Numeric-symbolic resolution Of Differential Equations

ANR PRME “NODE” — 320 730 € — 5 years — 150 p. months + 2 PhDs

Joris van der Hoeven

MAX team (symbolic computation)

CNRS, LIX, École polytechnique





Joris van der Hoeven

Grégoire Lecerf

Marc Mezzarobba

François Ollivier

Gleb Pogudin

Solving differential equations

$$\Phi_1(f_1(t), f_1'(t), \ldots, f_n(t), f_n'(t), \ldots) = 0$$

$$\ldots$$

$$\Phi_{n'}(f_1(t), f_1'(t), \ldots, f_n(t), f_n'(t), \ldots) = 0$$

Solving differential equations

$$\begin{aligned}\Phi_1(f_1(t), f_1'(t), \ldots, f_n(t), f_n'(t), \ldots) &= 0\\ &\ldots\\ \Phi_{n'}(f_1(t), f_1'(t), \ldots, f_n(t), f_n'(t), \ldots) &= 0\end{aligned}$$

State of the art

- Numerical solvers for machine precision
- Symbolic “solvers”: closed form, simplified equations

Part I: numerical resolution

Problem

Computational cost of solving a system of differential equations ?

Part I: numerical resolution

Problem

Computational cost of solving a system of differential equations ?

As a function of

- Precision (of the computations or of the guaranteed end-result)
- Type of equations (linear, special functions, stiff, singularities, ...)
- Computational model (sequential, parallel, hardware, ...)

Part I: numerical resolution

Problem

Computational cost of solving a system of differential equations ?

As a function of

- Precision (of the computations or of the guaranteed end-result)
- Type of equations (linear, special functions, stiff, singularities, ...)
- Computational model (sequential, parallel, hardware, ...)

Applications

- New library for the fast and reliable integration of systems of odes
- Differential Galois theory
- Control theory, flatness

Part II: symbolic resolution

Goal **(see also OCCAM project by Gleb Pogudin)**

Faster algorithms to solve/reduce systems of differential equations

Part II: symbolic resolution

Goal **(see also OCCAM project by Gleb Pogudin)**

Faster algorithms to solve/reduce systems of differential equations

Representation through power series solutions (instead of equations)

$$
\begin{aligned}
f_1(t) &= c_{1,0} + c_{1,1}t + c_{1,2}t^2 + c_{1,3}t^3 + c_{1,4}t^4 + \cdots \\
f_2(t) &= c_{2,0} + c_{2,1}t + c_{2,2}t^2 + c_{2,3}t^3 + c_{2,4}t^4 + \cdots \\
f_3(t) &= c_{3,0} + c_{3,1}t + c_{3,2}t^2 + c_{3,3}t^3 + c_{3,4}t^4 + \cdots
\end{aligned}
$$

$c_{i,j} \longrightarrow$ free parameters (initial conditions)

$c_{i,j} \longrightarrow$ constrained parameters (by polynomial equations)

Part II: symbolic resolution

Goal **(see also OCCAM project by Gleb Pogudin)**

Faster algorithms to solve/reduce systems of differential equations

Representation through power series solutions (instead of equations)

$$
\begin{aligned}
f_1(t) &= c_{1,0} + c_{1,1}t + c_{1,2}t^2 + c_{1,3}t^3 + c_{1,4}t^4 + \cdots \\
f_2(t) &= c_{2,0} + c_{2,1}t + c_{2,2}t^2 + c_{2,3}t^3 + c_{2,4}t^4 + \cdots \\
f_3(t) &= c_{3,0} + c_{3,1}t + c_{3,2}t^2 + c_{3,3}t^3 + c_{3,4}t^4 + \cdots
\end{aligned}
$$

$c_{i,j} \longrightarrow$ free parameters (initial conditions)

$c_{i,j} \longrightarrow$ constrained parameters (by polynomial equations)

Homotopy continuation for polynomial equations

- Solve simpler system that is similar (e.g. same degrees)
- Continuously deform into target system and follow solutions

Part II: symbolic resolution

Goal **(see also OCCAM project by Gleb Pogudin)**

Faster algorithms to solve/reduce systems of differential equations

Representation through power series solutions (instead of equations)

$$
\begin{aligned}
f_1(t) &= c_{1,0} + c_{1,1} t + c_{1,2} t^2 + c_{1,3} t^3 + c_{1,4} t^4 + \cdots \\
f_2(t) &= c_{2,0} + c_{2,1} t + c_{2,2} t^2 + c_{2,3} t^3 + c_{2,4} t^4 + \cdots \\
f_3(t) &= c_{3,0} + c_{3,1} t + c_{3,2} t^2 + c_{3,3} t^3 + c_{3,4} t^4 + \cdots
\end{aligned}
$$

$c_{i,j} \longrightarrow$ free parameters (initial conditions)

$c_{i,j} \longrightarrow$ constrained parameters (by polynomial equations)

Homotopy continuation for polynomial equations

- Solve simpler system that is similar (e.g. same degrees)
- Continuously deform into target system and follow solutions

Sparse interpolation

- Magic device to recover a sparse polynomial from evaluations