

# Analogue Characterisations of Complexity Classes

The cost of computing with real numbers

---

Manon Blanc

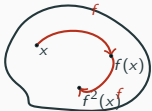
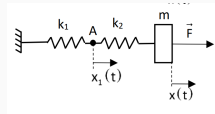
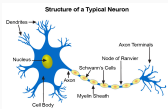
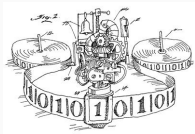


IT-UNIVERSITETET | KØBENHAVN

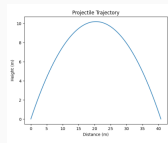
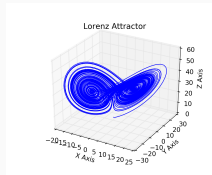
# Introduction

---

# Introduction: what are we studying?

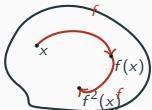
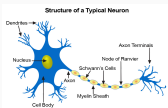


```
int x = 1;
int y = 1;
int sum = x + y;
cout << sum;
```



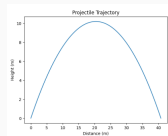
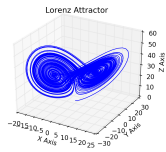
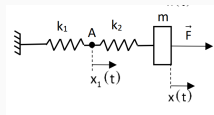
# Introduction: what are we studying?

## Discrete-time



```
int x = 1;  
int y = 1;  
int sum = x + y;  
cout << sum;
```

## Continuous-time



# Introduction

- Continuous-Time Systems: can be described by a (continuous) ODE

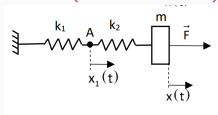
$$\frac{\partial \mathbf{f}(t, \mathbf{x})}{\partial t} = \mathbf{u}(\mathbf{f}(t, \mathbf{x}), t, \mathbf{x})$$

- Discrete-Time Systems: can be described by a discrete ODE

$$\frac{\delta \mathbf{f}(n, \mathbf{y})}{\delta n} = \mathbf{f}(n+1, \mathbf{y}) - \mathbf{f}(n, \mathbf{y}) = \mathbf{u}(\mathbf{f}(n, \mathbf{y}), n, \mathbf{y})$$

# Introduction

- Continuous-Time Systems: can be described by a (continuous) ODE



$$\frac{\partial^2 x}{\partial^2 t}(t) + \frac{k}{m}x(t) = 0$$

- Discrete-Time Systems: can be described by a discrete ODE

$$f : n \rightarrow 2^n$$

$$\frac{\delta f(n, \mathbf{y})}{\delta n} = f^2(n) - f(n)$$

# Motivations

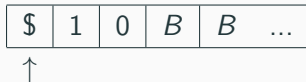
---

# Motivations

- **Verification**: can a system reach an unsafe state from an initial state (reachability)
- **Models of computation**: what is the computational power of a given model
- **Ressources**: how to measure time, memory (space) used by a computation

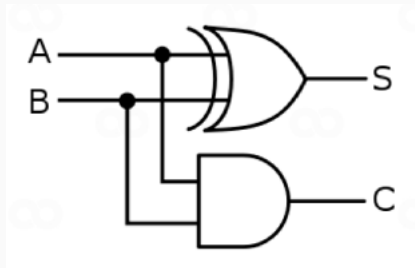
# Discrete models of computation: Turing machines

The model:

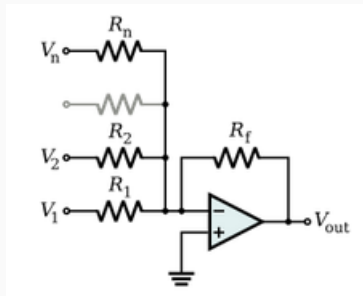
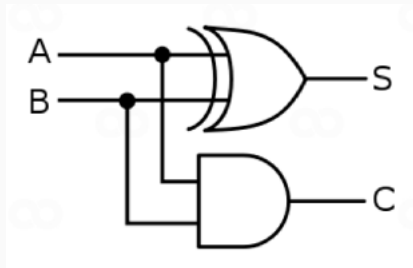


<b>Physical World</b>	<b>Mathematical Model</b>
Computer	Turing machines, boolean circuits...

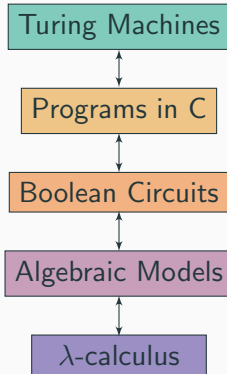
## Example: the addition



## Example: the addition



# Computational Power: model equivalence



## Why algebraic characterisation?

$$[f_1, f_2, \dots, f_k; o_1, o_2, \dots, o_n]$$

- Allows a more denotational point of view on complexity classes

## Why algebraic characterisation?

$$[f_1, f_2, \dots, f_k; o_1, o_2, \dots, o_n]$$

- Allows a more denotational point of view on complexity classes
- This is already done in computability:

$$[0, \mathbf{s}, \pi; \textit{composition}, \textit{minimisation}, \textit{primitive recursion}]$$

## On our example

Effective Church-Turing thesis: Every effectively and *reasonable* models of computation yields to the same complexity classes

→ Notions of time and space **do not depend on the model**, up to some polynomial

→ Verification is hard

## On our example

Effective Church-Turing thesis: Every effectively and *reasonable* models of computation yields to the same complexity classes

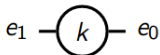
→ Notions of time and space **do not depend on the model**, up to some polynomial

→ Verification is hard

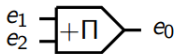
What about continuous-time models?

# The General Purpose Analog Computer (GPAC)

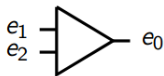
- Introduced by Shannon (1941);
- Mathematical model of the *Differential Analyzer*;
- Model based on circuits, with several interconnected units doing basic operations
- Corresponds to polynomial ODEs:  $y' = p(y)$



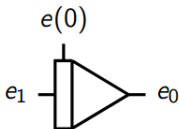
**constant:**  $e_0 = ke_1$



**product:**  $e_0 = e_1 e_2$

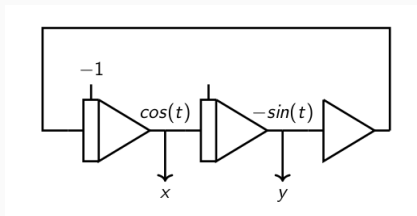


**summer:**  $e_0 = -(e_1 + e_2)$



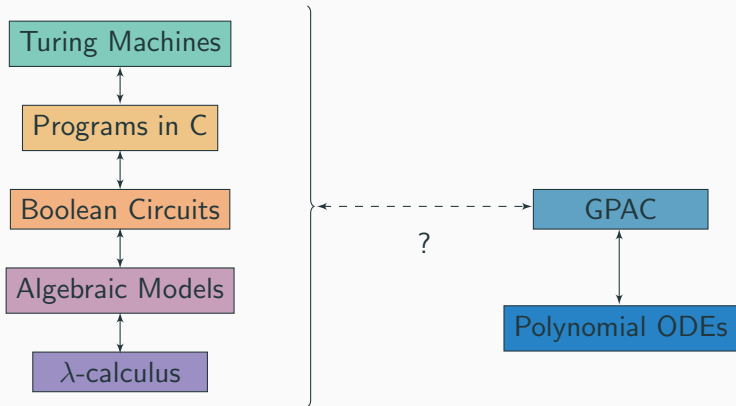
**integrator:**  
 $e_0 = -\int_0^t (e_1(u) du + e(0))$

## GPAC: how does it work?



$$\begin{cases} x'(t) = y(t) \\ y'(t) = -x(t) \\ x(0) = -1 \\ y(0) = 0 \end{cases} \Rightarrow \begin{cases} x(t) = \cos(t) \\ y(t) = -\sin(t) \end{cases}$$

# Computability: GPAC



## Is verification simpler for continuous-time systems?

- Difficulty: **simulating** Turing machines with continuous settings (e.g. Graça, Rojas) with
  - generalised shifts (Moore)
  - recurrent neural networks (Siegelmann, Sontag)
  - analytic functions (Graça, Buescu, Campagnolo)
  - hydrostatic equations in Beltrami fields (Cardona, Miranda, Peralta-Salas)
  - 2d continuous PAMs (Koiran, Cosnard, Garzon)
  - ...

## Solving continuous ODEs

$$\mathbf{f}(0, \mathbf{x}) = \mathbf{g}(\mathbf{x}) \quad \text{and} \quad \frac{\partial \mathbf{f}(t, \mathbf{x})}{\partial t} = \mathbf{u}(\mathbf{f}(t, \mathbf{x}), t, \mathbf{x}),$$

- Non-computable in the *general case*, for  $\mathbf{u}$  computable (Pour-El, Richards)

## Solving continuous ODEs

$$\mathbf{f}(0, \mathbf{x}) = \mathbf{g}(\mathbf{x}) \quad \text{and} \quad \frac{\partial \mathbf{f}(t, \mathbf{x})}{\partial t} = \mathbf{u}(\mathbf{f}(t, \mathbf{x}), t, \mathbf{x}),$$

- Non-computable in the **general case**, for  $\mathbf{u}$  computable (Pour-El, Richards)
- Computable for  $\mathbf{u}$  **computable** and unique solution (Collins, Graça, Ruohonen)

## Solving continuous ODEs

$$\mathbf{f}(0, \mathbf{x}) = \mathbf{g}(\mathbf{x}) \quad \text{and} \quad \frac{\partial \mathbf{f}(t, \mathbf{x})}{\partial t} = \mathbf{u}(\mathbf{f}(t, \mathbf{x}), t, \mathbf{x}),$$

- Non-computable in the **general case**, for  $\mathbf{u}$  computable (Pour-El, Richards)
- Computable for  $\mathbf{u}$  **computable** and unique solution (Collins, Graça, Ruohonen)
- FPSPACE-completeness on a **bounded domain** for  $\mathbf{u}$  computable in polynomial-time (Kawamura, Ko)

But now, how do we define the complexity?

- Measuring the complexity so it corresponds to the digital world

## **A bit of technical details**

---

- Time?:
  - Time = Length
  - Continuous ODEs : Bournez, Graça, Pouly
  - Discrete ODEs : Bournez, Durand, Kontinen, Antonelli, ...

- Time?:
    - Time = Length
    - Continuous ODEs : Bournez, Graça, Pouly
    - Discrete ODEs : Bournez, Durand, Kontinen, Antonelli, ...
  - Space?:
    - Effective Church thesis for probabilistic systems: Rojas
    - A characterisation of PSPACE: Gozzi, Graça
- **Our approach: Space = Precision**

# Models and basic definitions

---

- Mathematical framework: **Computable Analysis**
- Complexity framework: **Algebraic Models**

# Time and Space as algebras

---

# Algebraic characterisations

	With Discrete ODEs	With ODEs
FPTIME for $\mathbb{N}^{\mathbb{N}}$	(Bournez, Durand)	
FPTIME for $\mathbb{R}^{\mathbb{N}}$	$\overline{\text{LDL}}^{\bullet}$	
FPTIME for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{LDL}}^{\circ}$	(Bournez, Graça, Pouly)
FPSPACE for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{RLD}}^{\circ}$	$\overline{\text{RCD}}$

# Algebraic characterisations

	With Discrete ODEs	With ODEs
FPTIME for $\mathbb{N}^{\mathbb{N}}$	(Bournez, Durand)	
FPTIME for $\mathbb{R}^{\mathbb{N}}$	$\overline{\text{LDL}}^{\bullet}$	
FPTIME for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{LDL}}^{\circ}$	(Bournez, Graça, Pouly)
FPSPACE for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{RLD}}^{\circ}$	$\overline{\text{RCD}}$

- $\overline{\text{LDL}}^{\bullet}$ ,  $\overline{\text{LDL}}^{\circ}$  based on Linear Length ODEs;
- $\overline{\text{RLD}}^{\circ}$  based on Robust Linear Discrete ODEs;
- $\overline{\text{RCD}}$  based on Robust Continuous ODEs.

# Algebraic characterisation of PTIME for sequences

We consider:

$$\overline{\text{LDL}}^\bullet = [\mathbf{0}, \mathbf{1}, \pi_i^k, \ell(x), +, -, \times, \overline{\text{cond}}(x), \frac{x}{2};$$

*composition, linear length ODE, ELim]*

**Theorem (B., Bournez, MCU22 Best Student Paper Award)**

$$\text{FPTime} \cap \mathbb{R}^{\mathbb{N}} = \overline{\text{LDL}}^\bullet \cap \mathbb{R}^{\mathbb{N}}$$

# FPTIME for computable functions over the reals

We consider:

$$\overline{\text{LDL}}^\circ = [\mathbf{0}, \mathbf{1}, \pi_i^k, \ell(x), +, -, \tanh, \frac{x}{2}, \frac{x}{3};$$

*composition, linear length ODE, ELim]*

**Theorem (B., Bournez, MFCS23 Best Paper Award)**

$$\overline{\text{LDL}}^\circ \cap \mathbb{R}^{\mathbb{R}} = \text{FPTIME} \cap \mathbb{R}^{\mathbb{R}}$$

# FPSPACE for computable functions over the reals

We consider:

$$\overline{\text{RLD}}^\circ = [\mathbf{0}, \mathbf{1}, \pi_i^k, \ell(n), +, -, \tanh, \frac{x}{2}, \frac{x}{3};$$

*composition, robust linear ODE, ELim]*

**Theorem (B., Bournez, MFCS23 Best Paper Award)**

$$\overline{\text{RLD}}^\circ \cap \mathbb{R}^{\mathbb{R}} = \text{FPSPACE} \cap \mathbb{R}^{\mathbb{R}}$$

# Characterisation of PSPACE with continuous ODEs

We consider :

$$\overline{\text{RCD}} = [\mathbf{0}, \mathbf{1}, \pi_i^k, +, -, \times, \tanh, \cos, \pi, \frac{x}{2}, \frac{x}{3};$$

*composition, robust continuous ODE, ELim]*

**Theorem (5.2.2)**

$$\overline{\text{RCD}} \cap \mathbb{R}^{\mathbb{R}} = \text{FPSPACE} \cap \mathbb{R}^{\mathbb{R}}$$

## Conclusion

---

# Conclusion

	With Discrete ODEs	With ODEs
FPTIME for $\mathbb{N}^{\mathbb{N}}$	(Bournez, Durand)	
FPTIME for $\mathbb{R}^{\mathbb{N}}$	$\overline{\text{LDL}}^{\bullet}$	
FPTIME for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{LDL}}^{\circ}$	(Bournez, Graça, Pouly)
FPSPACE for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{RLD}}^{\circ}$	$\overline{\text{RCD}}$

- We extended the motto **time = length** to discrete ODEs;

# Conclusion

	With Discrete ODEs	With ODEs
FPTIME for $\mathbb{N}^{\mathbb{N}}$	(Bournez, Durand)	
FPTIME for $\mathbb{R}^{\mathbb{N}}$	$\overline{\text{LDL}}^{\bullet}$	
FPTIME for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{LDL}}^{\circ}$	(Bournez, Graça, Pouly)
FPSPACE for $\mathbb{R}^{\mathbb{R}}$	$\overline{\text{RLD}}^{\circ}$	$\overline{\text{RCD}}$

- We extended the motto **time = length** to discrete ODEs;
- We proved that **space = precision**;  
→ proper notions of costs for analogue models of computation;

## Conclusion: Perspectives

- Minimising the algebras?
- Would it work for GPAC and polynomial ODEs?
- Can we relate this approach to second-order complexity?
- Can we relate more thinly our notions of robustness?

## Conclusion: Perspective

- Studying non-determinism, and polynomial hierarchy;
- Algebraically characterising probabilistic classes;
- Studying the complexity for more general differential equations, e.g. distributions.