CS839: AI for Scientific Computing

Symbolic Regression

Misha Khodak

University of Wisconsin-Madison

12 February 2026

Outline

- **Course presentations and projects**
- **Goals of symbolic regression**
- **SINDy**
- **Limitations and extensions**
- **Reduced-order modeling**

Outline

- **Course presentations and projects**
- **Goals of symbolic regression**
- **SINDy**
- **Limitations and extensions**
- **Reduced-order modeling**

Paper presentations

- first presentation March 3rd
- 1-3 students per presentation
- talk length should scale with number of students (20-25 min per)
- sign-up sheet is up on Canvas
- please email me with a proposed topic at least a week in-advance

Course projects

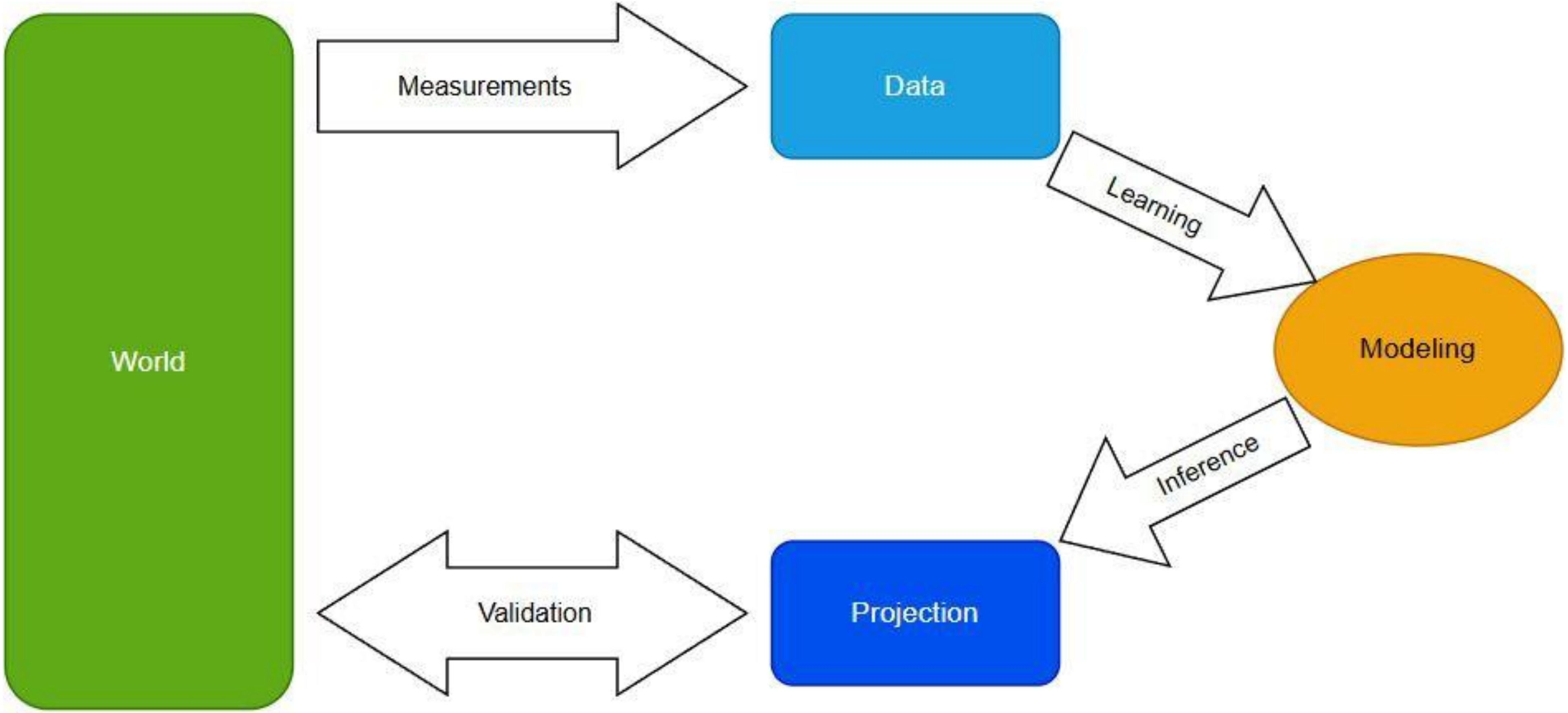
- 2-page proposal due March 27th
- can meet to brainstorm or even to discuss regularly if needed
- should involve both machine learning and scientific computing in some way, e.g.
  - an analysis of an AI approach used in scientific computing
  - an application of AI methods to a scientific tasks
- can provide topics ideas upon request

Outline

- Course presentations and projects
- **Goals of symbolic regression**
- SINDy
- Limitations and extensions
- Reduced-order modeling

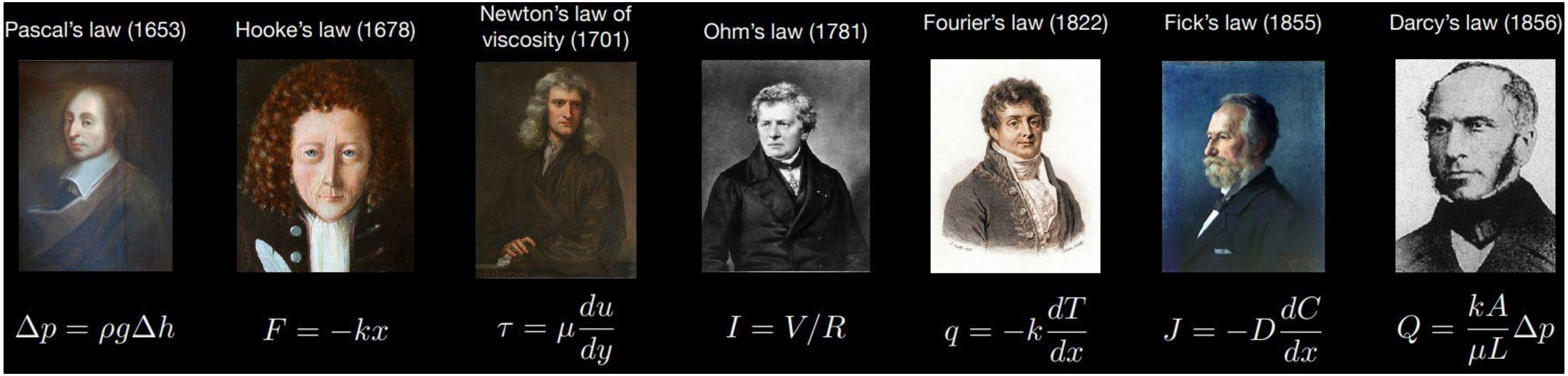
Physical modeling

- Physical modeling is a way of modeling and simulating systems that consist of real physical components
- Why use models?
  - Predict system behavior
  - Design and control processes
  - Gain insight into underlying mechanisms



Empirical Laws

- **Discovery of Physical Laws:** Based on well-designed experiments where one measurable quantity influences another.
- **Role of Simple Models:** Early discoveries relied on linear fits, assigning physical meaning to proportionality constants.
- **Limitations of Traditional Methods:** Bias toward simpler representations due to lack of high computational resources.
- **Revolution in Understanding:** Quantifying system properties and identifying relationships transformed physics.



Source: Bakarji. J, "Data, Modeling, and Inference: Empirical Laws and Linear Regression", https://www.ml4science.com/lectures

From Simple to Complex Models – Newton's Law to Navier-Stokes

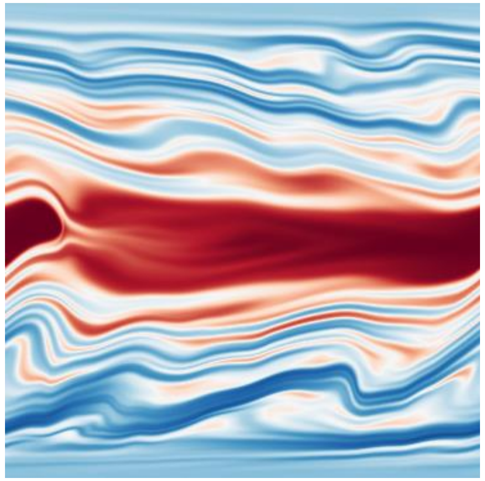
- Historically, scientists favored well-posed linear equations with minimal variables, using simple building blocks and avoiding complex interactions.

$$\underbrace{\rho}_{m}\underbrace{\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right)}_{a} = \underbrace{-\nabla p + \nabla\cdot\boldsymbol{\tau} + \mathbf{f}}_{F}$$

- The rise of powerful computers allowed scientists to solve nonlinear equations, overcoming the limitations of analytical approaches.

Data Collection for Modeling

Wikipedia



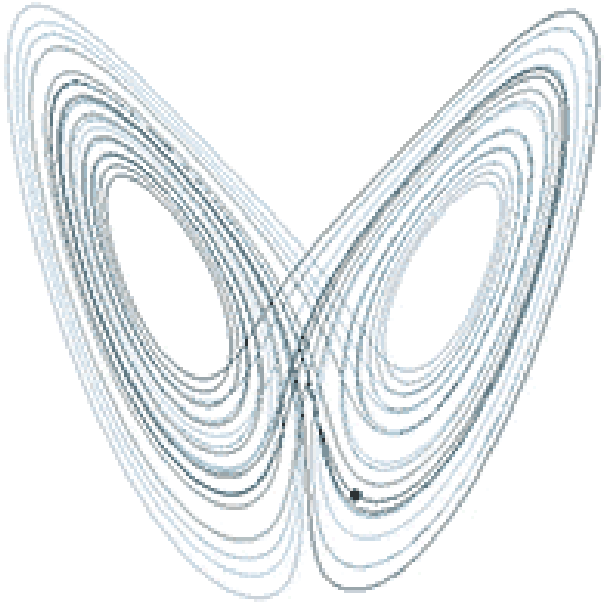
Polymathic AI

- Types of Data:
  - Time-series data: Sequential observations over time
  - Spatial-temporal data: Field measurements (e.g., weather patterns)
    - High-dimensional data: Complex systems (e.g. turbulence)
- Key Considerations:
  - Sampling rate: How often do we measure?
  - Data noise: How do we handle uncertainties?
  - Data preprocessing: Filtering, normalization, and interpolation

Observing Dynamical Systems

A dynamical system evolves over time based on a set of rules (differential equations).

- generic time–dependent model $\dot{x} = f(x)$
  - The (generally nonlinear) term f governs the temporal evolution of the system
- Methods of Observation:
  - Direct measurements (e.g., sensors, cameras)
  - Indirect estimation (e.g., Kalman filters, state observers)



Wikipedia

What are Data-Driven Models?

Data-driven models use observed data to infer system behavior rather than relying solely on first-principles physics.

We want to do this to:

- make predictions (neural operators)

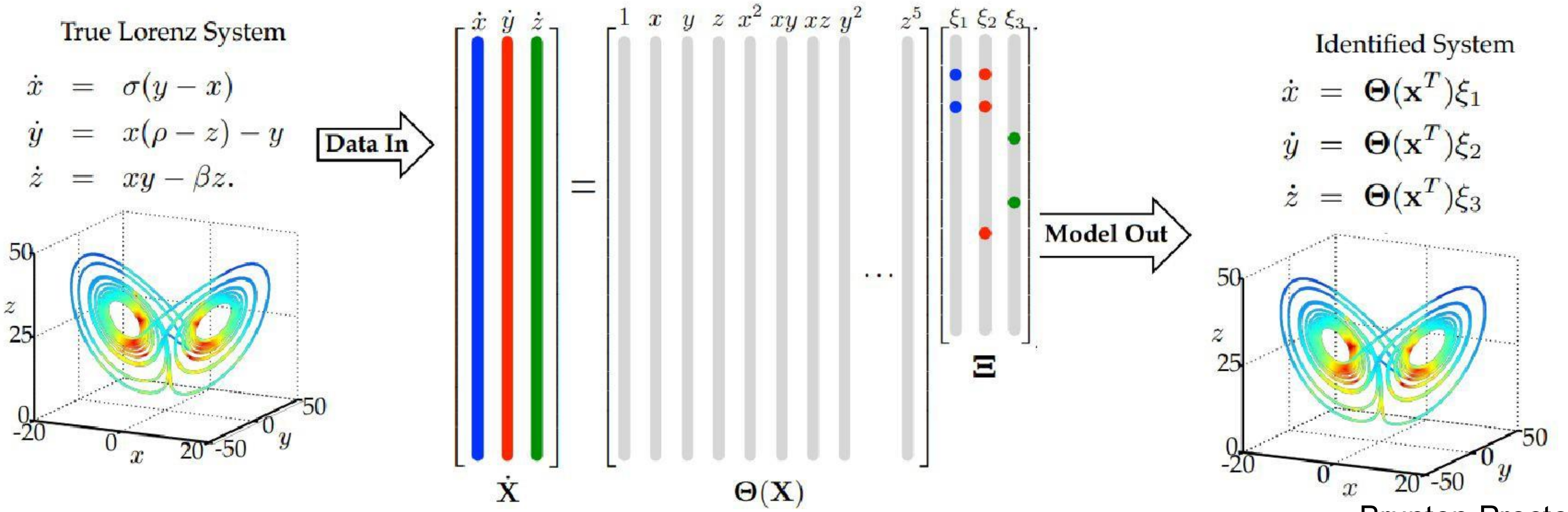
- **discover governing equations (symbolic regression)**

Outline

- **Course presentations and projects**
- **Goals of symbolic regression**
- **SINDy**
- **Limitations and extensions**
- **Reduced-order modeling**

Sparse Identification of Non-linear Dynamics

- SINDy is a tool for modeling of data driven dynamics
  - It based on the idea that dynamical systems can be modeled with just few terms
  - The sparse identification aims for the most relevant terms of the system
  - The Non linearities are represented by a collection of non linear function called a **Library**
- **Key Idea:**
  - Represent the system as a sparse combination of candidate functions.
  - Identify the most relevant terms using sparse regression.



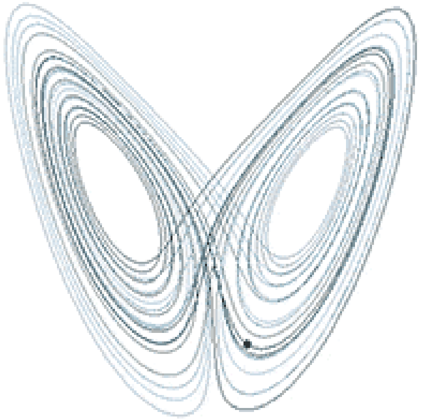
Brunton-Proctor-Kutz, PNAS(2016)

Mathematical Formulation

- Data: a series of measurements $\mathbf{x}(t) \in \mathbb{R}^d$
- We assume that the dynamics are modeled as $\dot{x} = f(x)$
- The key idea behind SINDy is that the function f is often **sparse** in the space of an appropriate set of basis functions
- To apply SINDy in practice one needs a **set of measurement** data collected at times $t_1, \ldots, t_n$, and the time derivatives of these measurements (either measured directly or **numerically approximated**).

$$X = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_n(t_m) \end{bmatrix}, \quad \dot{X} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \cdots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \cdots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \cdots & \dot{x}_n(t_m) \end{bmatrix}$$

Step 1: Data Collection



Measurement

$$X = \begin{bmatrix} x(t_0) & y(t_0) & z(t_0) \\ x(t_1) & y(t_1) & z(t_1) \\ \vdots & \vdots & \vdots \\ x(t_n) & y(t_n) & z(t_n) \end{bmatrix}$$

Data representation

Since derivative of the data is usually unknown, the main strategy is to use centered finite difference:

$$X = \begin{bmatrix} x(t_0) & y(t_0) & z(t_0) \\ x(t_1) & y(t_1) & z(t_1) \\ \vdots & \vdots & \vdots \\ x(t_n) & y(t_n) & z(t_n) \end{bmatrix} \rightarrow \boxed{\dot{x}(t_i) \approx \frac{x(t_{i+1}) - x(t_{i-1})}{2\Delta t}} \rightarrow \dot{X} = \begin{bmatrix} \dot{x}(t_0) & \dot{y}(t_0) & \dot{z}(t_0) \\ \dot{x}(t_1) & \dot{y}(t_1) & \dot{z}(t_1) \\ \vdots & \vdots & \vdots \\ \dot{x}(t_n) & \dot{y}(t_n) & \dot{z}(t_n) \end{bmatrix}$$

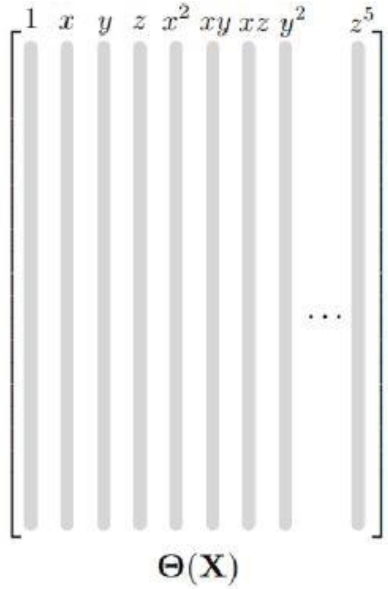
Mathematical Formulation

For approximating the function f we create a **library matrix** or **dictionary matrix** where the columns consists of a set of chosen basis functions applied to the data

$$\Theta(X) = \begin{bmatrix} | & | & & | \\ \theta_1(X) & \theta_2(X) & \cdots & \theta_l(X) \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times l}$$

One example of library (vector products are understood element-wise):

$$\Theta(X) = \begin{bmatrix} | & | & | & | & | & | & | & | & & | \\ 1 & x_1 & x_2 & x_3 & x_1^2 & x_1 x_2 & x_1 x_3 & x_1^2 & \cdots & x_3^5 \\ | & | & | & | & | & | & | & | & & | \end{bmatrix}$$



Mathematical Formulation

The goal of SINDy is to find a set of **sparse** coefficients $\Xi = \begin{bmatrix} | & | & \cdots & | \\ \xi_1 & \xi_2 & \cdots & \xi_n \\ | & | & \cdots & | \end{bmatrix}$

A **parsimonious model** will provide an **accurate model fit** with as **few terms** as possible in the approximation problem underlying SINDy:

$$\dot{X} \approx \Theta(X)\Xi$$

The problem of identifying the dynamical system thus becomes a **linear regression problem:** sparse regression techniques are used to promote solutions where many elements of Ξ are zero.

Sparse Regression

An approach for a sparse solution consists of solving the following optimization problem:

$$\Xi = arg\min_{W} \left\|\dot{X} - \Theta(X)W\right\|_2^2 + \lambda\mathcal{R}(W)$$

where $\mathcal{R}(W)$ is a sparse promoting regularizer. Since we seek a solution with many zeros in the coefficient, the ℓ_0-norm is the most natural regularizer, leading to the optimization problem:

$$\Xi = arg\min_{W} \left\|\dot{X} - \Theta(X)W\right\|_2^2 + \lambda\|W\|_0$$

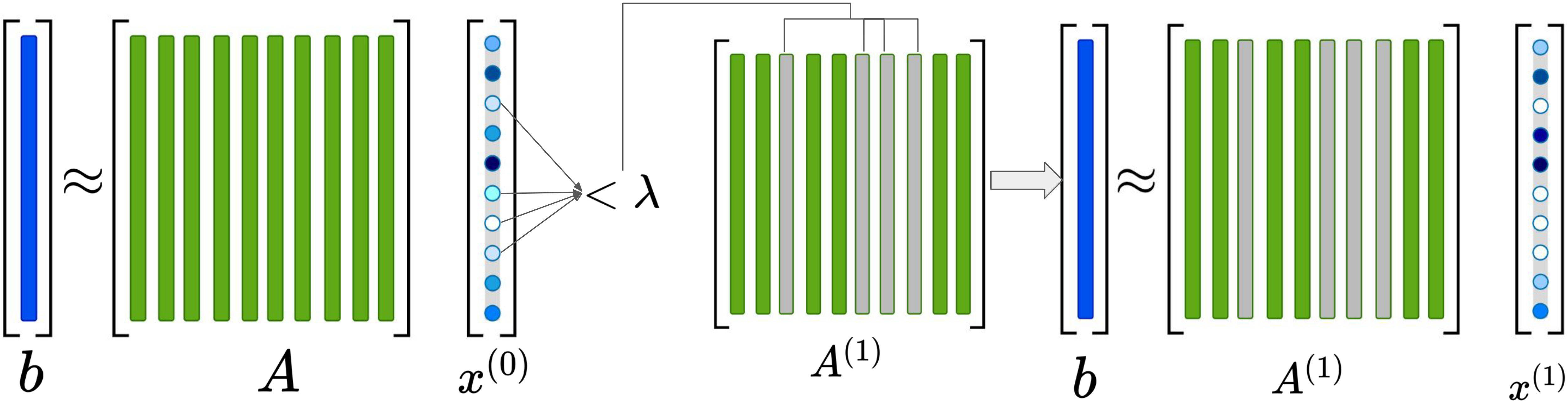
Since the problem would require an intractable combinatorial search, $\|\cdot\|_1$ norm regularization (LASSO) or a **sequential thresholding** procedure is used in place of the zero norm.

Sparse Regression with STLS

Suppose we want to solve the STLS for $\arg\min_x ||Ax - b||_2^2$

1. Solve the least square problem: $x^{(0)} = (A^T A)^{-1} A^T b = A^\dagger b$

2. Given a threshold λ, for every element where $|x_i^{(0)}| < \lambda$ we remove (put zeros) in the correspondent column of the matrix A (i-th column)
3. Repeat the steps 1 and 2 until no more columns are eliminated



Sparse Regression with STLS

- What is sequentially thresholded least squares algorithm?
  - A regression technique used in SINDy to find sparse representations of dynamical systems.
  - It iteratively **eliminates small coefficients**, enforcing sparsity in the discovered equations.
  - (ℓ_0 minimization) This algorithm produces a local minimizer of the cost function (Zhang-Schaeffer,2018):

$$\xi \mapsto \|A\xi - \mathbf{b}\|_2^2 + \lambda\|\xi\|_0$$

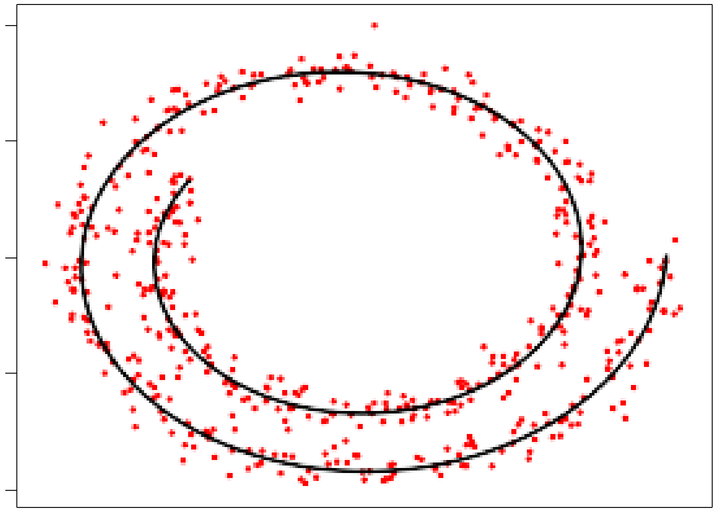
- Advantages of STLS
  - (Convergence) this algorithm terminates in finite steps
  - Faster than LASSO for some cases.
  - Provides interpretable, sparse solutions.
  - Easily adaptable to different basis functions

Step-by-Step Implementation of SINDy

1. Collect Data
   - Measure state $\mathbf{x}(t)$ variables over time.
   - Compute time derivatives $\dot{\mathbf{x}}$ (numerically if needed).
2. Construct Feature Library $\Theta(\mathbf{x})$
   - Example basis functions: $\Theta(\mathbf{x}) = \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_1 x_2 & x_2^2 & \sin(x_1) & \cos(x_2) \end{bmatrix}$
   - Choose basis functions relevant to the system (polynomials, trigonometric functions, etc.)
3. 3. Solve Sparse Regression Problem
   - $\Xi = arg \min_{\Xi'} ||\dot{X} - \Theta(X)\Xi'||_2 + \lambda \mathcal{R}(\Xi')$
   - Use LASSO or sequential thresholding.

Example application

Ground truth:



$$\begin{cases} \dot{x} = -0.1x + 2y \\ \dot{y} = -2x - 0.1y \end{cases}$$

STLS with $\lambda = 0.05$ and a polynomial library:

poly.-approx. derivatives:

$$\begin{cases} \dot{x} = -0.082x + 1.975y \\ \dot{y} = -1.972x - 0.110y \end{cases}$$

TV-regularized derivatives:

$$\begin{cases} \dot{x} = -0.082x + 1.975y \\ \dot{y} = -1.972x - 0.110y \end{cases}$$

Outline

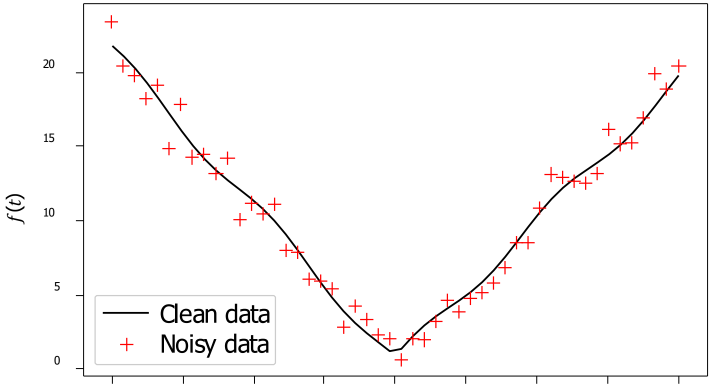
- **Course presentations and projects**
- **Goals of symbolic regression**
- **SINDy**
- **Limitations and extensions**
- **Reduced-order modeling**

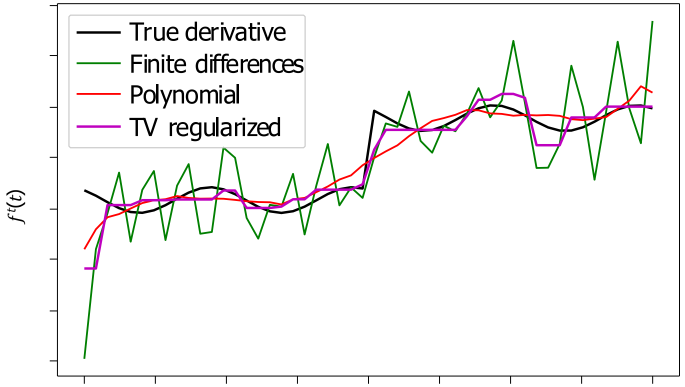
Symbolic regression challenges

- Complexity grows combinatorially with dimension d
  - the set of polynomials of order k is $\binom{k+d}{d}$, which is 462 for $\mathrm{d} = 6, k = 5$
  - one solution: reduce the model order via truncated SVD

$$X = U_r \Sigma_r V_r^{\top} \ \rightarrow x \approx U_r a \quad \text{for} \quad a \in \mathbb{R}^r, r \ll d$$

- Finite difference approximations of derivatives can be very poor:





- Choice of function library / coordinate system requires expertise
- Real data is much noisier than dynamical system + i.i.d. noise

Challenges with Real-World Applications

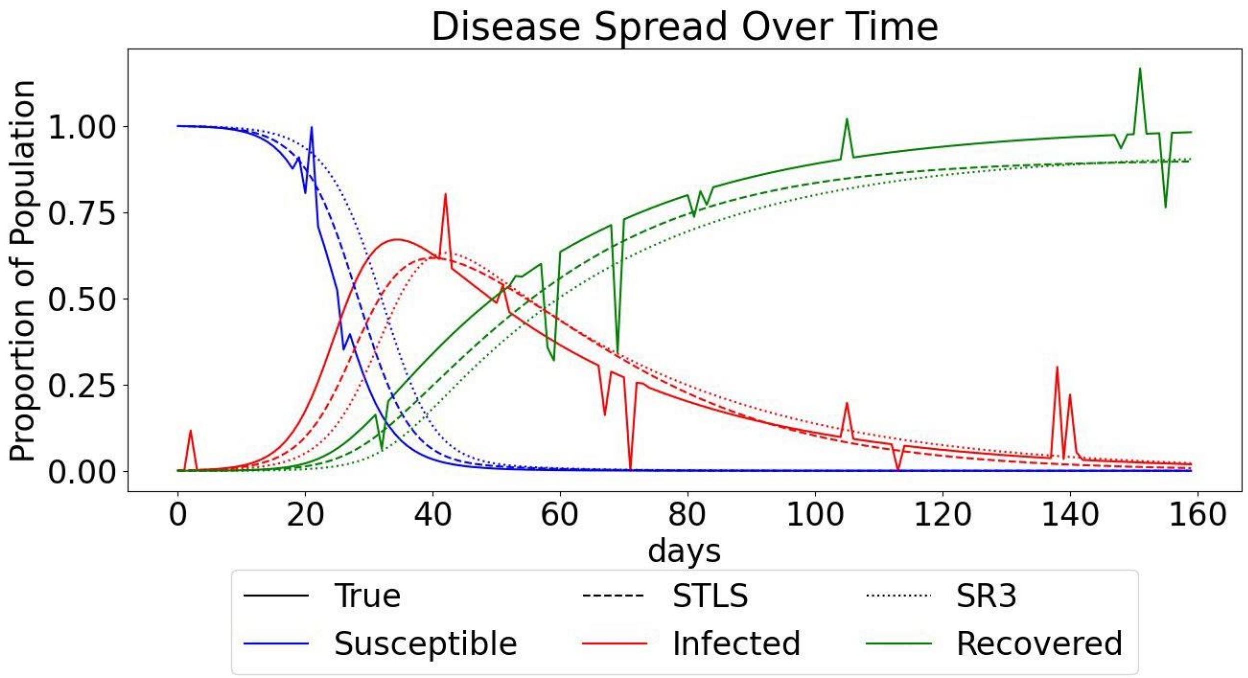
- Sensor scaling or resolution constraints.
  - Noise
  - Non-uniform time steps
- Incomplete state observation is common due to economic or practical constraints.
  - Missing data
- Library Selection:
  - Missing Essential functions
  - Incorrect parametrization (e.g. polynomial degree or frequency).
- Numerical precision
  - Differentiation scheme for the derivatives
  - smoothness of the observed trajectory

Real world data

The real-world situation of collecting and analyzing COVID-19 data posed unique and unprecedented challenges. A major issue was the **presence of outliers**, particularly additive outliers—unexpected, abrupt changes in data values that do not fit the usual trend or seasonal pattern.

Examples in the COVID-19 context:

- Sudden reporting backlogs released in a single day (e.g. thousands of **previously unreported cases**).
- **Underreporting** during weekends or holidays, followed by compensating spikes.
- Changes in testing policies (e.g. **expanding testing capacity**).
- Data corrections or revisions, such as **removing duplicates** or **false positives**



SINDy Variants

There are many follow up works that try to adapt SINDy to overcome some of those challenges, for example:

- Weak SINDy – Champion et al. (2020), SIAM
  - Uses integral formulations to reduce sensitivity to noise.
- Ensemble SINDy – Fasel et al. (2022), Proc. R. Soc. A
  - Averages multiple sparse models to handle missing and noisy data.
- Bayesian SINDy – Fung et al. (2025), Proc. R. Soc. A
  - Incorporates uncertainty quantification through probabilistic modeling.
- ADAM-SINDy – Schaeffer et al. (2024), ArXiv
  - Adapts candidate functions and coefficients using adaptive optimization.

Comparing SINDy Variants

Method	**Handles**	**Key Idea**	**Pros**	**Cons**
Weak SINDy	Noisy data	Integrates equations to smooth noise	No differentiation errors, robust to noise	Requires choosing test functions
E-SINDy	Limited Data	Uses an ensemble of SINDy models on different subsets	Works without imputing missing data	Higher computational cost
Bayesian SINDy	Uncertainty, sparse data	Models coefficients as probability distributions	Quantifies uncertainty	Computationally expensive
ADAM SINDy	Unknown library functions	Optimize the library and coefficients at the same time	Flexible function library (fraction exponents)	Non-convex optimization

Outline

- Course presentations and projects
- Goals of symbolic regression
- SINDy
- Limitations and extensions
- **Reduced-order modeling**

Motivation for Reduced-Order Modeling (ROM)

High-dimensional models are expensive to simulate and model

Many high-dimensional systems exhibit low-dimensional structure:

- Given data in a high dimension $\boldsymbol{x} \in \mathbb{R}^D$
- Identify reduced coordinates $z \in \mathbb{R}^d \ (d \ll D)$ where $z = \phi(x)$
- Find a reconstruction function $\boldsymbol{x} = \psi(z)$
- Examples: PCA/POD, Autoencoders, Galerkin projection

Objective: find low-dimensional structure that captures essential dynamics.

Proper Orthogonal Decomposition (POD)

- Decompose the vector field $\mathbf{u}(\mathbf{x}, t)$ into a set of spatial function $\Phi_k(\mathbf{x})$ and time coefficients $a_k(t)$

$$\mathbf{u}(\mathbf{x}, t) = \sum_{k=1}^{\infty} a_k(t)\Phi_k(\mathbf{x})$$

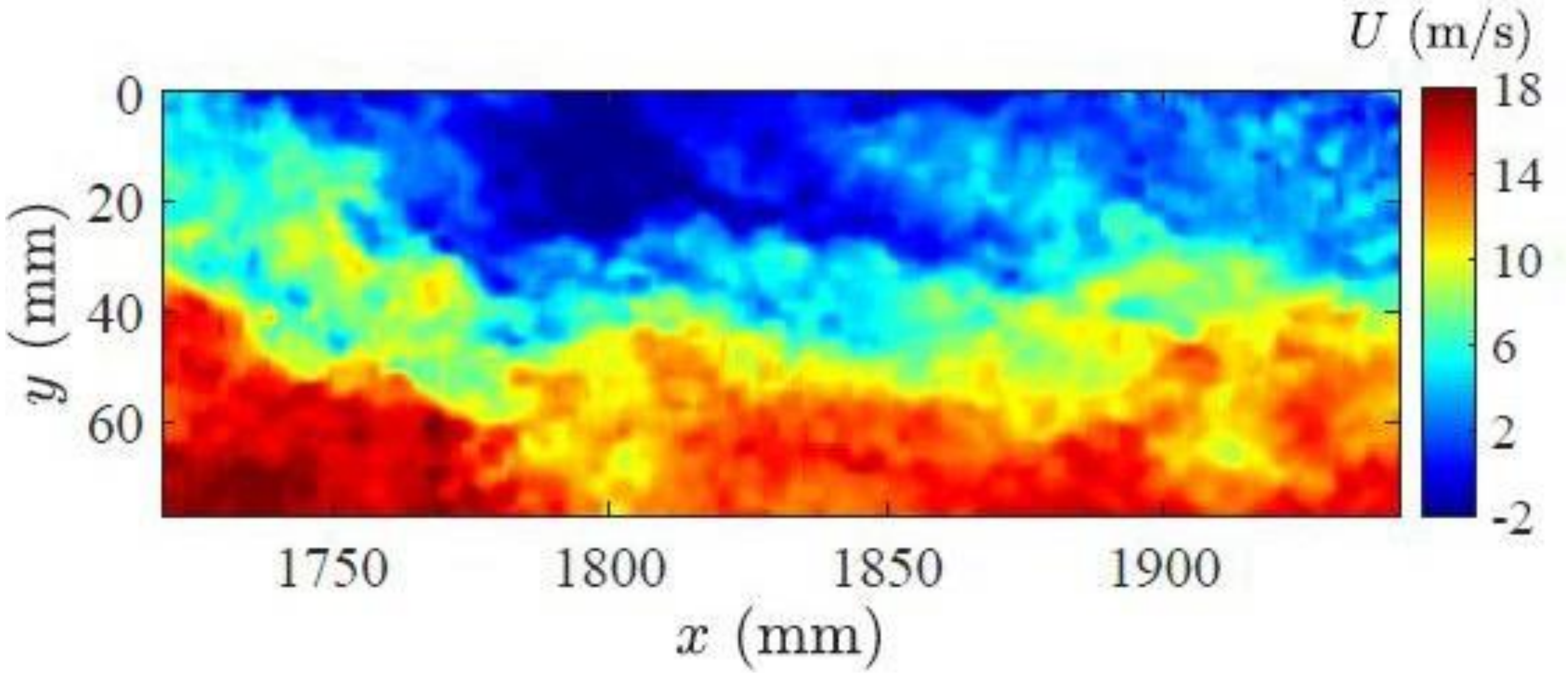
- **Proper** in the sense that the sequence $\sum_{k=1}^{n} a_k(t)\Phi_k(\mathbf{x})$ maximizes the energy that can be captured by the first *n* spatial modes
- **Orthogonal** comes from the fact that the modes are orthonormal

$$\langle \Phi_{k_1}, \Phi_{k_2} \rangle = \begin{cases} 1 & \text{if } k_1 = k_2, \\ 0 & \text{if } k_1 \neq k_2 \end{cases}$$

- This is just PCA!

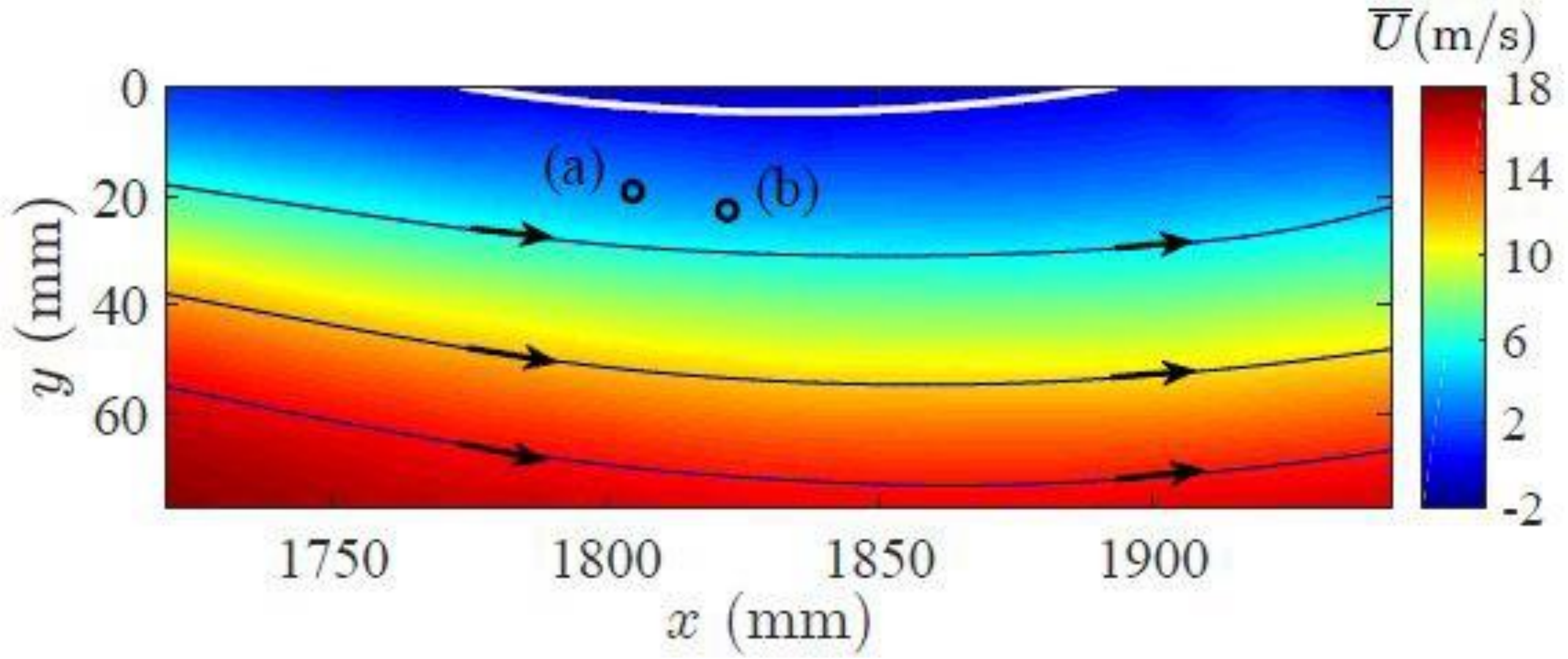
Proper Orthogonal Decomposition (POD)

- for data driven analysis, we should consider a finite dimensional basis
- let’s consider an 2D example of a velocity field



Proper Orthogonal Decomposition (POD)

- we can collect data for a number of snapshots in different spatial points
- let's consider the two points shown below:



Proper Orthogonal Decomposition (POD)

- Our data set will be a collection of snapshots taken from the control points

$$S = \begin{pmatrix} U_a(t_1) & U_b(t_1) \\ U_a(t_2) & U_b(t_2) \\ \vdots & \vdots \\ U_a(t_m) & U_b(t_m) \end{pmatrix}$$

- Since we are mostly interested in the dynamics, we usually start by removing the average of the respective columns:

$$\bar{U}_i = \frac{1}{m} \sum_{j=1}^{m} U_i(t_j),\ i = a, b$$

$$u'_a(t) = U_a(t) - \bar{U}_a$$

$$u_b(t) = U_b(t) - \bar{U}_b$$

Proper Orthogonal Decomposition (POD)

- The new dataset is given by the matrix:

$$U = \begin{pmatrix} u'_a(t_1) & u'_b(t_1) \\ u'_a(t_2) & u'_b(t_2) \\ \vdots & \vdots \\ u'_a(t_m) & u'_b(t_m) \end{pmatrix}$$

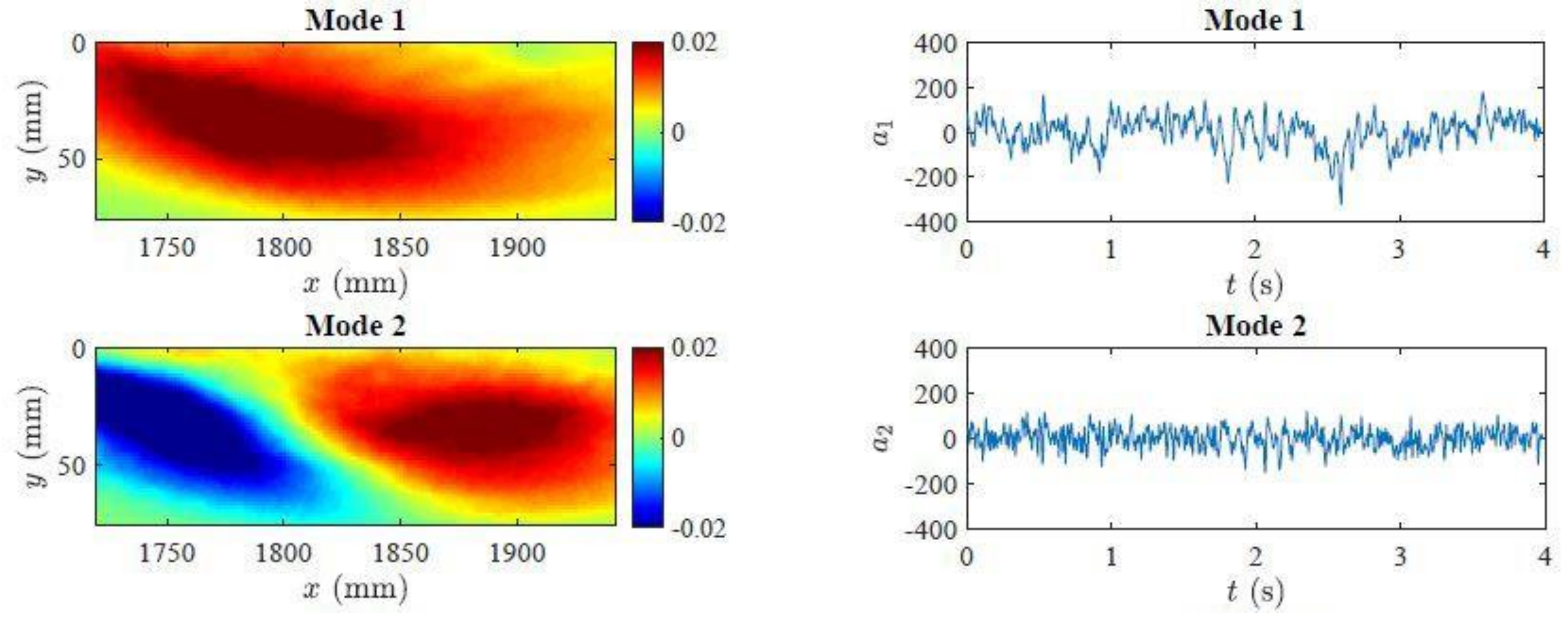
- Because the data comes from the same dynamics, there is a correlation in the data. This correlation can be observed by the covariance matrix

$$C = \frac{1}{m-1} U^T U = \frac{1}{m-1} \begin{pmatrix} \sum_{i=1}^m u_a'^2(t_i) & \sum_{i=1}^m u'_a(t_i) u'_b(t_i) \\ \sum_{i=1}^m u'_b(t_i) u'_a(t_i) & \sum_{i=1}^m u_b'^2(t_i) \end{pmatrix}$$

- The **POD modes** ϕ along which the projection $U\phi$ is maximized are just the eigenvectors of this matrix (as in PCA)

POD - Snapshots method

- POD is essentially symmetric in time and space, since there is no fundamental difference between temporal and spatial variables
- Thus we can also treat time as the deterministic basis and space as the varying coefficients



POD as a reduction method

- In a finite dimension decomposition (finite data) the POD can be written as:

$$\mathbf{u}(\mathbf{x},t) = \sum_{k=1}^{q} a_k(t)\Phi_k(\mathbf{x}) \qquad q = \min(m,n) \qquad U \in \mathbb{R}^{n\times m}$$

  where the coefficients a_k are associated with the singular values of U

- For convention, the singular values are ordered in decreasing order; as the value of the singular value decreases, less information the associated coefficients carry

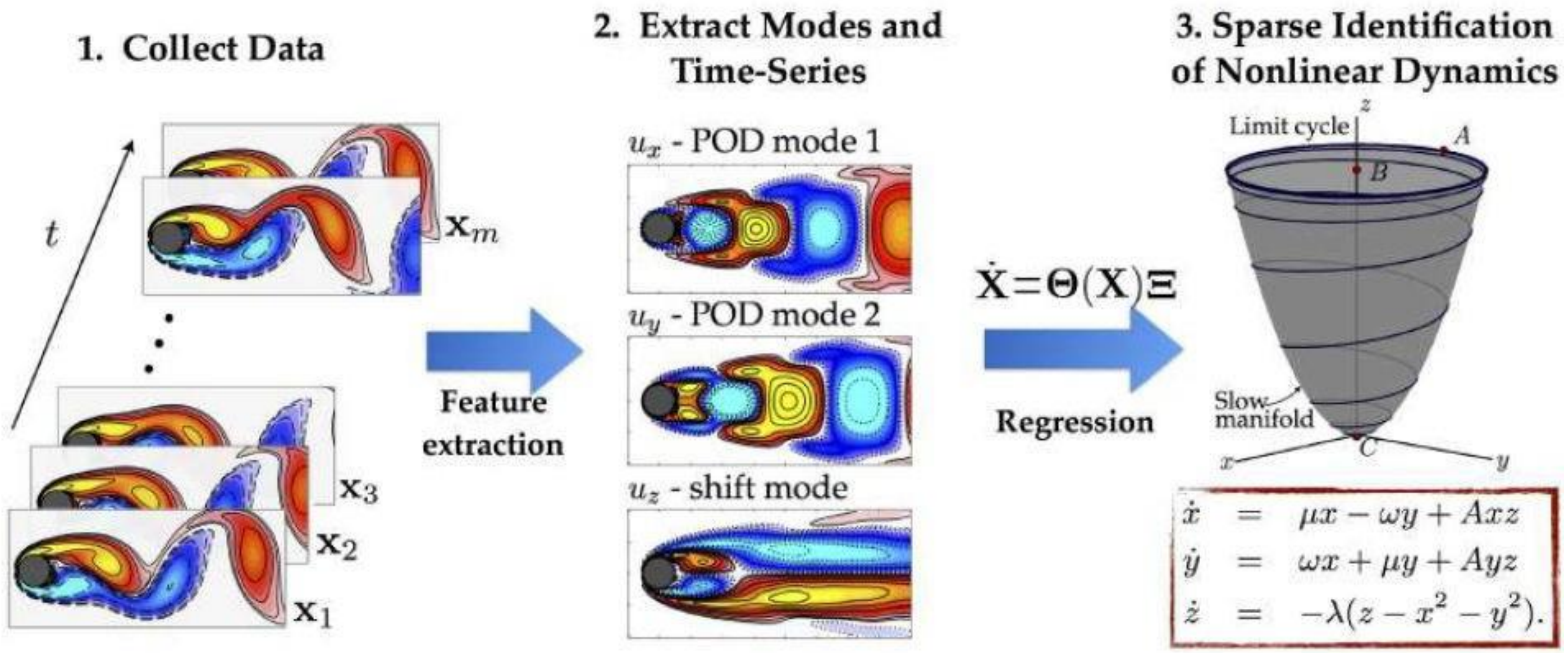
- The state space can be approximated by taking only a subset of the coefficients:

$$\mathbf{u}(\mathbf{x},t) \approx \sum_{k=1}^{r} a_k(t)\Phi_k(\mathbf{x}),\ r \ll q$$

SINDy with POD

Once we collect the POD modes from data, we can reduce all dynamics into a few time-varying scalar functions multiplied by the temporal modes: $\mathbf{u}(\mathbf{x},t) \approx \sum_{k=1}^{r} a_k(t)\Phi(\mathbf{x})$

$$\mathbf{a} = \begin{pmatrix} a_1(t_1) & a_2(t_1) & \dots & a_r(t_1) \\ a_1(t_2) & a_2(t_2) & \dots & a_r(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ a_1(t_m) & a_2(t_m) & \dots & a_r(t_m) \end{pmatrix}$$



We expect that the temporal modes follow a dynamical system modeled by:

$$\dot{\mathbf{a}} = f(\mathbf{a}) = \Theta(\mathbf{a})\Xi$$

Brunton, S. L.., Kutz, J. N., Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control. Índia: Cambridge University Press, 2022.

SINDy with POD

- Once the have the desired dynamical system, we can apply any SINDy variant to fit the data:

$$\Xi = \underset{\Xi}{arg\min} ||\dot{\mathbf{a}} - \Theta(\mathbf{a})\Xi||_2^2 + \lambda||\Xi||_1$$

- Once the low order dynamical system is discovered, we can solve it in time and apply the spatial modes for the **reconstruction** in the high order space

$$\mathbf{u}(\mathbf{x}, t) \approx \sum_{k=1}^{r} a_k(t)\Phi(\mathbf{x})$$

POD Limitations

- POD is a **linear reduction**
  - linear modes have been used for a long time for such problems for their ease of computation and their clear physical meaning
  - may need a prohibitive number of modes to reach convergence

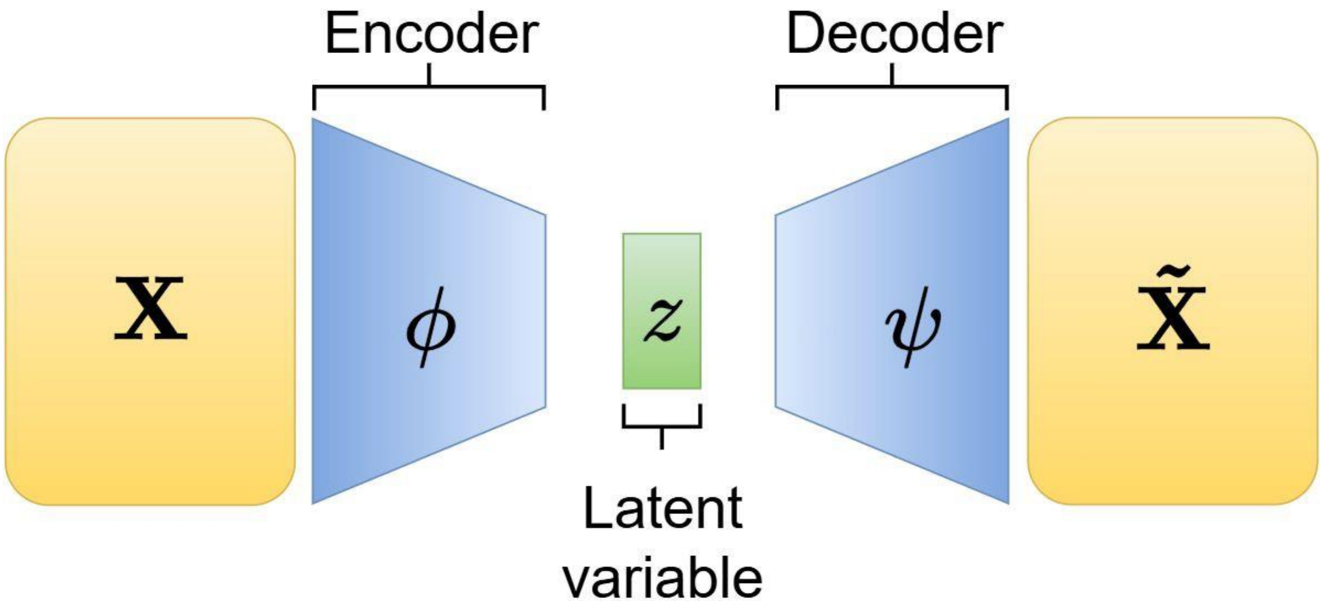
- Many systems live on **nonlinear manifolds** and can benefit from nonlinear reduction methods

- Autoencoders allows a nonlinear reduction
  - Autoencoders are neural networks that compress data (**encoding**) into a latent space and then reconstruct it back (**decoding**) to the original space.
  - They are nonlinear in the sense that they are based on defining new variables that are nonlinearly related to the initial ones

Autoencoders

Autoencoders are instances of encoder–decoder networks trained to reconstruct their input after having reduced it to a **lower dimension.**

- Encoding $\phi(\mathbf{X}) = z$
- Decoding $\psi(z) = \tilde{\mathbf{X}}$



The weights from autoencoders are trained by solving the minimization problem:

$$\mathbf{w} = arg\min \|\mathbf{X} - \psi(\phi(\mathbf{X}))\|_2^2$$

In standard autoencoders, no information about the latent variable z is known, therefore autoencoders are considered an **unsupervised** reduction method

Challenges of Using Neural Networks for Dynamical Systems

- **Extrapolation:** NNs have been successful on datasets that are fundamentally interpolatory

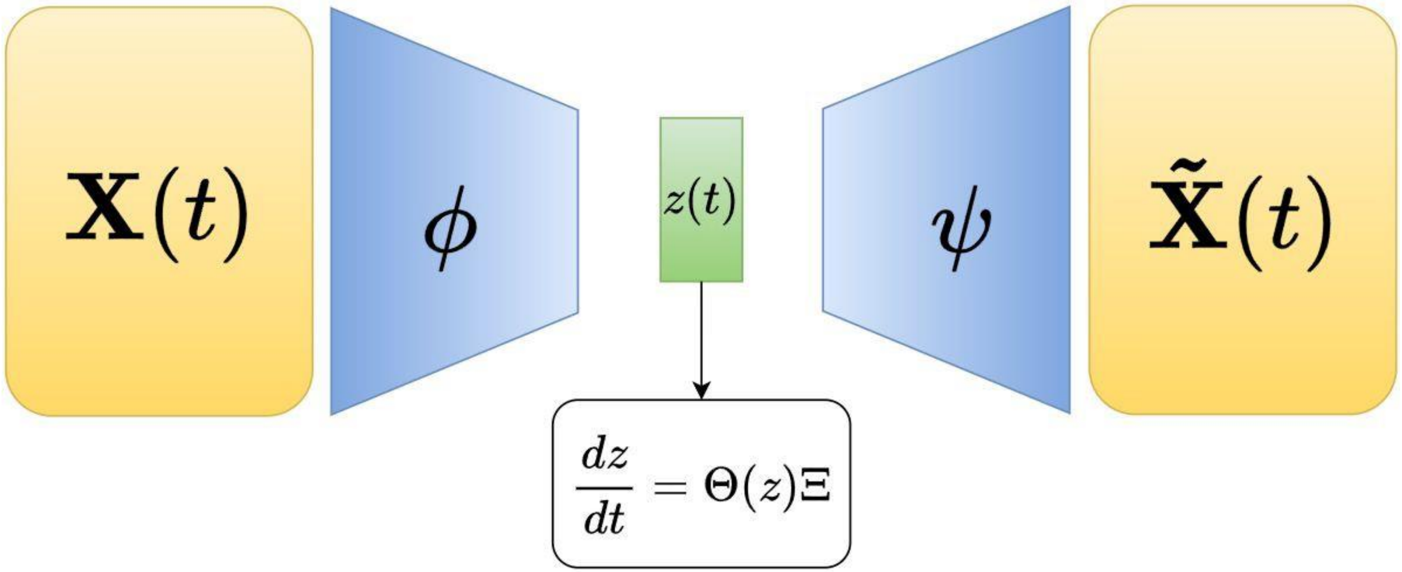
- **Interpretability:** NN are typically complicated with the number of parameters far exceeding the original dimension of the dynamical system

- **Generalization:** The lack of interpretability also makes it difficult to generalize models to new datasets and parameter regimes

SINDy Autoencoder

Goal: leverage the parsimony and interpretability of SINDy with the **universal approximation** capabilities of deep neural networks to produce **interpretable** and **generalizable models** capable of extrapolation and forecasting.

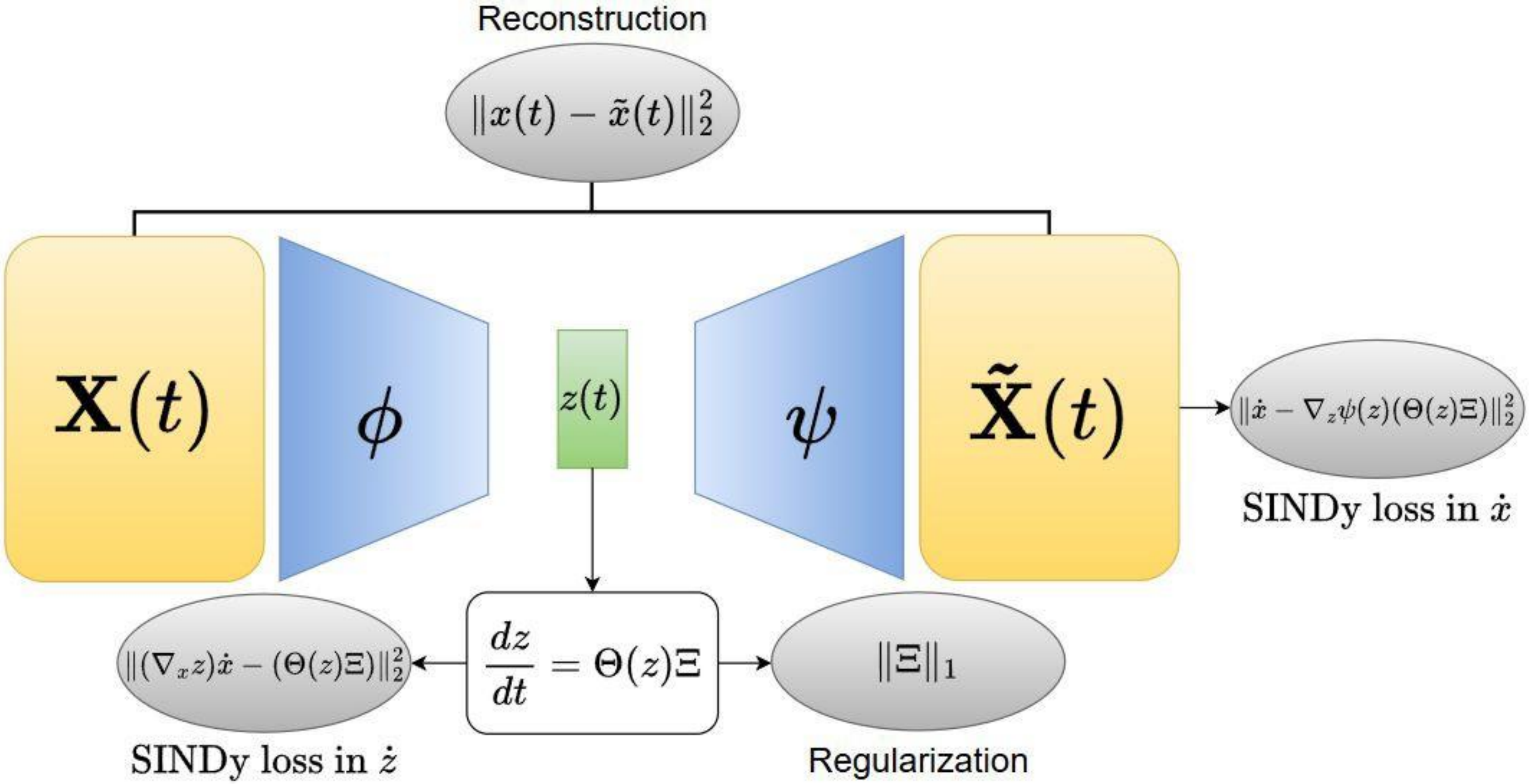
- Combine autoencoders with SINDy
  - Encoder: $\mathbf{z} = \phi(\mathbf{x})$
  - Decoder: $\hat{\mathbf{x}} = \psi(\mathbf{z})$
  - dynamics: $\dot{\mathbf{z}} = \Theta(\mathbf{z})\Xi$
- Simultaneously learn:
  - Nonlinear coordinates
  - Sparse dynamical system



Coefficients and weights fitting

In SINDy-AE, two sets of parameters must be optimized simultaneously:

- Autoencoder Parameters: $\mathbf{w} = \{W^{[\ell]}, \mathbf{b}^{[\ell]}\}_{\ell=1}^{L}$
- SINDy Coefficients: Ξ



SINDy-AE optimization

The loss function encourages the network to minimize both the autoencoder reconstruction error and the SINDy loss in $\mathbf{z}$ and $\mathbf{x}$:

$$\underset{\Xi,\,\mathbf{w}}{arg\min} \quad \underbrace{\|\mathbf{x}-\psi(\mathbf{z})\|_2^2}_{\text{reconstruction loss}}$$

$$+\underbrace{\lambda_1\|\dot{\mathbf{x}}-(\nabla_{\mathbf{z}}\psi(\mathbf{z}))(\Theta(\mathbf{z}^T)\Xi)\|_2^2}_{\text{SINDy loss in }\dot{\mathbf{x}}}$$

$$+\underbrace{\lambda_2\|(\nabla_{\mathbf{x}}\mathbf{z})\dot{\mathbf{x}}-\Theta(\mathbf{z}^T)\Xi\|_2^2}_{\text{SINDy loss in }\dot{\mathbf{z}}}$$

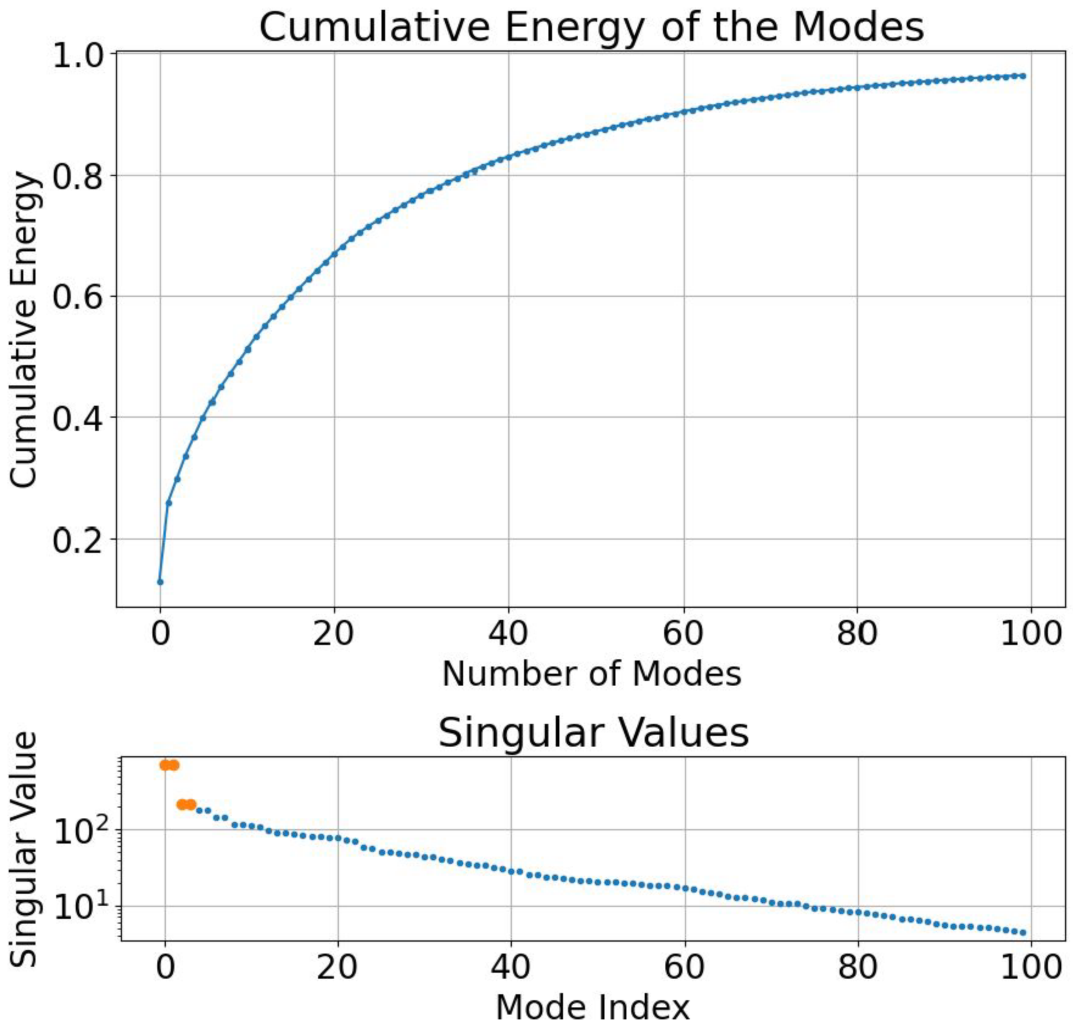
$$+\quad\underbrace{\lambda_3\|\Xi\|_1}_{\text{SINDy regularization}}$$

Latent Space Dimension

When doing reduced order modeling, a critical decision is selecting the appropriate dimensionality—that is, determining the number of modes or latent variables to retain.

- If the system dynamics are well understood, it may be possible to rely on **physical intuition** to guide this choice.

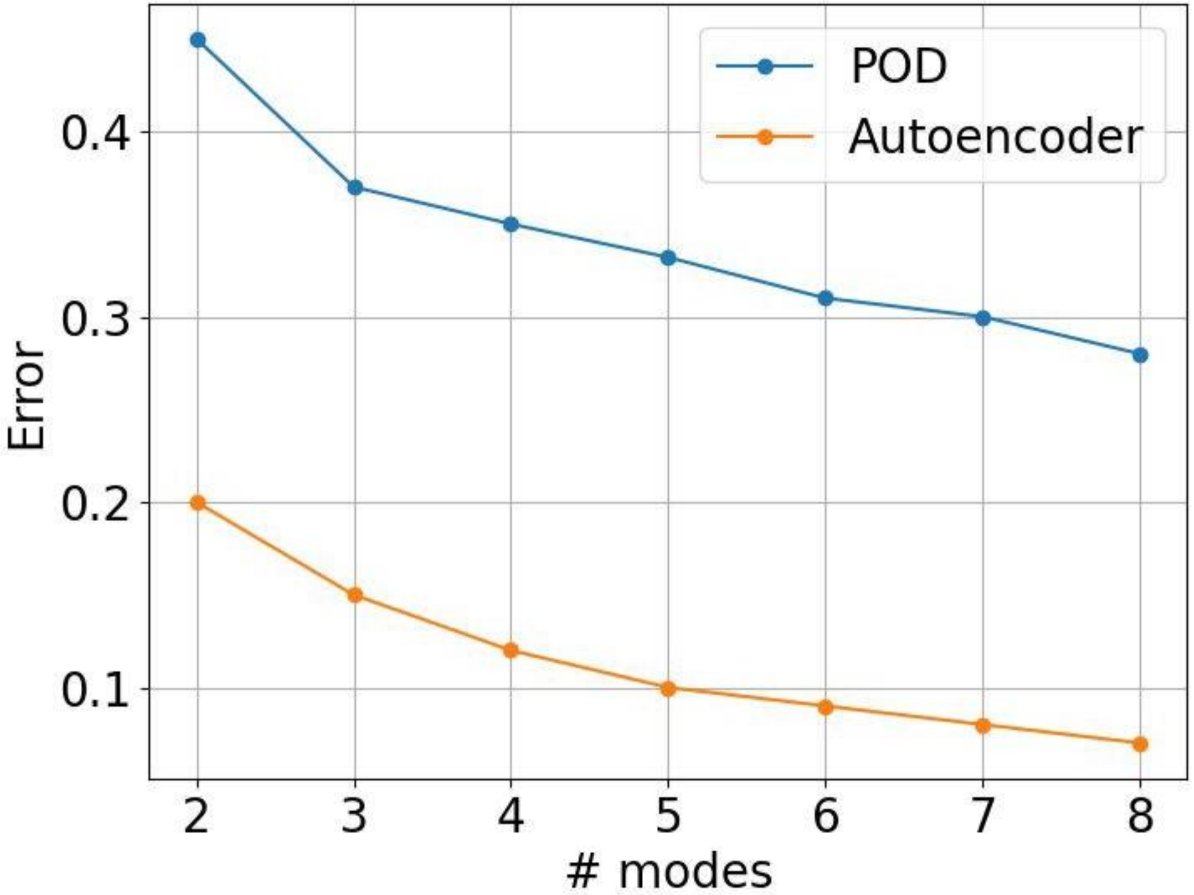
- For POD, a common strategy is to analyze the **cumulative energy** captured by the modes. The dimensionality is chosen to retain a sufficient percentage of the total energy, ensuring that the dominant dynamics are preserved.



Latent Space Dimension

For Autoencoders, there is no direct notion of energy. The typical approach involves experimenting with **different latent space dimensions** and selecting the smallest dimension that yields a **satisfactory reconstruction error**, balancing compression with accuracy.

In the context of SINDy-AE, selecting a low-dimensional latent space is particularly important to mitigate the effects of the **curse of dimensionality.**



Recap

- **symbolic regression** is used to discover governing equations of dynamical systems
- **SINDy** is the most widely used approach, with many extensions
- performance critically depends on choice of basis function library and (relatedly) the ability to reduce the dimension
- Can reduce the dimension via ROM methods such as POD; another popular method is **dynamic mode decomposition** (DMD)

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Cassio Oishi, Fabio Amaral, and Davide Murari.