Journal of Complexity ••• (••••) ••••••

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

Bit-complexity of classical solutions of linear evolutionary systems of partial differential equations ☆

Ivan Koswara [a], Gleb Pogudin [b], Svetlana Selivanova [a,*], Martin Ziegler [a]

[a] KAIST, School of Computing, Daejeon, Republic of Korea
[b] LIX, CNRS, École Polytechnique, Institute Polytechnique de Paris, France

ARTICLE INFO

Article history:
Received 15 February 2022
Received in revised form 10 November 2022
Accepted 19 December 2022
Available online xxxx

Keywords:
Reliable computing
Bit-cost
Partial differential equations
PSPACE

ABSTRACT

We study the bit-complexity intrinsic to solving the initial-value and (several types of) boundary-value problems for linear evolutionary systems of partial differential equations (PDEs), based on the Computable Analysis approach. We compute classical solutions to such problems approximately up to guaranteed precision $1/2^n$, so that n corresponds to the number of reliable bits of the output; algorithmic cost is measured with respect to n.

Computational Complexity Theory allows us to prove in a rigorous sense that PDEs with constant coefficients are algorithmically 'easier' than general ones. Indeed, solutions to the latter are shown (under natural assumptions) computable using a polynomial number of memory bits, and we prove that the complexity class PSPACE is in general optimal; while the case of constant coefficients (with periodic boundary conditions) can be solved in #P—also essentially optimally so: the Heat Equation 'requires' $\#\mathsf{P}_1$.

Our assumptions include max-norm convergence of related difference schemes, which leads to various smoothness hypotheses depending on the particular example PDE. A major tool for our efficient algorithms is implicit powering/repeated squaring of *exponential-sized matrices* (representing difference schemes) for the general case of finitely continuously differentiable initial and coefficient functions, given that they provide existence of classical solutions.

☆ Communicated by K.K. Meer.

* Corresponding author.
E-mail address: sweseliv@gmail.com (S. Selivanova).

https://doi.org/10.1016/j.jco.2022.101727

Many partial differential equations with periodic boundary conditions, including the Heat and the Wave Equations, admit difference schemes that are (tensor products of constantly many) circulant matrices of constant 'modular' bandwidth; and for these we show exponential-sized matrix powering, as well as the PDE solution, computable in #P. This is achieved by considering a multivariate polynomial naturally associated to the difference scheme matrix and calculating individual coefficients of its powers using Cauchy's Differentiation Theorem, which may be of independent interest. Exponentially powering modular two-band circulant matrices is established even feasible in P; and under additional conditions, also the solution to certain linear PDEs becomes polynomial time computable.

Contents

1. Introduction and summary

Computable Analysis [62] provides a framework for rigorous computability and complexity investigations of computational problems over real numbers and functions by approximation up to guaranteed absolute error $1/2^n$ [4,34,63,7,26]. This has been applied to ordinary [5,27,31] and partial [58,57,61] differential equations. It allows to prove asymptotic optimality of numerical algorithms, for example by relating the intrinsic computational bit-cost of a problem to a classical discrete complex-

ity class [22,30,3,1]. Note that already stating (not to mention proving) such a relation requires great care: Central notions in Computational Complexity, such as polynomial-time reduction or completeness, pertain to extensional (i.e. functional) problems with discrete input and output; while processing continuous data by means of discretization is inherently *non*-extensional: see [42] and Subsection 2.2 below.

The present work considers general classes of systems of linear evolutionary partial differential equations (PDEs). In order to solve some system of ordinary differential equations (ODEs) $\partial_t \mathbf{u} = \vec{f}(\mathbf{u}, t)$, common numerical approaches such as Euler's Method and its refinements discretize time $t \in [0; 1]$ into steps $\tau \ll 1$: From the fixed initial value $\mathbf{u}(0) = \mathbf{u}_0$ at $t = 0$ they iteratively proceed to approximations $\mathbf{u}(\tau)$, $\mathbf{u}(2\tau)$, ..., $\mathbf{u}(M \cdot \tau)$. In order for the last one to approximate $\mathbf{u}(1)$ up to error $1/2^n$, the number $M = 1/\tau \in \mathbb{N}$ of steps is generally exponential in the number n of reliable bits of output approximation (Remark 10); and the problem thus seen to belong to the discrete complexity class EXPTIME.[1] Closer inspection improves that to PSPACE [34, §7.2], which has been proven to be the best possible in general [22]. Bit complexity is measured w.r.t. the output precision parameter n.

Evolutionary PDEs generalize ODEs: by replacing the right-hand side function $\vec{f} = \vec{f}(\mathbf{u}, t)$ with a differential operator, commonly involving spatial derivatives. The mathematical theory of PDEs is considerably more involved than that of ODEs [12], in particular, when it comes to existence, uniqueness, and continuous dependence of solutions (=well-posedness in the sense of Hadamard). Recall that one of the Millennium Prize Problem asks such questions for the Navier-Stokes Equation. Computability investigations of PDEs have challenged the Church-Turing Hypothesis [53,64,61].

The present work considers linear evolutionary PDEs with initial conditions:

$$\begin{cases} \mathbf{u}_t = \sum_{|\vec{j}|} \mathbf{B}_{\vec{j}}(\vec{x}) \cdot \partial^{\vec{j}} \mathbf{u}, \quad 0 \le t \le 1, \quad \vec{x} \in \Omega, \\ \mathbf{u}\,|_{t=0} = \varphi(\vec{x}), \quad \vec{x} \in \Omega, \end{cases} \tag{1}$$

and, possibly, boundary conditions to guarantee uniqueness

$$\mathcal{L}\mathbf{u}(t, \vec{x})\,|_{\partial\Omega} = 0, \quad (t, \vec{x}) \in [0, 1] \times \partial\Omega, \tag{2}$$

where $\Omega = [0, 1]^d$ is the unit cube (for technical simplicity); $\partial\Omega$ is its boundary; the solution $\mathbf{u} = (u_1, \ldots, u_e) = \mathbf{u}(t, \vec{x})$ is an unknown function on Ω with values in $\mathbb{R}^d$; $\mathcal{L}$ in the boundary condition is a linear differential operator of order strictly less than the order of the differential operator $\sum_{|\vec{j}|} \mathbf{B}_{\vec{j}}(\vec{x}) \cdot \partial^{\vec{j}} \mathbf{u}$.

The coefficients of the differential operator are $d' \times d'$ matrices $\mathbf{B}_{\vec{j}}$ that may depend on $\vec{x}$, but not on t (autonomous case); $\vec{j} = (j_1, \ldots, j_d)$ denotes a multi-index of order $|\vec{j}| = j_1 + j_2 + \ldots + j_d$, $\partial^{\vec{j}} = \partial_1^{j_1} \cdots \partial_d^{j_d}$ denotes the corresponding differential operator, where $\partial_k^{j_k} = \frac{\partial^{j_k}}{\partial x_k^{j_k}}$; and $\varphi(\vec{x})$ is the initial condition. Note that the equations (1) are linear in the derivatives, but the matrix coefficients $\mathbf{B}_j$ can depend on $\vec{x}$ non-linearly.

Example 1. An important and rich class of PDEs of the form (1) are the first-order systems

$$\mathbf{u}_t = \sum_{j=1}^{d} \mathbf{B}_j(\vec{x}) \cdot \partial_{x_j} \mathbf{u}. \tag{3}$$

In particular, such systems are called *symmetric hyperbolic*, if $\mathbf{B}_j(\vec{x}) = \mathbf{B}_j^*(\vec{x}), \quad j = 0, 1, \ldots, d$. This class includes the linear acoustics, elasticity and Maxwell equations [58].

Also the (second-order) Wave Equation

$$u_{tt} = a^2 \sum_{j=1}^{d} \partial_{x_j}^2 u \tag{4}$$

and many others can be reduced to such a system by introducing extra unknown functions ([14]; see also [12, 4.6.3.a] for a reduction of higher-order systems to first-order systems (3)).

[1] For notational convenience we use the terminology from discrete complexity classes also for the real case. Formal Definitions of the latter are collected in Section 2.

The Heat Equation

$$u_t = a^2 \sum_{j=1}^{d} \partial_{x_j}^2 u \tag{5}$$

is not of the form (3), but still of the form (1). Periodic boundary conditions on the unit cube are captured by

$$\mathcal{L}\mathbf{u}(t, x_1, \ldots, x_{j-1}, 0, x_{j+1}, \ldots, x_d) := \mathbf{u}(t, x_1, \ldots, x_{j-1}, 0, x_{j+1}, \ldots, x_d) - \mathbf{u}(t, x_1, \ldots, x_{j-1}, 1, x_{j+1}, \ldots, x_d) \tag{6}$$

for first-order systems, and include similar conditions on spatial derivatives up to $\ell - 1$ order for ℓ-order systems.

Under suitable hypotheses, Euler's method generalizes from ODEs to evolutionary PDEs (1): by discretizing now both physical time and space, the latter with some grid of sufficiently (=exponentially in n) small width $h \ll 1$. This turns the initial condition φ into a vector of exponentially large dimension $\mathcal{O}(1/h)$. The right-hand side linear operator $\mathcal{A}$ may be approximated by a matrix $\mathbf{A}$, often referred to as *difference scheme*, see Definition 6. And if the evolution equation is autonomous, said matrix does not depend on time. In this case repeated time-stepping $\mathbf{u}(t) \mapsto \mathbf{u}(t+\tau) = \mathbf{A} \cdot \mathbf{u}(t)$ amounts to M-fold repeated multiplication by $\mathbf{A}$, i.e., to (exponential) matrix powering $\mathbf{A}^M$, where $M \cong 1/\tau$ is also exponential in n.

So all three, the discretized initial condition $\mathbf{u}(0) = \varphi$ and the matrix $\mathbf{A}$ and the resulting approximation to $\mathbf{u}(1)$, have dimension $K = \mathcal{O}(1/h)$ exponential in n: when considered as input and output, they thus information-theoretically leave no chance for sub-exponential computational cost. More interesting is therefore the following question:

Question 2. Fix *polynomial-time computable initial condition,* fix *polynomial-time computable matrix coefficients* $\mathbf{B}_j$ *in the right-hand side of PDE* (1)*, and similarly for boundary condition* $\mathcal{L}$*. Now consider* only $(t, \vec{x}) \in [0;1] \times \Omega$ *as input: What is the bit-cost (measured w.r.t. the parameter* n*) of approximating the solution* $\mathbf{u}(t, \vec{x})$ *at time* t *and point* $\vec{x}$ *up to absolute error* $1/2^n$*?*

Thus, non-uniformly fixing all data of exponential 'size' (formally: from spaces of exponential entropy [37,63,29]) and restricting to polynomial 'size' inputs $(t, \vec{x}) \in [0;1] \times \Omega$ avoids information-theoretic exponential lower complexity bounds.

Compact domains ensure that one can restrict to complexity considerations in terms of one parameter n [62, Theorem 7.2.7] and does not need to resort to second-order complexity [23]; see also, e.g. [24,52,28] for parameterized complexity applied to differential equations. We use n as parameter for producing approximations up to error bound $1/2^n$, not $1/n$, following the conventions of Real Complexity Theory [34, Definition 2.7]; see also Remark 10 below.

Poisson's elliptic (i.e. non-evolutionary) PDE characterizes $\#\mathsf{P}_1$ [30]; cmp. Fact 11c). Computation on grids of size $\mathcal{O}(N)$ incurs $\mathcal{O}(\log N)$ parallel runtime [49]. In our terminology of grid width $h \sim 1/2^n$ for guaranteed output approximation error $1/2^n$, this means $N = \mathcal{O}(1/h^d)$ and parallel runtime $\mathcal{O}(nd)$. That would amount to complexity class $\mathsf{PAR} = \mathsf{PSPACE}$, were it not for the superpolynomial number $\mathcal{O}(N/\log N) = \mathcal{O}(2^{nd}/nd)$ of processors.

In [38,56] we have proved the solutions to be polynomial-time computable if the initial functions and matrix coefficients are polynomial-time computable and analytic (an effective version of the Cauchy-Kovalevskaya theorem) and implemented the corresponding algorithms in the Exact Real Computation framework [51]. This extends results for ODEs with polynomial [5,43] and analytic right-hand sides [27,31]. In the present paper we study the case of classical solutions when the initial data are not analytic.

The main result of the present paper, Theorem 4, rigorously and in the sense of Question 2 establishes solutions to a large class of linear first-order evolutionary PDEs computable in PSPACE, and

optimally so; and it improves the latter to #P for the case of constant matrix coefficients with periodic boundary conditions. The latter applies, in particular, to the Heat Equation, where we show $\#\mathsf{P}_1$ as necessary.

This paper is essentially based on the conference proceedings [38] and provides more detailed proofs of the results sketched there.

1.1. Main results and overview

The complexity considerations in this work refer to (real counterparts, formalized in Section 2, of) the classical hierarchy commonly conjectured proper:

$$\mathsf{NC} \subseteq \mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{PSPACE} = \mathsf{PAR} \subseteq \mathsf{EXP} \ . \tag{7}$$

The following hypotheses are very natural and hold for many PDEs including many of the ones mentioned in Example 1, see Example 31 for more detail. For the notation of (iii) see Subsection 1.2; $\|\varphi\|_{C^l(\bar{\Omega})} = \sup_{\vec{x}\in\bar{\Omega}} \sum_{|\vec{j}|\le l} |\partial^{\vec{j}}\varphi(\vec{x})|$.

Hypotheses 3.

(i) The problem (1) is well-posed (Hadamard) in that the classical solution $\mathbf{u}(t,\vec{x})$ to (1) exists, is unique and depends continuously on the initial data in the following sense:

$$\varphi(\vec{x}) \in \mathcal{C}^l(\bar{\Omega}), \quad \mathbf{u}(t,\vec{x}) \in \mathcal{C}^2([0,1]\times\bar{\Omega}), \quad \|\mathbf{u}\|_{C^2([0,1]\times\bar{\Omega})} \le C_0\|\varphi\|_{C^l(\Omega)}, \tag{8}$$

for some fixed C_0, $l \ge 2$.

(ii) The initial functions $\varphi(\vec{x})$ and matrix coefficients $\mathbf{B_j}(\vec{x})$ as well as their partial derivatives up to order l are polynomial time computable.

(iii) The system (1) admits a difference scheme $\mathbf{A}_{h(n)}$ (see (9) below) which is polynomial time computable, and its solution $u^{(n)}$ converges to the solution $\mathbf{u}$ of (1) w.r.t. the maximum norm on the uniform grid $G_{h(n)}$ with the step $h = h(n)$:

$$\max_{x\in G_{h(n)}} \left|\mathbf{u}|_{G_{h(n)}} - u^{(n)}\right| < C\cdot h(n), \quad C \text{ does not depend on } n.$$

Note that technically a difference scheme is a family $\mathbf{A}_{h(n)}$ of matrices of dimension growing exponentially in $n\to\infty$ such as to approximate the operator $\mathcal{A}$ with increasing precision; the approximating solution $u^{(n)}$ is a sequence of vectors of dimension growing exponentially in n. See Definition 14 of Subsection 2.2 for adjustment of the complexity classes to this case.

The main results of the present paper are collected in the following two theorems.

Theorem 4 *(Complexity of linear evolutionary PDEs).*

a) *The solution* $\mathbf{u}$ *of* (1) *under Hypotheses 3 is computable in PSPACE.*

b) *And PSPACE is optimal in general:*
There exists a polynomial-time computable Lipschitz-continuous functions $b : [0;1]\times\Omega \to [0;1]$ *and* $v_0 : \Omega \to [0;1]$ *satisfying periodic boundary conditions* $b(t,0) \equiv b(t,1)$ *and* $b_x(t,0) \equiv b_x(t,1)$ *and* $v_0(0) = v_0(1)$ *and* $v_0'(0) = v_0'(1)$ *such that the solution* $v : [0;1]\times\Omega \to [0;1]$ *of the linear PDE initial value problem*

$$v_t(t,x) = b(t,x)\cdot v_x(t,x), \qquad v(0,x) = v_0(x)$$

with periodic boundary condition $v(t,0) \equiv v(t,1)$ *and* $v_x(t,0) \equiv v_x(t,1)$ *exists but is not polynomial-time computable unless* $\mathsf{P} = \mathsf{PSPACE}$.

c) *If, in addition to the hypothesis from (a), the difference scheme* $\mathbf{A}_h$ *from (iii) is a sum of tensor products of circulant block matrices of constant modular bandwidth (Definition 21), the solution* $(t, \vec{x}) \mapsto \mathbf{u}(t, \vec{x})$ *of (1) is computable in #P.*

d) *The solution* $\mathbf{u}$ *of (3) is even polynomial time computable if the matrices* $\mathbf{B}_j$ *are constant and mutually commute for* $j = 0, 1, \ldots d$.

e) *Item (c) is essentially optimal in general:*
For the Heat Equation (5) with periodic boundary conditions (6) there exists a polynomial time computable initial condition φ *such that the solution* u *is classical but cannot be computed in polynomial time unless* $FP_1 = \#P_1$.

'Essentially' optimal in (e) refers to the gap between $\#\mathsf{P}_1$ and #P in (c); this gap is discussed before Fact 11. Item b) harnesses a particular structure common to difference schemes, formalized in Definition 21 below.

Remark 5. Intuitively, for PDE systems with constant matrices, the *locality* of the grid discretization of the differential operator yields a difference scheme $\mathbf{A}_{h(n)}$ with *constant* modular bandwidth; and *periodic* boundary conditions yield a *circulant* structure. Higher-dimensional Euclidean domains translate to tensor products of such (families of) matrices; see Subsection 3.7.

The following table summarizes the current achievements on bit-complexity of linear PDE systems.

Type of PDE/Functional Class	Poisson problem for the Laplace equation: Elliptic	Linear Evolutionary Systems: include Hyperbolic and Parabolic
Analytic		P ([38]; [56] uniform version)
C^k, $k \geq 1$ (well posed)	#P; $\#\mathsf{P}_1$-'hard' [30]	Present paper: • PSPACE (general case) • #P (periodic w/const. coeff.) • $\#\mathsf{P}_1$-'hard' (Heat Equation) • P for constant mutually commuting matrix coefficients

Subsection 1.2 collects notational conventions and some preliminaries. Section 2 formalizes computational bit-complexity theory of real vectors and matrices of exponential dimension, after recalling the theory [32] for constant dimension. Subsection 3.1 presents the main techniques for proving Theorem 4 a), c), including powering matrices of exponentially growing dimensions (representing difference schemes) to exponential powers. Section 4 is devoted to the proof of Theorems 4, respectively, while Section 5 shows some examples of applications (including the conditions on smoothness, that need to be imposed for particular cases), as well as gives more insight into the tools described in Section 3. We conclude in Section 6 with a summary and some open questions.

1.2. Notation

We use $n \in \mathbb{N}$ to parametrize the absolute output approximation error bound $1/2^n$; $d \in \mathbb{N}$ is the dimension of the torus $\Omega = [0; 1)^d \bmod \vec{1}$ as compact spatial domain of the partial differential equation under consideration and $e \in \mathbb{N}$ denotes the dimension of the solution function vector $\mathbf{u}$.

Definition 6 *(Difference schemes).*

a) Consider, for any positive integer N, the uniform rectangular grid G_N on Ω defined by the points

$$\left(\frac{i_1 - \frac{1}{2}}{2^N}, \frac{i_2 - \frac{1}{2}}{2^N}, \ldots, \frac{i_m - \frac{1}{2}}{2^N} \right)$$

where $1 \le i_1, i_2, \ldots, i_m \le 2^N$. Let $h = 1/2^N$ be the corresponding spatial grid step and τ be a time step. Denote $G_N^\tau = G_N \times \{l\tau\}_{l=1}^M$, where M is the number of time steps. The choice of steps h and τ, depending on the output precision parameter n, is specified below in Subsection 4.1. We consider the following grid norm: $|g^{(h)}| = \max_{x \in G_N} |g^{(h)}(x)|$.

b) For a linear differential operator $\mathcal{A}$, the matrix $\mathbf{A}_h$ (with the grid step $h = h(n)$) defines the corresponding *difference scheme*

$$u^{(h,(l+1)\tau)} = \mathbf{A}_h u^{(h,l\tau)}, \quad u^{(h,0)} = \varphi^{(h)} \tag{9}$$

under consideration. Its entries are denoted $(\mathbf{A}_{(h)})_{I,J}$, $1 \le I, J \le K$. Here $K \sim 2^{\mathcal{O}(n)}$ is the dimension of the vectors $\mathbf{u}^{(h,m\tau)}$ approximating the solution $\mathbf{u}(m\tau, \vec{x})$ at time $m\tau \le 1$, i.e., for $1 \le m \le M := 1/\tau \sim 2^n$. $\tau, h \sim 1/2^n$ denote the temporal and spatial grid widths, respectively. Generally speaking, capital letters denote quantities (ranging up to) exponential in n.

c) The solution $u^{(h)}$ of the difference scheme (9) *converges* to the solution u of (1) if there is a constant C not depending on h and τ such that

$$\left|u|_{G_h^\tau} - u^{(h)}\right| \le Ch^p \ . \tag{10}$$

Due to the Lax Convergence Theorem, a difference scheme (9) converges in the sense of (10) to the solution of the PDEs (1) if and only if it is *approximating* and *stable*; see e.g. [60]. The latter means that the matrix A_h has bounded powers, see Definition 7 b) below.

Definition 7 *(Matrices).* Fix a (not necessarily commutative) ring $\mathcal{R}$.

a) If $\mathcal{R}$ comes with norm $|\cdot|$, we equip vectors over $\mathcal{R}$ with the induced maximum norm, and matrices with the induced operator norm:

$$\|\mathbf{u}\| = \max_j |u_j|, \quad \|\mathbf{A}\| = \max\left\{\|\mathbf{A} \cdot \mathbf{u}\| / \|\mathbf{u}\|\right\} \ .$$

b) A square matrix $\mathbf{A}$ with entries $\mathbf{A}_{I,J}$ $(0 \le I, J < K)$ is said to have *bounded powers* if its powers are uniformly bounded, i.e., iff there exists some $C \in \mathbb{N}$ such that $\|\mathbf{A}^M\| \le C$ holds for all $M \in \mathbb{N}$. Similarly for a family $\mathbf{A}_k$ of square matrices of possibly varying format, $k \in \mathbb{N}$: Here $\|\mathbf{A}_k^M\|$ must be bounded independently of both $M \in \mathbb{N}$ and of $k \in \mathbb{N}$.

c) Let $\mathcal{R}_D^{N \times N}$ denote the vector space of $N \times N$ matrices of modular *bandwidth* $< D$:

$$\mathbf{A} \in \mathcal{R}_D^{N \times N} \quad \Leftrightarrow \quad |I - J| \bmod N \ge D \Rightarrow \mathbf{A}_{I,J} = 0.$$

Modular bandwidth slightly relaxes traditional bandwidth in allowing 'off-diagonals' to wrap around, as in circulant matrices.

The (one-)norm of an *integer* multi-index $\vec{j} = (j_1, \ldots, j_L) \in \mathbb{Z}^L$ is $|\vec{j}| = j_1 + \cdots + j_L$. We write "$\vec{j} \ge \vec{0}$" to indicate non-negative (i.e. natural number) multi-indices. $D = \max\left\{|\vec{j}| : \mathbf{B}_{\vec{j}} \ne 0\right\}$ denotes the *order* of the PDE (1).

A main tool in our algorithms translates circulant matrix powering to polynomial powering. Indeed, the above notions have immediate counterparts:

Definition 8 *(Polynomials).* Fix a (not necessarily commutative) ring $\mathcal{R}$.

a) A polynomial in L commuting variables of *componentwise degree* less than $\vec{D} \in \mathbb{N}^L$ has the form

$$P(\vec{X}) = P(X_1, \ldots, X_L) = \sum_{\vec{0} < J \le \vec{D}} p_{\vec{J}} \vec{X}^{\vec{J}} = \sum_{\vec{J} \ge \vec{0}, |\vec{J}| < \vec{D}} p_{\vec{J}} \cdot \prod_{\ell=1}^L X_\ell^{j_\ell} \ .$$

Write $\mathcal{R}_{\vec{D}}[X_1, \ldots, X_L] = \mathcal{R}_{\vec{D}}[\vec{X}]$ for the vector space of such polynomials.

b) An L-variate *Laurent* polynomial of componentwise degree $< \vec{D}$ has the form

$$P(\vec{X}) = P(X_1, \ldots, X_L) = \sum_{-\vec{D} < \vec{j} < \vec{D}} p_{\vec{j}} \vec{X}^{\vec{j}} ,$$

that is, including negative powers of the variables. Write

$$\mathcal{R}_{\vec{D}}[X_1, X_1^{-1}, \ldots, X_L, X_L^{-1}] = \mathcal{R}_{\vec{D}}[\vec{X}, \vec{X}^{-1}]$$

for the vector space of such Laurent polynomials; and $P\left[\vec{X}^{\vec{j}}\right] := p_{\vec{j}}$ for the coefficient to $\vec{X}^{\vec{j}}$ in P, $\vec{j} \in \mathbb{Z}^L$.

c) Suppose $\mathcal{R}$ is equipped with a norm $|\cdot|$. Consider the ring $\mathcal{R}[\vec{X}, \vec{X}^{-1}]$ of all (Laurent) polynomials, equipped with the induced norm $\|P\| := \sum_{|\vec{j}|} |p_{\vec{j}}|$. A (Laurent) polynomial P has *bounded powers* iff $\|P^M\|$ is bounded independently of $M \in \mathbb{N}$.
Similarly for a family P_k of polynomials of possibly varying number of variables, $k \in N$: Here $\|P_k^M\|$ must be bounded independently of both $M \in \mathbb{N}$ and of $k \in N$.

Multivariate degree is understood componentwise and w.r.t. strict inequality "$<$" (not $=$ nor $\leq$). For example $X^2 \cdot Y^3 + X^3 \cdot Y^2$ has componentwise degree $< \vec{D} = (4, 4)$, but not $< (3, 4)$ nor $< (4, 3)$. An L-variate Laurent polynomial P of componentwise degree $< \vec{D}$ can be converted to an ordinary polynomial by multiplying P with $X_1^{D_1} \cdots X_L^{D_L}$.

In the sequel, *exponential* growth is to be understood as bounded by $2^{p(k)}$, $k \to \infty$, for some polynomial p.

2. Real complexity theory

In [34], major classical complexity classes have been adapted from the discrete case to the setting of real numbers and (continuous) real functions. There the integer parameter n governing the output approximation error $1/2^n$ replaces the role of the binary input length.

2.1. Complexity of reals/vectors/polynomials in fixed dimension

Let us recall the definitions of polynomial/exponential time/space computability of real numbers, (fixed-dimensional) real vectors, sequences of real numbers, and partial real functions [34].

Definition 9.

a) *Computing* a **real number** $r \in \mathbb{R}$ means to output, given $n \in \mathbb{N}$, some numerators $a_n \in \mathbb{Z}$ in binary with $|r - a_n/2^{\alpha(n)}| \leq 1/2^n$ for some polynomial $\alpha \in \mathbb{N}[N]$. Such a computation runs in *polynomial time* (P) if said a_n is output within a number of steps bounded by a polynomial in n. It runs in *exponential time* (EXPTIME) if the number of steps is bounded exponentially in n. The computation runs in *polynomial space* (PSPACE) if the amount of memory is bounded polynomially in n.

b) Computing a **(finite-dimensional) real vector** (in P, EXPTIME, PSPACE) means to compute each of its entries separately (in P, EXPTIME, PSPACE).

c) Computing a **sequence** $\bar{r} = (r_k) \in \mathbb{R}$ **of real numbers** means to output, given n and k, some $a_{n,k} \in \mathbb{Z}$ in binary with $|r_k - a_{n,k}/2^{\alpha(n,k)}| \leq 1/2^n$ for some bivariate polynomial $\alpha \in \mathbb{N}[N, K]$. Such a computation runs in *polynomial time* (P) if said $a_{n,k}$ is output within a number of steps bounded by a polynomial in $n+k$. Similarly for *exponential time* (EXPTIME) and *polynomial space* (PSPACE).

d) Computing a **partial real function** $f :\subseteq \mathbb{R} \to \mathbb{R}$ (w.r.t. some recursive integer functions $\alpha, \beta, \gamma \in \mathbb{N}^{\mathbb{N}}$) means, given $n \in \mathbb{N}$ and any numerator $a \in \mathbb{Z}$ in binary with $|x - a/2^{\gamma(n)}| \leq 1/2^{\beta(n)}$ for some $x \in \mathrm{dom}(f)$, to output some $b = b_n(a) \in \mathbb{Z}$ in binary with $|f(x) - b_n/2^{\alpha(n)}| \leq 1/2^n$. The computation may behave arbitrarily on inputs a that do not satisfy the hypothesis.

e) A computation according to (d) runs in *polynomial time* (P) if α, β, γ are polynomials and if said b_n is output within a number of steps bounded by a polynomial in n. Similarly for *polynomial space* (PSPACE).

Note that the bivariate polynomial $\alpha(n,k)$ in (c) can equivalently be replaced by a univariate polynomial in the sum $n+k$. Definition 9d) is equivalent to general computability [62, §4.2+§4.3], and (e) to other common complexity-theoretic notions [34, §7.2.1]: Either one requires a P/PSPACE computable function to have a polynomial modulus of continuity [34, Theorem 2.19]. Moreover for a sequence r_k of integers, classical P/PSPACE/EXPTIME computability is equivalent to (c).

Also record that a polynomial-time computable real sequence r_k according to (9c) can grow in magnitude at most exponentially (in k). Memory-bounded computation here is understood to charge for all three, input and working and output tape; hence sequences of reals computable in PSPACE also satisfy exponential bounds of growth.

In the following Remark we collect reasons to use error bound $1/2^n$ instead of another natural option $1/n$.

Remark 10. In addition to following the conventions of Real Complexity Theory [34, Definition 2.7], we prefer error bound $1/2^n$ (as opposed to $1/n$) for four reasons:

a) It corresponds to measuring computational cost of discrete problems, such as of integer factorization, in dependence of the binary (as opposed to unary) length n.
b) It reflects that (for instance Chudnovsky's or Borwein's) algorithms can approximate π in time polynomial in n up to error $1/2^n$ (while error bound $1/n$ is trivial to achieve).
c) It gives rise to the aforementioned and subsequent and many more [34] numerical characterizations of discrete complexity classes.
d) The first-order theory of the two-sorted structure

$$(\mathbb{Z}, 0, 1, +, >) \cup (\mathbb{R}, 0, 1, +, \times, >)$$

with 'error embedding' $\iota : \mathbb{Z} \ni n \mapsto 2^{-n} \in \mathbb{R}$ (capturing *Exact Real Computation*) is decidable, while that with $\mathbb{N}_+ \ni n \mapsto 1/n \in \mathbb{R}$ is not [51, Theorem 4.4].

Recall that the discrete complexity class #P consists of all total functions $\psi : \{0,1\}^* \to \mathbb{N}$ such that some non-deterministic polynomial-time Turing machine on input $\vec{x} \in \{0,1\}^*$ has precisely $\psi(\vec{x})$ accepting computations [32,2]. P_1, NP_1, $\#\mathsf{P}_1$ and denote the so-called *unary* variants of P, NP, and #P: classes of problems whose inputs are restricted to unary (aka *tally*) strings $\{1\}^*$, as opposed to binary $\{0,1\}^*$. It is easy to see that $\mathsf{P}_1 = \mathsf{NP}_1 \Leftrightarrow \mathsf{EXPTIME} = \mathsf{NEXPTIME}$; and consult for instance [15] regarding further relations between unary and binary complexity classes. $\#\mathsf{P}_1$ and #P are known to 'characterize' in/definite Riemann Integration [34, Theorem 5.32+5.33] in the following non-uniform sense:

Fact 11.

a) *For every fixed polynomial-time computable (and hence continuous) function* $h : [0;1]^d \to [-1;1]$*, the real number* $\int_{[0;1]^d} h(\vec{x})\, d\vec{x}$ *is computable in P, provided that* $FP_1 = \#P_1$.
b) *There exists a polynomial-time computable function* $h : [0;1] \to [-1;1]$ *(and even of class* $C_0^\infty[0;1]$*, i.e. infinitely often differentiable, vanishing with all derivatives at 0 and at 1) such that the real number* $\int_0^1 h(x)\, dx$ *is not computable in P unless* $FP_1 = \#P_1$.
c) *For every polynomial-time computable (and hence continuous) function* $h : [0;1]^d \to [-1;1]$*, the function*

$$\int h : [0;1]^d \ni \vec{y} \mapsto \int_{[\vec{0};\vec{y}]} h(x)\, d\vec{x} \in [-1;1] \tag{11}$$

is computable in P, provided that $FP = \#P$.

d) *There exists a polynomial-time computable function* $h : [0;1] \to [-1;1]$ *(and even of class* $\mathcal{C}_0^\infty[0;1]$*), such that Equation* (11) *with* $d = 1$ *is not computable in* P *unless* $FP = \#P$.

e) *For every polynomial time computable* analytic *function* $h : [0;1]^d \to \mathbb{R}$, *Equation* (11) *is computable in* P [34, bottom of page 208].

f) *For every Lipschitz-continuous polynomial-time computable function* $f : [0;1] \times [-1;1] \to [-1;1]$, *the unique solution* $u : [0;1] \to [-1;1]$ *to the non-linear ordinary differential equation*

$$\dot{u}(t) = f\big(t, u(t)\big), \quad u(0) = 0 \tag{12}$$

is computable in PSPACE [34, §7.2].

g) *Item (f) is optimal in that there exists a polynomial-time computable* $f : [0;1] \times [-1;1] \to \mathbb{R}$, *of class* $\mathcal{C}^\infty$ *in the first 'temporal' argument and Lipschitz-continuous in the second 'spatial' argument, such that the solution* $u : [0;1] \to [-1;1]$ *to Equation* (12) *exists but is not polynomial-time computable unless* P *= PSPACE [22, Theorem 3.2].*

g′) *Item (g) has been strengthened from Lipschitz* f *to* $\mathcal{C}^1$ *in the spatial argument [25]. Moreover its first spatial derivative* $\partial_x f$ *is in turn polynomial-time computable (personal communication 2022); and the proof can be adapted to ensure* $f(t,-1) \equiv 0 \equiv f(t,+1)$ *and* $f_x(t,-1) \equiv 0 \equiv f_x(t,+1)$ *for all* $t \in [0;1]$. *On the other hand, the single value* $u(1) \in [-1;1]$ *is only* $PSPACE_1$*-'hard' [25, §5.1].*

Of course Items (b) and (d) and (g) hold *a fortiori* also in higher dimensions.

Remark 12. Note that Items (b) and (d) are of the form

XYZ is not computable in polynomial time unless FP = #P

and similarly for (g). We shall abbreviate such statements as XYZ being "#P-hard"; similarly for $\#P_1$ and PSPACE: although "hardness" in Computational Complexity is defined via polynomial-time reduction between discrete decision problems and not applicable to the continuous problems considered here. Refer to [6] for the definition and survey of computable reducibility between continuous problems, and to [23] for its polynomial-time strengthening.

Also Items (a), (c) in Fact 11 are non-uniform and do not refer to "computability in #P" or $\#P_1$, as introduced in Definition 13. So if towards the Millenium problem someone manages to prove $FP \neq \#P$, then a statement according to Remark 12 will become empty: *ex falso quodlibet. Fact 11f) on the other hand does remain meaningful even in case* $P \neq PSPACE$, *because we do have defined "computability in PSPACE".* A reasonable complexity class of real numbers/sequences/functions should arguably be closed under arithmetic operations such as subtraction, multiplication and (bounded) division. However which arithmetic integer operations #P is closed under remains an open question. For instance GapP has been introduced as the closure of #P under subtraction [16]. We therefore follow and generalize [34, p. 184] to define real counterparts to #P via oracles:

Definition 13. Call a real number *computable in #P* ('polynomially countable') if it is computable in $P^{\#P}$, that is, in polynomial-time (Definition 9a, but) with oracle access to #P.

Similarly for finite-dimensional vectors (Definition 9b), sequences (Definition 9c), and partial real functions (Definition 9d)

Fact 11(a)+(c) can now be strengthened to the statements that in/definite Riemann integration is in $\#P/\#P_1$, following the proofs of [34, Theorem 5.32+Theorem 5.33]. Note that we could also define computability in PSPACE equivalently as polynomial *time* computability with PSPACE oracle. As in the discrete case, real polynomial-time computability implies computability in #P, which in turn implies computability in PSPACE. In particular also any real sequence r_k computable in #P has $|r_k|$ growing at most exponentially in k.

2.2. Complexity of reals/vectors/polynomials in growing dimension

In order to guarantee absolute error bound $1/2^n$, even within the class of smooth real functions in 1D, requires discretization on a grid of exponential size: for entropy reasons. Exponential running time thus can only be beaten when taking each such information-theoretically 'large' argument as fixed polynomial-time computable; recall the above discussion around Question 2. On the other hand the discretizations as real vectors, and the difference schemes operating on the latter, cannot be fixed: they do and must depend on n, exponentially.

To formalize the latter in Definition 14 below, we consider a compromise 'semi-uniform' setting: neither fixing the real vector/matrix (and its dimension), nor accepting it as input, but rather fixing a *sequence* of real vectors $\vec{v}_k$/matrices $\mathbf{A}_k$/polynomials P_k of growing dimensions/degrees D_k. Since these D_k tend to grow exponentially fast, we only require that any desired entry/coefficient be computable in time polynomial

- in the output precision parameter n (=the length of n in *un*ary) and
- in the sequence index k (=its length in *un*ary) and
- in the *bin*ary length of indices I, J of the entry $(A_k)_{I,J}$, or in the *bin*ary length of the index J of the coefficient $(P_k)_J$.

Thus non-uniquely 'discretizing' a polynomial-time computable function yields a polynomial-time computable vector sequence in this sense. The complexity of computing said dimension D_k itself must be taken into account as well. Up to a polynomial (quadratic) difference in complexity, we may identify a $D \times E$-dimensional matrix $B = (b_{I,J})$ with the $D \times E$-dimensional vector $B_{\langle I,J\rangle}$, for the pairing function $\langle I, J\rangle = J + (I + J) \cdot (I + J + 1)/2$.

Definition 14.

a) Computing a **sequence** $\mathbf{r}_k = (r_{k,J})_{J \le D_k} \in \mathbb{R}^{D_k}$ of D_k**-dimensional real vectors** means to output, given $n, k \in \mathbb{N}$ (in unary) and $J \le \dim(\mathbf{r}_k) = D_k$ in binary, some $a_{n,k,J} \in \mathbb{Z}$ in binary with

$$|r_{k,J} - a_{n,k,J}/2^{\alpha(n,k)}| \le 1/2^n$$

for some bivariate polynomial $\alpha \in \mathbb{N}[N, K]$.
Such a computation runs in *polynomial time* (P) if said $a_{n,k,J}$ is output within a number of steps bounded by a (univariate) polynomial in $n + k$ but independently of J. Similarly for *exponential time* (EXPTIME) and *polynomial space* (PSPACE) and *polynomial countability* (#P).

b) More generally, computing a **sequence** $\mathbf{f}_k :\subseteq \mathbb{R}^d \to \mathbb{R}^{D_k}$ **of partial vector functions** (w.r.t. some bivariate polynomials $\alpha, \beta, \gamma \in \mathbb{N}[N, K]$) means, given $n, k, J \in \mathbb{N}$ and any numerator $\vec{a} \in \mathbb{Z}^d$ with

$$||\vec{x} - \vec{a}/2^{\alpha(n,k)}|| \le 1/2^{\beta(n,k)}$$

for any $\vec{x} \in \mathrm{dom}(\mathbf{f}_k)$ and $J \le D_k = \dim f_k$, to output some $b = b_{n,k,J}(a) \in \mathbb{Z}$ with

$$||f_{k,J}(\vec{x}) - b/2^{\gamma(n,k)}|| \le 1/2^n \ .$$

Here

$$\|\vec{y}\| = \max\{|y_1|, \ldots, |y_d|\}$$

denotes the maximum norm.
Such a computation runs in *polynomial time* (P) if said $b = b_{n,k,J}(a)$ is output within a number of steps bounded by a polynomial in $n + k$ but independently of J and a. Similarly for *exponential time* (EXPTIME) and *polynomial space* (PSPACE) and *polynomial countability* (#P).

c) A sequence (D_k) of natural numbers is computable in *unary polynomial time* (P_1) if the mapping $\{1\}^* \ni 0^k \mapsto \mathrm{bin}(D_k) \in \{0, 1\}^*$ is computable in time polynomial in the input length.

More generally call a double sequence $\vec{D}_{k,\ell} \in \mathbb{Z}^{D_k}$ of integer vectors computable in *unary polynomial time* (P_1) if the partial mapping $\{0,1\}^* \ni 1^k\,0\,1^\ell\,0\ \mathrm{bin}(J) \mapsto \mathrm{bin}(D_{k,\ell}) \in \{0,1\}^*$ is computable in time polynomial in $k+\ell$ but independently of $J < D_k$.

An integer (vector) sequence is computable in unary polynomial time (Definition 14c) iff it is polynomial-time computable as a real sequence in the sense of Definitions 9c) and 14a). Both an integer and a real vector sequence computable in (unary) polynomial time can grow in magnitude at most exponentially in k, and so can the dimension; same for real vector functions according to (14b), and for PSPACE and #P-computable real sequences. Exponential growth of $D_k > 0$ is equivalent to polynomial growth of $\log(D_k)$; and (14c) strengthens this condition, namely it implies but does not follow from exponential growth.

Our real counterparts of P and #P and PSPACE above agree with both the conception of real numbers as 'streams' of approximations [62,63] as well as with the oracle-based approach [34,23].

2.3. Some observations

Exploiting sparsity of a $K \times K$ matrix can improve the efficiency of computations, such as from $\mathcal{O}(K^3)$ to $\mathcal{O}(K \operatorname{poly}\log K)$ [21], but in our case with dimension $K \cong 2^{\mathcal{O}(n)}$ cannot improve from exponential to polynomial time; similarly for proceeding from uniform $K \times \cdots \times K = K^d$ grids in dimension d to adaptive ones focusing on critical regions of lower dimension [44].

Remark 15.

a) Rational numbers with denominator 2^n can approximate any real number up to error $1/2^n$. Allowing unbounded binary denominators destroys complexity bounds [62, Examples 7.2.1+7.2.3]. In Definitions 9 and 14 we permit a polynomial 'slack' in denominator binary length $p(n)$.
b) Time polynomial in n suffices to output only numerators $a = a_{n,k}$ of length polynomial in $n+k$ in Definition 9c; hence a sequence r_k of vectors computable in polynomial time has dimension $\dim(r_k)$ and magnitude $\max_J |r_{k,J}|$ grow at most exponentially in k.
c) Common numerical difference schemes discretize the initial condition φ into a vector (sequence) $\varphi^{(h)}$ of dimension growing exponentially with the output precision parameter n. There is no way to input/output such 'huge' object entirely within polynomial time. Definition 14 therefore formalizes computing individual entries $\varphi^{(h)}_J$, with index $J \in \mathbb{N}$ given in binary, which thus does remain polynomial in n. More precisely, the sequence index k enters (like the precision parameter n) in 'unary' in the time/memory bound, while the index J enters in binary; see also the arguments to the polynomial p in Definition 14 and to ψ in Definition 13 above.
d) Recall that every computable real function is continuous [62, §4.3]; hence uniformly continuous on compact domains. And every polynomial time computable real function has a polynomial modulus of continuity [34, Theorem 2.19]. Definition 14a imposes a polynomial modulus of continuity p; and Definition 14b additionally lets that depend on the index k (in unary) of the vector function sequence. It is no loss of generality, and notationally convenient, to use the same polynomial bound for the modulus of continuity as for the denominator binary length in (a).

3. Exponential-size algorithmic algebra

In this Section we explain the main ideas of proving Theorem 4 and establish the related results concerning linear algebra of exponential dimension and raising polynomials to exponential powers.

3.1. Techniques and proof ideas

For the difference scheme approach (9), the computational problem is equivalent to: first raise a matrix $\mathbf{A}$ (or, rather, any desired one from a sequence of matrices $\mathbf{A}_{h(n)}$) of exponential dimension $K \sim 2^{O(n)}$ to an exponential power $M \sim 2^{O(n)}$; then multiply the intermediate result to a K-dimensional

sample vector $\varphi^{(h)}$ of the initial condition φ; and finally return an approximation to any desired entry with given index $\#J$ of the result vector $\left(\mathbf{A}_{h(n)}^{M(n)} \cdot \varphi^{(h(n))}\right)_J$ up to error $1/2^n$, $0 \leq J < K$. Naïvely the intermediate result matrices and vectors have exponential dimension, hence leading to complexity class EXPTIME.

On the other hand, according to Question 2, only one entry $\#J$ of the result is required; and both the difference-scheme matrix $\mathbf{A}_{h(n)}$ and the sampled initial vector $\varphi^{(h(n))}$ do not need to be stored, but by hypothesis any desired of its entries can be (re-)computed on-the-fly in P, whenever and however often required: only $J \in \{0, 1, \ldots, K-1\}$ and M are part of the input, given in binary with a linear number $\mathcal{O}(n)$ of bits.

The discrete counterpart to our real problem would ask for any desired entry of a polynomial time computable Boolean matrix of exponential dimension to exponential power; and this can be solved in PSPACE by Savitch's Algorithm. When applying the same approach to the integer or to the present real case, the hypothesis of **A** having bounded powers (Definition 7b) becomes crucial: to guarantee that the resulting entries do not blow up, nor do they require excessive initial precision in order to keep the rounding error propagation in check (details omitted). We thus have an improved, namely PSPACE algorithm, and establish Theorem 4a).

Regarding Theorem 4c), Subsection 3.6 reduces the problem of (recovering any desired entry of a) circulant matrix raised to an exponential power to that of (recovering any desired coefficient of a) polynomial raised to such power; in a way that relates modular bandwidth to degrees. Subsection 3.4 solves the latter problem in #P by reduction to Riemann integration via Cauchy's Differentiation Theorem. Linear polynomials can even be raised to exponential powers in P according to Proposition 19. Subsection 4.4 establishes Theorem 4b); and Subsections 4.2, 4.3 establish Theorem 4d), e), respectively.

3.2. Exponential-size inner product is #P-"complete"

In the following Lemma we determine the complexity classes capturing the bit-cost of exponential-size inner products of real vectors. This will be later applied to (i) vectors approximating the initial data of the considered PDEs on a grid and, (ii) vectors representing rows of the difference scheme matrices; see Subsection 4.1. The complexity of difference scheme matrix powering is subject of the next subsection.

Proposition 16.

a) *Given vectors* $\mathbf{u}, \mathbf{v} \in [-2^\ell, +2^\ell]^D \subseteq \mathbb{R}^D$, *their inner product* $\langle \mathbf{u}, \mathbf{v} \rangle \in [-D \cdot 4^\ell, +D \cdot 4^\ell]$ *can be computed in time polynomial in* $n + \ell + D$. *[Indeed* D *instead of* $\log D$*] In particular if* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^D$ *are fixed and polynomial-time computable (in* n*), then so is their inner product* $\langle \mathbf{u}, \mathbf{v} \rangle$.

b) *Let* $\mathbf{u}_k = (u_{k,J})_{0 \leq J < D_k}$ *and* $\mathbf{c}_k = (v_{k,J})_{0 \leq J < D_k}$ *denote two sequences of* D_k*-dimensional real vectors. Suppose that both real vector sequences are computable in #P in the sense of Definition 14a), and suppose that the integer sequence* (D_k) *is computable in* P_1 *in the sense of Definition 14c). Then the sequence of real numbers* $\langle \mathbf{u}_k, \mathbf{v}_k \rangle$ *is computable in #P.*

c) *Let* $\mathbf{f}_k, \mathbf{g}_k :\subseteq \mathbb{R}^d \to \mathbb{R}^{D_k}$ *denote sequences of vector functions computable in #P, where* D_k *denotes the sequence of their joint dimensions supposed to be computable in* P_1. *Then also the sequence* $\langle \mathbf{f}_k, \mathbf{g}_k \rangle = \sum_{J \leq D_k} f_{k,J} \cdot g_{k,J}$ *of their inner products is computable in #P.*

d) *Items (b) and (c) are almost tight: There exist polynomial-time computable integer vector sequences* $\mathbf{s}_k, \mathbf{r}_k$ *of dimension* $D_k = 2^{\mathrm{poly}(k)}$ *such that the integer sequence* $\langle \mathbf{s}_k, \mathbf{r}_k \rangle$ *is not computable in unary polynomial time (=real P) unless* $FP_1 = \#P_1$.

Recall from after Definition 14c) that the dimensions D_k may be exponential in k. Nevertheless, Proposition 16b+c) beat the complexity class EXPTIME, since all exponential-size inputs are fixed.

Proof of Proposition 16. a) By triangle inequality,

$$\begin{aligned} \sum\nolimits_j |u_j v_j - u'_j v'_j| &\leq \sum\nolimits_j |u_j| \cdot |v_j - v'_j| + |u_j - u'_j| \cdot |v'_j| \\ &\leq 2D \cdot 2^{\ell+1} \cdot 2^{-m} \;\leq\; 2^{-n} \end{aligned}$$

provided that $|v_j - v'_j|, |u_j - u'_j| \le 2^{-m}$ for $m \ge n + \ell + 2 + \log D$. Hence it suffices to calculate the inner product on input of component-wise dyadic approximations to $\mathbf{u}, \mathbf{v}$ up to error 2^{-m}. The numerators $a \in \mathbb{Z}$ of these dyadic approximations $a/2^m$ are integers of binary length $\mathcal{O}(m + \ell)$, and can be processed in the claimed runtime.

b) Let $\psi, \phi : \{0,1\}^* \to \mathbb{N}$ be in #P such that $\mathbf{u}_k$ is computable in P^{ψ} and $\mathbf{v}_k$ is computable in P^{ϕ}. Note that the componentwise sum $\mathbf{u}_k + \mathbf{v}_k$ and componentwise product $\mathbf{u}_k \cdot \mathbf{v}_k$ are computable in $\mathsf{P}^{\psi \oplus \phi}$, where

$$\psi \oplus \phi \,:\, \{0,1\}^* \ni 0 \circ \vec{w} \mapsto \psi(\vec{w}), \quad 1 \circ \vec{w} \mapsto \phi(\vec{w}) \in \mathbb{N}$$

again belongs to #P.
Now according to the proof of (a) it suffices to calculate the inner product $k \mapsto \langle \mathbf{a}_k, \mathbf{b}_k \rangle$ of integer vectors $\mathbf{a}_k$ and $\mathbf{b}_k$, numerators to componentwise approximations $\mathbf{a}_k/2^m$ and $\mathbf{b}_k/2^m$ up to error $1/2^m$ to $\mathbf{u}_k$ and $\mathbf{v}_k$ for $m \cong n + \ell + \log D \le \mathrm{poly}(n + k)$. Note that the integer sequence $\langle \mathbf{a}_k, \mathbf{b}_k \rangle = \#X_k - \#Y - k$ belongs to the complexity class $\mathsf{FP}^{\#\mathsf{P}}$ since

$$\begin{aligned} X_k := \big\{ (J, a, b) : 0 \le J < D_k, (0 \le a < a_{k,J} \wedge 0 \le b < b_{k,J}) \\ \vee (0 \le a < -a_{k,J}, 0 \le b < -b_{k,J}) \big\}, \\ Y_k := \big\{ (J, a, b) : 0 \le J < D_k, (0 \le a < -a_{k,J} \wedge 0 \le b < b_{k,J}) \\ \vee (0 \le a < a_{k,J}, 0 \le b < -b_{k,J}) \big\} \end{aligned}$$

are subsets of $\mathbb{N}^3$ decidable in $\mathsf{P}^{\#\mathsf{P}}$ by hypothesis.

c) Similarly to (b), now pointwise.

d) Let $\psi : \{1\}^* \to \mathbb{N}$ be $\#\mathsf{P}_1$-complete,

$$\psi(1^k) \,=\, \#\big\{ \vec{w} \in \{0,1\}^{\mathrm{poly}(k)} : 1^k \,0\, \vec{w} \in V \big\}, \quad V \in \mathsf{P} \ .$$

Define $\mathbf{r}_k \equiv \vec{1} \in \{1\}^{D_k}$ and $\mathbf{s}_k \in \{0,1\}^{D_k}$ by $\mathbf{s}_{J,k} := 1$ in case $1^k \,0\, \mathrm{bin}(J) \in V$ and $\mathbf{s}_{J,k} := 0$ in case $1^k \,0\, \mathrm{bin}(J) \notin V$. Both are obviously computable in (unary) time polynomial in k; and their inner product coincides with ψ. □

3.3. Exponential matrix powering is PSPACE-"complete"

In the following Lemma we determine the complexity classes capturing the bit-cost of raising exponential-size real matrices to exponential powers. This will be later applied to difference scheme matrices of dimension $\mathcal{O}(2^n)$, see Subsection 4.1.

Proposition 17.

a) *Given matrices $\mathbf{A}, \mathbf{B} \in [-2^{\ell}, +2^{\ell}]^{D \times D} \subseteq \mathbb{R}^{D \times D}$, their product $\mathbf{C} := \mathbf{A} \cdot \mathbf{B} \in [-D \cdot 4^{\ell}, +D \cdot 4^{\ell}]^{D \times D}$ can be computed in time polynomial in $n + \ell + D$. In particular if $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{D \times D}$ are polynomial-time computable, then so is their product.*

b) *Let $\mathbf{A}_k = (\mathrm{A}_{k,I,J})_{0 \le I,J < D_k}$ denote a sequence of $D_k \times D_k$-dimensional real matrices of bounded powers (Definition 7), computable in PSPACE in the sense of Definition 14a). Suppose that the integer sequence (D_k) of dimensions is computable in P_1 in the sense of Definition 14c). Furthermore fix P_1-computable sequence (K_k) of natural numbers. Then the sequence of powered real matrices $\mathbf{A}_k^{K_k}$ is computable in PSPACE.*

c) *Item (b) is tight: There exist polynomial-time computable 0/1 matrix sequences $\mathbf{A}_k$ of bounded powers of dimension $D_k = 2^{\mathrm{poly}(k)}$ such that the power matrix sequence $\mathbf{A}_k^{D_k}$ is not computable in polynomial time (=real P) unless P = PSPACE.*
The problem remains PSPACE-'hard' when restricting to a binary tensor product $\mathbf{A}_k = \mathbf{X}_k \otimes \mathbf{Y}_k$ of matrices $\mathbf{X}_k, \mathbf{Y}_k$ of constant (traditional) bandwidth each.

Note that, without the hypothesis of bounded powers, matrix exponentiation can result in entries blowing up super-exponentially and thus impossible to process in PSPACE.

Proof of Proposition 17. a) Express matrix product as D^2 inner products according to Proposition 16a).

b) By hypothesis the entries of all matrix powers $\mathbf{A}_k^K$ are uniformly bounded by 2^ℓ for some $\ell \in \mathbb{N}$. Then, according to the sensitivity analysis in the proof of Proposition 16a+b), the entries of a product $\mathbf{A}_k^L \cdot \mathbf{A}_k^M$ can be approximated up to error $1/2^n$ from approximations to the entries of $\mathbf{A}_k^L$ and $\mathbf{A}_k^M$ up to error $1/2^m$ each, for $m = n + \ell + 2 + \log D$. Now recall powering by repeated-squaring:

$$\left(\mathbf{A}^K\right)_{I,J} = \begin{cases} \left(\left(\mathbf{A}^{K/2}\right)^2\right)_{I,J} = \sum_L \left(\mathbf{A}^{K/2}\right)_{I,L} \cdot \left(\mathbf{A}^{K/2}\right)_{L,J} & \text{if } K \text{ is even} \\ \left(\mathbf{A}^{K-1} \cdot \mathbf{A}\right)_{I,J} = \sum_L \left(\mathbf{A}^{K-1}\right)_{I,L} \cdot \left(\mathbf{A}\right)_{L,J} & \text{if } K \text{ is odd } . \end{cases} \tag{13}$$

As recursive algorithm, it has depth $\mathcal{O}(\log K)$; and since the increase of precision at each recursive step is by an additive amount $\ell + 2 + \log D$, the lower levels of the recursion tree will suffice with precision $\mathcal{O}\big(n + \log(K) \cdot (\ell + 2 + \log D)\big) \le \mathrm{poly}(n + \log K + \ell + \log D)$. Now recall that P_1-computability of D_k and K_k implies $\log(D_k), \log(K_k) \le \mathrm{poly}(k)$.

Moreover recursion allows to reuse and thus save memory: Given K, I and J, the formula incurs D sequential recursive calls to calculate $A_{I,L}^{K/2}$ and possibly $A_{L,J}^{K/2}$; each time storing only the current accumulator. It is clear that each recursion level adds the memory to store the number (already studied before) and the loop index. In other words, the recursion algorithm calculates not the entire matrix but just one coefficient, recall Definition 14a).

This incurs many re-calculations in exponential runtime, but does yield a memory bound polynomial in $n+k$: Recall from the proof of Proposition 16a) that each real matrix entry is stored as numerator of binary length $\mathcal{O}(m+\ell)$ to dyadic approximation up to error $1/2^m$, $m \cong n+\ell+2+\log D$, where $D = D_k$ is at most exponential in k. Also the loop index $L < D_k$ can be stored in $\mathrm{poly}(k)$ bits.

c) We perform a master reduction and consider a Turing machine with polynomial space bound $\mathrm{poly}(k)$, hence exponential runtime bound $K_k := 2^{\mathrm{poly}(k)}$ and exponential bound $D_k := 2^{\mathrm{poly}(k)}$ on the number of configurations. Let $\mathbf{A}_k$ denote the adjacency matrix of the directed configuration graph of said machine, with entry 1 at position (I, J) if J is the successor configuration to I and 0 otherwise. Hence $\left(\mathbf{A}_k^K\right)_{I,J}$ contains the number of paths (computations) of length K from configuration I to J. Since our Turing machine is deterministic, there can be at most such computation; in particular $\mathbf{A}_k$ has bounded powers. And since the runtime is at most exponential, $\left(\mathbf{A}_k^K\right)_{I_0,J_0}$ tells whether there exists a computation at all from start configuration I_0 to the (w.l.o.g. unique) accepting end configuration J_0.

For the strengthened claim, recall that a two-counter machine with only instructions

```
INC1, INC2, DEC1, DEC2, JZ1, JZ2
```

is Turing-complete [47, Theorem Ia]. The proof furthermore reveals that such machines, although possibly exponentially slower than a Turing machine, are polynomially equivalent with respect to their memory consumption. Moreover also a polynomially space-bounded counter machine can make at most exponentially steps before repeating. It thus suffices to express configuration changes of counter (instead of Turing) machines as matrices $\mathbf{A}_k$ of dimension exponential in k. Now observe that configuration changes of the first counter are captured by a tridiagonal matrix $\mathbf{X}_k$, corresponding to instructions `INC1` and `DEC1`; similarly $\mathbf{Y}_k$ for the second counter; and configuration changes corresponding to the instruction index (i.e. `JZ1` and `JZ2`) are captured by a constant-size (and hence constant-bandwidth) matrix $\mathbf{Z}$. So $\mathbf{A}_k := \mathbf{X}_k \otimes \left(\mathbf{Y}_k \otimes \mathbf{Z}\right)$ suffices. □

3.4. Raising small polynomials to exponential powers

Consider the problem of raising a fixed polynomial-time computable univariate polynomial P of 'small' degree to an exponential M-th power, and read off the J-th coefficient $P[X^J]$ with $J, M \in \mathbb{N}$ in binary as only actual input. Of course having bounded powers (Definition 8c) is again a crucial hypothesis to prevent coefficient explosion and numerical instability. Note that already for the quadratic polynomial $P(X) = (1 + X + X^2)/3$, naïve evaluation of the 'explicit' formula

$$\left(\tfrac{1}{3} + \tfrac{1}{3}X + \tfrac{1}{3}X^2\right)^K [X^J] \;=\; 3^{-K} \cdot \sum_{\substack{0 \le \mu, \nu \le K \\ \mu + 2\nu = M}} \frac{K!}{\mu! \cdot \nu! \cdot (K - \mu - \nu)!} \tag{14}$$

involves terms like $K!$ of value, and the sum with a number of terms, doubly exponential in k: not at all obvious to compute in #P.

However recall Cauchy's Differentiation Formula for analytic functions, for notational convenience recalled here in the univariate case:

$$f(z) \;=\; \sum_j c_j z^j \quad \Rightarrow \quad c_j \;=\; f^{(j)}/j! \;=\; \int_{\partial\, \mathrm{B}(0,r)} \frac{f(z)}{z^{j+1}}\, dz \;/\; (2\pi i) \tag{15}$$

for any $r > 0$ such that $\mathrm{B}(0, r) \subseteq \mathrm{dom}(f)$. Here $\mathrm{B}(z, r) = \{u : d(u, z) < r\}$ denotes the open ball with center z and radius r, and ∂ is the topological boundary operator.

Applying Equation (15) to the analytic polynomial $f := P^M$ with $r = 1$ allows to recover each of the exponentially (in n) many coefficients $P^M[X^j]$ of P^M, individually. Note that evaluating $z \mapsto P^M(z)$ is feasible via repeated squaring using $\mathcal{O}(\log M)$ multiplications, that is polynomial in the precision n. Turning (the analysis of) this algebraic into a numerical algorithm, note that said powering of complex numbers (rather than of symbolic polynomial) is stable: since P by hypothesis has bounded powers and $|z| = r = 1$, if follows that $|P^M(z)|/|z^{j+1}|$ is bounded independently of M. Moreover differentiation

$$\frac{d}{dz} P^M(z) \;=\; M \cdot P^{M-1}(z) \cdot P'(z) \tag{16}$$

shows that also the dependence of $P^M(z)/z^{j+1}$ on z under small perturbations is well-behaved: To attain absolute output error $1/2^n$ and for $M, j, |P'(z)| \le 2^{\mathrm{poly}(n)}$, z and $P(z)$ suffice to be known and processed with $\mathrm{poly}(n)$ additional bits of precision; hence $z \mapsto P^M(z)/z^{j+1}$ can be computed in polynomial time, provided that the original coefficients of P are. Finally the path integral in Equation (15) boils down in canonical parameterization to an ordinary definite Riemann integral, and the latter is known computable in $\#\mathsf{P}_1$ [34, §5.4].

These considerations straightforwardly generalize from the univariate to the multivariate setting, provided that the number of variables is constant w.r.t. n. A little more delicate is our 'semi-uniform' case of sequences of polynomials P_k according to Definition 14: With coefficient indices $\vec{j}$ given in binary (in addition to the sequence index k and output precision n in unary), the thus parameterized Riemann integration problem climbs in complexity from $\#\mathsf{P}_1$ to #P [34, Theorem 5.32]. Note that the derivatives $P_k'(z)$ in Equation (16) grow at most with $\deg(P_k) \le 2^{\mathrm{poly}(n)}$ according to Bernstein's Theorem. In summary we have established the following theorem:

Theorem 18. *Fix a sequence $P_k(\vec{X})$ of L-variate polynomials of componentwise degree $< \vec{D}$ and bounded powers. Let K_k denote a sequence of natural numbers with binary representation computable in time polynomial (and thus of value exponentially bounded) in k.*

Then each coefficient $P_k^{K_k}[\vec{X}^{\vec{j}}]$ of $P_k^{K_k}$, $\vec{0} \le \vec{j} < \vec{D}$, formalized as mapping $\mathrm{bin}(1^k, \mathrm{bin}(\vec{j})) \mapsto P_k^{K_k}[\vec{X}^{\vec{j}}]$, is computable in #P.

Note that $P_k^{K_k}$ has componentwise degree $< \vec{D} \cdot K^k$, growing at most exponentially in k. Therefore the binary length of $\vec{j}$ is polynomial in k, which in turn is the binary length of 1^k.

Cauchy's Differentiation Formula (15) does not apply to Laurent polynomials, due to possible singularities in the unit circle. Nevertheless, Theorem 18 does extend to Laurent polynomials P'_k: Multiplying P' with a suitable (say $\vec{j}_0$-th) power of $\vec{X}$ converts to an ordinary polynomial $P := \vec{X}^{\vec{j}_0} \cdot P'$; and simultaneously adjust the indices $\vec{j} \mapsto \vec{j} + K_k \cdot \vec{j}_0$ in Theorem 18.

3.5. Powering linear polynomials in P

Raising linear polynomials to exponential powers is feasible in P: Note that $\binom{N}{K} p^K q^{N-K}$ is the coefficient of X^K of the polynomial $(pX+q)^N$.

Proposition 19. *Fix polynomial-time computable $q \in (0;1)$ and $k \in \mathbb{N}$. Abbreviate $p := 1-q$. Given $K \le N \in \mathbb{N}$, one can approximate $\binom{N}{K} p^K q^{N-K}$ to absolute error N^{-k} in time polynomial in $\log N$, the binary length of N.*

To emphasize, k is constant while K, N are given as inputs.

Proof of Proposition 19. Without loss of generality, assume $p \le 1/2$. Interpret this as a Bernoulli trial. Let X be a random variable equal to the number of successes among n independent Bernoulli trials with the probability of success being p. Then the probability $P(X=K) = \binom{N}{K} p^K q^{N-K}$. The mean μ of X is pN. Applying the multiplicative Chernoff bound [8] with $\varepsilon = 1/2$, we obtain

$$P\left(X \le \frac{\mu}{2}\right) \le e^{-\frac{\mu}{8}} \qquad P\left(X \ge \frac{3\mu}{2}\right) \le e^{-\frac{\mu}{12}}$$

Therefore,

$$P\left(|X-\mu| \ge \frac{\mu}{2}\right) \le 2e^{-\frac{\mu}{8}}.$$

Suppose $K \notin \left(\frac{pN}{2}, \frac{3pN}{2}\right)$, then $|K-\mu| \ge \frac{\mu}{2}$. Thus $P(X=K) \le P\left(|X-\mu| \ge \frac{\mu}{2}\right)$.

For large enough N, we have $2e^{-\frac{\mu}{8}} = 2e^{-\frac{pN}{8}} \le N^{-k}$, so the algorithm will output 0 as it is within the allowed margin of error.

Otherwise, K is in the interval $\left(\frac{pN}{2}, \frac{3pN}{2}\right)$. Express $\binom{N}{K} = \frac{N!}{(N-K)!K!}$. The expanded Stirling's approximation [65, Equations (1) and (6)] up to sufficiently many (depending on k, and thus a constant number of) terms yields an approximation to $K \mapsto K!$ up to relative error

$$BK^{-k-1} \le \frac{2^{k+1}B}{p^{k+1}} N^{-k-1}$$

for some $B = B(k) \in \mathbb{N}$; and evaluating this expansion takes time polynomial in $\log K$. Similarly, we can compute $N!, (N-K)!, a^K, b^{N-K}$ to the same relative error bound. Therefore $\binom{N}{K} a^K b^{N-K}$ can be computed by multiplying the results and their inverses together, which gives the result to relative error N^{-k}. Since $0 \le \binom{N}{K} a^K b^{N-K} \le 1$, our output is also of absolute error N^{-k}. All of this can be done in time polynomial in $\log N$. □

3.6. Circulant matrices as polynomials

Consider an $N \times N$ circulant matrix with parameters $\vec{a} = (a_0, \ldots, a_{N-1})$:

$$\mathcal{C}(\vec{a}) = \left(a_{I-J \bmod N}\right)_{0 \le I,J < N} = \sum\nolimits_{J=0}^{N-1} a_J \cdot \mathcal{C}_N^J \tag{17}$$

for the 'generator' circulant matrix $\mathcal{C}_N := \mathcal{C}(0,1,0,0,\ldots,0)$ with N parameters $(0,1,0,0,\ldots,0)$. Let

$$P_{\mathcal{C}(\vec{a})}(X) = \sum\nolimits_{j=0}^{N-1} a_j X^j \tag{18}$$

denote the *associated polynomial* to circulant matrix $\mathcal{C}(\vec{a})$. Note that $\mathcal{C}(\vec{a}) = P_{\mathcal{C}(\vec{a})}(\mathcal{C}_N)$; moreover the entry of the matrix power $\mathcal{C}(\vec{a})^M$ with indices (I, J) is

$$\left(\mathcal{C}(\vec{a})^M\right)_{I,J} = \sum_{K \equiv I-J \bmod N} P^M_{\mathcal{C}(\vec{a})}[X^K]$$

in the terminology of Definition 8. The associated polynomial is a well-known concept [39].

Recall that $\mathcal{R}_D^{N\times N}$ denotes the vector space of $N \times N$ matrices over $\mathcal{R}$ of modular bandwidth $< D$, Definition 7c). Circulant such matrices correspond to *Laurent* polynomials with degree $< D$: linear combinations of monomials X^j of possibly negative exponents j between $-D$ and $+D$, as formalized in the following lemma.

Lemma 20. *Fix a (not necessarily commutative) ring $\mathcal{R}$ of characteristic zero. Let $\mathrm{CIRC}_D^N(\mathcal{R}) \subseteq \mathcal{R}_D^{N\times N}$ denote the subspace of $N \times N$* circulant *matrices of modular bandwidth $< D$; $\mathrm{CIRC}_D := \bigcup_N \mathrm{CIRC}_D^N(\mathbb{R})$. Recall that $\mathcal{C}_N \in \mathrm{CIRC}_2^N(\mathcal{R})$ denotes the $N \times N$ cyclic permutation matrix from (17), and recall that we write $P[\vec{X}^{\vec{j}}] \in \mathcal{R}$ for the coefficient to $\vec{X}^{\vec{j}} = X_1^{j_1} \cdots X_L^{j_L}$ in an L-variate Laurent polynomial $P \in \mathcal{R}[\vec{X}, \vec{X}^{-1}]$, $\vec{j} \in \mathbb{Z}^L$.*

a) *For any $P \in \mathcal{R}_D[X, X^{-1}]$ it holds $P(\mathcal{C}_N) \in \mathrm{CIRC}_D^N(\mathcal{R})$. More precisely*

$$P(\mathcal{C}_N)_{I,J} = \sum_{n\in\mathbb{Z}} P[X^{J-I+nN}] \in \mathcal{R},$$

and the sum is finite. The mapping $\mathcal{R}[X, X^{-1}] \ni P \mapsto P(\mathcal{C}_N) \in \mathrm{CIRC}^N(\mathcal{R})$ is a homomorphism of (not necessarily commutative) algebras. For $D \le N/2$ and normed $\mathcal{R}$, the restriction $\mathcal{R}_D[X, X^{-1}] \ni P \mapsto P(\mathcal{C}_N) \in \mathrm{CIRC}_D^N(\mathcal{R})$ is an isometry of vector spaces with respect to the induced norms from Definitions 8c) and 7a).

b) *First specializing (a) to $\mathcal{R} := \mathbb{R}^{m\times m}$ and then generalizing from the univariate to the L-variate case, let us fix $m, L, N_1, \ldots, N_L \in \mathbb{N}$ and consider the ring homomorphism*

$$\Phi_{m,\vec{N}} : \mathbb{R}^{m\times m}[X_1, X_1^{-1}, \ldots, X_L, X_L^{-1}] \to \mathrm{CIRC}^{N_1} \otimes \cdots \otimes \mathrm{CIRC}^{N_L} \otimes \mathbb{R}^{m\times m}$$
$$\mathbf{B}_{\vec{j}} \cdot X_1^{j_1} \cdots X_L^{j_L} \mapsto \mathcal{C}_{N_1}^{j_1} \otimes \cdots \otimes \mathcal{C}_{N_L}^{j_L} \otimes \mathbf{B}_{\vec{j}}$$

where $\otimes$ denotes Kronecker/tensor product. Then $\left(\Phi_{m,\vec{N}}(P)\right)_{I_1,J_1,\ldots,I_L,J_L} =$

$$= \sum_{\vec{n}\in\mathbb{Z}^L} P\left[X_1^{J_1-I_1+n_1N_1}, \ldots X_L^{J_L-I_L+n_LN_L}\right] \in \mathbb{R}^{m\times m} \tag{19}$$

and the sum is finite. Moreover, for $D_\ell \le N_\ell/2$, the restriction

$$\Phi_{m,\vec{N}} : \mathbb{R}^{m\times m}_{\vec{D}}[\vec{X}, \vec{X}^{-1}] \to \mathrm{CIRC}_{D_1}^{N_1} \otimes \cdots \otimes \mathrm{CIRC}_{D_L}^{N_L} \otimes \mathbb{R}^{m\times m}$$

is well-defined and isometric and bijective.

Proof of Lemma 20. a) One can see easily by induction that $(\mathcal{C}_N^j)_{I,J} = 1$ if $J - I \equiv j \pmod N$, otherwise 0. Therefore,

$$\begin{aligned} P(\mathcal{C}_N)_{I,J} &= \sum_{j\in\mathbb{Z}} (\mathcal{C}_N^j)_{I,J} P[X^j] \\ &= \sum_{j\not\equiv J-I \pmod N} 0 \cdot P[X^j] + \sum_{j\equiv J-I \pmod N} 1 \cdot P[X^j] \\ &= \sum_{n\in\mathbb{Z}} P[X^{J-I+nN}]. \end{aligned}$$

And the sum is not only finite but has merely one term, since $\deg(P) < D \le N/2$. Note that all powers of $\mathcal{C}_N$ are permutation matrices; and if it holds $-N/2 < I, J < N/2$ and $I \ne J$, then $\mathcal{C}_N^I$

and $\mathcal{C}_N^J$ do not share the positions of any 1's. In this case each coefficient of P therefore appears exactly once in each row (and in each column) of $P(\mathcal{C}_N)$; and for each entry of $P(\mathcal{C}_N)$, at most one term of P contributes to it. Thus the operator norm of the matrix $P(\mathcal{C}_N)$ is the same as the 1-norm of any row (and any column) of it, which is equal to $\|P\|$.

b) First note that $\Phi_{m,\vec{N}}$ is well-defined and multiplicative. Next, $\Phi_{m,\vec{N}}$ maps precisely all monomials of the form $X_1^{j_1+n_1N_1}\cdots X_L^{j_L+n_LN_L}$ to the same $\mathcal{C}_{N_1}^{j_1}\otimes\cdots\otimes\mathcal{C}_{N_L}^{j_L}\circ\mathbb{I}_m$. In particular, for $-N_\ell/2\le -D_\ell < j_\ell < +D_\ell \le +N_\ell/2$, this mapping from monomials to tensor products of powers of cyclic permutation matrices is injective, and surjective; hence the claim follows by linearity. □

For the complexity investigations in Theorem 24, consider the following stratification of (sequences in) $\mathrm{CIRC}_{D_1}\otimes\cdots\otimes\mathrm{CIRC}_{D_L}\otimes\mathbb{R}^{m\times m}$:

Definition 21 *(Circulant bandwidth matrix sequence).* For fixed $m, L, J, D_1, \ldots, D_L \in \mathbb{N}$, call a sequence of real square matrices $\mathbf{A}_k$ of varying dimensions *circulant bandwidth* of format $(m, L, J, \vec{D})$ if there exists

- a family $C_{k,\ell,j}$ of real circulant matrices of constant modular bandwidth $< D_\ell$ and varying dimensions $N = N(k,\ell,j)$, where $-J < j < J$ and $1 \le \ell \le L$
- a family $B_{k,j}$ of real square matrices of constant dimension m, $(-J < j < J,\ 1 \le \ell \le L)$, such that

$$\mathbf{A}_k \;=\; \sum\nolimits_{j=-J+1}^{J-1} C_{k,1,j}\otimes C_{k,2,j}\otimes\cdots\otimes C_{k,L,j}\otimes B_{k,j}\ . \tag{20}$$

Sequence $\mathbf{A}_k$ is *circulant bandwidth* if it is circulant bandwidth of format $(m, L, J, \vec{D})$ for some $m, L, J, \vec{D}$. We say that Equation (20) is in *normal form* if all circulant matrices are suitable powers of cyclic permutation matrices $\mathcal{C}_N$ (the 'generators') of suitable dimensions N; more formally, if it holds

$$\forall k,\ell,j:\ C_{k,\ell,j} \;=\; \mathcal{C}^{j}_{N(k,\ell,j)}\ .$$

Remark 23 below motivates circulant bandwidth matrices as difference schemes arising from a rich class of PDEs; and the below Theorem 24 asserts that such matrices can be raised to exponential powers in #P rather than PSPACE.

Lemma 22. *Circulant bandwidth matrices are closed under binary addition, multiplication, and tensor product; and can be brought to normal form:*

a) *The sum of two circulant bandwidth matrix sequences of formats* $(m, L, J, \vec{D})$ *and* $(m, L, J', \vec{D})$*, respectively, is a circulant bandwidth matrix sequence of format* $(m, L, J + J', \vec{D})$.

b) *Componentwise multiplication of two circulant bandwidth matrix sequences of format* $(m, L, J, \vec{D})$ *and of format* $(m, L, J', \vec{D}')$*, respectively, yields a circulant bandwidth matrix sequence of format* $(m, L, 2J\cdot J', \vec{D}+\vec{D}')$.

c) *The tensor product of two circulant bandwidth matrix sequences of format* $(m, L, J, \vec{D})$ *and of format* $(m', L', J', \vec{D})$*, respectively, yields a circulant bandwidth matrix sequence of format* $\big(m+m', L+L', 2J\cdot J', (\vec{D},\vec{D}')\big)$.

d) *Any circulant bandwidth matrix sequence of format* $(m, L, J, \vec{D})$ *can be converted into one in normal form of format* $(m, L, |\vec{D}|, \vec{D})$.

Proof. a) Obvious.

b) The product $C_{k,\ell,j}\cdot C'_{k,\ell,j}$ of two circulant matrices $C_{k,\ell,j}$ and $C'_{k,\ell,j}$ is again circulant. If $C_{k,\ell,j}$ has modular bandwidth $< D_\ell$ and $C'_{k,\ell,j}$ has modular bandwidth $< D'_\ell$, then $C_{k,\ell,j}\cdot C'_{k,\ell,j}$ has modular bandwidth $< D_\ell + D'_\ell - 1$. The product of two sums $\sum_{j=-J+1}^{J-1}$ and $\sum_{j'=-J'+1}^{J'-1}$ of $2J-1$ and $2J'-1$ terms, respectively, is a sum of $2\cdot(2JJ'-J-J')+1 \le 2\cdot(2JJ')-1$ terms.

c) Similarly to (b).
d) Expand circulant $C_{k,\ell,j}$ of modular bandwidth $< D_\ell$ as linear combination of $\mathcal{C}^{-D_\ell+1}_{N(k,\ell,j)}, \ldots, \mathcal{C}^{D_\ell-1}_{N(k,\ell,j)}$. □

3.7. Powering circulant bandwidth matrices in #P

In difference schemes for PDEs, spatial discretization replaces continuous functions with sample vectors $\mathbf{u}$ and operators with matrices $\mathbf{A}_n$ of dimension exponential in n (corresponding to grid width $h \cong 1/2^{\mathrm{poly}(n)}$). The latter are then to be raised to a power exponential in n (corresponding to time steps $\tau \cong 1/2^{\mathrm{poly}(n)}$). According to Subsection 3.3, such powering has computational complexity PSPACE in general, even for matrices of bounded powers. But many difference schemes are circulant bandwidth in the sense of Definition 21:

Remark 23. Consider a system of evolutionary linear partial differential equations (1) and periodic boundary conditions (6) on the d-dimensional unit cube $\Omega = [0;1)^d \bmod \vec{1}$, where $\mathbf{B}_{\vec{j}} \in \mathbb{R}^{e\times e}$ are constant coefficient matrices, i.e., they may *not* depend on $\vec{x}$ or t.

a) The partial differential operator ∂_ℓ applied to f is naturally approximated up to error $1/2^k$ on a spatial grid $(x_j)_j$ of width $\delta = 2^{-\mathcal{O}(k)}$ by a first-order finite difference like $\big(f(x_j+\delta) - f(x_j-\delta)\big)/\delta$. The latter amounts to a matrix of bandwidth 3 applied to the sample vector $\big(f(x_j)\big)_j$.
b) Similarly, finite differences of higher (but constant) order accuracy to higher (but constant) order-j derivatives ∂^j_ℓ both correspond to matrices of larger (but constant) bandwidth. And mixed partial derivatives correspond to tensor products of such matrices.
c) Differential operators of bounded order with constant (with respect to time, but not necessarily spatially) coefficients yield constant such matrices.
d) Differential operators with coefficients which depend neither on time nor on space, together with periodic spatial boundary conditions, finally yield circulant such matrices.

Section 4.1 expands on this remark, and Section 5 collects some explicit examples. And the following Theorem 24 asserts that such difference schemes can be raised to exponential powers in #P (instead of PSPACE).

Theorem 24. *Let $\mathbf{A}_k$ have bounded powers and be of circulant modular bandwidth with terminology from Definition 21. Furthermore suppose*

- *that the integer triple sequence $N = N(k,\ell,j) > 0$ is computable in unary polynomial time (Definition 14c),*
- *that the real matrix triple sequence $(k,\ell,j) \mapsto C_{k,\ell,j}$ of dimension $N(k,\ell,j)$ is polynomial-time computable (Definition 14a),*
- *and that the real matrix double sequence $(k,j) \mapsto B_{k,j}$ of dimension m is polynomial-time computable.*
- *Finally consider a sequence of positive integer exponents $M = M(k)$ computable in unary polynomial time.*

Then the sequence $\mathbf{A}_k^{M(k)}$ of real matrix powers is computable in #P.

Recall (remark after Definition 14) that N and M can grow up to exponentially in k.

Proof. Recall that in our non-uniform setting the task is to compute any desired individual among the exponentially entries!

First translate the matrix $\mathbf{A}_k$ of Equation (20) into normal form according to Lemma 22d). Note that expanding each $C_{k,\ell,j}$ individually as linear combination of $\mathcal{C}^{-D_\ell+1}_{N(k,\ell,j)}, \ldots, \mathcal{C}^{D_\ell-1}_{N(k,\ell,j)}$ incurs at most $|\vec{D}| = D_1 + \cdots + D_L$ additional terms to then be re-collected into the coefficient matrices $B_{k,j}$; and $|\vec{D}|$ is constant with respect to k.

Next translate the normal form into a multivariate Laurent polynomial P_k according to the inverse isometry from Lemma 20b).

Then raise P_k to the desired power $M(k)$: within #P following Theorem 18.

And finally recover the desired coefficient of $\mathbf{A}_k^{M(k)}$ from the corresponding one in $P_k^{M(k)}$ according to Lemma 20b). □

Replacing Theorem 18 with Proposition 19, we obtain the following corollary which may be of independent interest:

Corollary 25. *If all circulant matrices $C_{k,j,\ell}$ in Equation (20) have modular bandwidth two, then the sequence $\mathbf{A}_k^{M(k)}$ of matrix powers is computable in P.*

Difference schemes of modular bandwidth two correspond to simple transport equations, which admit explicit solutions anyway: see Subsection 4.2. Also, fast exponentiation A^M of a difference scheme A by itself is insufficient for the difference method approach as it remains to apply the inner product with the initial condition in #P: see Proposition 16.

4. Complexity of PDEs

In this section we prove Theorem 4: the proofs of items a), c) are based on the difference scheme approach and heavily rely on the techniques developed in Section 3; item d) is proved via the characteristic method; the hardness result e) significantly uses complexity properties of integration (Fact 11).

4.1. Linear PDEs with non-/constant coefficients in PSPACE/#P

By Proposition 17, raising a difference scheme of exponential dimension to an exponential power (=making that many time steps of the autonomous system) is feasible in PSPACE; and multiplying the resulting matrix to the sample vector of the initial condition remains in PSPACE.

If the underlying difference scheme is *circulant bandwidth (Definition 21), then it can be raised to an exponential power within #P by Theorem 24; and multiplying the resulting matrix to the sample vector of the initial condition remains in #P according to Proposition 16.*

To finalize proofs of Claims (a) and (c) of Theorem 4, it remains to interpolate the grid function computed in Section 3 and apply several well known estimates.

Proof of Theorem 4a, c. To compute the solution $\mathbf{u}$ 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 the space and time grid steps, h and τ, in the following way:
   - h is any binary-rational number of the form $h = 2^{-N}$, where $N = \mathcal{O}(n)$, satisfying the Inequality (22) below;
   - τ is any binary-rational number meeting the Courant inequality $\tau \le \nu h$, where ν is the Courant number guaranteeing convergence property of the considered difference scheme which can be computed from the coefficients of the system (1).
2.) For a grid point $(t,\vec{x})$ put $l = \frac{t}{\tau}$ (note that $l \le M = \left[\frac{1}{\tau}\right] = \mathcal{O}(2^n)$) and calculate the matrix powers and vector products $\mathbf{A}_{h(n)}^{M(n)} \cdot \varphi^{(h(n))}$. Note that we use matrix powering (Subsection 3.3) instead of the step-by-step iterations commonly employed in connection with difference schemes (9).
3.) For non-grid points take (e.g.) a multilinear interpolation $\widetilde{u^{(h)}}$ of $u^{(h)}$, which can be computed in polynomial time from the (constant number of) "neighbor" grid points. Due to well-known properties of multilinear interpolations, the following inequalities hold:

$$\sup_{t,\vec{x}} \left|\widetilde{\mathbf{u}^{(h)}}(t,\vec{x})\right| \le \tilde{C} \sup_{G_N^\tau} \left|u^{(h)}\right|; \quad \sup_{t,\vec{x}} \left|\mathbf{u}(t,\vec{x}) - \widetilde{\mathbf{u}|_{G_N^\tau}}(t,\vec{x})\right| \le \bar{C} \sup_{t,\vec{x}} \left|\partial^{\vec{2}}\mathbf{u}(t,\vec{x})\right| \cdot h^2, \qquad (21)$$

where $\tilde{C}$ and $\bar{C}$ are absolute constants.

Based on (21) and on the continuous dependence property, as well as on linearity of the interpolation operator, infer

$$\sup_{t,\vec{x}} \left|\mathbf{u}(t,\vec{x}) - \widetilde{u^{(h)}}(t,\vec{x})\right| \leq \sup_{t,\vec{x}} \left(\left|\mathbf{u}(t,\vec{x}) - \widetilde{\mathbf{u}|_{G_N^\tau}}(t,\vec{x})\right| + \left|\widetilde{\mathbf{u}|_{G_N^\tau}}(t,\vec{x}) - \widetilde{u^{(h)}}(t,\vec{x})\right| \right)$$
$$\leq \tilde{C}C_0 ||\varphi||_{C^l(\bar{\Omega})} h^2 + \tilde{C}C \cdot h \leq 2^{-n}.$$

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

$$h \leq C_h \cdot 2^{-n}, \quad C_h = \tilde{C}C_0 ||\varphi||_{C^l(\bar{\Omega})} + \tilde{C}C, \tag{22}$$

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 polynomial time computable functions φ, B_i and therefore is a fixed constant). □

Remark 26. Dating back to the Heat Equation, Fourier methods are nowadays a common tool in the analysis of PDEs—although arguably less common in their numerical treatment. Calculating a Fourier integral (over a compact domain) as transformation between spatial and frequency representation amounts to algorithmic cost #P according to Fact 11c). This observation suggests an alternative approach to establish (variants of) Theorem 4c).

4.2. Symmetric hyperbolic PDEs with commuting coefficients are polynomial-time computable

Towards proving Theorem 4d), first consider the following *scalar* linear PDE with constant coefficients and its explicit solution

$$\lambda_0 u_t = \sum\nolimits_j \lambda_j \partial_j u, \qquad u(\vec{x},t) = u_0\big(t \sum\nolimits_j \lambda_j + \lambda_0 \sum\nolimits_j x_j\big) \tag{23}$$

which is obviously polynomial-time computable, provided that $\lambda_0, \lambda_1, \ldots, \lambda_d \in \mathbb{R}$ are.

Theorem 4d) considers *vector systems* of such PDEs, with constant symmetric mutually commuting matrix coefficients $\mathbf{B}_{\vec{j}}$. These hypotheses assert a simultaneous diagonalization, that is, a basis of joint eigenvectors: which 'decouples' the system into e independent scalar equations (23). It remains to prove that such a joint spectral decomposition can be computed in P: which is wrong in case (even a single symmetric) matrix $\mathbf{B}_{\vec{j}}$ is given as input [66]. Fortunately, in agreement with Question 2, these matrices are not part of the input but fixed polynomial-time computable. And for this case we have and apply the following

Theorem 27. *Fix symmetric and mutually commuting matrices $\mathbf{B}_{\vec{j}} = \mathbf{B}_{\vec{j}}^*$ of fixed dimension $e \in \mathbb{N}$ with fixed real polynomial-time computable entries. Then a joint spectral decomposition $\mathbf{B}_{\vec{j}} = \mathbf{T}^\intercal \cdot \mathbf{D}_{\vec{j}} \cdot \mathbf{T}$ can be computed in time polynomial in the output precision parameter n (and unspecified dependence on the dimension e as well as on the 'input' matrix entries). Here $\mathbf{D}_{\vec{j}}$ is a diagonal matrix with the eigenvalues of $\mathbf{B}$; and $\mathbf{T}$ is an orthonormal matrix consisting of a joint basis of eigenvectors of all $\mathbf{B}_{\vec{j}}$.*

Proof. We first treat the case of one single symmetric matrix $\mathbf{B}_{\vec{j}} = B$. By [48], the e-tuple of eigenvalues, repeated according to their multiplicities (i.e. the sought diagonal matrix D), can be approximated up to absolute error $1/2^n$ in time polynomial in n. Note that the (integer) multiplicities themselves depend discontinuously on the entries of B and therefore cannot be computed from B [67, §3.5]; but, since the matrix is fixed, they can be hardcoded into the thus non-uniform computation. It then remains to compute, for each eigenvalue λ with correct multiplicity $k = k(\lambda)$, an orthonormal basis of the k-dimensional kernel of $B - \lambda \cdot \mathrm{id}$ [50].

Gaussian Elimination employs tests for in/equality during pivot search, which are 'discouraged' numerically and actually known undecidable in the rigorous real realm. Instead, we shall also hardcode the results of these tests: Recall that Gauss makes at most $\mathcal{O}(e^3)$ steps, including tests. In order to

obtain runtime polynomial in the variable precision n (instead of the constant dimension e), proceed from Gauss real to Bareiss integer algorithm. More precisely operate on some $e \times e$ matrix $\tilde{B}_{m,\lambda}$ of integer numerators such that dyadic $\tilde{B}_{m,\lambda}/2^m$ approximates $B - \lambda \cdot \mathrm{id}$ 'sufficiently' well. Bareiss is known to perform a number of integer arithmetic operations, and to incur bit-cost, at most polynomial in the (constant dimension and in the variable) binary length of the largest integer entry. The latter is $\leq m$, plus the (constant) magnitude of the largest real entry. In particular it suffices to take m polynomially larger than n to guarantee output approximations up to error $1/2^n$.

Regarding the case of several symmetric commuting matrices $\mathbf{B}_{\vec{\jmath}}$, similarly hardcode the dimensions $k = k(\vec{\lambda})$ of their joint eigenspaces, namely to each eigenvector-tuple $\vec{\lambda} = \lambda_{\vec{\jmath}}$; and for each such tuple find a basis of the joint kernel of all square matrices $\mathbf{B}_{\vec{\jmath}} - \lambda_{\vec{\jmath}}$ as the kernel of a single rectangular matrix comprised of the aforementioned $\mathbf{B}_{\vec{\jmath}} - \lambda_{\vec{\jmath}}$ 'stacked' on top of each other. □

4.3. $\#\mathsf{P}_1$ is necessary for the heat equation

In order to establish Theorem 4e), we record the following fact:

Fact 28.

a) *Let $u = u(t, \vec{x})$ solve the Heat Equation (5), (6), with the initial function $\varphi \in \mathcal{C}^2([0,1]^d)$. Then the solution $\mathbf{u}(t_0, \vec{x})$ is analytic in $\vec{x}$ for each fixed $t_0 > 0$ [46, Chapter VI, 1.1, Theorem 1].*
b) *It is straightforward to check via integrating (5) by $\vec{x}$ over $[0,1]^d$ and taking (6) into account, that the 'overall heat' $\int_0^1 u(t, \vec{x})\, d\vec{x}$ does not depend on t.*

Now consider initial condition $\varphi := h \in \mathcal{C}_0^\infty([0,1]^d)$ from the (multidimensional analog of) Fact 11b): h is polynomial time computable and satisfies the boundary conditions (6), yet $\int_{[0,1]^d} h(\vec{x})\, d\vec{x}$ is not polynomial-time computable unless $\mathsf{FP}_1 = \#\mathsf{P}_1$. Now suppose that the solution $u = u(t, \vec{x})$ is computable in P. According to Fact 28a), $u(t_0, \vec{x})$ is analytic for every $t_0 > 0$. Thus $\int_{[0,1]^d} u(t_0, \vec{x})\, d\vec{x}$ is computable in P, see Fact 11e). However, this integral equals the initial $\int_{[0,1]^d} \varphi(\vec{x})\, d\vec{x}$, see Fact 28b): contradiction, unless $\mathsf{FP}_1 = \#\mathsf{P}_1$. This proves Theorem 4e). □

4.4. PSPACE is generally necessary for evolutionary linear PDEs

This subsection establishes Claim (b) of Theorem 4 as counterpart to (a). It basically says that *non*-constant coefficients may make linear PDEs PSPACE-hard, while the case of constant coefficients can be solved in #P. The *Method of Characteristics* is a well-known approach of solving linear PDEs by reduction to a family of non-linear ODEs. Here we take the opposite direction and reduce the PSPACE-'hard' non-linear ODE from Fact 11g′) to a linear PDE with non-constant coefficients of the form (24) in a way that preserves polynomial-time computability. For notational convenience we consider $\tilde{\Omega} = [-2; 2) \bmod 4$ as 1D spatial domain with periodic boundary conditions instead of $\Omega = [0; 1) \bmod 1$.

Proposition 29. *There exists a polynomial-time computable function $b : [0; 1] \times \tilde{\Omega} \to \mathbb{R}$,*

- *of class $\mathcal{C}^\infty$ in the first temporal argument and continuously differentiable ($\mathcal{C}^1$) in the second spatial argument,*
- *satisfying periodic boundary conditions $b(t, -2) \equiv b(t, +2)$ and $b_x(t, -2) \equiv b_x(t, +2)$,*

such that the classical solution $v : [0; 1] \times \tilde{\Omega} \to \mathbb{R}$

- *of the linear PDE initial value problem*

$$v_t(t,x) \;=\; b(t,x)\cdot v_x(t,x), \qquad v(0,x)=v_0(x) \tag{24}$$

- *with initial condition*

$$\begin{aligned} v_0(x) &:= 2-2x^2 \quad \textit{for} \quad |x|\le 1,\\ v_0(x) &:= -2+2(x-2)^2 = 2x^2-8x+6 \quad \textit{for} \quad +1\le x\le +2,\\ v_0(x) &:= -2+2(x+2)^2 = 2x^2+8x+6 \quad \textit{for} \quad -1\ge x\ge -2 \end{aligned} \tag{25}$$

- *subject to periodic boundary conditions* $v(t,-2)\equiv v(t,+2)$ *and* $v_x(t,-2)\equiv v_x(t,+2)$

exists but is not polynomial-time computable unless $P=PSPACE$.

Observe that v_0 does satisfy the required boundary conditions. This initial condition is also Hölder-unimodal in the sense of Lemma 30 below. Recall that the dependency of the 1D coefficient function $b=b(t,x)$ on time t can be turned into an autonomous linear system with coefficient $b=b(z,x)$ depending instead on a second 'spatial' coordinate z satisfying $\dot z\equiv 1$; so the latter form is PSPACE-'hard' as well.

Before proceeding to the proof of Proposition 29, let us record the following tool about the computational complexity of argmax for certain real functions on the unit circle.

Lemma 30 *(Polytime argmax for quantitatively unimodal functions). Recall the unit circle* $\Omega=[0;1) \bmod 1$ *and consider some* wrapped *half-open proper subinterval*

$$[u_1;u_2) \bmod 1 \;:=\; [u_1;1)\cup[0;u_2) \;\subseteq\; \Omega, \quad 0\le u_2<u_1\le 1\ .$$

Equip $[u_1;u_2) \bmod 1$ *with a total order by defining* $[0;u_2)\preccurlyeq[u_1;1)$ *pointwise (while making* $\preccurlyeq$ *recover the usual real order "*$\le$*" on* $[0;u_2)$ *and on* $[u_1;1)$ *individually). Furthermore equip* $[u_1;u_2) \bmod 1$ *with the metric*

$$d(u,u') := u'-u+1 \quad \textit{for} \quad u'\in[0;u_2) \textit{ and } u\in[u_1;1)$$

while letting $d(u,u') := |u-u'|$ *recover the usual real distance for* $u,u'\in[0;u_2)$ *and for* $u,u'\in[u_1;1)$ *individually.*

Call real-valued continuous $f:\Omega\to\mathbb{R}$ unimodal *if there exist distinct* $u_-,u_+\in\Omega$ *such that* f *is strictly increasing on the (possibly wrapped) subinterval* $[u_-;u_+) \bmod 1$ *and strictly decreasing on the subinterval* $[u_+;u_-) \bmod 1$ *(wrapped iff the previous subinterval is not wrapped).*

Quantitatively, for $\varepsilon>0$*, call continuous* f Hölder-unimodal *if there exist* $u_+,u_-\in\Omega$ *and* $\varepsilon>0$ *and* $k\in\mathbb{N}$ *such that*

$$\begin{aligned} &\forall u,u'\in[u_-;u_+) \bmod 1,\ u\preccurlyeq u': \quad f(u')-f(u) \;\ge\; \varepsilon\cdot d(u,u')^k\\ &\forall u,u'\in[u_+;u_-) \bmod 1,\ u\preccurlyeq u': \quad f(u)-f(u') \;\ge\; \varepsilon\cdot d(u,u')^k \end{aligned} \tag{26}$$

Now suppose that the function $v:[0;1]\times\Omega\to\mathbb{R}$ *is polynomial-time computable and that furthermore* $v(t,\cdot):\Omega\to\mathbb{R}$ *is Hölder-unimodal with parameters* (ε,k) *independent of* t*. Then* $u_+:[0;1]\to\Omega$ *is polynomial-time computable.*

Our ad-hoc notion of Hölder-unimodality adapts and combines the well-known concepts of quantitative *unicity* [36, §16.1] and of *unimodality* to the unit circle. The initial condition (25) in Proposition 29 is Hölder-unimodal with parameter $(2,2)$. We emphasize that some such hypothesis is necessary to avoid counterexamples like [59] and [34, §4.2]. For example unique argmax on a 2D co-domain (i.e. lacking a total order) is not feasible in polynomial time unless $\mathsf{P}_1=\mathsf{UP}_1\cap\mathsf{coUP}_1$ [34, Theorem 4.32]. Note that computational versions of the *Implicit Function Theorem require computability of the first derivative [45] and are local, thus allowing to recover the solution only locally to assert* $PSPACE_1$*-'hardness', recall Fact 11g′).*

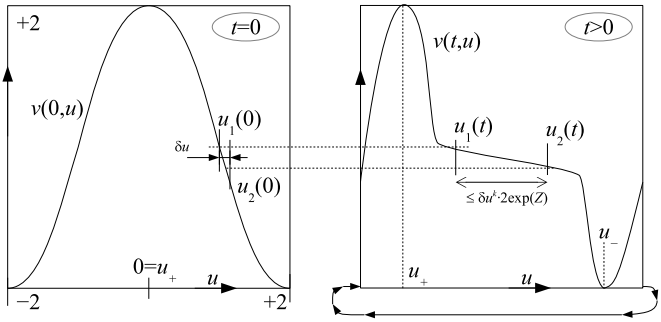

Fig. 1. Illustration to (the proof of) Proposition 29. Note how quantitative continuity of $u(0) \mapsto u(t)$ yields quantitative unimodality of $v(t,\cdot)$, as required in Lemma 30.

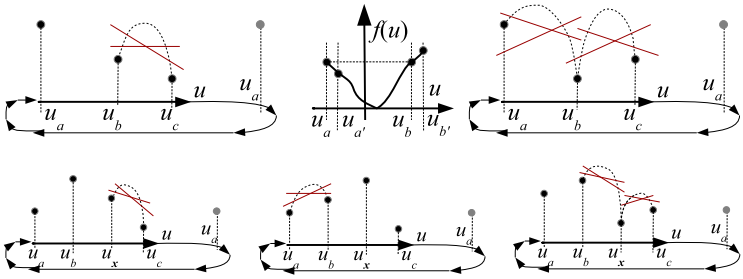

Fig. 2. Illustrating the adaptation of one trisection step in the proof of Lemma 30.

Proof of Lemma 30. We adapt the *tri*section method [19, bottom of p. 336] from simple root finding, which uses two intermediate samples (instead of one) in order to avoid possible degenerate cases involving undecidable real equality.

Fix $t \in [0;1]$ and consider the restriction $f := v(t,\cdot) : \Omega \to \mathbb{R}$, Hölder-unimodal with parameters (ε, k) by hypothesis. Note that f attains its maximum uniquely at the to-be-determined $u_+ \in \Omega$, and its minimum uniquely at the also unknown $u_- \in \Omega$. We shall approximate u_+ up to any desired absolute error $1/2^n$ using the following iteration.

Suppose u_a, u_c are distinct polynomial-time computable points on the unit circle $[0;1) \bmod 1 = \mathrm{dom}(f)$ such that the proper and possibly wrapping subinterval $[u_a; u_c) \bmod 1$ is guaranteed to contain u_+ but not u_-. Consider equidistant points $u_b, u_x \in [u_a; u_c) \bmod 1$, again polynomial-time computable, and distinguish the possible order types of $\big(f(u_a), f(u_b), f(u_x), f(u_c), f(u_a)\big)$.

In the case $f(u_a) > f(u_b) > f(u_c)$ depicted in the top left of Fig. 2, the assumption $u_+ \in [u_b; u_c) \bmod 1$ requires either a (local) minimum or another local maximum to be attained in $[u_a; u_b) \bmod 1$: either way a contradiction. Similarly for the case $f(u_a) < f(u_b) < f(u_c)$: by symmetry it suffices here and in the sequel to suppose $f(u_a) > f(u_c)$. In the case $f(u_a) > f(u_c) > f(u_b)$ depicted in the top right of Fig. 2, both assumptions $u_+ \in [u_b; u_b) \bmod 1$ and $u_+ \in [u_b; u_c) \bmod 1$ lead to a contradiction. The case $f(u_b) > f(u_a) > f(u_c)$ is split further into three subcases depicted in the bottom of Fig. 2 depending on the relative location of $f(u_x)$.

Each case analysis yields a subinterval either of the form $[u_b; u_{c'}) \bmod 1$ or of the form $[u_{b'}; u_c) \bmod 1$ which by contraposition is again guaranteed to contain u_+ (and a fortiori still not contain u_-). Moreover the new subinterval has at most $\frac{2}{3}$ times the length of the previous one: arriving at an approximation to u_+ up to absolute error $1/2^n$ after $\mathcal{O}(n)$ iterations.

Refining this analysis in terms of bit (rather than unit) cost, polynomial-time computability of f and of u_a, u_b asserts that any comparison "$f(u_a) < f(u_b)$?" can be performed in time polynomial in $-\log|f(u_a) - f(u_b)|$. Here is where Hölder-unimodality (26) enters with parameters (ε, k), although that does not directly guarantee $|f(u_a) - f(u_b)| \geq \mathcal{O}(\varepsilon) \cdot d(u_a, u_b)^k$; see the top middle of Fig. 2 where

$f(u_a) = f(u_b)$. Instead consider, in addition to u_a and u_b, perturbed polynomial-time computable points $u_{a'}$ and $u_{b'}$; and perform four comparisons in parallel:

$$f(u_a) < f(u_b)? \quad f(u_{a'}) < f(u_b)? \quad f(u_a) < f(u_{b'})? \quad f(u_{a'}) < f(u_{b'})?$$

Now if the perturbation is chosen such as to satisfy $d(u_a, u_{a'}), d(u_b, u_{b'}) \geq \delta \cdot d(u_a, u_b)^k$ for some constant δ, then Hölder-unimodality (26) does guarantee at least one of the comparisons to return in time polynomial in $-\log d(u_a, u_b)$; similarly for u_c, u_x: resulting in an overall bit-cost polynomial in n. □

Proof of Proposition 29. Let us first recall the Method of Characteristics at the example of Equation (24). To this end consider some solution $u = u(t)$ of the nonlinear ODE $\dot{u} = -b(t, u)$. Then, for a differentiable function $v = v(t, x)$, the Chain Rule yields

$$\begin{aligned} \tfrac{d}{dt} v\big(t, u(t)\big) &= v_t\big(t, u(t)\big) \ + \ v_x\big(t, u(t)\big) \cdot \dot{u}(t) \\ &= v_t\big(t, u(t)\big) \ - \ v_x\big(t, u(t)\big) \cdot b\big(t, u(t)\big) \ = \ 0 \ . \end{aligned} \tag{27}$$

Hence $v\big(t, u(t)\big) \equiv v\big(0, u(0)\big)$: v is constant along the trajectory $u = u(t)$. Solving the nonlinear ODE $\dot{u} = -b(t, u)$ for all possible initial values $u(0)$ thus implicitly yields the solution to the linear PDE (24) as collection of level curves: since trajectories do not cross and continuously depend on the initial value.

Here we conduct a reduction in the opposite direction: Intuitively, take the right-hand side f of PSPACE-hard nonlinear ODE to initial value 0 from Fact 11g′). We write $u_0 : [0; 1] \to [-1; 1]$ for the solution to said special initial value: $u_0(0) = 0$. Then consider all possible initial value problems $\dot{u} = -b(t, u)$ for $b := -f$, that is, let $u(0)$ vary. Next define $v\big(t, u(t)\big) :\equiv v\big(0, u(0)\big)$ and read Equation (27) backwards to verify that said v indeed solves the linear PDE (24) with $v\big(0, u_0(0)\big) = 2$. Finally employ Lemma 30 to see that polynomial-time computability of said v implies polynomial-time computability of the special solution u_0 to Hölder-unimodal initial value $u_0(0) = 0$ as peak contour line $v\big(t, u_0(t)\big) \equiv 2$: contradicting Fact 11g′).

However in order to make this intuitive reduction rigorous, these aspects require further attention:

i) The coefficient function b in PDE (24) must satisfy the periodic boundary condition.
ii) The solution to ODE $\dot{u} = -b(t, u)$ must exist on the entire time interval $[0; 1]$ for every possible initial value $u(0)$, not only for the special case $u(0) = 0$ covered by Fact 11g′.
iii) The (only thus well-defined) solution v to PDE (24) must satisfy the hypothesis of Lemma 30 — technically with $\tilde{\Omega}$ instead of Ω.

Regarding (i), extend $-f(t, \cdot)$ from $[-1; 1]$ to $b(t, \cdot) : [-2; 2]$ with $b(t, x) \equiv 0$ for $|x| \geq 1$. Note that this extension preserves polynomial-time computability, and continuous differentiability $\mathcal{C}^1$ in x since $f(t, \pm 1) \equiv 0 \equiv f_x(t, \pm 1)$ by virtue of Fact 11g′). Moreover the PSPACE-hard special solution u_0 to $\dot{u} = f(t, u)$ remains solution to $\dot{u} = -b(t, u)$ at initial value 0. This also takes care of Property (ii): although a solution $\dot{u} = f(t, u)$ to initial value $u(0) \in [-1; 1] \setminus \{0\}$ could reach the boundary of $\mathrm{dom}(f) = [-1; 1]$, the extension being identically zero outside of $[-1; 1]$ prevents said solution from leaving $[-1; 1]$. To finally see that the hypothesis of Lemma 30 holds, continuous dependence of $u(t)$ on t and the initial value $u(0)$ asserts continuity of v. Moreover recall that trajectories do not cross: therefore $v(t, \cdot)$ preserves the unimodality of the initial $v(0, \cdot) : \tilde{\Omega} \to \mathbb{R}$, qualitatively; see Fig. 1. Quantitatively, $v(0, \cdot)$ is Hölder-unimodal with parameters $(\varepsilon, k) = (2, 2)$. This carries over to $v(t, \cdot)$ also being Hölder-unimodal, since f is Lipschitz-continuous by Fact 11g): $|b(t, x_1) - b(t, x_2)| \leq Z \cdot |x_1 - x_2|$ for some $Z \in \mathbb{N}$ independent of $t \in [0; 1]$. Hence solutions $u_1 = u_1(t)$ and $u_2 = u_2(t)$ to different initial values satisfy

$$|\dot{u}_1(t) - \dot{u}_2(t)| \ = \ \big|b\big(t, u_1(t)\big) - b\big(t, u_2(t)\big)\big| \ \leq \ Z \cdot |u_1(t) - u_2(t)|$$

and thus

$$|u_1(t) - u_2(t)| \le \exp(Zt) \cdot |u_1(0) - u_2(0)| \le \exp(Z) \cdot |u_1(0) - u_2(0)| \ .$$

To see the latter more formally, express $|x| = \max\{+x, -x\}$ and $\dot{u}(t) = \lim_{\delta \searrow 0} \big(u(t+\delta) - u(t)\big)/\delta$ and consider all combinations of cases $\dot{u} > 0$, $\dot{u} < 0$, $u > 0$, and $u < 0$ at t, or rather on some sufficiently small neighborhood of t. Finally, by definition,

$$\begin{aligned} |v\big(t, u_2(t)\big) - v\big(t, u_1(t)\big)| &= |v\big(0, u_2(0)\big) - v\big(0, u_1(0)\big)| \\ &\ge \varepsilon \cdot |u_1(0) - u_2(0)|^k \ge \varepsilon / \exp(Z) \cdot |u_1(t) - u_2(t)|^k. \quad \square \end{aligned}$$

5. Examples

This section provides some illustrative examples for the main Theorem 4 and the techniques developed in Section 3.

5.1. Some PDE systems to which our theorems apply

Here we summarize several well known equations and systems, illustrating Theorem 4 a), c). We emphasize that Hypotheses 3 (iii) about convergence of the difference scheme w.r.t. the maximum norm might lead to strong smoothness assumptions, which are different for different examples.

Example 31. Many evolutionary linear PDEs admit difference schemes that satisfy the hypotheses of Theorem 4 c), and thus can be computed in #P, including:

a) The Heat Equation (5) with periodic boundary conditions and a polynomial time computable initial function, provided that $u(t, \vec{x}) \in C^{(1,4)}([0, T] \times \bar{\Omega})$: see [40, §2.11] or [20, p. 1168] for maximum norm difference scheme convergence, and e.g. [12] for well-posedness.

b) The Wave Equation (4) with periodic boundary conditions and polynomial time computable initial functions. Indeed, the wave equation admits a maximum norm convergent difference scheme under additional smoothness assumptions, see [41, Theorem 3.1] for the two-dimensional case, given that $u(t, x, y) \in C^{(4,5)}([0, T] \times \bar{\Omega})$. The continuous dependence condition (8) can be verified combining the well-known continuous dependence w.r.t. L_2-norms (which generalizes to W_2^l after applying the L_2 estimate to the differentiated equation), and Sobolev Embedding Theorem, see e.g. [12, §5.6.3]: for $\varphi \in C^{p+1}(\Omega)$ and $p > d/2$,

$$||u||_{C(\bar{\Omega})} \le ||u||_{W_2^p(\Omega)} \le C_0 ||\varphi||_{W_2^p(\Omega)} \le C_0 ||\varphi||_{C^p(\Omega)},$$

where C_0 does not depend on φ; thus $||u||_{C^2(\bar{\Omega})} \le ||\varphi||_{C^{p+2}(\Omega)}$.

c) Note that the two-dimensional acoustics system

$$\begin{cases} \rho_0 \dfrac{\partial u}{\partial t} + \dfrac{\partial p}{\partial x} = 0, \\ \rho_0 \dfrac{\partial v}{\partial t} + \dfrac{\partial p}{\partial y} = 0, \\ \dfrac{\partial p}{\partial t} + \rho_0 c_0^2 \left(\dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} \right) = 0 \end{cases}$$

can be equivalently reduced to the two-dimensional wave equation (e.g. [13]); see also [14] for other examples of symmetric hyperbolic systems (3) (with constant coefficients $\mathbf{B}_j$), which are equivalent to higher-order wave equations.

Note that if periodic boundary conditions are replaced by others such that Hypotheses 3 holds, the solution u in the examples a), b), c) above can be computed in PSPACE, according to Theorem 4 a).

5.2. Circulant matrices of difference schemes and corresponding polynomials

To illustrate the methods of Section 3, we write out the difference scheme (9) matrices of the form (20) and the corresponding polynomials (18) for the heat equation (5), (6) for $d = 1, 2, 3$.

For $d = 1$ the difference scheme/recurrence relation looks as: $u^k = u_k + \tau a^2 \left(\frac{u_{k+1} - 2u_k + u_{k-1}}{h^2}\right)$, where u_k denotes the grid function value at a certain time step $k\tau$, while u^{k+1} is its value at the next time step $(k+1)\tau$. Thus, denoting $\lambda = \frac{\tau a^2}{h^2}$, $\mu = 1 - 2\lambda$, we get

$$\mathbf{A}_h^{(1)} = \begin{bmatrix} \mu & \lambda & & \lambda \\ \lambda & \mu & \lambda & \\ & & \cdots & \\ \lambda & & \lambda & \mu \end{bmatrix} = \mu\mathbb{I} + \lambda\mathcal{C}^1 + \lambda\mathcal{C}^{-1},$$

where $\mathbb{I}$ is the identity matrix of a corresponding dimension;

$$\mathcal{C}^1 = \begin{bmatrix} 0 & 1 & & \\ & \cdots & & \\ & & & 1 \\ 1 & & & 0 \end{bmatrix}; \quad \mathcal{C}^{-1} = \begin{bmatrix} 0 & & & 1 \\ 1 & 0 & & \\ & \cdots & & \\ & & 1 & 0 \end{bmatrix}.$$

The associated Laurent polynomial is $P_h^{(1)}(X) = \mu + \lambda X + \lambda X^{-1}$.

For $d = 2$, $\mathbf{A}_h^{(2)} =$

$$= \left[\begin{array}{cccc|cccc|ccc|ccc} \mu & \lambda & & \lambda & \lambda & & & & & & & \lambda & & \\ \lambda & \mu & \lambda & & & \lambda & & & & & & & \lambda & \\ & & \cdots & & & & \cdots & & & \cdots & & & & \cdots \\ \lambda & & \lambda & \mu & & & & \lambda & & & & & & \lambda \\ \hline \lambda & & & & \mu & \lambda & & \lambda & \lambda & & & & & \\ & \lambda & & & \lambda & \mu & \lambda & & & \lambda & & & & \\ & & \cdots & & & & \cdots & & & & \cdots & & & \\ & & & \lambda & \lambda & & \lambda & \mu & & & \lambda & & & \\ \hline & & & & & & & & & & & & & \\ & & & & & & \cdots & & & \cdots & & & \cdots & \\ & & & & & & & & & & & & & \\ \hline \lambda & & & & & & & & \lambda & & & \mu & \lambda & & \lambda \\ & \lambda & & & & & & & & \lambda & & \lambda & \mu & \lambda & \\ & & \cdots & & & & \cdots & & & & \cdots & & & \cdots & \\ & & & \lambda & & & & & & & \lambda & \lambda & & \lambda & \mu \end{array}\right] =$$

$$\mu\mathbb{I} + \lambda \begin{bmatrix} \mathcal{C}^1 & & \\ & \ddots & \\ & & \mathcal{C}^1 \end{bmatrix} + \lambda \begin{bmatrix} \mathcal{C}^{-1} & & \\ & \ddots & \\ & & \mathcal{C}^{-1} \end{bmatrix} + \lambda \begin{bmatrix} 0 & \mathbb{I} & & \\ & \ddots & \ddots & \\ & & 0 & \mathbb{I} \\ \mathbb{I} & & & 0 \end{bmatrix} + \lambda \begin{bmatrix} 0 & & & \mathbb{I} \\ \mathbb{I} & 0 & & \\ & \ddots & \ddots & \\ & & \mathbb{I} & 0 \end{bmatrix}$$

$$= \mu \cdot \mathbb{I} \otimes \mathbb{I} + \lambda \cdot \mathcal{C}^1 \otimes \mathbb{I} + \lambda \cdot \mathcal{C}^{-1} \otimes \mathbb{I} + \lambda \cdot \mathbb{I} \otimes \mathcal{C}^1 + \lambda \cdot \mathbb{I} \otimes \mathcal{C}^{-1}.$$

Its associated Laurent polynomial is

$$P_h^{(2)}(X, Y) = \mu + \lambda X + \lambda X^{-1} + \lambda Y + \lambda Y^{-1}.$$

Finally, for $d = 3$,

$$\mathbf{A}_h^{(3)} = \mu \cdot \mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I} + \lambda \cdot \mathcal{C} \otimes \mathbb{I} \otimes \mathbb{I} + \lambda \cdot \mathcal{C}^{-1} \otimes \mathbb{I} \otimes \mathbb{I} + \lambda \cdot \mathbb{I} \otimes \mathcal{C} \otimes \mathbb{I} + \\ + \lambda \cdot \mathbb{I} \otimes \mathcal{C}^{-1} \otimes \mathbb{I} + \lambda \cdot \mathbb{I} \otimes \mathbb{I} \otimes \mathcal{C} + \lambda \cdot \mathbb{I} \otimes \mathbb{I} \otimes \mathcal{C}^{-1},$$

$$P_h^{(3)}(X, Y, Z) = \mu + \lambda X + \lambda X^{-1} + \lambda Y + \lambda Y^{-1} + \lambda Z + \lambda Z^{-1}.$$

6. Conclusion and perspective

Real Complexity Theory determines the computational cost intrinsic to numerical problems. Previous work has for example established that

- In/definite continuous maximization corresponds to NP and NP_1, respectively [34, Theorems 3.7+3.16].
- In/definite Riemann integration corresponds to #P and $\#\mathsf{P}_1$, respectively [34, Theorems 5.32+5.33].
- Solving Lipschitz-continuous ODEs corresponds to PSPACE [22].
- 1D real function inversion corresponds to P [33].
- Poisson's elliptic linear PDE can be solved in #P, and is hard for $\#\mathsf{P}_1$ [30] (not #P, as claimed in [30, §6]).
- Volterra's Integral Equation can be solved in EXPSPACE, and EXPTIME is necessary [35].

The present work has extended these considerations to evolutionary (including hyperbolic and parabolic linear systems of PDEs: The difference scheme approach for finding classical solutions to linear systems of PDEs is, in a certain sense, PSPACE-"complete" (w.r.t. the output precision parameter n, with the result computed up to guaranteed precision 2^{-n}), since so is matrix powering for general matrices of dimension exponential in n. For broad classes of systems and scalar PDEs with constant coefficients and periodic boundary conditions we prove that these complexity bounds can be further improved to #P, but not to P: since inner product is, in a certain sense, #P-"complete".

6.1. Spectrum problem in real complexity

Determining the cardinality of some model (such as the Stone-Čech compactification of the integers) is a classical topic in Mathematics. Recall that, by Schröder-Bernstein, cardinality means maximal with respect to surjective images/injective pre-images. In 1952, Heinrich Scholz asked conversely which cardinalities arise as models of first-order sentences. This has become known as the famous *Spectrum Problem* [11].

Similarly, the classical goal of Complexity Theory is to classify a decision problem under consideration, that is, to show it complete (=maximal with respect to a suitable reduction) for some complexity class. The textbook of Garey&Johnson can be regarded as conversely identifying many natural problems complete for NP. Various combinatorial (two-player) games have been demonstrated PSPACE-complete [9]. An analogous direction of research is concerned with identifying natural problems complete for the discrete complexity class $\mathsf{NP}_{\mathbb{R}}$, also known as *Existential Theory of the Reals* [54,17,18,10]. By *Schaefer's Theorem* [55], constraint satisfaction problems are either in P or NP-complete.

As expanded on at the beginning of Section 6, Real Complexity characterizes discrete complexity classes in terms of numerical problems. The (families of linear) PDEs considered so far seem to satisfy a *tri*chotomy similar to Schaefer's: P in the analytic case, PSPACE in general, and #P for many popular PDEs including Poisson and Heat. This suggests the following question:

Question 32 *(Spectrum problem in real complexity).*

a) *Do other non/linear PDEs characterize discrete complexity classes beyond P/#P/PSPACE/EXPTIME?*
b) *Which discrete complexity classes arise from other 'natural' numerical problems?*

Acknowledgements

We dedicate this work to the memory of Ker-I Ko [34], and of Arthur W. Chou, who popularized computational complexity investigations in Analysis.

This research was supported by the National Research Foundation of Korea (grant 2017R1E1A1A03071032) and by the International Research & Development Program of the Korean Ministry of Science and ICT (grant 2016K1A3A7A03950702) and by the NRF Brain Pool program (grant 2019H1D3A2A02102240). GP was supported by NSF grants CCF-1564132, CCF-1563942, DMS-1853482, DMS-1853650, and DMS-1760448, by PSC-CUNY grants #69827-0047 and #60098-0048.

We thank Lina Bondar for a helpful discussion on smoothness assumptions and different versions of the Sobolev Embedding Theorem (Example 31 a, b)). Our gratitude extends to the anonymous reviewers for providing helpful feedback to a previous version of this work.

References

[1] A. Bacho, H. Boche, G. Kutyniok, Complexity blowup for solutions of the Laplace and the diffusion equation, https://arxiv.org/abs/2212.00693, 2022.
[2] J. Balcázar, J. Díaz, J. Gabarró, Structural Complexity I, Springer, 1994.
[3] H. Boche, V. Pohl, Complexity blowup in simulating analog linear time-invariant systems on digital computers, IEEE Trans. Signal Process. 69 (2021) 5005–5020, https://doi.org/10.1109/TSP.2021.3102826.
[4] A. Borodin, I. Munro, The Computational Complexity of Algebraic and Numeric Problems, Elsevier, 1975.
[5] O. Bournez, D.S. Graça, A. Pouly, Solving analytic differential equations in polynomial time over unbounded domains, in: F. Murlak, P. Sankowski (Eds.), Mathematical Foundations of Computer Science 2011, in: Lecture Notes in Computer Science, vol. 6907, Springer, Heidelberg, 2011, pp. 170–181.
[6] V. Brattka, G. Gherardi, A. Pauly, Weihrauch complexity in computable analysis, in: V. Brattka, P. Hertling (Eds.), Handbook of Computability and Complexity in Analysis, in: Theory and Applications of Computability, Springer, Cham, 2021, pp. 367–417.
[7] M. Braverman, S.A. Cook, Computing over the reals: foundations for scientific computing, Not. Am. Math. Soc. 53 (3) (2006) 318–329.
[8] H. Chernoff, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, Ann. Math. Stat. 23 (4) (1952) 493–507, https://doi.org/10.1214/aoms/1177729330.
[9] E.D. Demaine, R.A. Hearn, Playing games with algorithms: algorithmic combinatorial game theory, in: M.H. Albert, R.J. Nowakowski (Eds.), Games of No Chance 3, in: Mathematical Sciences Research Institute Publications, vol. 56, Cambridge University Press, 2009, pp. 3–56.
[10] M.G. Dobbins, A.F. Holmsen, A. Hubard, Realization spaces of arrangements of convex bodies, Discrete Comput. Geom. 58 (1) (2017) 1–29, https://doi.org/10.1007/s00454-017-9869-8.
[11] A. Durand, N. Jones, J. Makowsky, M. More, Fifty years of the spectrum problem: survey and new results, Bull. Symb. Log. 18 (07 2009), https://doi.org/10.2178/bsl.1804020.
[12] L. Evans, Partial Differential Equations, Graduate Studies in Mathematics, vol. 19, American Mathematical Society, 1998.
[13] S. Godunov, V. Ryaben'kii, Difference Schemes: An Introduction to the Underlying Theory, Studies in Mathematics and Its Applications, vol. 19, North-Holland, 1987.
[14] V. Gordienko, Hyperbolic systems equivalent to the wave equation, Sib. Math. J. 50 (2009) 14–21.
[15] J. Hartmanis, N. Immerman, V. Sewelson, Sparse sets in np-p: exptime versus nexptime, Inf. Control 65 (2) (1985) 158–181, https://doi.org/10.1016/S0019-9958(85)80004-8.
[16] L.A. Hemachandra, M. Ogiwara, Is #P Closed Under Subtraction?, Current Trends in Theoretical Computer Science, 1993, pp. 523–536.
[17] C. Herrmann, J. Sokoli, M. Ziegler, Satisfiability of cross product terms is complete for real nondeterministic polytime Blum-Shub-Smale machines, in: T. Neary, M. Cook (Eds.), Proceedings Machines, Computations and Universality 2013, MCU 2013, Zürich, Switzerland, September 9–11, 2013, in: EPTCS, vol. 128, 2013, pp. 85–92.
[18] C. Herrmann, M. Ziegler, Computational complexity of quantum satisfiability, J. ACM 63 (2) (2016) 19:1–19:31, https://doi.org/10.1145/2869073.
[19] P. Hertling, Topological complexity with continuous operations, J. Complex. 12 (1996) 315–338, https://doi.org/10.1006/jcom.1996.0021.
[20] D. Hoff, Stability and convergence of finite difference methods for systems of nonlinear reaction-diffusion equations, SIAM J. Numer. Anal. 15 (6) (1978) 1166–1177, https://doi.org/10.1137/0715077.
[21] T. Kailath, S.Y. Kung, M. Morf, Displacement ranks of a matrix, Bull. Am. Math. Soc. 1 (5) (1979) 769–773.
[22] A. Kawamura, Lipschitz continuous ordinary differential equations are polynomial-space complete, Comput. Complex. 19 (2) (2010) 305–332, https://doi.org/10.1007/s00037-010-0286-0.
[23] A. Kawamura, S.A. Cook, Complexity theory for operators in analysis, ACM Trans. Comput. Theory 4 (2) (2012) 5:1–5:24, https://doi.org/10.1145/2189778.2189780.
[24] A. Kawamura, N. Müller, C. Rösnick, M. Ziegler, Computational benefit of smoothness: parameterized bit-complexity of numerical operators on analytic functions and Gevrey's hierarchy, J. Complex. 31 (5) (2015) 689–714, https://doi.org/10.1016/j.jco.2015.05.001.

[25] A. Kawamura, H. Ota, Small complexity classes for computable analysis, in: Mathematical Foundations of Computer Science 2014. Part II, in: Lecture Notes in Comput. Sci., vol. 8635, Springer, Heidelberg, 2014, pp. 432–444.
[26] A. Kawamura, H. Ota, C. Rösnick, M. Ziegler, Computational complexity of smooth differential equations, Log. Methods Comput. Sci. 10 (1) (2014) 6, https://doi.org/10.2168/LMCS-10(1:6)2014.
[27] A. Kawamura, F. Steinberg, H. Thies, Parameterized complexity for uniform operators on multidimensional analytic functions and ODE solving, in: Proc 25th Int. Workshop on Logic, Language, Information, and Computation, WOLLIC, 2018, pp. 223–236.
[28] A. Kawamura, F. Steinberg, H. Thies, Parameterized complexity for uniform operators on multidimensional analytic functions and ODE solving, in: L. Moss, R. de Queiroz, M. Martinez (Eds.), Logic, Language, Information, and Computation, in: Lecture Notes in Computer Science, vol. 10944, Springer, Berlin, 2018, pp. 223–236.
[29] A. Kawamura, F. Steinberg, M. Ziegler, Complexity theory of (functions on) compact metric spaces, in: Proc. 31st Ann. Symposium on Logic in Computer Science, LICS, ACM, 2016, pp. 837–846.
[30] A. Kawamura, F. Steinberg, M. Ziegler, On the computational complexity of the Dirichlet problem for Poisson's equation, Math. Struct. Comput. Sci. 27 (8) (2017) 1437–1465, https://doi.org/10.1017/S096012951600013X.
[31] A. Kawamura, H. Thies, M. Ziegler, Average-case polynomial-time computability of Hamiltonian dynamics, in: I. Potapov, P.G. Spirakis, J. Worrell (Eds.), 43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, Liverpool, UK, August 27–31, 2018, in: LIPIcs, vol. 117, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018, pp. 30:1–30:17.
[32] K. Ko, Compuational Complexity of Real Functions, Birkhäuser, Boston, 1991.
[33] K.I. Ko, Inverting a one-to-one real function is inherently sequential, in: S. Buss, P. Scott (Eds.), Feasible Mathematics, Proc. Workshop, June 16–18, 1989, vol. 9, Cornell Univ., Ithaca, NY, 1990, pp. 239–257.
[34] K.I. Ko, Complexity Theory of Real Functions, Progress in Theoretical Computer Science, Birkhäuser, Boston, 1991.
[35] K.I. Ko, On the computational complexity of integral equations, Ann. Pure Appl. Log. 58 (1992) 201–228.
[36] U. Kohlenbach, Applied Proof Theory: Proof Interpretations and Their Use in Mathematics, Springer, Berlin, 2008.
[37] A.N. Kolmogorov, V.M. Tikhomirov, E-entropy and E-capacity of sets in functional spaces, in: A. Shiryayev (Ed.), Selected Works of A.N. Kolmogorov, vol. III, Springer, 1993, pp. 86–170; Originally in Usp. Mat. Nauk 14 (2) (1959) 3–86.
[38] I. Koswara, S. Selivanova, M. Ziegler, Computational complexity of real powering and improved solving linear differential equations, in: Computer Science—Theory and Applications, in: Lecture Notes in Computer Science, vol. 11532, Springer, Cham, 2019, pp. 215–227.
[39] I. Kra, S.R. Simanca, On circulant matrices, Not. Am. Math. Soc. 59 (3) (2012) 368–377, https://doi.org/10.1090/noti804.
[40] R.J. LeVeque (Ed.), Finite Difference Methods for Differential Equations, SIAM, Philadelphia, 2007.
[41] H.l. Liao, Z.z. Sun, Maximum norm error estimates of efficient difference schemes for second-order wave equations, J. Comput. Appl. Math. 235 (8) (2011) 2217–2233, https://doi.org/10.1016/j.cam.2010.10.019.
[42] H. Luckhardt, A fundamental effect in computations on real numbers, Theor. Comput. Sci. 5 (3) (1977) 321–324.
[43] R.A.E. Manssour, M. Härkönen, B. Sturmfels, Linear PDE with constant coefficients, Glasg. Math. J. (2021) 1–26, https://doi.org/10.1017/S0017089521000355.
[44] S.F. McCormick, S.M. McKay, J. Thomas, Computational complexity of the fast adaptive composite grid (FAC) method, Appl. Numer. Math. 6 (4) (1990) 315–327, https://doi.org/10.1016/0168-9274(90)90023-9.
[45] T.H. McNicholl, A uniformly computable implicit function theorem, Math. Log. Q. 54 (3) (2008) 272–279.
[46] V. Mikhailov, Partial Differential Equations, Central Books Ltd., 1979.
[47] M.L. Minsky, Recursive unsolvability of post's problem of "tag" and other topics in theory of Turing machines, Ann. Math. 74 (3) (1961) 437–455.
[48] C.A. Neff, Specified precision polynomial root isolation is in NC, J. Comput. Syst. Sci. 48 (3) (1994) 429–463, https://doi.org/10.1016/S0022-0000(05)80061-3.
[49] V. Pan, J. Reif, The bit-complexity of discrete solutions of partial differential equations: compact multigrid, Comput. Math. Appl. 20 (1990) 9–16.
[50] S. Park, Reliable degenerate matrix diagonalization, Tech. Rep. CS-TR-2018-415, KAIST, 2018.
[51] S. Park, F. Braußе, P. Collins, S. Kim, M. Konečný, G. Lee, N. Müller, E. Neumann, N. Preining, M. Ziegler, Foundation of computer (algebra) analysis systems: semantics, logic, programming, verification, https://arxiv.org/abs/1608.05787, 2020.
[52] A. Pouly, D. Graça, Computational complexity of solving polynomial differential equations over unbounded domains, Theor. Comput. Sci. 626 (2016) 627–682, https://doi.org/10.1016/j.tcs.2016.02.002.
[53] M.B. Pour-El, J.I. Richards, The wave equation with computable initial data such that its unique solution is not computable, Adv. Math. 39 (1981) 215–239.
[54] M. Schaefer, Complexity of some geometric and topological problems, in: D. Eppstein, E.R. Gansner (Eds.), Graph Drawing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 334–344.
[55] T.J. Schaefer, The complexity of satisfiability problems, in: R.J. Lipton, W.A. Burkhard, W.J. Savitch, E.P. Friedman, A.V. Aho (Eds.), Proceedings of the 10th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, May 1–3, 1978, ACM, 1978, pp. 216–226.
[56] S. Selivanova, F. Steinberg, H. Thies, M. Ziegler, Exact real computation of solution operators for linear analytic systems of partial differential equations, in: Proc. Computer Algebra in Scientific Computing, CASC 2021, in: LNCS, 2021, pp. 370–390.
[57] S. Selivanova, V. Selivanov, Bit complexity of computing solutions for symmetric hyperbolic systems of PDEs with guaranteed precision, Computability 10 (2) (2021) 123–140, https://doi.org/10.3233/COM-180215.
[58] S. Selivanova, V.L. Selivanov, Computing solution operators of boundary-value problems for some linear hyperbolic systems of PDEs, Log. Methods Comput. Sci. 13 (4) (2017), https://doi.org/10.23638/LMCS-13(4:13)2017.
[59] E. Specker, Der Satz vom Maximum in der rekursiven Analysis, in: A. Heyting (Ed.), Constructivity in Mathematics, Proc. Colloq., Amsterdam, Aug. 26–31, 1957, in: Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam, 1959, pp. 254–265.

[60] J.C. Strikwerda, Finite Difference Schemes and Partial Differential Equations, SIAM, 2004.
[61] S. Sun, N. Zhong, M. Ziegler, Computability of the solutions to Navier-Stokes equations via effective approximation, in: D. Du, J. Wang (Eds.), Complexity and Approximation – In Memory of Ker-I Ko, in: Lecture Notes in Computer Science, vol. 12000, Springer, 2020, pp. 80–112.
[62] K. Weihrauch, Computable Analysis, Springer, Berlin, 2000.
[63] K. Weihrauch, Computational complexity on computable metric spaces, Math. Log. Q. 49 (1) (2003) 3–21.
[64] K. Weihrauch, N. Zhong, Is wave propagation computable or can wave computers beat the Turing machine?, Proc. Lond. Math. Soc. 85 (2) (2002) 312–332.
[65] J.W.J. Wrench, Concerning two series for the gamma function, Math. Comput. 22 (1968) 617–626, https://doi.org/10.1090/S0025-5718-1968-0237078-4.
[66] M. Ziegler, V. Brattka, A computable spectral theorem, in: J. Blanck, V. Brattka, P. Hertling (Eds.), Computability and Complexity in Analysis, 4th International Workshop, CCA 2000, Swansea, UK, September 2000, in: Lecture Notes in Computer Science, vol. 2064, Springer, Berlin, 2001, pp. 378–388.
[67] M. Ziegler, V. Brattka, Computability in linear algebra, Theor. Comput. Sci. 326 (1–3) (2004) 187–211.

Sponsor names

Do not correct this page. Please mark corrections to sponsor names and grant numbers in the main text.

National Research Foundation of Korea, *country*=South Korea, *grants*=2017R1E1A1A03071032

Ministry of Science and ICT, *country*=South Korea, *grants*=2016K1A3A7A03950702

NSF, *country*=United States, *grants*=CCF-1564132, CCF-1563942, DMS-1853482, DMS-1853650, DMS-1760448