Formal Verification of ODE-Solvers (and an Application to the Lorenz Attractor)

Fabian Immler

Chair for Logic and Verification (Tobias Nipkow)
Institut für Informatik, Technische Universität München

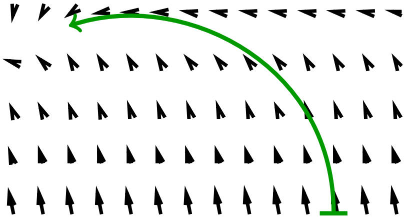
SpecFun Seminar "Computations and Proofs"
2014-12-08

Ordinary Differential Equations (ODEs)

- ODEs: modelling continuous “real world”

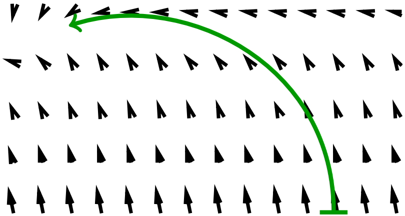
Ordinary Differential Equations (ODEs)

- ODEs: modelling continuous "real world"



Ordinary Differential Equations (ODEs)

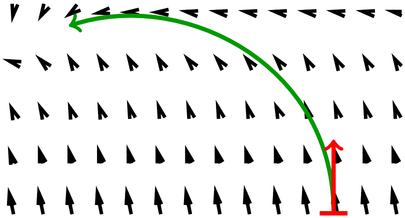
- ODEs: modelling continuous "real world"



- numerical approximations

Ordinary Differential Equations (ODEs)

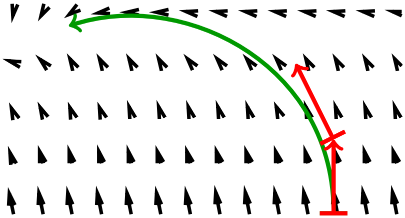
- ODEs: modelling continuous "real world"



- numerical approximations

Ordinary Differential Equations (ODEs)

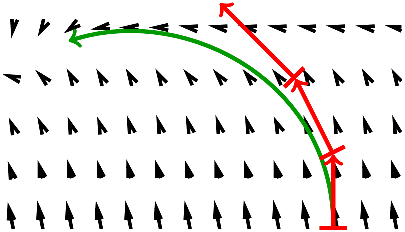
- ODEs: modelling continuous “real world”



- numerical approximations

Ordinary Differential Equations (ODEs)

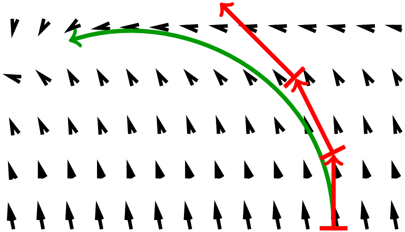
- ODEs: modelling continuous “real world”



- numerical approximations

Ordinary Differential Equations (ODEs)

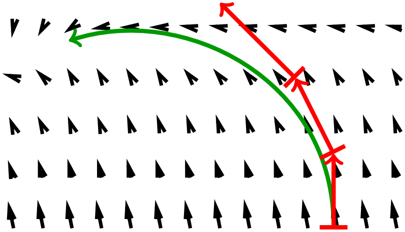
- ODEs: modelling continuous "real world"



- numerical approximations
  - discretization error

Ordinary Differential Equations (ODEs)

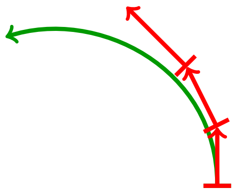
- ODEs: modelling continuous “real world”



- numerical approximations
  - discretization error
  - finite precision

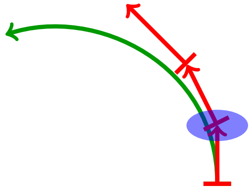
“Guaranteed Integration”

- computing with enclosures



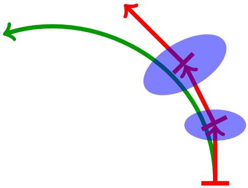
“Guaranteed Integration”

- computing with enclosures



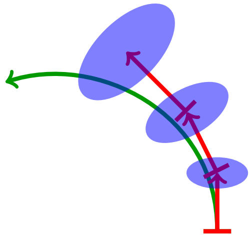
"Guaranteed Integration"

- computing with enclosures



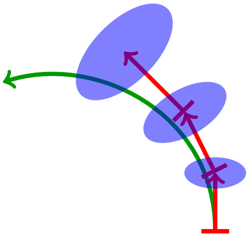
“Guaranteed Integration”

- computing with enclosures



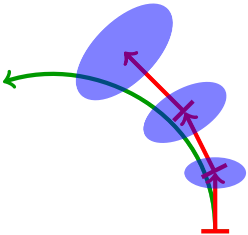
“Guaranteed Integration”

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series



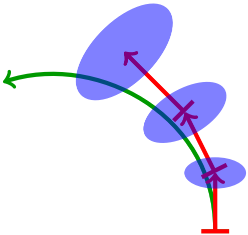
"Guaranteed Integration"

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration



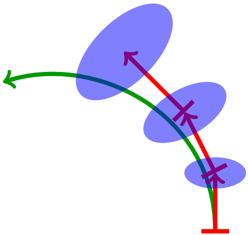
"Guaranteed Integration"

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta



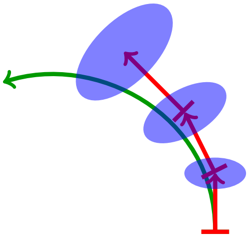
“Guaranteed Integration”

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta
- issue: correctness of computed enclosures?



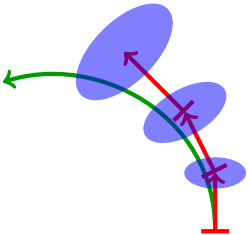
"Guaranteed Integration"

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta
- issue: correctness of computed enclosures?
  - VNODE-LP – literate programming



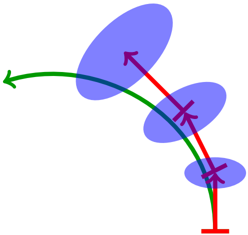
“Guaranteed Integration”

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta
- issue: correctness of computed enclosures?
  - VNODE-LP – literate programming
  - Taylor Models: formalizations in Coq



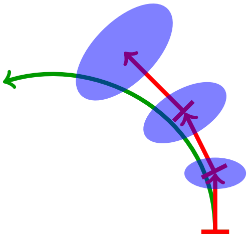
"Guaranteed Integration"

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta
- issue: correctness of computed enclosures?
  - VNODE-LP – literate programming
  - Taylor Models: formalizations in Coq
  - Picard iteration: formalization in Coq



“Guaranteed Integration”

- computing with enclosures
  - VNODE-LP (Nedialkov):
    interval arithmetic + Taylor series
  - COSY (Makino, Berz):
    Taylor models + Picard iteration
  - GRKLIB (Bouissou *et al.*):
    affine arithmetic + Runge-Kutta
- issue: correctness of computed enclosures?
  - VNODE-LP – literate programming
  - Taylor Models: formalizations in Coq
  - Picard iteration: formalization in Coq



Contribution

verification of (Bouissou et al.'s) algorithm
in interactive theorem prover Isabelle/HOL

- formal language to express mathematics

- formal language to express mathematics
- mechanically checked proofs

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
  - $0, \mathsf{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
  - $0, \mathsf{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$
  - multivariate analysis $\mathbb{R}^n$

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
  - $0, \mathsf{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$
  - multivariate analysis $\mathbb{R}^n$
  - ODEs, existence/uniqueness

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
  - $0, \mathsf{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$
  - multivariate analysis $\mathbb{R}^n$
  - ODEs, existence/uniqueness
- functional programming

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
    - $0, \mathrm{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$
    - multivariate analysis $\mathbb{R}^n$
    - ODEs, existence/uniqueness
- functional programming
- code extraction

- formal language to express mathematics
- mechanically checked proofs
- constructed from first principles:
  - $0, \mathrm{Succ} \rightsquigarrow \mathbb{N} \rightsquigarrow \mathbb{Z} \rightsquigarrow \mathbb{Q} \rightsquigarrow \mathbb{R}$
  - multivariate analysis $\mathbb{R}^n$
  - ODEs, existence/uniqueness
- functional programming
- code extraction

Theorem (mechanically checked)

functional program computes enclosure for unique solution of ODE

Overview

- Numerics

Overview

- Numerics
- Rigorous Numerics / Enclosures

Overview

- Numerics
- Rigorous Numerics / Enclosures
- ODEs

Overview

- Numerics
- Rigorous Numerics / Enclosures
- ODEs
  - certification

Overview

- Numerics
- Rigorous Numerics / Enclosures
- ODEs
  - certification
  - discretization

Overview

- Numerics
- Rigorous Numerics / Enclosures
- ODEs
  - certification
  - discretization
- optimizations

Overview

- Numerics
- Rigorous Numerics / Enclosures
- ODEs
  - certification
  - discretization
- optimizations
- application: Tucker's "computer-aided" proof for the Lorenz attractor

Numerics

- algorithms formalized for abstract type $\mathbb{R}$

Numerics

- algorithms formalized for abstract type $\mathbb{R}$
- $\mathbb{R}$ represented by $\mathbb{F} = \{m \cdot 2^e \mid m, e \in \mathbb{Z}\}$

Numerics

- algorithms formalized for abstract type $\mathbb{R}$
- $\mathbb{R}$ represented by $\mathbb{F} = \{m \cdot 2^e \mid m, e \in \mathbb{Z}\}$
  - $\mathit{Real}(m \cdot 2^e) \in \mathbb{R}$

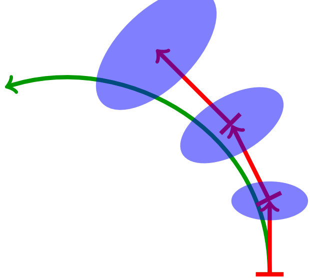
Numerics

- algorithms formalized for abstract type $\mathbb{R}$
- $\mathbb{R}$ represented by $\mathbb{F} = \{m \cdot 2^e \mid m, e \in \mathbb{Z}\}$
  - $\mathit{Real}(m \cdot 2^e) \in \mathbb{R}$
  - e.g., $\mathit{Real}(x) + \mathit{Real}(y) = \mathit{Real}(x + y)$

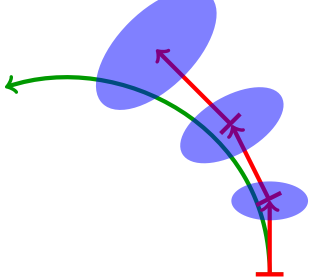
Numerics

- algorithms formalized for abstract type $\mathbb{R}$
- $\mathbb{R}$ represented by $\mathbb{F} = \{m \cdot 2^e \mid m, e \in \mathbb{Z}\}$
  - $\mathit{Real}(m \cdot 2^e) \in \mathbb{R}$
  - e.g., $\mathit{Real}(x) + \mathit{Real}(y) = \mathit{Real}(x + y)$
- explicitly round m

Rigorous Numerics / Enclosures

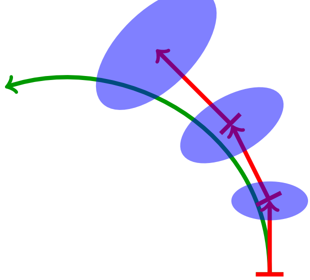


Rigorous Numerics / Enclosures



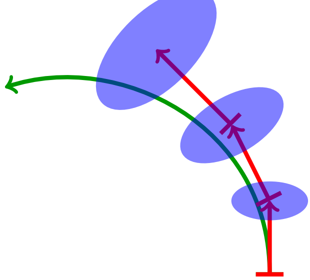
interval arithmetic:

Rigorous Numerics / Enclosures



interval arithmetic:
dependency problem:

Rigorous Numerics / Enclosures

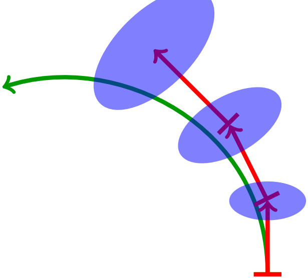


interval arithmetic:

dependency problem:

$x \in [0;1] \Longrightarrow x - x \in [0;1] - [0;1] = [-1;1]$

Rigorous Numerics / Enclosures

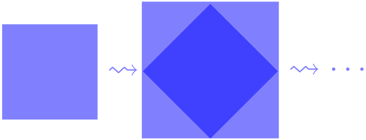


interval arithmetic:

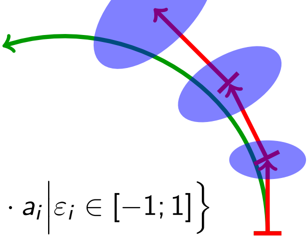
dependency problem:

$x \in [0;1] \Longrightarrow x - x \in [0;1] - [0;1] = [-1;1]$

wrapping effect:

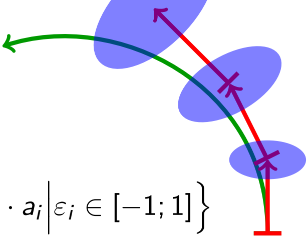


Rigorous Numerics / Enclosures



affine form: $\gamma\langle a_0, \ldots, a_k\rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

Rigorous Numerics / Enclosures

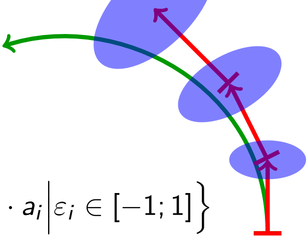


affine form: $\gamma\langle a_0, \ldots, a_k \rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \ldots, a_k \rangle = \langle Aa_0, \ldots, Aa_k \rangle$

Rigorous Numerics / Enclosures

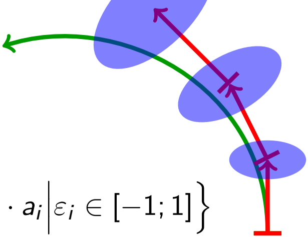


affine form: $\gamma\langle a_0, \ldots, a_k\rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \ldots, a_k\rangle = \langle Aa_0, \ldots, Aa_k\rangle$
- rotation without wrapping effect

Rigorous Numerics / Enclosures



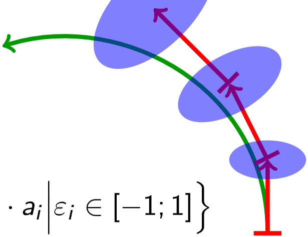
affine form: $\gamma\langle a_0, \ldots, a_k \rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \ldots, a_k \rangle = \langle Aa_0, \ldots, Aa_k \rangle$
- rotation without wrapping effect

nonlinear operations $(*, /)$: approximation with quadratic error

Rigorous Numerics / Enclosures



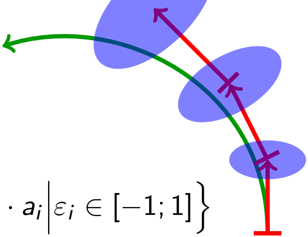
affine form: $\gamma\langle a_0, \ldots, a_k\rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \ldots, a_k\rangle = \langle Aa_0, \ldots, Aa_k\rangle$
- rotation without wrapping effect

nonlinear operations $(*, /)$: approximation with quadratic error
round-off errors: fresh noise symbols ε_i

Rigorous Numerics / Enclosures



affine form: $\gamma\langle a_0, \ldots, a_k\rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1; 1] \right\}$

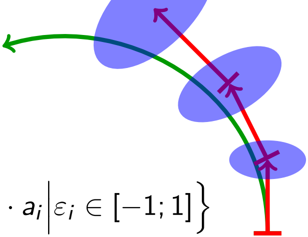
linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \ldots, a_k\rangle = \langle Aa_0, \ldots, Aa_k\rangle$
- rotation without wrapping effect

nonlinear operations $(*, /)$: approximation with quadratic error
round-off errors: fresh noise symbols ε_i

formalization: functions $\{a : \mathbb{N} \to \mathbb{R}^n \mid \mathit{finite}\ \{i \mid a_i \neq 0\}\}$

Rigorous Numerics / Enclosures



affine form: $\gamma\langle a_0, \dots, a_k\rangle = \left\{ a_0 + \sum_{i=1}^{k} \varepsilon_i \cdot a_i \middle| \varepsilon_i \in [-1;1] \right\}$

linear operation $A : \mathbb{R}^n \to \mathbb{R}^m$:

- $A\langle a_0, \dots, a_k\rangle = \langle Aa_0, \dots, Aa_k\rangle$
- rotation without wrapping effect

nonlinear operations $(*, /)$: approximation with quadratic error
round-off errors: fresh noise symbols ε_i

formalization: functions $\{a : \mathbb{N} \to \mathbb{R}^n \mid \textit{finite}\ \{i \mid a_i \neq 0\}\}$
implementation: sparse lists of sorted coefficients $(\mathbb{N} \times \mathbb{R}^n)$ *list*

ODEs

Initial value problem (IVP)

- $\dot{x}(t) = f(x(t))$

ODEs

Initial value problem (IVP)

- $\dot{x}(t) = f(x(t))$
- $x(0) = x_0$

ODEs

Initial value problem (IVP)

- $\dot{x}(t) = f(x(t))$
- $x(0) = x_0$

flow $\varphi(x_0, t)$ for solution of ODE depending on initial value

ODEs

Initial value problem (IVP)

- $\dot{x}(t) = f(x(t))$
- $x(0) = x_0$

flow $\varphi(x_0, t)$ for solution of ODE depending on initial value

- $\varphi(x_0, t + s) = \varphi(\varphi(x_0, t), s)$

ODEs

Initial value problem (IVP)

- $\dot{x}(t) = f(x(t))$
- $x(0) = x_0$

flow $\varphi(x_0, t)$ for solution of ODE depending on initial value

- $\varphi(x_0, t + s) = \varphi(\varphi(x_0, t), s)$
- $\varphi(x_0, 0) = x_0$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$
  $Q_h(X) = X_0 + [0; h] \cdot f(X)$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$
  $Q_h(X) = X_0 + [0; h] \cdot f(X)$
  post fixed point $Y \supseteq Q_h(Y)$

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$
  $Q_h(X) = X_0 + [0; h] \cdot f(X)$
  post fixed point $Y \supseteq Q_h(Y)$

implemented in *certify-stepsize*

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$
  $Q_h(X) = X_0 + [0; h] \cdot f(X)$
  post fixed point $Y \supseteq Q_h(Y)$

implemented in *certify-stepsize*

Theorem

If $x_0 \in X_0$ *and certify-stepsize*$(X_0) = (h, Y)$,

Certification of Existence

With Banach fixed point theorem:
unique solution $\varphi(x_0, t)$ exists on $[0; h]$, if:

- $P_h : \mathcal{C}([0; h], \mathbb{R}^n) \to \mathcal{C}([0; h], \mathbb{R}^n)$
  $P_h(\varphi) = (t \mapsto x_0 + \int_0^h f(\varphi(t))\mathrm{d}t)$
  is a contraction
- establishing for $x_0 \in X_0$, $f \in \mathcal{C}^1$:
  $Q_h : \mathcal{I}(\mathbb{R}^n) \to \mathcal{I}(\mathbb{R}^n)$
  $Q_h(X) = X_0 + [0; h] \cdot f(X)$
  post fixed point $Y \supseteq Q_h(Y)$

implemented in *certify-stepsize*

Theorem

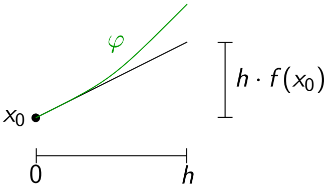
If $x_0 \in X_0$ and certify-stepsize$(X_0) = (h, Y)$,
then unique solution $\varphi(x_0, [0; h]) \subseteq Y$

Discretization: One-Step Methods

- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$

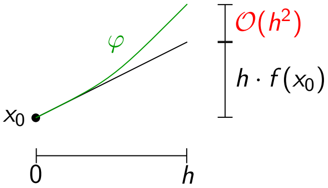
Discretization: One-Step Methods

- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0)$

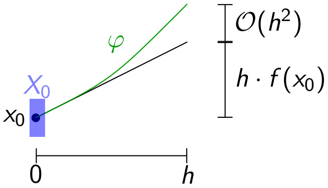


Discretization: One-Step Methods

- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$

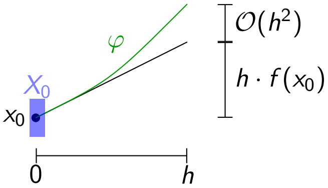


Discretization: One-Step Methods



- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$

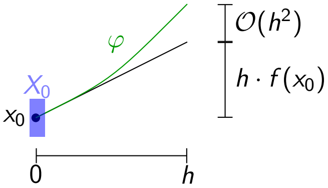
Discretization: One-Step Methods



- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$

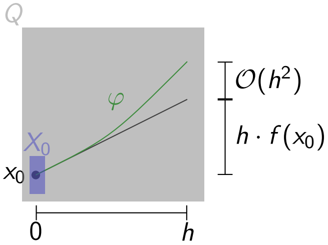
Discretization: One-Step Methods

- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$



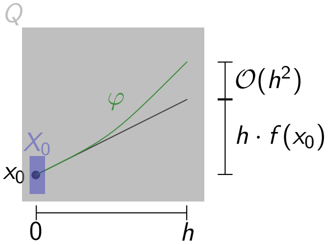
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$
  - $\mathrm{D}f$ enclosed by $\mathrm{D}F$

Discretization: One-Step Methods



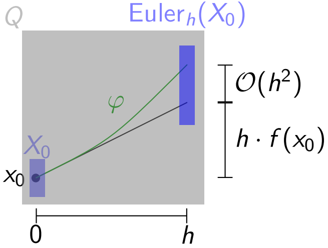
- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$
  - $\mathrm{D}f$ enclosed by $\mathrm{D}F$
  - $\varphi(X_0, [0; h]) \subseteq Q$

Discretization: One-Step Methods



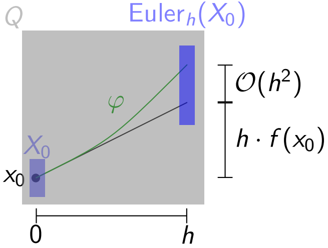
- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$
  - $\mathrm{D}f$ enclosed by $\mathrm{D}F$
  - $\varphi(X_0, [0; h]) \subseteq Q = \mathit{certify\text{-}stepsize}(X_0)$

Discretization: One-Step Methods



- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$
  - $\mathrm{D}f$ enclosed by $\mathrm{D}F$
  - $\varphi(X_0, [0; h]) \subseteq Q = \textit{certify-stepsize}(X_0)$
  - $\mathrm{Euler}_h(X_0) = X_0 + h \cdot F(X_0) + \frac{1}{2}h^2 \cdot \mathit{box}\,(\mathrm{D}F(Q)(F(Q)))$

Discretization: One-Step Methods

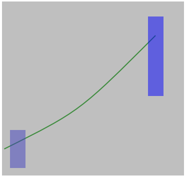


- "approximate" Euler step:
  - $\varphi(x_0, 0) = x_0$
  - $\varphi(x_0, h) \approx x_0 + h \cdot f(x_0) + \mathcal{O}(h^2)$
- set-based Euler step
  - $x_0 \in X_0$
  - f enclosed by F, i.e. $f(X) \subseteq F(X)$
  - $\mathrm{D}f$ enclosed by $\mathrm{D}F$
  - $\varphi(X_0, [0; h]) \subseteq Q = \textit{certify-stepsize}(X_0)$
  - $\mathrm{Euler}_h(X_0) = X_0 + h \cdot F(X_0) + \frac{1}{2} h^2 \cdot \textit{box}\left(\mathrm{D}F(Q)(F(Q))\right)$

Theorem

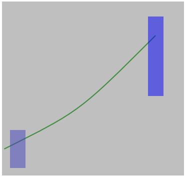
$\varphi(X_0, h) \subseteq \mathrm{Euler}_h(X_0)$

Discretization: One-Step Methods



Iterate:
$X_1 = \mathrm{Euler}_h(X_0)$

Discretization: One-Step Methods

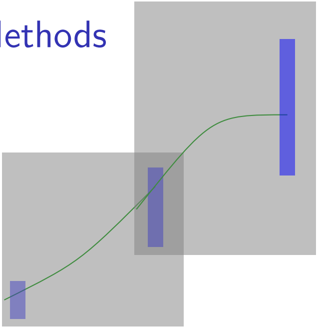


Iterate:

$X_1 = \mathsf{Euler}_h(X_0)$

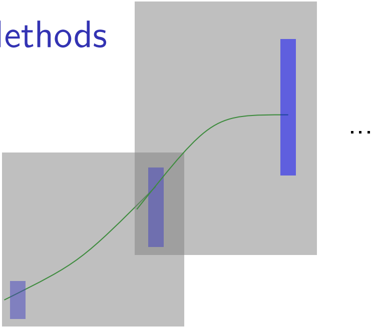
$\textit{certify-stepsize}(X_1) \rightsquigarrow \mathsf{Euler}_h(X_1)$

Discretization: One-Step Methods



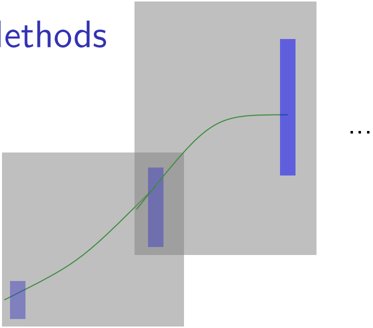
Iterate:
$X_1 = \mathrm{Euler}_h(X_0)$
certify-stepsize$(X_1) \rightsquigarrow \mathrm{Euler}_h(X_1)$

Discretization: One-Step Methods



Iterate:
$X_1 = \mathrm{Euler}_h(X_0)$
certify-stepsize$(X_1) \rightsquigarrow \mathrm{Euler}_h(X_1)$
...

Discretization: One-Step Methods



Iterate:

$X_1 = \mathrm{Euler}_h(X_0)$

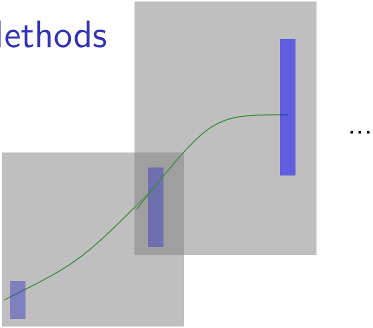
certify-stepsize$(X_1) \rightsquigarrow \mathrm{Euler}_h(X_1)$

...

optimizations:

- steps with error $\ll \mathcal{O}(h^2)$ (Runge-Kutta, TSE)

Discretization: One-Step Methods



Iterate:

$X_1 = \mathrm{Euler}_h(X_0)$

certify-stepsize$(X_1) \rightsquigarrow \mathrm{Euler}_h(X_1)$

...

optimizations:

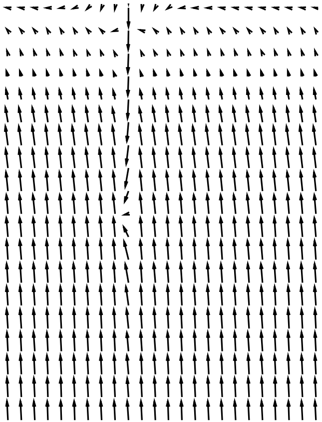
- steps with error $\ll \mathcal{O}(h^2)$ (Runge-Kutta, TSE)
- adaptive step size control

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML



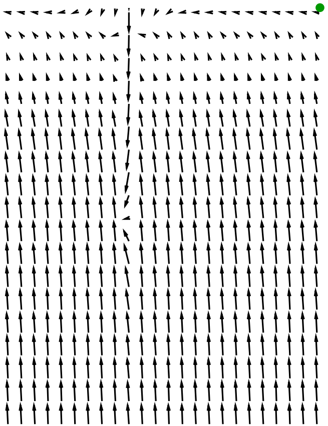
Oil reservoir problem

$y'(t) = z(t)$

$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML



Oil reservoir problem

$y'(t) = z(t)$

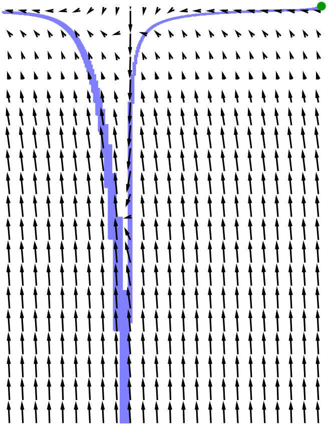
$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

$y(0) = 10; z(0) = 0$

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML

results:



Oil reservoir problem

$y'(t) = z(t)$

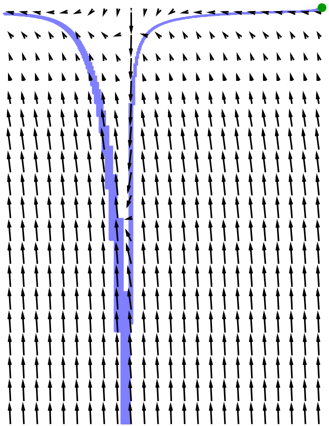
$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

$y(0) = 10; z(0) = 0$

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML

results:



Oil reservoir problem

$y'(t) = z(t)$

$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

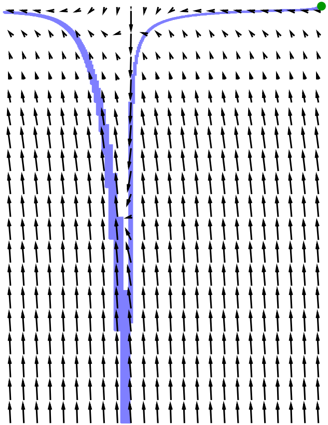
$y(0) = 10; z(0) = 0$

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML

results:

- efficient enough to compute non-trivial examples



Oil reservoir problem

$y'(t) = z(t)$

$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

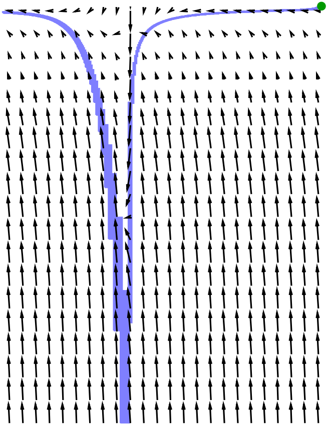
$y(0) = 10; z(0) = 0$

Experiments

Isabelle's code generator: translation of executable fragment of HOL to SML

results:

- efficient enough to compute non-trivial examples
- verified algorithm – rigorous proof for enclosures



Oil reservoir problem

$y'(t) = z(t)$

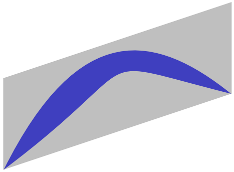
$z'(t) = z(t)^2 - \frac{1}{10^{-3}+y(t)^2}$

$y(0) = 10; z(0) = 0$

Further Optimizations

problem: affine arithmetic represents only convex sets

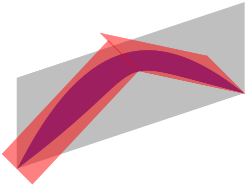
- splitting (fixed scale)



Further Optimizations

problem: affine arithmetic represents only convex sets

- splitting (fixed scale)

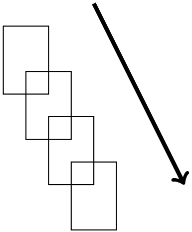


Further Optimizations

problem: affine arithmetic represents only convex sets

- splitting (fixed scale)

problem: splitting large sets / redundant computations



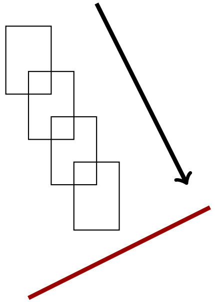
Further Optimizations

problem: affine arithmetic represents only convex sets

- splitting (fixed scale)

problem: splitting large sets / redundant computations

- reduction transversal to flow:



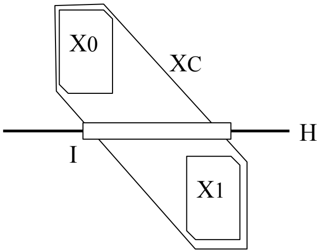
Further Optimizations

problem: affine arithmetic represents only convex sets

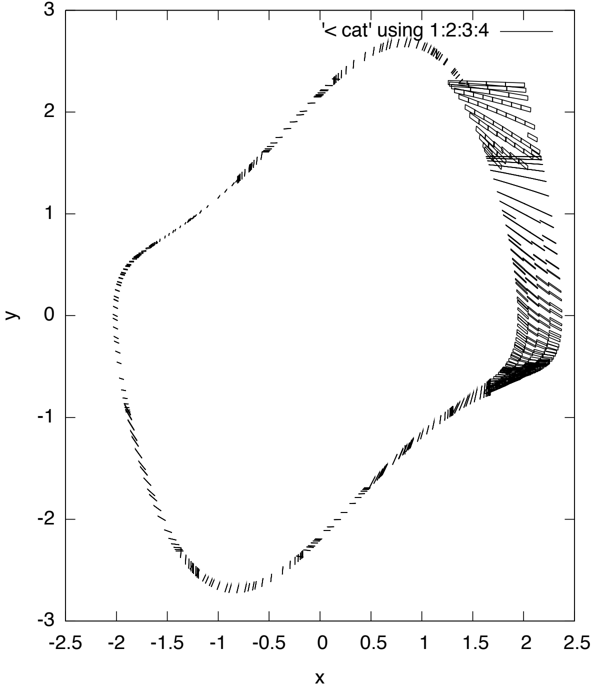
- splitting (fixed scale)

problem: splitting large sets / redundant computations

- reduction transversal to flow: geometric intersection of zonotopes with hyperplanes

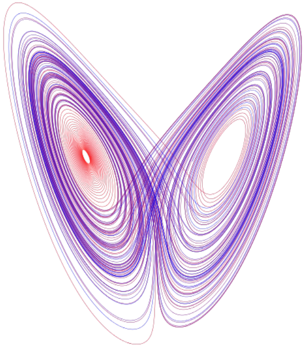


van der Pol:
$\dot{x} = y$
$\dot{y} = y(1 - x^2) - x$



Application: Existence of Lorenz attractor [Tucker 2002]

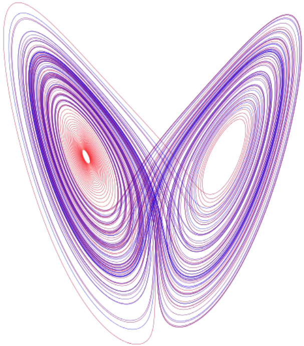
- Lorenz 1963: ODEs modeling a model for atmospheric flows



$$\dot{x} = 10(y - z)$$
$$\dot{y} = x(28 - z) - y$$
$$\dot{z} = xy - \frac{8}{3}z$$

Application: Existence of Lorenz attractor [Tucker 2002]

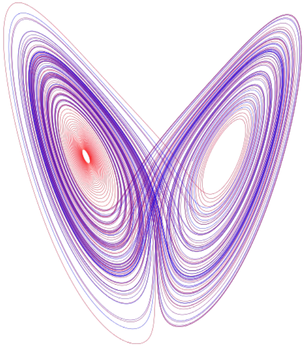
- Lorenz 1963: ODEs modeling a model for atmospheric flows
- numeric experiments: chaotic structure/strange attractor



$$\begin{aligned}\dot{x} &= 10(y - z)\\ \dot{y} &= x(28 - z) - y\\ \dot{z} &= xy - \tfrac{8}{3}z\end{aligned}$$

Application: Existence of Lorenz attractor [Tucker 2002]

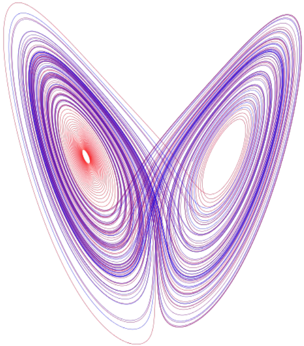
- Lorenz 1963: ODEs modeling a model for atmospheric flows
- numeric experiments: chaotic structure/strange attractor
- classic example of chaos



$$\dot{x} = 10(y - z)$$
$$\dot{y} = x(28 - z) - y$$
$$\dot{z} = xy - \frac{8}{3}z$$

Application: Existence of Lorenz attractor [Tucker 2002]

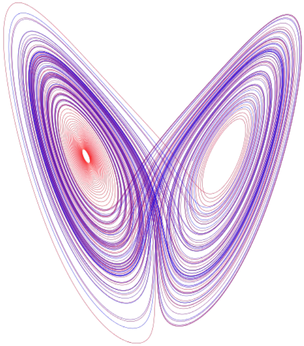
- Lorenz 1963: ODEs modeling a model for atmospheric flows
- numeric experiments: chaotic structure/strange attractor
- classic example of chaos
- Tucker 1999: proof combining guaranteed numerics and analysis



$$\begin{aligned}\dot{x} &= 10(y - z)\\ \dot{y} &= x(28 - z) - y\\ \dot{z} &= xy - \tfrac{8}{3}z\end{aligned}$$

Application: Existence of Lorenz attractor [Tucker 2002]

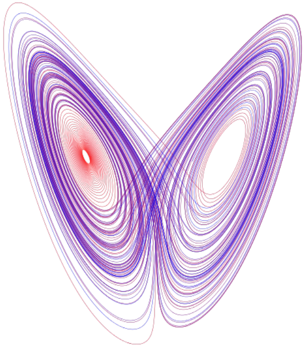
- Lorenz 1963: ODEs modeling a model for atmospheric flows
- numeric experiments: chaotic structure/strange attractor
- classic example of chaos
- Tucker 1999: proof combining guaranteed numerics and analysis
- unverified C++ program: some (mathematically nontrivial) bugs repaired since



$$\dot{x} = 10(y - z)$$
$$\dot{y} = x(28 - z) - y$$
$$\dot{z} = xy - \frac{8}{3}z$$

Application: Existence of Lorenz attractor [Tucker 2002]

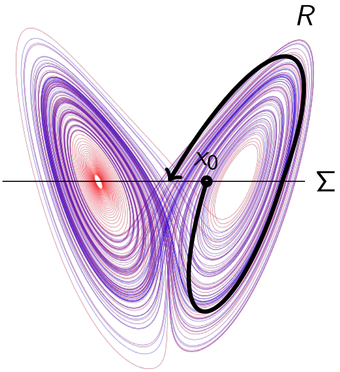
- Lorenz 1963: ODEs modeling a model for atmospheric flows
- numeric experiments: chaotic structure/strange attractor
- classic example of chaos
- Tucker 1999: proof combining guaranteed numerics and analysis
- unverified C++ program: some (mathematically nontrivial) bugs repaired since
- certify computations with Isabelle/HOL



$$\dot{x} = 10(y - z)$$
$$\dot{y} = x(28 - z) - y$$
$$\dot{z} = xy - \frac{8}{3}z$$

Computations for Tucker's Proof

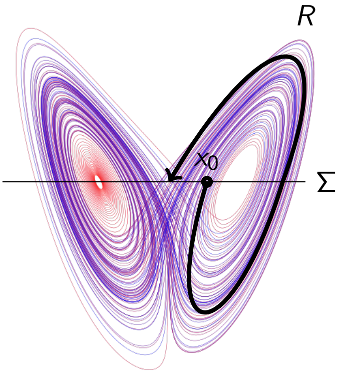
Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$



Computations for Tucker's Proof

Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$

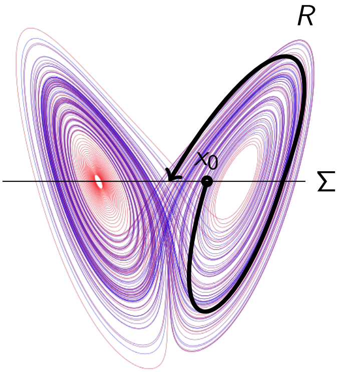
- numerical propagation of small initial Rectangles $N_i \subseteq \Sigma$



Computations for Tucker's Proof

Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$

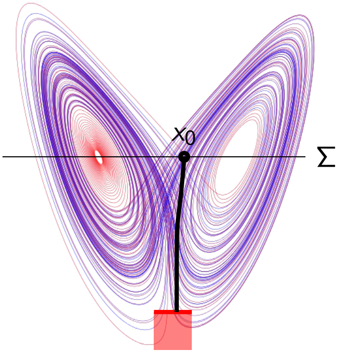
- numerical propagation of small initial Rectangles $N_i \subseteq \Sigma$
- iterate R until invariant set $R(N) \subseteq N \subseteq \Sigma$



Computations for Tucker's Proof

Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$

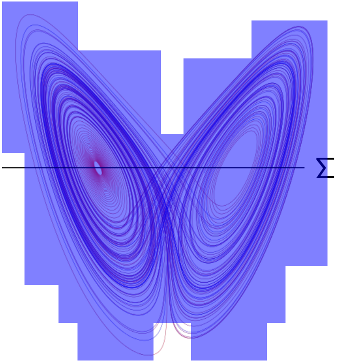
- numerical propagation of small initial Rectangles $N_i \subseteq \Sigma$
- iterate R until invariant set $R(N) \subseteq N \subseteq \Sigma$
- abort computation close to origin



Computations for Tucker's Proof

Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$

- numerical propagation of small initial Rectangles $N_i \subseteq \Sigma$
- iterate R until invariant set $R(N) \subseteq N \subseteq \Sigma$
- abort computation close to origin
- $\approx$ 100 CPU-days (embarrassingly parallel)

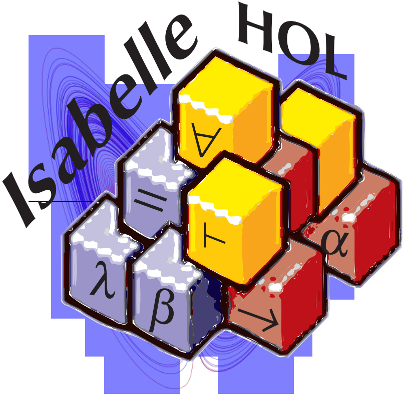


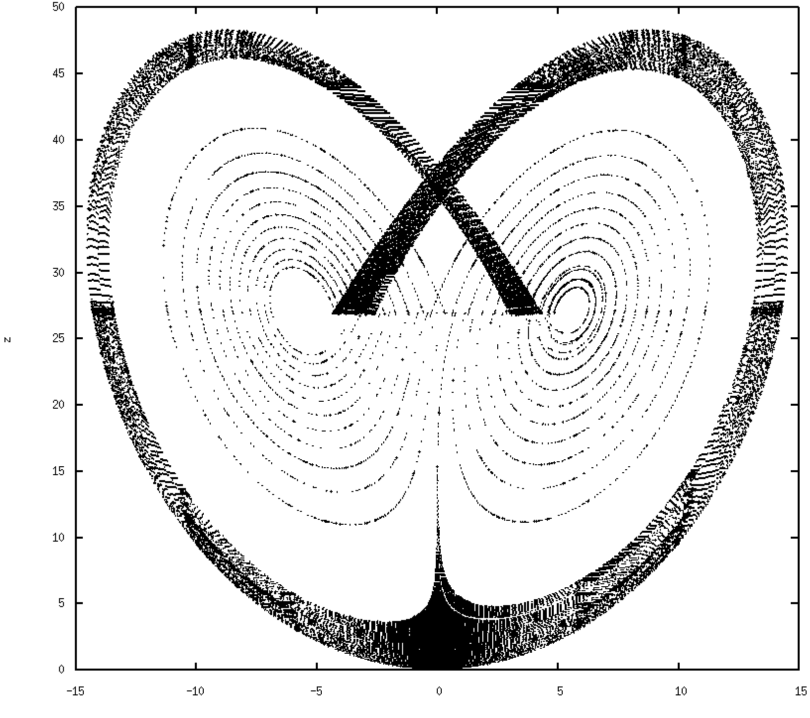
Computations for Tucker's Proof

Attractor studied via Poincaré map:
first return map R of dynamics on section $\Sigma \subseteq \mathbb{R}^2$

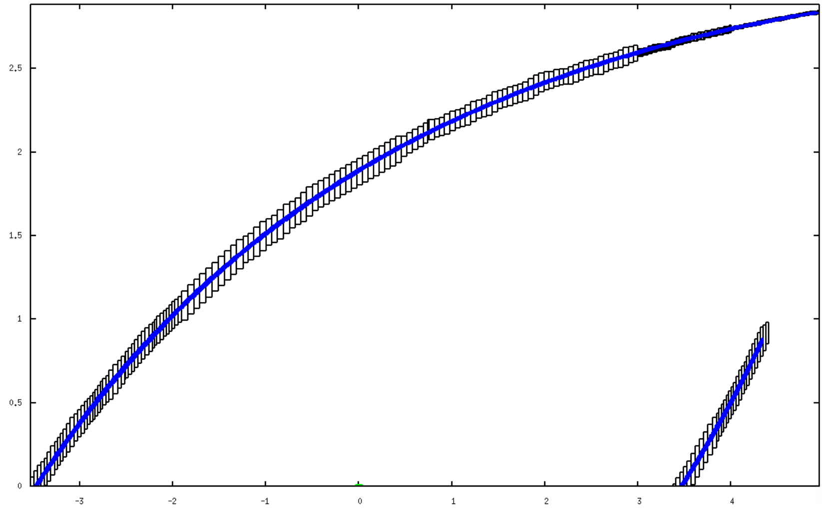
- numerical propagation of small initial Rectangles $N_i \subseteq \Sigma$
- iterate R until invariant set $R(N) \subseteq N \subseteq \Sigma$
- abort computation close to origin
- $\approx$ 100 CPU-days (embarrassingly parallel)

certified with




50
45
40
35
30
25
20
15
10
5
0
z
−15
−10
−5
0
5
10
15

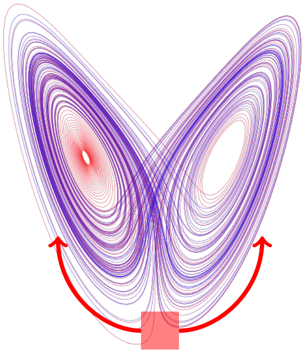
An Invariant Subset of the Return Plane Σ



Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

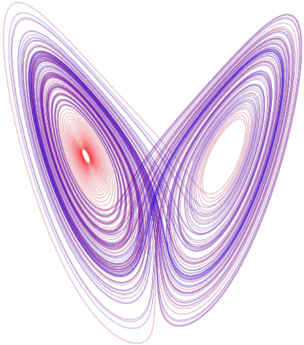
- symbolic propagation close to origin: few additional computations



Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

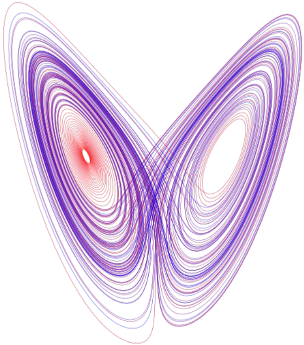
- symbolic propagation close to origin: few additional computations
- propagation of DR



Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

- symbolic propagation close to origin: few additional computations
- propagation of DR

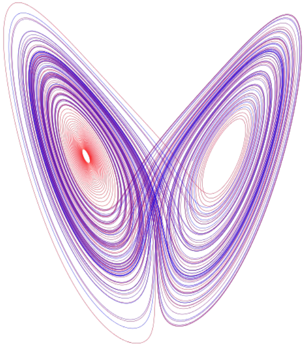


Is Tucker's proof then formalized?

Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

- symbolic propagation close to origin: few additional computations
- propagation of DR

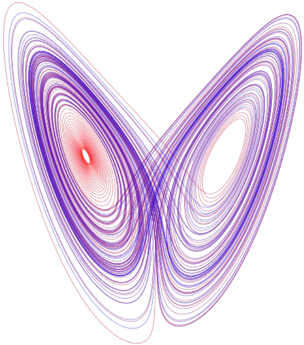


Is Tucker's proof then formalized? No:

Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

- symbolic propagation close to origin: few additional computations
- propagation of *DR*



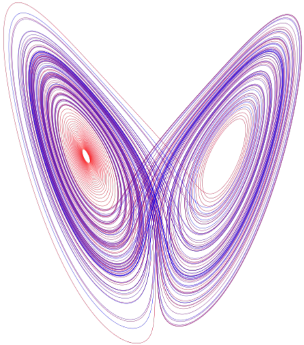
Is Tucker's proof then formalized? No:

- correctness of symbolic propagation

Missing Computations for Tucker's Proof

Current/future work:
establish further properties used in Tucker's proof

- symbolic propagation close to origin: few additional computations
- propagation of DR



Is Tucker's proof then formalized? No:

- correctness of symbolic propagation
- no formalization of chaos theory

Summary

formalization in Isabelle/HOL

- rigorous numerics and enclosures with affine arithmetic

Summary

formalization in Isabelle/HOL

- rigorous numerics and enclosures with affine arithmetic
- discretization of ODEs via one-step methods

Summary

formalization in Isabelle/HOL

- rigorous numerics and enclosures with affine arithmetic
- discretization of ODEs via one-step methods
- reduction/splitting for nonlinear dynamics

Summary

formalization in Isabelle/HOL

- rigorous numerics and enclosures with affine arithmetic
- discretization of ODEs via one-step methods
- reduction/splitting for nonlinear dynamics
- application: Lorenz attractor

Thank you for your attention

Formal Verification of ODE-Solvers (and an Application to the Lorenz Attractor)

Fabian Immler

Chair for Logic and Verification (Tobias Nipkow)
Institut für Informatik, Technische Universität München

SpecFun Seminar "Computations and Proofs"
2014-12-08