Deep Learning for Integro-Differential Modelling

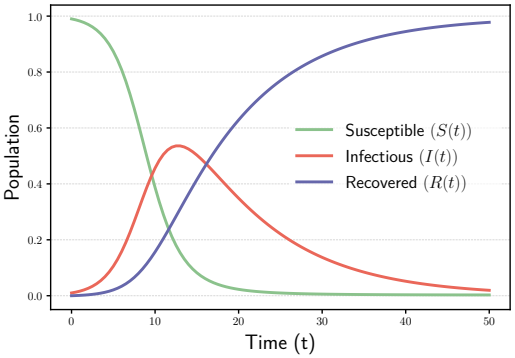
SCML-2026, Hagenberg, Austria

Louis Roussel and François Lemaire - Laboratoire CRIStAL, Université de Lille, France

July 7, 2026

Introduction

Modelling an epidemic - SIR Model



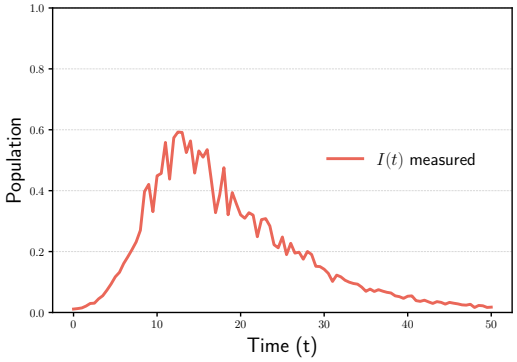
$$\Sigma : \begin{cases} \dot{S}(t) = -\beta I(t)S(t) \\ \dot{I}(t) = \beta I(t)S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

β transmission rate

γ recovery rate

- Many physical and biological phenomena can be modelled using differential equation models.
  $\rightarrow$ Models often have parameters (i.e. numerical constants) which values are unknowns.

Parameter Estimation: a widely studied problem



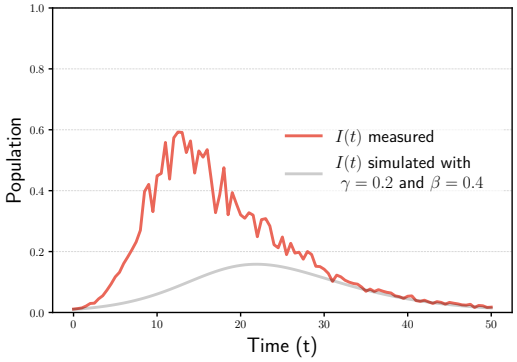
$$\Sigma : \begin{cases} \dot{S}(t) = -\beta I(t) S(t) \\ \dot{I}(t) = \beta I(t) S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

β transmission rate

γ recovery rate

From noisy measurements on $I(t)$, how to estimate the parameter values of β and γ?

Parameter Estimation: a widely studied problem



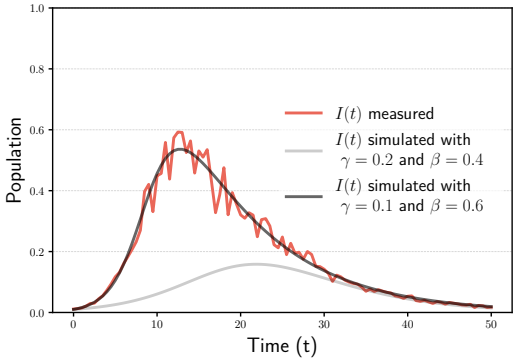
$$\Sigma : \begin{cases} \dot{S}(t) = -\beta I(t) S(t) \\ \dot{I}(t) = \beta I(t) S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

β transmission rate

γ recovery rate

From noisy measurements on $I(t)$, how to estimate the parameter values of β and γ?
$\rightarrow$ find the parameter values that make the simulations fit the measurements

Parameter Estimation: a widely studied problem



$$\Sigma : \begin{cases} \dot{S}(t) = -\beta I(t) S(t) \\ \dot{I}(t) = \beta I(t) S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

β transmission rate

γ recovery rate

From noisy measurements on $I(t)$, how to estimate the parameter values of β and γ?

$\rightarrow$ find the parameter values that make the simulations fit the measurements

*$\rightarrow$ the simulated black curve **best fits** the measurements*

An approach to parameter estimation: Eliminate to Estimate

$$\Sigma \quad : \begin{cases} \dot{S}(t) = -\boldsymbol{\beta} I(t) S(t) \\ \dot{I}(t) = \boldsymbol{\beta} I(t) S(t) - \boldsymbol{\gamma} I(t) \\ \dot{R}(t) = \boldsymbol{\gamma} I(t) \end{cases}$$

An approach to parameter estimation: Eliminate to Estimate

$$\Sigma \quad : \begin{cases} \dot{S}(t) = -\beta I(t) S(t) \\ \dot{I}(t) = \beta I(t) S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

$\downarrow$ *elimination of* $S(t)$ *and* $R(t)$

- $\gamma\beta I(t)^3 + \beta I(t)^2 \dot{I}(t) + I(t)\ddot{I}(t) - \dot{I}(t)^2 = 0$
- $\beta\gamma \int I(t) \int I(t) + \beta \int I(t)^2 + (\gamma - \beta I(0) - \beta S(0)) \int I(t) - I(0) + I(t) = 0$

An approach to parameter estimation: Eliminate to Estimate

$$\Sigma \quad : \begin{cases} \dot{S}(t) = -\beta I(t) S(t) \\ \dot{I}(t) = \beta I(t) S(t) - \gamma I(t) \\ \dot{R}(t) = \gamma I(t) \end{cases}$$

$\downarrow$ *elimination of* $S(t)$ *and* $R(t)$

- $\gamma\beta I(t)^3 + \beta I(t)^2 \dot{I}(t) + I(t)\ddot{I}(t) - \dot{I}(t)^2 = 0$
- $\beta\gamma \int I(t) \int I(t) + \beta \int I(t)^2 + (\gamma - \beta I(0) - \beta S(0)) \int I(t) - I(0) + I(t) = 0$

Input/Output (I/O) equations [Fliess 1989]

Equations that **only** contain the parameters (β and γ) and measured quantities ($I(t)$).
$\rightarrow$ $S(t)$ *and* $R(t)$ *has been* ***eliminated***
$\rightarrow$ I/O equations can be used for parameter estimation.

This talk: Compute integral I/O equations with Deep Learning

Why consider integral I/O equations?

Integral I/O equations:

- tend to be smaller,
- tend to offer a larger variety of possible equations,
- tend to be less sensitive to noisy measurements,
- naturally encodes the initial conditions.
  $\rightarrow e^{-t}$ is coded by $u(t) = 1 - \int_0^t u(\tau)d\tau$ (simply $u = 1 - \int u$)

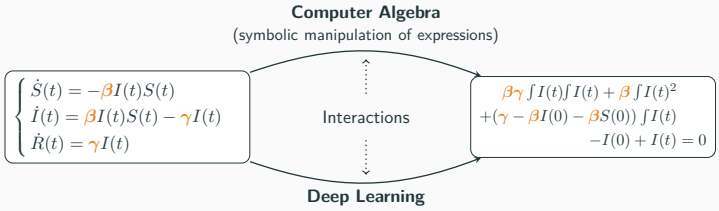
Problems:

- they are difficult to obtain/manipulate,
- they may involve exponentials, logarithms, or fractions,
  $\rightarrow$ whereas differential I/O equations are polynomials
- few algorithms exist to compute them.

Summary

Deep Learning for Integro-Differential Modelling

A hybrid approach



This talk

Compute integral I/O equations from a given differential system using deep learning.

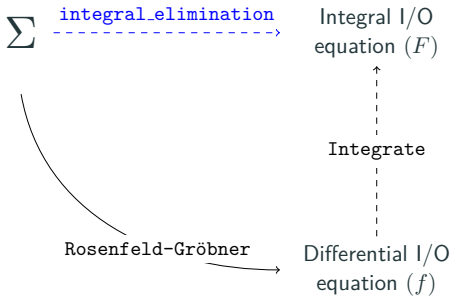
Why?

Use deep learning to guide theoritical improvements

Background

The integral_elimination Algorithm

The integral_elimination algorithm is the **first** algorithm that performs integral elimination for systems of nonlinear ordinary differential equations.



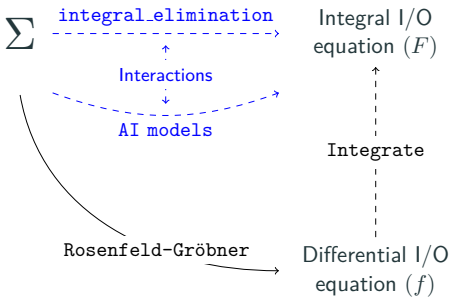
- Written in Python using SymPy
  $\rightarrow$ Around 2500 lines of code
- Non-trivial examples solved:

  18 terms (int. equation)

  vs.

  232 145 terms (diff. equation)

So why using deep learning?



Problem:

The `integral_elimination` algorithm is **partial** and **very difficult** to improve
→ it does not handle **fractions**!

Deep learning models (from Σ to F):

- to guide the improvement of `integral_elimination`
- to allow "fast" experimentations

→ We restricted ourselves to systems of 2 equations

This talk: present examples solved using trained models for integral elimination

Experimental Setup

Experimental Setup (1/2)

Disclaimer

I will not talk about the experimental parameters (generation, training, ...)

SCML Paper

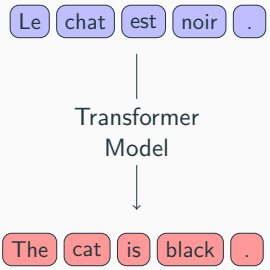
François Lemaire and Louis Roussel [Mar. 2026]. "Deep Learning for Integro-Differential Modelling". In: *RISC Proceedings on Symbolic Computation and Machine Learning, Johannes Kepler Universität Linz, 2026*. 2. Hagenberg Castle, Austria. DOI: 10.35011/risc-proceedings-scml.2. URL: https://hal.science/hal-05230281

Source Code

https://codeberg.org/louis-roussel/IntegroDifferentialDeepLearning

Experimental Setup (2/2) - The Transformer Architecture [Vaswani et al. 2017]

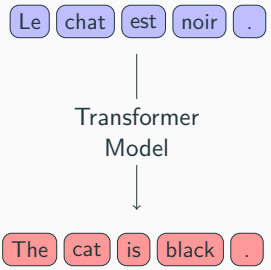
For Translation



Sentences are list of words (tokens).

Experimental Setup (2/2) - The Transformer Architecture [Vaswani et al. 2017]

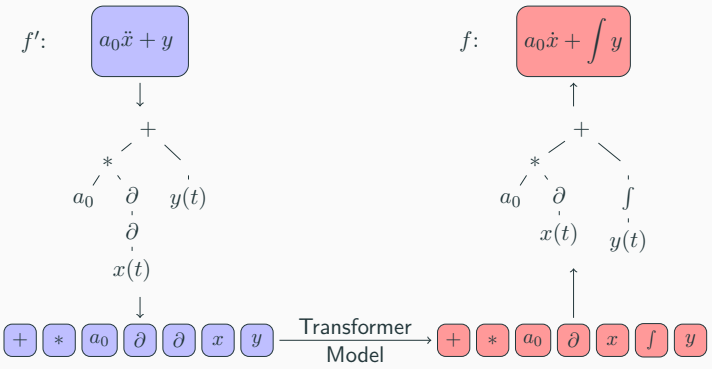
For Translation



Sentences are list of words (tokens).

For mathematical tasks [Lample and Charton 2020]

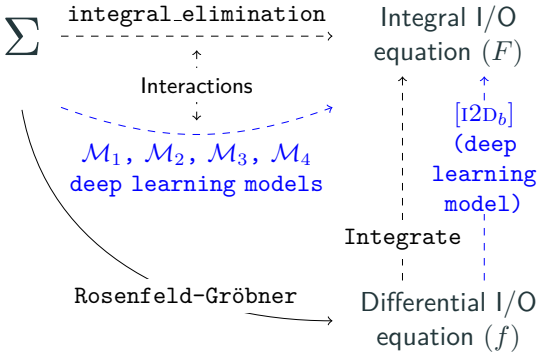
Integration is seen as a text translation.



To train the model, pairs (f', f) are needed.

Trained models

5 trained deep learning models



Deep learning models deal with **fractions**, exponentials and logarithms!
$\rightarrow$ `integral_elimination` **cannot** deal with **fractions**

How did we train our deep learning models (from Σ to F)?

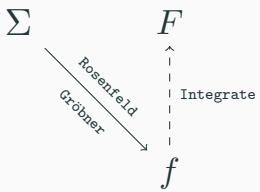
Data are needed to train deep learning models. Three ways to produce pairs (Σ, F):

$$\Sigma \xrightarrow[\text{elimination}]{\text{integral}} F$$

How did we train our deep learning models (from Σ to F)?

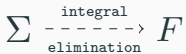
Data are needed to train deep learning models. Three ways to produce pairs (Σ, F):

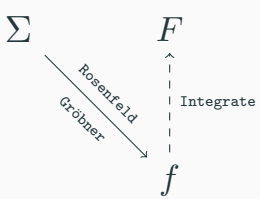
$$\Sigma \xrightarrow[\text{elimination}]{\text{integral}} F$$

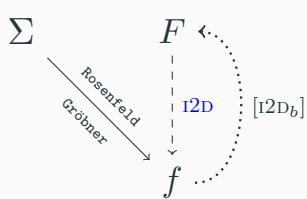


How did we train our deep learning models (from Σ to F)?

Data are needed to train deep learning models. Three ways to produce pairs (Σ, F):

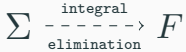


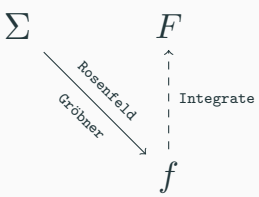


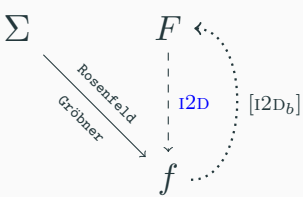


How did we train our deep learning models (from Σ to F)?

Data are needed to train deep learning models. Three ways to produce pairs (Σ, F):







• I2D (IntToDiff) is a partial algorithm that converts an integral expression F into a differential expression f by removing integrals, exponentials, logarithms and **fractions**.

$$F = y^2 - y + y\int y + \ln(y) + \int\left(\frac{\int(y)}{y}\right) + \iint y^2$$

I2D →

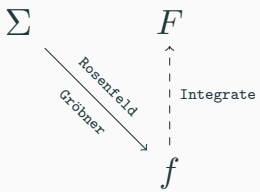
← [I2Db]

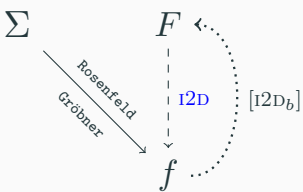
$$\begin{aligned} f = &- y^5\,\ddot{y} + 2y^4\dot{y}\ddot{y} - 3y^4\dot{y}\,\dddot{y} + 4y^4\ddot{y}^2 + 3y^3\dot{y}^2\ddot{y} - 2y^3\dot{y}^2\,\dddot{y} \\ &+ 6y^3\dot{y}\ddot{y}^2 + y^3\ddot{y} - y^3\,\dddot{y} - 4y^2\dot{y}^2 - 2y^2\dot{y}\ddot{y} - 2y^2\dot{y}\,\dddot{y} \\ &+ 2y^2\ddot{y}^2 - 9y\dot{y}^3 - 8y\dot{y}^2\ddot{y} + y\dot{y}^2\,\dddot{y} - 3y\dot{y}\ddot{y}^2 + y\dot{y}\,\dddot{y} \\ &- y\ddot{y}^2 + y\ddot{y} - 4\dot{y}^4 + 2\dot{y}^3\ddot{y} + 2\dot{y}^2\ddot{y} - \dot{y}^2 - \dot{y}\,\dddot{y} + \ddot{y}^2 \end{aligned}$$

How did we train our deep learning models (from Σ to F)?

Data are needed to train deep learning models. Three ways to produce pairs (Σ, F):

$$\Sigma \xrightarrow[\text{elimination}]{\text{integral}} F$$





A significant computational cost

- Generating the three datasets $= 50$ computers running for three days
- In total, considering all our deep learning experiments, we performed a lot of trainings (more than 30)
  $\rightarrow 1$ training $= 1$ day using an NVIDIA A100-40 GB
- Estimated electricity consumption: 1000 kWh ≈ 50 kgCO_2e (in France) $\approx$ Lille $\leftrightarrow$ Paris

I2D Example

We summarise the main steps of Algorithm I2D applied to

$$y^2 - y + y\int y + \ln(y) + \underline{\int(\frac{\int y}{y})} + \int\int y^2$$

$$\downarrow \frac{d}{dt} \text{ then take the numerator}$$

$$(y\dot{y}+1)\int y + \underline{y\int y^2} + \dot{y} + y^3 - y\dot{y} + 2y^2\dot{y}$$

$$\downarrow \times\frac{1}{y} \text{ then } \frac{d}{dt} \text{ then take the numerator}$$

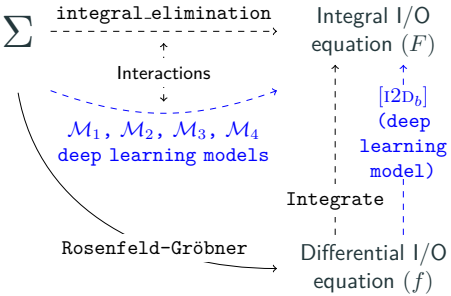
$$y^4 + 3y^3\dot{y} + 2y^3\ddot{y} + \underline{(y^2\ddot{y} - \dot{y})\int y} - y^2\ddot{y} + y^2 + y\ddot{y} - \dot{y}^2 + 2y^2\dot{y}^2$$

$$\downarrow \times\frac{1}{y^2\ddot{y}-\dot{y}} \text{ then } \frac{d}{dt} \text{ then take the numerator}$$

$$-y^5\dddot{y} + 2y^4\dot{y}\ddot{y} - 3y^4\dot{y}\dddot{y} + 4y^4\ddot{y}^2 + 3y^3\dot{y}^2\ddot{y} - 2y^3\dot{y}^2\dddot{y} + 6y^3\dot{y}\ddot{y}^2 + y^3\ddot{y} - y^3\dddot{y} - 4y^2\dot{y}^2 - 2y^2\dot{y}\ddot{y} - 2y^2\dot{y}\dddot{y} + 2y^2\ddot{y}^2 - 9y\dot{y}^3 - 8y\dot{y}^2\ddot{y} + y\dot{y}^2\dddot{y} - 3y\dot{y}\ddot{y}^2 + y\dot{y}\dddot{y} - y\ddot{y}^2 + y\ddot{y} - 4\dot{y}^4 + 2\dot{y}^3\ddot{y} + 2\dot{y}^2\ddot{y} - \dot{y}^2 - \dot{y}\dddot{y} + \ddot{y}^2.$$

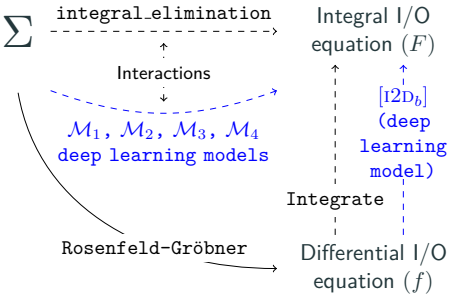
Results

5 trained deep learning models



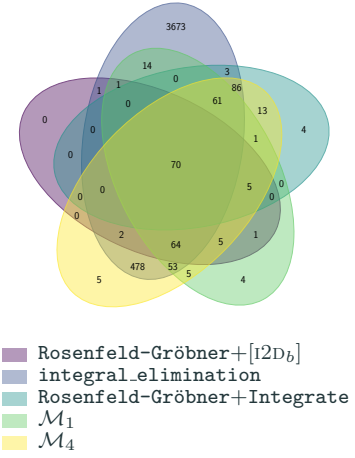
Deep learning models deal with **fractions**, exponentials and logarithms!
$\rightarrow$ `integral_elimination` **cannot** deal with **fractions**

5 trained deep learning models



Deep learning models deal with **fractions**, exponentials and logarithms!
→ `integral_elimination` **cannot** deal with **fractions**

Number of systems (out of 10k) where an integral I/O equation was successfully obtained.



Difficult examples with fractions!

Two examples where Deep Learning models succeeds,
whereas `integral_elimination` fails.

$$\Sigma \begin{cases} \dot{x} = x^3 \\ \dot{y} = 2x - y \end{cases}$$

$$f = 4\ddot{y} - \dot{y}^3 - 3y\dot{y}^2 - 3\dot{y}y^2 + 4\dot{y} - y^3$$

$$\hat{F} = y + \int y + 4 \int \frac{1}{y + \int y} \quad \text{(Model output)}$$

$$\Sigma \begin{cases} \dot{x} = 4x^2y - 2x^3 \\ \dot{y} = y^2 - xy \end{cases}$$

$$f = 2y^6 - 2y^4\dot{y} - y^3\dot{y} - 2y^2\dot{y}^2 + y^2\ddot{y} - y\dot{y}^2 + 2\dot{y}^3$$

$$\hat{F} = -2y + \int \frac{y}{\ln(y) + \int y} + 2 \int y^2 \quad \text{(Model output)}$$

Question

$\rightarrow$ How to handle **fractions** in a future version of `integral_elimination`?

Difficult examples with fractions!

Two examples where Deep Learning models succeeds,
whereas `integral_elimination` fails.

$$\Sigma \begin{cases} \dot{x} = x^3 \\ \dot{y} = 2x - y \end{cases}$$

$$f = 4\ddot{y} - \dot{y}^3 - 3y\dot{y}^2 - 3\dot{y}y^2 + 4\dot{y} - y^3$$

$$\hat{F} = y + \int y + 4\int \frac{1}{y + \int y} \quad \text{(Model output)}$$

$$F_{ics} = y - y_0 + \int y + 4x_0 \int \frac{1}{x_0 \int y + x_0 y - x_0 y_0 - 2}$$

$$\Sigma \begin{cases} \dot{x} = 4x^2 y - 2x^3 \\ \dot{y} = y^2 - xy \end{cases}$$

$$f = 2y^6 - 2y^4\dot{y} - y^3\dot{y} - 2y^2\dot{y}^2 + y^2\ddot{y} - y\dot{y}^2 + 2\dot{y}^3$$

$$\hat{F} = -2y + \int \frac{y}{\ln(y) + \int y} + 2\int y^2 \quad \text{(Model output)}$$

$$F_{ics} = 2y_0 - 2y - 2x_0 y_0 \int \frac{y}{y_0 - 2x_0 y_0 (\ln(\frac{y}{y_0}) + \int y)} + 2\int y^2$$

Question

→ How to handle **fractions** in a future version of `integral_elimination`?

How to study parameter estimation? (1/2)

$$\Sigma \begin{cases} \dot{x} = x^3 \\ \dot{y} = 2x - y \end{cases}$$

$$f = 4\ddot{y} - \dot{y}^3 - 3y\dot{y}^2 - 3\dot{y}y^2 + 4\dot{y} - y^3$$

$$\hat{F} = y + \int y + 4 \int \frac{1}{y + \int y} \quad \text{(Model output)}$$

$$F_{ics} = y - y_0 + \int y + 4x_0 \int \frac{1}{x_0 \int y + x_0 y - x_0 y_0 - 2}$$

Recent Work (CASC'26)

- detailled computations for F_{ics}
- introduction of parameters in Σ
- practical use of equations with fractions for parameter estimation
- study of the identifiability using integral equations

How to study parameter estimation? (2/2)

$$\Sigma_\theta \begin{cases} \dot{x} = \alpha x^3 \\ \dot{y} = 2x - \beta y \end{cases}$$

$$f_\theta = 4\ddot{y} - \alpha\dot{y}^3 - 3\alpha\beta\dot{y}^2 y - 3\alpha\beta^2\dot{y}y^2 + 4\beta\dot{y} - \alpha\beta^3 y^3$$

$$F_\theta = y - y_0 + \beta\int y + 4\int \frac{1}{\alpha\beta\int y + \alpha(y - y_0) - \frac{2}{x_0}}$$

1. start from the intial guess obtained by our models
   $\hat{F} = y + \int y + 4\int \frac{1}{y + \int y}$
2. introduce initial conditions (apply I2D and integrate)
   $F_{ics} = y - y_0 + \int y + 4\int \frac{1}{y - y_0 + \int y - \frac{2}{x_0}}$
3. introduces parameters
   $F_\theta = y - y_0 + \beta\int y + 4\int \frac{1}{\alpha\beta\int y + \alpha(y - y_0) - \frac{2}{x_0}}$

Open Question

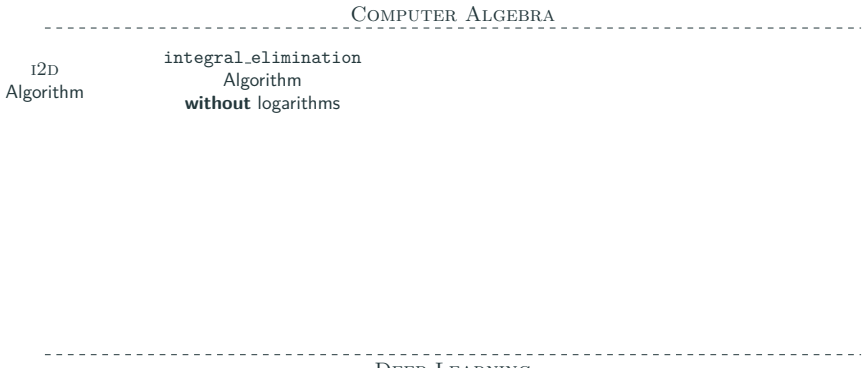
How to compute F_θ using elimination techniques ?

Interactions between Computer Algebra and Deep Learning

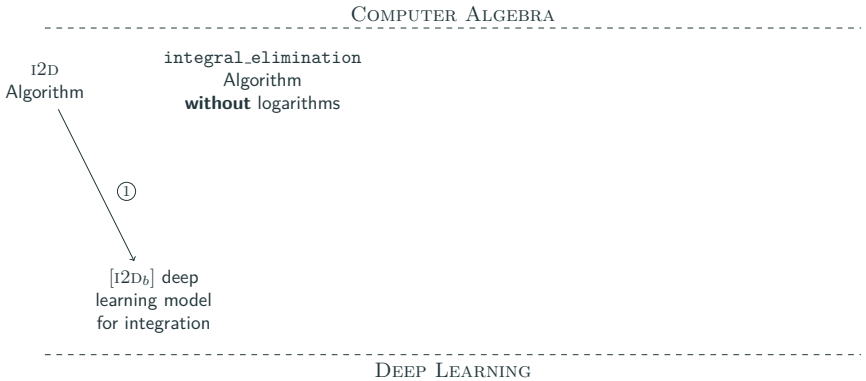
Computer Algebra

Deep Learning

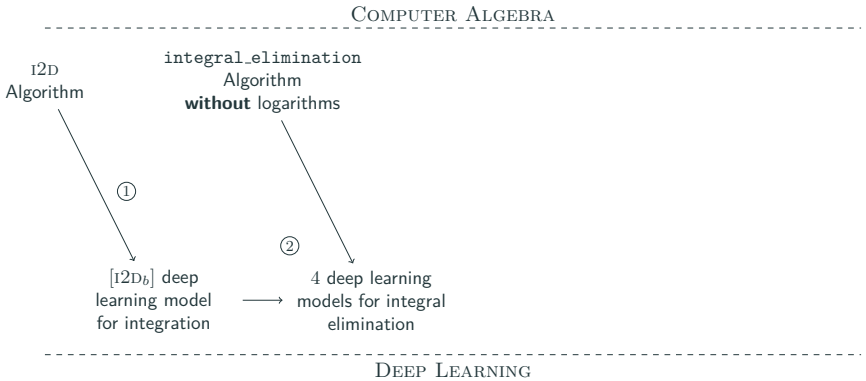
Interactions between Computer Algebra and Deep Learning



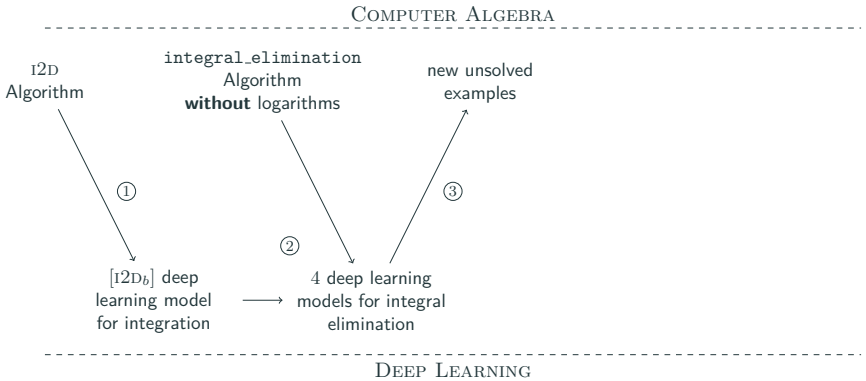
Interactions between Computer Algebra and Deep Learning



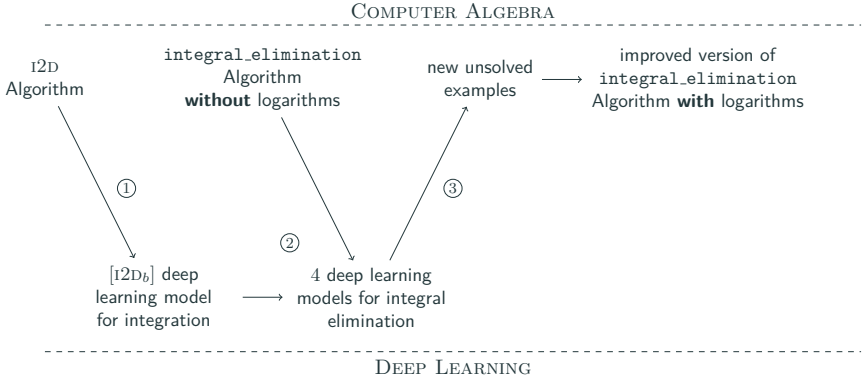
Interactions between Computer Algebra and Deep Learning



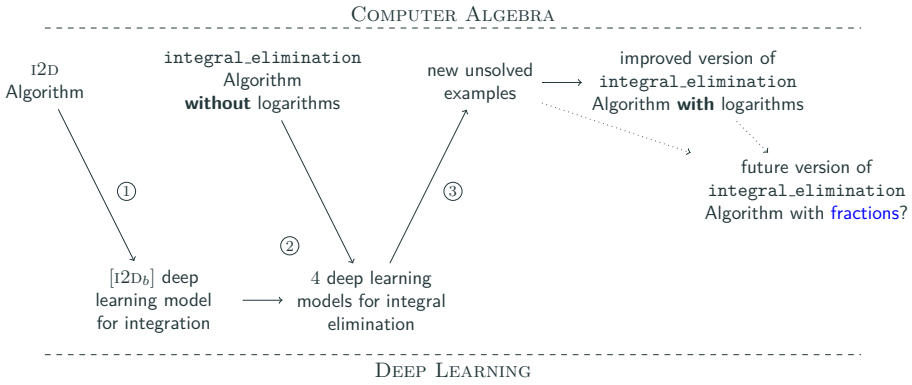
Interactions between Computer Algebra and Deep Learning



Interactions between Computer Algebra and Deep Learning



Interactions between Computer Algebra and Deep Learning



Conclusion

Conclusion

Main Results

- Five trained models to compute integral I/O equations
- The I2D Algorithm
- Accuracy of the `integral_elimination` Algorithm on a test dataset:
  - Before 41,09% (no logarithms)
  - After 45,06% (with logarithms introduced thanks to this experiment)

Paper and Source Code

- "Deep Learning for Integro-Differential Modelling" – SCML'26 proceedings
- https://codeberg.org/louis-roussel/IntegroDifferentialDeepLearning

Future work:

- Deal with systems with more than 2 equations
- Consider other deep learning architecture (e.g. TreeTransformer)

Thank you for your attention!

References

References i

Fliess, Michel (Jan. 1989). "Automatique et corps différentiels". In: *Forum Mathematicum - FORUM MATH* 1, pp. 227–238. DOI: 10.1515/form.1989.1.227.

Lample, Guillaume and François Charton (2020). "Deep Learning for Symbolic Mathematics". In: *The Ninth International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=S1eZYeHFDS.

Lemaire, François and Louis Roussel (Mar. 2026). "Deep Learning for Integro-Differential Modelling". In: *RISC Proceedings on Symbolic Computation and Machine Learning, Johannes Kepler Universität Linz, 2026*. 2. Hagenberg Castle, Austria. DOI: 10.35011/risc-proceedings-scml.2. URL: https://hal.science/hal-05230281.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Appendix

Detailled Results

	$\mathcal{M}_2$			$\mathcal{M}_3$			$\mathcal{M}_1$			$\mathcal{M}_4$			Rosenfeld-Gröbner+Integrate	Rosenfeld-Gröbner+$[\mathrm{I2D_b}]$	integral_elimination
	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10			
$\mathrm{Test}_{\texttt{Rosenfeld-Gröbner+Integrate}}$	89,60	90,30	90,30	2,40	30,7	37,20	0,00	61,00	54,60	73,30	88,00	91,40	100,00	23,90	76,40
$\mathrm{Test}_{\texttt{Rosenfeld-Gröbner+}[\mathrm{I2D_b}]}$	1,80	43,10	42,20	83,40	88,20	87,40	0,00	41,40	43,10	66,00	88,00	95,7	46,60	100,00	83,10
$\mathrm{Test}_{\texttt{integral\_elimination}}$	0,00	17,20	17,40	0,00	13,20	19,30	73,70	76,70	53,00	49,10	70,90	76,10	17,80	8,70	100,00
$\mathrm{Test}_{\mathrm{mix}}$	30,46	50,33	50,90	28,60	43,80	48,00	24,56	60,30	50,93	62,79	82,50	87,70	54,80	44,20	86,50
$\mathrm{Test}_{\mathrm{Unlabelled}}$		2,32	2,34		2,07	2,55		6,27	5,50		6,77	8,04	2,43	1,49	41,09

Table 1: Comparison of the success rate of the 7 approaches to compute an integral I/O equation from a dynamical system on 5 different datasets.

	$\mathcal{M}_2$			$\mathcal{M}_3$			$\mathcal{M}_1$			$\mathcal{M}_4$			Rosenfeld-Gröbner+Integrate	Rosenfeld-Gröbner+$[\mathrm{I2D_b}]$	integral_elimination
	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10	Syntax	Beam1	Beam10			
$\mathrm{Test}_{\mathrm{UnlabeledFiltered}}$		3,20	3,22		2,63	3,24		8,13	7,12		8,73	10,12	3,35	1,92	18,12

Table 2: Comparison of the success rate of the 7 approaches to compute an integral I/O equation from a dynamical system on the $\mathrm{Test}_{\mathrm{UnlabeledFiltered}}$ obtained after removing from $\mathrm{Test}_{\mathrm{Unlabelled}}$ all the pairs (S, F) in $\Sigma \times \mathrm{E}_{\int}$ where F contains more than 200 tokens.

Timeouts

Timeouts account for 16.29% of the errors of the deep learning based approaches (i.e. not `integral_elimination` or `Integrate`).