Example 9. Consider the space $\mathcal{B}$ of complex-valued functions $v : [-1; 1] \to \mathbb{C}$ infinitely often continuously differentiable on the real interval. Write $v^{(j)}$ for its j-th iterated derivative, and abbreviate $|v|_\infty := \sup_x |v(x)|$. For each $d \in \mathbb{N}$ let

$$\mathcal{V}_d \; = \; \big\{ v : [-1; 1] \to \mathbb{C}, \;\; \forall j \in \mathbb{N} \; |v^{(j)}|_\infty \le 2^d \cdot j! \cdot d^j \big\}, \quad \|v\| := \sum\nolimits_j |v^{(j)}|_\infty / (j!)^2$$

Represent each $v \in \mathcal{V}_d$ by the $(2d+1)$-tuple of its germs around positions 1, $(d-1)/d, \ldots, (-d+1)/d, 1 \in [-1; 1]$; and represent each germ by its power series coefficient sequence.

(i) $\partial^K : \mathcal{V}_d \ni v \mapsto v^{(K)} \in \mathcal{V}_{3d+\lceil K \log d\rceil + \lceil K \log K\rceil}$ is well-defined.
(ii) satisfies $\|\partial^K v\| \le (2e^2)^d \cdot d^K \cdot K!$ for all $v \in \mathcal{V}_d$
(iii) $\partial^K\big|_{\mathcal{V}_d}$ is computable in sequential time polynomial in $n + d + K$
(iii') and in polylogarithmic depth $\mathrm{poly}(\log n + \log d + \log K)$.
(iv) $\mathcal{V}_d$ is compact.

Proof (Sketch). For (i) and (iii) see the proof of Theorem 16(d) in [7, §3.2]. The underlying algorithm basically shifts and scales the coefficient sequences to symbolically take the derivative of the power series: easy to parallelize [19]. It also needs to add new germs/points of expansion in $[-1; 1]$: which again can be performed in parallel, thus establishing (iii'). □

4 Complexity of Differential Equations

In this section we apply results on matrix/operator powering from the previous section to prove Theorems 1–3.

Proof (Theorem 1, sketch). Recall from Proposition 6(a) that A^k has entries bounded by d^{k-1}. Hence the tail $\sum_{k>K} \frac{t^k}{k!} A^k \vec{v}$ is bounded, according to Stirling formula, by

$$\sum\nolimits_{k>K} 1/k! \cdot 2^{k\cdot \log d} \;\le\; \sum\nolimits_{k>K} \mathcal{O}\big(2^{-k\cdot(\log k - \log d)}\big) \;\le\; \sum\nolimits_{k>K} 2^{-k} \;=\; 2^{-K}$$

for $K \ge 2d$. Thus we can calculate the first $K := \max\{2d, n\}$ terms of the power series in Eq. (3) simultaneously within depth $\mathcal{O}\big(\log(\max\{n, 2d\}) \cdot (\log n + \log d + \log\log\max\{n, 2d\})\big) = \mathcal{O}\big((\log n + \log d)^2\big)$ (by Proposition 6(c)); and then add them, incurring additional depth of the same magnitude, which completes the proof of the theorem. □

Proof (Theorem 3, Sketch). Example 9 generalizes to functions of several variables [10]. The statements of the theorem thus follow from Theorem 8 with $A := f_1\partial_1 + \cdots + f_e\partial_e$. □

Before proceeding to the proof of Theorem 2 let us briefly recall basic definitions and facts about difference schemes. Theorem 2 implicitly assumes the domain Ω be "good enough"; w.l.o.g. we will consider uniform grids G_h on $\bar{\Omega}$ and G_h^τ on $[0,1] \times \bar{\Omega}$ for BVPs (resp. on the intersection of the domain of existence and uniqueness with $[0,1] \times \bar{\Omega}$ for IVPs). Here h, τ are respectively the space and time steps. Unlikely choosing them heuristically as it is usually done in Numerics, we will compute them from the output precision as $h = C_h/2^n$, $\tau = C_\tau/2^n$, giving precise estimates on C_h, C_τ. That's the reason why we denote the matrix of the difference scheme A_n (depending on n) in Definition 10 below.

Definition 10. *(a) For the IVP or BVP for a system of PDEs considered in Theorem 2, an* explicit difference scheme *is a system of algebraic equations*

$$u^{(h,(l+1)\tau)} = \mathbf{A_n} u^{(h,l\tau)}, \quad u^{(h,0)} = \varphi^{(h)} \tag{7}$$

Here A_n is a difference operator, which is in our case linear, i.e. a matrix of dimension $\mathcal{O}(2^n)$ (for BVPs it also includes the boundary conditions); $u^{(h,l\tau)}$ and $\varphi^{(h)}$ are grid functions, i.e. vectors of dimension $\mathcal{O}(2^n)$, approximating the corresponding continuously differentiable functions.

(b) The scheme (7) is said to approximate *the given differential problem with order of accuracy p (where p is a positive integer) on a solution $u(t,x)$ of the considered boundary-value problem if $\left|(u_t - Au)|_{G_h^\tau} - L_h u^{(h)}\right| \leq C_1 h^p$, and $\left|\varphi|_{G_h} - \varphi^{(h)}\right| \leq C_1 h^p$ for some constant C_1 not depending on h and τ. Here $u^{(h)} = \{u^{(h,l\tau)}\}_{l=1}^M$, L_h is the linear difference operator corresponding to (7) rewritten in the form $L_h u^{(h)} = 0$, M is the number of time steps.*

(c) The difference scheme (7) is called stable *if its solution $\mathbf{u}^{(h)}$ satisfies $|\mathbf{u}^{(h)}| \leq c_2 |\varphi^{(h)}|$ for a constant c_2 independent of h, τ and $\varphi^{(h)}$. Here $|\cdot|$ is the* sup*-norm, i.e. the maximal value over all grid cells.*

(d) We will call the complexity of the difference scheme (7) *be the complexity of the corresponding matrix A_n in the in sense of Definition 5(b).*

Fact 11 *(Well-known facts, see e.g. [5]): (a) Let the difference scheme be stable and approximate (1) on the solution u with order p. Then the solution $u^{(h)}$ of the recursively defined linear algebraic systems (7) uniformly converges to the solution u in the sense that*

$$|u|_{G_h^\tau} - u^{(h)}| \leq C h^p \tag{8}$$

for C not depending on h and τ (but possibly depending on the inputs; this dependence will occur later in the proofs).

(b) The difference scheme (7) is stable iff there is a constant C_2 uniformly bounding all powers of A_n:

$$|A_n^q| \leq C_2, \; q = 1, 2, \ldots, M. \tag{9}$$

The stability property implies $\tau \leq \nu h$; ν is called the Courant number.

(c) The convergence constant in (8) is $C = C_1 \cdot C_2$, where C_2 is from (9); C_1 is from the approximation property.

Proof (Theorem 2, sketch). To evaluate the solution u at a fixed point (t, x) with the prescribed precision 2^{-n} and estimate the bit-cost of the computation, consider the following computation steps.

1. Choose binary-rational grid steps $h = 2^{-N}$ (where $N = \mathcal{O}(n)$) and $\tau \leq \nu h$ (where ν is the Courant number existing due to stability property): τ just any binary-rational meeting this inequality; h defined by the inequality (12) below.
2. For a grid point (t, x) put $l = \frac{t}{\tau}$ (note that $l \leq M = \left[\frac{1}{\tau}\right] = \mathcal{O}(2^n)$) and calculate the matrix powers and vector products
$$u^{(h,l\tau)} = A_n^l \varphi^{(h)}. \tag{10}$$
Note that (10) uses matrix powering instead of step-by-step iterations initially suggested by the difference scheme (7).
3. For non-grid points take (e.g.) a multilinear interpolation $\widetilde{u^{(h)}}$ of $u^{(h)}$ computed from the (constant number of) "neighbor" grid points.

Due to well-known properties of multilinear interpolations,

$$\sup_{t,x} |\widetilde{u^{(h)}}(t,x)| \leq \tilde{C} \sup_{G_h^\tau} |u^{(h)}|; \quad \sup_{t,x} |\widetilde{u\,|_{G_h^\tau}}(t,x)| \leq \bar{C} \sup_{t,x} |D^2 u(t,x)| \cdot h^2, \tag{11}$$

where $\tilde{C}$ and $\bar{C}$ are absolute constants. Based on (11) and on the continuous dependence property, as well as on linearity of the interpolation operator, infer

$$\sup_{t,x} |u(t,x) - \widetilde{u^{(h)}}(t,x)| \leq \sup_{t,x} \Big(|u(t,x) - \widetilde{u\,|_{G_h^\tau}}(t,x)| + |\widetilde{u\,|_{G_h^\tau}}(t,x) - \widetilde{u^{(h)}}(t,x)| \Big)$$
$$\leq \tilde{C} C_0 \sup_x |D^2 \varphi(x)| h^2 + \bar{C} C_1 C_2 \cdot h \leq 2^{-n}.$$

Thus choosing a grid step $h = 2^{-N}$ such that

$$h \leq C_h \cdot 2^{-n}, \quad C_h = \tilde{C} C_0 \sup_x |D^2 \varphi(x)| + \bar{C} C_1 C_2, \tag{12}$$

will guarantee the computed function $\widetilde{u^{(h)}}$ approximate the solution u with the prescribed precision 2^{-n} (here C_h depends only on the fixed $s(n)$ space computable functions φ, B_i and therefore is a fixed constant).

According to (10), item (a) of Theorem 2 follows from items (f) and (d) of Proposition 6; item (b) of Theorem 2 follows from item (d) of Theorem 7 combined with item (d) of Proposition 6. □

Conditions of Theorem 2 hold for large classes of IVPs and BVPs for linear PDEs. E.g. for certain BVPs for symmetric hyperbolic systems $u_t + \sum_{i=1}^{m} B_i u_{x_i} = 0$ with constant matrices $B_i = B_i^*$ (to which also the wave equation $p_{tt} - a^2 \sum_{i=1}^{m} p_{x_i x_i} = 0$ can be reduced), as well as for the heat equation

$p_t - a^2 \sum_{i=1}^{m} p_{x_i x_i} = 0$, difference schemes with circulant constant bandwidth matrices can be constructed. For equations with non-constant coefficients the matrices are more complicated, thus the corresponding problems might have higher complexity. Deriving optimal complexity bounds for the considered (and possibly broader) classes of equations is one of the directions of future work.

References

1. Allender, E., Bürgisser, P., Kjeldgaard-Pedersen, J., Miltersen, P.B.: On the complexity of numerical analysis. SIAM J. Comput. **38**(5), 1987–2006 (2009)
2. Borodin, A.: Relating time and space to size and depth. SIAM J. Comput. **6**(4), 733–744 (1977). https://doi.org/10.1137/0206054
3. Bournez, O., Graça, D.S., Pouly, A.: Solving analytic differential equations in polynomial time over unbounded domains. In: Murlak, F., Sankowski, P. (eds.) MFCS 2011. LNCS, vol. 6907, pp. 170–181. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22993-0_18
4. Evans, L.: Partial Differential Equations, Graduate Studies in Mathematics, vol. 19. American Mathematical Society, Providence (1998)
5. Godunov, S., Ryaben'kii, V.: Difference Schemes: An Introduction to the Underlying Theory. Elsevier Science Ltd., New York (1987)
6. Hoover, H.J.: Feasible real functions and arithmetic circuits. SIAM J. Comput. **19**(1), 182–204 (1990). https://doi.org/10.1137/0219012
7. Kawamura, A., Müller, N.T., Rösnick, C., Ziegler, M.: Computational benefit of smoothness: parameterized bit-complexity of numerical operators on analytic functions and Gevrey's hierarchy. J. Complex. **31**(5), 689–714 (2015)
8. Kawamura, A., Ota, H.: Small complexity classes for computable analysis. In: Csuhaj-Varjú, E., Dietzfelbinger, M., Ésik, Z. (eds.) MFCS 2014. LNCS, vol. 8635, pp. 432–444. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44465-8_37
9. Kawamura, A., Ota, H., Rösnick, C., Ziegler, M.: Computational complexity of smooth differential equations. Logical Methods in Computer Science **10**, 1:6,15 (2014). https://doi.org/10.2168/LMCS-10(1:6)2014
10. Kawamura, A., Steinberg, F., Thies, H.: Parameterized complexity for uniform operators on multidimensional analytic functions and ODE solving. In: Moss, L.S., de Queiroz, R., Martinez, M. (eds.) WoLLIC 2018. LNCS, vol. 10944, pp. 223–236. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-662-57669-4_13
11. Kawamura, A., Steinberg, F., Ziegler, M.: On the computational complexity of the Dirichlet problem for Poisson's equation. Math. Struct. Comput. Sci. **27**(8), 1437–1465 (2017). https://doi.org/10.1017/S096012951600013X
12. Ko, K.I.: Complexity Theory of Real Functions. Progress in Theoretical Computer Science, Birkhäuser, Boston (1991)
13. Mayr, E., Meyer, A.: The complexity of the word problem for commutative semigroups and polynomial ideals. Adv. Math. **46**, 305–329 (1982)
14. Pan, V., Reif, J.: The bit-complexity of discrete solutions of partial differential equations: compact multigrid. Comput. Math. Appl. **20**, 9–16 (1990)
15. Selivanova, S., Selivanov, V.L.: Computing solution operators of boundary-value problems for some linear hyperbolic systems of PDEs. Logical Meth. Comput. Sci. **13**(4) (2017). https://doi.org/10.23638/LMCS-13(4:13)2017

16. Selivanova, S.V., Selivanov, V.L.: Bit complexity of computing solutions for symmetric hyperbolic systems of PDEs (extended abstract). Proc. CiE **2018**, 376–385 (2018). https://doi.org/10.1007/978-3-319-94418-0_38
17. Vollmer, H.: Introduction to Circuit Complexity. EATCS Text in Theoretical Computer Science. Springer, Heidelberg (1999)
18. Weihrauch, K.: Computable Analysis. Springer, Berlin (2000)
19. Yu, F., Ko, K.I.: On parallel complexity of analytic functions. Theoret. Comput. Sci. **489**(490), 48–57 (2013). https://doi.org/10.1016/j.tcs.2013.04.008