POTSDAM INSTITUTE FOR CLIMATE IMPACT RESEARCH

PIK

Leibniz Association

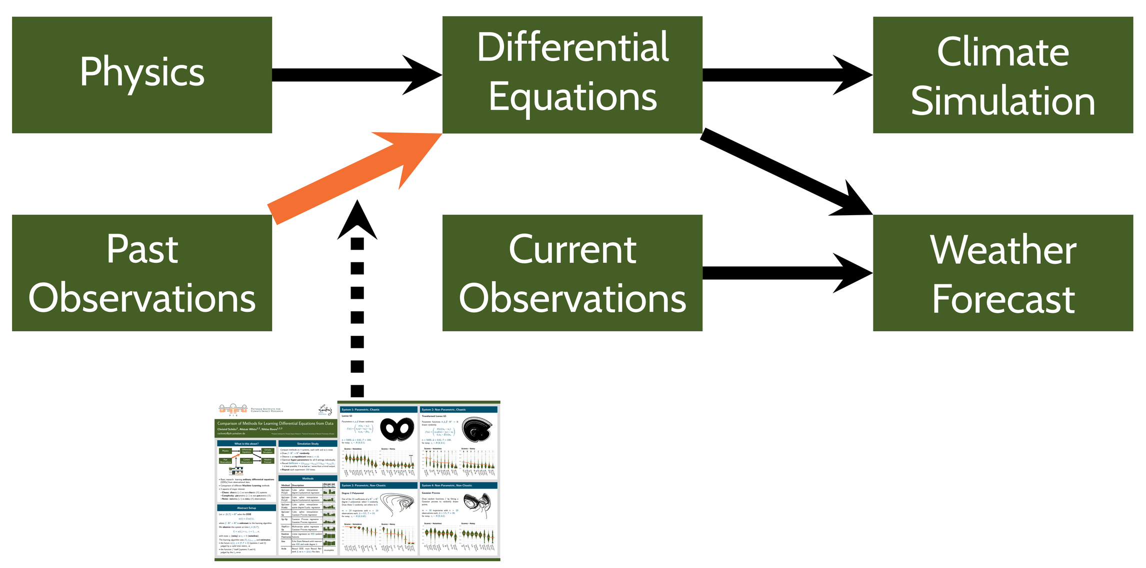
Comparison of Methods for Learning Differential Equations from Data

Christof Schötz[1], Alistair White[1,2], Niklas Boers[1,2,3]

cschoetz@pik-potsdam.de

[1]Potsdam Institute for Climate Impact Research, [2]Technical University of Munich, [3]University of Exeter

What is this about?



- Basic research: learning **ordinary differential equations** (ODEs) from observational data
- Comparison of different **Machine Learning** methods
- 3 aspects of major interest:
  - **Chaos: cha**otic (✓) vs non-**cha**otic (✗) systems
  - **Complexity: par**ametric (✓) vs non-**par**ametric (✗)
  - **Noise: noi**sy (✓) vs **noi**seless (✗) observations

Abstract Setup

Let $u\colon [0,T] \to \mathbb{R}^d$ solve the **ODE**

$$\dot{u}(t) = f(u(t))\,,$$

where $f\colon \mathbb{R}^d \to \mathbb{R}^d$ is **unknown** to the learning algorithm.

We **observe** this system at time $t_i \in [0,T]$,

$$Y_i = u(t_i) + \varepsilon_i\,,\ i = 1, \dots, n\,,$$

with noise ε_i (**noisy**) or $\varepsilon_i = 0$ (**noiseless**).

The learning algorithm sees $(Y_i, t_i)_{i=1,\dots,n}$ and **estimates**

- the future $u(s)$, $s \in [T, T+S]$ (systems 1 and 2) judged by a *valid time* metric, or
- the function f itself (systems 3 and 4) judged by the L_2-error.

Simulation Study

Compare methods on 4 systems, each with and w/o noise.

- Draw $f\colon \mathbb{R}^3 \to \mathbb{R}^3$ **randomly**.
- Observe u at **equidistant** times $t_i = i\Delta$.
- Optimize **hyper-parameters** for all 8 settings individually.
- Record SkillScore $= ((v_{\text{method}} - v_{\text{trivial}}) / (v_{\text{best}} - v_{\text{trivial}}))_+$. 1 is best possible; 0 is as bad as / worse than a trivial output.
- **Repeat** each experiment 100 times.

Methods

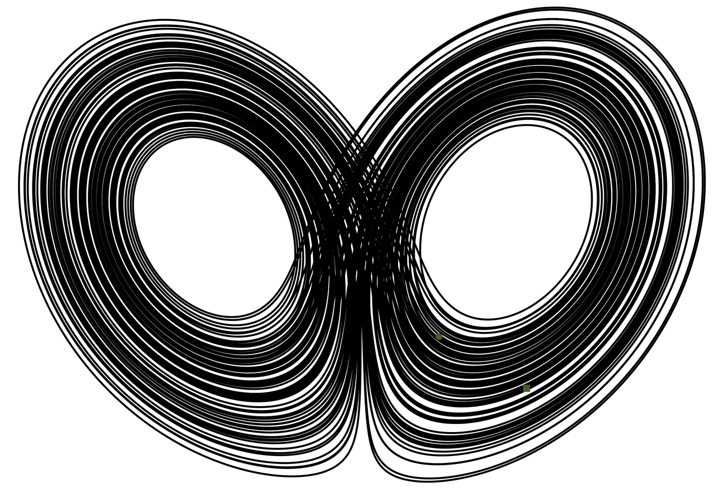
Method	Description	cha ✗✓ par ✗✓ noi ✗✓
`Spline-Poly2`	Cubic spline interpolation + degree-2-polynomial regression	
`Spline-Poly5`	Cubic spline interpolation + degree-5-polynomial regression	
`Spline-Sindy`	Cubic spline interpolation + sparse degree-5-poly. regression	
`Spline-Gp`	Cubic spline interpolation + Gaussian Process regression	
`Gp-Gp`	Gaussian Process regression + Gaussian Process regression	
`ThnPlt-Gp`	Multivariate spline regression + Gaussian Process regression	
`Random Features`	Linear regression on 400 random features	
`Esn`	Echo State Network with reservoir size 400 and node degree 6	
`Node`	Neural ODE: train Neural Network f_θ so $\dot{u} = f_\theta(u)$ fits data	*incomplete*

System 1: Parametric, Chaotic

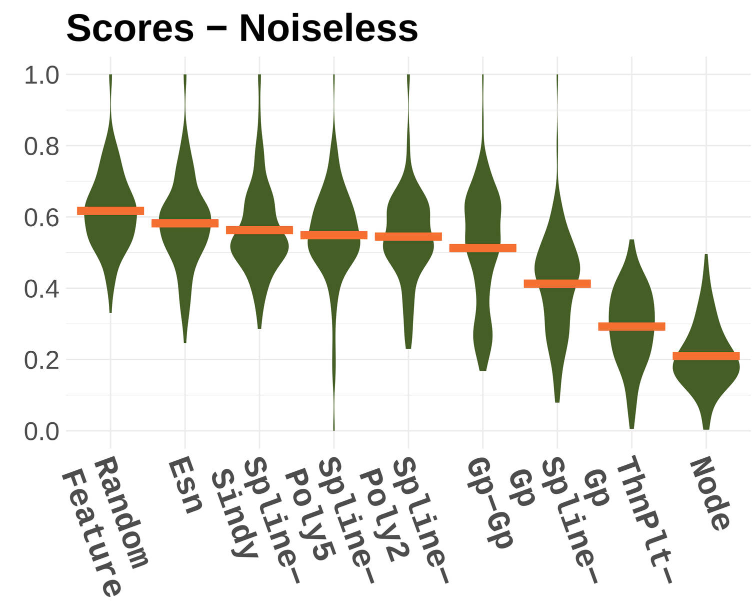
Lorenz 63

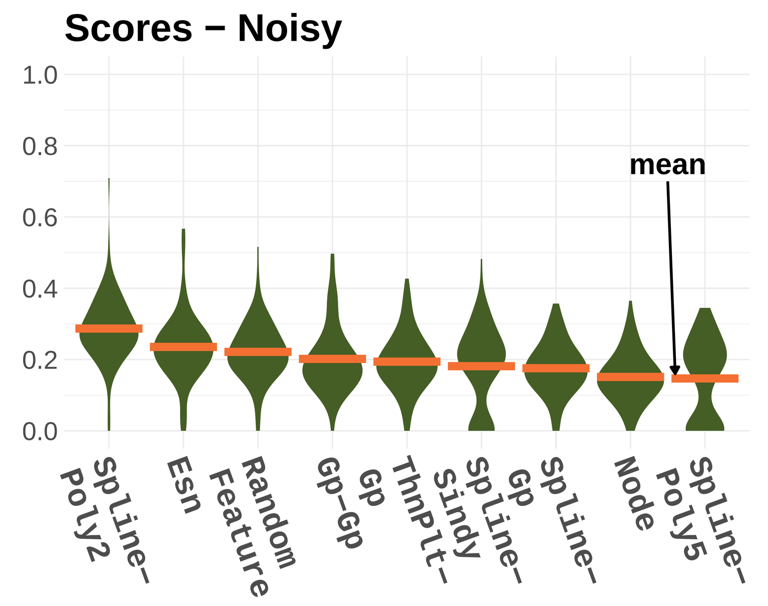
Parameters σ, ρ, β drawn randomly.

$$f(u) = \begin{pmatrix} \sigma(u_2 - u_1) \\ u_1(\rho - u_3) - u_2 \\ u_1 u_2 - \beta u_3 \end{pmatrix}$$



$n = 5000$, $\Delta = 0.02$, $T = 100$,
for noisy: $\varepsilon_i \sim \mathcal{N}(0, 0.1)$



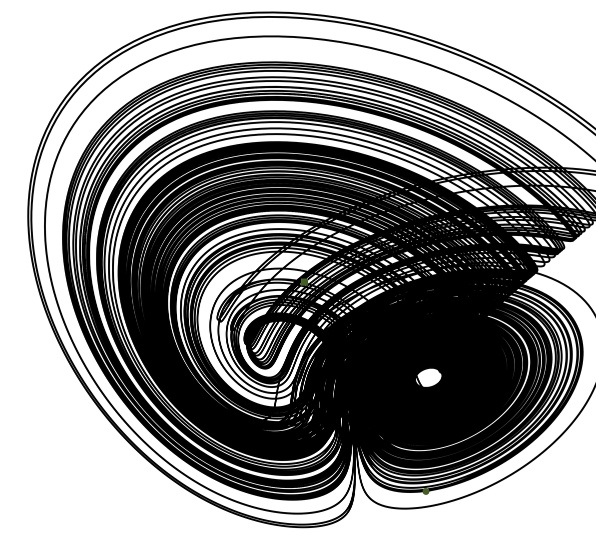


System 2: Non-Parametric, Chaotic

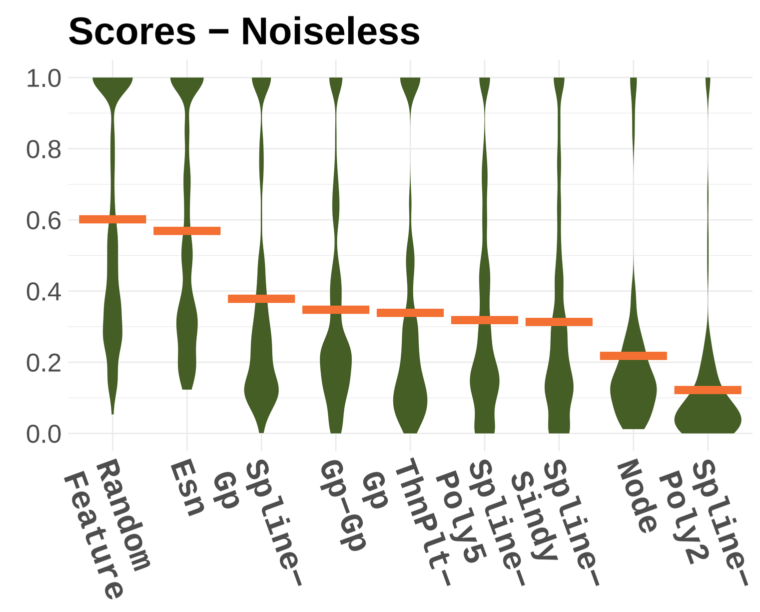
Transformed Lorenz 63

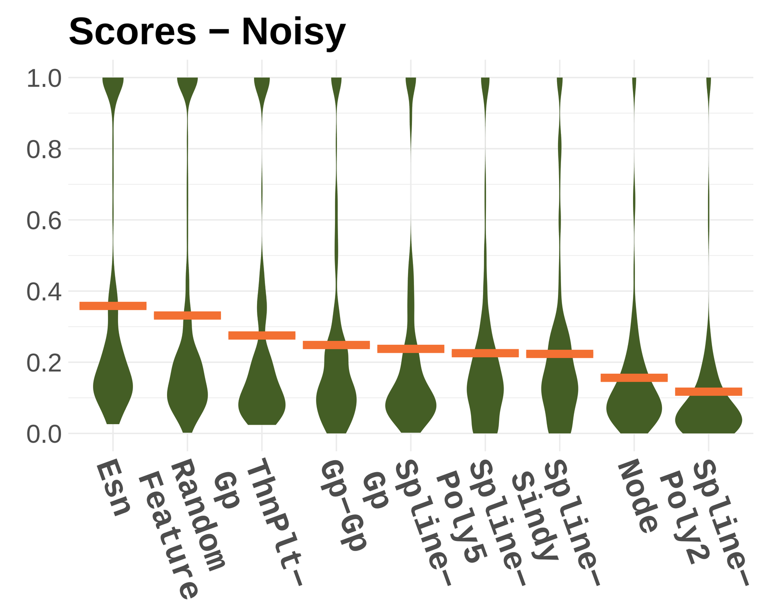
Parameter functions $\tilde{\sigma}, \tilde{\rho}, \tilde{\beta}\colon \mathbb{R}^3 \to \mathbb{R}$ drawn randomly.

$$f(u) = \begin{pmatrix} \tilde{\sigma}(u)(u_2 - u_1) \\ u_1(\tilde{\rho}(u) - u_3) - u_2 \\ u_1 u_2 - \tilde{\beta}(u) u_3 \end{pmatrix}$$



$n = 5000$, $\Delta = 0.02$, $T = 100$,
for noisy: $\varepsilon_i \sim \mathcal{N}(0, 0.1)$

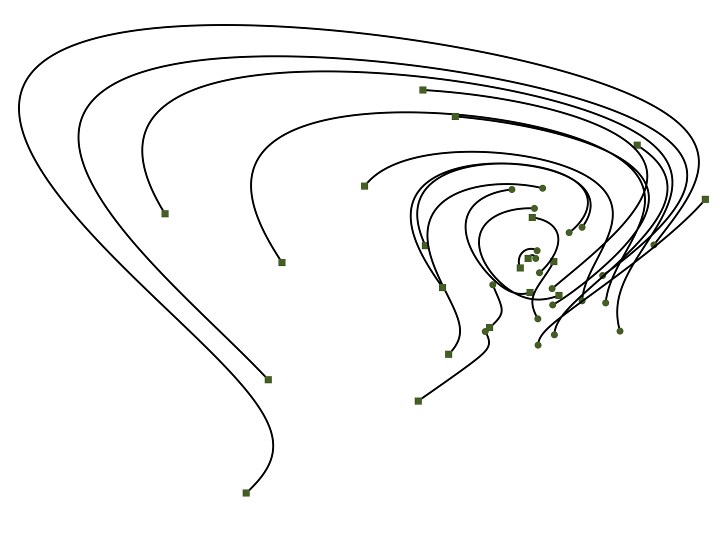




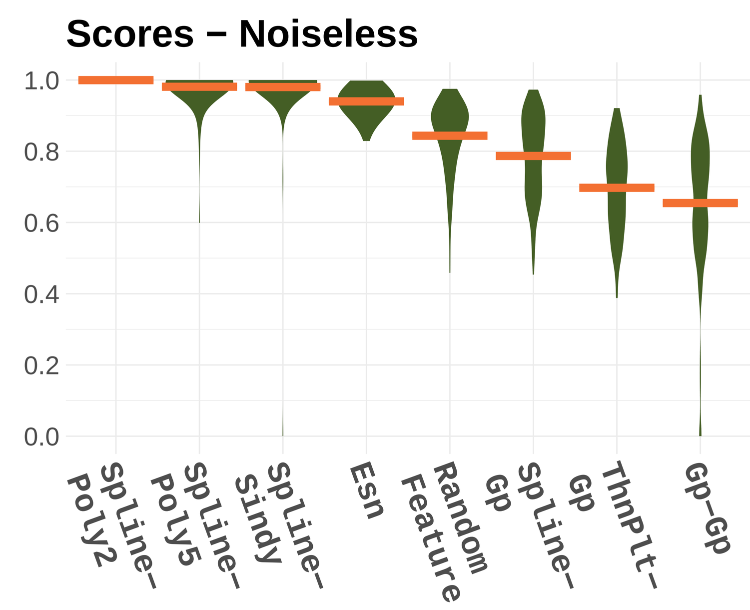
System 3: Parametric, Non-Chaotic

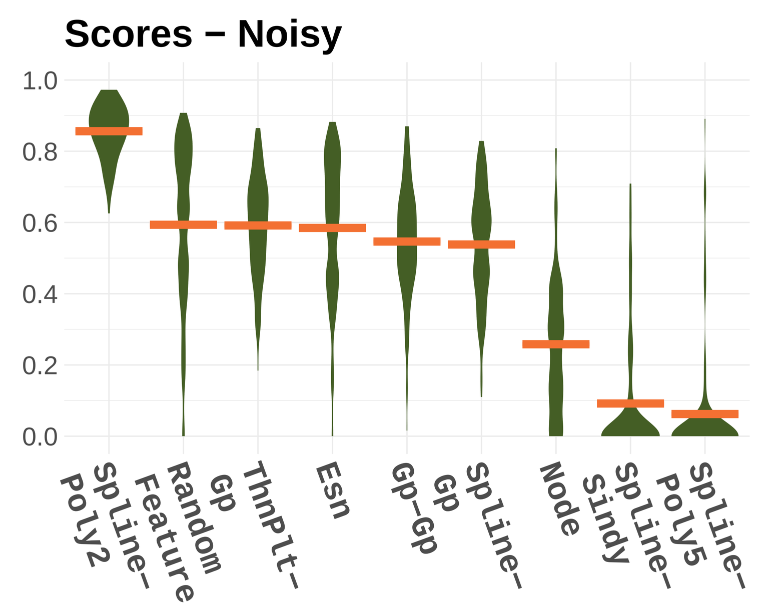
Degree 2 Polynomial

Out of the 30 coefficients of a $\mathbb{R}^3 \to \mathbb{R}^3$ degree 2 polynomial, select 5 randomly. Draw these 5 randomly; set others to 0.



$m = 20$ trajectories with $n = 20$ observations each, $\Delta = 0.5$, $T = 10$,
for noisy: $\varepsilon_i \sim \mathcal{N}(0, 0.05)$

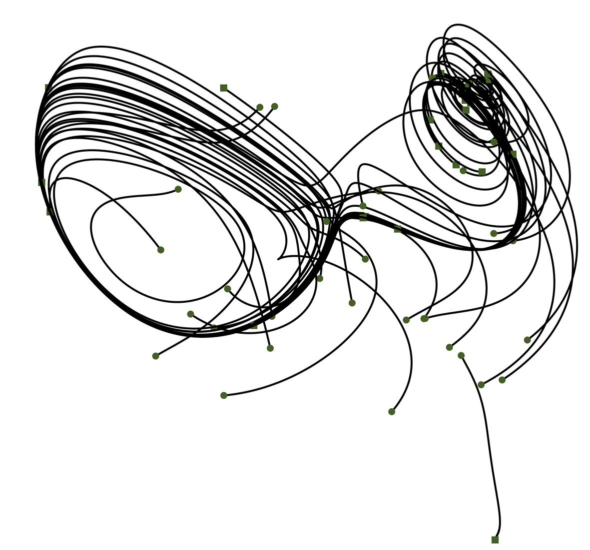




System 4: Non-Parametric, Non-Chaotic

Gaussian Process

Draw random functions f by fitting a Gaussian process to randomly drawn points.



$m = 30$ trajectories with $n = 20$ observations each, $\Delta = 1.5$, $T = 30$,
for noisy: $\varepsilon_i \sim \mathcal{N}(0, 0.2)$