Data-types for Multidimensional Functions in Reliable Numerics

Implementations Inspired by Real Complexity Theory

Akitoshi Kawamura, Florian Steinberg, **Holger Thies**

The 19th Japan-Korea Joint Workshop on Algorithms and Computation
Hakodate, Japan

August 31, 2016

Ordinary Differential Equations

Consider the Initial Value Problem

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

for $F : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$.
For F continuous in t and Lipschitz-continuous in y the IVP has a unique solution y.
We want to find this solution.

Approach 1: Numerical Analysis

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y \end{aligned}$$

for $t \in \mathbb{R}$ **approximate** the solution $y(t)$.

Approach 1: Numerical Analysis

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y \end{aligned}$$

for $t \in \mathbb{R}$ **approximate** the solution $y(t)$.

Problem

- Usually there is no guarantee on how good the approximation is.
- The result might be useless

Approach 2: Interval Arithmetic

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y) \\ y(t_0) &\in [y_0] \end{aligned}$$

for $t \in \mathbb{R}$ **compute some interval** $[y(t)]$ containing the solution.

Approach 2: Interval Arithmetic

Given the IVP

for $t \in \mathbb{R}$ **comp**

Interval Arithmetic

$$[a] = [a^-, a^+],\ [b] = [b^-, b^+]$$

$$\begin{aligned}
[a] + [b] &= \left[a^- + b^-, a^+ + b^+\right] \\
[a] - [b] &= \left[a^- - b^+, a^+ - b^-\right] \\
[a] \times [b] &= \left[\min(a^- b^-, a^- b^+, a^+ b^-, a^+ b^+),\right. \\
&\qquad \left.\max(a^- b^-, a^- b^+, a^+ b^-, a^+ b^+)\right]
\end{aligned}$$

Approach 2: Interval Arithmetic

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y) \\ y(t_0) &\in [y_0] \end{aligned}$$

for $t \in \mathbb{R}$ **compute some interval** $[y(t)]$ containing the solution.

Problem

- The intervals might become very large.
- The result might still be useless

Approach 3: Computable Analysis

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

for $t \in \mathbb{R}$ **compute** the solution $y(t)$.

Approach 3: Computable Analysis

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

for $t \in \mathbb{R}$ **compute** the solution $y(t)$.
Or **compute** the solution function $y : \mathbb{R} \to \mathbb{R}^d$.

Approach 3: Computable Analysis

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

for $t \in \mathbb{R}$ **compute** the solution $y(t)$.
Or **compute** the solution function $y : \mathbb{R} \to \mathbb{R}^d$.

What does it mean to compute a real number or function?

TTE: Computable Real Numbers

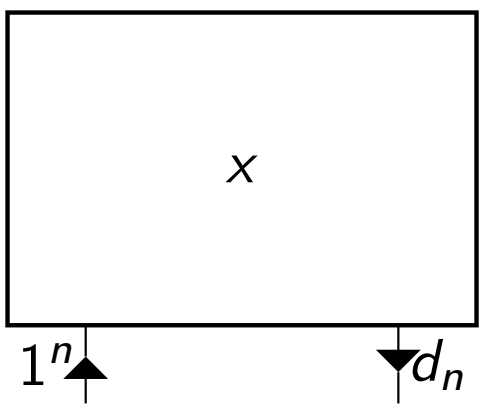
Computable Real Number

A real number is called computable if it can be approximated up to any desired precision.

TTE: Computable Real Numbers

Computable Real Number

A real number is called computable if it can be approximated up to any desired precision.



d_n rational approximations

$$|d_n - x| \leq 2^{-n}.$$

TTE: Computable Real Functions

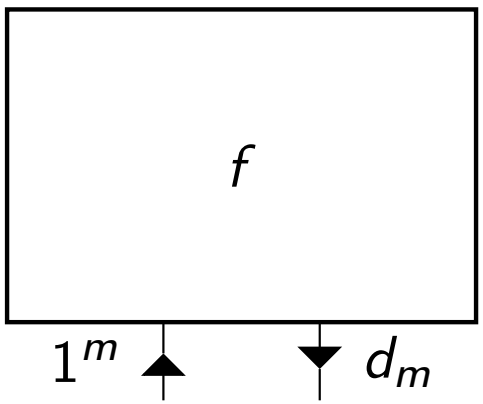
Computable Real Function

A function $f : [0,1] \to \mathbb{R}$ is called computable if the values $f(x)$ can be approximated up to any desired precision.

TTE: Computable Real Functions

Computable Real Function

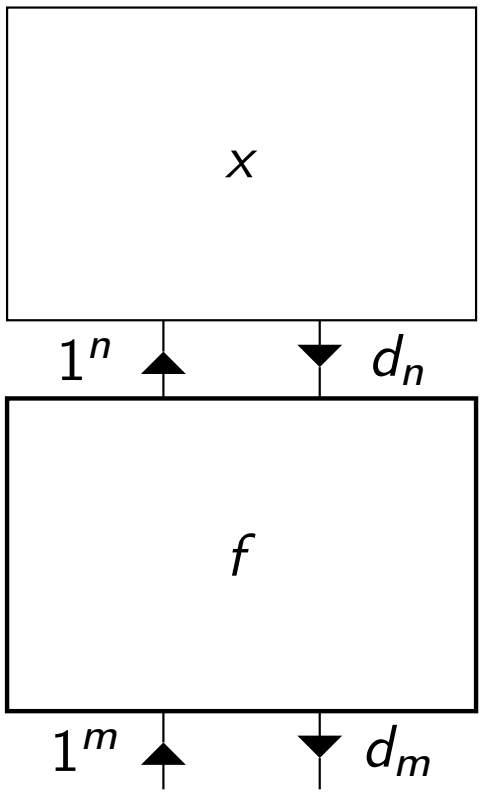
A function $f : [0,1] \to \mathbb{R}$ is called computable if the values $f(x)$ can be approximated up to any desired precision.



TTE: Computable Real Functions

Computable Real Function

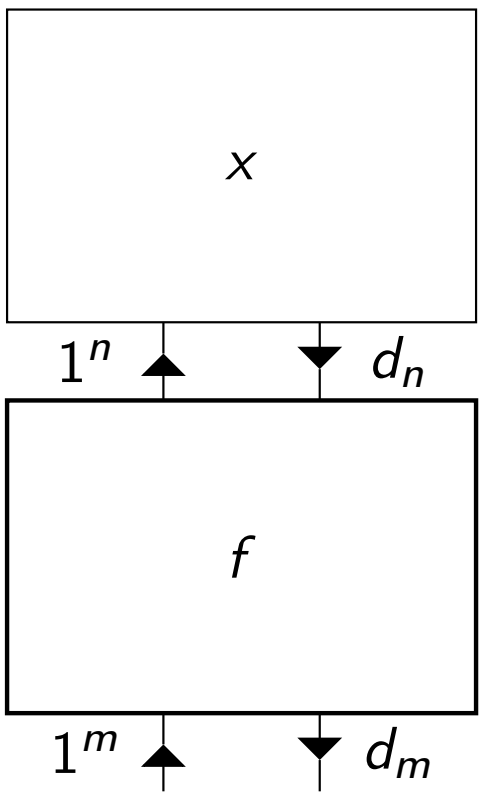
A function $f : [0,1] \to \mathbb{R}$ is called computable if the values $f(x)$ can be approximated up to any desired precision.



TTE: Computable Real Functions

Computable Real Function

A function $f : [0,1] \to \mathbb{R}$ is called computable if the values $f(x)$ can be approximated up to any desired precision.



Notes

- Computable functions are continuous.
- multidimensional functions can be defined by having several input oracles and several outputs.

The iRRAM Framework

- iRRAM is a C++ framework for exact real arithmetic written by Norbert Müller (University of Trier).
- REAL as data-type
- Interval Arithmetic is combined with Multiple Precision Arithmetic to get arbitrary small intervals.
- The computation is iterated with higher starting precision until the resulting interval is small enough
- New real numbers and functions can be defined (limit operator)

Some Results from Real Complexity Theory

Fact

For general polynomial time computable functions, many important operators have been shown to be computationally hard. For example

- *Polynomial time computable functions may have noncomputable derivatives. (Ko 1983)*
- *Parametric maximization is NP-hard. (Ko/Friedman (1982))*
- *Integration is #P-hard. (Friedman (1984))*

Complexity of Ordinary Differential Equations

Theorem (Kawamura, 2010)

Consider the IVP

$$y'(t) = f(t, y(t)) \quad ; \quad y(0) = 0.$$

There exists functions $f : [0,1] \times [-1,1] \to \mathbb{R}$ *and* $y : [0,1] \to [-1,1]$ *as above such that*

1. f *is Lipschitz-continuous and polynomial time computable*
2. y *is PSPACE-hard.*

Analytic Function

An analytic function is a function locally given by a complex power series.

Definition (Analytic Function)

$f : D \to \mathbb{C}$, $D \subseteq \mathbb{C}$ is analytic if for any $x_0 \in D$ the Taylor-series

$$T(x) := \sum_{n=0}^{\infty} a_n (x - x_0)^n$$

converges to $f(x)$ for x in a neighborhood of x_0.

Analytic Fu

An analyt
series.

Definitio

$f : D \to \mathbb{C}$, $D \subseteq \mathbb{C}$ is analytic if for any $x_0 \in D$ the Taylor-series

$$T(x) := \sum_{n=0}^{\infty} a_n (x - x_0)^n$$

converges to $f(x)$ for x in a neighborhood of x_0.

Computable Sequence

$(a_m)_{m \in \mathbb{N}}$ is computable if there is a machine that on input $1^n, 1^m$ outputs a rational approximation $d_{n,m}$ with $|d_{n,m} - a_m| \leq 2^{-n}$

Analytic Fu

An analyt
series.

Computable Sequence

$(a_m)_{m\in\mathbb{N}}$ is computable if there is a machine that on input $1^n, 1^m$ outputs a rational approximation $d_{n,m}$ with $|d_{n,m} - a_m| \leq 2^{-n}$

Definitio

$f : D \to \mathbb{C}$, $D \subseteq \mathbb{C}$ is analytic if for any $x_0 \in D$ the Taylor-series

$$T(x) := \sum_{n=0}^{\infty} a_n (x - x_0)^n$$

converges to $f(x)$ for x in a neighborhood of x_0.

Theorem (Pour-El, Richards, Ko, Friedman, Müller (1987/1989))

f is (polytime) computable iff $(a_m)_{m\in\mathbb{N}}$ is.

From that polynomial time computability of the derivative and the anti-derivative of a function follows immediately.

Analytic Functions as Type

How to represent analytic functions?

Analytic Functions as Type

How to represent analytic functions?

Theorem (Müller (1995))

The evaluation operator $((a_m)_{m\in\mathbb{N}}, x) \to f(x)$ *is not computable.*

Analytic Functions as Type

How to represent analytic functions?

Theorem (Müller (1995))

The evaluation operator $((a_m)_{m\in\mathbb{N}}, x) \to f(x)$ *is not computable.*

Lemma

Let $f : \overline{B}(0,1) \to \mathbb{R}$ *be analytic and* $(a_n)_{n\in\mathbb{N}}$ *its power series around* 0.
Then there exist $k, A \in \mathbb{N}$ *such that*

1. $\sqrt[k]{2}$ *is a lower bound on the radius of convergence*
2. $|a_n| \leq A \cdot 2^{-\frac{n}{k}}$

Analytic Functions as Type

How to represent analytic functions?

Theorem (Müller (1995))

The evaluation operator $((a_m)_{m\in\mathbb{N}}, x) \to f(x)$ *is not computable.*

Lemma

Let $f : \overline{B}(0,1) \to \mathbb{R}$ *be analytic and* $(a_n)_{n\in\mathbb{N}}$ *its power series around* 0.
Then there exist $k, A \in \mathbb{N}$ *such that*

1. $\sqrt[k]{2}$ *is a lower bound on the radius of convergence*
2. $|a_n| \leq A \cdot 2^{-\frac{n}{k}}$

Power series + A and k given $\Rightarrow$ many operations (e.g. evaluation, multiplication, differentiation) polynomial time computable [Kawamura, Rösnick, Müller, Ziegler (2013)]

Single Step Methods

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Compute a sequence $t_0 < t_1 < \cdots < t_n = T$ and values $y_0, \ldots, y_n$ such that $y_i = y(t_i)$.

Single Step Methods

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Compute a sequence $t_0 < t_1 < \cdots < t_n = T$ and values $y_0, \ldots, y_n$ such that $y_i = y(t_i)$.

Example (Euler's method)

Choose some step size h and iterate

$$\begin{aligned} t_{n+1} &= t_n + h \\ y_{n+1} &= y_n + h \cdot f(t_n, y_n) \end{aligned}$$

Single Step Methods

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Compute a sequence $t_0 < t_1 < \cdots < t_n = T$ and values $y_0, \ldots, y_n$ such that $y_i = y(t_i)$.

Example (Euler's method)

Choose some step size h and iterate

$$\begin{aligned} t_{n+1} &= t_n + h \\ y_{n+1} &= y_n + h \cdot f(t_n, y_n) \end{aligned}$$

Error in one step approx. proportional to h^2
$\Rightarrow$ need **exponentially** in n many steps for precision 2^{-n}.

Single Step Methods

Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Compute a sequence $t_0 < t_1 < \cdots < t_n = T$ and values $y_0, \ldots, y_n$ such that $y_i = y(t_i)$.

Example (Euler's method)

Choose some step size h and iterate

$$\begin{aligned} t_{n+1} &= t_n + h \\ y_{n+1} &= y_n + h \cdot f(t_n, y_n) \end{aligned}$$

Error in one step approx. proportional to h^2
$\Rightarrow$ need **exponentially** in n many steps for precision 2^{-n}.
The same is true for any fixed order method (e.g. Runge-Kutta).

Power Series Method

Given the IVP

$$\begin{aligned} \dot{y}_i(t) &= F_i(t, y_1(t), \ldots, y_d(t)) \\ y_i(0) &= 0 \end{aligned}$$

And M, r such that

- all F_i are analytic on a ball $\overline{B_r(0)}$ of radius r around 0
- M is such that $|F_i(x)| \leq M$ for all $x \in \overline{B_r(0)}$

Power Series Method

Given the IVP

$$\begin{aligned} \dot{y_i}(t) &= F_i(t, y_1(t), \ldots, y_d(t)) \\ y_i(0) &= 0 \end{aligned}$$

And M, r such that

- all F_i are analytic on a ball $\overline{B_r(0)}$ of radius r around 0
- M is such that $|F_i(x)| \leq M$ for all $x \in \overline{B_r(0)}$

Then

- The y_i are again analytic on a ball of radius $\frac{r}{M}$ and their absolute value is bounded by r.
- The coefficients of the power series can be computed in polynomial time if the F_v are polynomial time computable.

Power Series Method

Given the IVP

$$\begin{aligned} \dot{y}_i(t) &= F_i(t, y_1(t), \ldots, y_d(t)) \\ y_i(0) &= 0 \end{aligned}$$

And M, r such that

- all F_i are analytic on a ball $\overline{B_r(0)}$ of radius r around 0
- M is such that $|F_i(x)| \leq M$ for all $x \in \overline{B_r(0)}$

Then

- The y_i are again analytic on a ball of radius $\frac{r}{M}$ and their absolute value is bounded by r.
- The coefficients of the power series can be computed in polynomial time if the F_v are polynomial time computable.

However: The complexity is **exponential** in the dimension d

Power Series Method

The method can be used to construct a single-step method:

Power Series Method

The method can be used to construct a single-step method:
Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Choose a sequence $t_0 < \cdots < t_n = T$ s.t. $h_i := t_{i+1} - t_i < \frac{r}{M}$.
To compute values $y_i, \ldots y_n$ iterate for $j = 0 \ldots n-1$

Power Series Method

The method can be used to construct a single-step method:
Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Choose a sequence $t_0 < \cdots < t_n = T$ s.t. $h_i := t_{i+1} - t_i < \frac{r}{M}$.
To compute values $y_i, \ldots y_n$ iterate for $j = 0 \ldots n-1$

- Let $F'(t, y) := F(t + t_j, y + y_j)$
- Let $r' = r - \max(t_j, y_j)$
- Solve the IVP $\dot{y'} = F'(t', y')\,;\ y'(0) = 0$
- It is $y(t) = y'(t - t_j) + y_j$
- Thus let $y_{j+1} = y_j + y'(h_i)$

Power Series Method

The method can be used to construct a single-step method:
Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Choose a sequence $t_0 < \cdots < t_n = T$ s.t. $h_i := t_{i+1} - t_i < \frac{r}{M}$.
To compute values $y_i, \ldots y_n$ iterate for $j = 0 \ldots n-1$

- Let $F'(t, y) := F(t + t_j, y + y_j)$
- Let $r' = r - \max(t_j, y_j)$
- Solve the IVP $\dot{y}' = F'(t', y')\,;\, y'(0) = 0$
- It is $y(t) = y'(t - t_j) + y_j$
- Thus let $y_{j+1} = y_j + y'(h_i)$

The step size does not depend on the precision!

Power Series Method

The method can be used to construct a single-step method:
Given the IVP

$$\begin{aligned} \dot{y}(t) &= F(t, y(t)) \\ y(t_0) &= y_0 \end{aligned}$$

Choose a sequence $t_0 < \cdots < t_n = T$ s.t. $h_i := t_{i+1} - t_i < \frac{r}{M}$.
To compute values $y_i, \ldots y_n$ iterate for $j = 0 \ldots n-1$

- Let $F'(t, y) := F(t + t_j, y + y_j)$
- Let $r' = r - \max(t_j, y_j)$
- Solve the IVP $\dot{y}' = F'(t', y')\,;\, y'(0) = 0$
- It is $y(t) = y'(t - t_j) + y_j$
- Thus let $y_{j+1} = y_j + y'(h_i)$

The step size does not depend on the precision!
For small output precision or high dimension fixed order methods might still be more efficient.

Implementation

`ANALYTIC<d>`: `C++` Class Template for d-dimensional real analytic functions $F : \mathbb{R}^d \to \mathbb{R}$.

Representation: power series + real numbers r, M.

Implementation

`ANALYTIC<d>`: C++ ... analytic functions $F : \mathbb{R}^d \to$ ...
Representation: pow...

Multidimensional Powerseries

$$\sum_{i\in\mathbb{N}}\sum_{j\in\mathbb{N}} a_{i,j}x_1^i x_2^j = \sum_{i\in\mathbb{N}} b_i x_1^i$$

$$\text{with } b_i := \sum_{j\in\mathbb{N}} a_{i,j}x_2^j$$

Computing $b_i \to$ evaluating an analytic function.

Implementation

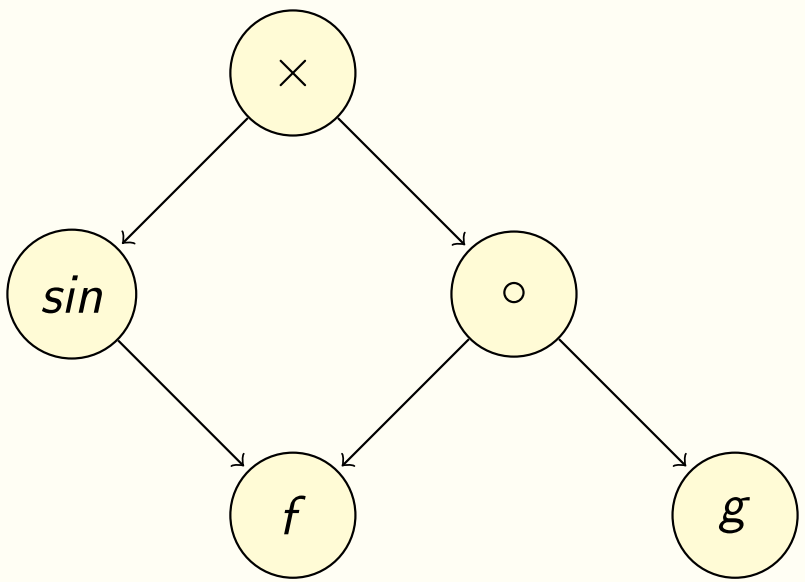
`ANALYTIC<d>`: `C++` Class Template for d-dimensional real analytic functions $F : \mathbb{R}^d \to \mathbb{R}$.

Representation: power series + real numbers r, M.

Implementation

`ANALYTIC<d>`: `C++` Class Template for d-dimensional real analytic functions $F : \mathbb{R}^d \to \mathbb{R}$.
Representation: power series + real numbers r, M.

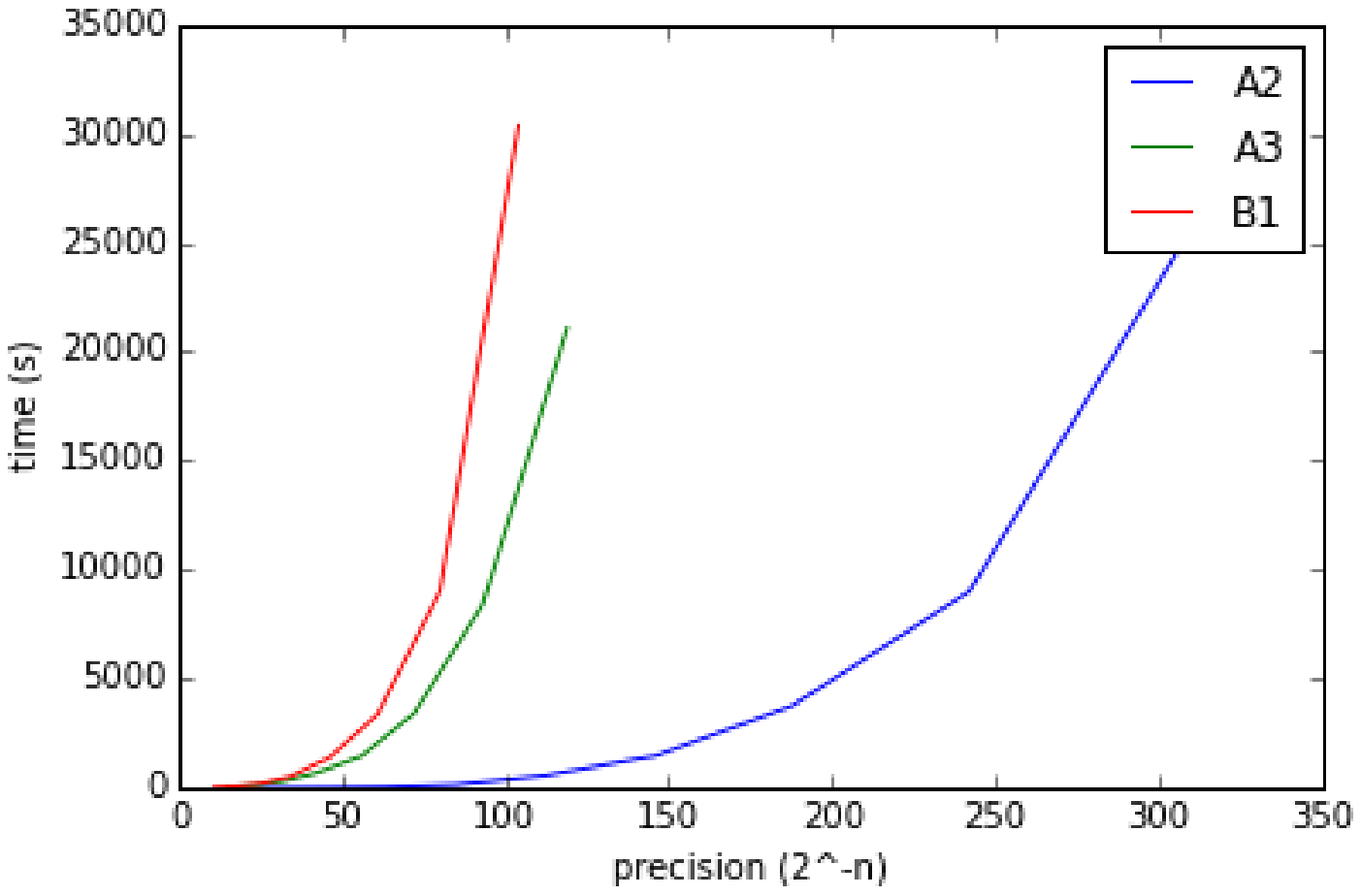
DAG Approach



Implemented e.g.

- Arithmetic $+$, $-$, $\times$, $\div$
- Partial differentiation
- Computing power series around new point
- Solving Initial Value Problems

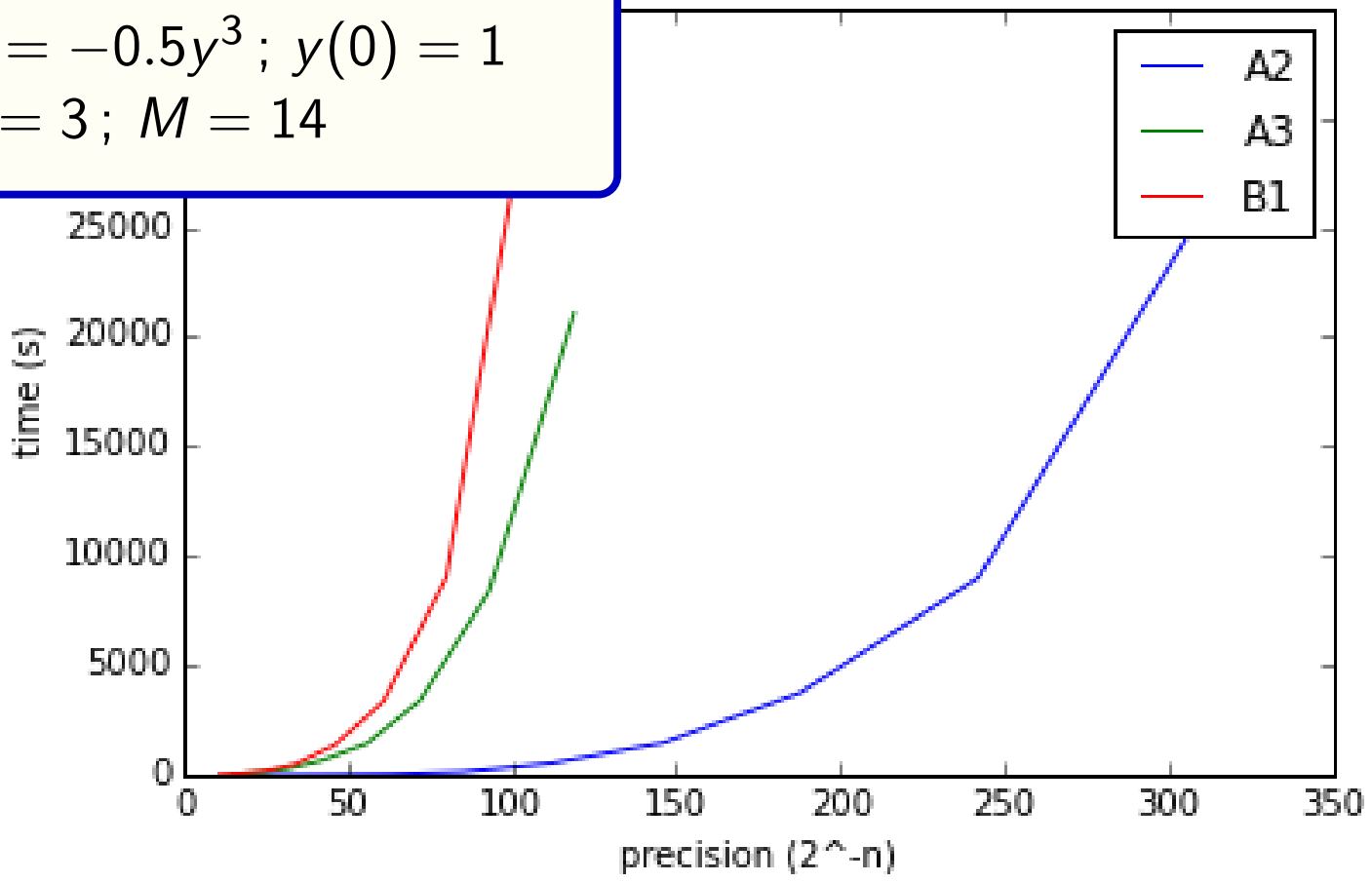
Evaluation: Precision vs CPU time



Evaluation: Precision vs CPU time

Problem A2

$\dot{y} = -0.5y^3\,;\; y(0) = 1$
$r = 3\,;\; M = 14$



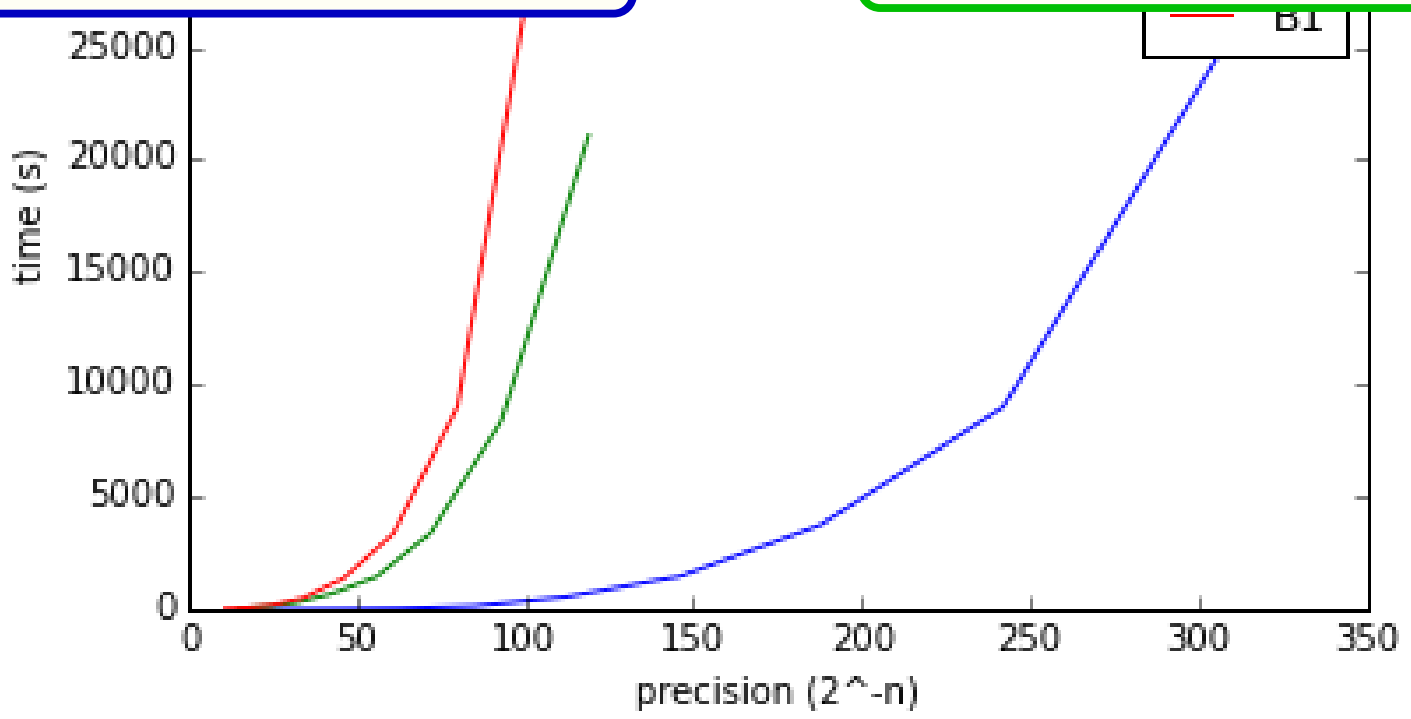
Evaluation: Precision vs CPU time

Problem A2

$\dot{y} = -0.5y^3\,;\ y(0) = 1$
$r = 3\,;\ M = 14$

Problem A3

$\dot{y} = ycos(t)\,;\ y(0) = 1$
$r = 4\,;\ M = 220$



Evaluation: Precision vs CPU time

Problem A2

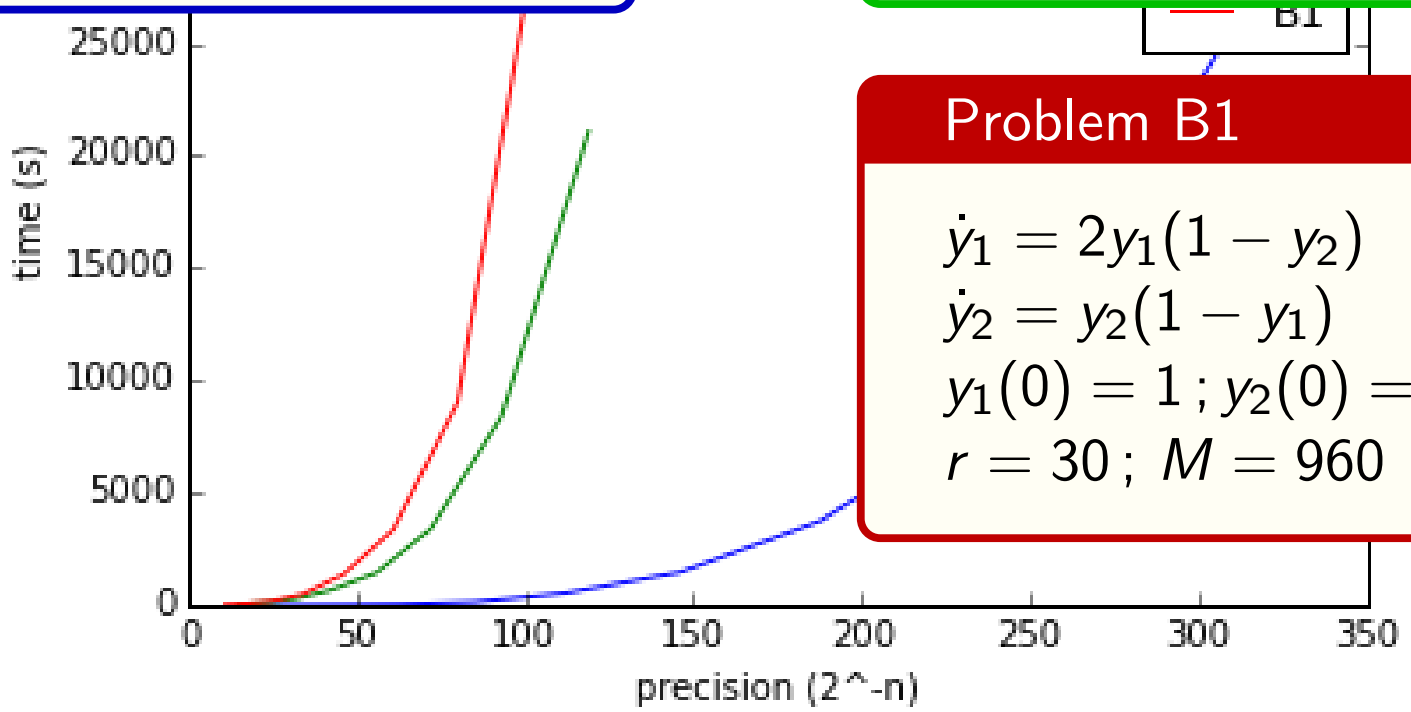
$\dot{y} = -0.5y^3\,;\; y(0) = 1$
$r = 3\,;\; M = 14$

Problem A3

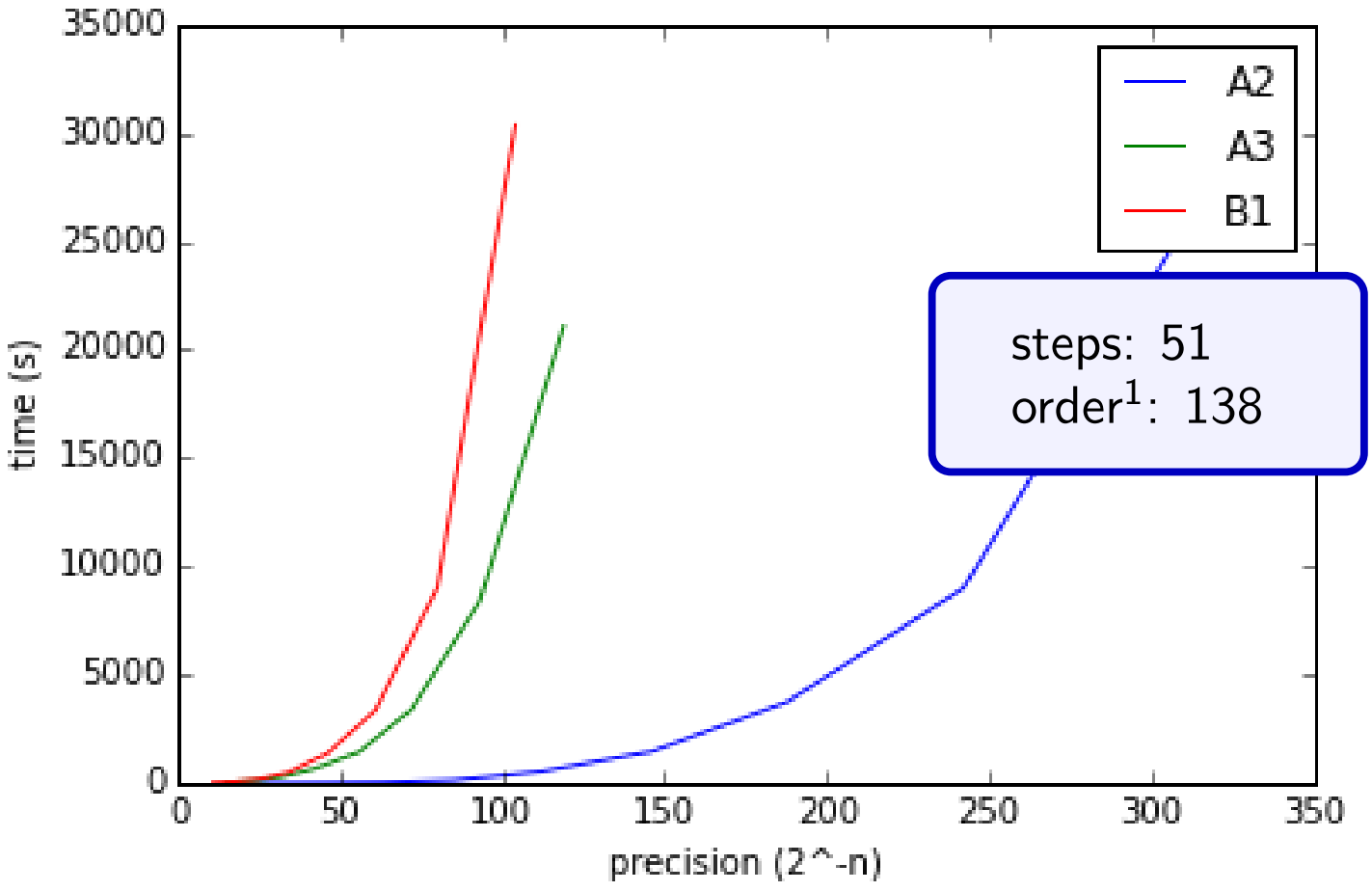
$\dot{y} = ycos(t)\,;\; y(0) = 1$
$r = 4\,;\; M = 220$

Problem B1

$\dot{y}_1 = 2y_1(1 - y_2)$
$\dot{y}_2 = y_2(1 - y_1)$
$y_1(0) = 1\,; y_2(0) = 3$
$r = 30\,;\; M = 960$

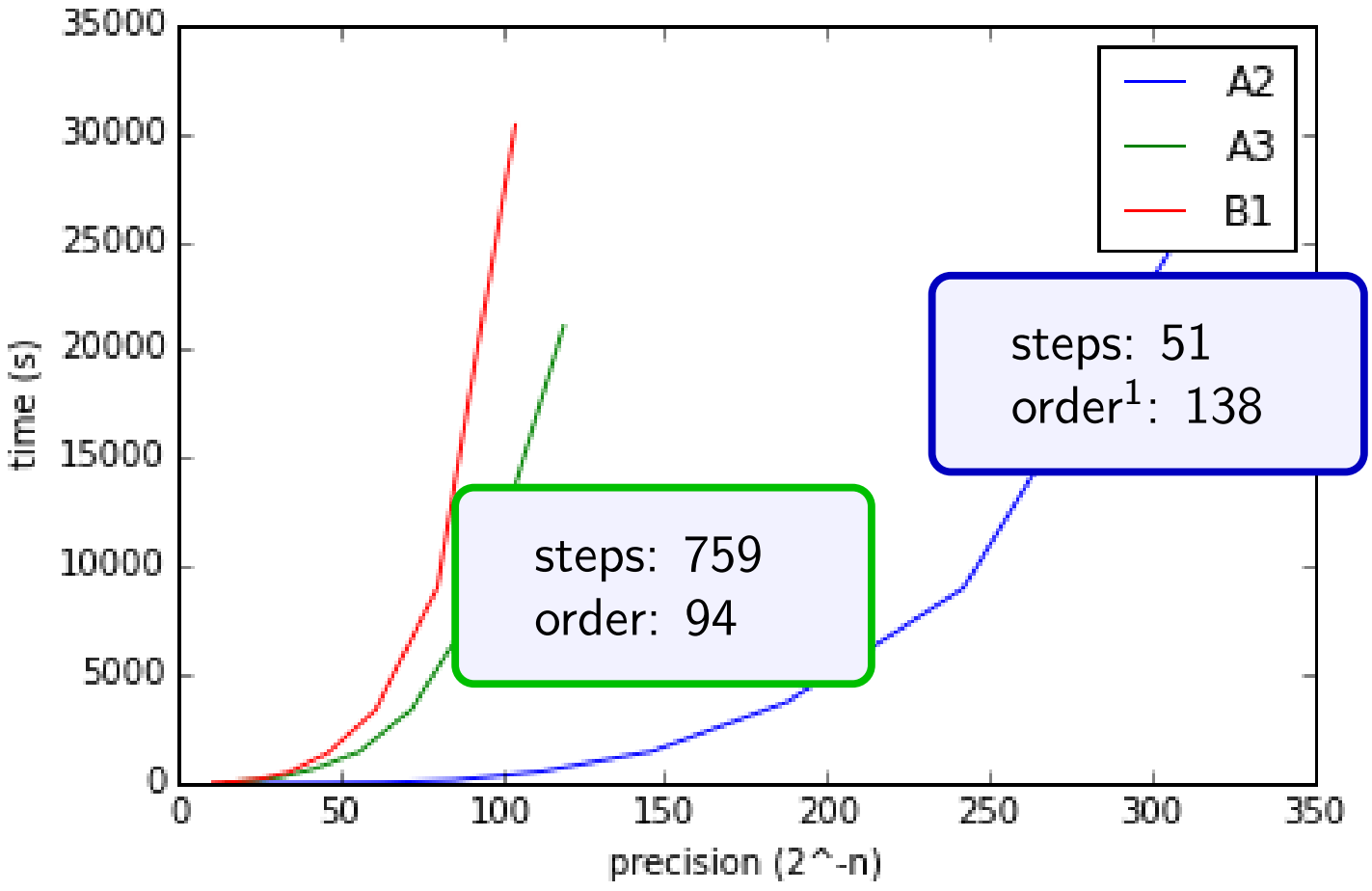


Evaluation: Precision vs CPU time



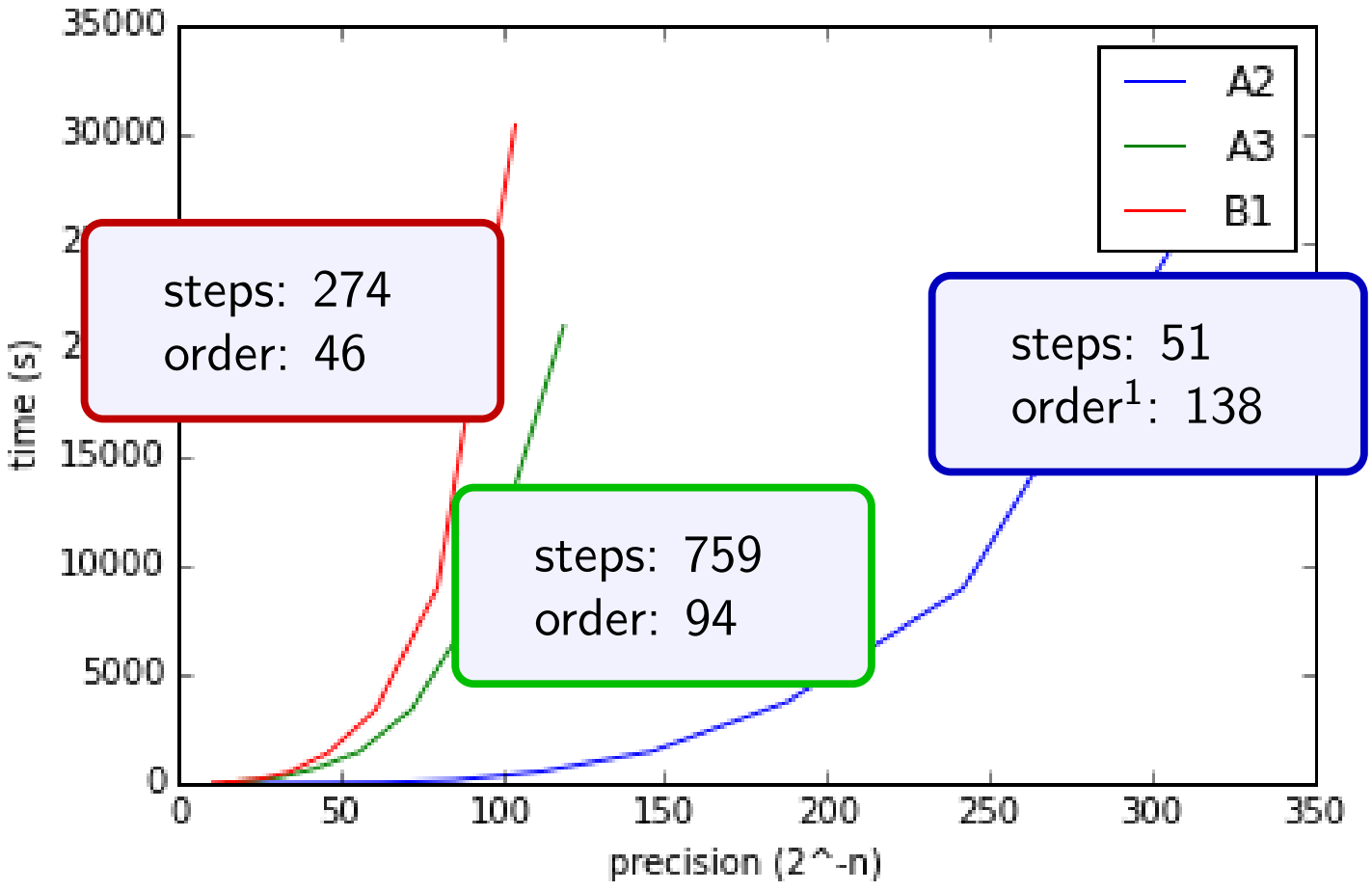
[1] coefficients read at the top-most level when initial precision is -153.

Evaluation: Precision vs CPU time



[1] coefficients read at the top-most level when initial precision is -153.

Evaluation: Precision vs CPU time



[1] coefficients read at the top-most level when initial precision is -153.

Evaluation: Dimension vs CPU time

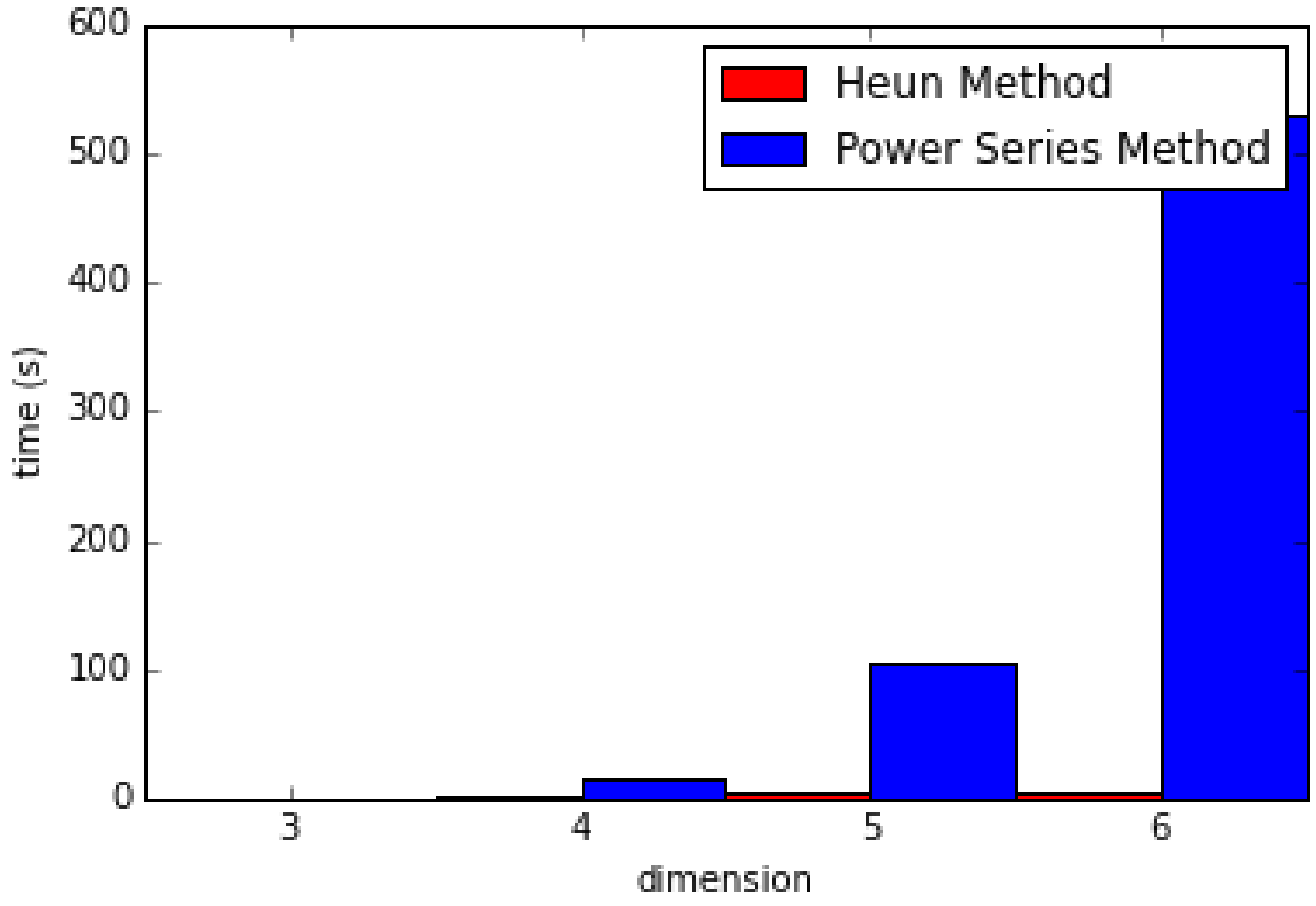


Figure: Evaluating at T=1.0 with precision 2^{-20}

Evaluation: Dimension vs CPU time

DETEST Problem C1 (modified)

$$\dot{y} = \begin{bmatrix} 0 & -1 & 0 & 0 & \dots & \\ 0 & 1 & -1 & 0 & \dots & \\ \vdots & 0 & \ddots & \ddots & \ddots & \\ & & & & -1 & 0 \\ & & & & 1 & 0 \end{bmatrix} \begin{bmatrix} t \\ y_1 \\ \\ \vdots \\ y_{d-1} \end{bmatrix} \quad y(0) = \begin{bmatrix} 1 \\ 0 \\ \\ \vdots \\ 0 \end{bmatrix}$$

$r = 100\,;\ M = 200$

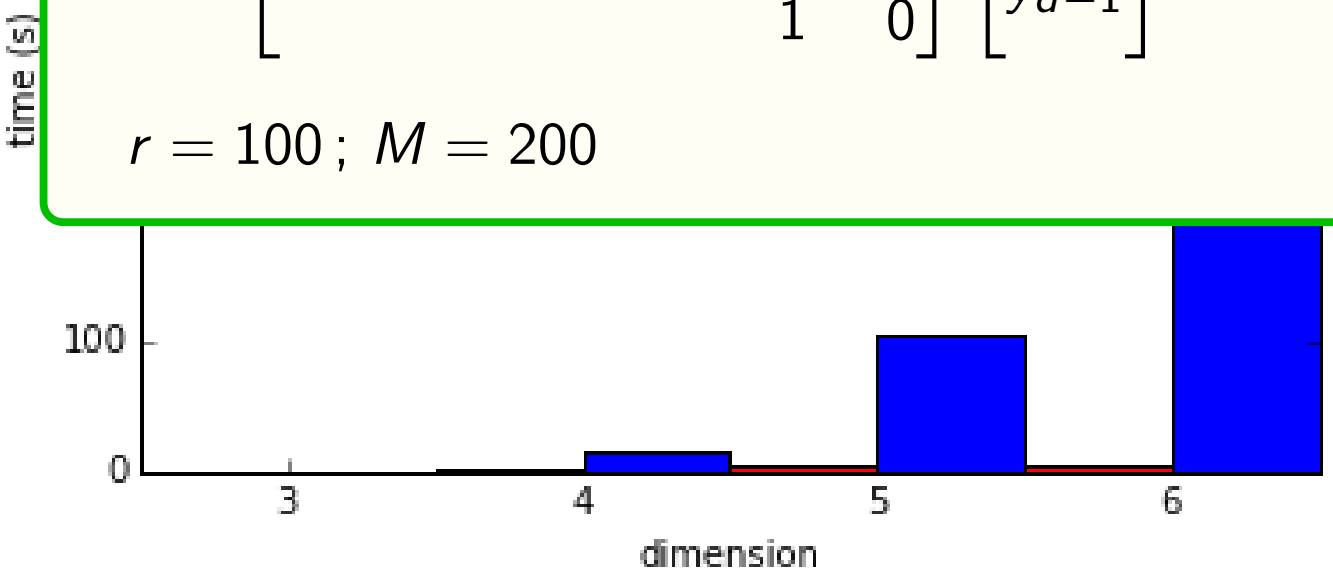


Figure: Evaluating at T=1.0 with precision 2^{-20}

Evaluation: Dimension vs CPU time

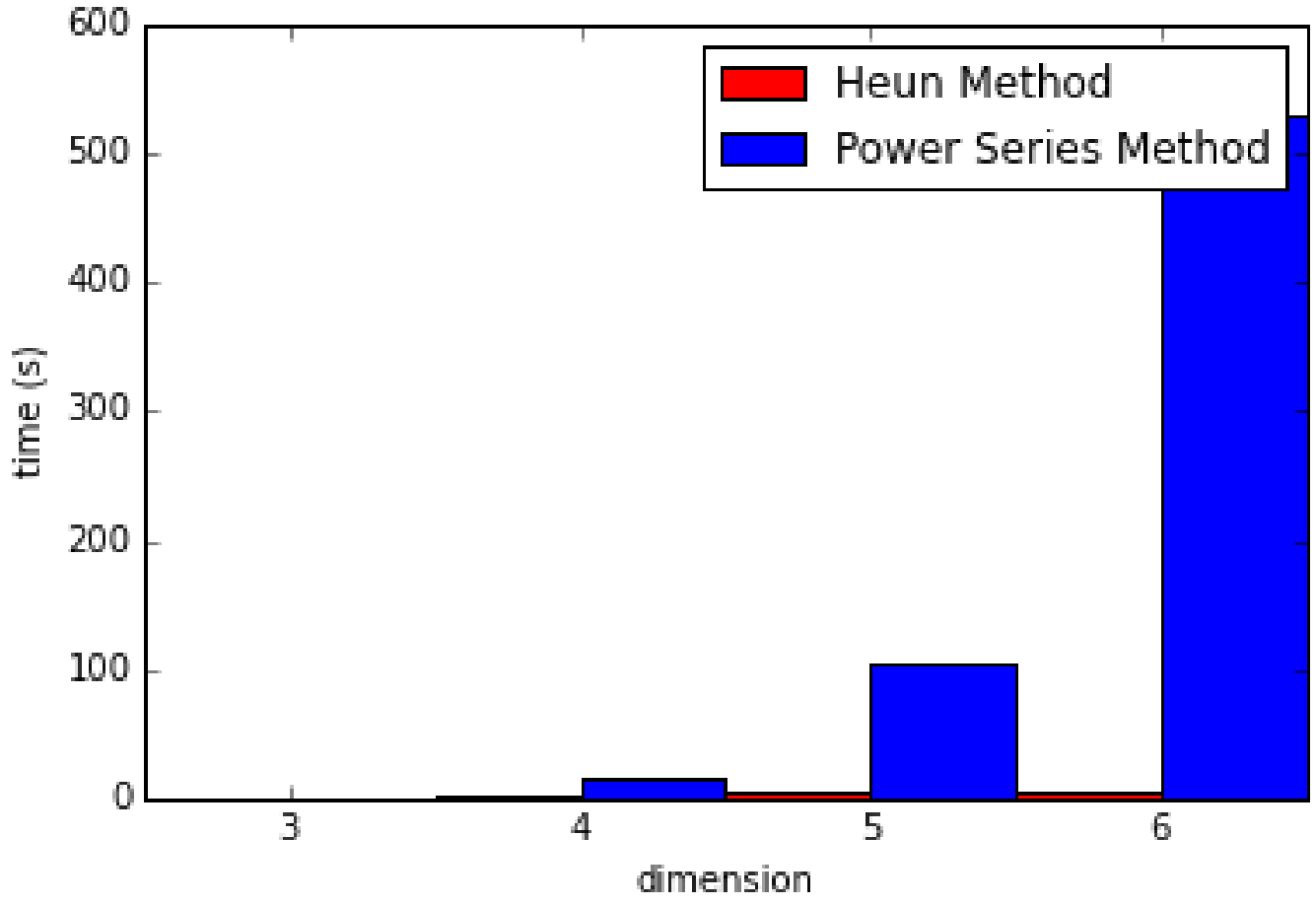


Figure: Evaluating at T=1.0 with precision 2^{-20}

Evaluation: Dimension vs CPU time

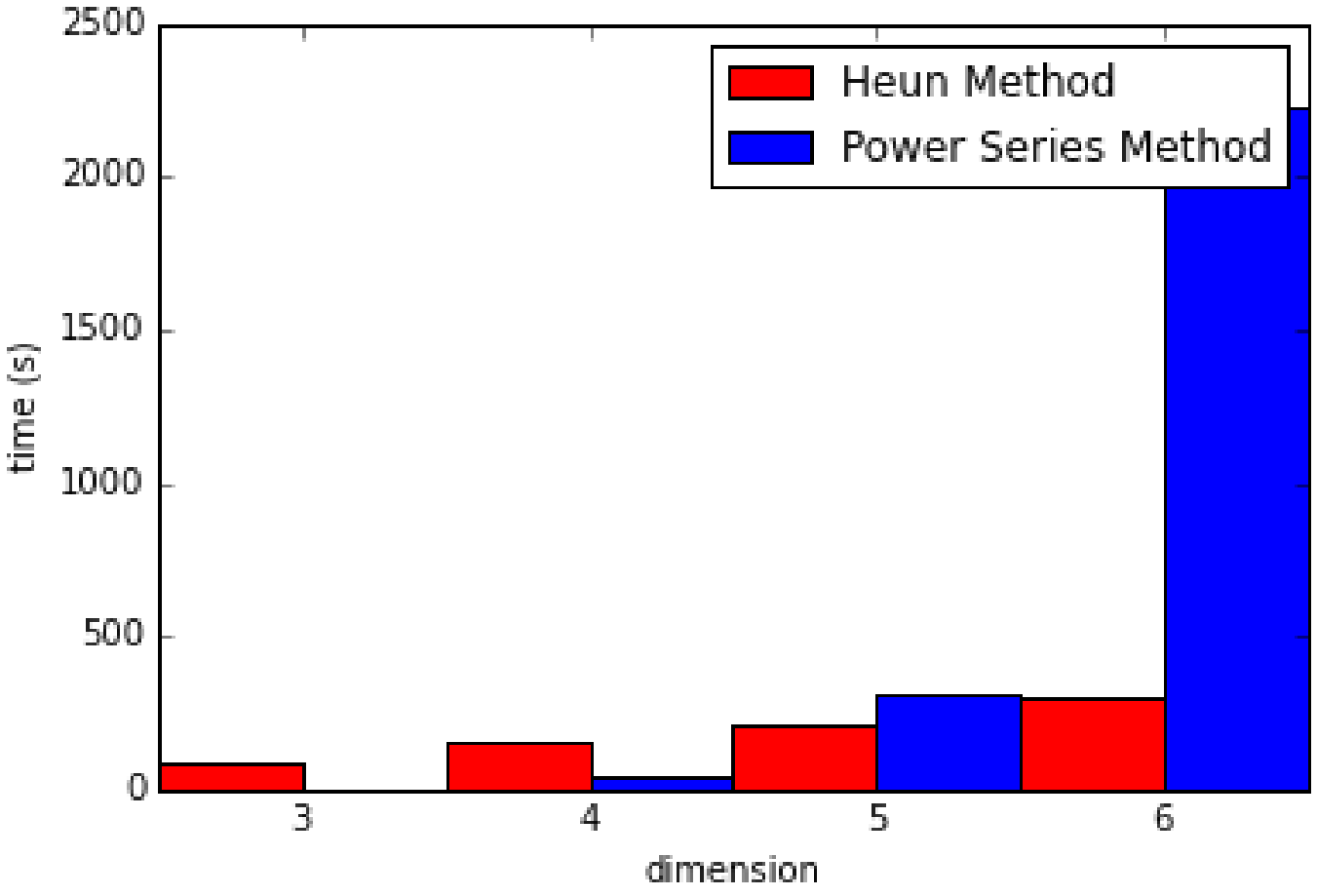


Figure: Evaluating at T=1.0 with precision 2^{-35}

Future Work

- Analyze complexity of algorithms in detail
- Implement higher order Runge-Kutta method
- Local parameters instead of M, r
- Specialized functions for different types of ODEs
- Multi-step Methods
- Combination of different methods and automatic choice depending on dimension/precision
- Extensions to Interval Arithmetic (Taylor models, Affine arithmetic)
- Parallelization
- Applications