Deep Learning for Universal Linear Embeddings of Nonlinear Dynamics

IPAM 2019

Operator Theoretic Methods in Dynamic Data Analysis and Control

J. Nathan Kutz & Kathleen Champion & Steven Brunton

Department of Applied Mathematics
University of Washington
Email: kutz@uw.edu
Web: faculty.washington.edu/kutz

Bernard Koopman 1931

Definition: Koopman Operator (Koopman 1931): *For a dynamical system*

$$\frac{d\mathbf{x}}{dt} = \mathbf{N}(\mathbf{x}),$$

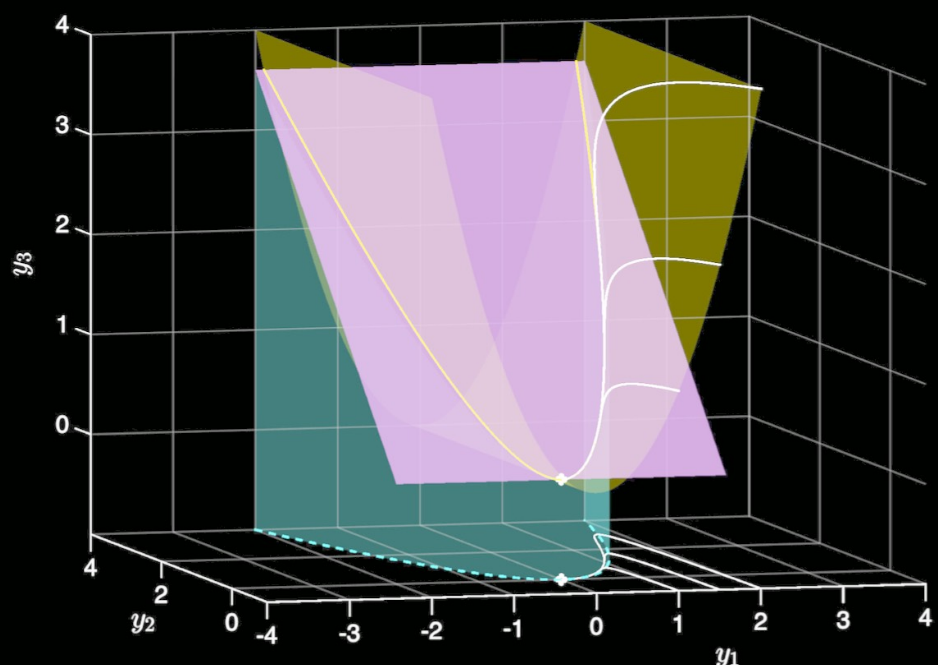
where $\mathbf{x} \in \mathbb{R}^n$ *is in a state space* $\mathbf{x} \in \mathscr{M}$. *The Koopman operator* $\mathscr{K}$ *acts on a set of scalar observable variables* g_j *which comprise the vector* $\mathbf{g} : \mathscr{M} \to \mathbb{C}$ *so that*

$$\mathscr{K}\,\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{N}(\mathbf{x}))\,.$$

Mezic (2004)
Coifman, Kevrekidis, co-workers - Diffusion Maps
Williams et al - EDMD

Koopman Invariant Subspaces

$$\left.\begin{aligned}\dot{x}_1 &= \mu x_1 \\ \dot{x}_2 &= \lambda(x_2 - x_1^2)\end{aligned}\right\} \implies \frac{d}{dt}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \mu & 0 & 0 \\ 0 & \lambda & -\lambda \\ 0 & 0 & 2\mu \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \quad \text{for} \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \end{bmatrix}$$



Brunton, Proctor & Kutz, PLOS ONE (2018)

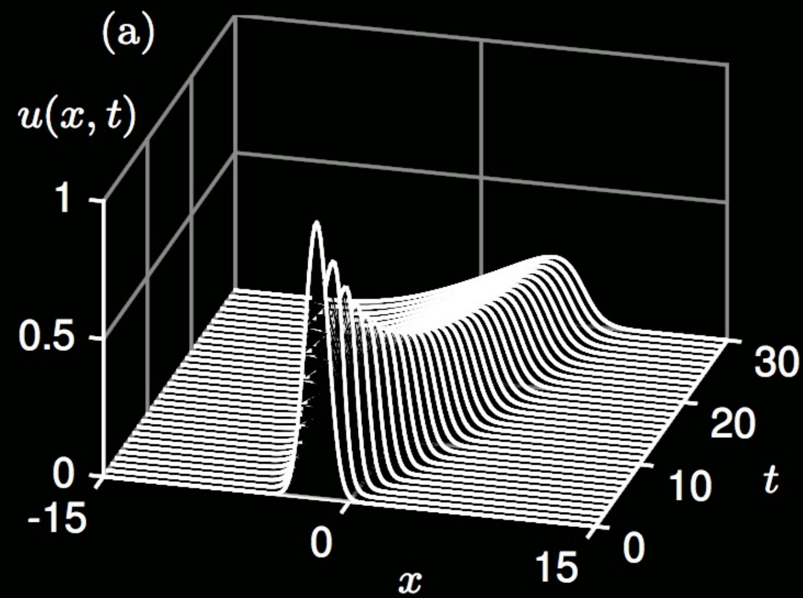
Burgers' Equation

$$u_t + uu_x - \epsilon u_{xx} = 0 \qquad \epsilon > 0,\ x \in [-\infty, \infty]$$

Cole-Hopf

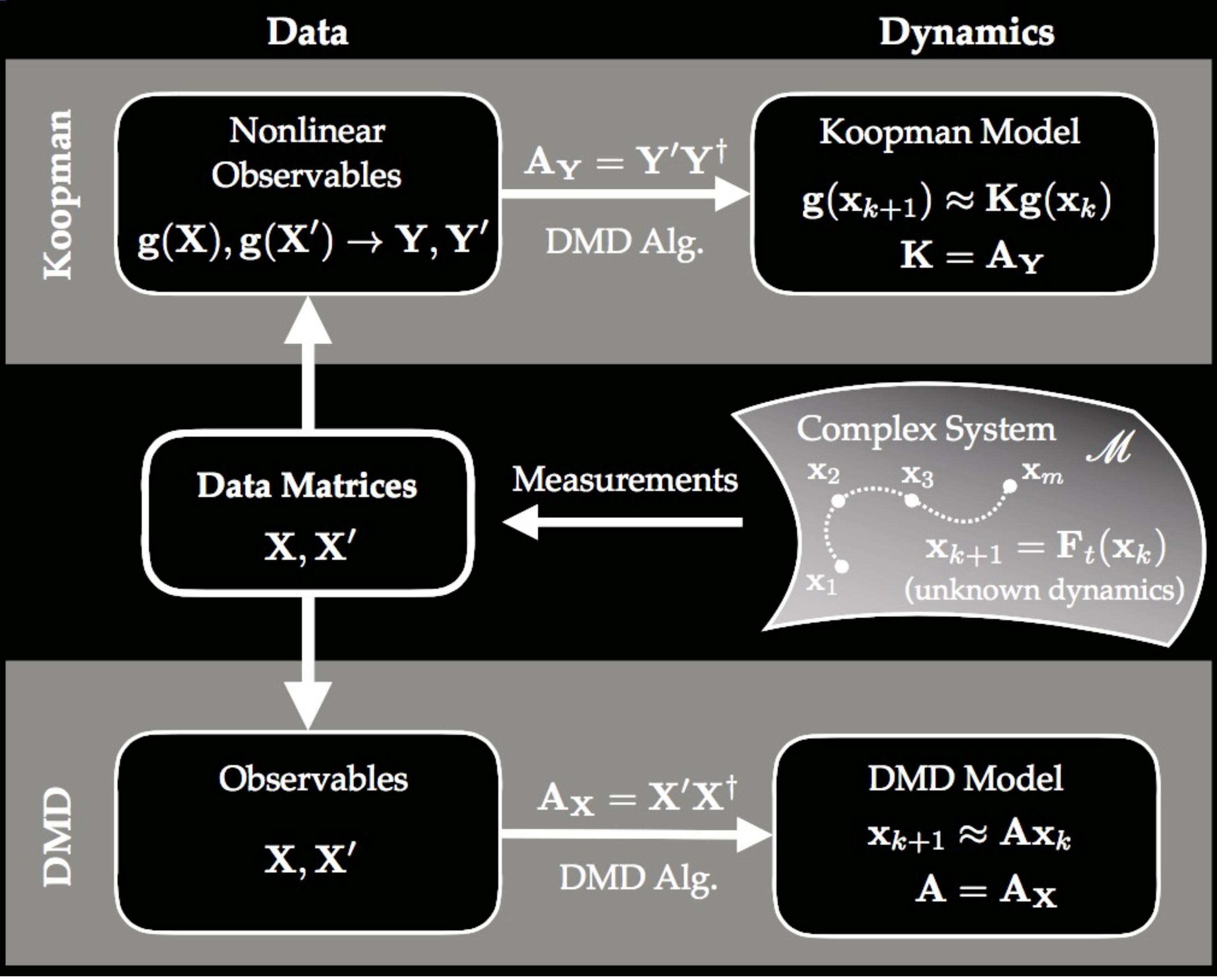
$$u = -2\epsilon v_x / v$$

$$v_t = \epsilon v_{xx}$$



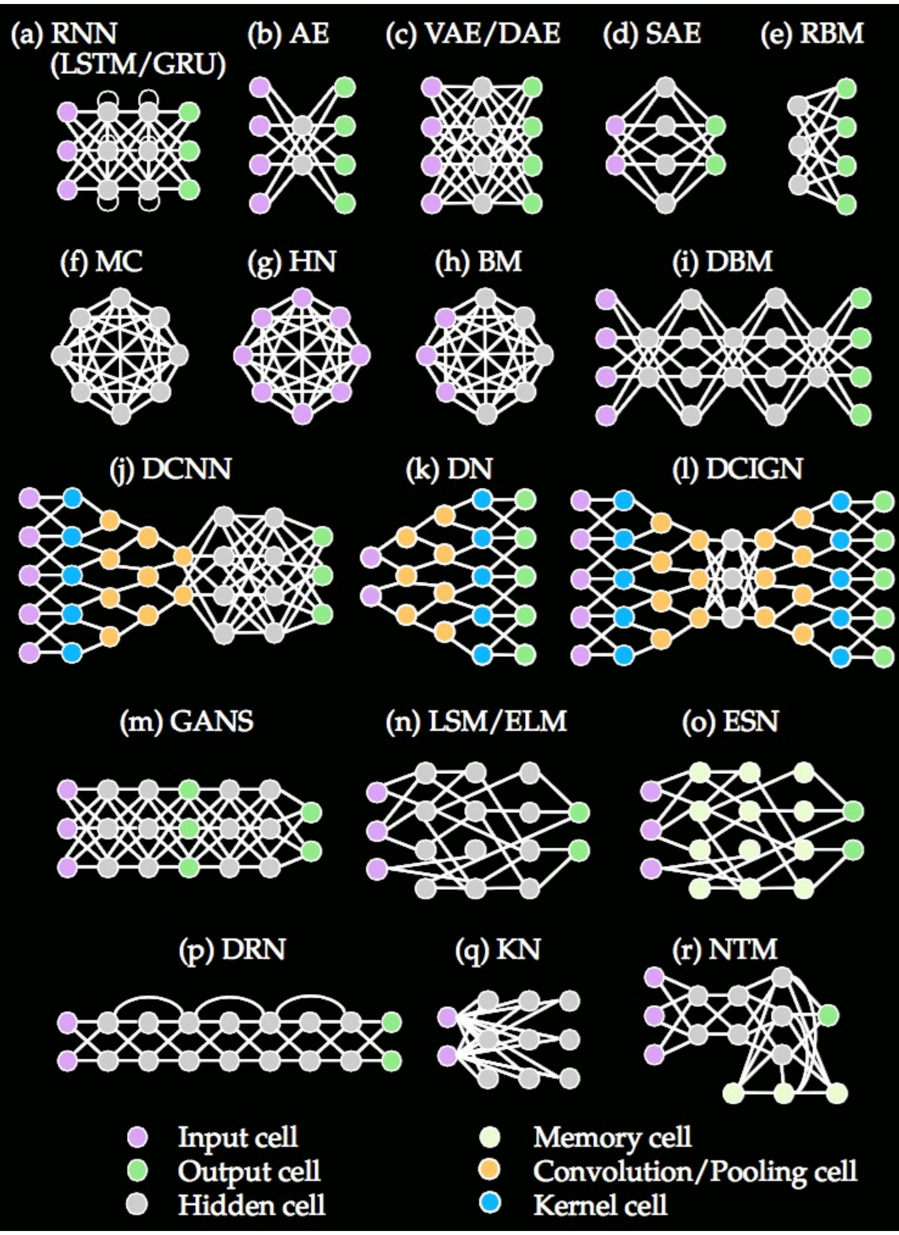
Kutz, Proctor & Brunton, Complexity (2018)

Koopman vs DMD: All about Observables!

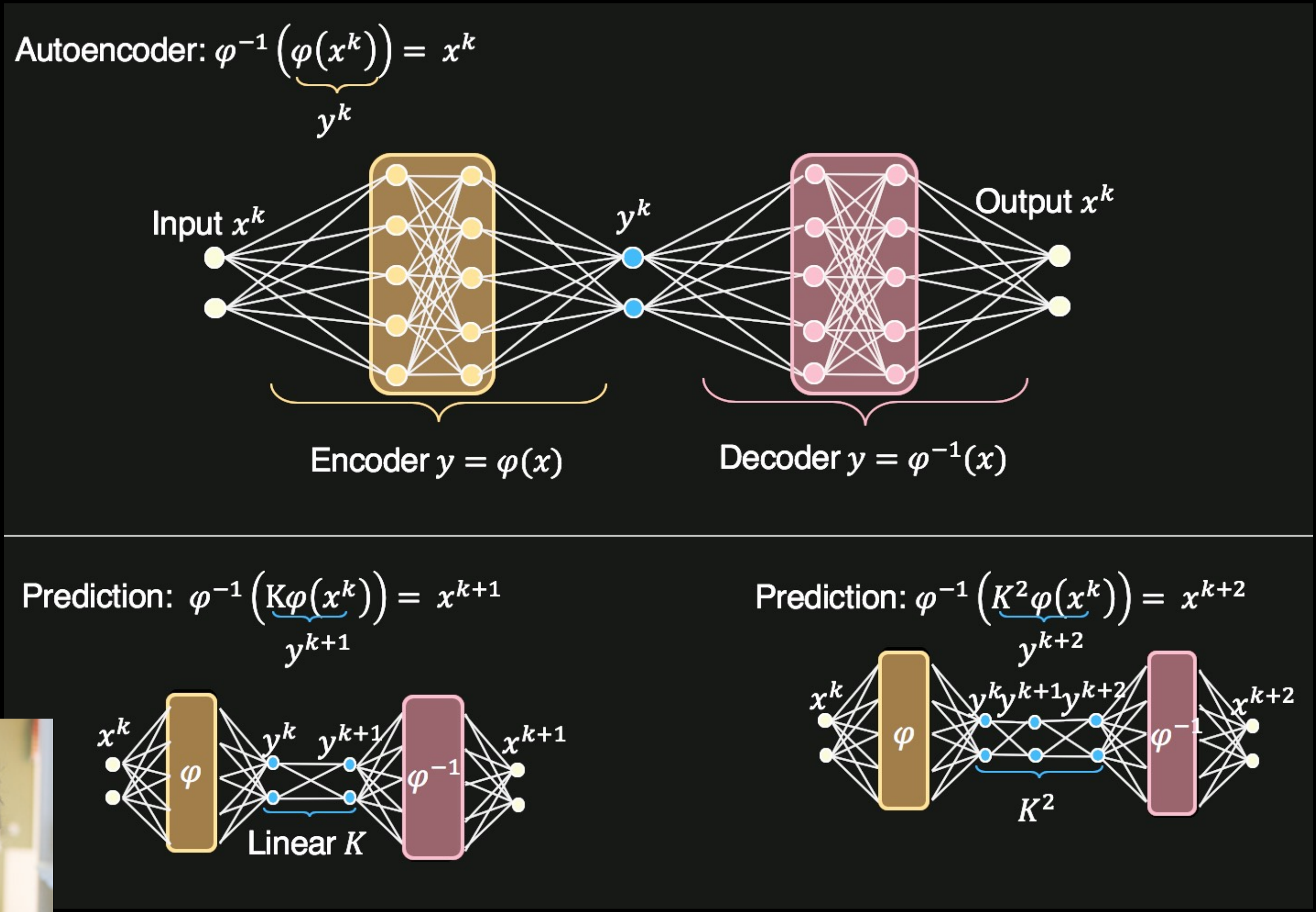


Neural Nets

NN Zoo


(a) RNN (LSTM/GRU)
(b) AE
(c) VAE/DAE
(d) SAE
(e) RBM
(f) MC
(g) HN
(h) BM
(i) DBM
(j) DCNN
(k) DN
(l) DCIGN
(m) GANS
(n) LSM/ELM
(o) ESN
(p) DRN
(q) KN
(r) NTM
Input cell
Output cell
Hidden cell
Memory cell
Convolution/Pooling cell
Kernel cell

NNs for Koopman Embedding



Bethany Lusch

Lusch et al. Nat. Comm (2018)

Failure!

(obviously)

Duffing Oscillator

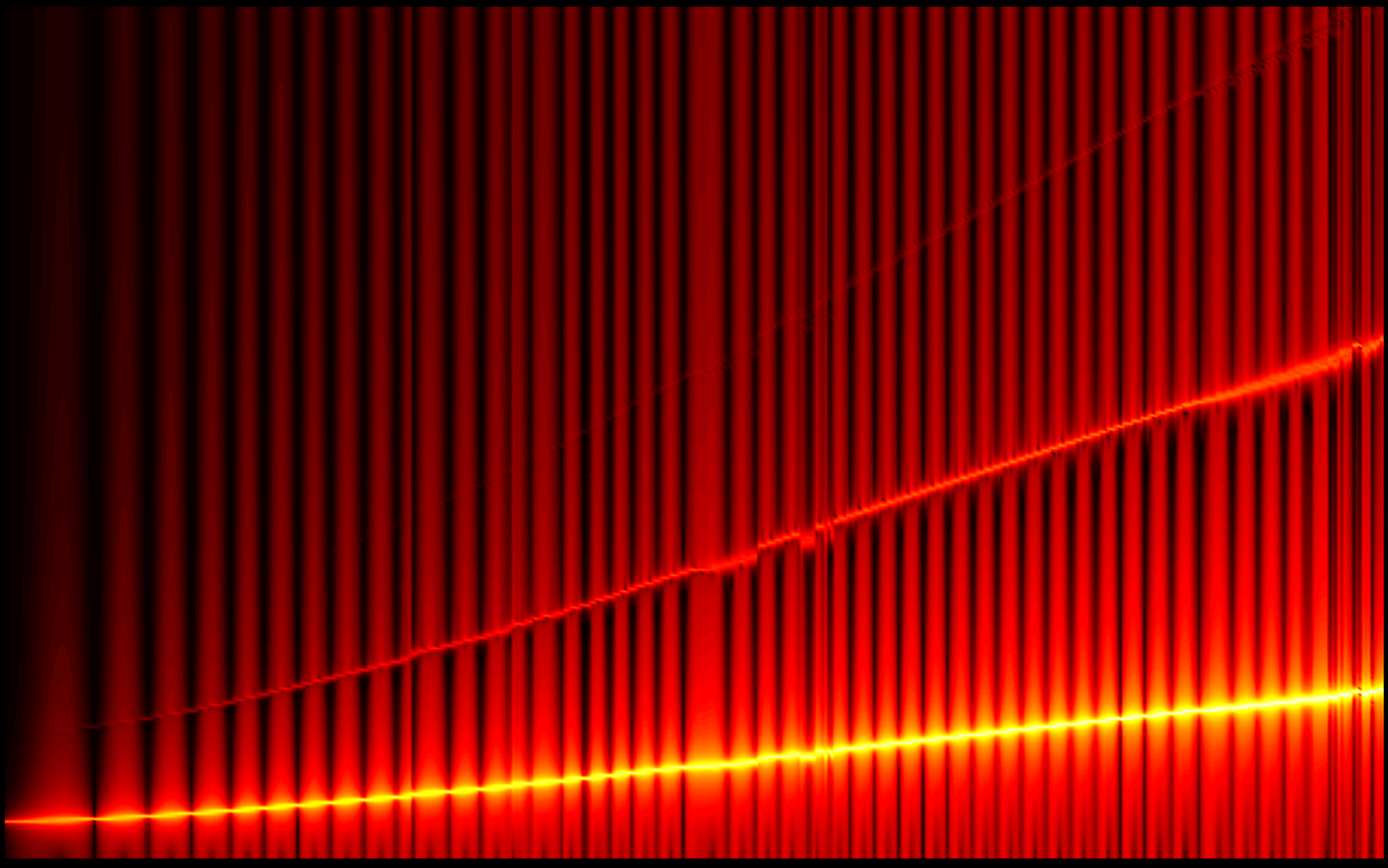
Poincare-Lindstedt Expansion: let $\tau = \omega t$ so that

$$y_{tt} + y + \epsilon y^3 = 0 \Rightarrow \omega^2 y_{\tau\tau} + y + \epsilon y^3 = 0$$

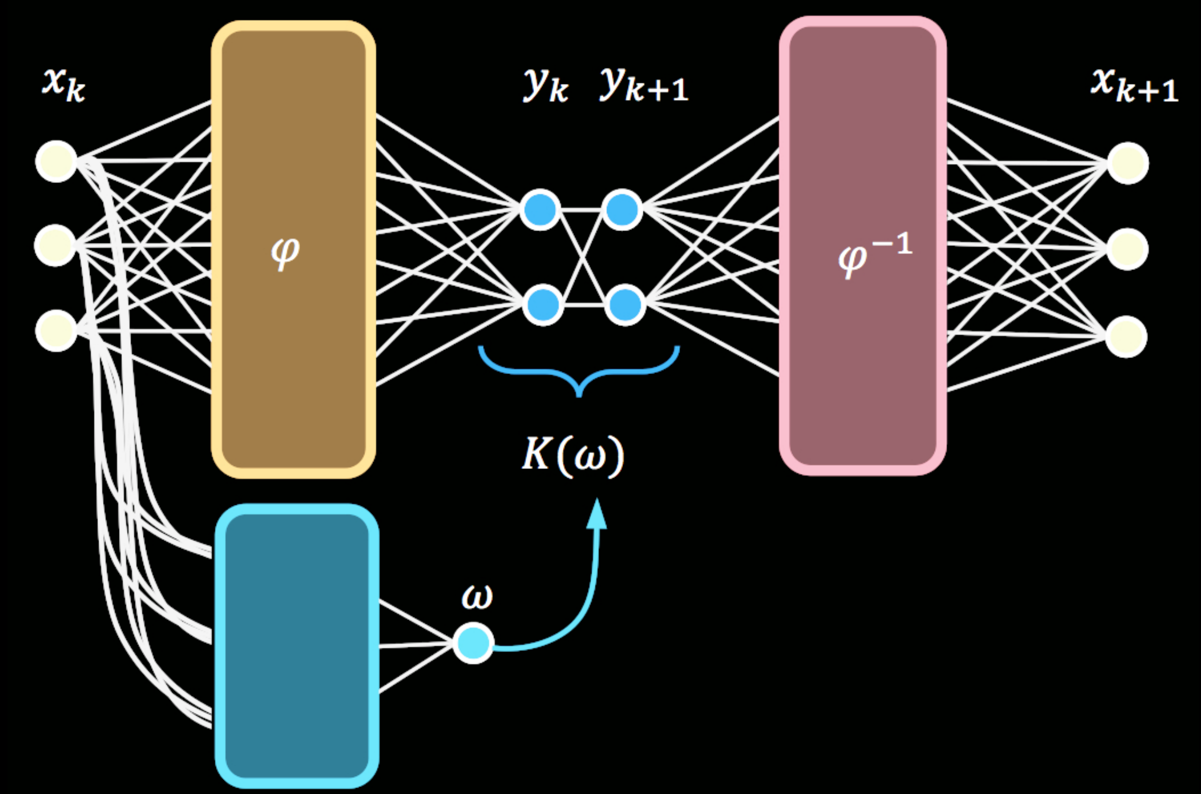
Nonlinearity: Shifts Frequencies + Generates Harmonics

$$y = A \sin[(1+\epsilon 3A^2/8)t] + \epsilon \left\{ \frac{3A^3}{32} \sin\left[\left(1+\epsilon \frac{3A^2}{8}\right)t\right] - \frac{A^3}{32} \sin\left[3\left(1+\epsilon \frac{3A^2}{8}\right)t\right] \right\}$$

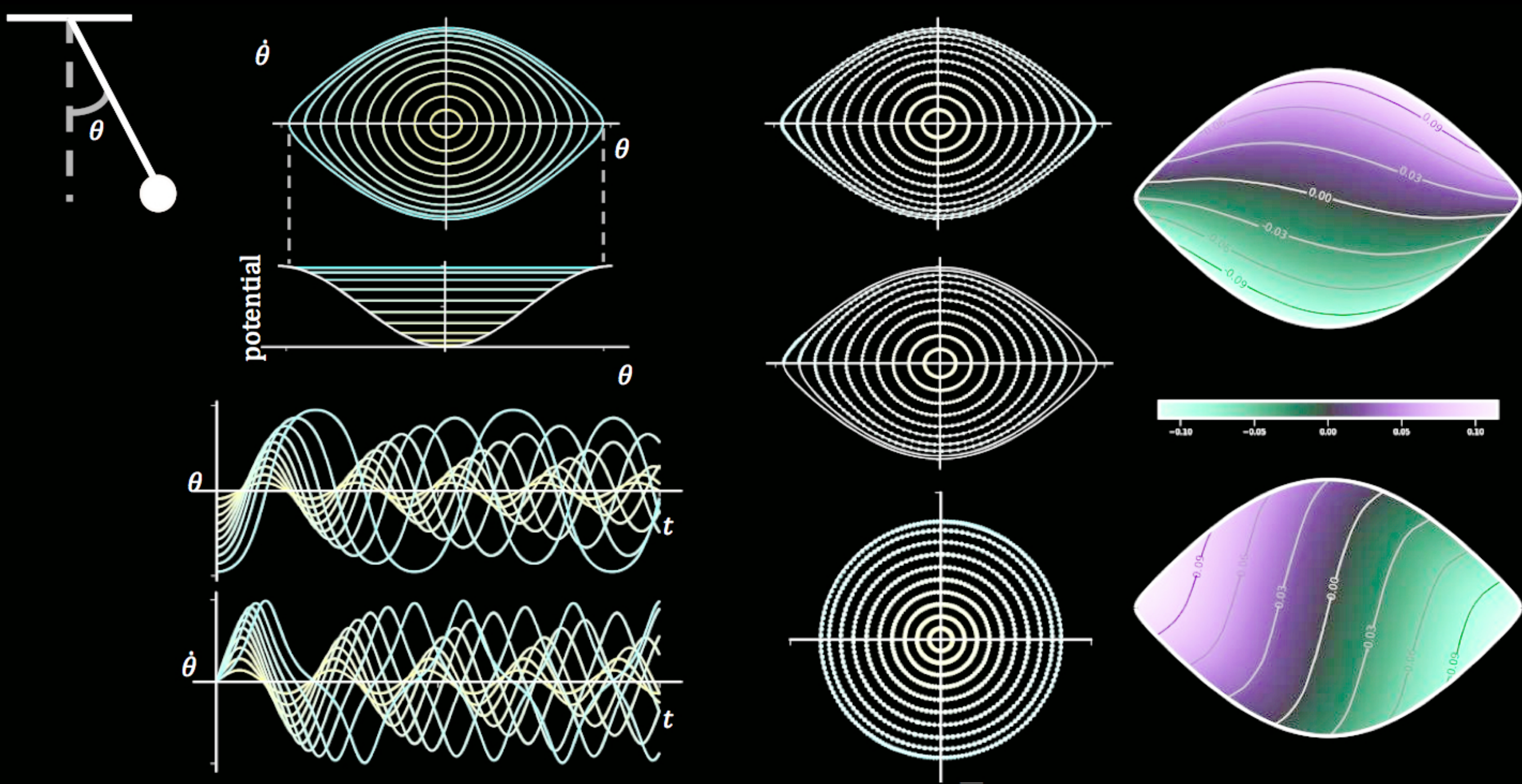
Spectrogram



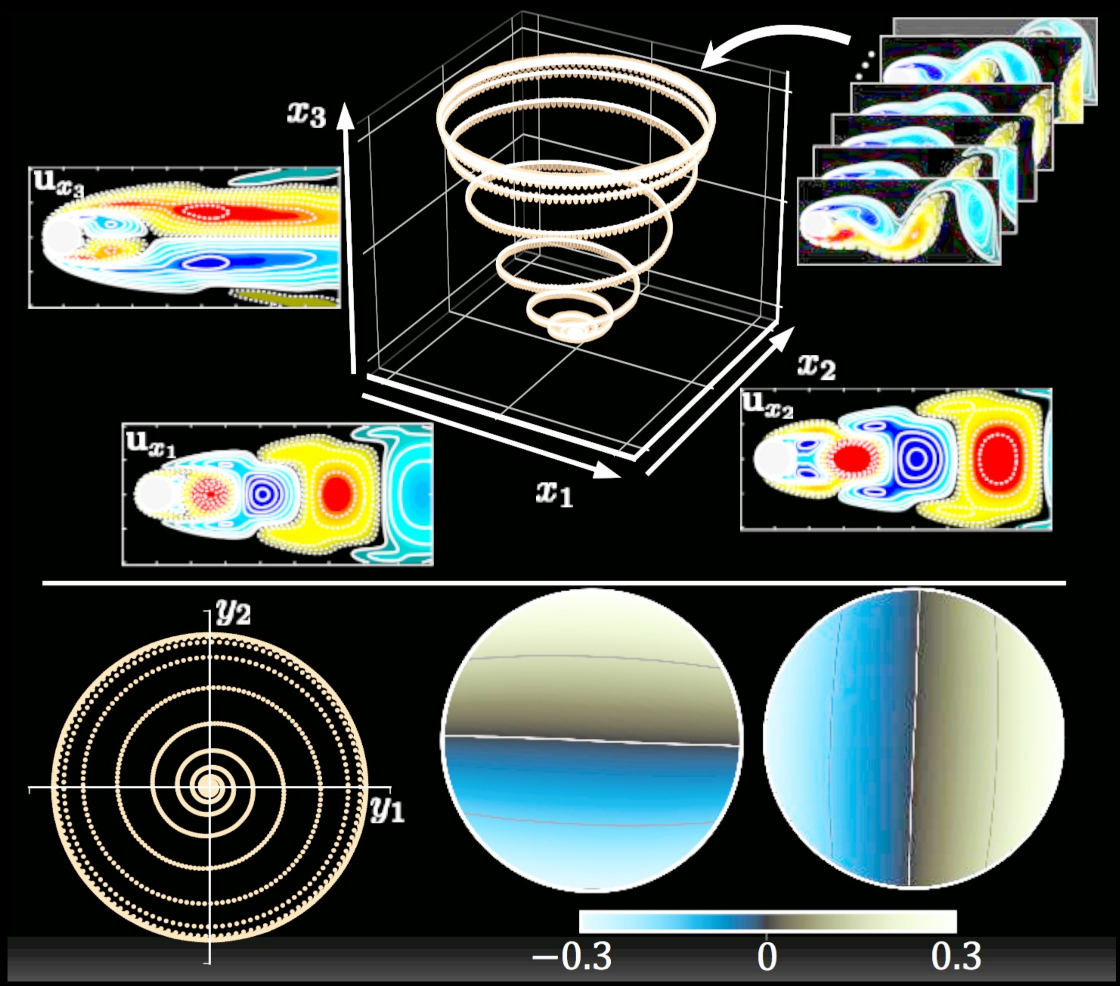
Handling the Continuous Spectra



The Pendulum



Flow Around a Cylinder



Relax Koopman

Sparse Identification of Nonlinear Dynamics (SINDy)

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t))$$

↓

$$\mathbf{x}(t) \in \mathbb{R}^n$$

Sparse Identification of Nonlinear Dynamics (SINDy)

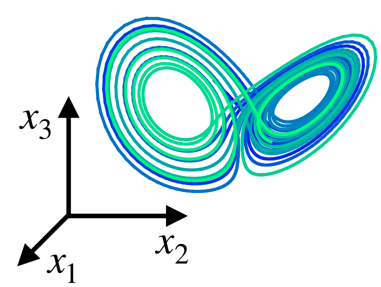
$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t))$$

⬇

$$\mathbf{x}(t) \in \mathbb{R}^n$$

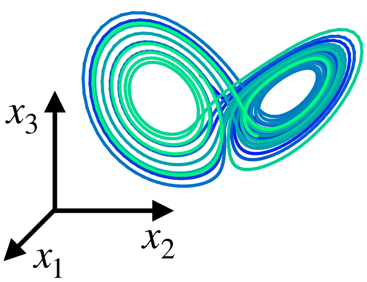
Example: Lorenz

$$\dot{x}_1 = \sigma(x_2 - x_1)$$
$$\dot{x}_2 = x_1(\rho - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - \beta x_3$$



Sparse Identification of Nonlinear Dynamics (SINDy)

True System

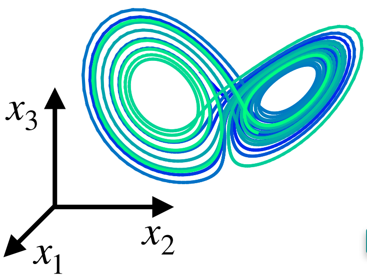


$$\dot{x}_1 = \sigma(x_2 - x_1)$$
$$\dot{x}_2 = x_1(\rho - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - \beta x_3$$

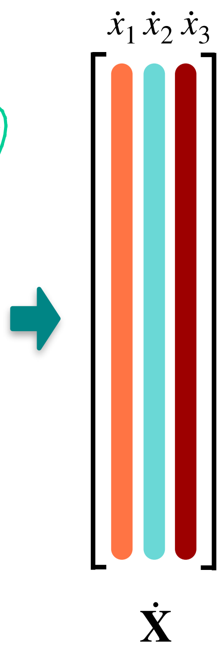
Sparse Identification of Nonlinear Dynamics (SINDy)

True System

SINDy fitting



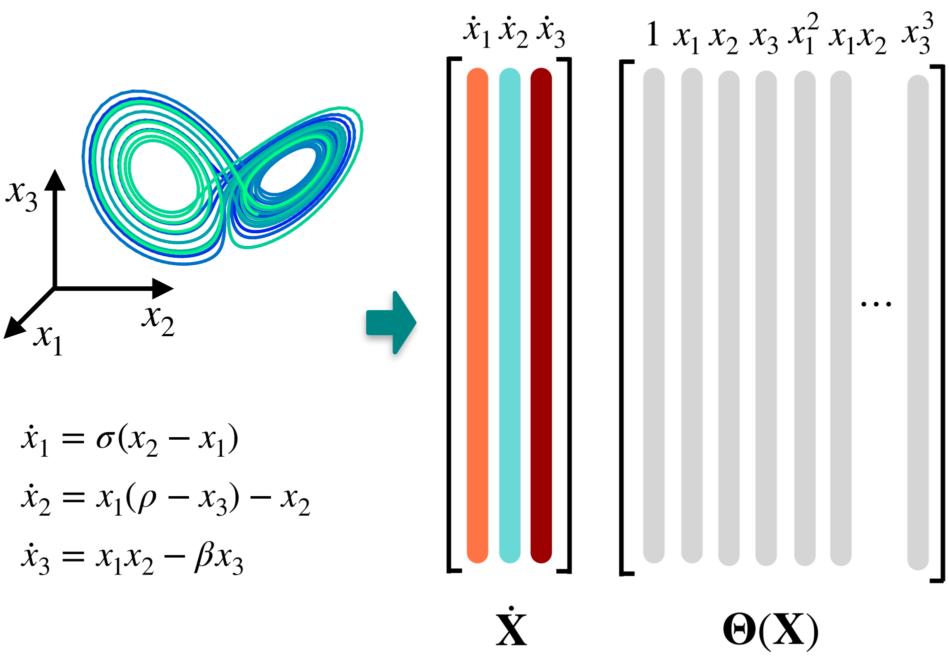
$$\dot{x}_1 = \sigma(x_2 - x_1)$$
$$\dot{x}_2 = x_1(\rho - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - \beta x_3$$



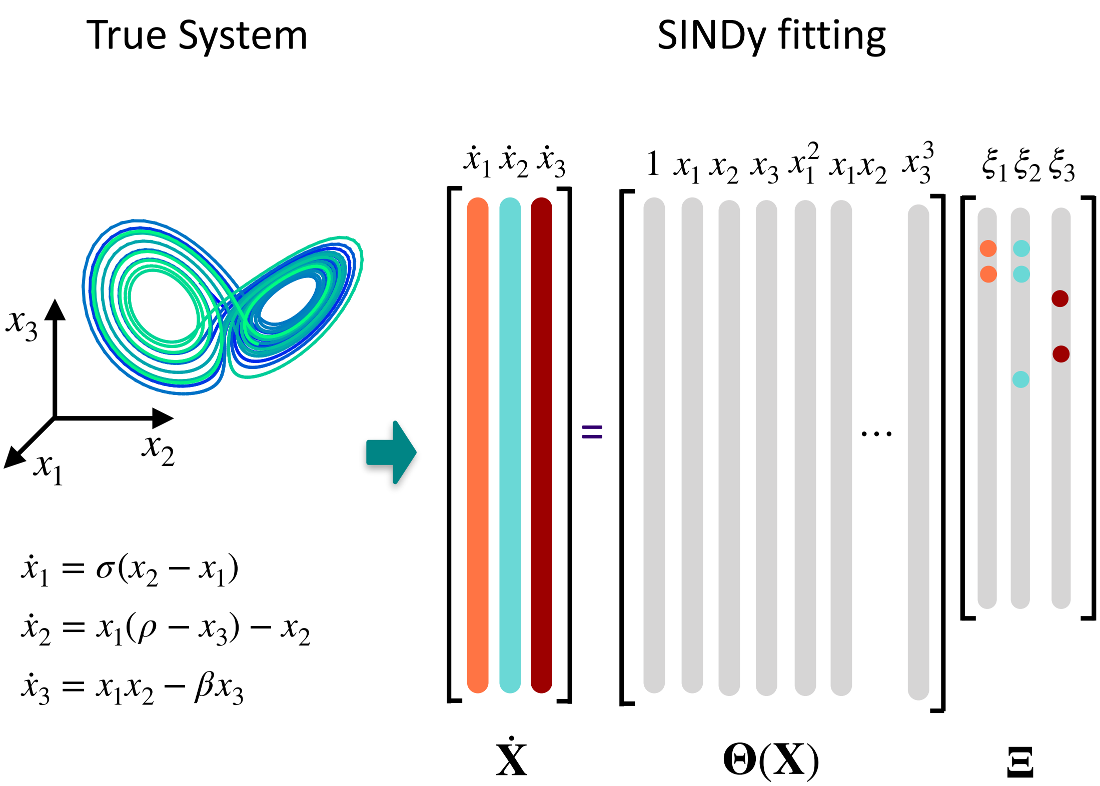
Sparse Identification of Nonlinear Dynamics (SINDy)

True System

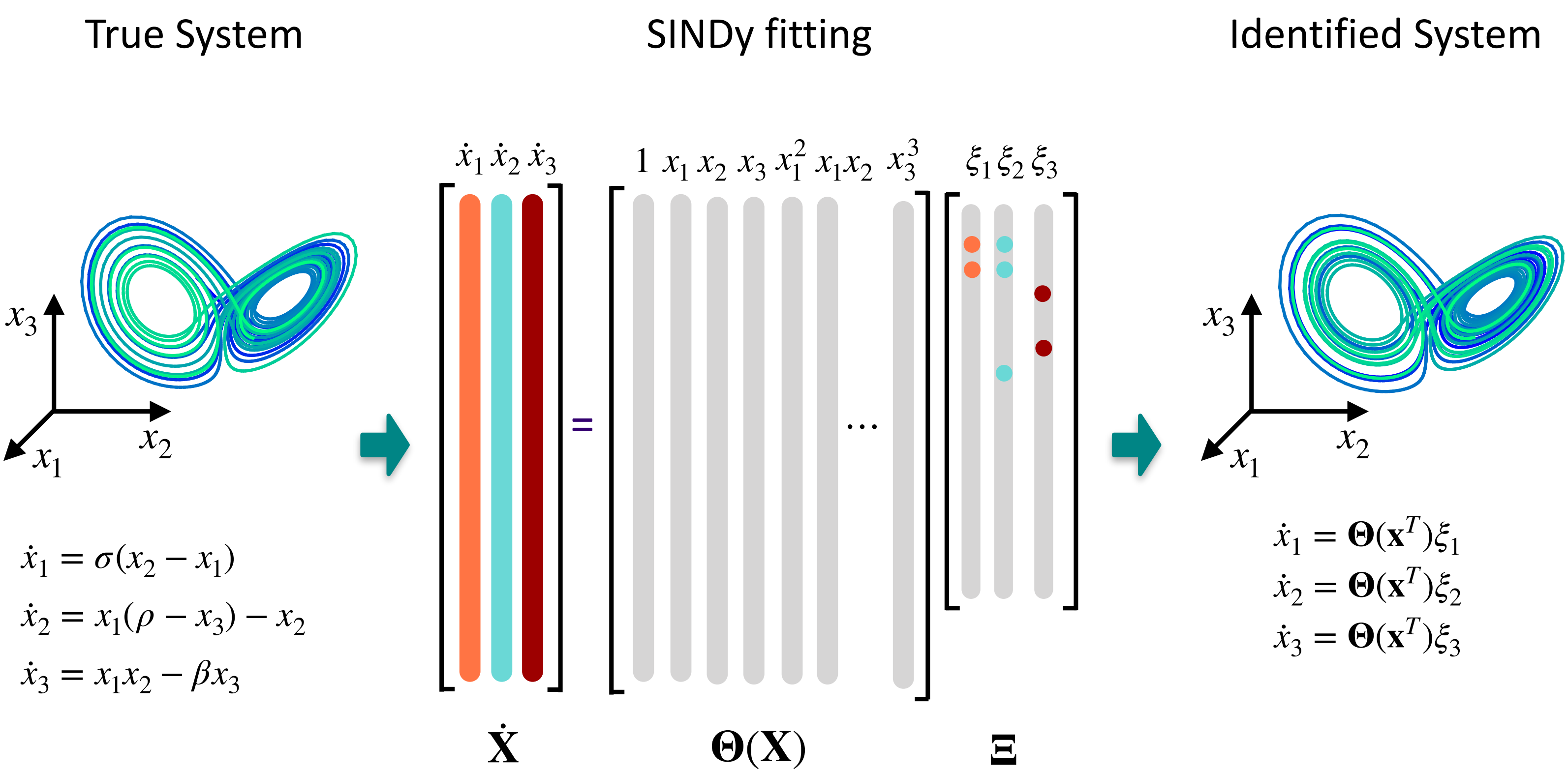
SINDy fitting



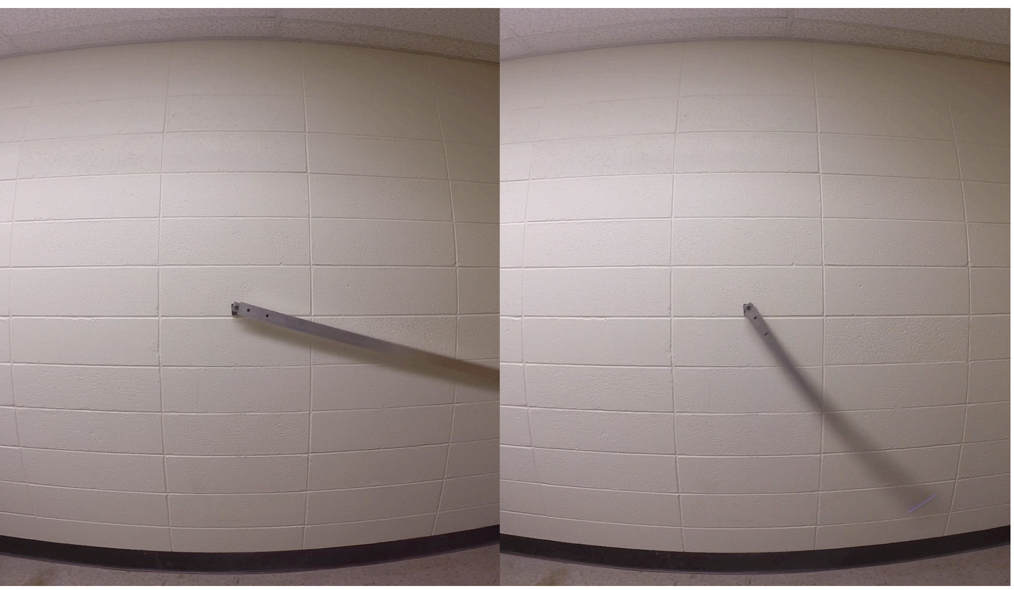
Sparse Identification of Nonlinear Dynamics (SINDy)

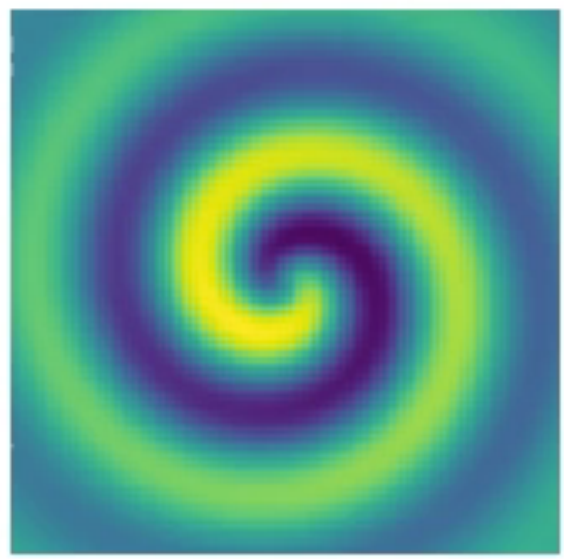


Sparse Identification of Nonlinear Dynamics (SINDy)

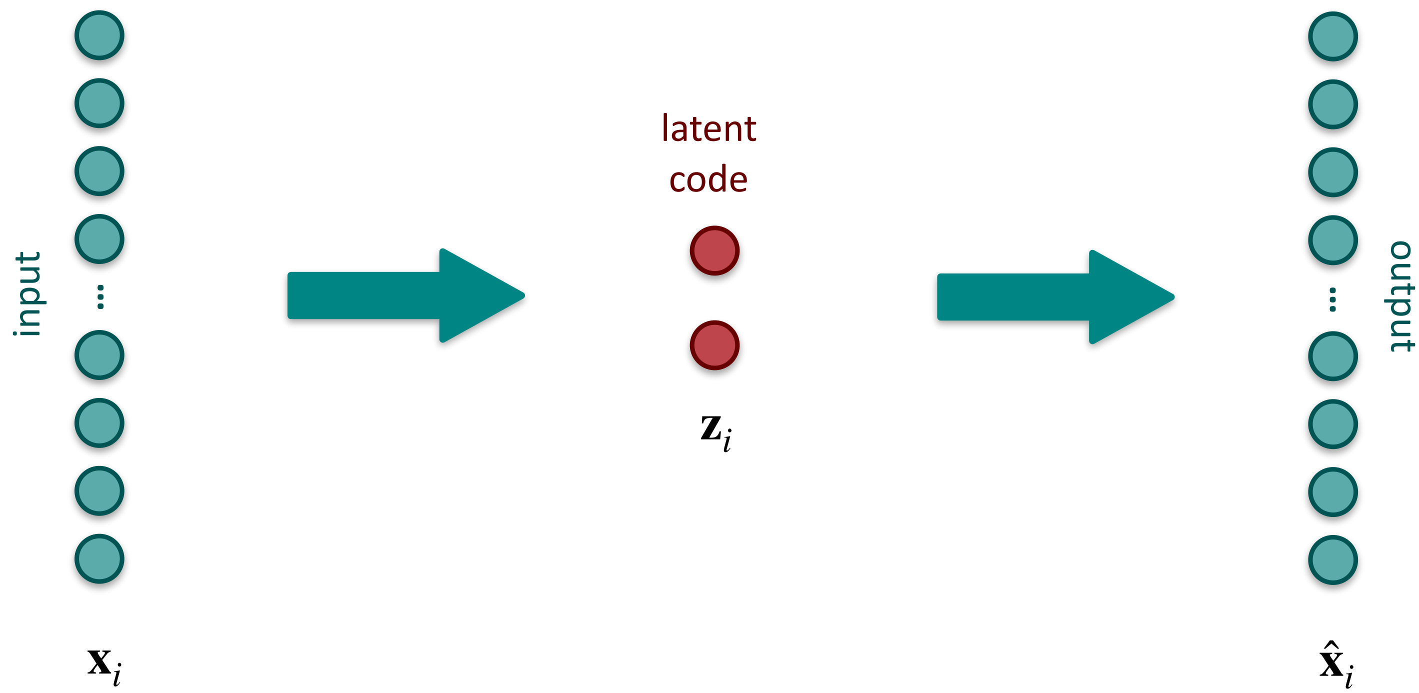


What if we don’t know the right coordinates?



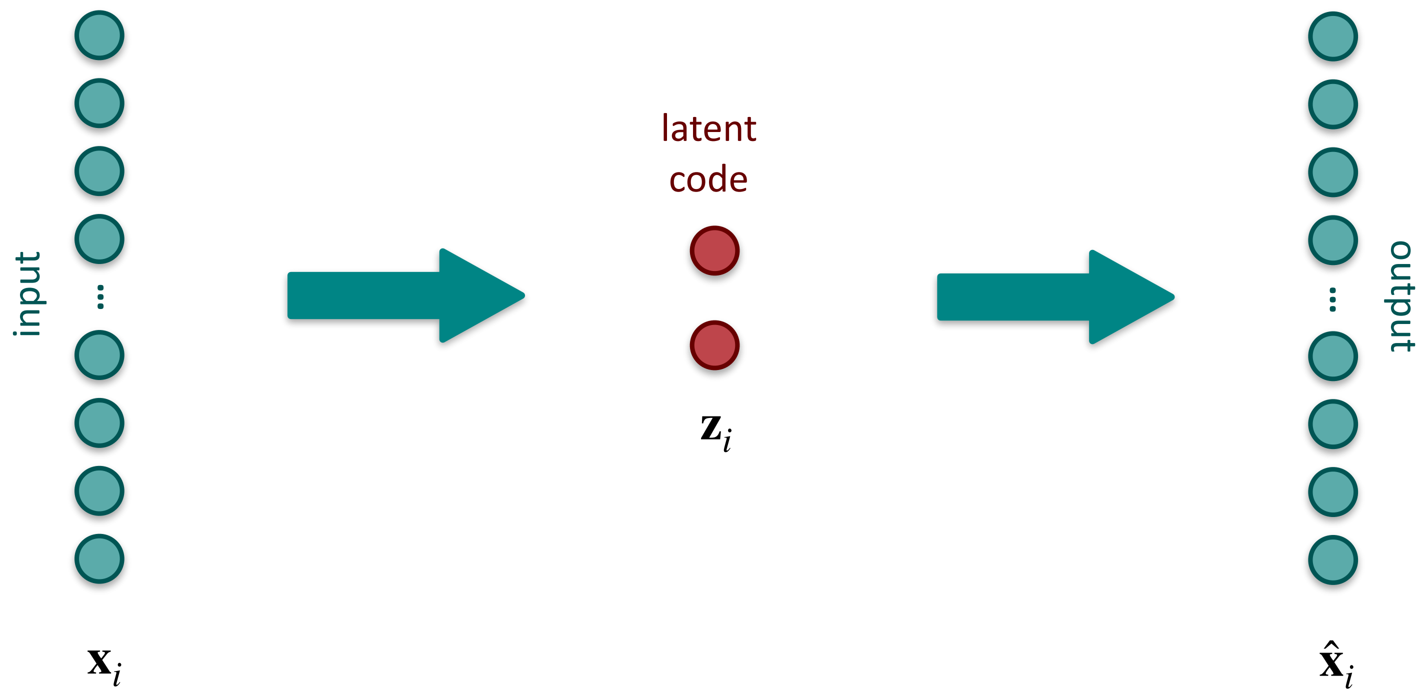


Autoencoder



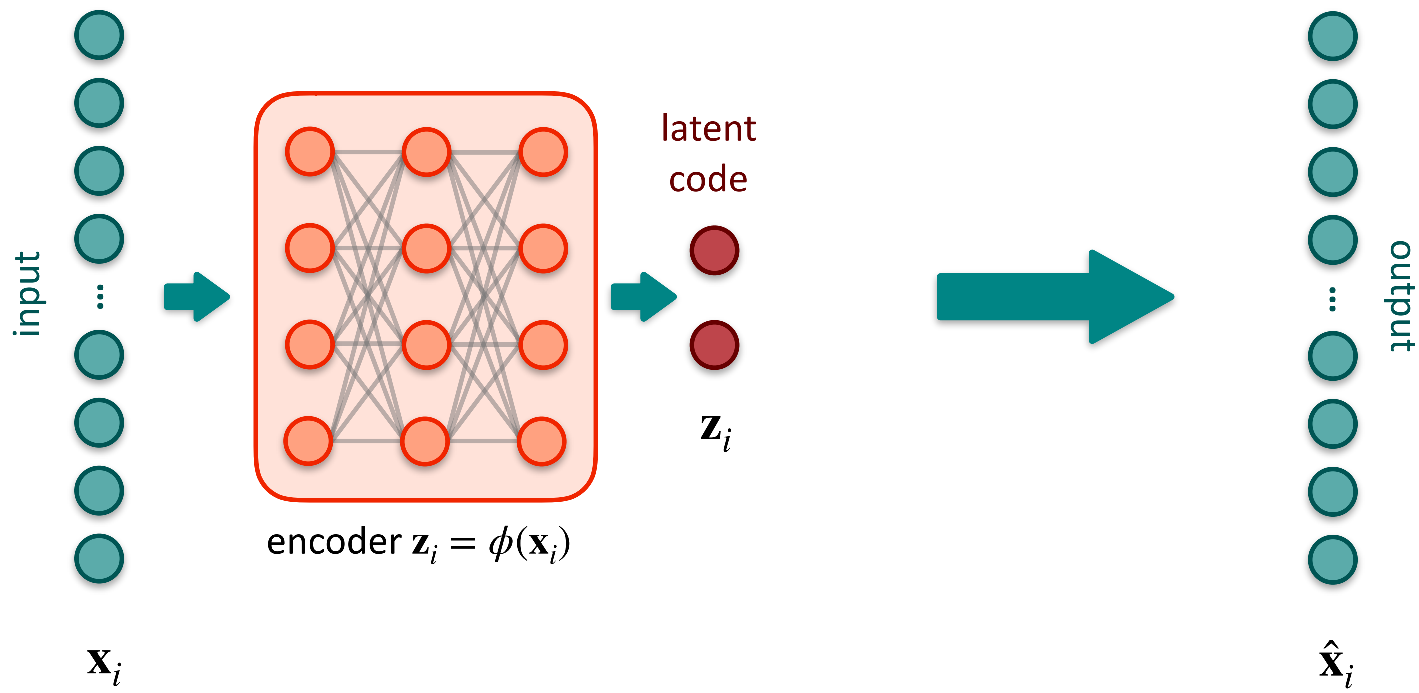
loss function: $$\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

Autoencoder



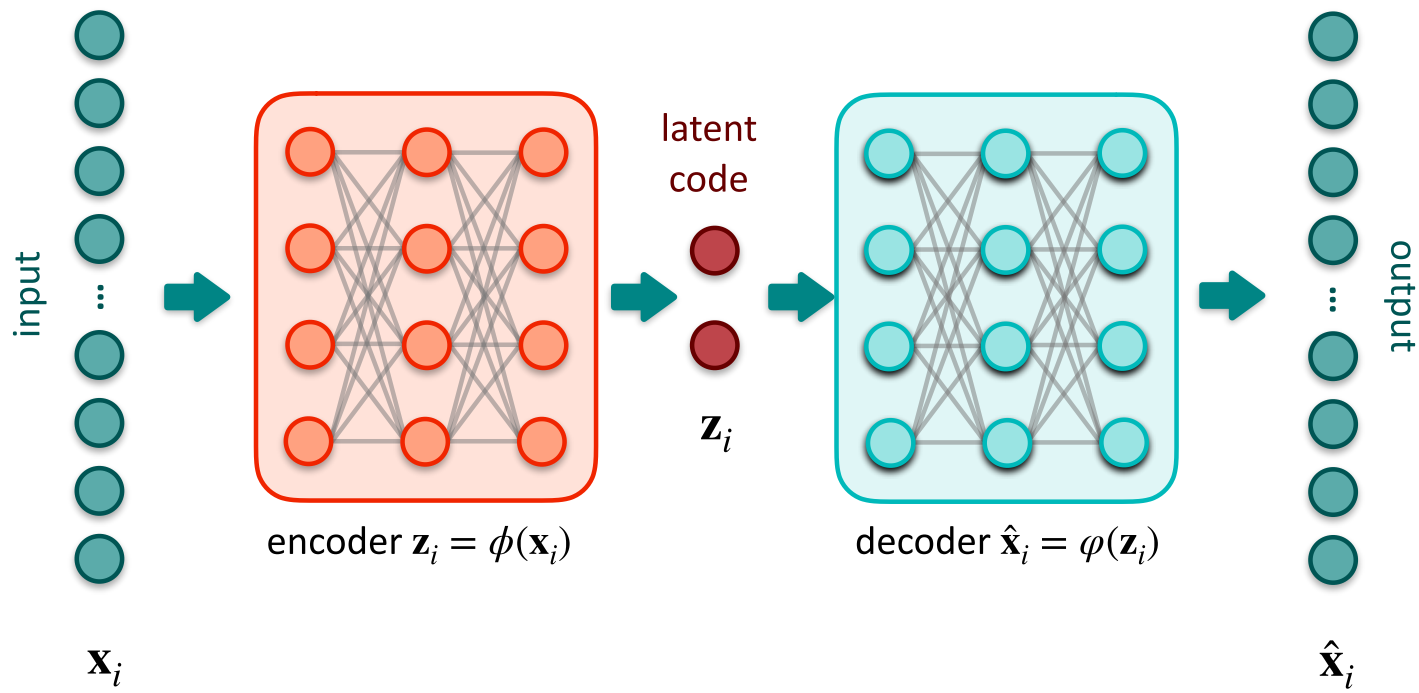
loss function: $$\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

Autoencoder



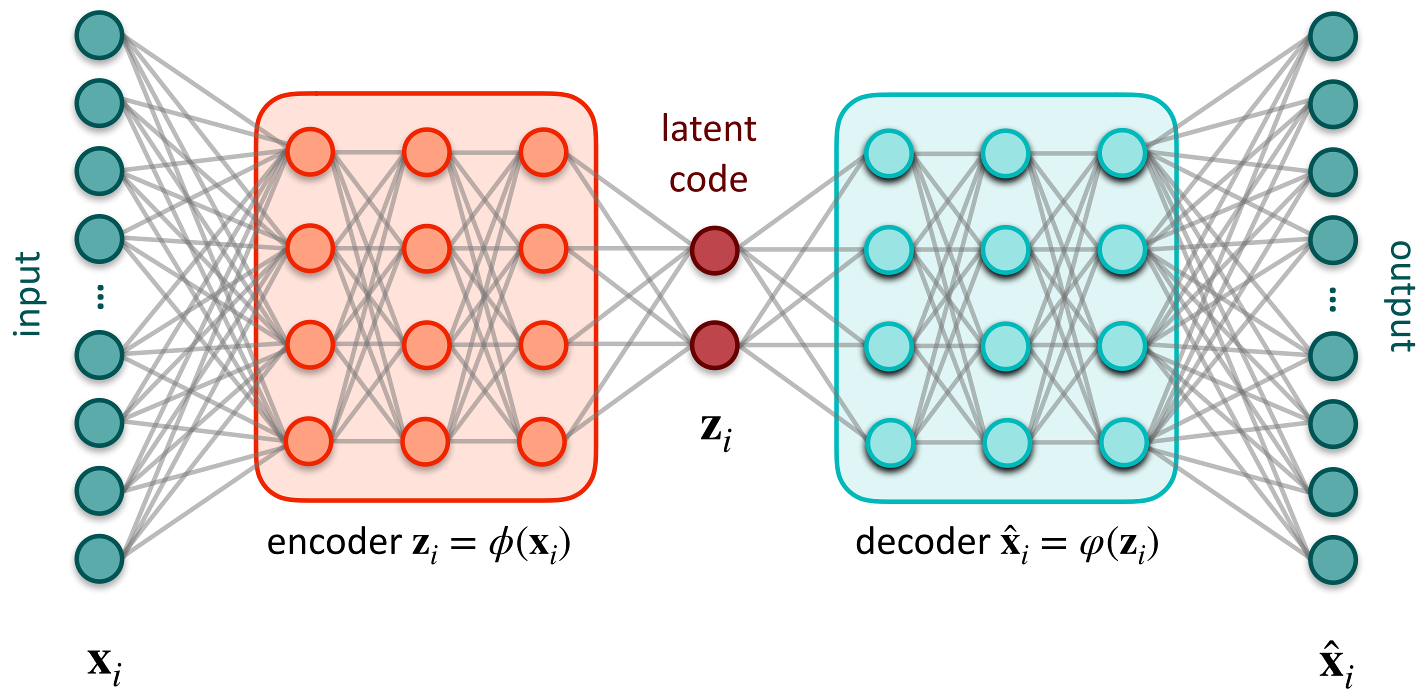
loss function: $\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$

Autoencoder



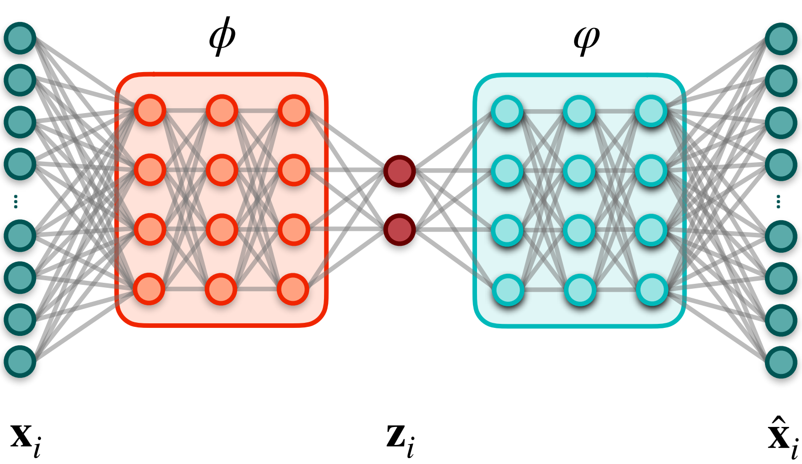
loss function: $$\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

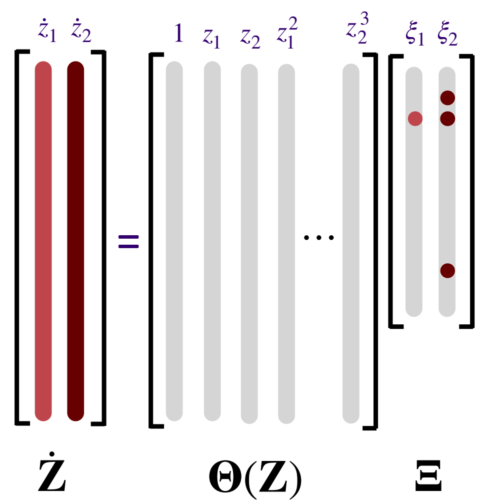
Autoencoder



loss function: $$\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

Autoencoder + SINDy

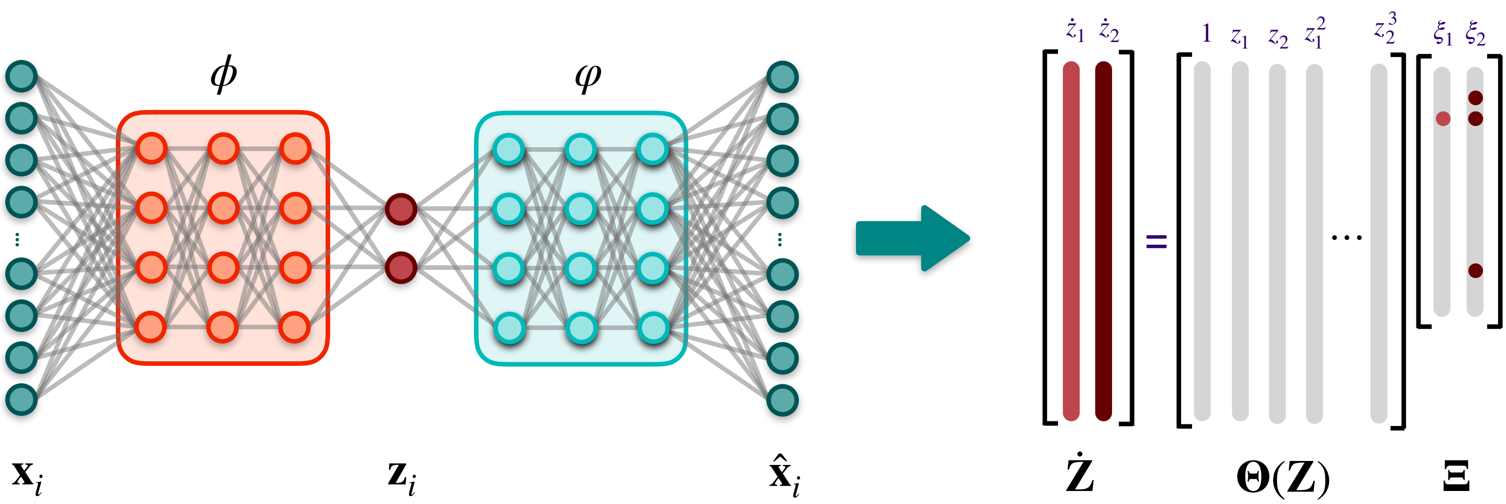




loss: $$\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

loss: $$\frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2$$

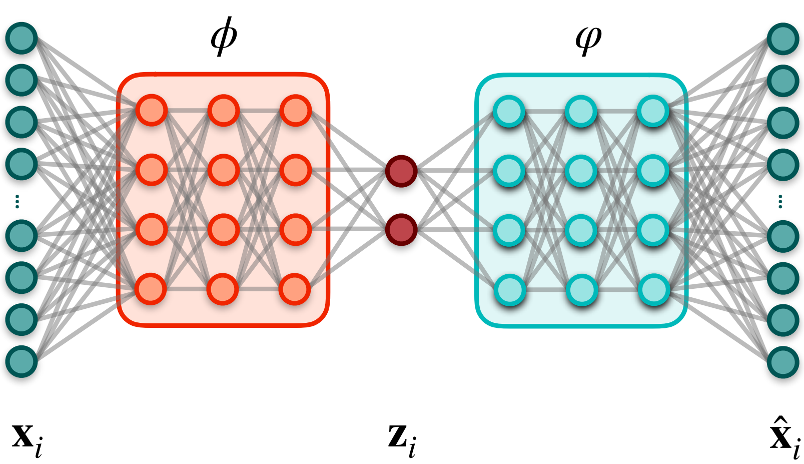
Autoencoder + SINDy

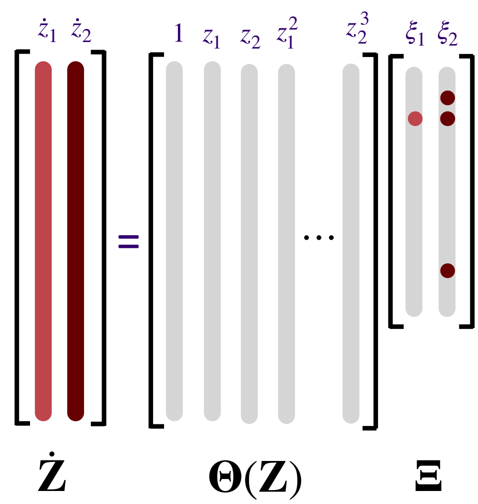


loss: $$\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

loss: $$\frac{1}{N}\sum_{i=1}^{N}\|\dot{\mathbf{z}}_i - \mathbf{\Theta}(\mathbf{z}_i^T)\mathbf{\Xi}\|_2^2$$

Autoencoder + SINDy

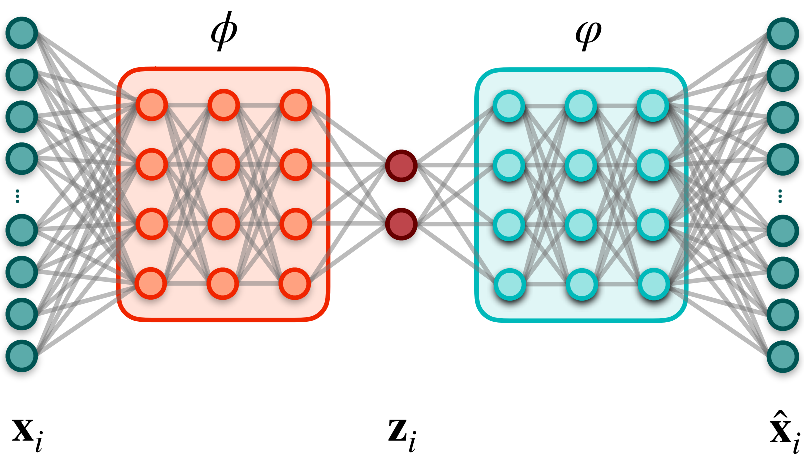


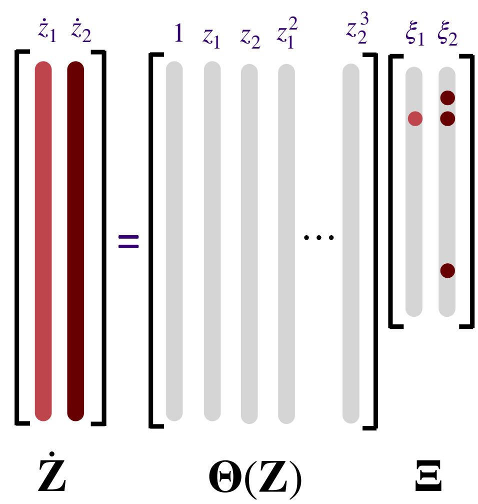


loss: $$\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$$

loss: $$\frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \mathbf{\Theta}(\mathbf{z}_i^T)\mathbf{\Xi}\|_2^2$$

Autoencoder + SINDy



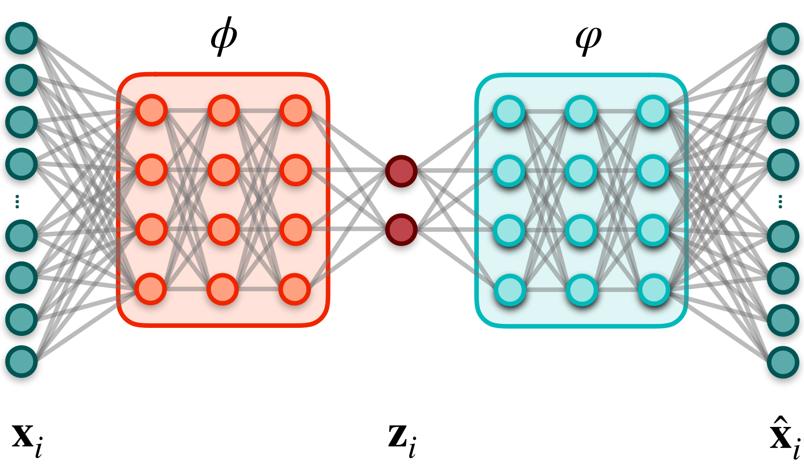


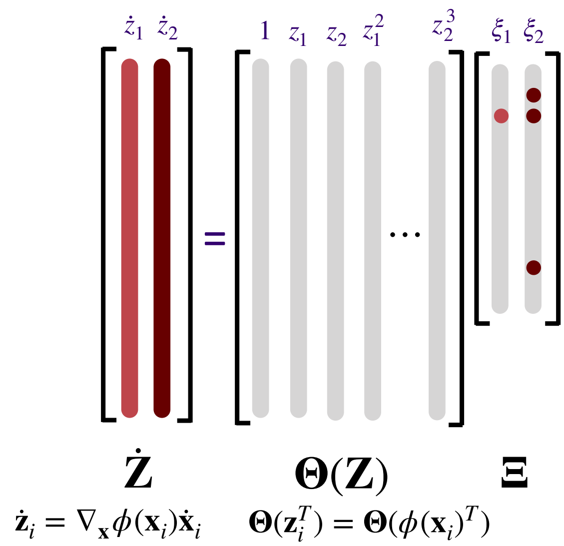
$\dot{\mathbf{z}}_i = \nabla_{\mathbf{x}}\phi(\mathbf{x}_i)\dot{\mathbf{x}}_i$

loss: $\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$

loss: $\frac{1}{N}\sum_{i=1}^{N}\|\dot{\mathbf{z}}_i - \boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2$

Autoencoder + SINDy

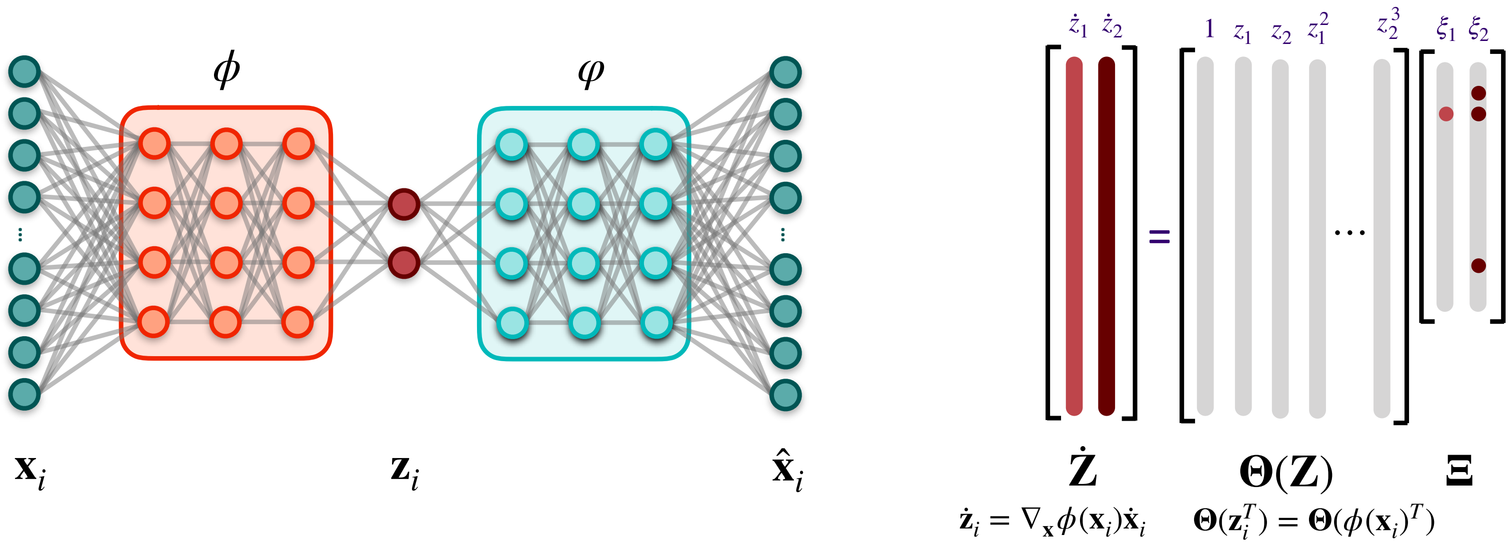




loss: $\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$

loss: $\frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2$

Autoencoder + SINDy

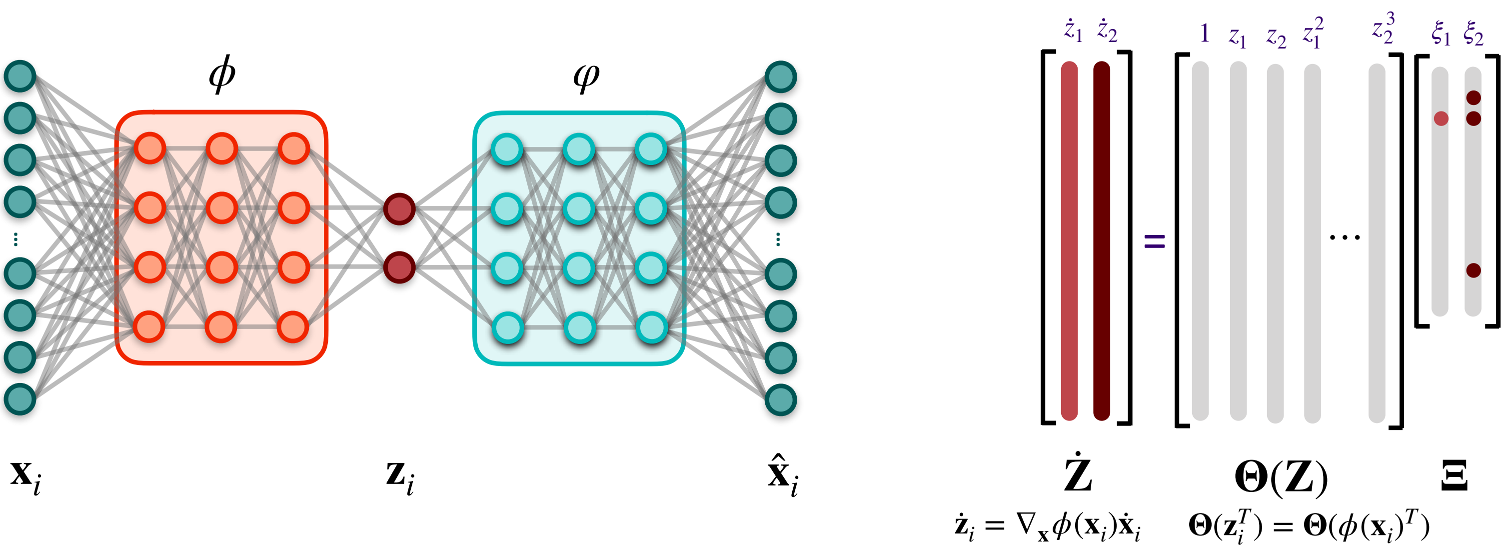


loss: $$\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \varphi(\phi(\mathbf{x}_i))\|_2^2 \; + \; \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\nabla_{\mathbf{x}}\phi(\mathbf{x}_i)\dot{\mathbf{x}}_i - \boldsymbol{\Theta}(\phi(\mathbf{x}_i)^T)\boldsymbol{\Xi}\|_2^2$$

autoencoder component

SINDy component

Autoencoder + SINDy

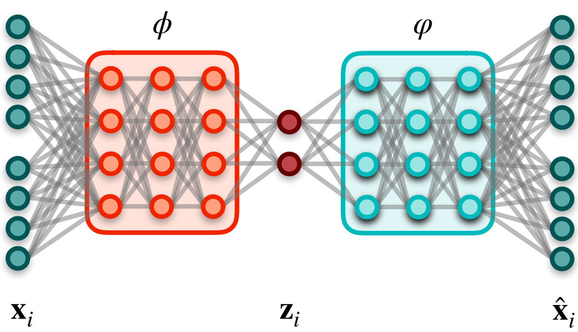


loss: $\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \varphi(\phi(\mathbf{x}_i))\|_2^2 + \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\nabla_{\mathbf{x}}\phi(\mathbf{x}_i)\dot{\mathbf{x}}_i - \boldsymbol{\Theta}(\phi(\mathbf{x}_i)^T)\boldsymbol{\Xi}\|_2^2 + \lambda_3\|\boldsymbol{\Xi}\|_1$

autoencoder component

SINDy component

Autoencoder + SINDy



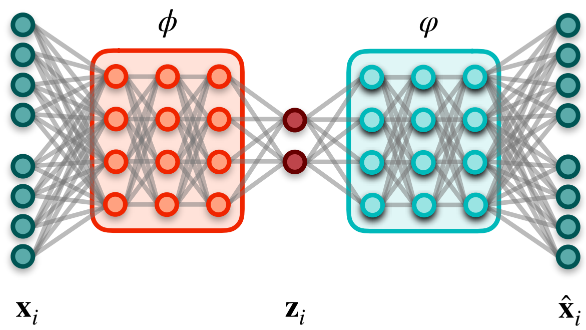
loss:

$$\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2 + \lambda_3 \|\boldsymbol{\Xi}\|_1$$

L_1 L_2 L_3

> ***Issue:*** training shrinks norm of **z** to minimize loss function

Autoencoder + SINDy



loss:

$$\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \mathbf{\Theta}(\mathbf{z}_i^T)\mathbf{\Xi}\|_2^2 + \lambda_3 \|\Xi\|_1$$

L_1 L_2 L_3

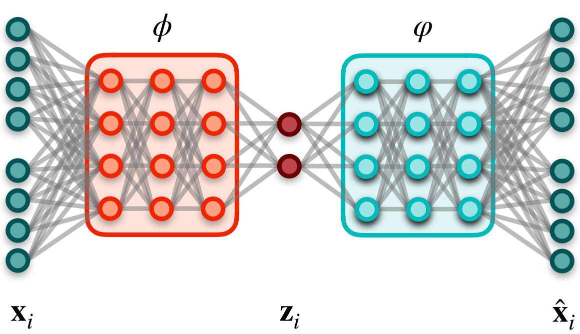
> ***Issue:*** training shrinks norm of **z** to minimize loss function

> ***Solution:*** use the following to enforce SINDy loss

new L_2 :

$$\frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}}\varphi(\mathbf{z}_i)\underbrace{\mathbf{\Theta}(\mathbf{z}_i^T)\mathbf{\Xi}}_{\dot{\mathbf{z}}_i}\|_2^2 = \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}}\varphi(\phi(\mathbf{x}_i))\underbrace{\mathbf{\Theta}(\phi(\mathbf{x}_i)^T)\mathbf{\Xi}}_{\dot{\mathbf{z}}_i}\|_2^2$$

Autoencoder + SINDy



loss:

$$\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{z}}_i - \boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2 + \lambda_3 \|\Xi\|_1$$

L_1 L_2 L_3

> ***Issue:*** training shrinks norm of **z** to minimize loss function

> ***Solution:*** use the following to enforce SINDy loss

new L_2 : $$\frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}}\varphi(\mathbf{z}_i)\underbrace{\boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}}_{\dot{\mathbf{z}}_i}\|_2^2 = \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}}\varphi(\phi(\mathbf{x}_i))\underbrace{\boldsymbol{\Theta}(\phi(\mathbf{x}_i)^T)\boldsymbol{\Xi}}_{\dot{\mathbf{z}}_i}\|_2^2$$

> New loss function:

$$\lambda_1 \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \lambda_2 \frac{1}{N}\sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}}\varphi(\mathbf{z}_i)\boldsymbol{\Theta}(\mathbf{z}_i^T)\boldsymbol{\Xi}\|_2^2 + \lambda_3 \|\Xi\|_1$$

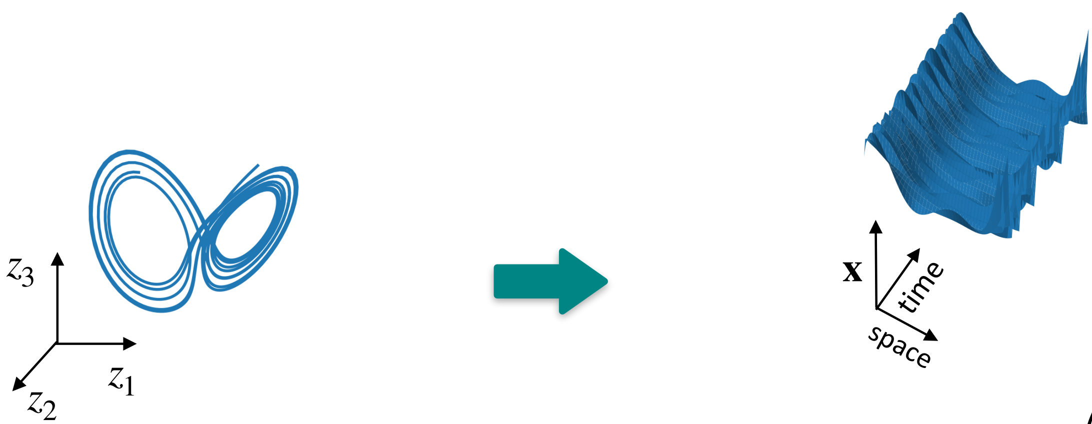
Achieving sparsity

- With L1 penalty alone, get model that has many very small coefficients but is not truly sparse

$$\lambda_1 \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \|\dot{\mathbf{x}}_i - \nabla_{\mathbf{z}} \varphi(\mathbf{z}_i) \boldsymbol{\Theta}(\mathbf{z}_i^T) \boldsymbol{\Xi}\|_2^2 + \lambda_3 \|\Xi\|_1$$

- Instead combine L1 penalty with sequential thresholding

Example problem



$$\mathbf{x}(t) = \mathbf{A}\mathbf{z}(t) + \mathbf{B}\begin{pmatrix} z_1^3(t) \\ z_2^3(t) \\ z_3^3(t) \end{pmatrix}$$

$$\mathbf{x}(t) \in \mathbb{R}^{128}$$

$$\mathbf{A}, \mathbf{B} \in \mathbb{R}^{128\times3}$$

Example problem

Lorenz model

Equations

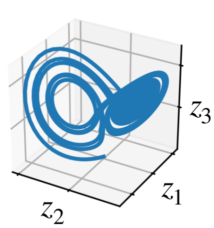
$$\dot{z}_1 = -10z_1 + 10z_2$$

$$\dot{z}_2 = 28z_1 - z_2 - z_1z_3$$

$$\dot{z}_3 = -2.7z_3 + z_1z_2$$

Coefficient Matrix Ξ

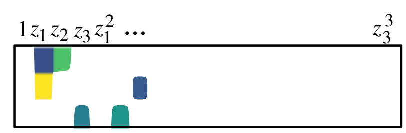
Dynamics



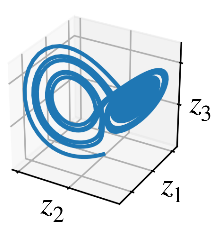
Example problem

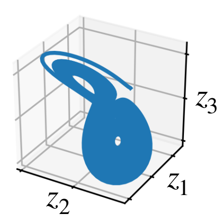
	Lorenz model	Discovered model	Discovered model (transformed)
Equations	$\dot{z}_1 = -10z_1 + 10z_2$	$\dot{z}_1 = -8.5z_2z_3$	$\dot{z}_1 = -10.2z_1 + 8.8z_2$
	$\dot{z}_2 = 28z_1 - z_2 - z_1z_3$	$\dot{z}_2 = 9.2 - 2.9z_2 + 1.1z_1z_3$	$\dot{z}_2 = 26.7z_1 - 8.5z_1z_3$
	$\dot{z}_3 = -2.7z_3 + z_1z_2$	$\dot{z}_3 = -8.8z_1 - 10.3z_3$	$\dot{z}_3 = -2.9z_3 + 1.1z_2$

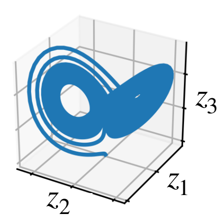
Coefficient Matrix Ξ



Dynamics







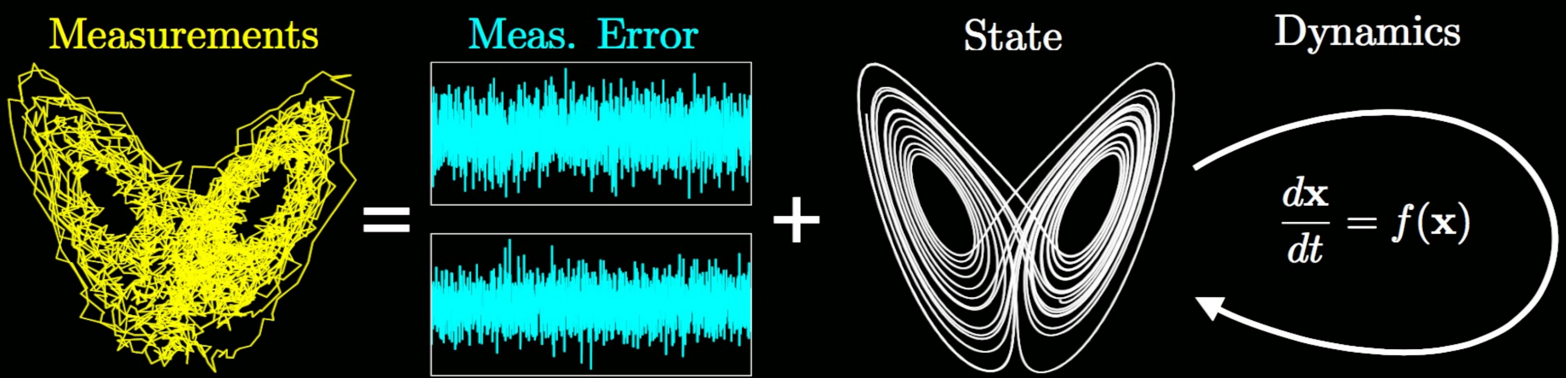
Realities of Life

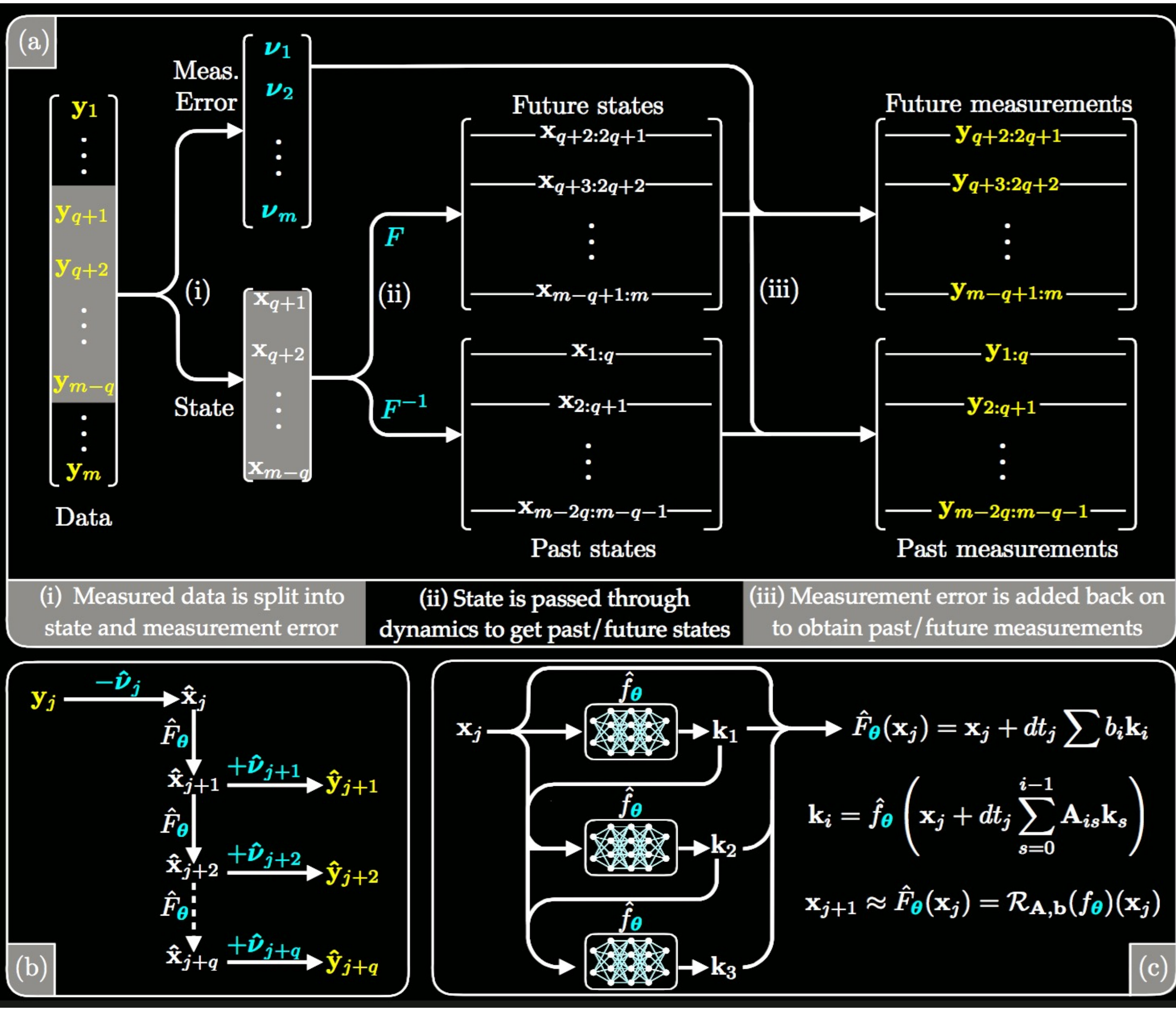
&

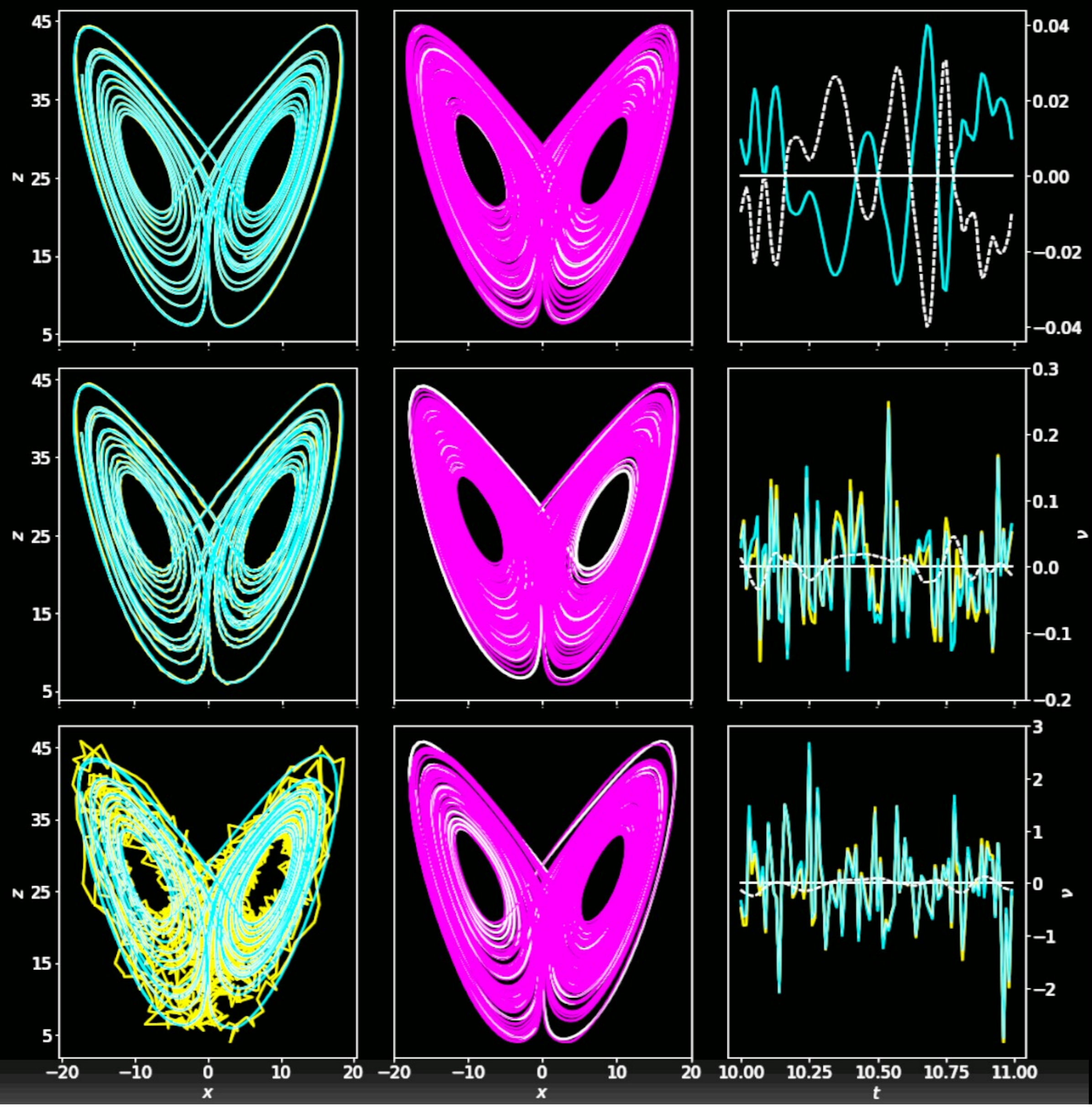
NN Magic

Noise

Sam Rudy


Measurements
Meas. Error
State
Dynamics
$\frac{d\mathbf{x}}{dt} = f(\mathbf{x})$


(a)
Meas.
Error
ν_1
ν_2
ν_m
y_1
y_{q+1}
y_{q+2}
y_{m-q}
y_m
Data
(i)
State
x_{q+1}
x_{q+2}
x_{m-q}
(ii)
F
F^{-1}
Future states
$x_{q+2:2q+1}$
$x_{q+3:2q+2}$
$x_{m-q+1:m}$
$x_{1:q}$
$x_{2:q+1}$
$x_{m-2q:m-q-1}$
Past states
(iii)
Future measurements
$y_{q+2:2q+1}$
$y_{q+3:2q+2}$
$y_{m-q+1:m}$
$y_{1:q}$
$y_{2:q+1}$
$y_{m-2q:m-q-1}$
Past measurements
(i) Measured data is split into state and measurement error
(ii) State is passed through dynamics to get past/future states
(iii) Measurement error is added back on to obtain past/future measurements
(b)
y_j $-\hat{\nu}_j$ $\hat{x}_j$
$\hat{F}_\theta$
$\hat{x}_{j+1}$ $+\hat{\nu}_{j+1}$ $\hat{y}_{j+1}$
$\hat{x}_{j+2}$ $+\hat{\nu}_{j+2}$ $\hat{y}_{j+2}$
$\hat{x}_{j+q}$ $+\hat{\nu}_{j+q}$ $\hat{y}_{j+q}$
(c)
x_j
$\hat{f}_\theta$
k_1
k_2
k_3
$\hat{F}_\theta(x_j) = x_j + dt_j \sum b_i k_i$
$k_i = \hat{f}_\theta \left(x_j + dt_j \sum_{s=0}^{i-1} A_{is} k_s \right)$
$x_{j+1} \approx \hat{F}_\theta(x_j) = \mathcal{R}_{A,b}(f_\theta)(x_j)$

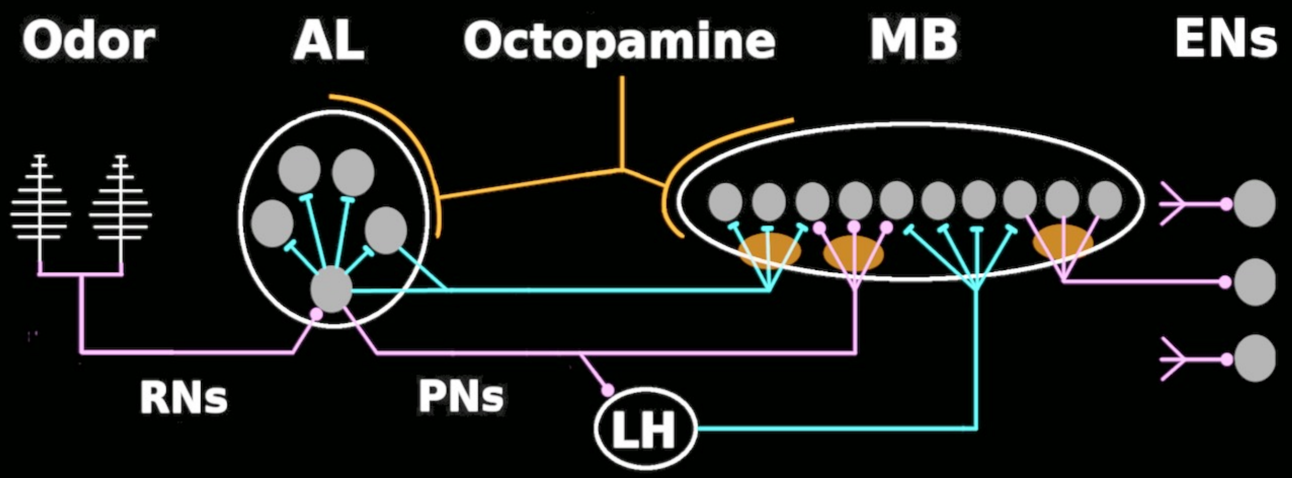


45
35
z 25
15
5
0.04
0.02
0.00
−0.02
−0.04
0.3
0.2
0.1
0.0
−0.1
−0.2
y
3
2
1
0
−1
−2
−20 −10 0 10 20
x
10.00 10.25 10.50 10.75 11.00
t

Fast Learning

Charles Delahunt

Moth Olfactory System

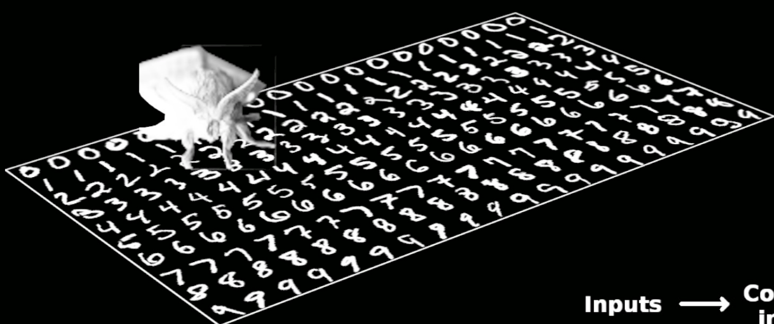


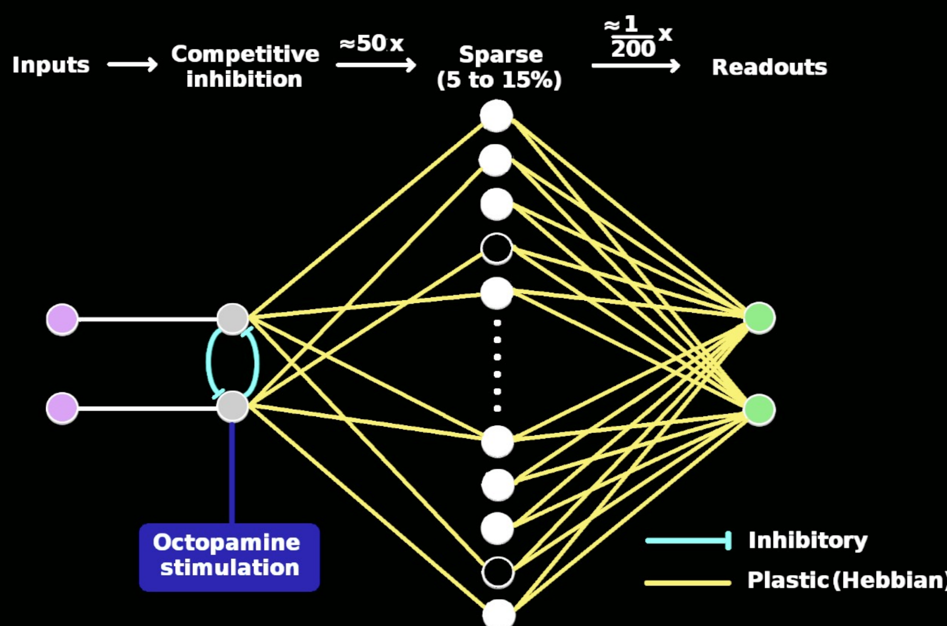
Riffell et al. Science 2013
Campbell et al. J Neuro 2013
Olson et al. Neuron 2010
Turner et al. J NeuroPhysiol 2008 Hong, Wilson. Neuron 2015

Gupta, Stopfer. J NeuroSci 2012
Silbering et al. J NeuroSci 2003
Galizia. Eur J NeuroSci 2014
Caron et al. Nature 2013

Delahunt, Riffell, Kutz (2018) FNS

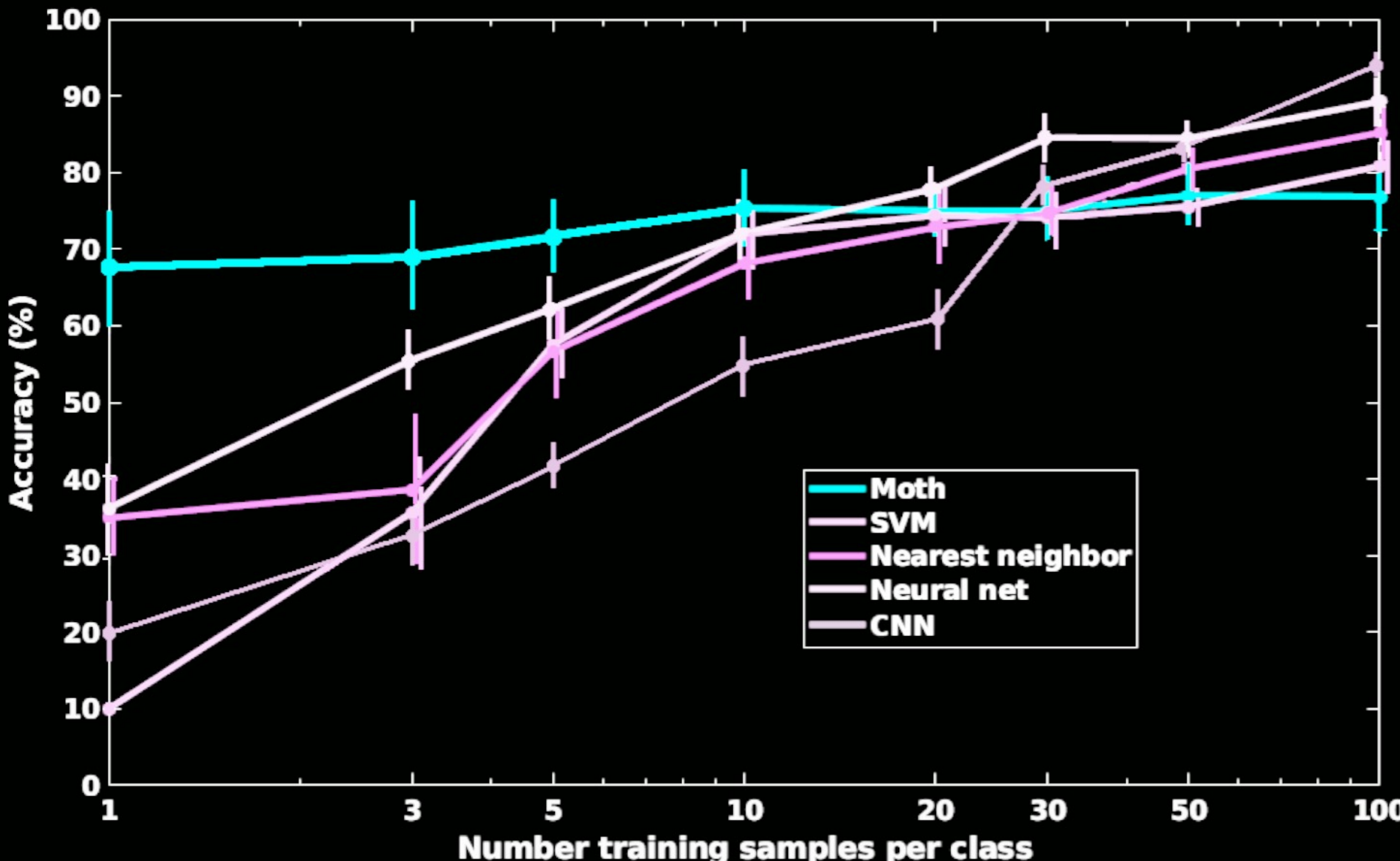
Rapid Learning in NNs





Delahunt & Kutz (2018) arxiv

Comparisons



Conclusion

Model Discovery: Sparse regression provides parsimonious dynamical models

Coordinates: Learning Koopman embeddings can provide optimal basis for dynamics

Neural Networks: Structure and function matter

- Connect discovery and coordinates
- Structure can lead to fast (one-shot) learning with limited data
- Discovery of improved coordinate embeddings through decoding

COMING SOON: A multi scale physics challenge set

Coming Soon

Multiscale Physics Challenge Data Set: Taking on the model of computer vision

Coming Soon

Multiscale Physics Challenge Data Set: Taking on the model of computer vision