A tutorial on SciML

What is scientific machine learning and how do you use it?

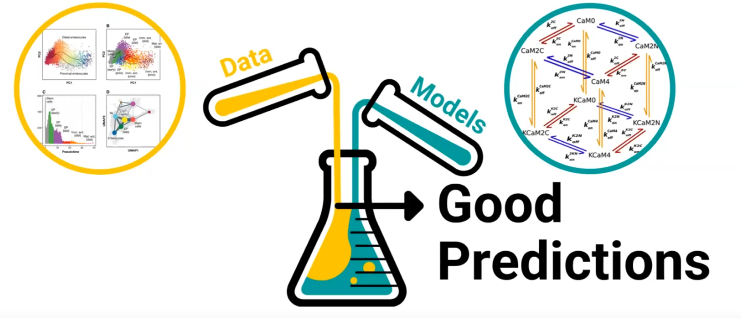
Frederik Baymler Mathiesen

22nd of November, 2024

Delft Center for Systems and Control
Delft University of Technology

TUDelft

What is SciML?



Scientific Computing + Machine Learning[1]

[1] C. Rackauckas (2023). *Generalizing Scientific Machine Learning and Differentiable Simulation Beyond Continuous Models*. presentation at The Alan Turing Institute

Why?

Why not!

- Improved accuracy
- Data efficiency
- Physical consistency
- Better generalization
- Interpretability
- Uncertainty quantification
- From extrapolation to interpolation
- Cheaper surrogate models

Agenda

a. Approaches (10 min)
b. Practical example in Julia
   1. Quick intro to Julia (10 min)
   2. Lotka-Voltera SciML example (35 min)
   3. Controlled bouncing ball (15 min)
c. Challenges and opportunities (15 min)
d. Questions (? min)

01

Approaches

Physics-Informed Neural Network

Partial differential equation

$$\frac{\partial u}{\partial t} + \mathcal{N}[u] = 0, \quad t \in [0, T], x \in \Omega$$

subject to

$$u(0, x) = g(x), \quad x \in \Omega \qquad \text{(Initial conditions)}$$
$$\mathcal{B}[u] = 0, \quad t \in [0, T], x \in \Omega \qquad \text{(Boundary conditions)}$$

Physics-Informed Neural Network

Partial differential equation

$$\frac{\partial u}{\partial t} + \mathcal{N}[u] = 0, \quad t \in [0, T], x \in \Omega$$

subject to

$$u(0, x) = g(x), \quad x \in \Omega \qquad \text{(Initial conditions)}$$
$$\mathcal{B}[u] = 0, \quad t \in [0, T], x \in \Omega \qquad \text{(Boundary conditions)}$$

Idea: Let a neural network represent an unknown solution to the PDE.

Physics-Informed Neural Network

If $u_\theta(t, x)$ is a neural network, then PINN optimizes the following objective[2]:

$$\mathcal{L}(\theta) = MSE\left(\frac{\partial u_\theta}{\partial_t}(t^i, x^i) + \mathcal{N}[u_\theta](t^i, x^i)\right) +$$

$$MSE(u_\theta(0, x^i) - g(x^i)) +$$

$$MSE(\mathcal{B}[u_\theta](t^i, x^i))$$

[2] S. Wang, S. Sankaran, H. Wang, and P. Perdikaris (2023). *An Expert's Guide to Training Physics-informed Neural Networks*. arXiv: 2308.08468

Universal Differential Equations[3]

$$\mathcal{N}[u(t), u(\alpha(t)), W(t), U_\theta(u, \beta(t))] = 0$$

[3] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman (2021). *Universal Differential Equations for Scientific Machine Learning*. arXiv: 2001.04385

Universal Differential Equations[3]

$$\mathcal{N}[u(t), u(\alpha(t)), W(t), U_\theta(u, \beta(t))] = 0$$

Specializations

- ODEs
- SDEs
- DDEs
- DAEs
- PDEs

Generalizations

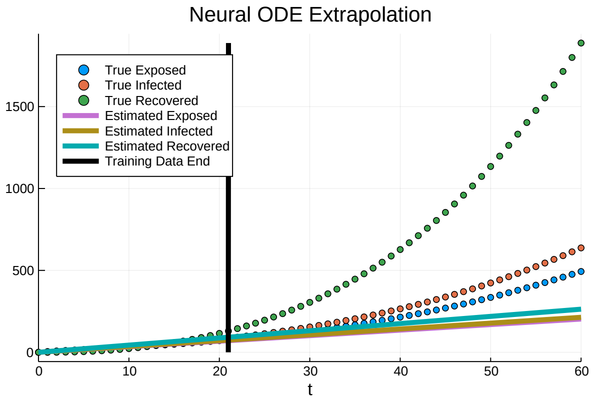
- Hybrid and jump equations
- Uncertainty quantification
- Discrete random processes
- Chaotic systems
- Non-linear mixed effect models

[3] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman (2021). *Universal Differential Equations for Scientific Machine Learning*. arXiv: 2001.04385

Universal ODE: COVID-19 modeling

SEIRHD-model

$$\dot{S} = -\eta(t)S$$
$$\dot{E} = \eta(t)S - \alpha E$$
$$\dot{I} = \alpha E - (\gamma_1 + \delta)I$$
$$\dot{R} = \gamma_1 I + \gamma_2 H$$
$$\dot{H} = \delta I - (\mu + \gamma_2)H$$
$$\dot{D} = \mu H$$

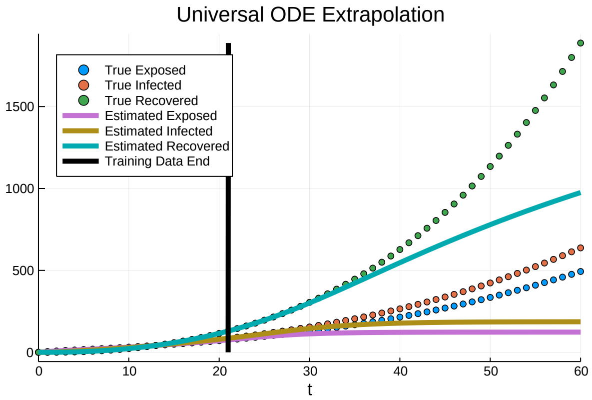


https://github.com/ChrisRackauckas/universal_differential_equations/tree/master/SEIR_exposure

Universal ODE: COVID-19 modeling

SEIRHD-model

$$\dot{S} = -\eta(t)S$$
$$\dot{E} = \eta(t)S - \alpha E$$
$$\dot{I} = \alpha E - (\gamma_1 + \delta)I$$
$$\dot{R} = \gamma_1 I + \gamma_2 H$$
$$\dot{H} = \delta I - (\mu + \gamma_2)H$$
$$\dot{D} = \mu H$$

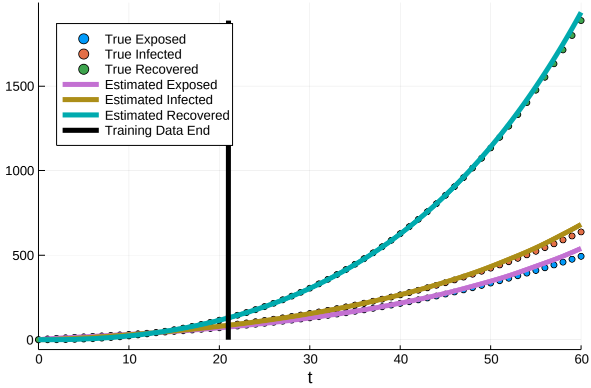


https://github.com/ChrisRackauckas/universal_differential_equations/tree/master/SEIR_exposure

Universal ODE: COVID-19 modeling

SEIRHD-model

$$\dot{S} = -\eta(t)S$$
$$\dot{E} = \eta(t)S - \alpha E$$
$$\dot{I} = \alpha E - (\gamma_1 + \delta)I$$
$$\dot{R} = \gamma_1 I + \gamma_2 H$$
$$\dot{H} = \delta I - (\mu + \gamma_2)H$$
$$\dot{D} = \mu H$$



https://github.com/ChrisRackauckas/universal_differential_equations/tree/master/SEIR_exposure

How do we train UDEs?

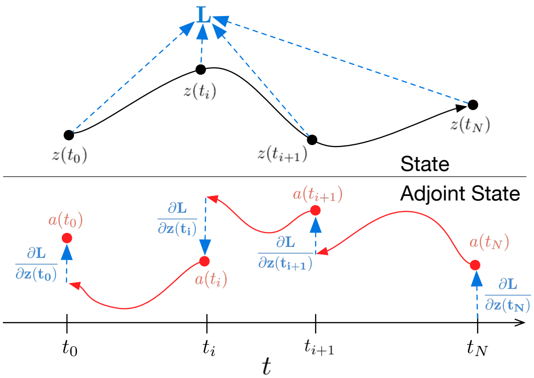
- Using derivative-data directly?
- Differentiable simulators

$$\mathcal{L}(\theta) = \int_0^T \|u_\theta(t) - r(t)\| \, dt \qquad \text{(Distance-based)}$$

Chain-rule: $\frac{d\mathcal{L}}{d\theta} = \frac{\partial \mathcal{L}}{\partial u_\theta(t)} \frac{\partial u_\theta(t)}{\partial \theta}$

Adjoint: $\frac{da(t)}{dt} = -a(t)^\top \frac{\partial f(z(t), t, \theta)}{\partial z}$ where $a(t) = \frac{\partial L}{\partial z(t)}$

Gradients![4] [5]



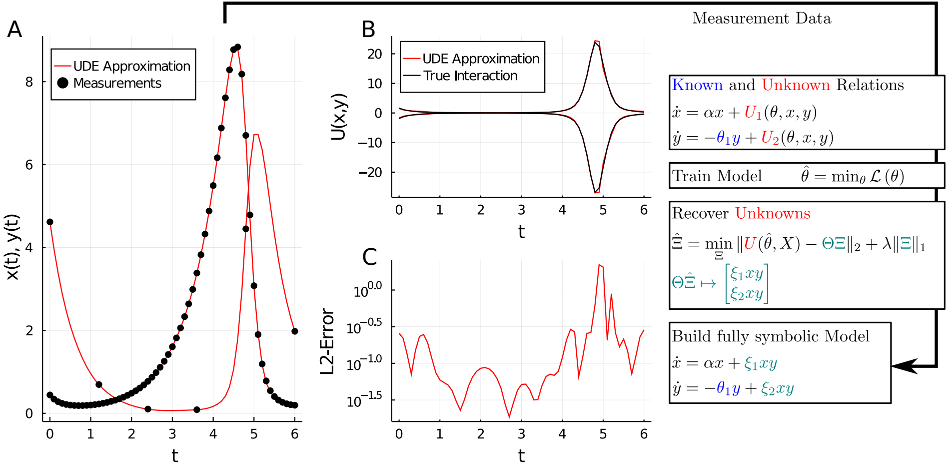
[4] S. Kim, W. Ji, S. Deng, Y. Ma, and C. Rackauckas (2021). "Stiff neural ordinary differential equations". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science*

[5] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud (2019). *Neural Ordinary Differential Equations*. arXiv: 1806.07366

02

Practical example

Process for dynamics discovery



03

Challenges and opportunities

TU Delft

More on gradients! [6]

Adjoint methods

Method	Stability	Memory usage
BacksolveAdjoint	Poor	Low. $O(1)$.
InterpolatingAdjoint	Good	High. Requires full continuous solution of forward.
QuadratureAdjoint	Good	Higher. Requires full continuous solution of forward and Lagrange multiplier.
BacksolveAdjoint (checkpointed)	Okay	Medium. $O(c)$ where c is the number of checkpoints.
InterpolatingAdjoint (checkpointed)	Good	Medium. $O(c)$ where c is the number of checkpoints.
ReverseDiffAdjoint	Best	Highest. Requires full forward and reverse AD of solve.
TrackerAdjoint	Best	Highest. Requires full forward and reverse AD of solve.

[6] Y. Ma, V. Dixit, M. Innes, X. Guo, and C. Rackauckas (2021). *A Comparison of Automatic Differentiation and Continous Sensitivity Analysis for Derivatives of Differential Equation Solutions*. arXiv: 1812.01892

Tricks of the trade: neural networks

Figure: Trouble Shooting Deep Neural Networks - Josh Tobin

Figure: A Recipe for Training Neural Networks - Andrej Karpathy

`http://josh-tobin.com/assets/pdf/troubleshooting-deep-neural-networks-01-19.pdf`

`https://karpathy.github.io/2019/04/25/recipe/`

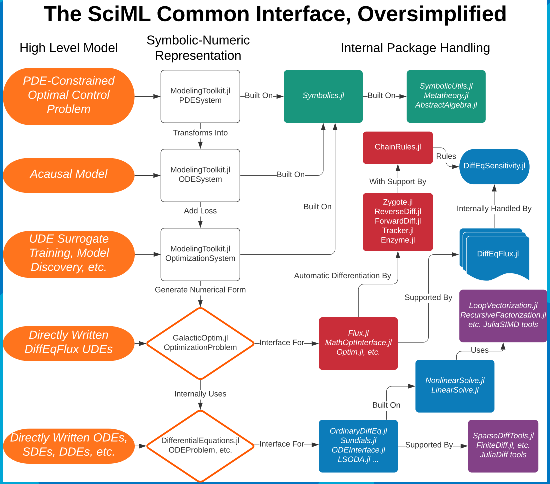
Successful applications

- Learning relativistic physics for binary black hole dynamics
- COVID-19 epidemiological models
- Pharmaceutical modeling
- Battery degradation models
- Combustion models
- Controlling qubits in quantum circuits
- Crash test system modeling
- Williams Formula 1 team!!

Chris Rackauckas: Accurate and Efficient Physics-Informed Learning
Through Differentiable Simulation

More on differentiable programming

LinearSolve.jl	$A(p)x = b$
NonlinearSolve.jl	$f(u, p) = 0$
DifferentialEquations.jl	$\dot{u} = f(u, p, t)$
Integrals.jl	$\int_a^b f(p, t)\, dt$
Optimization.jl	$\min f(u, p)$ s.t. $g(u, p) \leq 0$
(I)MDPs?	$V^{\star}_{M(p)}(s)$



State of Julia's SciML Ecosystem, Chris Rackauckas, JuliaCon 2024

Thank you for your attention

Frederik Baymler Mathiesen

TU Delft

Bibliography I

Chen, R. T. Q., Y. Rubanova, J. Bettencourt, and D. Duvenaud (2019). *Neural Ordinary Differential Equations*. arXiv: 1806.07366.

Kim, S., W. Ji, S. Deng, Y. Ma, and C. Rackauckas (2021). "Stiff neural ordinary differential equations". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science*.

Ma, Y., V. Dixit, M. Innes, X. Guo, and C. Rackauckas (2021). *A Comparison of Automatic Differentiation and Continuous Sensitivity Analysis for Derivatives of Differential Equation Solutions*. arXiv: 1812.01892.

Rackauckas, C. (2023). *Generalizing Scientific Machine Learning and Differentiable Simulation Beyond Continuous Models*. presentation at The Alan Turing Institute.

Rackauckas, C., Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman (2021). *Universal Differential Equations for Scientific Machine Learning*. arXiv: 2001.04385.

Wang, S., S. Sankaran, H. Wang, and P. Perdikaris (2023). *An Expert's Guide to Training Physics-informed Neural Networks*. arXiv: 2308.08468.