Continuous models of computation: computability, complexity, universality

Amaury Pouly

Université de Paris, IRIF, CNRS

27 january 2020

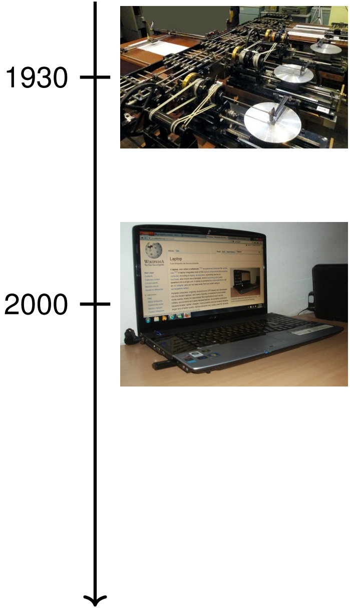
Analog computers : the come back !

1930

analog computers :

- hard to program
- highly specialized

Analog computers : the come back !



analog computers :

- hard to program
- highly specialized

digital computers :

- « easy » to program
- general purpose

Analog computers : the come back !

1930

2000

analog computers :

- hard to program
- highly specialized
- obsolete ?

digital computers :

- « easy » to program
- general purpose

Analog computers : the come back !

1930

analog computers :

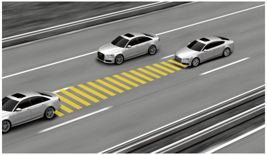
- hard to program
- highly specialized
- obsolete ?

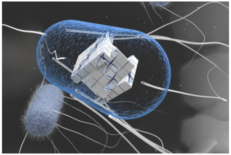
2000

digital computers :

- « easy » to program
- general purpose

2030 ?





- analog ?
- digital ?

1930

analog computers :

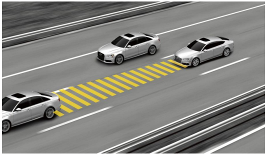
- hard to program
- highly specialized
- obsolete ?

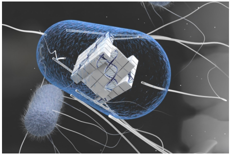
2000

digital computers :

- « easy » to program
- general purpose

2030 ?

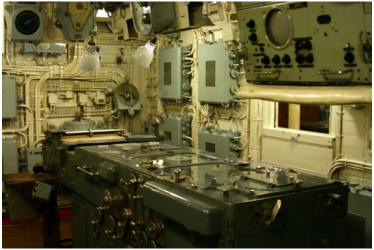




- analog ?
- digital ?
- both!

Analog computers

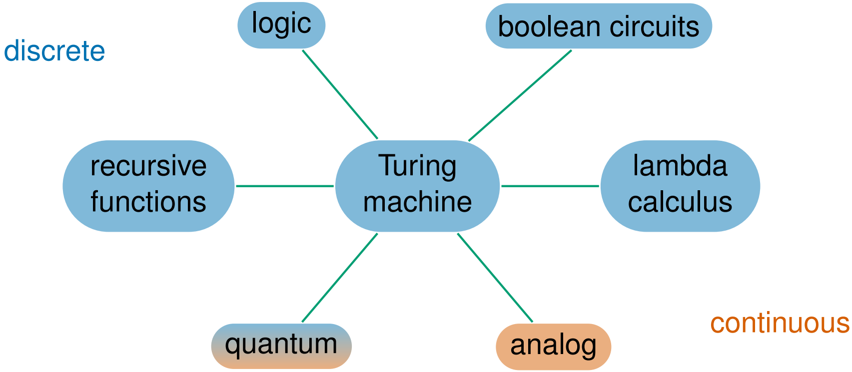
Differential Analyser
“Mathematica of 1920”



Admiralty Fire Control Table
British Navy (WW2)

Church Thesis

Computability

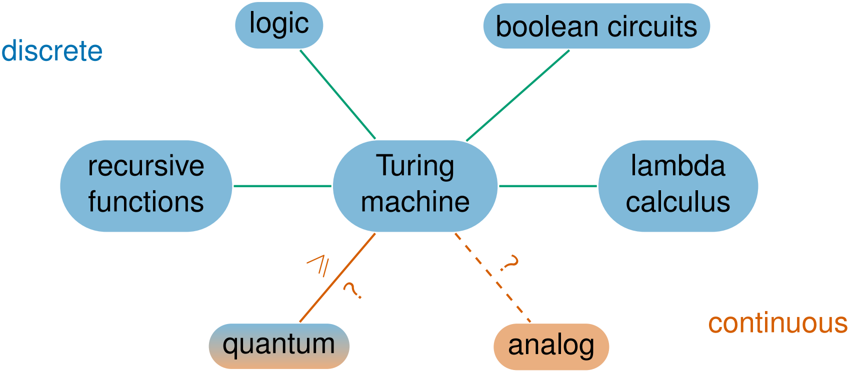


Church Thesis

All **reasonable** models of computation are equivalent.

Church Thesis

Complexity



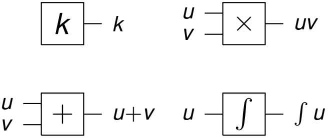
Effective Church Thesis

All **reasonable** models of computation are equivalent for complexity.

From machines to models

Differential analyzer

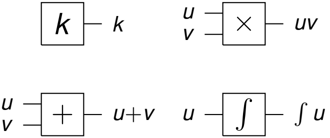
From machines to models



General Purpose Analog Computer, Shannon 1936

Differential analyzer

From machines to models

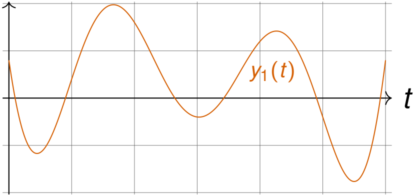


General Purpose Analog Computer, Shannon 1936

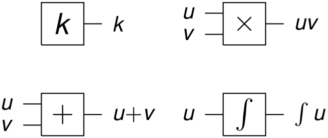
Differential analyzer

$$y(0) = y_0, \quad y'(t) = p(y(t))$$

Polynomial Differential Equation, Graça 2004



From machines to models

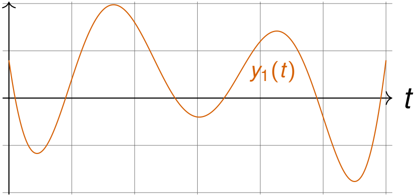


General Purpose Analog Computer, Shannon 1936

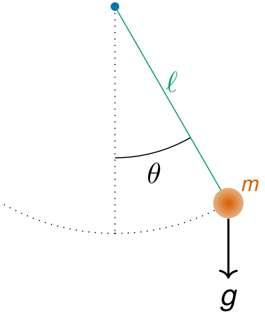
Differential analyzer

$$y(0) = y_0, \quad y'(t) = p(y(t))$$

Polynomial Differential Equation, Graça 2004

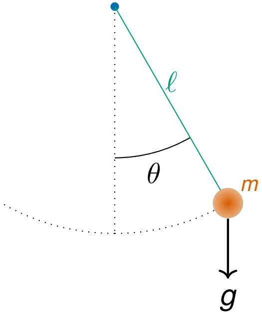


Example of dynamical system



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

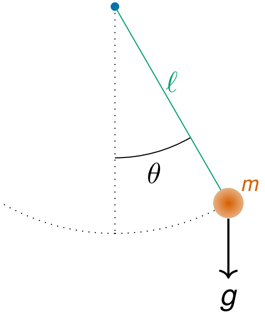
Example of dynamical system

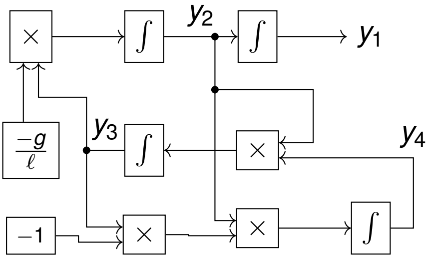


$$\ddot{\theta} + \frac{g}{\ell}\sin(\theta) = 0$$

$$\begin{cases} y_1' = y_2 \\ y_2' = -\frac{g}{l} y_3 \\ y_3' = y_2 y_4 \\ y_4' = -y_2 y_3 \end{cases} \Leftrightarrow \begin{cases} y_1 = \theta \\ y_2 = \dot{\theta} \\ y_3 = \sin(\theta) \\ y_4 = \cos(\theta) \end{cases}$$

Example of dynamical system

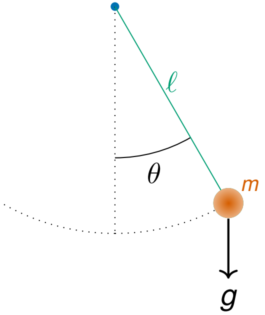


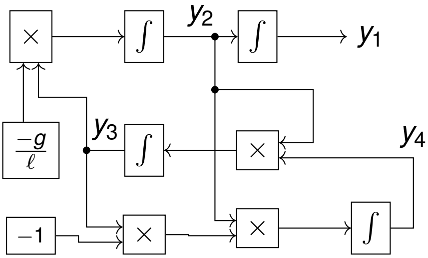


$$\ddot{\theta} + \frac{g}{\ell}\sin(\theta) = 0$$

$$\begin{cases} y_1' = y_2 \\ y_2' = -\frac{g}{l} y_3 \\ y_3' = y_2 y_4 \\ y_4' = -y_2 y_3 \end{cases} \Leftrightarrow \begin{cases} y_1 = \theta \\ y_2 = \dot{\theta} \\ y_3 = \sin(\theta) \\ y_4 = \cos(\theta) \end{cases}$$

Example of dynamical system





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

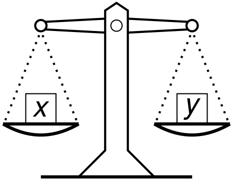
$$\begin{cases} y_1' = y_2 \\ y_2' = -\frac{g}{l} y_3 \\ y_3' = y_2 y_4 \\ y_4' = -y_2 y_3 \end{cases} \Leftrightarrow \begin{cases} y_1 = \theta \\ y_2 = \dot{\theta} \\ y_3 = \sin(\theta) \\ y_4 = \cos(\theta) \end{cases}$$

Historical remark : the word “analog”

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

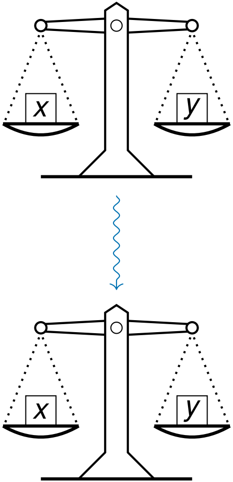
Does a balance scale compute a function ?

Inputs : $x, y \in [0, +\infty)$



Does a balance scale compute a function ?

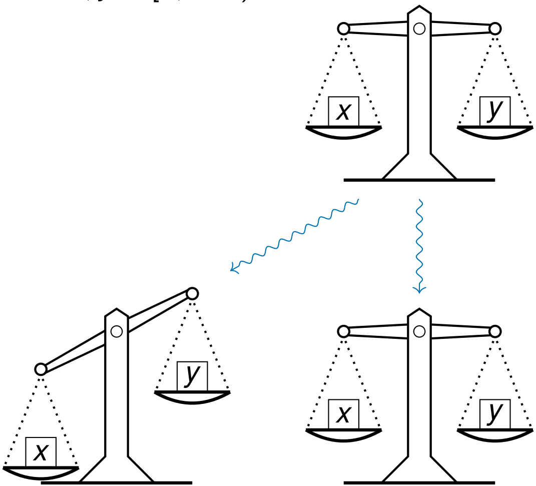
Inputs : $x, y \in [0, +\infty)$



$x = y$

Does a balance scale compute a function ?

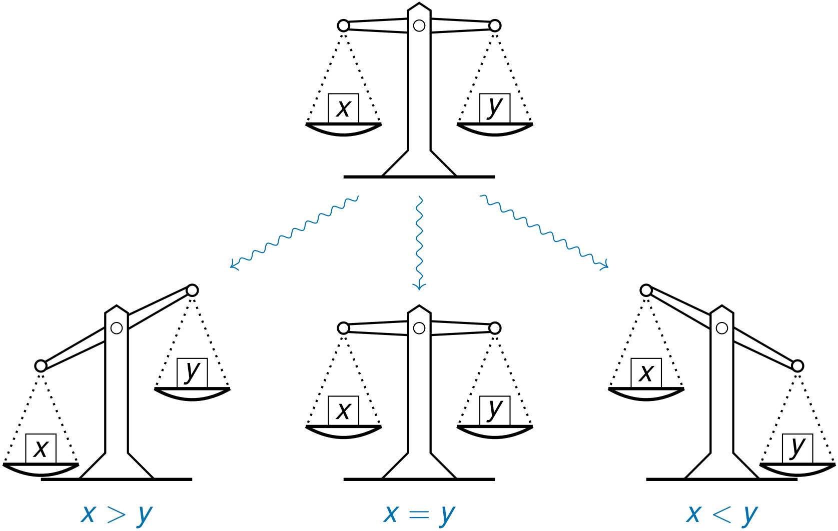
Inputs : $x, y \in [0, +\infty)$



$x > y$ $x = y$

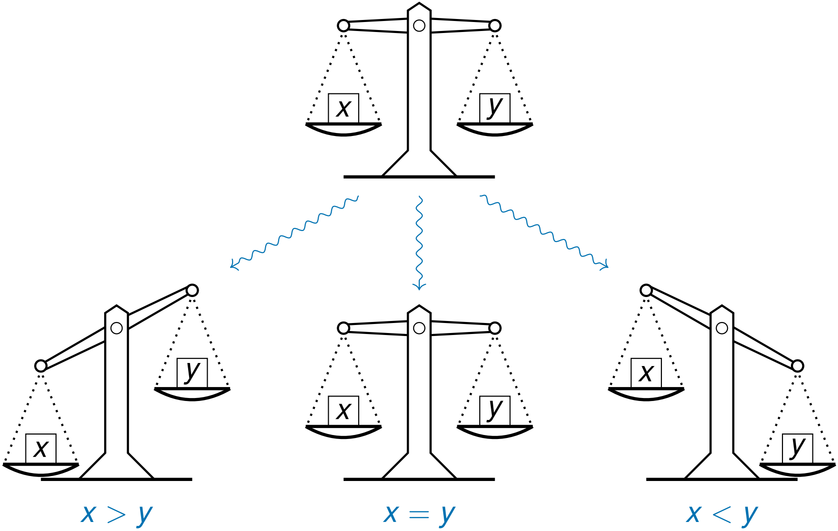
Does a balance scale compute a function ?

Inputs : $x, y \in [0, +\infty)$



Does a balance scale compute a function ?

Inputs : $x, y \in [0, +\infty)$



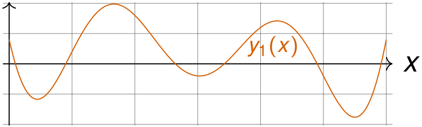
Output : $\mathrm{sign}(x - y)$?

Computing with differential equations

Generable functions

$$\begin{cases} y(0) = y_0 \\ y'(x) = p(y(x)) \end{cases} \qquad x \in \mathbb{R}$$

$$f(x) = y_1(x)$$



Shannon's notion

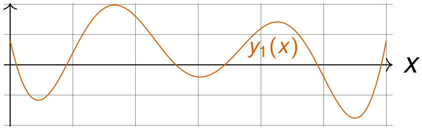
sin, cos, exp, log, ...

Computing with differential equations

Generable functions

$$\begin{cases} y(0) = y_0 \\ y'(x) = p(y(x)) \end{cases} \qquad x \in \mathbb{R}$$

$$f(x) = y_1(x)$$



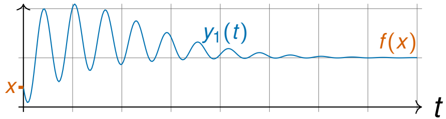
Shannon's notion

sin, cos, exp, log, ...

Computable

$$\begin{cases} y(0) = q(x) & x \in \mathbb{R} \\ y'(t) = p(y(t)) & t \in \mathbb{R}_+ \end{cases}$$

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



Modern notion

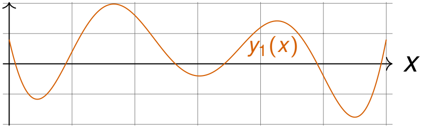
sin, cos, exp, log, Γ, ζ, ...

Computing with differential equations

Generable functions

$$\begin{cases} y(0) = y_0 \\ y'(x) = p(y(x)) \end{cases} \qquad x \in \mathbb{R}$$

$$f(x) = y_1(x)$$



Shannon's notion

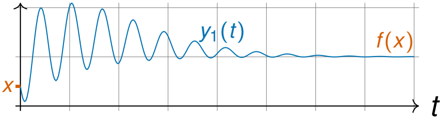
$\sin, \cos, \exp, \log, \ldots$

Considered "weak" : not Γ and ζ
Only analytic functions

Computable

$$\begin{cases} y(0) = q(x) & x \in \mathbb{R} \\ y'(t) = p(y(t)) & t \in \mathbb{R}_+ \end{cases}$$

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



Modern notion

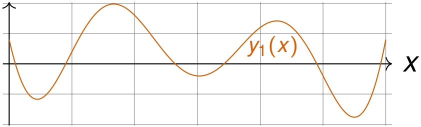
$\sin, \cos, \exp, \log, \Gamma, \zeta, \ldots$

Computing with differential equations

Generable functions

$$\begin{cases} y(0) = y_0 \\ y'(x) = p(y(x)) \end{cases} \qquad x \in \mathbb{R}$$

$$f(x) = y_1(x)$$



Shannon's notion

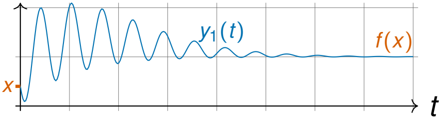
sin, cos, exp, log, ...

Considered "weak" : not Γ and ζ
Only analytic functions

Computable

$$\begin{cases} y(0) = q(x) & x \in \mathbb{R} \\ y'(t) = p(y(t)) & t \in \mathbb{R}_+ \end{cases}$$

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

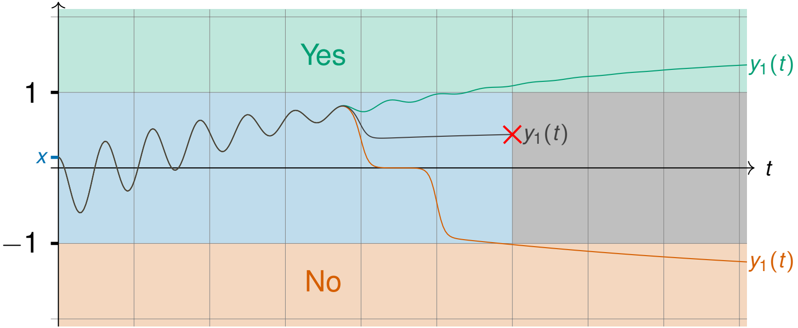


Modern notion

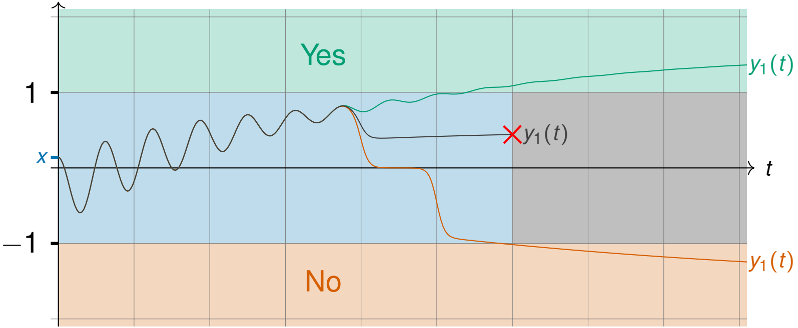
sin, cos, exp, log, Γ, ζ, ...

Turing powerful
[Bournez et al., 2007]

More formally



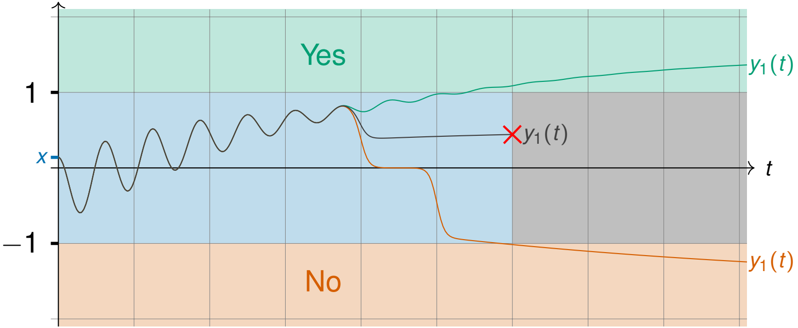
More formally



Theorem (Bournez et al, 2010)

This is equivalent to a Turing machine.

More formally



Theorem (Bournez et al, 2010)

This is equivalent to a Turing machine.

- analog computability theory
- purely continuous characterization of classical computability

How does one prove such a result ?

By computing/programming with differential equations ! Two levels :

Generable functions :

- « simple » basic blocks
- lots of way to combine them
- very low level

Computable functions :

- more comprehensible
- harder to combine
- higher level

The theory of generable functions

Generable functions (total, univariate)

Definition

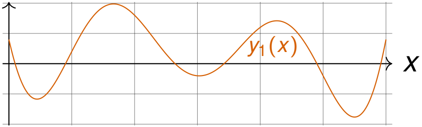
$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$



Note : existence and unicity of y by Cauchy-Lipschitz theorem.

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f(x) = x$ ▶ identity

$$y(0) = 0, \qquad y' = 1 \quad \rightsquigarrow \quad y(x) = x$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f(x) = x^2$ ► squaring

$$\begin{array}{lll} y_1(0) = 0, & y_1' = 2y_2 & \leadsto \quad y_1(x) = x^2 \\ y_2(0) = 0, & y_2' = 1 & \leadsto \quad y_2(x) = x \end{array}$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f(x) = x^n$ ► n^{th} power

$$\begin{array}{lll}
y_1(0) = 0, & y_1' = ny_2 & \rightsquigarrow \quad y_1(x) = x^n \\
y_2(0) = 0, & y_2' = (n-1)y_3 & \rightsquigarrow \quad y_2(x) = x^{n-1} \\
\ldots & \ldots & \qquad \ldots \\
y_n(0) = 0, & y_n = 1 & \rightsquigarrow \quad y_n(x) = x
\end{array}$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d,p and y_0 such that the solution y to

$$y(0)=y_0,\qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f(x)=\exp(x)$ ► exponential

$$y(0)=1,\qquad y'=y\quad\rightsquigarrow\quad y(x)=\exp(x)$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0)=y_0, \qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f(x)=\sin(x)$ or $f(x)=\cos(x)$

- sine/cosine

$$\begin{aligned} y_1(0)&=0, & y_1'&=y_2 & \rightsquigarrow\quad y_1(x)&=\sin(x)\\ y_2(0)&=1, & y_2'&=-y_1 & \rightsquigarrow\quad y_2(x)&=\cos(x)\end{aligned}$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

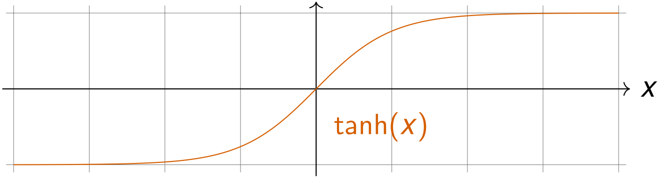
satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f(x) = \tanh(x)$ ► hyperbolic tangent

$$y(0) = 0, \qquad y' = 1 - y^2 \quad \rightsquigarrow \quad y(x) = \tanh(x)$$



Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d,p and y_0 such that the solution y to

$$y(0)=y_0,\qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f(x)=\frac{1}{1+x^2}$ ▶ rational function

$$f'(x)=\frac{-2x}{(1+x^2)^2}=-2xf(x)^2$$

$$\begin{aligned} y_1(0)&=1, & y_1'&=-2y_2y_1^2 & \rightsquigarrow\quad y_1(x)&=\frac{1}{1+x^2}\\ y_2(0)&=0, & y_2'&=1 & \rightsquigarrow\quad y_2(x)&=x \end{aligned}$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0)=y_0, \qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f=g\pm h$

- sum/difference

$$(f\pm g)'=f'\pm g'$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d,p and y_0 such that the solution y to

$$y(0)=y_0,\qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f=gh$ ▸ product

$$(gh)'=g'h+gh'$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d,p and y_0 such that the solution y to

$$y(0)=y_0,\qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f=\frac{1}{g}$ ► inverse

$$f'=\frac{-g'}{g^2}=-g'f^2$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f = \int g$ ▶ integral

$$f' = g$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f = g'$ ▶ derivative

$$f' = g'' = (p_1(z))' = \nabla p_1(z) \cdot z'$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f = g \circ h$ ▶ composition

$$(z \circ h)' = (z' \circ h)h' = p(z \circ h)h'$$

Generable functions (total, univariate)

Definition

$f:\mathbb{R}\to\mathbb{R}$ is generable if there exists d,p and y_0 such that the solution y to

$$y(0)=y_0,\qquad y'(x)=p(y(x))$$

satisfies $f(x)=y_1(x)$ for all $x\in\mathbb{R}$.

Types

- $d\in\mathbb{N}$: dimension
- $p\in\mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0\in\mathbb{R}^d, y:\mathbb{R}\to\mathbb{R}^d$

Example : $f'=\tanh\circ f$ ► Non-polynomial differential equation

$$f''=(\tanh'\circ f)f'=(1-(\tanh\circ f)^2)f'$$

Generable functions (total, univariate)

Definition

$f : \mathbb{R} \to \mathbb{R}$ is generable if there exists d, p and y_0 such that the solution y to

$$y(0) = y_0, \qquad y'(x) = p(y(x))$$

satisfies $f(x) = y_1(x)$ for all $x \in \mathbb{R}$.

Types

- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{R}^d[\mathbb{R}^n]$: polynomial vector
- $y_0 \in \mathbb{R}^d, y : \mathbb{R} \to \mathbb{R}^d$

Example : $f(0) = f_0, f' = g \circ f$ ▶ Initial Value Problem (IVP)

$$f' = g'' = (p(z))' = \nabla p(z) \cdot z'$$

Generable functions : a first summary

Nice theory for the class of total and univariate generable functions :

- analytic
- contains polynomials, $\sin$, $\cos$, $\tanh$, $\exp$
- stable under $\pm, \times, /, \circ$ and Initial Value Problems (IVP)
- technicality on the field $\mathbb{K}$ of coefficients for stability under $\circ$

Generable functions : a first summary

Nice theory for the class of total and univariate generable functions :

- analytic
- contains polynomials, sin, cos, tanh, exp
- stable under $\pm, \times, /, \circ$ and Initial Value Problems (IVP)
- technicality on the field $\mathbb{K}$ of coefficients for stability under $\circ$

Limitations :

- total functions
- univariate

Generable functions (generalization)

Definition

$f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ is generable if X is open **connected** and $\exists d, p, x_0, y_0, y$ such that

$$y(x_0) = y_0, \qquad J_y(x) = p(y(x))$$

and $f(x) = y_1(x)$ for all $x \in X$.

$J_y(x) =$ Jacobian matrix of y at x

Types

- $n \in \mathbb{N}$: input dimension
- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{K}^{d\times d}[\mathbb{R}^d]$: polynomial matrix
- $x_0 \in \mathbb{K}^n$
- $y_0 \in \mathbb{K}^d, y : X \to \mathbb{R}^d$

Notes :

- Partial differential equation !
- Unicity of solution y...
- ... **but not existence** (ie you have to show it exists)

Generable functions (generalization)

Definition

$f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ is generable if X is open **connected** and $\exists d, p, x_0, y_0, y$ such that

$$y(x_0) = y_0, \qquad J_y(x) = p(y(x))$$

and $f(x) = y_1(x)$ for all $x \in X$.

$J_y(x) =$ Jacobian matrix of y at x

Types

- $n \in \mathbb{N}$: input dimension
- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{K}^{d \times d}[\mathbb{R}^d]$: polynomial matrix
- $x_0 \in \mathbb{K}^n$
- $y_0 \in \mathbb{K}^d, y : X \to \mathbb{R}^d$

Example : $f(x_1, x_2) = x_1 x_2^2 \quad (n = 2, d = 3)$ ▶ monomial

$$y(0,0) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad J_y = \begin{pmatrix} y_3^2 & 3y_2y_3 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \rightsquigarrow \quad y(x) = \begin{pmatrix} x_1 x_2^2 \\ x_1 \\ x_2 \end{pmatrix}$$

Generable functions (generalization)

Definition

$f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ is generable if X is open **connected** and $\exists d, p, x_0, y_0, y$ such that

$$y(x_0) = y_0, \qquad J_y(x) = p(y(x))$$

and $f(x) = y_1(x)$ for all $x \in X$.

$J_y(x) =$ Jacobian matrix of y at x

Types

- $n \in \mathbb{N}$: input dimension
- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{K}^{d \times d}[\mathbb{R}^d]$: polynomial matrix
- $x_0 \in \mathbb{K}^n$
- $y_0 \in \mathbb{K}^d, y : X \to \mathbb{R}^d$

Example : $f(x_1, x_2) = x_1 x_2^2$ ► monomial

$$\begin{array}{llll} y_1(0,0) = 0, & \partial_{x_1} y_1 = y_3^2, & \partial_{x_2} y_1 = 3y_2 y_3 & \rightsquigarrow \quad y_1(x) = x_1 x_2^2 \\ y_2(0,0) = 0, & \partial_{x_1} y_2 = 1, & \partial_{x_2} y_2 = 0 & \rightsquigarrow \quad y_2(x) = x_1 \\ y_3(0,0) = 0, & \partial_{x_1} y_3 = 0, & \partial_{x_2} y_3 = 1 & \rightsquigarrow \quad y_3(x) = x_2 \end{array}$$

This is tedious !

Generable functions (generalization)

Definition

$f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ is generable if X is open **connected** and $\exists d, p, x_0, y_0, y$ such that

$$y(x_0) = y_0, \qquad J_y(x) = p(y(x))$$

and $f(x) = y_1(x)$ for all $x \in X$.

$J_y(x) =$ Jacobian matrix of y at x

Types

- $n \in \mathbb{N}$: input dimension
- $d \in \mathbb{N}$: dimension
- $p \in \mathbb{K}^{d \times d}[\mathbb{R}^d]$: polynomial matrix
- $x_0 \in \mathbb{K}^n$
- $y_0 \in \mathbb{K}^d, y : X \to \mathbb{R}^d$

Last example : $f(x) = \frac{1}{x}$ for $x \in (0, \infty)$ ► inverse function

$$y(1) = 1, \qquad \partial_x y = -y^2 \quad \rightsquigarrow \quad y(x) = \frac{1}{x}$$

Generable functions : summary

Nice theory for the class of multivariate generable functions (over connected domains) :

- analytic
- contains polynomials, $\sin$, $\cos$, $\tanh$, $\exp$
- stable under $\pm, \times, /, \circ$ and Initial Value Problems (IVP)
- technicality on the field $\mathbb{K}$ of coefficients for stability under $\circ$

Generable functions : summary

Nice theory for the class of multivariate generable functions (over connected domains) :

- analytic
- contains polynomials, $\sin$, $\cos$, $\tanh$, $\exp$
- stable under $\pm, \times, /, \circ$ and Initial Value Problems (IVP)
- technicality on the field $\mathbb{K}$ of coefficients for stability under $\circ$

Natural questions :

- analytic $\rightarrow$ isn't that very limited ?
- can we generate all analytic functions ?

Generable functions : summary

Nice theory for the class of multivariate generable functions (over connected domains) :

- analytic
- contains polynomials, sin, cos, tanh, exp
- stable under $\pm, \times, /, \circ$ and Initial Value Problems (IVP)
- technicality on the field $\mathbb{K}$ of coefficients for stability under $\circ$

Natural questions :

- analytic $\rightarrow$ isn't that very limited ?
- can we generate all analytic functions ? No

Riemann Γ and ζ are not generable.

Why is this useful?

Writing polynomial ODEs by hand is hard.

Why is this useful?

Writing polynomial ODEs by hand is hard.

Using generable functions, we can build complicated **multivariate partial functions** using other operations, and we know they are solutions to polynomial ODEs **by construction**.

Why is this useful ?

Writing polynomial ODEs by hand is hard.

Using generable functions, we can build complicated **multivariate partial functions** using other operations, and we know they are solutions to polynomial ODEs **by construction**.

Example : almost rounding function

There exists a generable function round such that for any $n \in \mathbb{Z}$, $x \in \mathbb{R}$, $\lambda > 2$ and $\mu \geqslant 0$:

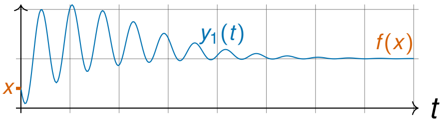
- if $x \in \left[n - \frac{1}{2}, n + \frac{1}{2}\right]$ then $|\operatorname{round}(x, \mu, \lambda) - n| \leqslant \frac{1}{2}$,
- if $x \in \left[n - \frac{1}{2} + \frac{1}{\lambda}, n + \frac{1}{2} - \frac{1}{\lambda}\right]$ then $|\operatorname{round}(x, \mu, \lambda) - n| \leqslant e^{-\mu}$.

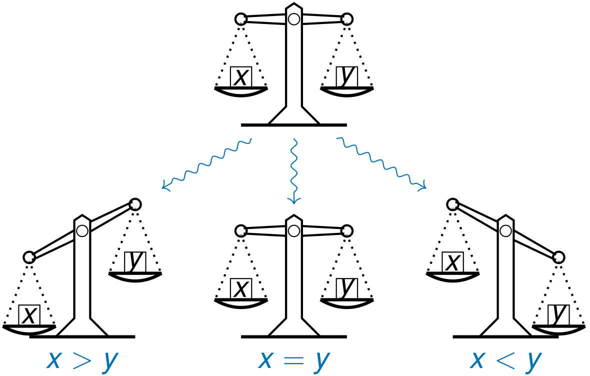
The theory of computable functions

Computable function

$$\begin{cases} y(0) = q(x) & x \in \mathbb{R} \\ y'(t) = p(y(t)) & t \in \mathbb{R}_+ \end{cases}$$

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





Inputs : $x, y \in [0, +\infty)$ Output : $\mathrm{sign}(x - y)$?

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$
- stable under $\circ$

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$
- stable under $\circ$
- stable under limits

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$
- stable under $\circ$
- stable under limits
- stable under iteration (with conditions)

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$
- stable under $\circ$
- stable under limits
- stable under iteration (with conditions)

Enough to simulate a Turing machine !

The theory of computable functions

Important fact :

- contains generable functions
- continuous functions
- stable under $\pm, \times, /$
- stable under $\circ$
- stable under limits
- stable under iteration (with conditions)

Enough to simulate a Turing machine !

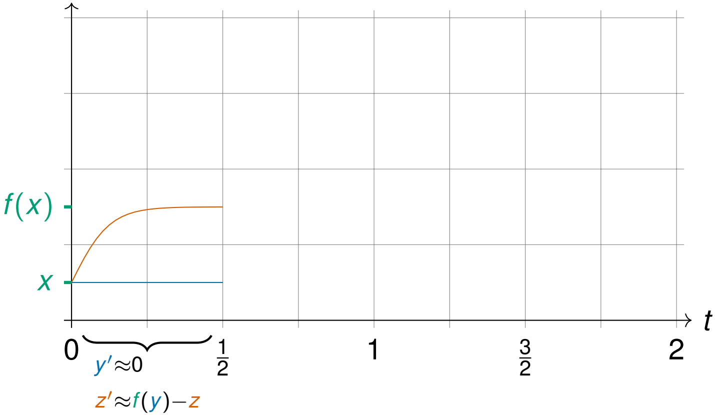
Proof are too complicated but essentially this is all error management.

Proof gem : iteration with differential equations

Assume f is generable, can we **iterate** f with an ODE ?
That is, build a generable y such that $y(x, n) \approx f^{[n]}(x)$ for all $n \in \mathbb{N}$

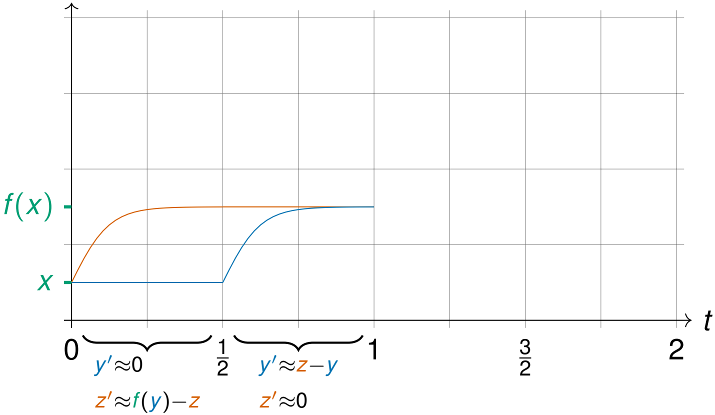
Proof gem : iteration with differential equations

Assume f is generable, can we **iterate** f with an ODE ?
That is, build a generable y such that $y(x,n) \approx f^{[n]}(x)$ for all $n \in \mathbb{N}$



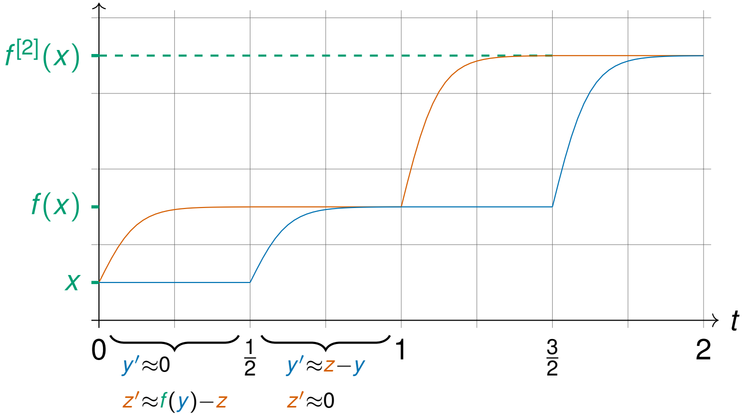
Proof gem : iteration with differential equations

Assume f is generable, can we **iterate** f with an ODE ?
That is, build a generable y such that $y(x, n) \approx f^{[n]}(x)$ for all $n \in \mathbb{N}$

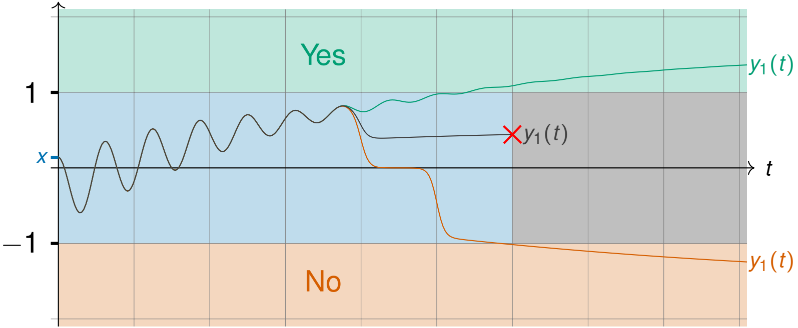


Proof gem : iteration with differential equations

Assume f is generable, can we **iterate** f with an ODE ?
That is, build a generable y such that $y(x,n) \approx f^{[n]}(x)$ for all $n \in \mathbb{N}$



Recap



Theorem (Bournez et al, 2010)

This is equivalent to a Turing machine.

- analog computability theory
- purely continuous characterization of classical computability

The **complexity** theory of computable functions

Complexity of analog systems

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

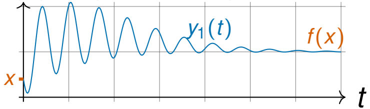
Complexity of analog systems

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

Tentative definition

$T(x) = ??$

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



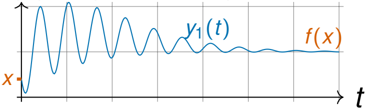
Complexity of analog systems

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

Tentative definition

$T(x, \mu) =$

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



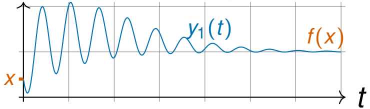
Complexity of analog systems

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

Tentative definition

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

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

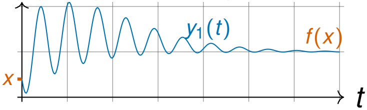


- Turing machines : $T(x) =$ number of steps to compute on x
- GPAC :

Tentative definition

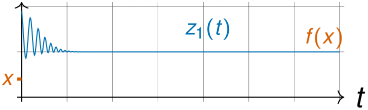
$T(x, \mu) =$ first time t so that $|y_1(t) - f(x)| \leqslant e^{-\mu}$

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



$\rightsquigarrow$

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



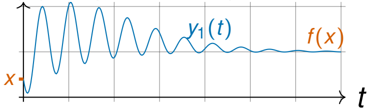
Complexity of analog systems

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

Tentative definition

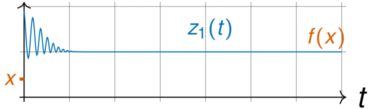
$T(x,\mu) =$ first time t so that $|y_1(t) - f(x)| \leqslant e^{-\mu}$

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

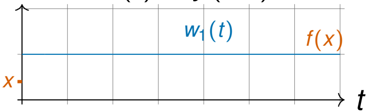


$\rightsquigarrow$

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



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



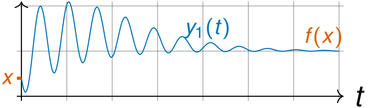
Complexity of analog systems

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

Tentative definition

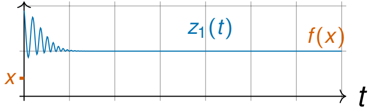
$T(x,\mu)$ = first time t so that $|y_1(t) - f(x)| \leqslant e^{-\mu}$

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

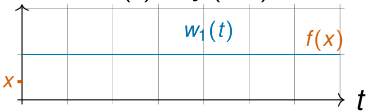


$\rightsquigarrow$

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



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

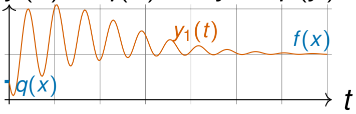


Something is wrong...

All functions have constant time complexity.

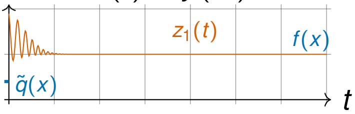
Time-space correlation of the GPAC

$y(0) = q(x) \qquad y' = p(y)$

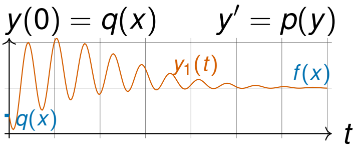


$\rightsquigarrow$

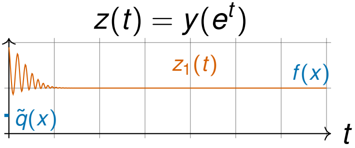
$z(t) = y(e^t)$



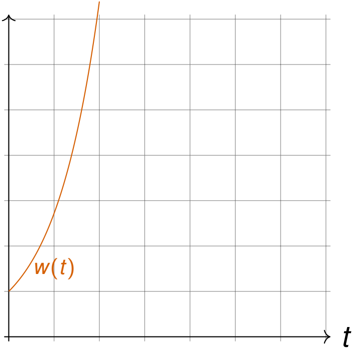
Time-space correlation of the GPAC



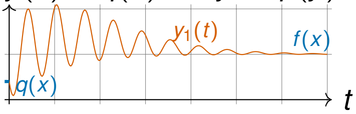
$\rightsquigarrow$



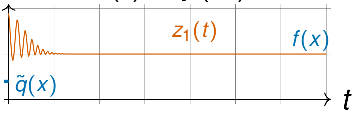
extra component : $w(t) = e^t$



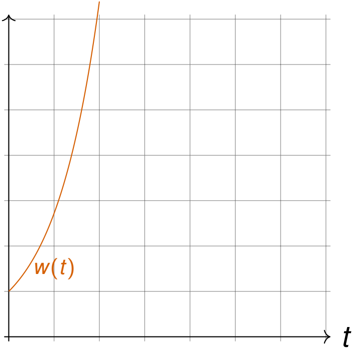
Time-space correlation of the GPAC



$\rightsquigarrow$



extra component : $w(t) = e^t$



Observation

Time scaling costs "space".

$\rightsquigarrow$

Time complexity for the GPAC must involve time and space !

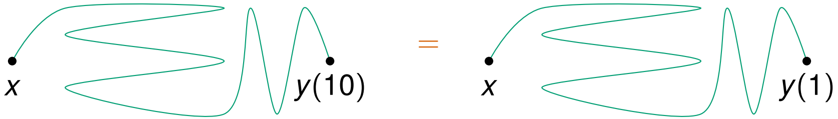
Complexity in the analog world

Complexity measure : length of the curve

Time acceleration : same curve = same complexity !

Complexity in the analog world

Complexity measure : length of the curve



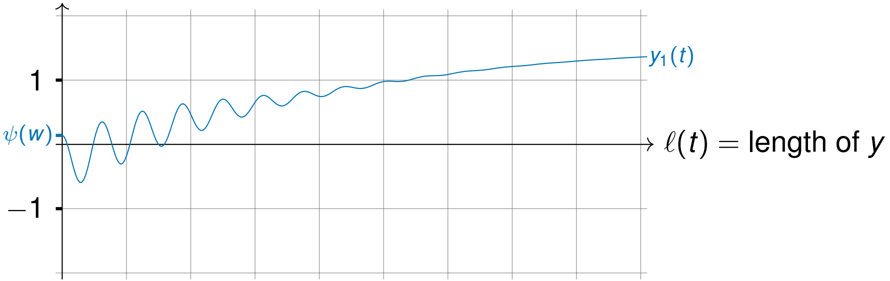
Time acceleration : same curve = same complexity !

Same time, different curves : different complexity !

Characterization of polynomial time

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

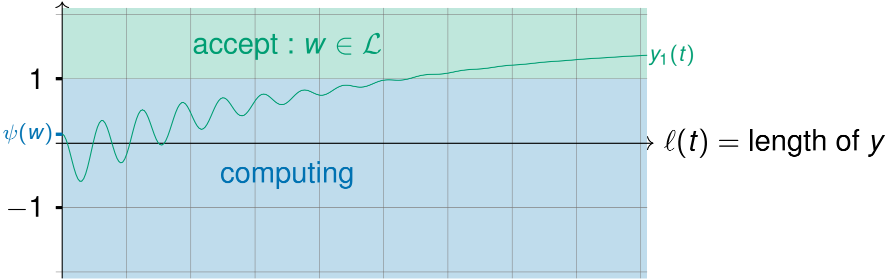
$$y(0) = (\psi(w), |w|, 0, \ldots, 0) \qquad y' = p(y) \qquad \psi(w) = \sum_{i=1}^{|w|} w_i 2^{-i}$$



Characterization of polynomial time

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

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



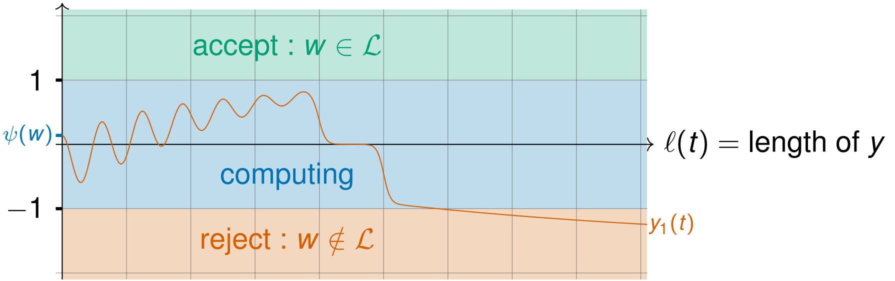
satisfies

1. if $y_1(t) \geqslant 1$ then $w \in \mathcal{L}$

Characterization of polynomial time

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

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



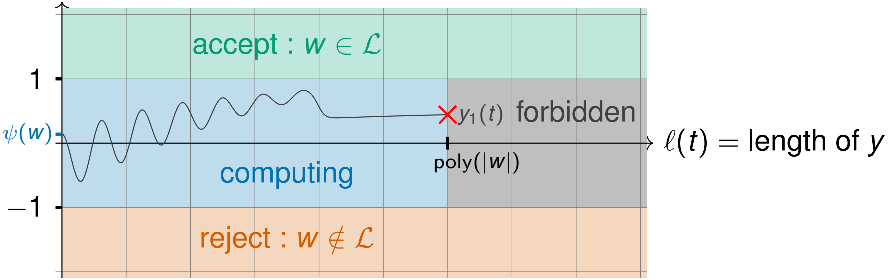
satisfies

2. if $y_1(t) \leqslant -1$ then $w \notin \mathcal{L}$

Characterization of polynomial time

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

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



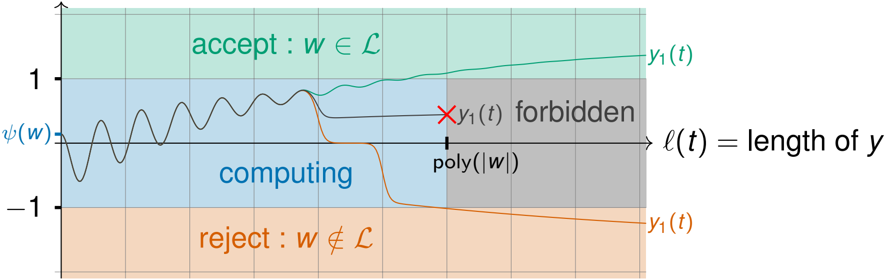
satisfies

3. if $\ell(t) \geqslant \text{poly}(|w|)$ then $|y_1(t)| \geqslant 1$

Characterization of polynomial time

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

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

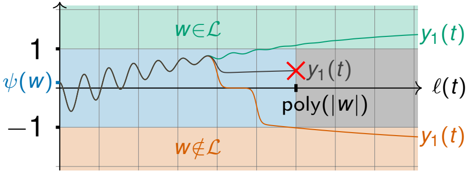


Theorem

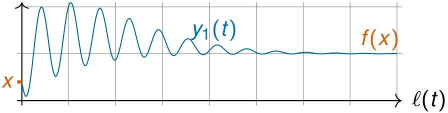
PTIME = ANALOG-PTIME

Summary

ANALOG-PTIME



ANALOG-$P_{\mathbb{R}}$



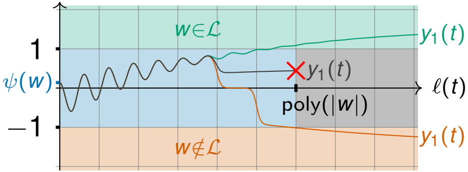
Theorem

- $\mathcal{L} \in$ PTIME *of and only if* $\mathcal{L} \in$ ANALOG-PTIME
- $f : [a, b] \to \mathbb{R}$ *computable in polynomial time* $\Leftrightarrow f \in$ ANALOG-$P_{\mathbb{R}}$

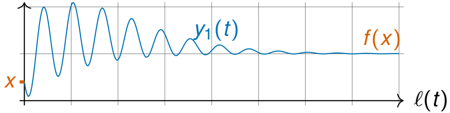
- Analog complexity theory based on length
- Time of Turing machine $\Leftrightarrow$ length of the GPAC
- Purely continuous characterization of PTIME

Summary

ANALOG-PTIME



ANALOG-$P_{\mathbb{R}}$



Theorem

- $\mathcal{L} \in$ PTIME *of and only if* $\mathcal{L} \in$ ANALOG-PTIME
- $f : [a, b] \to \mathbb{R}$ *computable in polynomial time* $\Leftrightarrow f \in$ ANALOG-$P_{\mathbb{R}}$

- Analog complexity theory based on length
- Time of Turing machine $\Leftrightarrow$ length of the GPAC
- Purely continuous characterization of PTIME
- Only rational coefficients needed

Chemical Reaction Networks

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 = rate)

Example (any resemblance to chemistry is purely coincidental) :

$$\begin{array}{ccccc} 2H & + & O & \to & H_2O \\ C & + & O_2 & \to & CO_2 \end{array}$$

Chemical Reaction Networks

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

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

Example (any resemblance to chemistry is purely coincidental) :

$$\begin{array}{ccccc} 2\mathrm{H} & + & \mathrm{O} & \rightarrow & \mathrm{H_2O} \\ \mathrm{C} & + & \mathrm{O_2} & \rightarrow & \mathrm{CO_2} \end{array}$$

Assumption : law of mass action

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

Chemical Reaction Networks

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

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

Example (any resemblance to chemistry is purely coincidental) :

$$\begin{array}{rcccl} 2\text{H} & + & \text{O} & \rightarrow & \text{H}_2\text{O} \\ \text{C} & + & \text{O}_2 & \rightarrow & \text{CO}_2 \end{array}$$

Assumption : law of mass action

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

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 $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \qquad (a_i, b_i \in \mathbb{N}, f = \text{rate})$

Example (any resemblance to chemistry is purely coincidental) :

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

Assumption : law of mass action

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

Semantics :

- discrete
- differential $\rightarrow$
- 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 $y_1, \dots, y_n$
- reactions of the form $\sum_i a_i y_i \xrightarrow{f} \sum_i b_i y_i \qquad (a_i, b_i \in \mathbb{N}, f = \text{rate})$

Example (any resemblance to chemistry is purely coincidental) :

$$\begin{array}{rcccl} 2\mathrm{H} & + & \mathrm{O} & \rightarrow & \mathrm{H_2O} \\ \mathrm{C} & + & \mathrm{O_2} & \rightarrow & \mathrm{CO_2} \end{array}$$

Assumption : law of mass action

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

Semantics :

- discrete
- differential $\rightarrow$
- 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

Chemical Reaction Networks (CRNs)

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

CRNs are Turing complete ?

Chemical Reaction Networks (CRNs)

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 ($y_i < 0$) ► easy to solve
- arbitrary reactions are not realistic ► what is realistic ?

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
$\Rightarrow$ 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
$\Rightarrow$ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to **quadratic** ODEs :

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

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
$\Rightarrow$ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to **quadratic** ODEs :

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

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
$\Rightarrow$ can be implemented with DNA, RNA or proteins

Elementary reactions correspond to **quadratic** ODEs :

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

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 :

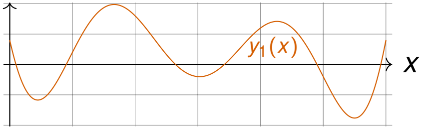
- proof preserves polynomial length
- in fact the following elementary reactions suffice :

$$\varnothing \xrightarrow{k} x \qquad x \xrightarrow{k} x + z \qquad x + y \xrightarrow{k} x + y + z \qquad x + y \xrightarrow{k} \varnothing$$

Universal differential equation

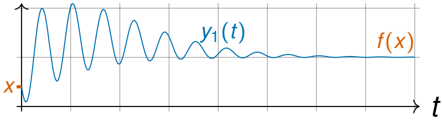
Universal differential equations

Generable functions



subclass of analytic functions

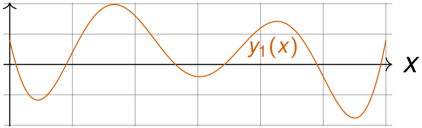
Computable functions



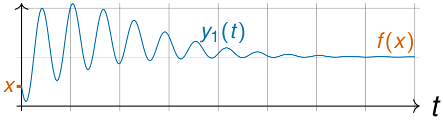
any computable function

Universal differential equations

Generable functions

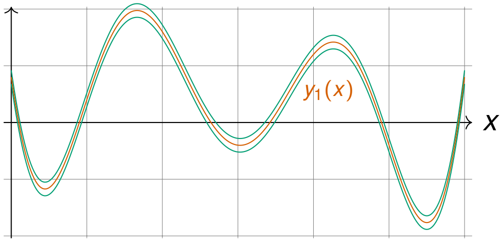


Computable functions

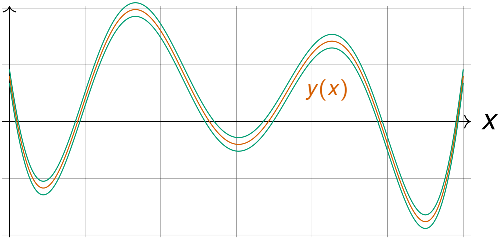


subclass of analytic functions

any computable function



Universal differential algebraic equation (DAE)



Theorem (Rubel, 1981)

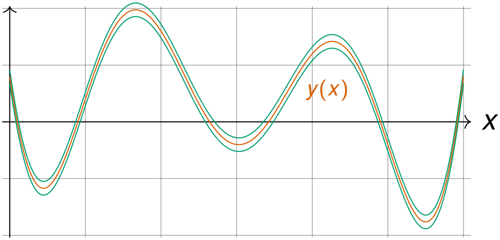
For any continuous functions f and ε, there exists $y:\mathbb{R}\to\mathbb{R}$ solution to

$$\begin{aligned}3y'^4y''y''''^2 &-4y'^4y'''^2y''''+6y'^3y''^2y'''y''''+24y'^2y''^4y'''' \\ &-12y'^3y''y'''^3-29y'^2y''^3y'''^2+12y''^7 \qquad = 0\end{aligned}$$

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

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

Universal differential algebraic equation (DAE)



Theorem (Rubel, 1981)

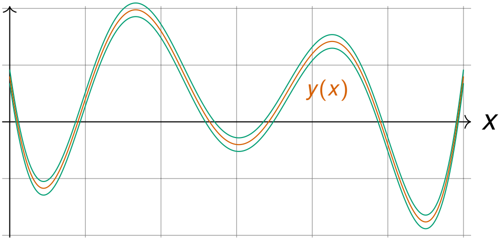
*There exists a **fixed** polynomial p and $k \in \mathbb{N}$ such that for any continuous functions f and ε, there exists a solution $y : \mathbb{R} \to \mathbb{R}$ to*

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

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

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

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 ε, there exists a solution $y : \mathbb{R} \to \mathbb{R}$ to*

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

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

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

Problem : this is «weak» result.

The problem with Rubel's DAE

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

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

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** :

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

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

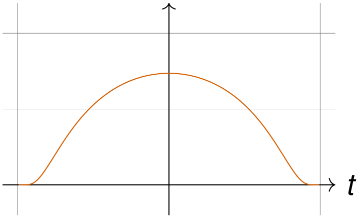
Open Problem (Rubel, 1981)

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 $\quad (1-t^2)^2 f''(t) + 2tf'(t) = 0.$



Rubel's proof in one slide

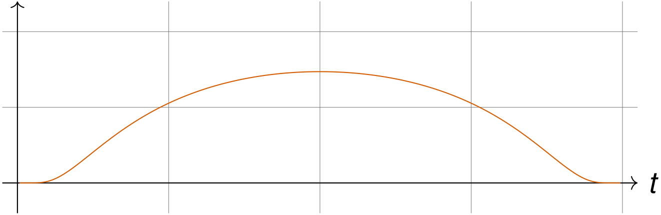
- Take $f(t) = e^{\frac{-1}{1-t^2}}$ for $-1 < t < 1$ and $f(t) = 0$ otherwise.

  $$\text{It satisfies} \quad (1-t^2)^2 f''(t) + 2tf'(t) = 0.$$

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

$$\begin{aligned} 3y'^4 y'' y''''^2 \quad & -4y'^4 y''^2 y'''' + 6y'^3 y''^2 y''' y'''' + 24y'^2 y''^4 y'''' \\ & -12y'^3 y'' y'''^3 - 29y'^2 y''^3 y'''^2 + 12y''^7 = 0 \end{aligned}$$

Translation and rescaling :



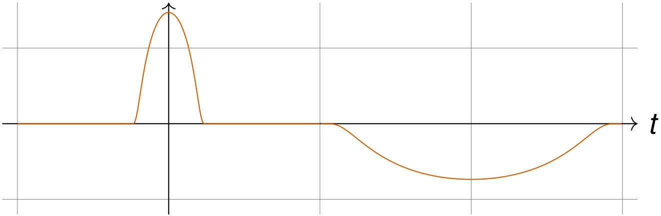
Rubel's proof in one slide

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

  It satisfies $\quad (1 - t^2)^2 f''(t) + 2tf'(t) = 0.$
- For any $a, b, c \in \mathbb{R}$, $y(t) = cf(at + b)$ satisfies

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

- Can glue together arbitrary many such pieces



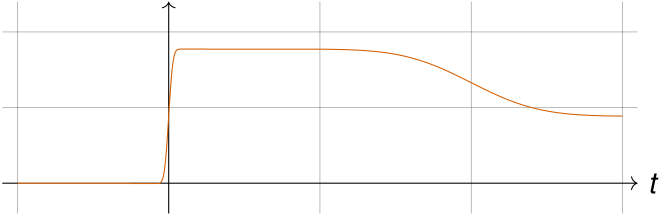
Rubel's proof in one slide

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

  It satisfies $\quad (1 - t^2)^2 f''(t) + 2tf'(t) = 0.$
- For any $a, b, c \in \mathbb{R}$, $y(t) = cf(at + b)$ satisfies

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

- Can glue together arbitrary many such pieces
- Can arrange so that $\int f$ is solution : piecewise pseudo-linear

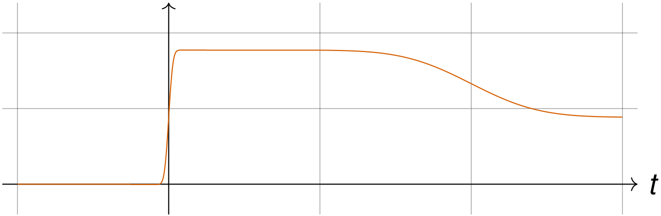


Rubel's proof in one slide

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

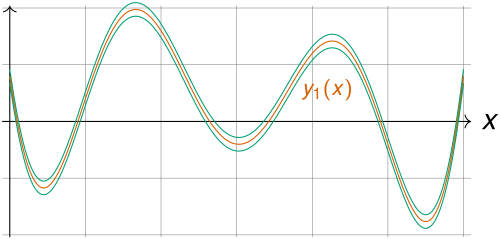
  It satisfies $(1-t^2)^2 f''(t) + 2tf'(t) = 0$.
- For any $a, b, c \in \mathbb{R}$, $y(t) = cf(at+b)$ satisfies

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

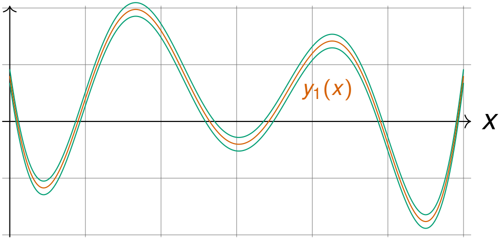
*There exists a **fixed** (vector of) polynomial p such that for any continuous functions f and ε, there exists $\alpha \in \mathbb{R}^d$ such that*

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

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

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

Universal initial value problem (IVP)



Notes :

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

Theorem

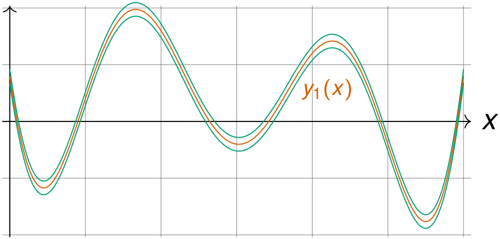
There exists a ***fixed*** *(vector of) polynomial p such that for any continuous functions f and* ε*, there exists* $\alpha \in \mathbb{R}^d$ *such that*

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

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

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

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* ε*, there exists* $\alpha \in \mathbb{R}^d$ *such that*

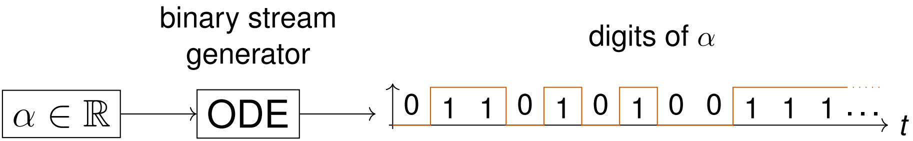
$$y(0) = \alpha, \qquad y'(t) = p(y(t))$$

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

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

Remark : α is usually transcendental, but computable from f and ε

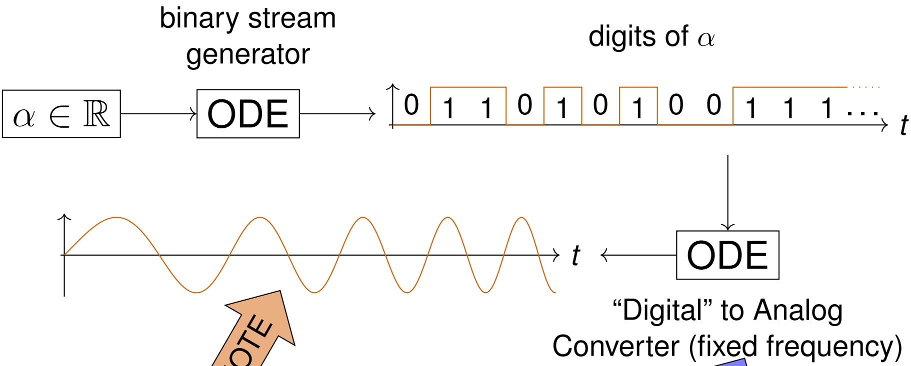
A simplified proof



NOTE

This is the ideal curve, the real one is an approximation of it.

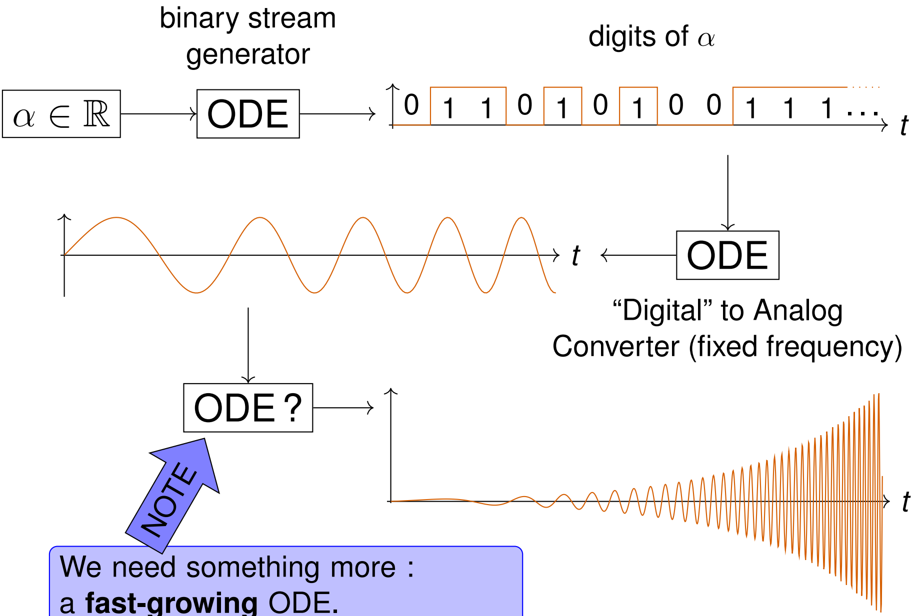
A simplified proof



Approximate Lipschitz and bounded functions with **fixed** precision.

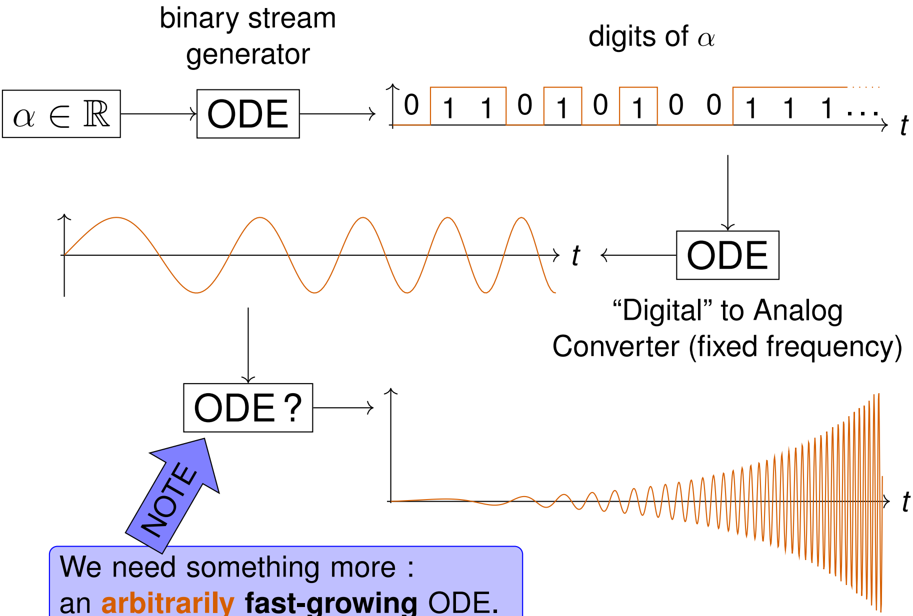
That's the trickiest part.

A simplified proof



We need something more :
a **fast-growing** ODE.

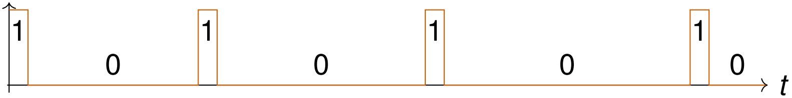
A simplified proof



We need something more :
an **arbitrarily** **fast-growing** ODE.

A less simplified proof

binary stream generator : digits of $\alpha \in \mathbb{R}$



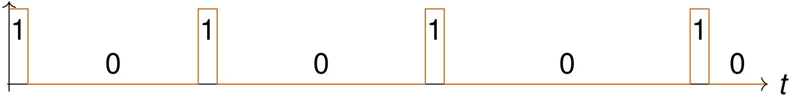
$$f(\alpha, \mu, \lambda, t) = \frac{1}{2} + \frac{1}{2}\tanh(\mu \sin(2\alpha\pi 4^{\text{round}(t-1/4,\lambda)} + 4\pi/3))$$

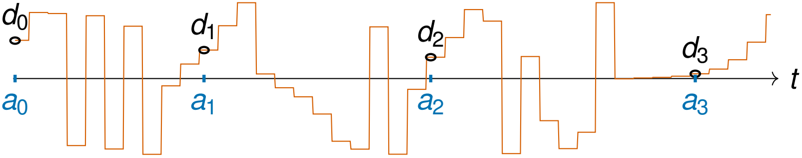
It’s horrible, but generable

round is the mysterious rounding function...

A less simplified proof

binary stream generator : digits of $\alpha \in \mathbb{R}$



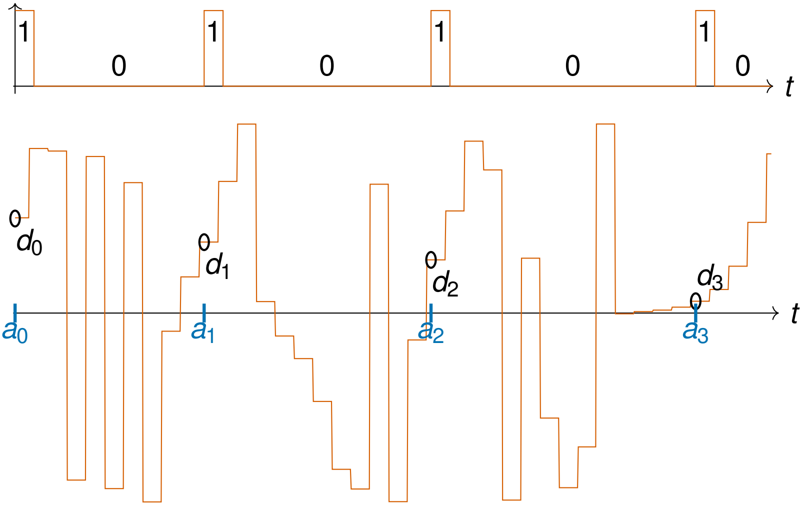


dyadic stream generator : $d_i = m_i 2^{-d_i}$, $a_i = 9i + \sum_{j<i} d_j$

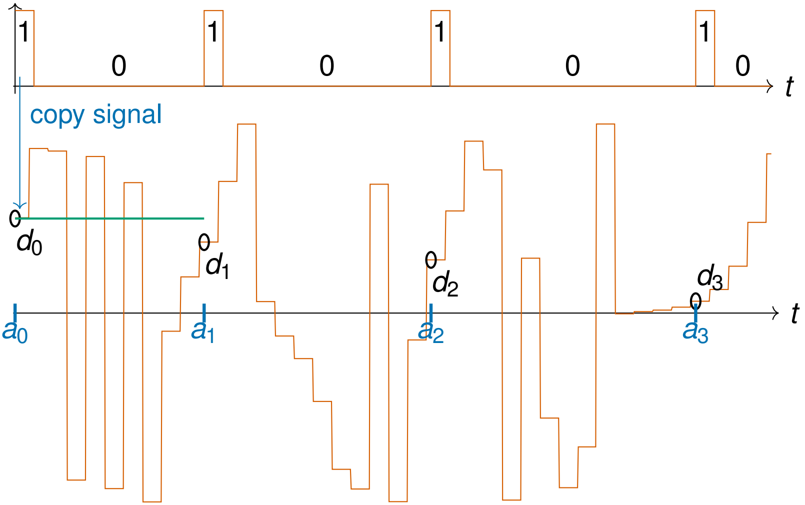
$$f(\alpha, \gamma, t) = \sin(2\alpha\pi 2^{\text{round}(t-1/4,\gamma)}))$$

round is the mysterious rounding function...

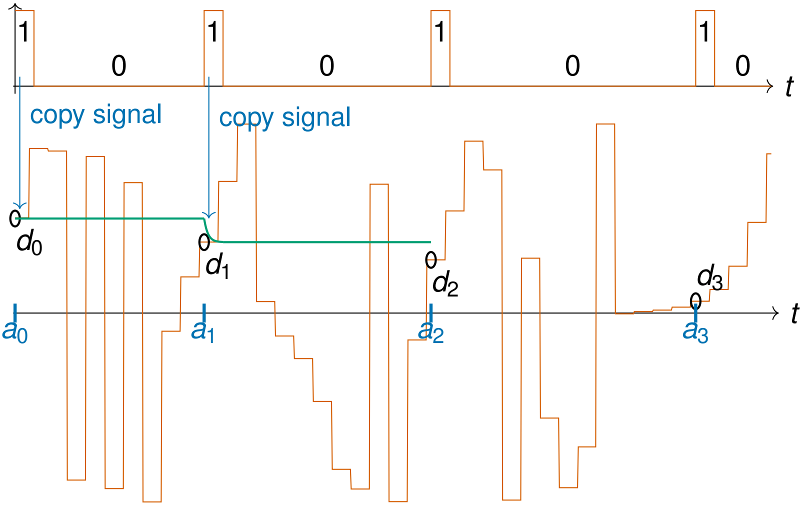
A less simplified proof



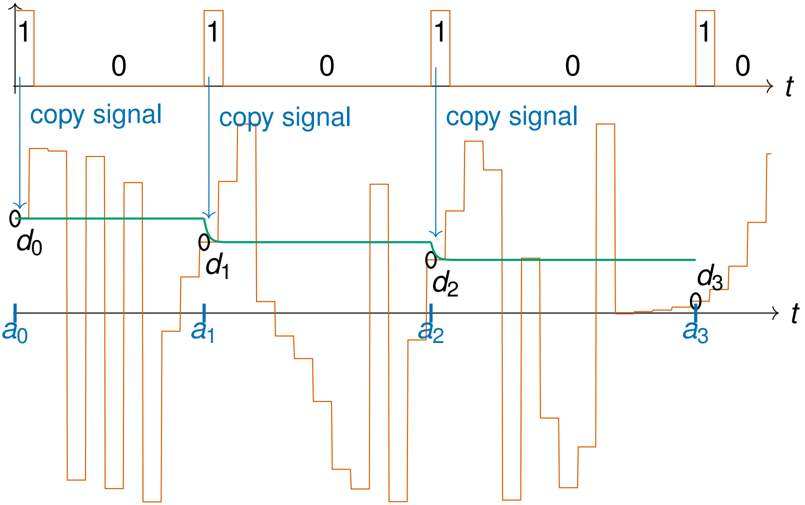
A less simplified proof



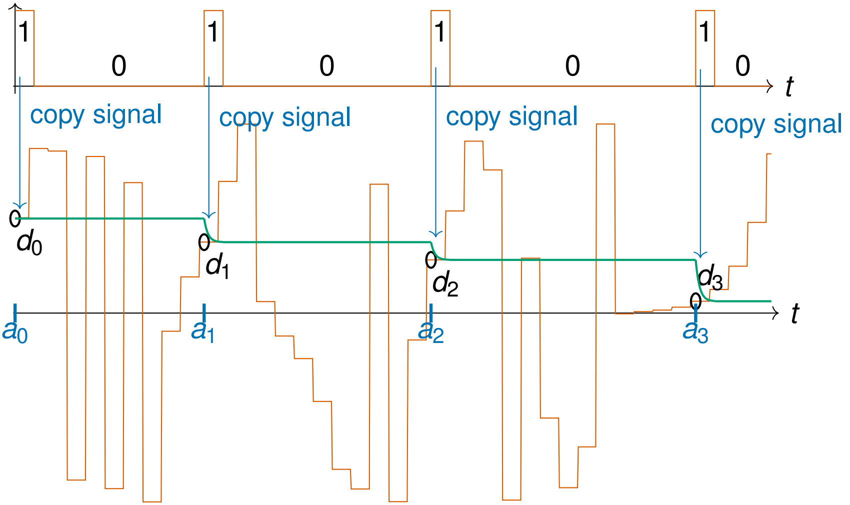
A less simplified proof



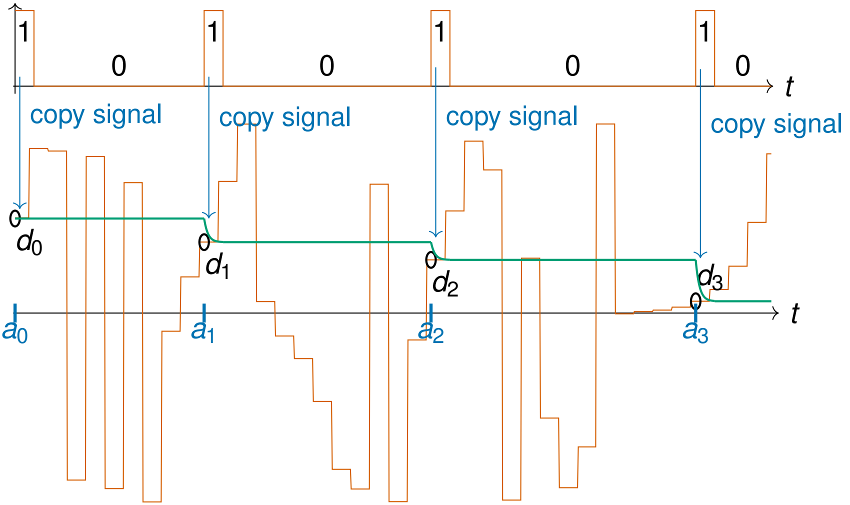
A less simplified proof



A less simplified proof

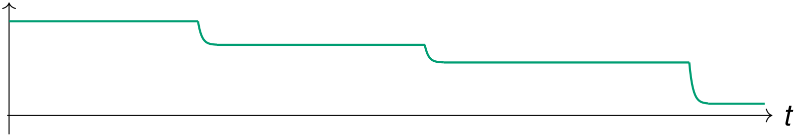


A less simplified proof



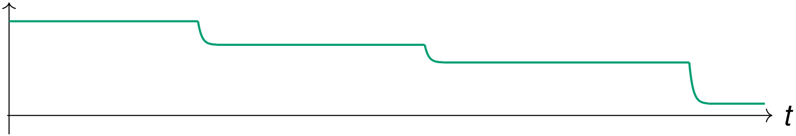
This copy operation is the “non-trivial” part.

A less simplified proof



We can do **almost piecewise constant functions...**

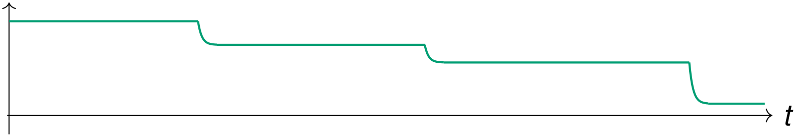
A less simplified proof



We can do **almost piecewise constant functions...**

- ...that are bounded by 1...
- ...and have super slow changing frequency.

A less simplified proof



We can do **almost piecewise constant functions...**

- ...that are bounded by 1...
- ...and have super slow changing frequency.

How do we go to arbitrarily large and growing functions ? Can a polynomial ODE even have arbitrary growth ?

An old question on growth

Building a fast-growing ODE, that exists over $\mathbb{R}$:

$$y_1' = y_1 \qquad \rightsquigarrow \qquad y_1(t) = \exp(t)$$

An old question on growth

Building a fast-growing ODE, that exists over $\mathbb{R}$:

$$
\begin{array}{lcl}
y_1' = y_1 & \rightsquigarrow & y_1(t) = \exp(t) \\
y_2' = y_1 y_2 & \rightsquigarrow & y_1(t) = \exp(\exp(t))
\end{array}
$$

An old question on growth

Building a fast-growing ODE, that exists over $\mathbb{R}$:

$$\begin{array}{lcl} y_1' = y_1 & \rightsquigarrow & y_1(t) = \exp(t) \\ y_2' = y_1 y_2 & \rightsquigarrow & y_1(t) = \exp(\exp(t)) \\ \ldots & & \ldots \\ y_n' = y_1 \cdots y_n & \rightsquigarrow & y_n(t) = \exp(\cdots \exp(t) \cdots) := e_n(t) \end{array}$$

An old question on growth

Building a fast-growing ODE, that exists over $\mathbb{R}$:

$$\begin{array}{lcl} y_1' = y_1 & \rightsquigarrow & y_1(t) = \exp(t) \\ y_2' = y_1 y_2 & \rightsquigarrow & y_1(t) = \exp(\exp(t)) \\ \dots & & \dots \\ y_n' = y_1 \cdots y_n & \rightsquigarrow & y_n(t) = \exp(\cdots \exp(t) \cdots) := e_n(t) \end{array}$$

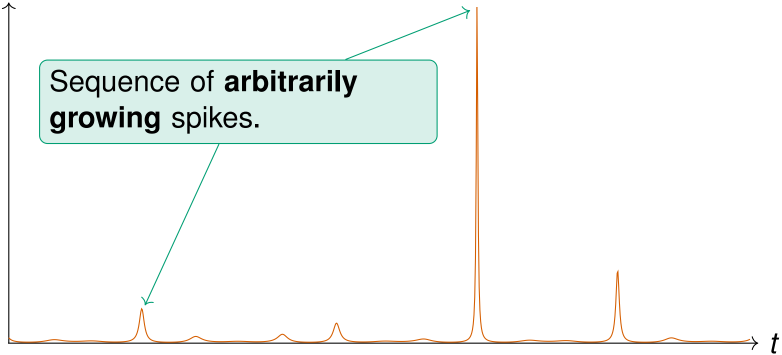
Conjecture (Emil Borel, 1899)

With n variables, cannot do better than $\mathcal{O}_t(e_n(At^k))$.

An old question on growth

Counter-example (Vijayaraghavan, 1932)

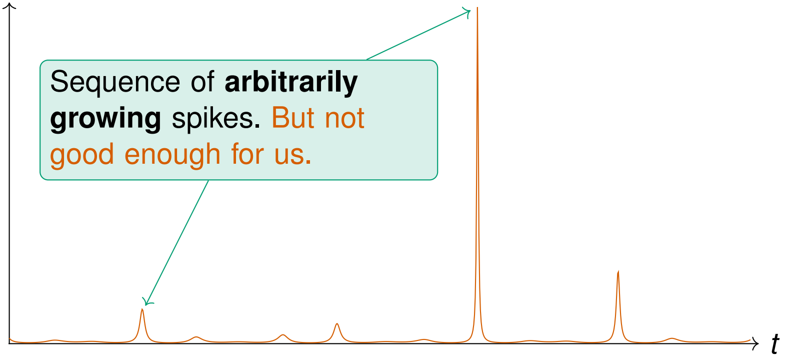
$$\frac{1}{2 - \cos(t) - \cos(\alpha t)}$$



An old question on growth

Counter-example (Vijayaraghavan, 1932)

$$\frac{1}{2 - \cos(t) - \cos(\alpha t)}$$



An old question on growth

Theorem

There exists a polynomial $p : \mathbb{R}^d \to \mathbb{R}^d$ *such that for any continuous function* $f : \mathbb{R}_+ \to \mathbb{R}$*, we can find* $\alpha \in \mathbb{R}^d$ *such that*

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

satisfies

$$y_1(t) \geqslant f(t), \qquad \forall t \geqslant 0.$$

An old question on growth

Theorem

There exists a polynomial $p : \mathbb{R}^d \to \mathbb{R}^d$ *such that for any continuous function* $f : \mathbb{R}_+ \to \mathbb{R}$*, we can find* $\alpha \in \mathbb{R}^d$ *such that*

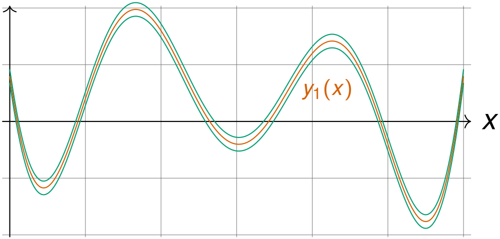
$$y(0) = \alpha, \qquad y'(t) = p(y(t))$$

satisfies

$$y_1(t) \geqslant f(t), \qquad \forall t \geqslant 0.$$

Note : both results require α to be **transcendental.** Conjecture still open for **rational** (or algebraic) coefficients.

Universal initial value problem (IVP)



Theorem

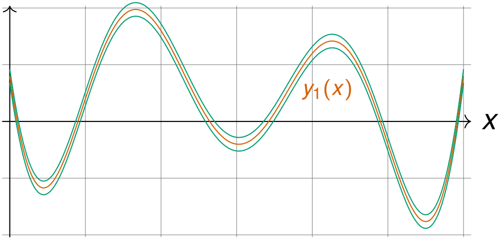
There exists a ***fixed*** *(vector of) polynomial p such that for any continuous functions f and ε, there exists $\alpha \in \mathbb{R}^d$ such that*

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

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

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

Universal initial value problem (IVP)



Notes :

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

Theorem

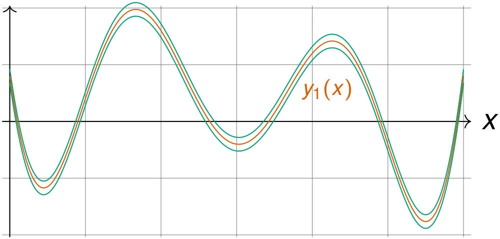
There exists a ***fixed*** *(vector of) polynomial p such that for any continuous functions f and ε, there exists $\alpha \in \mathbb{R}^d$ such that*

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

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

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

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* ε*, there exists* $\alpha \in \mathbb{R}^d$ *such that*

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

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

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

Remark : α is usually transcendental, but computable from f and ε

Future work

Reaction networks :

- chemical
- enzymatic

$$y' = p(y)$$

?

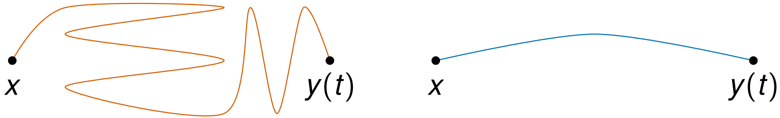
$$y' = p(y) + e(t)$$

- Finer time complexity (linear)
- Nondeterminism
- Robustness
- « Space» complexity
- Other models
- Stochastic

Backup slides

Complexity of solving polynomial ODEs

$$y(0) = x \qquad y'(t) = p(y(t))$$



Complexity of solving polynomial ODEs

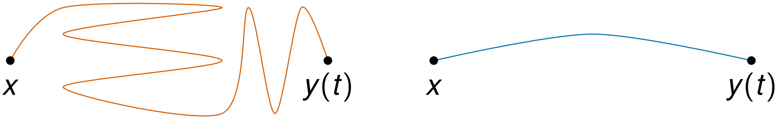
$$y(0) = x \qquad y'(t) = p(y(t))$$

Theorem

If $y(t)$ exists, one can compute p, q such that $\left|\frac{p}{q} - y(t)\right| \leqslant 2^{-n}$ in time

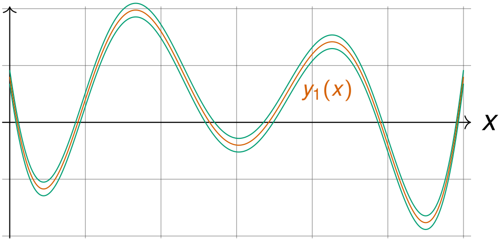
$$\operatorname{poly}(\textit{size of } x \textit{ and } p, n, \ell(t))$$

where $\ell(t) \approx$ length of the curve (between x and $y(t)$)



length of the curve = complexity = ressource

Universal DAE revisited



Theorem

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

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

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

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