Continuous models of computation: computability, complexity, universality

Amaury Pouly

Joint work with Olivier Bournez and Daniel Graça

21 January 2019
What is a computer?
What is a computer?
What is a computer?
Church Thesis

All *reasonable* models of computation are equivalent.
Effective Church Thesis

All **reasonable** models of computation are equivalent for complexity.
Polynomial Differential Equations

\[
\begin{align*}
&k + k \\
&\times uv \\
&u + v \\
&\int u
\end{align*}
\]

General Purpose Analog Computer

Differential Analyzer

Newton mechanics

Reaction networks:
- chemical
- enzymatic

polynomial differential equations:
\[
\begin{cases}
y(0) = y_0 \\
y'(t) = p(y(t))
\end{cases}
\]

Rich class

Stable (+, ×, ⊙, ÷, ED)

No closed-form solution
Example of dynamical system

\[ \ddot{\theta} + \frac{g}{\ell} \sin(\theta) = 0 \]

Historical remark: the word "analog" The pendulum and the circuit have the same equation. One can study one using the other by analogy.
Example of dynamical system

\[ \ddot{\theta} + \frac{g}{\ell} \sin(\theta) = 0 \]

\begin{align*}
\begin{cases}
    y_1' &= y_2 \\
    y_2' &= -\frac{g}{\ell} y_3 \\
    y_3' &= y_2 y_4 \\
    y_4' &= -y_2 y_3
\end{cases}
\quad \iff \quad \\
\begin{cases}
    y_1 &= \theta \\
    y_2 &= \dot{\theta} \\
    y_3 &= \sin(\theta) \\
    y_4 &= \cos(\theta)
\end{cases}
\end{align*}

Historical remark: the word "analog"
Example of dynamical system

\[ \ddot{\theta} + \frac{g}{\ell} \sin(\theta) = 0 \]

\[
\begin{align*}
y'_1 &= y_2 \\
y'_2 &= -\frac{g}{\ell} y_3 \\
y'_3 &= y_2 y_4 \\
y'_4 &= -y_2 y_3 \\
\end{align*}
\]

\[
\begin{align*}
y_1 &= \theta \\
y_2 &= \dot{\theta} \\
y_3 &= \sin(\theta) \\
y_4 &= \cos(\theta) \\
\end{align*}
\]

Historical remark: the word "analog"
Example of dynamical system

\[ \ddot{\theta} + \frac{g}{\ell} \sin(\theta) = 0 \]

Historical remark: the word “analog”

The pendulum and the circuit have the same equation. One can study one using the other by analogy.
Computing with differential equations

Generable functions

\[
\begin{aligned}
\begin{cases}
y(0) &= y_0 \\
y'(x) &= p(y(x))
\end{cases} \\
x &\in \mathbb{R}
\end{aligned}
\]

\[f(x) = y_1(x)\]

Shannon’s notion

Strictly weaker than Turing machines [Shannon, 1941]

Modern notion

\[\sin, \cos, \exp, \log, \ldots\]

Turing powerful [Bournez et al., 2007]
Computing with differential equations

Generable functions

\[
\begin{align*}
\left\{ 
    y(0) &= y_0 \\
y'(x) &= p(y(x)) \\
x &\in \mathbb{R}
\right. \\
f(x) &= y_1(x)
\end{align*}
\]

Shannon’s notion

\[\text{sin, cos, exp, log, ...}\]

Strictly weaker than Turing machines [Shannon, 1941]
Computing with differential equations

Generable functions
\[
\begin{align*}
  y(0) &= y_0 \\
  y'(x) &= p(y(x)) \\
  x &\in \mathbb{R}
\end{align*}
\]
\[ f(x) = y_1(x) \]

Shannon’s notion
sin, cos, exp, log, ...

Strictly weaker than Turing machines [Shannon, 1941]

Computable
\[
\begin{align*}
  y(0) &= q(x) \\
  y'(t) &= p(y(t)) \\
  x &\in \mathbb{R} \\
  t &\in \mathbb{R}_+
\end{align*}
\]
\[ f(x) = \lim_{t \to \infty} y_1(t) \]

Modern notion
Generable functions

\[
\begin{align*}
\left\{ \begin{array}{l}
y(0) &= y_0 \\
y'(x) &= p(y(x)) \\
x &\in \mathbb{R}
\end{array} \right.
\]

\[f(x) = y_1(x)\]

Shannon's notion
sin, cos, exp, log, ...
Strictly weaker than Turing machines [Shannon, 1941]

Computable

\[
\begin{align*}
\left\{ \begin{array}{l}
y(0) &= q(x) \\
y'(t) &= p(y(t)) \\
x &\in \mathbb{R} \\
t &\in \mathbb{R}_+
\end{array} \right.
\]

\[f(x) = \lim_{t \to \infty} y_1(t)\]

Modern notion
sin, cos, exp, log, $\Gamma$, $\zeta$, ...
Turing powerful [Bournez et al., 2007]
Equivalence with computable analysis

Definition (Bournez et al, 2007)

\( f \) computable by GPAC if \( \exists p \) polynomial such that \( \forall x \in [a, b] \)

\[
y(0) = (x, 0, \ldots, 0) \quad y'(t) = p(y(t))
\]

satisfies \( |f(x) - y_1(t)| \leq y_2(t) \) et \( y_2(t) \xrightarrow{t \to \infty} 0 \).

Theorem (Bournez et al, 2007)

\( f : [a, b] \to \mathbb{R} \) computable \( \iff \) \( f \) computable by GPAC

1. In Computable Analysis, a standard model over reals built from Turing machines.
Equivalence with computable analysis

Definition (Bournez et al, 2007)

\( f \) computable by GPAC if \( \exists p \) polynomial such that \( \forall x \in [a, b] \)

\[ y(0) = (x, 0, \ldots, 0) \quad y'(t) = p(y(t)) \]

satisfies \( |f(x) - y_1(t)| \leq y_2(t) \) and \( y_2(t) \xrightarrow{t \to \infty} 0 \).

Theorem (Bournez et al, 2007)

\( f : [a, b] \to \mathbb{R} \) computable \( \iff \) \( f \) computable by GPAC

---

1. In Computable Analysis, a standard model over reals built from Turing machines.
**Equivalence with computable analysis**

**Definition (Bournez et al, 2007)**

A function $f$ is **computable by GPAC** if there exists a polynomial $p$ such that for all $x \in [a, b]$

$$y(0) = (x, 0, \ldots, 0) \quad y'(t) = p(y(t))$$

satisfies $|f(x) - y_1(t)| \leq y_2(t)$ and $y_2(t) \xrightarrow{t \to \infty} 0$. 

**Theorem (Bournez et al, 2007)**

$f : [a, b] \to \mathbb{R}$ is computable \(^1\) if and only if $f$ is computable by GPAC.

---

\(^1\) In Computable Analysis, a standard model over reals built from Turing machines.
Complexity of analog systems

- Turing machines: \(T(x)\) = number of steps to compute on \(x\)
Complexity of analog systems

- Turing machines: $T(x) =$ number of steps to compute on $x$
- GPAC:

**Tentative definition**

$T(x) =$ ??

$y(0) = (x, 0, \ldots, 0) \quad y' = p(y)$

Something is wrong...

All functions have constant time complexity.

$w(t) = y(e^e t)$

$x \mapsto x_1(t)$

$y \mapsto f(x)$

$t \mapsto t$
Complexity of analog systems

- Turing machines: \( T(x) = \) number of steps to compute on \( x \)
- GPAC:

Tentative definition

\[
T(x, \mu) =
\]

\[
y(0) = (x, 0, \ldots, 0) \quad y' = p(y)
\]

Something is wrong...

All functions have constant time complexity.
Complexity of analog systems

- Turing machines: \( T(x) = \) number of steps to compute on \( x \)
- GPAC:

**Tentative definition**

\[
T(x, \mu) = \text{first time } t \text{ so that } |y_1(t) - f(x)| \leq e^{-\mu}
\]

\[
y(0) = (x, 0, \ldots, 0) \quad y' = p(y)
\]

![Graph showing y_1(t), f(x), and y'(t)]
Complexity of analog systems

- Turing machines: \( T(x) \) = number of steps to compute on \( x \)
- GPAC:

Tentative definition

\[
T(x, \mu) = \text{first time } t \text{ so that } |y_1(t) - f(x)| \leq e^{-\mu}
\]

\[
y(0) = (x, 0, \ldots, 0) \quad y' = p(y)
\]

\[
z(t) = y(e^t)
\]
Complexity of analog systems

- Turing machines: \(T(x)\) = number of steps to compute on \(x\)
- GPAC:

**Tentative definition**

\[ T(x, \mu) = \text{first time } t \text{ so that } |y_1(t) - f(x)| \leq e^{-\mu} \]

\[ y(0) = (x, 0, \ldots, 0) \quad y' = p(y) \]

\[ z(t) = y(e^t) \]

\[ w(t) = y(e^{e^t}) \]
Complexity of analog systems

- Turing machines: $T(x) =$ number of steps to compute on $x$
- GPAC: time contraction problem $\rightarrow$ open problem

Tentative definition

$T(x, \mu) =$ first time $t$ so that $|y_1(t) - f(x)| \leq e^{-\mu}$

Something is wrong...

All functions have constant time complexity.
Time-space correlation of the GPAC

\[ y(0) = q(x) \quad y' = p(y) \]

\[ z(t) = y(e^t) \]

Observation: Time scaling costs “space”.
Time complexity for the GPAC must involve time and space!
Time-space correlation of the GPAC

\[ y(0) = q(x) \quad y' = p(y) \]

\[ z(t) = y(e^t) \]

extra component : \( w(t) = e^t \)
Time-space correlation of the GPAC

\[ y(0) = q(x) \quad y' = p(y) \]

\[ z(t) = y(e^t) \]

**Observation**

Time scaling costs "space".

```
\sim
```

Time complexity for the GPAC must involve time and space!
Definition: $\mathcal{L} \in \text{ANALOG-PTIME} \iff \exists p \text{ polynomial, } \forall \text{ word } w$

$y(0) = (\psi(w), |w|, 0, \ldots, 0) \quad y' = p(y) \quad \psi(w) = \sum_{i=1}^{\frac{|w|}{2}} w_i 2^{-i}$

$\ell(t) = \text{length of } y$
**Definition**: $L \in \text{ANALOG-PTIME} \iff \exists p \text{ polynomial, } \forall \text{ word } w$

$$y(0) = (\psi(w), |w|, 0, \ldots, 0) \quad y' = p(y) \quad \psi(w) = \sum_{i=1}^{||w||} w_i 2^{-i}$$

1. if $y_1(t) \geq 1$ then $w \in L$
Characterization of polynomial time

**Definition** : $\mathcal{L} \in \text{ANALOG-PTIME} \iff \exists p \text{ polynomial}, \forall \text{ word } w$

\[
y(0) = (\psi(w), |w|, 0, \ldots, 0) \quad y' = p(y) \quad \psi(w) = \sum_{i=1}^{|w|} w_i 2^{-i}
\]

satisfies

1. $y(0)$ starts at $0$ and $y'$ is never negative.
2. if $y_1(t) \leq -1$ then $w \notin \mathcal{L}$
Characterization of polynomial time

**Definition**: \( \mathcal{L} \in \text{ANALOG-PTIME} \iff \exists p \) polynomial, \( \forall \) word \( \mathbf{w} \)

\[
y(0) = (\psi(\mathbf{w}), |\mathbf{w}|, 0, \ldots, 0) \quad y' = p(y) \quad \psi(\mathbf{w}) = \sum_{i=1}^{|\mathbf{w}|} w_i 2^{-i}
\]

1. \( y(0) = (\psi(\mathbf{w}), |\mathbf{w}|, 0, \ldots, 0) \)
2. \( y' = p(y) \)
3. \( \psi(\mathbf{w}) = \sum_{i=1}^{|\mathbf{w}|} w_i 2^{-i} \)

- **accept**: \( \mathbf{w} \in \mathcal{L} \)
- **reject**: \( \mathbf{w} \notin \mathcal{L} \)

satisfies

3. if \( \ell(t) \geq \text{poly}(|\mathbf{w}|) \) then \( |y_1(t)| \geq 1 \)
Characterization of polynomial time

**Definition:** \( \mathcal{L} \in \text{ANALOG-PTIME} \iff \exists p \text{ polynomial, } \forall \text{ word } w \)

\[
y(0) = (\psi(w), |w|, 0, \ldots, 0) \quad y' = p(y) \quad \psi(w) = \sum_{i=1}^{|w|} w_i 2^{-i}
\]

- **accept:** \( w \in \mathcal{L} \)
- **reject:** \( w \notin \mathcal{L} \)

**Theorem**

\( \text{PTIME} = \text{ANALOG-PTIME} \)
Theorem

- \( \mathcal{L} \in \text{PTIME of and only if } \mathcal{L} \in \text{ANALOG-PTIME} \)
- \( f : [a, b] \rightarrow \mathbb{R} \) computable in polynomial time \( \iff f \in \text{ANALOG-P}_\mathbb{R} \)

- Analog complexity theory based on \textit{length}
- Time of Turing machine \( \iff \) length of the GPAC
- Purely continuous characterization of \text{PTIME}
Theorem

- $\mathcal{L} \in \text{PTIME of and only if } \mathcal{L} \in \text{ANALOG-PTIME}$
- $f : [a, b] \rightarrow \mathbb{R}$ computable in polynomial time $\iff f \in \text{ANALOG-P}_R$

- Analog complexity theory based on **length**
- Time of Turing machine $\iff$ length of the GPAC
- Purely continuous characterization of PTIME
- Only **rational coefficients** needed
In the remaining time...

Two applications of the techniques we have developed:

- Chemical Reaction Networks

Universal differential equation
Chemical Reaction Networks

Definition: a reaction system is a finite set of
- molecular species $y_1, \ldots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i$ (where $a_i, b_i \in \mathbb{N}$, $f$ = rate)

Example:

$$2H + O \rightarrow H_2O$$
$$C + O_2 \rightarrow CO_2$$
**Chemical Reaction Networks**

**Definition**: a reaction system is a finite set of

- molecular species $y_1, \ldots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \quad (a_i, b_i \in \mathbb{N}, f = \text{rate})$

**Example**:

\[
\begin{align*}
2\text{H} + \text{O} & \rightarrow \text{H}_2\text{O} \\
\text{C} + \text{O}_2 & \rightarrow \text{CO}_2
\end{align*}
\]

**Assumption**: law of mass action

\[
\sum_i a_i y_i \xrightarrow{k} \sum_i b_i y_i \quad \sim \quad f(y) = k \prod_i y_i^{a_i}
\]
Chemical Reaction Networks

**Definition**: a **reaction system** is a finite set of

- molecular species $y_1, \ldots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \quad (a_i, b_i \in \mathbb{N}, f = \text{rate})$

**Example**:

\[
\begin{align*}
2H + O & \rightarrow H_2O \\
C + O_2 & \rightarrow CO_2
\end{align*}
\]

**Assumption**: law of mass action

\[\sum_i a_i y_i \overset{k}{\rightarrow} \sum_i b_i y_i \leadsto f(y) = k \prod_i y_i^{a_i}\]

**Semantics**:

- discrete
- differential
- stochastic
Chemical Reaction Networks

**Definition**: a reaction system is a finite set of
- molecular species $y_1, \ldots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \quad (a_i, b_i \in \mathbb{N}, f = \text{rate})$

**Example**:

$2\text{H} + \text{O} \rightarrow \text{H}_2\text{O}$
$\text{C} + \text{O}_2 \rightarrow \text{CO}_2$

**Assumption**: law of mass action

$$\sum_i a_i y_i \xrightarrow{k} \sum_i b_i y_i \sim f(y) = k \prod_i y_i^{a_i}$$

**Semantics**:
- discrete
- differential
- stochastic

$$y_i' = \sum_{\text{reaction } R} (b_i^R - a_i^R) f^R(y)$$
Chemical Reaction Networks

**Definition**: a reaction system is a finite set of
- molecular species $y_1, \ldots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \quad (a_i, b_i \in \mathbb{N}, \ f = \text{rate})$

**Example**:

$$
\begin{align*}
2H + O & \rightarrow H_2O \\
C + O_2 & \rightarrow CO_2
\end{align*}
$$

**Assumption**: law of mass action

$$
\sum_i a_i y_i \xrightarrow{k} \sum_i b_i y_i \quad \sim \quad f(y) = k \prod_i y_i^{a_i}
$$

**Semantics**:
- discrete
- differential
- stochastic

$$
y_i' = \sum_{\text{reaction } R} (b_i^R - a_i^R)k^R \prod_j y_j^{a_j}
$$
Chemical Reaction Networks (CRNs)

- CRNs with differential semantics and mass action law = polynomial ODEs
- Polynomial ODEs are Turing complete

Two "slight" problems:
- Concentrations cannot be negative ($y_i < 0$)
- Arbitrary reactions are not realistic

Definition: a reaction is elementary if it has at most two reactants.

Elementary reactions correspond to quadratic ODEs:

\[ ay + bz \rightarrow \cdots \]

\[ f(y, z) = ky^a z^b \]

Theorem (Folklore): Every polynomial ODE can be rewritten as a quadratic ODE.
Chemical Reaction Networks (CRNs)

- CRNs with differential semantics and mass action law = polynomial ODEs
- Polynomial ODEs are Turing complete

CRNs are Turing complete?

Definition: A reaction is elementary if it has at most two reactants. Elementary reactions correspond to quadratic ODEs:

\[ ay + bz \rightarrow \cdots \]

where \( f(y, z) = ky^a z^b \).

Theorem (Folklore): Every polynomial ODE can be rewritten as a quadratic ODE.
CRNs are Turing complete? Two “slight” problems:

- concentrations cannot be negative ($y_i < 0$)
- arbitrary reactions are not realistic
Chemical Reaction Networks (CRNs)

CRNs are Turing complete? Two “slight” problems:

- Concentrations cannot be negative ($y_i < 0$)
- Arbitrary reactions are not realistic

Easy to solve

What is realistic?
Chemical Reaction Networks (CRNs)

CRNs are Turing complete? Two “slight” problems:

- concentrations cannot be negative ($y_i < 0$)
- arbitrary reactions are not realistic

▶ easy to solve
▶ what is realistic?

Definition: a reaction is elementary if it has at most two reactants
⇒ can be implemented with DNA, RNA or proteins
Chemical Reaction Networks (CRNs)

CRNs are Turing complete? Two “slight” problems:

▶ concentrations cannot be negative ($y_i < 0$) ➤ easy to solve
▶ arbitrary reactions are not realistic ➤ what is realistic?

Definition: a reaction is elementary if it has at most two reactants
⇒ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to quadratic ODEs:

$$ay + bz \xrightarrow{k} \ldots \sim f(y, z) = ky^a z^b$$
Chemical Reaction Networks (CRNs)

CRNs are Turing complete? Two “slight” problems:

▶ concentrations cannot be negative ($y_i < 0$)  ► easy to solve
▶ arbitrary reactions are not realistic  ► what is realistic?

Definition: a reaction is elementary if it has at most two reactants
⇒ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to quadratic ODEs:

$$ay + bz \xrightarrow{k} \cdots \sim f(y, z) = ky^a z^b$$

Theorem (Folklore)

Every polynomial ODE can be rewritten as a quadratic ODE.
Chemical Reaction Networks (CRNs)

Definition: a reaction is **elementary** if it has at most two reactants
⇒ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to **quadratic** ODEs:

\[ ay + bz \xrightarrow{k} \ldots \sim f(y, z) = ky^a z^b \]

Theorem (Work with François Fages, Guillaume Le Guludec)

*Elementary mass-action-law reaction system on finite universes of molecules are Turing-complete under the differential semantics.*

Notes:

- proof preserves polynomial length
- in fact the following elementary reactions suffice:

\[
\emptyset \xrightarrow{k} x \quad x \xrightarrow{k} x + z \quad x + y \xrightarrow{k} x + y + z \quad x + y \xrightarrow{k} \emptyset
\]
In the remaining time...

Two applications of the techniques we have developed:

- Chemical Reaction Networks

~ Universal differential equation
Universal differential equations

Generable functions

subclass of analytic functions

Computable functions

any computable function
Universal differential equations

Generable functions

subclass of analytic functions

 Computable functions

any computable function
Theorem (Rubel, 1981)

For any continuous functions $f$ and $\varepsilon$, there exists $y : \mathbb{R} \to \mathbb{R}$ solution to

$$3y'^4 y'' y''''^2 - 4y'^4 y''''^2 y'''' + 6y'^3 y''^2 y''' y'''' + 24y'^2 y''^4 y''''' - 12y'^3 y'' y''''^3 - 29y'^2 y^3 y'''^2 + 12y''^7 = 0$$

such that $\forall t \in \mathbb{R}$,

$$|y(t) - f(t)| \leq \varepsilon(t).$$
Theorem (Rubel, 1981)

There exists a **fixed** polynomial $p$ and $k \in \mathbb{N}$ such that for any continuous functions $f$ and $\varepsilon$, there exists a solution $y : \mathbb{R} \to \mathbb{R}$ to

$p(y, y', \ldots, y^{(k)}) = 0$

such that $\forall t \in \mathbb{R}$,

$|y(t) - f(t)| \leq \varepsilon(t)$. 

This is a "weak" result.
Universal differential algebraic equation (DAE)

Theorem (Rubel, 1981)

There exists a **fixed** polynomial $p$ and $k \in \mathbb{N}$ such that for any continuous functions $f$ and $\varepsilon$, there exists a solution $y : \mathbb{R} \rightarrow \mathbb{R}$ to

$$p(y, y', \ldots, y^{(k)}) = 0$$

such that $\forall t \in \mathbb{R}$,

$$|y(t) - f(t)| \leq \varepsilon(t).$$

Problem: this is «weak» result.
The solution $y$ is not unique, **even with added initial conditions** :

$$p(y, y', \ldots, y^{(k)}) = 0, \quad y(0) = \alpha_0, \ y'(0) = \alpha_1, \ldots, \ y^{(k)}(0) = \alpha_k$$

In fact, this is fundamental for Rubel’s proof to work!
The problem with Rubel’s DAE

The solution $y$ is not unique, **even with added initial conditions** :

\[ p(y, y', \ldots, y^{(k)}) = 0, \quad y(0) = \alpha_0, \ y'(0) = \alpha_1, \ldots, \ y^{(k)}(0) = \alpha_k \]

In fact, this is fundamental for Rubel’s proof to work!

- Rubel’s statement: this DAE is universal
- More realistic interpretation: this DAE allows almost anything

Open Problem (Rubel, 1981)

Is there a universal ODE $y' = p(y)$?

**Note**: explicit polynomial ODE $\Rightarrow$ unique solution
Rubel’s proof in one slide

Take \( f(t) = e^{\frac{-1}{1-t^2}} \) for \(-1 < t < 1\) and \( f(t) = 0 \) otherwise.

It satisfies \((1 - t^2)^2 f''(t) + 2tf'(t) = 0\).
Rubel’s proof in one slide

- Take $f(t) = e^{\frac{-1}{1-t^2}}$ for $-1 < t < 1$ and $f(t) = 0$ otherwise.
  - It satisfies $(1 - t^2)^2 f''(t) + 2tf'(t) = 0$.

- For any $a, b, c \in \mathbb{R}$, $y(t) = cf(at + b)$ satisfies
  \[3y'^4y''y''''^2 - 4y'^4y''^2y'''' + 6y'^3y''^2y'''y'''' + 24y'^2y''^4y'''' - 12y'^3y''y'''^3 - 29y'^2y'''^3y'''' + 12y''''^7 = 0\]

Translation and rescaling:
Take \( f(t) = e^{\frac{-1}{1-t^2}} \) for \(-1 < t < 1\) and \( f(t) = 0 \) otherwise. It satisfies \((1 - t^2)^2 f''(t) + 2tf'(t) = 0\).

For any \( a, b, c \in \mathbb{R} \), \( y(t) = cf(at + b) \) satisfies

\[
3y^4 y''^3 y'''^2 - 4y^4 y''^2 y'''^3 + 6y'^3 y''^2 y'''' + 24y'^2 y'''^4 y''' - 12y'^3 y'' y''''^2 - 29y'^2 y'''^3 y''''^2 + 12y''''^7 = 0
\]

Can glue together arbitrary many such pieces.
Rubel’s proof in one slide

- Take \( f(t) = e^{\frac{-1}{1-t^2}} \) for \(-1 < t < 1\) and \( f(t) = 0 \) otherwise. It satisfies \( (1 - t^2)^2 f''(t) + 2tf'(t) = 0 \).

- For any \( a, b, c \in \mathbb{R} \), \( y(t) = cf(at + b) \) satisfies

\[
3y^4 y'' y'''^2 - 4y^4 y'' y''' + 6y^3 y''^2 y''' + 24y^2 y'' y''' y''' - 12y^3 y'' y'''^2 - 29y^2 y''' y''^2 + 12y''^7 = 0
\]

- Can glue together arbitrary many such pieces

- Can arrange so that \( \int f \) is solution: **piecewise pseudo-linear**
Rubel’s proof in one slide

- Take $f(t) = e^{\frac{-1}{1-t^2}}$ for $-1 < t < 1$ and $f(t) = 0$ otherwise. It satisfies $(1 - t^2)^2 f''(t) + 2tf'(t) = 0$.
- For any $a, b, c \in \mathbb{R}$, $y(t) = cf(at + b)$ satisfies
  $$3y''^4 y'''^2 - 4y''^4 y'''^2 y'''' + 6y''^3 y''^2 y''' y'''' + 24y''^2 y''^4 y''' - 12y''^3 y'' y'''^3 - 29y''^2 y''^3 y'''^2 + 12y''^7 = 0$$
- Can glue together arbitrary many such pieces
- Can arrange so that $\int f$ is solution: piecewise pseudo-linear

Conclusion: Rubel’s equation allows any piecewise pseudo-linear functions, and those are dense in $C^0$
Universal initial value problem (IVP)

Theorem

There exists a **fixed** (vector of) polynomial $p$ such that for any continuous functions $f$ and $\varepsilon$, there exists $\alpha \in \mathbb{R}^d$ such that

$$y(0) = \alpha, \quad y'(t) = p(y(t))$$

has a **unique solution** $y : \mathbb{R} \to \mathbb{R}^d$ and $\forall t \in \mathbb{R}$,

$$|y_1(t) - f(t)| \leq \varepsilon(t).$$
Universal initial value problem (IVP)

Notes:
- system of ODEs,
- $y$ is analytic,
- we need $d \approx 300$.

Theorem

There exists a **fixed** (vector of) polynomial $p$ such that for any continuous functions $f$ and $\varepsilon$, there exists $\alpha \in \mathbb{R}^d$ such that

$$y(0) = \alpha, \quad y'(t) = p(y(t))$$

has a **unique solution** $y : \mathbb{R} \to \mathbb{R}^d$ and $\forall t \in \mathbb{R}$,

$$|y_1(t) - f(t)| \leq \varepsilon(t).$$
Universal initial value problem (IVP)

Notes:
- system of ODEs,
- $y$ is analytic,
- we need $d \approx 300$.

Theorem

There exists a fixed (vector of) polynomial $p$ such that for any continuous functions $f$ and $\varepsilon$, there exists $\alpha \in \mathbb{R}^d$ such that

$$y(0) = \alpha, \quad y'(t) = p(y(t))$$

has a unique solution $y : \mathbb{R} \rightarrow \mathbb{R}^d$ and $\forall t \in \mathbb{R}$,

$$|y_1(t) - f(t)| \leq \varepsilon(t).$$

Remark: $\alpha$ is usually transcendental, but computable from $f$ and $\varepsilon$.
Future work

Reaction networks:
- chemical
- enzymatic

\[ y' = p(y) \]

\[ y' = p(y) + e(t) \]

- Finer time complexity (linear)
- Nondeterminism
- Robustness
- « Space» complexity
- Other models
- Stochastic
Backup slides
Complexity of solving polynomial ODEs

\[ y(0) = x \quad y'(t) = p(y(t)) \]
Complexity of solving polynomial ODEs

\[ y(0) = x \quad y'(t) = p(y(t)) \]

**Theorem**

If \( y(t) \) exists, one can compute \( p, q \) such that
\[
\left| \frac{p}{q} - y(t) \right| \leq 2^{-n} \text{ in time poly (size of } x \text{ and } p, n, \ell(t))
\]

where \( \ell(t) \approx \text{length of the curve (between } x \text{ and } y(t)) \)

length of the curve = complexity = ressource
Characterization of real polynomial time

**Definition**: \( f : [a, b] \rightarrow \mathbb{R} \) in \( \text{ANALOG-P}_\mathbb{R} \iff \exists p \) polynomial, \( \forall x \in [a, b] \)

\[
y(0) = (x, 0, \ldots, 0) \quad y' = p(y)
\]
Characterization of real polynomial time

**Definition**: \( f : [a, b] \to \mathbb{R} \) in \( \text{ANALOG-P}_\mathbb{R} \iff \exists p \text{ polynomial}, \forall x \in [a, b] \)

\[
y(0) = (x, 0, \ldots, 0) \quad y' = p(y)
\]
satisfies:

1. \(|y_1(t) - f(x)| \leq 2^{-\ell(t)}

   «greater length \( \Rightarrow \) greater precision»

2. \(\ell(t) \geq t\)

   «length increases with time»
Characterization of real polynomial time

**Definition**: \( f : [a, b] \rightarrow \mathbb{R} \) in \( \text{ANALOG-}P_{\mathbb{R}} \iff \exists p \) polynomial, \( \forall x \in [a, b] \)

\[ y(0) = (x, 0, \ldots, 0) \quad y' = p(y) \]

satisfies:

1. \( |y_1(t) - f(x)| \leq 2^{-\ell(t)} \)
   «greater length \( \Rightarrow \) greater precision»

2. \( \ell(t) \geq t \)
   «length increases with time»

**Theorem**

\( f : [a, b] \rightarrow \mathbb{R} \) computable in polynomial time \( \iff f \in \text{ANALOG-}P_{\mathbb{R}} \).
Theorem

There exists a **fixed** polynomial \( p \) and \( k \in \mathbb{N} \) such that for any continuous functions \( f \) and \( \varepsilon \), there exists \( \alpha_0, \ldots, \alpha_k \in \mathbb{R} \) such that

\[
p(y, y', \ldots, y^{(k)}) = 0, \quad y(0) = \alpha_0, \quad y'(0) = \alpha_1, \ldots, \quad y^{(k)}(0) = \alpha_k
\]

has a **unique analytic solution** and this solution satisfies such that

\[
|y(t) - f(t)| \leq \varepsilon(t).
\]