Differential Equations: Complexity and Exact Real Computation

Svetlana Selivanova

KAIST, School of Computing, 16 May 2019

1. General problem setting and motivation
2. Recent results (on linear ODEs and PDEs)
3. Future plans and open questions

General problem statement, differential problem

We consider **boundary-value problems** for systems of PDEs

$$\begin{cases} Lu(y) = f(y), y \in \Omega \subset \mathbb{R}^k \\ \mathcal{L}u(y)|_\Gamma = \varphi(y\,|_\Gamma), \Gamma \subseteq \partial\Omega. \end{cases} \tag{1}$$

L and $\mathcal{L}$ are differential operators; the differential order of $\mathcal{L}$ is less than the one of L; $\mathcal{L}u = \varphi$ are the initial and boundary conditions
E.g. for the Cauchy problem for the $3D$ wave equation

$$L = \frac{\partial^2}{\partial t^2} - c_0^2 \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right)$$

$$\mathcal{L}u = \begin{pmatrix} u \\ \frac{\partial u}{\partial t} \end{pmatrix}, \Gamma = \{t = 0\}$$

General problem statement, differential problem

Usually the higher-order differential equations

$$Lu = \sum_{|\alpha|=k} a_\alpha(D^{k-1}\mathbf{u}, \ldots, \mathbf{u}, y) D^\alpha \mathbf{u} + a_0(D^{k-1}\mathbf{u}, \ldots, \mathbf{u}, y) = f(y)$$

can be reduced by change of variables to 1st order **systems** of equations (like the wave equation reduces to the acoustic equations)

$$\mathbf{u}_t = \sum_{i=1}^{m} B_i(x, \mathbf{u}) \frac{\partial}{\partial x_i}$$

Here B_i are matrices.
Many types of L (hyperbolic, parabolic, elliptic, subelliptic, subparabolic, ... ; linear, quasilinear, nonlinear) and b-v problems (conservative, dissipative, ... initial function: analytic, C^k-smooth, continuous, ...) with different properties and many open theoretical questions

General problem statement, approaches

Approaches to solve (1):

- Explicit solution formulas.
  - Group analysis (physical symmetries)
  - Fourier transformation method
  - ...

  Exist rarely and usually involve complicated expressions, not obvious how to evaluate
- Iterative methods of proofs of existence. *Usually not optimal*
- Numerical methods (difference schemes, finite element methods, Monte-Carlo, ...).
  *Usually heuristic, no rigorous foundations: exact convergence rates, complexity bounds***?**
  Not always reliable because of approximation errors!

Goals:

(1) describe the candidate algorithms in a uniform computational framework and work out reliable and efficient algorithms, i.e. computing the solution with a guaranteed prescribed precision or reporting there are not enough resources to do so;

(2) determine what minimal resources (time, storage space,?) a given algorithm would require; and more generally minimal amount of resources needed to solve a differential equation problem, i.e. computational complexity of this problem; (bridging classical discrete computability and complexity theories with continuous theory and numerics of equations in mathematical physics and engineering)

(3) implement optimal and reliable algorithms using exact real computation software (trial versions for linear systems of PDEs implemented; plan to extend to more general setting and create new PDE solver libraries)

Avoid numerical instabilities caused by double precision arithmetic (e.g. for discontinuous problems or problems with big modulus of continuity)

General problem statement, computability questions

We study: computability properties of the solution operator

$$R : (L, \mathcal{L}, f, \varphi) \mapsto \mathbf{u}$$

Questions:

- Is the solution operator **computable** from f, φ?
- From the coefficients of $L, \mathcal{L}$? In which classes of functions?
- What is the **complexity** of computations?

$O(2^n)$, $O(n^k)$, $O(log(n))$ ...?

- What is the **optimal** complexity bound?

Which algorithm provides it?

Motivation:

- A rigorous framework of computation, includes existing methods and helps to classify them
- Gives rise to new optimal and reliable algorithms

Some Recent Results (2018, jointly with M. Ziegler and I. Koswara)

The solution operator $R : \varphi \mapsto \mathbf{u}$ of the Cauchy problem for the system of ODE or PDE

$$\frac{\partial \mathbf{u}}{\partial t} = \mathcal{A}\mathbf{u}, \;\; \mathbf{u}(0) = \varphi$$

is **LOG-square-space computable** ($\Longrightarrow$ in particular, P-time computable)

Here $\varphi \in \mathcal{K}$, where $\mathcal{K}$ is a compact, convex, symmetric normed set, e.g $\varphi \in C^\infty([0,1]^m)$ is an analytic function; $\mathcal{A} : \mathcal{K} \to \mathcal{K}$ is a linear, symmetric, LOG-space computable operator. E.g.

$\mathcal{A}$ is a matrix $\Longrightarrow$ ODE

$\mathcal{A} = \sum_{i=1}^{m} B_i(x) \frac{\partial}{\partial x_i} \Longrightarrow$ PDE

If $\varphi \in C^k([0,1]^m)$ then **P-space computable**, for some cases $\#P^{\#P}$, goal: $\#P$

Setting (differential equations)

Systems of partial differential equations (**PDEs**)

$$\frac{\partial}{\partial t}\vec{u}(t,x) \;=\; \mathcal{A}(x)\vec{u}(t,x), \quad t \in [0;T], x \in Q = [0,1]^m$$

Cauchy problem: $\vec{u}(0,x) = \vec{\varphi}(x)$
If, in addition, $L\vec{u}\,|_{\partial Q} = \psi(y)$, $y \in \partial Q$, L linear operator $\Longrightarrow$
Boundary-value problem

- In general $\mathcal{A} = \sum_{j=1}^{m} B_j(x)\frac{\partial}{\partial x_j}$ first-order differential operator
  $\vec{u} = (u_1, u_2, \ldots, u_d)^T$, $B_j(x)$ are $d \times d$ matrices
- If $\mathcal{A}$ involves higher-order partial derivatives $\frac{\partial^k}{\partial x_{i_1}^{k_1} \ldots x_{i_j}^{k_j}}$, it can be reduced to a first-order system by introducing more variables

Setting (differential equations)

Systems of partial differential equations (**PDEs**)

$$\frac{\partial}{\partial t}\vec{u}(t,x) = \mathcal{A}(x)\vec{u}(t,x), \quad t \in [0;T], x \in Q = [0,1]^m$$

Cauchy problem: $\vec{u}(0,x) = \vec{\varphi}(x)$
If, in addition, $L\vec{u}\,|_{\partial Q} = \psi(y)$, $y \in \partial Q$, L linear operator $\Longrightarrow$
Boundary-value problem

- In general $\mathcal{A} = \sum_{j=1}^{m} B_j(x)\frac{\partial}{\partial x_j}$ first-order differential operator
  $\vec{u} = (u_1, u_2, \ldots, u_d)^T$, $B_j(x)$ are $d \times d$ matrices
- If $\mathcal{A}$ involves higher-order partial derivatives $\frac{\partial^k}{\partial x_{i_1}^{k_1} \ldots x_{i_j}^{k_j}}$, it can be reduced to a first-order system by introducing more variables
- If $\mathcal{A}(x) = A$ (a $d \times d$ matrix), $\vec{u}(t,x) = \vec{u}(t) \Longrightarrow$ linear **ODEs**

Brief Outline

- **Exact real** computations + **space complexity** vs **boolean circuit parallel complexity** (definitions below)

  We prove:
- $\mathbf{O}(\log^2 n)$ space complexity for Cauchy problems and analytic functions (proof by "Taylor series") with accuracy 2^{-n}
- $\mathbf{O}(n^2)$ space complexity for boundary-value problems and C^p functions (proof by difference schemes) with accuracy 2^{-n}

Space complexity: save **memory**, not time

$f : X \subseteq \mathbb{R}^d \to [0;1]$.

- Type-2 Turing Machine (TM) **computes** f: for any $\vec{a}_m$ of (integer, binary encoding) s.th.

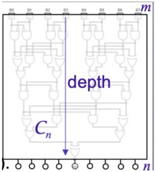
$$|\vec{x} - \vec{a}_m/2^m| \leq 2^{-m}$$

it outputs $b_n \in \mathbb{Z}$ s.th.

$$|f(\vec{x}) - b_n/2^n| \leq 2^{-n}$$

- The computation runs in **space** $s : \mathbb{N} \to \mathbb{N}$ if, until producing said $\vec{b}_n$, **the machine accesses at most** $s(n)$ **different cells of the working tape**
  f belongs to $\mathbb{R}\mathrm{SPACE}(s(n))$.

Boolean circuit Complexity: **parallel time**



[Hoover 1990]

- (C_n) **computes** f if C_n, on every input $\mathrm{bin}(\vec{a})$ with $|\vec{x} - \vec{a}/2^m| \leq 2^{-m}$ for some $\vec{x} \in X$, outputs $\mathrm{bin}(b_n)$ such that $|f(\vec{x}) - b_n/2^n| \leq 2^{-n}$.
- f is computable in **polysize-depth** $t(n)$, written $\mathbb{R}\mathrm{DEPTH}\big(t(n)\big)$, if there is a sequence (C_n) of Boolean circuits depth bounded $t(n)$ and size (#gates) bounded polynomially in n. $\mathbb{R}\mathrm{NC}^i$ abbreviates $\mathbb{R}\mathrm{DEPTH}\big(\log^i n\big)$.

$$\mathbb{R}\mathrm{NC}^1 \subseteq \mathbb{R}\mathrm{SPACE}(\log n) \subseteq \mathbb{R}\mathrm{NC}^2 \subseteq \mathbb{R}\mathrm{SPACE}(\log^2 n) \subseteq$$
$$\subseteq \mathbb{R}\mathrm{NC}^4 \subseteq \mathbb{R}\mathrm{P} \subseteq \mathbb{R}\mathrm{PSPACE}$$

Fact [Borodin 1977]: a) C_n of depth $d(n)$ and size $s(n)$ can be evaluated by a TM in time $O(s(n))$ and memory $O(d(n))$.
b) a TM with time $t(n)$ and memory $s(n)$ can be simulated by circuits C_n of depth $s^2(n)$ and of size $t(n) \cdot \log t(n)$

Theorem 1 Given $A \in [-1;1]^{d\times d}$ and $\vec{v} \in [-1;1]^d$, the solution to the first-order system of linear **ODEs**

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

is

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

and computable in space $O\big(\log(n) \cdot (\log d + \log n)\big)$.

Theorems for Cauchy problems $\varphi \mapsto u$

Theorem 1 Given $A \in [-1;1]^{d\times d}$ and $\vec{v} \in [-1;1]^d$, the solution to the first-order system of linear **ODEs**

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

is

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

and computable in space $O\big(\log(n) \cdot (\log d + \log n)\big)$.

Theorem 2 $\mathcal{B}$ Banach space; $\mathcal{V}_d \subseteq \mathcal{W}_d \subseteq \mathcal{B}$, linear operators $\mathbf{A_d} : \mathcal{V}_d \to \mathcal{W}_d$ s.th. $A_d^k : \mathcal{V}_d \to \mathcal{W}_d$ are well-defined and continuous.

Then $u(t) = \exp(tA)\vec{v} \in \mathcal{W}_d$ is well-defined for $\vec{\varphi} \in \mathcal{V}_d$, I $|t| \leq R_d/2$ and solves (2).

Here $R_d := 1/\limsup_k \sqrt[k]{|||A_d^k|||/k!}$.

If $\mathbf{A_d}$is computable on $\mathcal{V}_d$'s unit ball in space $O\big(s(d \cdot n)\big)$, $s(n) \geq \log n$, then $\mathbf{u}(\mathbf{t})$ is computable in space $O\big(s(d \cdot n) \cdot \log(n)\big)$.

Examples of application of Theorem 2 to PDEs

Example of spaces for PDEs: analytic functions

$$\mathcal{V}_d = \{ f : [0;1] \to \mathbb{R} \mid \exists K \in \mathbb{N}\ \forall k \in \mathbb{N}\ |f^{(k)}|_\infty \le K \cdot k! \cdot d^k \},$$

$$\|f\| := \sum\nolimits_k |f^{(k)}|_\infty / k!^2$$

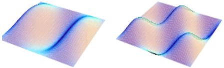
where $f^{(k)}$ denotes k-th iterated derivative and $|f|_\infty := \sup_x |f(x)|$. Then

$$\partial^k : \mathcal{V}_d \ni f \mapsto f^{(k)} \in \mathcal{V}_{2d} =: \mathcal{W}_d$$

is well-defined and $e^d d^k k!$-Lipschitz (so $R_d = 1/d$)
• ∂ is computable in $\mathbf{O}(\mathbf{log}(\mathbf{nd})) \Longrightarrow$ solution of the Cauchy problem for $\frac{\partial \mathbf{u}}{\partial \mathbf{t}} = \frac{\partial \mathbf{u}}{\partial \mathbf{x}}$ is computable in space $O\big(\log(nd) \cdot \log(n)\big)$

Examples of application of Theorem 2 to PDEs

- $\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \Longrightarrow \mathbf{u(t,x)} = \varphi(x+at) = \sum_{j=0}^{\infty} \frac{t^k}{k!}\left(a\frac{\partial}{\partial x}\right)^k \varphi(x)$
- heat equation $\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2} \Longrightarrow \mathbf{u(t,x)} = \sum_{j=0}^{\infty} \frac{t^k}{k!}\left(a\frac{\partial^2}{\partial x^2}\right)^k \varphi(x)$
- symmetric hyperbolic system $\vec{u}_t + \sum_{j=1}^{m} A^{-1}B_j\vec{u}_{x_j} = 0$ with constant matrix coefficients $B_j = B_j^*$, $A = A^* > 0$,
  $\mathbf{u(t,x)} = \sum_{j=0}^{\infty} \frac{t^k}{k!}\left(\sum_{j=1}^{m} B_j\frac{\partial}{\partial x_j}\right)^k \vec{\varphi}(\vec{x})$



- wave equation $u_{tt} = c^2\sum_{j=1}^{n} u_{x_jx_j}$
- more general Cauchy-Kovalevskaya type system
  $\vec{u}_t + \sum_{j=1}^{m} C_j(\vec{x})\vec{u}_{x_j} = 0$

Main lemmas:

- The sequence $1/k!$, considered as constant real functions, is computable in polysize-depth $O\big(\log^2(k) + \log n\big)$.
- The sequence of **matrix powering** functions $[-1;1]^{d\times d} \ni A \mapsto \mathbf{A^k} \in [-d^{k-1}; d^{k-1}]^{d\times d}$ is computable in polysize-depth $O\big(\log(k) \cdot (\log d + \log n + \log k)\big)$.

The proof heavily uses a special integer multiplication algorithm and recursive matrix powering.
Recently:

- Huge (exponential) matrix powering $A^{2^n}_{2^n}$ (Ivan's talk)
- (Differential) operator powering

Conclusion

- The above algorithm for Cauchy problems and analytic functions is highly efficient, not (yet) used in numerics.
- For correctly posed **boundary-value problems**, constant coefficients, $\varphi \in C^p(Q)$ (continuously differential functions, more general than analytic), we can get $O(n^2)$ space complexity estimates by using the above mentioned efficient matrix powering algorithms and classical approach of difference schemes.

Conclusion

- Future work:

  ◇ more general PDEs and boundary value problems;

  ◇ more general functional classes, e.g. Sobolev spaces;

  ◇ investigate other methods (e.g. Fourier transformation, group symmetry methods,...), by now: difference schemes, analytic power series

  ◇ optimality (by now improve from $\#P^{\#P}$ to $\#P$)

  Dirichlet Problem for $2D$ Poisson's Equation [Kawamura, Steinberg, Ziegler 2016] $\begin{cases} \Delta u = f \text{ on } B_2(0,1), \\ u = 0 \text{ on } \partial B_2(0,1) \end{cases}$ is $\#P$-"complete"

THANK YOU FOR YOUR ATTENTION!