Discovering governing equations from data:

Sparse Identification of Nonlinear Dynamical Systems (SINDy models)

Giorgi Viviana - 993727
Morandi Riccardo - 225333
Negri Matteo - 103265

Sparse Identification of Nonlinear Dynamics

- The main objective is **retrieving** the nonlinear equations governing a physical dynamical system from observed data, i.e. find $\boldsymbol{f}(\cdot)$ s.t. $\dot{\boldsymbol{x}} = \boldsymbol{f}\big(\boldsymbol{x}(\boldsymbol{t})\big)$.
- For many systems, the function $f\big(x(t)\big)$ consists of just few nonlinear terms that govern the dynamics. Our goal is to recover a parsimonious representation through the use of **Sparse Regression.**
- We aim to solve the linear system $\dot{\boldsymbol{X}} = \boldsymbol{\Theta}(\boldsymbol{X})\boldsymbol{\Xi}$, where $\boldsymbol{\Xi}$ is a sparse vector of coefficients related to the active nonlinear terms.
- Sparse regression allows our model to be **interpretable** and computationally easier to manage, while also avoiding overfitting.
- In order to achieve good results, we must be very careful in the choice of an appropriate coordinate system. We will address this issue using autoencoders.

Sequential Least Squares Thresholding

SINDy Algorithm

Input: Candidate function library Θ, Time derivative data $\dot{X}$, Threshold λ, Number of iterations N.
Output: Candidate function coefficients Ξ.

1. Initialize Ξ using LS, that is $\Xi = \text{argmin}_{\Xi^{\mathrm{T}}} \|\dot{X} - \Theta\ \Xi^{\mathrm{T}}\|_2$

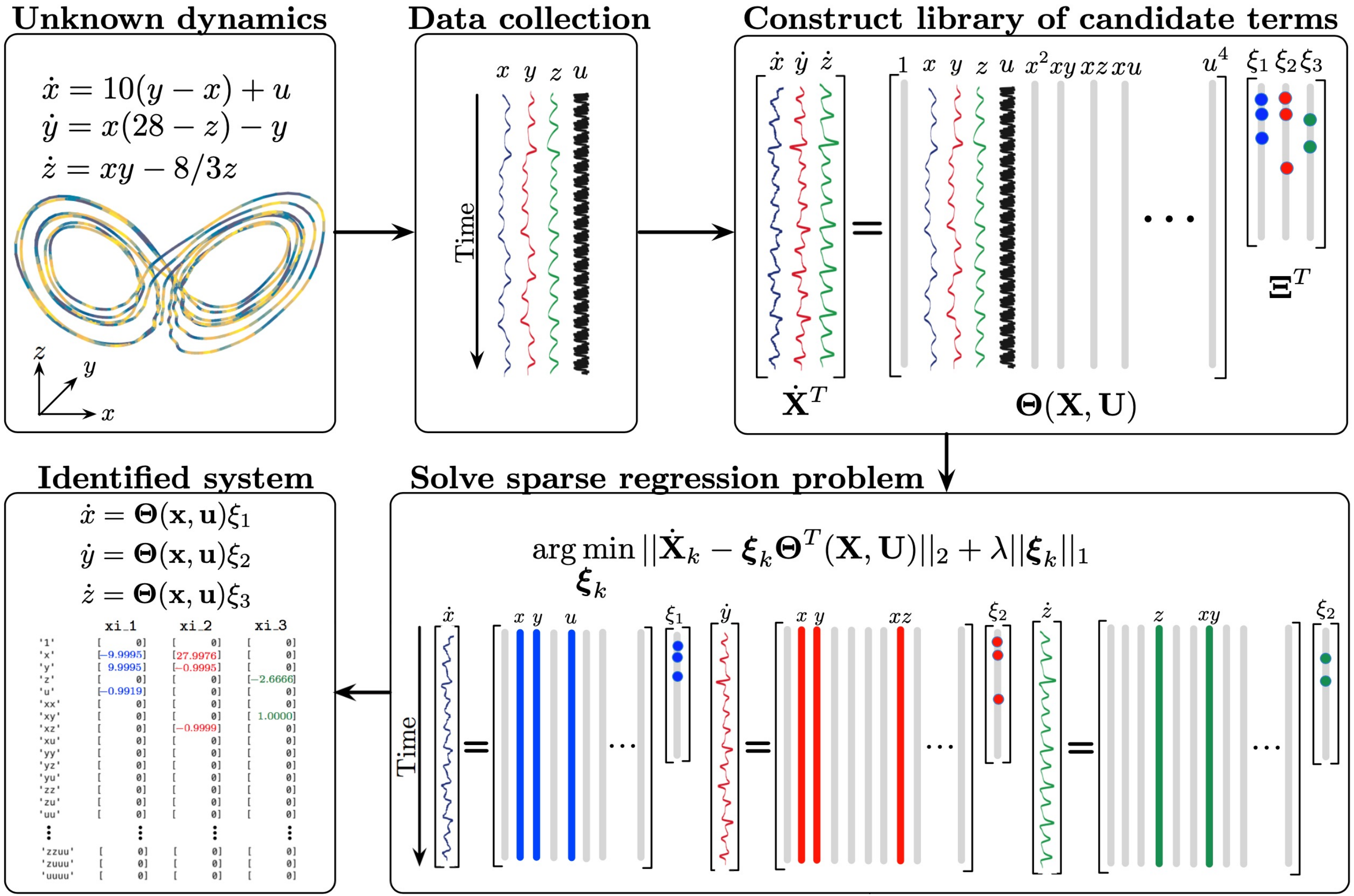
for i = 1,2,...,N **do**

  2. Set to 0 the coefficients in Ξ whose absolute values are below the cutoff threshold λ
  3. Reduce the library of candidate function Θ to the terms whose coefficients are different from 0
  4. Solve again a LS problem with the reduced library: $\Xi = \text{argmin}_{\Xi^{\mathrm{T}}} \|\dot{X} - \Theta\ \Xi^{\mathrm{T}}\|_2$

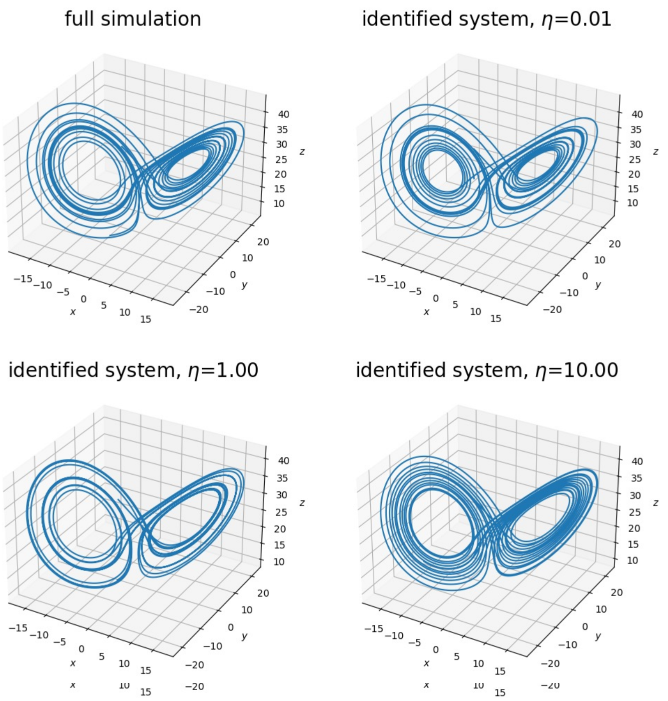
end

5. **return** Ξ

The choice of the **threshold** and of the **library** is crucial!


Unknown dynamics
$\dot{x} = 10(y - x) + u$
$\dot{y} = x(28 - z) - y$
$\dot{z} = xy - 8/3z$
Data collection
x y z u
Time
Construct library of candidate terms
$\dot{x}$ $\dot{y}$ $\dot{z}$
1 x y z u x^2 xy xz xu u^4
ξ_1 ξ_2 ξ_3
$\dot{\mathbf{X}}^T$
$\boldsymbol{\Theta}(\mathbf{X}, \mathbf{U})$
$\boldsymbol{\Xi}^T$
Solve sparse regression problem
$\underset{\boldsymbol{\xi}_k}{\arg\min} ||\dot{\mathbf{X}}_k - \boldsymbol{\xi}_k \boldsymbol{\Theta}^T(\mathbf{X}, \mathbf{U})||_2 + \lambda||\boldsymbol{\xi}_k||_1$
Identified system
$\dot{x} = \boldsymbol{\Theta}(\mathbf{x}, \mathbf{u})\xi_1$
$\dot{y} = \boldsymbol{\Theta}(\mathbf{x}, \mathbf{u})\xi_2$
$\dot{z} = \boldsymbol{\Theta}(\mathbf{x}, \mathbf{u})\xi_3$
xi_1 xi_2 xi_3
'1' [0] [0] [0]
'x' [−9.9995] [27.9976] [0]
'y' [9.9995] [−0.9995] [0]
'z' [0] [0] [−2.6666]
'u' [−0.9919] [0] [0]
'xx' [0] [0] [0]
'xy' [0] [0] [1.0000]
'xz' [0] [−0.9999] [0]
'xu' [0] [0] [0]
'yy' [0] [0] [0]
'yz' [0] [0] [0]
'yu' [0] [0] [0]
'zz' [0] [0] [0]
'zu' [0] [0] [0]
'uu' [0] [0] [0]
'zzuu' [0] [0] [0]
'zuuu' [0] [0] [0]
'uuuu' [0] [0] [0]
Time
$\dot{x}$ x y u ξ_1
$\dot{y}$ x y xz ξ_2
$\dot{z}$ z xy ξ_2

Numerical results: the Lorenz system



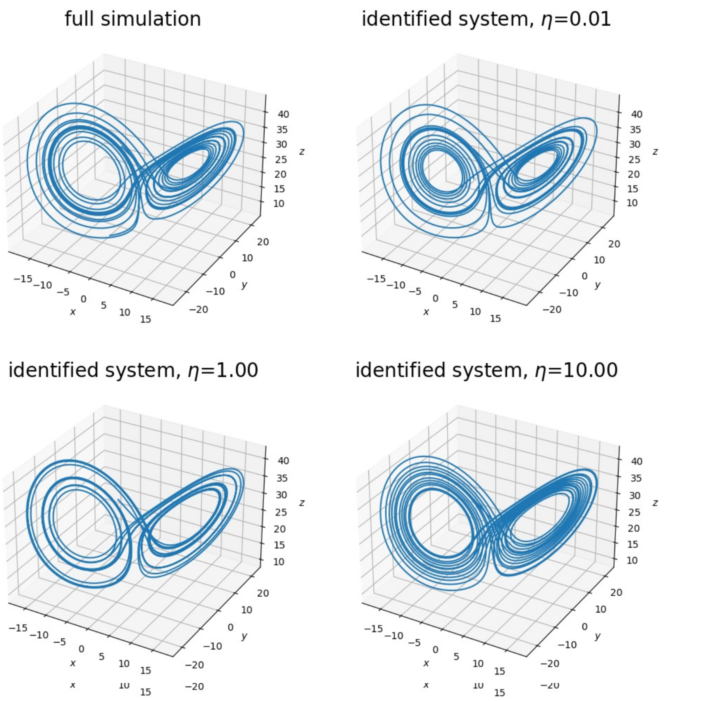
Simulations of Lorenz system with different levels of noise

The canonical Lorenz system is described by the following system of linear ODE:

$$\begin{cases} \dot{x} = -10x + 10y \\ \dot{y} = x(28 - z) - y \\ \dot{z} = xy - \frac{8}{3}z \end{cases}$$

```
model identified with noise level = 0.01:
(x0)' = -10.000 x0 + 10.000 x1
(x1)' = 28.000 x0 + -1.000 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 1.00:
(x0)' = -9.999 x0 + 9.999 x1
(x1)' = 27.999 x0 + -0.999 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 10.00:
(x0)' = 0.053 1 + -10.007 x0 + 10.015 x1
(x1)' = 27.980 x0 + -0.982 x1 + -1.000 x0 x2
(x2)' = -2.670 x2 + 1.000 x0 x1
```

Numerical results: the Lorenz system



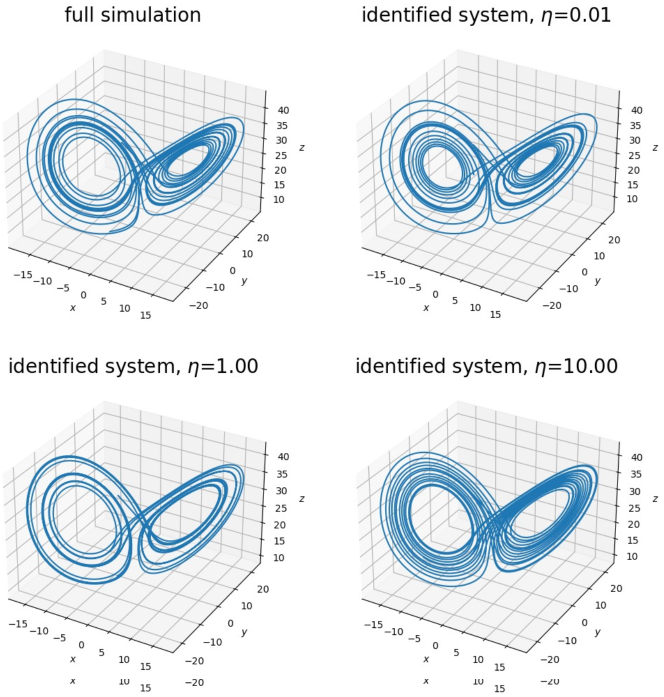
Simulations of Lorenz system with different levels of noise

The canonical Lorenz system is described by the following system of linear ODE:

$$\begin{cases} \dot{x} = -10x + 10y \\ \dot{y} = x(28 - z) - y \\ \dot{z} = xy - \frac{8}{3}z \end{cases}$$

```
model identified with noise level = 0.01:
(x0)' = -10.000 x0 + 10.000 x1
(x1)' = 28.000 x0 + -1.000 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 1.00:
(x0)' = -9.999 x0 + 9.999 x1
(x1)' = 27.999 x0 + -0.999 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 10.00:
(x0)' = 0.053 1 + -10.007 x0 + 10.015 x1
(x1)' = 27.980 x0 + -0.982 x1 + -1.000 x0 x2
(x2)' = -2.670 x2 + 1.000 x0 x1
```

Numerical results: the Lorenz system



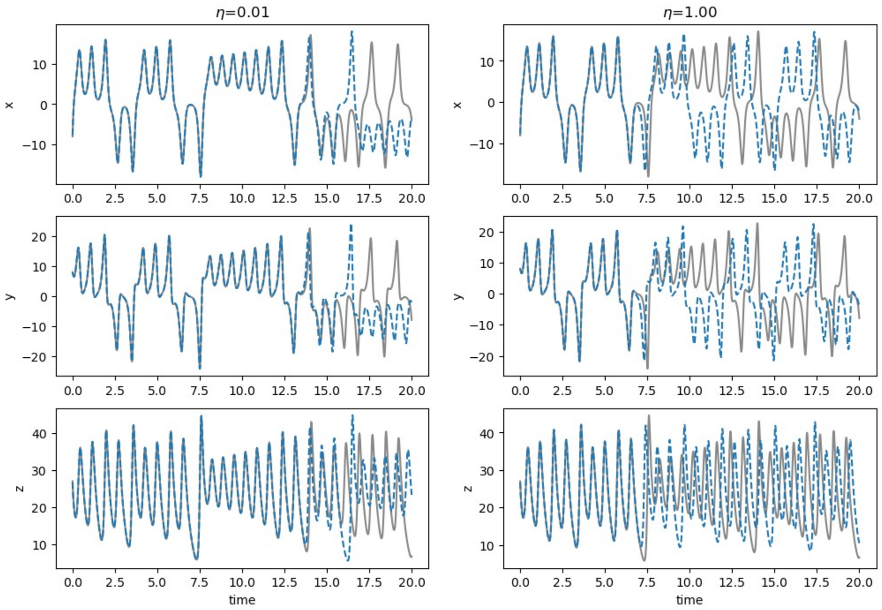
Simulations of Lorenz system with different levels of noise

The canonical Lorenz system is described by the following system of linear ODE:

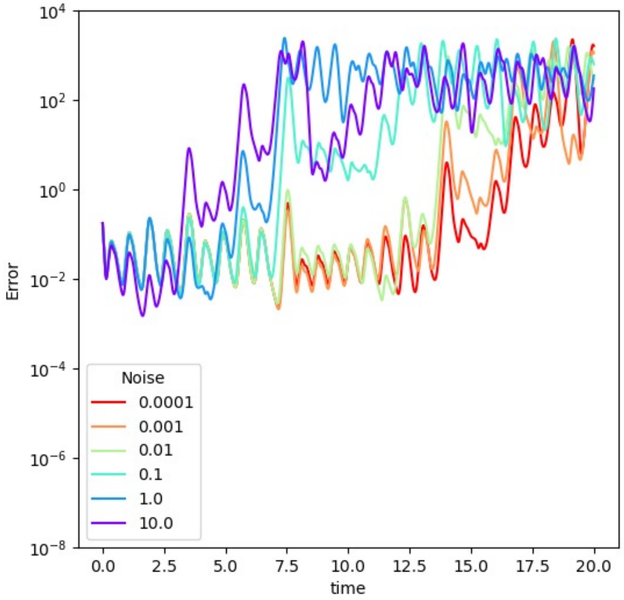
$$\begin{cases} \dot{x} = -10x + 10y \\ \dot{y} = x(28 - z) - y \\ \dot{z} = xy - \frac{8}{3}z \end{cases}$$

```
model identified with noise level = 0.01:
(x0)' = -10.000 x0 + 10.000 x1
(x1)' = 28.000 x0 + -1.000 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 1.00:
(x0)' = -9.999 x0 + 9.999 x1
(x1)' = 27.999 x0 + -0.999 x1 + -1.000 x0 x2
(x2)' = -2.667 x2 + 1.000 x0 x1
model identified with noise level = 10.00:
(x0)' = 0.053 1 + -10.007 x0 + 10.015 x1
(x1)' = 27.980 x0 + -0.982 x1 + -1.000 x0 x2
(x2)' = -2.670 x2 + 1.000 x0 x1
```

Numerical results: the Lorenz system



Evolution of each coordinate of the system: true system vs identified model



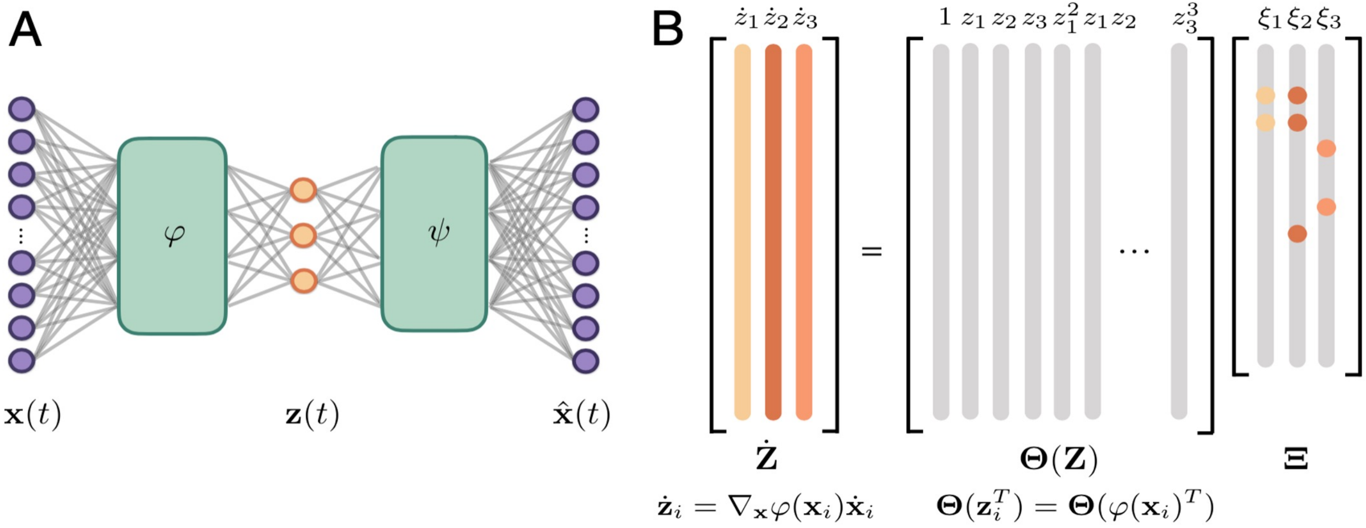
Evolution of the error against time for different noise levels

Discovering the coordinate system

- Obtaining parsimonious models is **linked** to the **coordinate system** in which the dynamics is measured.
- The **SINDy** approach assumes that the data is already measured in the coordinate system for which one has a sparse representation.
- For more complex applications of data-driven discovery, we cannot be sure that we measure the correct variables to admit a simple representation of the dynamics.
- An effective coordinate system needs to be **learnt from data**:
  - PCA: representing high-dimensional data in a low-dimensional linear subspace;
  - We need a **nonlinear** extensions of PCA → **Autoencoder**.

SINDy Autoencoders

- **Autoencoder**: feedforward neural network with a hidden layer that represents the intrinsic coordinates, trained to output an approximate reconstruction of its input. It learns a **non-linear** embedding into a reduced latent space.
- The use of NN has challenges:
  - They fail to generalize to events that are not in the training data;
  - Lack of interpretability of the resulting model.
- If we do the two steps separately, there is **NO guarantee** that the learnt intrinsic coordinates will have an associated **sparse** dynamical model.
- **SINDy autoencoder**: we require the network to learn coordinates associated with parsimonious dynamics by **simultaneously** learning a SINDy model for the dynamics of the intrinsic coordinates in a **joint optimization** using a **custom loss** function.



$$\underbrace{\|\mathbf{x}-\psi(\mathbf{z})\|_2^2}_{\text{reconstruction loss}} + \underbrace{\lambda_1 \left\|\dot{\mathbf{x}} - (\nabla_{\mathbf{z}}\psi(\mathbf{z}))\left(\boldsymbol{\Theta}(\mathbf{z}^T)\boldsymbol{\Xi}\right)\right\|_2^2}_{\text{SINDy loss in } \dot{\mathbf{x}}} + \underbrace{\lambda_2 \left\|(\nabla_{\mathbf{x}}\mathbf{z})\,\dot{\mathbf{x}} - \boldsymbol{\Theta}(\mathbf{z}^T)\boldsymbol{\Xi}\right\|_2^2}_{\text{SINDy loss in } \dot{\mathbf{z}}} + \underbrace{\lambda_3 \|\boldsymbol{\Xi}\|_1}_{\substack{\text{SINDy} \\ \text{regularization}}}$$

The model is trained using **stochastic gradient descent**, to further promote the sparsity of the model in $\boldsymbol{z}$. A sequential **thresholding** on Ξ (similar to the SINDy algorithm) is performed every 500 epochs. A refinement procedure is added to improve the final estimate.

SINDy Autoencoder Loss:

$$\|\mathbf{x}-\psi(\varphi(\mathbf{x}))\|_2^2$$

Reconstruction loss (standard autoencoder loss), to ensure that the network can reconstruct the input .

$$\left\|\nabla_{\mathbf{x}}\varphi(\mathbf{x})\dot{\mathbf{x}}-\mathbf{\Theta}(\varphi(\mathbf{x})^T)\mathbf{\Xi}\right\|_2^2$$

SINDy loss in the encoder variables, to discover the governing equations in the encoder variables.

$$\left\|\dot{\mathbf{x}}-(\nabla_{\mathbf{z}}\psi(\varphi(\mathbf{x})))\left(\mathbf{\Theta}(\varphi(\mathbf{x})^T)\mathbf{\Xi}\right)\right\|_2^2$$

SINDy loss in the original variables, to ensure the reconstruction of the original derivatives.

$$\|\mathbf{\Xi}\|_1$$

L_1 **Regularzation** term, to promote sparsity of the coefficients.

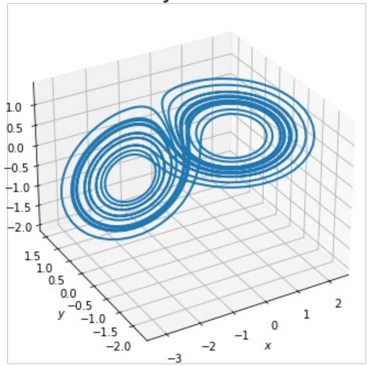
Numerical results: Lorenz system

Identified system

$$\begin{cases} \dot{x} = -6.4y^2 + 7.9z^2 \\ \dot{y} = -7.5x - 5.4y + 2.7z + 2.5xy + 2.7xz \\ \dot{z} = 6.9 + 7.1x + 4.8y - 7.7z + 2.3xy - 2.3xz \end{cases}$$

13 active terms, lower than the
81 we started from in the library

identified system attractor

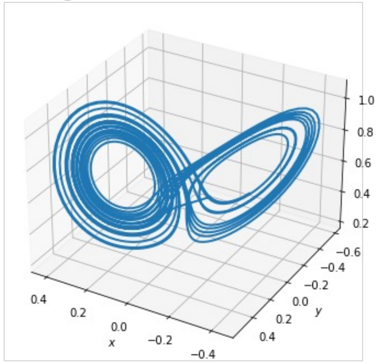


Ground truth

$$\begin{cases} \dot{x} = -10x + 10y \\ \dot{y} = x(28 - z) - y \\ \dot{z} = xy - \frac{8}{3}z \end{cases}$$

Only **7** active terms in the original system

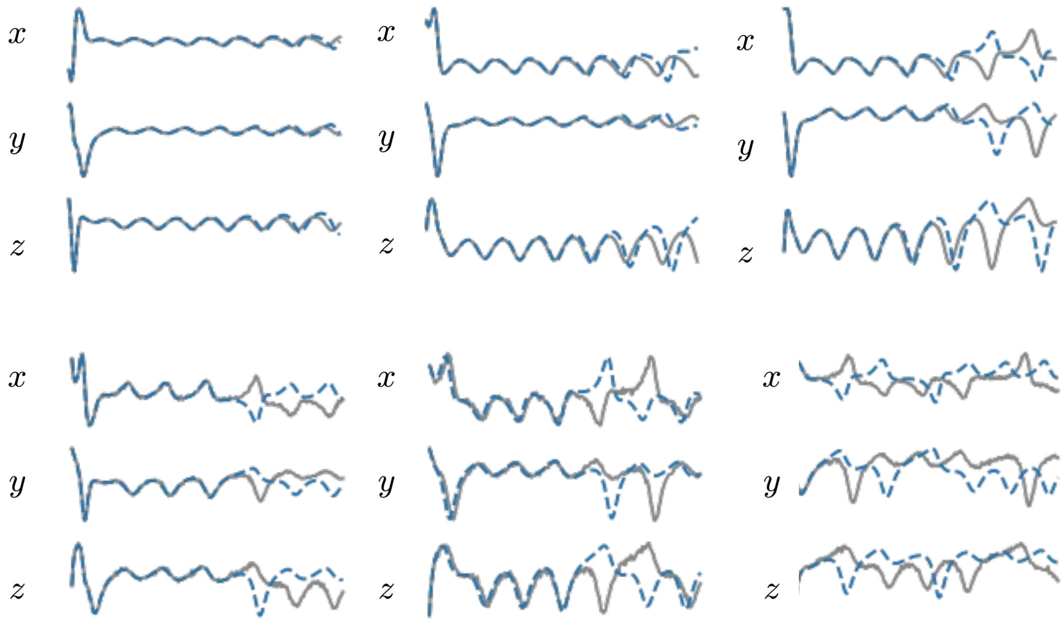
ground truth attractor



Test set analysis – In distribution

Noise level
$\eta = 0.01$

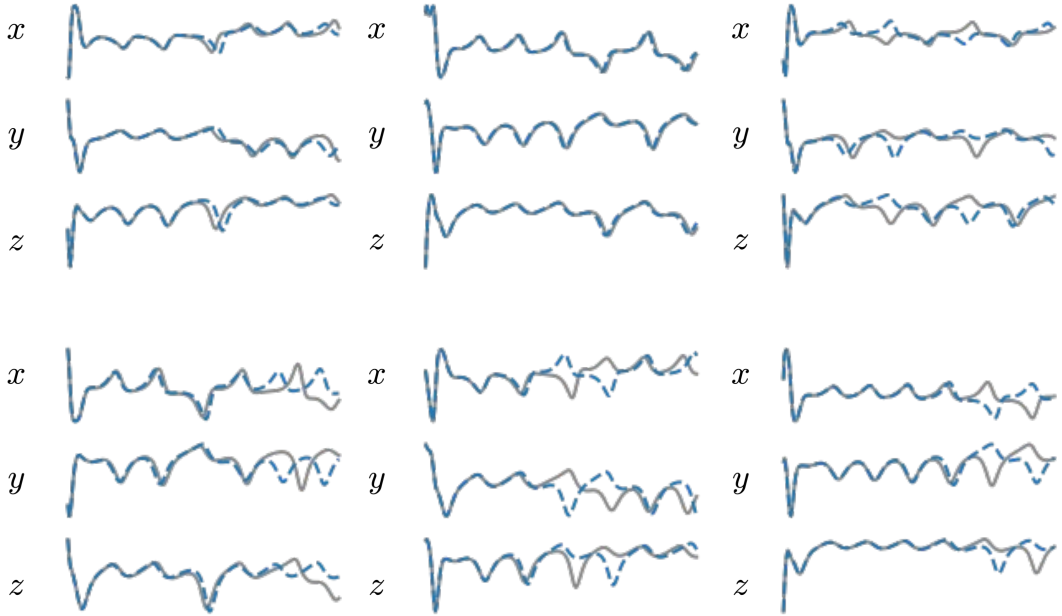
Noise level
$\eta = 0.1$



Test set analysis – Out of distribution

Noise level
$\eta = 0.01$

Noise level
$\eta = 0.1$



Hyperparameters' choice

- The most critical hyperparameter is the **number of intrinsic coordinates $\boldsymbol{d}$** in the new coordinates system:
  - Using low d the model will fail to capture the full dynamics;
  - Using high d the learned equations will be less sparse.

- $\boldsymbol{\lambda_3}$ (strength of L_1 regularization) governs the sparsity of the model, and is the most critical to tune since we need to balance sparsity and performance.

- The literature tells us that $\boldsymbol{\lambda_2}$ should generally be one order of magnitude smaller than $\boldsymbol{\lambda_1}$.

- The choice of threshold and library are critical as in the SINDy model.

Bayesian SINDy

- Instead of targeting point estimates of the SINDy coefficients, they can be estimated via **sparse Bayesian inference**.
- The data is modelled as deviations from the SINDy model prediction

$$\boldsymbol{y}_i^\top = \boldsymbol{x}_0^\top + \int_0^{t_i} \Theta(\boldsymbol{x}(t'))\Xi \, \mathrm{d}t' + \boldsymbol{\epsilon}_i.$$

- The coefficients of the models are estimated using the posterior distribution

$$p(\Xi, \boldsymbol{x}_0, \boldsymbol{\phi}|\boldsymbol{X}) \propto p(\boldsymbol{X}|\Xi, \boldsymbol{x}_0, \boldsymbol{\phi})p(\boldsymbol{\phi})p(\Xi)p(\boldsymbol{x}_0).$$

- Once the posterior has been estimated, prediction can be performed via the posterior predictive distribution
- We assume the governing equation to be **sparse**, we incorporate this information using **sparsity-inducing priors**

Bayesian SINDy

- Multiple **sparsity-inducing** priors have been proposed:
  - **Laplace** prior: Bayesian extension of the LASSO since its MAP corresponds to regression with L_1 regularization;
  - **Spike and slab** prior:

$$\xi_{i,j}|\gamma_{i,j} \sim \mathcal{N}(0, \sigma^2)\gamma_{i,j} \qquad \gamma_{i,j}|\pi_{i,j} \sim Ber(\pi_{i,j})$$

    the posterior mean of $\gamma_{i,j}$ represents the estimated **inclusion probability** of $\xi_{i,j}$ in the model.
- The posterior distribution in general is not analytically tractable, sampling-based method such as **MCMC** may be used.
- The computational **cost** of sampling **high-dimensional posterior** distribution via MCMC makes this approach unusable in the SINDy-Autoencoder.

Bayesian SINDy autoencoder

- The SINDy autoencoder can be viewed as a **parametric** model with likelihood

$$p(\mathcal{D}|\theta) \propto \exp\left(\|\mathbf{x} - g_{\theta_2}(\mathbf{z})\|_2^2 + \lambda_1 \left\|\dot{\mathbf{x}} - (\nabla_{\mathbf{z}} g_{\theta_2}(\mathbf{z}))\left(\boldsymbol{\Theta}(\mathbf{z}^T)\boldsymbol{\Xi}\right)\right\|_2^2 + \lambda_2 \left\|(\nabla_{\mathbf{x}}\mathbf{z})\,\dot{\mathbf{x}} - \boldsymbol{\Theta}(\mathbf{z}^T)\boldsymbol{\Xi}\right\|_2^2\right)$$

- As prior for the encoder and decoder part the weight initialization distribution is used, for $\boldsymbol{\Xi}$ the spike and slab prior is adopted. Estimation is performed via the **posterior distribution**

$$\pi(\theta, \gamma|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta_1)p(\theta_2)p(\Xi|\gamma)p(\gamma)$$

- To perform posterior sampling **Stochastic Gradient Langevin Dynamics** is used, a procedure that transitions from a **stochastic optimization** algorithm to a **posterior sampling** algorithm as the step size $\varepsilon^{(t)}$ decreases.

$$\Delta\theta_{t+1} = \frac{\epsilon^{(t)}}{2}\left(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(X_i|\theta_t)\right) + \eta_t, \quad \eta_t \sim \mathcal{N}(0, \epsilon^{(t)})$$

Bibliography

- Steven L. Brunton, Joshua L. Proctor, J. Nathan Kutz. *Discovering governing equations from data: Sparse identification of nonlinear dynamical systems* (Sept 2015)
- Daniel E. Shea, Steven L. Brunton, J. Nathan Kutz. Sparse Identification of Nonlinear Dynamics for Boundary Value Problems (May 2020)
- Kathleen Champion, Bethany Lusch, J. Nathan Kutz, Stevel L. Brunton. *Data-driven discovery of coordinates and governing equations* (Nov 2019)
- Seth M. Hirsh , David A. Barajas-Solano , J. Nathan Kutz. *Sparsifying Priors for Bayesian Uncertainty Quantification in Model* (Jul 2021)
- L. Mars Gao, J. Nathan Kutz. *Bayesian autoencoders for data-driven discovery of coordinates, governing equations and fundamental constants* (Nov 2022)