Automatically identifying ordinary differential equations from data

Kevin Egan [1] Weizhen Li [1] Rui Carvalho[1,2]

May 3, 2023

[1]Department of Engineering, Durham University, Lower Mountjoy, South Road, Durham, DH1 3LE, United Kingdom

[2]Institute for Data Science, Durham University, South Road, Durham, DH1 3LE, United Kingdom

Motivation

- Dynamical systems are fundamental models in many fields, including physics, engineering, biology, and economics.
- Often, we have data from these systems but not their underlying equations, which hinders our ability to understand and predict their behaviour.
- Traditional modelling methods can be time-consuming and may require prior knowledge of the system.
- Can we automatically infer the governing equations of dynamical systems from data? This is the problem we aim to tackle.

Modeling systems of ODEs with linear regression

- We are interested in systems of ODEs:

$$\frac{d}{dt}x_j(t) = \dot{x}_j(t) = f_j(x(t)) \qquad j = 1, \ldots, m, \tag{1}$$

where

  - $x = x(t) = (x_1(t) \;\; x_2(t) \;\; \cdots \;\; x_m(t))^T \in \mathbb{R}^m$ represents the state space with m dimensions.
  - $f(x(t)) : \mathbb{R}^m \to \mathbb{R}^m$ describes the system's evolution in time.

- Approximate $\dot{x}$ symbolically by

$$\dot{x}_j \approx \theta_F^T(x)\beta_j, \qquad j = 1, \ldots, m, \tag{2}$$

where

  - $\theta_F(x)$ is a feature vector containing p symbolic functions, each representing an ansatz that we can use to describe the dynamics.
  - $\beta_j \in \mathbb{R}^p$ is a sparse coefficient vector with elements representing the system's parameters.

Modeling systems of ODEs with linear regression

- Create a state matrix $\tilde{\mathbf{X}} \in \mathbf{R}^{n \times m}$ from measurements $x(t)$ taken at times $t_1, t_2, ..., t_n$.
- Apply the Savitzky-Golay filter to smooth each column $\mathbf{x}_j = SG(\tilde{\mathbf{x}}_j)$ and calculate the derivative $\dot{\mathbf{x}}_j$.
- Consolidate $\mathbf{X}$ and $\dot{\mathbf{X}}$.
- Build the block design matrix $\mathbf{\Theta}(\mathbf{X}) \in \mathbf{R}^{n \times p}$:

$$\mathbf{\Theta}(\mathbf{X}) = \begin{pmatrix} | & | & | & & | & | \\ \mathbf{1} & \mathbf{X} & \mathbf{X}^{[2]} & \dots & \mathbf{X}^{[d]} & \mathbf{\Phi}(\mathbf{X}) \\ | & | & | & & | & | \end{pmatrix}, \qquad (3)$$

where

- $\mathbf{X}^{[i]}$ for $i = 1, \dots, d$ is a matrix whose column vectors denote all possible monomials of order i in $x(t)$.
- $\mathbf{\Phi}(\mathbf{X})$ can contain nonlinear functions such as trigonometric, logarithmic, or exponential.

Modeling systems of ODEs with linear regression

- Derive a linear regression:

$$\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{B} + \mathbf{E}, \tag{4}$$

where

- $\mathbf{X}, \dot{\mathbf{X}} \in \mathbf{R}^{n \times m}$.
- $\mathbf{\Theta}(\mathbf{X}) \in \mathbf{R}^{n \times p}$.
- $\mathbf{B} \in \mathbf{R}^{p \times m}$ is the matrix of coefficients.
- $\mathbf{E} \in \mathbf{R}^{n \times m}$ are the residuals.

for

- n: # observations.
- m: dimension of state space.
- p: # candidate functions.

PNAS

Discovering governing equations from data by sparse identification of nonlinear dynamical systems

Steven L. Brunton[a,1], Joshua L. Proctor[b], and J. Nathan Kutz[c]

[a]Department of Mechanical Engineering, University of Washington, Seattle, WA 98195; [b]Institute for Disease Modeling, Bellevue, WA 98005; and [c]Department of Applied Mathematics, University of Washington, Seattle, WA 98195

Edited by William Bialek, Princeton University, Princeton, NJ, and approved March 1, 2016 (received for review August 31, 2015)

Extracting governing equations from data is a central challenge in many diverse areas of science and engineering. Data are abundant whereas models often remain elusive, as in climate science, neuroscience, ecology, finance, and epidemiology, to name only a few examples. In this work, we combine sparsity-promoting techniques and machine learning with nonlinear dynamical systems to discover governing equations from noisy measurement data. The only assumption about the structure of the model is that there are only a few important terms that govern the dynamics, so that the equations are sparse in the space of possible functions; this assumption holds for many physical systems in an appropriate basis. In particular, we use sparse regression to determine the fewest terms in the dynamic governing equations required to accurately represent the data. This results in parsimonious models that balance accuracy with model complexity to avoid overfitting. We demonstrate the algorithm on a wide range of problems, from simple canonical systems, including linear and nonlinear oscillators and the chaotic Lorenz system, to the fluid vortex shedding behind an obstacle. The fluid example illustrates the ability of this method to discover the underlying dynamics of a system that took experts in the community nearly 30 years to resolve. We also show that this method generalizes to parameterized systems and systems that are time-varying or have external forcing.

dynamical systems | machine learning | sparse regression | system identification | optimization

Advances in machine learning (1) and data science (2) have promised a renaissance in the analysis and understanding of complex data, extracting patterns in vast multimodal data that are

dynamical systems from data. However, symbolic regression is expensive, does not scale well to large systems of interest, and may be prone to overfitting unless care is taken to explicitly balance model complexity with predictive power. In ref. 4, the Pareto front is used to find parsimonious models. There are other techniques that address various aspects of the dynamical system discovery problem. These include methods to discover governing equations from time-series data (6), equation-free modeling (7), empirical dynamic modeling (8, 9), modeling emergent behavior (10), and automated inference of dynamics (11–13); ref. 12 provides an excellent review.

Sparse Identification of Nonlinear Dynamics (SINDy)

In this work, we reenvision the dynamical system discovery problem from the perspective of sparse regression (14–16) and compressed sensing (17–22). In particular, we leverage the fact that most physical systems have only a few relevant terms that define the dynamics, making the governing equations sparse in a high-dimensional nonlinear function space. The combination of sparsity methods in dynamical systems is quite recent (23–30). Here, we consider dynamical systems (31) of the form

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)). \quad [1]$$

The vector $\mathbf{x}(t) \in \mathbb{R}^n$ denotes the state of a system at time t, and the function $\mathbf{f}(\mathbf{x}(t))$ represents the dynamic constraints that define the equations of motion of the system, such as Newton's second law. Later, the dynamics will be generalized to include parameterization, time dependence, and forcing.

Identifying the Lorenz Equations

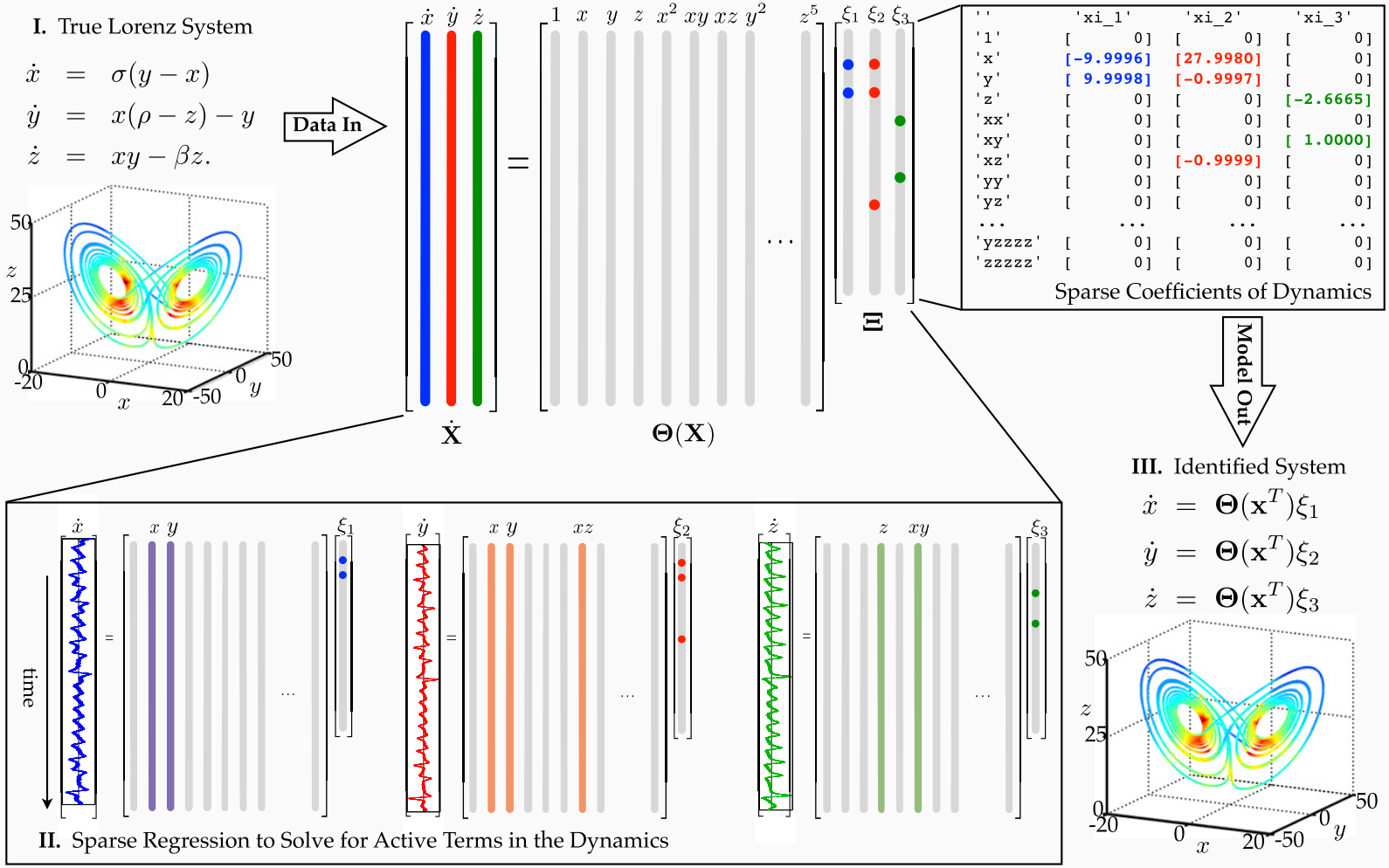


Fig. 1. Schematic of the SINDy algorithm, demonstrated on the Lorenz equations. Data are collected from the system, including a time history of the states $\mathbf{X}$ and derivatives $\dot{\mathbf{X}}$; the assumption of having $\dot{\mathbf{X}}$ is relaxed later. Next, a library of nonlinear functions of the states, $\Theta(\mathbf{X})$, is constructed. This nonlinear feature library is used to find the fewest terms needed to satisfy $\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}$. The few entries in the vectors of $\boldsymbol{\Xi}$, solved for by sparse regression, denote the relevant terms in the right-hand side of the dynamics. Parameter values are $\sigma = 10, \beta = 8/3, \rho = 28$, $(x_0, y_0, z_0)^T = (-8, 7, 27)^T$. The trajectory on the Lorenz attractor is colored by the adaptive time step required, with red indicating a smaller time step.

Sparse Identification of Nonlinear Dynamics (SINDy)

Sequential thresholded least-squares algorithm

- Start with a least-squares solution for Ξ and then threshold all coefficients that are smaller than some cutoff value λ_{SINDy}.
- Once the indices of the remaining non-zero coefficients are identified, we obtain another least-squares solution for Ξ onto the remaining indices.
- These new coefficients are again thresholded using λ_{SINDy}, and the procedure continues until the non-zero coefficients converge.

Automatic regression for governing equations (ARGOS)

- Combines Savitzky-Golay filter, sparse regression, and bootstrap sampling.
- Effectively addresses the inverse problem of inferring underlying dynamics from observational data.
- Outperforms the established SINDy with AIC, especially in three-dimensional systems.

The Savitzky-Golay filter

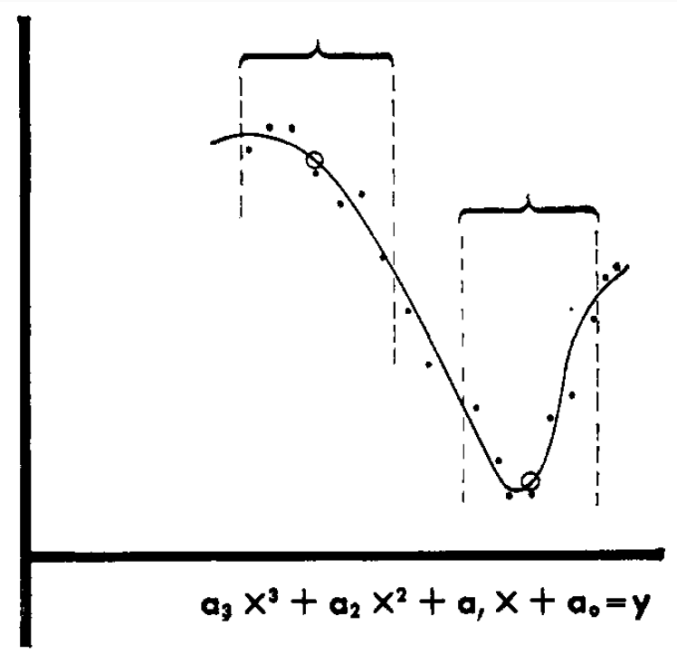


Figure 1: Representation of a 7-point moving polynomial smooth (from Savitzky and Golay, 1964).

- Applies local polynomial regression to smooth the data and approximate the derivatives.
- Build a grid of window lengths l with polynomial order $o = 4$.
- For each column of $\tilde{\mathbf{X}}$, find optimal l^* corresponding to the minimum MSE between the noisy $\tilde{\mathbf{x}}_j$ and its corresponding smoothed signal $\mathbf{x}_j$.

Ridge, Lasso and Adaptive Lasso for variable selection

$$\underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(\dot{x}_i - \beta_0 - \sum_{k=1}^{p} \theta(\mathbf{X})_{i,k} \beta_k \right)^2 + \lambda \sum_{k=1}^{p} w_k |\beta_k|^q \right\}. \quad (5)$$

- All weights $w_k = 1$ for $k = 1, \ldots, p$:
  - Lasso: $q = 1$ (sparse but unstable estimates when predictors are collinear).
  - Ridge regression: $q = 2$ (not sparse but stable when predictors are collinear).
- Adaptive lasso (oracle property): $q = 1$.
  - first stage: pilot estimates $\tilde{\beta}$.
  - second stage: $w_k = 1/|\tilde{\beta}_k|^\nu$ $(\nu = 1)$

  The weighted penalty in the adaptive lasso can be interpreted as an approximation of the ℓ_p penalties with $p = 1 - \nu$.

Bootstrap Sampling for Confidence Interval Estimation

- Generates multiple datasets by resampling original data with replacement.
- Computes empirical distribution of estimates for robust coefficient estimation.
- Eliminates spurious terms and selects the ones that best represent underlying equations.

Sparse regression

- Extract each column of $\dot{\mathbf{X}}$ and $\mathbf{B}$ from Eq. (4):

$$\dot{\mathbf{x}}_j = \boldsymbol{\Theta}(\mathbf{X})\beta_j + \epsilon_j, \qquad j = 1, \ldots, m, \tag{6}$$

- Variable selection with either the lasso or the adaptive lasso (`glmnet` with CV).
- Upon determining an initial estimate of β_j in Eq. (6), trim $\boldsymbol{\Theta}$ to contain only monomial terms up to that estimate's highest-order variable with a nonzero coefficient.
- New round of variable selection with lasso/adaptive lasso on the trimmed design matrix. Apply a grid of thresholded values to the identified coefficients.
- Perform OLS on each subset of selected variables to identify unbiased estimates for each model.
- Model selection with BIC.
- Use the trimmed design matrix to bootstrap this process and obtain 2000 sample estimates.
- Construct 95% bootstrap confidence intervals with the sample estimates and identify a final model consisting of variables whose confidence intervals do not contain zero and whose point estimates fall within their confidence intervals.

Automatic regression for governing equations (ARGOS)

$$\dot{x}_1 = -0.1x_1 + 2x_2$$
$$\dot{x}_2 = -2x_1 - 0.1x_2$$

a Noisy data

$$\tilde{\mathbf{X}} = \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix}^T$$

b Trim design matrix

$$\dot{x}_1 = \begin{pmatrix} | & | & | & | & | & | & | & | & | & | \\ 1 & x_1 & x_2 & x_1^2 & x_1x_2 & x_2^2 & \cancel{x_1^3} & \cancel{x_1^2x_2} & \cdots & \cancel{x_2^5} \\ | & | & | & | & | & | & | & | & | & | \end{pmatrix} \beta^{(0)}$$

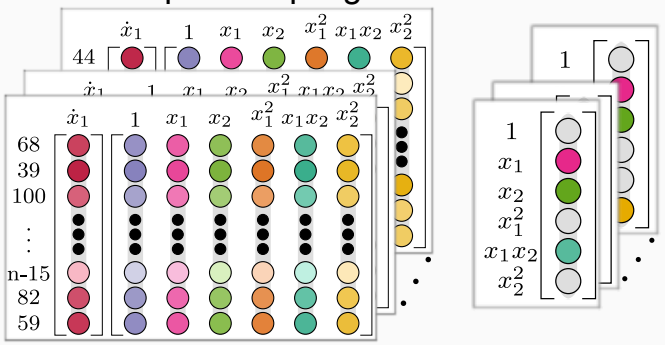
$$\boldsymbol{\Theta}^{(0)}(\mathbf{X})$$

c Point estimate

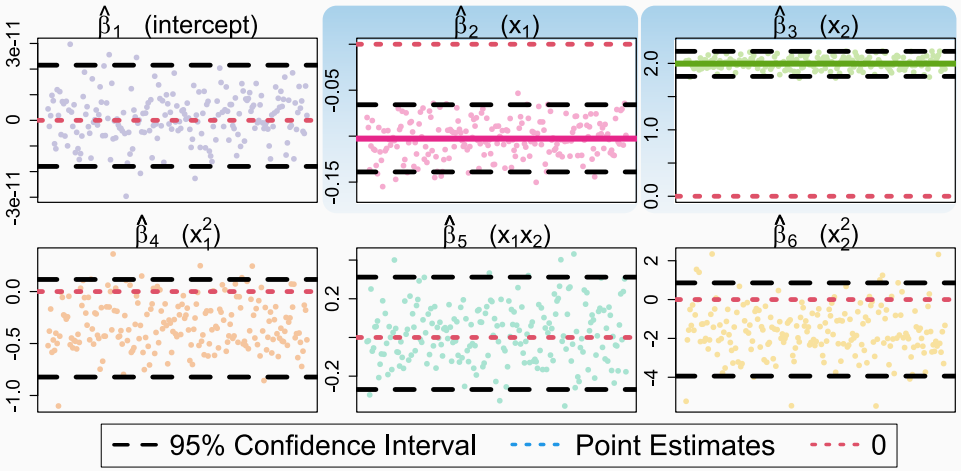
$$(\dot{x}_1)\,, \begin{pmatrix} | & | & | & | & | & | \\ 1 & x_1 & x_2 & x_1^2 & x_1x_2 & x_2^2 \\ | & | & | & | & | & | \end{pmatrix} \rightarrow \hat{\beta}^{(1)}$$

$$\boldsymbol{\Theta}^{(1)}(\mathbf{X})$$

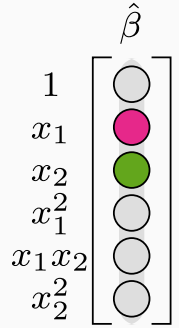
d Bootstrap resampling



e Bootstrap confidence intervals



f Identified model



$$\hat{\dot{x}}_1 = -0.1026x_1 + 1.9964x_2$$
$$\hat{\dot{x}}_2 = -2.0582x_1 - 0.1001x_2$$

Automatic regression for governing equations (ARGOS)

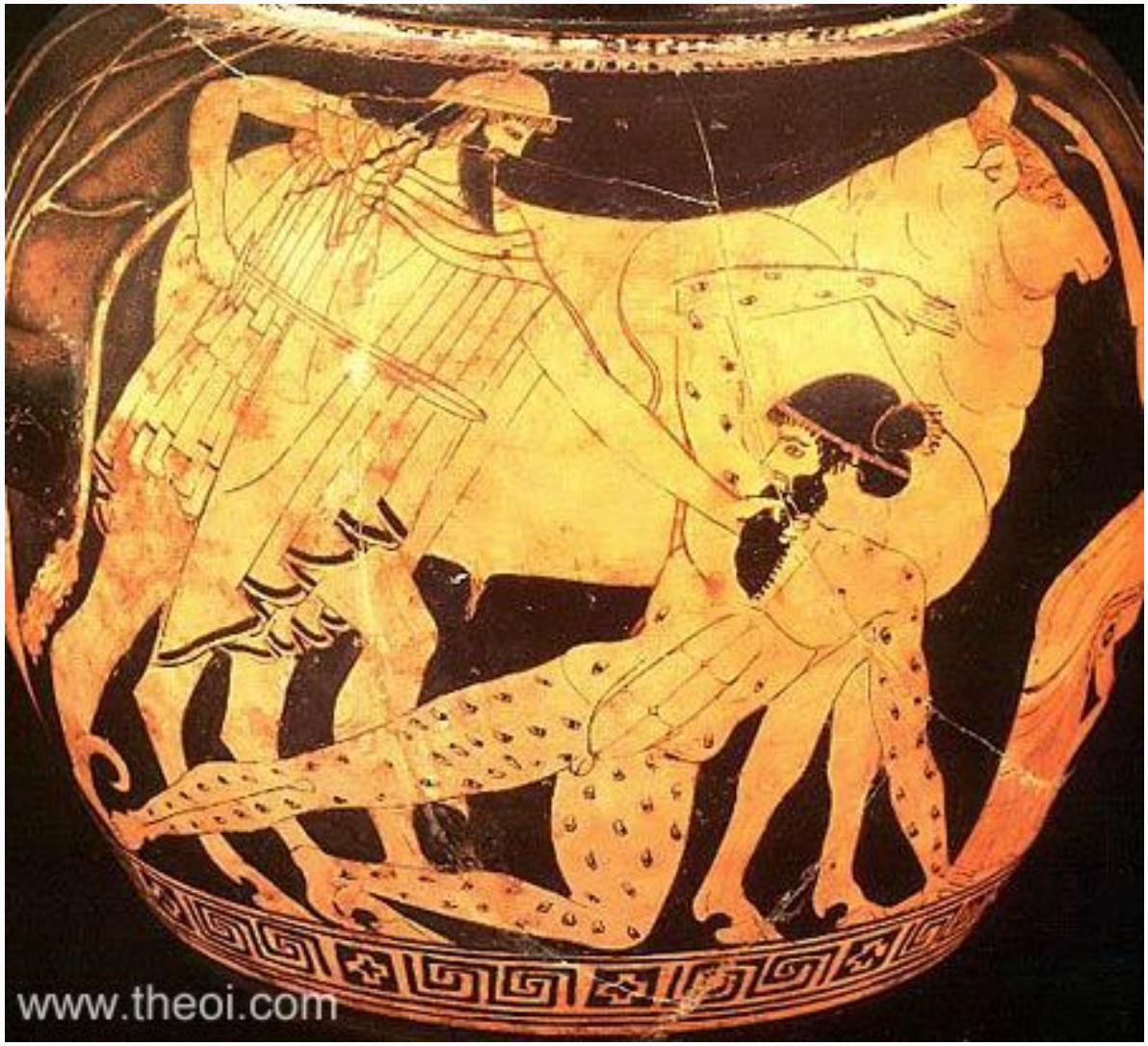


Figure 2: Hermes slaying Argus Panoptes, Athenian red-figure vase C5th B.C (credit:Theoi.com)

Assessing ARGOS systematically: linear systems

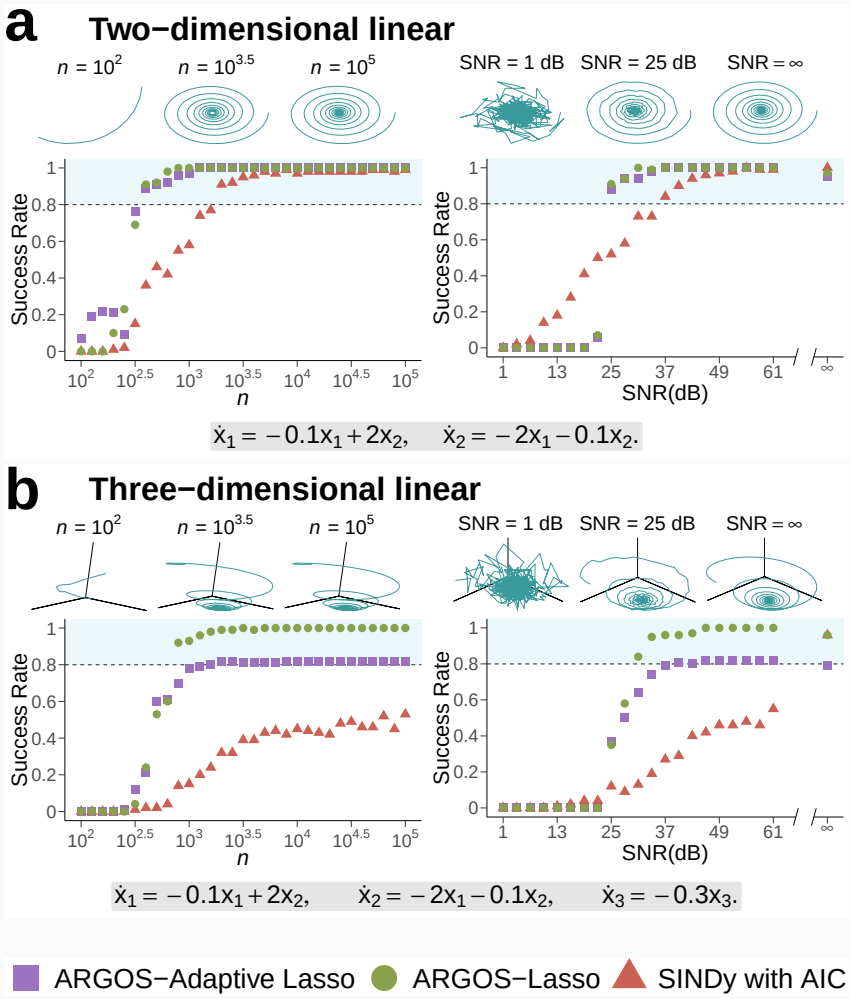


Figure 3: Success rate (prob. of identifying the true terms) for the identification of linear systems with 100 random initial conditions. (left) Increasing size of the time series n. (right) Increasing signal-to-noise ratio (SNR).

Assessing ARGOS systematically: non-linear systems

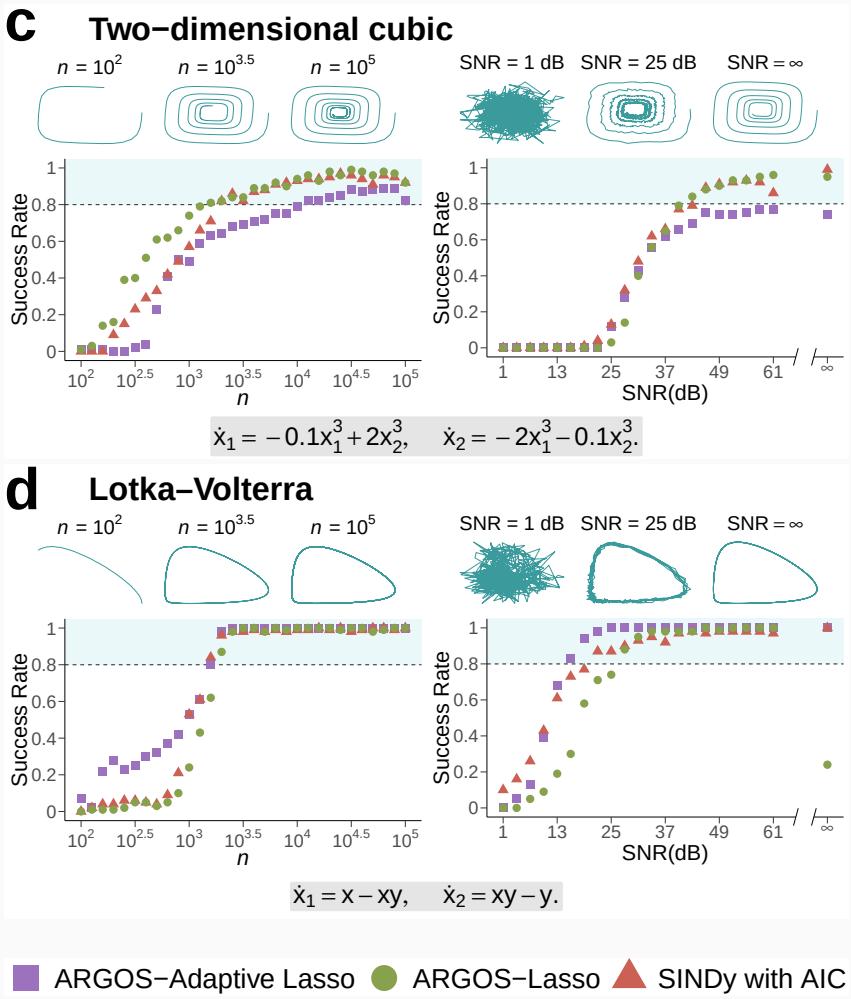


Figure 4: (left) Increasing size of the time series n. (right) Increasing signal-to-noise ratio (SNR).

Assessing ARGOS systematically: non-linear systems (cont.)

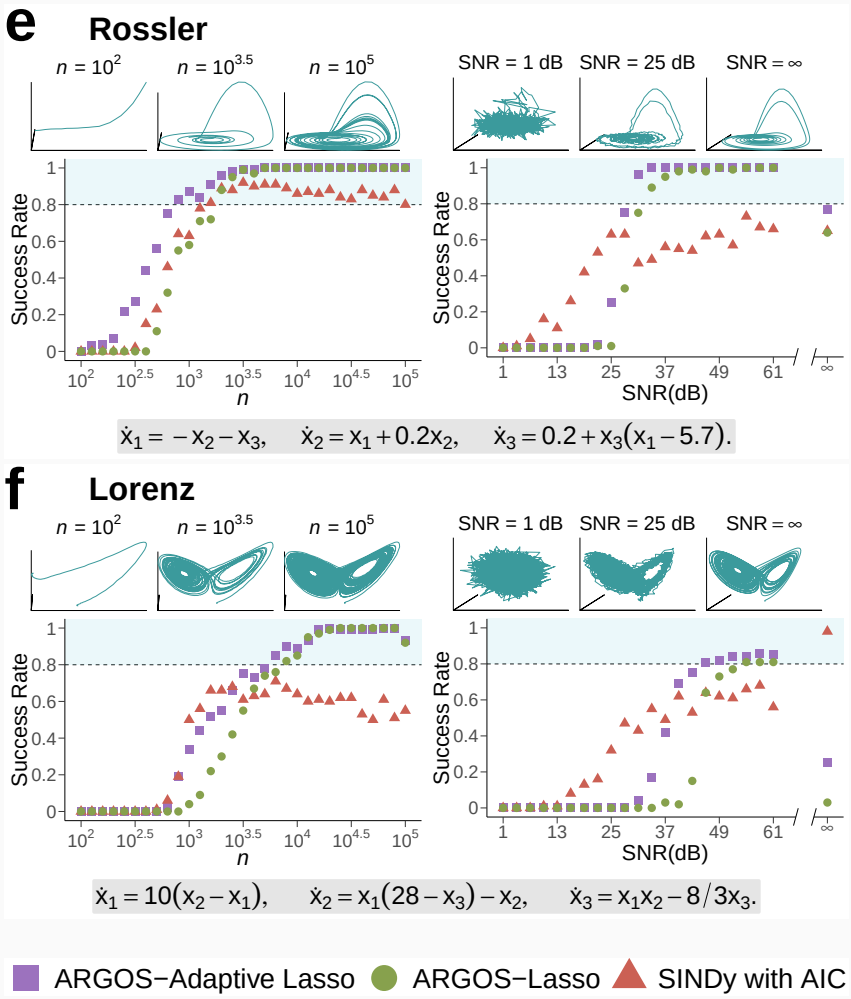


Figure 5: (left) Increasing size of the time series n. (right) Increasing signal-to-noise ratio (SNR).

Assessing ARGOS systematically: non-linear systems (cont.)

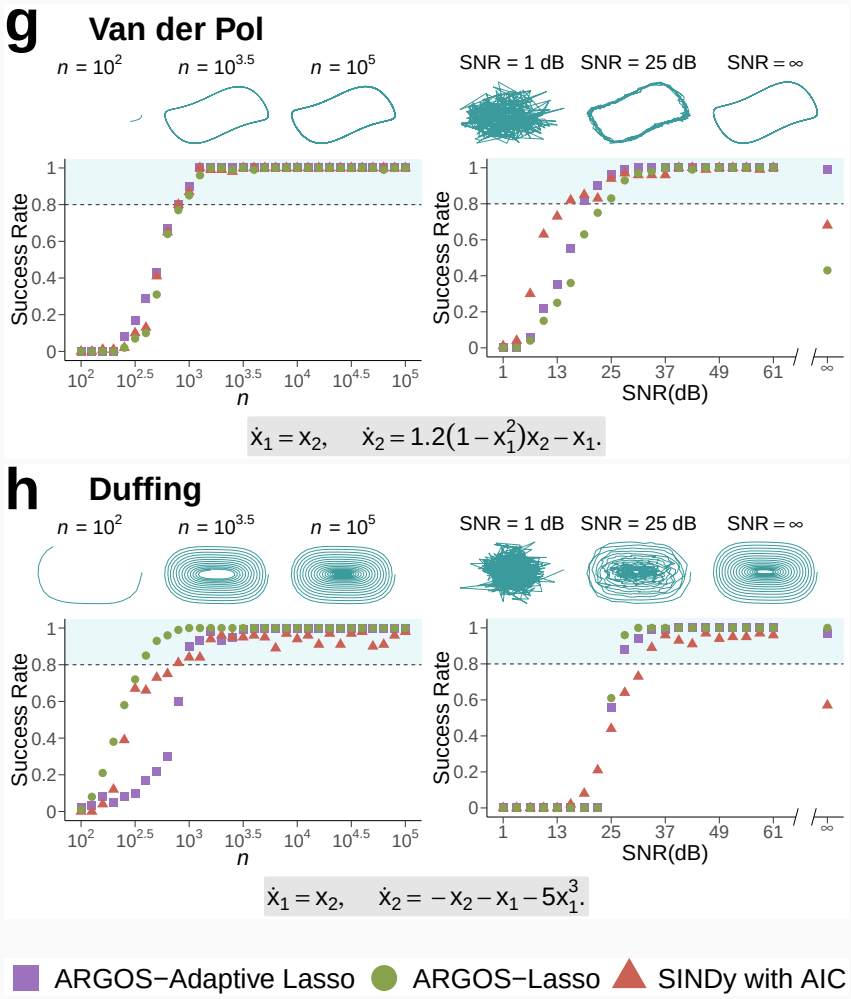


Figure 6: (left) Increasing size of the time series n. (right) Increasing signal-to-noise ratio (SNR).

Identified variables for the Lorenz (SNR = 49)

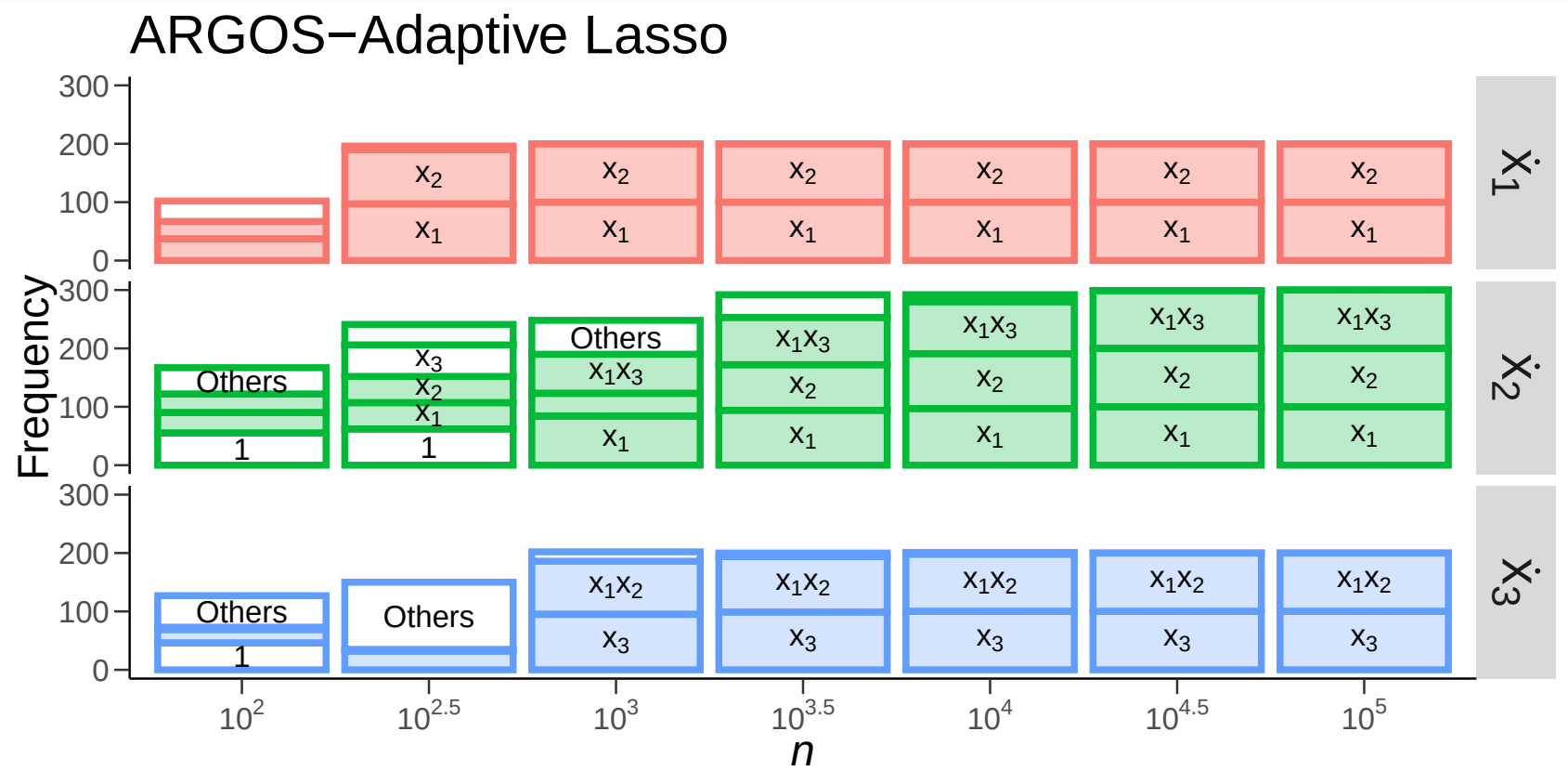
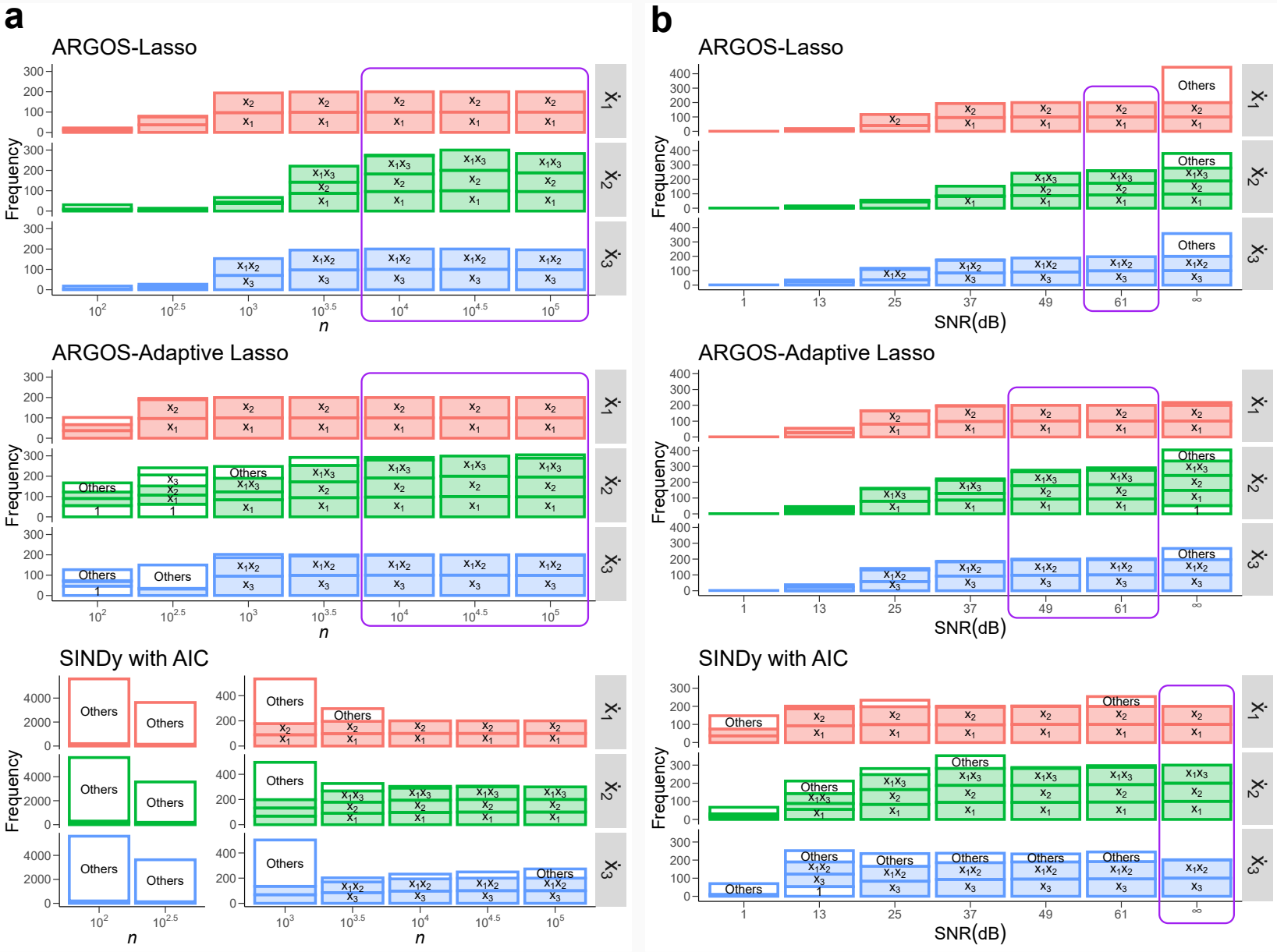


Figure 7: We fill the boxes for correctly identified variables with color, and any incorrectly selected variables with white, providing labels for each term when possible. The size of the time-series n increases for (SNR = 49).

Identified variables for the Lorenz (SNR = 49 and $n = 5000$)



Discussion

- ARGOS contributes to the search for elusive laws governing complex systems.
- We applied ARGOS to diverse trajectories and systems of ODEs.
- ARGOS is a robust identification method for several first- and second-order nonlinear systems (e.g., Lorenz and Van der Pol).
- Our method identifies systems of ODEs in three dimensions.
- When data are noiseless, multicollinearity in the design matrix impacts how ℓ_1 regularization methods perform variable selection, leading to selecting one collinear term and ignoring the rest.
- Better performance with low levels of noise in data.
- Confidence intervals (built with the boostrap) help to perform variable selection.

Machine Intelligence for Science Group (MI4Science)



Preprint:

Kevin Egan, Weizhen Li, Rui Carvalho

Automatically identifying ordinary differential equations from data

`https://arxiv.org/abs/2304.11182`