Using dynamical systems ideas to combine in a principled way data-driven models and domain-driven models

Michael W. Mahoney
ICSI and Department of Statistics, UC Berkeley

April 2020

(Joint work with Benjamin Erichson,
Michael Muehlebach, Omri Azencot, and others.)

Outline

Introduction and Overview

Physics-informed Autoencoders for Lyapunov-stable Fluid Flow Prediction (Benjamin Erichson and Michael Muehlebach)

Forecasting Sequential Data using Consistent Koopman Autoencoders (Omri Azencot, Benjamin Erichson, and Vanessa Lin)

Conclusions

Paradigms of Modeling Complex Systems

Statistical Modelling (Data/Theory-driven)

$$y = Ax + \epsilon$$

- Interpretable
- Strong assumptions
- Low expressivity

Machine Learning (Data-driven)

$$y = F(x) + \epsilon$$

- Black-box
- No assumptions
- High expressivity

Dynamical Systems (Theory-driven)

$$\dot{y} = f(t, y, x)$$

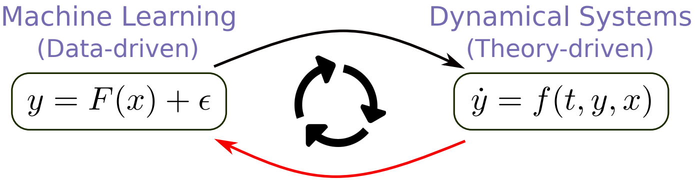
- Gray-box
- Noise ?!
- Robust / Stable

What can we learn from dynamical systems and control theory?

Residual Networks (ResNets)	$\longleftrightarrow$	Differential Equations
Network Architecture Design	$\longleftrightarrow$	Numerical Methods
Training	$\longleftrightarrow$	Optimal Control

Recent Related Mahoney Lab's Research Outcomes

- ML to Dynamical Systems:
  - Shallow neural networks for fluid flow reconstruction with limited sensors (Erichson et al.)



- Ideas from Dynamical Systems to ML:
  - ANODEV2: A coupled neural ODE framework (Gholami et al.)
  - Stochastic normalizing flows (Hodgkinson et al.)
  - Physics-informed autoencoders for lyapunov-stable fluid flow prediction (Erichson et al.)
  - Forecasting sequential data using consistent koopman autoencoders (Azencot et al.)
  - Improving ResNets with a corrected dynamical systems interpretation (Queiruga et al.)
  - Noise-response analysis for rapid detection of backdoors in deep nets (Erichson et al.)

Connection between Deep Learning and Differential Equations

- The essential building blocks of ResNets are so-called residual units.

$$x_{t+1} = \epsilon \cdot x_t + \sigma_t(x_t, \theta_t). \tag{1}$$

- The function $\sigma_t : \mathbb{R}^n \to \mathbb{R}^n$ denotes the t-th residual module (a non-linear map), parameterized by θ_t, which takes a signal $x_t \in \mathbb{R}^n$ as input. ϵ is the step size.
- For simplicity, let's consider a linear unit

$$x_{t+1} = \epsilon \cdot x_t + Ax_t. \tag{2}$$

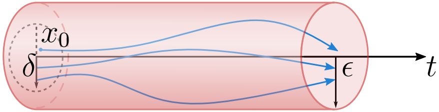
- Through the lens of differential equations, residual units can be seen as a some (!?) discretization scheme for the following ordinary differential equation:

$$\frac{\partial x}{\partial t} = Ax. \tag{3}$$

- This connection between differential equations and residual units can help to study network architecture as well as provide inspiration for the design of new network architectures.

What can we Learn from Dynamical Systems Theory?

- Dynamical systems theory is mainly concerned with describing the **long-term qualitative behavior** of dynamical systems, which typically can be describe as differential equations.
- **Stability theory** plays an essential role in the analysis of differential equation.
- We might be interested to study whether trajectories of a given dynamical systems, under small perturbations of the initial condition x_0, are stable.



- If the dynamics $\frac{\partial x}{\partial t} = Ax$ are linear, stability can be checked with an eigenvalue analysis.
- We can use linearization or input-to-state stability to study nonlinear systems.
- Does stability matter in deep learning? Well, it depends ....
- Feedforward neural networks (FNNs): each residual unit takes only a single step. Thus, stability might not matter?!
- Recurrent neural networks: stability matters! If the recurrent unit is unstable, then we observe exploding gradients. We will discuss this later.

How can we Integrate Prior Physical Knowledge?

- **Option 1: Physics-informed network architectures.** We integrate prior knowledge (e.g., symmetries) via specialized physics-informed layers or convolution kernels.

$$\theta_k = T(W_k) := \beta \cdot (W + W^T) + (1-\beta) \cdot (W - W^T) \tag{4}$$

- **Option 1: Physics-informed regularizers.** We integrate prior knowledge (e.g., stability) via additional energy terms

$$\min_\theta \mathcal{L}(\theta) := \frac{1}{n} \sum_{i=1}^{n} \underbrace{\ell_i(h_\theta(x_i), y_i)}_{\text{Loss}} + \lambda \cdot \underbrace{\mathcal{R}(\theta_k)}_{\text{regularizer}}, \tag{5}$$

- **Option 1: Physics-constrained models.** We integrate prior knowledge (e.g., an ODE model) via additional constraints on the outputs

$$\min_\theta \mathcal{L}(\theta) := \frac{1}{n} \sum_{i=1}^{n} \ell_i(h_\theta(x_i), y_i) \quad \text{s.t.} \quad \mathcal{R}(f_\theta(x)) \leq \eta, \tag{6}$$

Outline

Introduction and Overview

Physics-informed Autoencoders for Lyapunov-stable Fluid Flow Prediction (Benjamin Erichson and Michael Muehlebach)

Forecasting Sequential Data using Consistent Koopman Autoencoders (Omri Azencot, Benjamin Erichson, and Vanessa Lin)

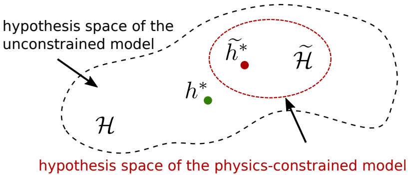
Conclusions

Physics-constrained learning (PCL)

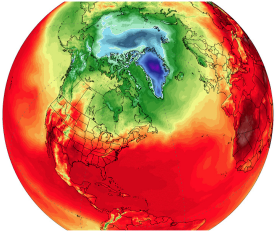
- Supervised ML aims to learn a model $\mathcal{H}$ that best maps a set of inputs $\mathcal{X}$ to a set of outputs $\mathcal{Y}$:

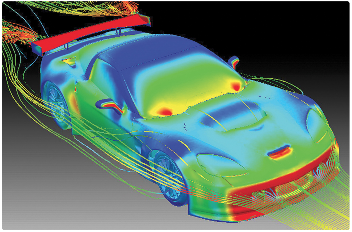
$$\mathcal{H} : \mathcal{X} \to \mathcal{Y}$$

- We hope that this model also works on new inputs.



PCL aims to introduce prior knowledge about the problem into the learning process.





Problem setup: Fluid flow prediction

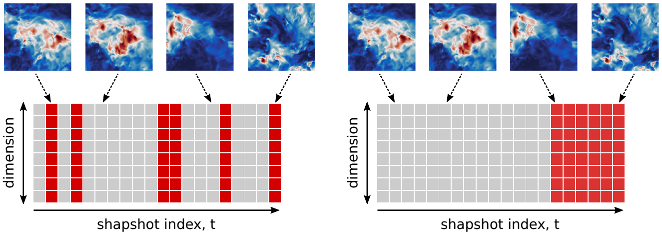
- We assume that the dynamical system of interest can be modeled as

$$\mathbf{x}_{t+1} = \mathcal{A}(\mathbf{x}_t) + \eta_t, \quad t = 0, 1, 2, \ldots, T.$$

- In a data-driven setting we might only have access to (high-dimensional) observations

$$\mathbf{y}_t = \mathcal{G}(\mathbf{x}_t) + \xi_t, \quad t = 0, 1, 2, \ldots, T.$$

- Given a sequence of observations $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_T \in \mathbb{R}^m$ for training, the objective of this work is to learn a model which maps the snapshot $\mathbf{y}_t$ to $\mathbf{y}_{t+1}$.



Physics-agnostic model

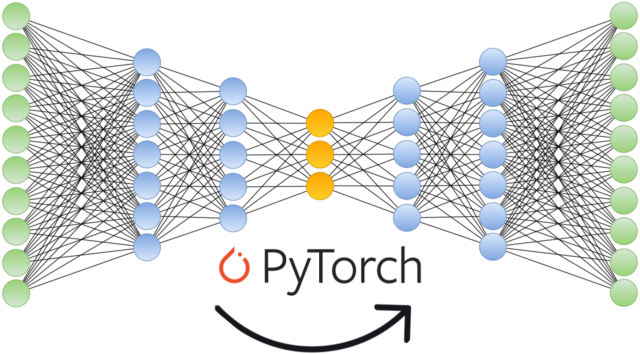
- Given the pairs $\{\mathbf{y}_t, \mathbf{y}_{t+1}\}_{t=1,2,\ldots T}$, we train a model by minimizing the MSE

$$\min \frac{1}{T} \sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \mathcal{F}(\mathbf{y}_t)\|_2^2.$$

- During inference time, we can obtain predictions by composing the learned model k-times

$$\hat{\mathbf{y}}_k = \mathcal{F} \circ \mathcal{F} \circ \mathcal{F} \circ \ \ldots \ \circ \mathcal{F}(\mathbf{y}_0).$$

A typical black box model



Automatic differentiation

- I will talk more about the specific architecture later....

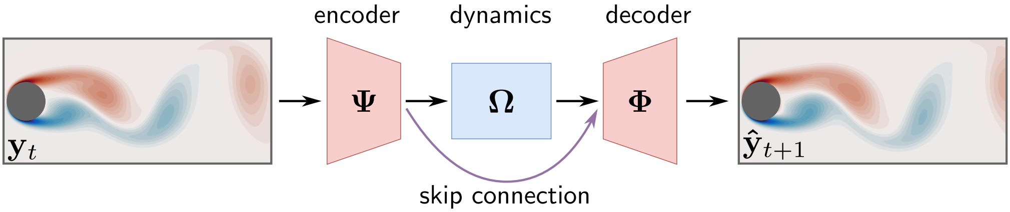
From black box to gray box models

- We to add meaningful constraints to our model:

$$\min \frac{1}{T}\sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \boldsymbol{\Phi} \circ \boldsymbol{\Omega} \circ \boldsymbol{\Psi}(\mathbf{y}_t)\|_2^2 + \lambda \|\mathbf{y}_t - \boldsymbol{\Phi} \circ \boldsymbol{\Psi}(\mathbf{y}_t)\|_2^2.$$

- If the model obeys the assumption that $\boldsymbol{\Psi}$ approximates $\mathcal{G}^{-1}$, then we have that

$$\hat{\mathbf{y}}_k \approx \boldsymbol{\Phi} \circ \boldsymbol{\Omega}^k \circ \boldsymbol{\Psi}(\mathbf{y}_0).$$



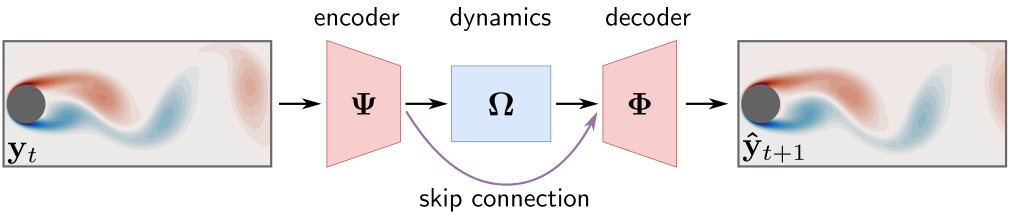
From black box to gray box models

- We start by adding a meaningful constraint to our model:

$$\min \frac{1}{T}\sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \mathbf{\Phi} \circ \mathbf{\Omega} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \lambda \|\mathbf{y}_t - \mathbf{\Phi} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \kappa\, \rho(\mathbf{\Omega}).$$

- If the model obeys the assumption that $\mathbf{\Psi}$ approximates $\mathcal{G}^{-1}$, then we have that

$$\hat{\mathbf{y}}_k \approx \mathbf{\Phi} \circ \mathbf{\Omega}^k \circ \mathbf{\Psi}(\mathbf{y}_0).$$

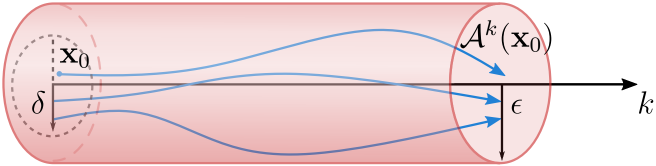


Lyapunov stability in a nutshell

- The origin of a dynamic system

$$\mathbf{x}_{t+1} = \mathcal{A}(\mathbf{x}_t) + \eta_t \quad t = 0, 1, 2, \ldots, T.$$

is stable if all trajectories starting arbitrarily close to the origin (in a ball of radius δ) remain arbitrarily close (in a ball of radius ϵ).



- If the dynamics $\mathcal{A}$ are linear, stability can be checked with an eigenvalue analysis.

Lyapunov's method... an idea from over 120 years ago[1]

- For linear systems, Lyapunov's second method states that a dynamic system

  $$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \eta_t \quad t = 0, 1, 2, \ldots, T$$

  is stable if and only if for any (symmetric) positive definite matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ there exists a (symmetric) positive definite matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ satisfying

  $$\mathbf{A}^\top \mathbf{P}\mathbf{A} - \mathbf{P} = -\mathbf{Q}.$$

[1] https://stanford.edu/~boyd/papers/pdf/springer_15_colloquium.pdf

Lyapunov's method... an idea from over 120 years ago[1]

- For linear systems, Lyapunov's second method states that a dynamic system

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \eta_t \quad t = 0, 1, 2, \ldots, T$$

  is stable if and only if for any (symmetric) positive definite matrix $\mathbf{Q} \in \mathbb{R}^{n\times n}$ there exists a (symmetric) positive definite matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ satisfying

$$\mathbf{A}^\top \mathbf{P}\mathbf{A} - \mathbf{P} = -\mathbf{Q}.$$

- Using this idea, we impose that the symmetric matrix $\mathbf{P}$, defined by

$$\mathbf{\Omega}^\top \mathbf{P}\mathbf{\Omega} - \mathbf{P} = -\mathbf{I},$$

  is positive definite.

[1] https://stanford.edu/~boyd/papers/pdf/springer_15_colloquium.pdf

To gain some intuition...

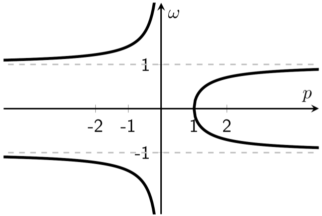
- ... we consider the case where $\boldsymbol{\Omega}$ is diagonalizable and $\mathbf{Q}$ chosen appropriately.
- Then, for a particular choice of coordinates the following problem

$$\boldsymbol{\Omega}^\top \mathbf{P} \boldsymbol{\Omega} - \mathbf{P} = -\mathbf{I}, \tag{1}$$

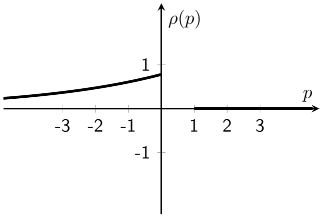
reduces to the system of linear equations

$$\omega_i p_i \omega_i - p_i = -1, \tag{2}$$

where ω_i, p_i, for $i = 1, 2, \ldots, n$, denote the eigenvalues of $\boldsymbol{\Omega}$ and $\mathbf{P}$, respectively.



(a) Discrete-time Lyapunov function.



(b) Stability promoting prior.

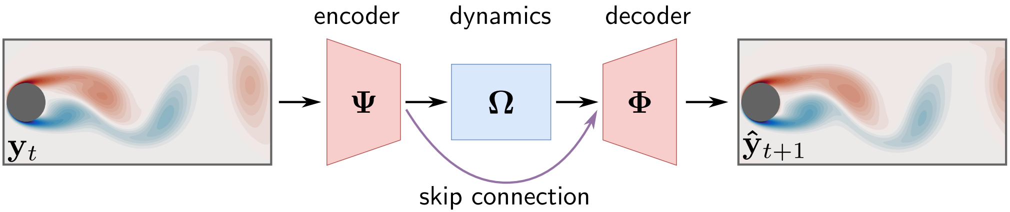
Physics-aware model that preserves stability

- The physics-informed autoencoder is trained by minimizing the following objective

$$\min \frac{1}{T}\sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \mathbf{\Phi} \circ \mathbf{\Omega} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 \; + \; \lambda\|\mathbf{y}_t - \mathbf{\Phi} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \; \kappa \sum_i \rho(p_i).$$

- The prior p can take various forms. We use the following in our experiments:

$$\rho(p) := \begin{cases} \exp\left(-\frac{|p-1|}{\gamma}\right) & \text{if } p < 0 \\ 0 & \text{otherwise.} \end{cases}$$



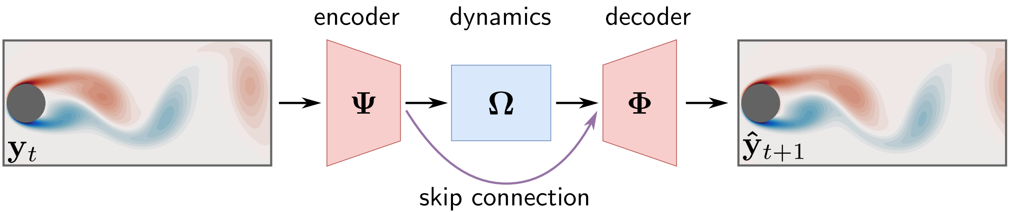
Physics-aware model that preserves stability

- The physics-informed autoencoder is trained by minimizing the following objective

$$\min \frac{1}{T} \sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \mathbf{\Phi} \circ \mathbf{\Omega} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \|\mathbf{y}_{t+2} - \mathbf{\Phi} \circ \mathbf{\Omega} \circ \mathbf{\Omega} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \lambda \|\mathbf{y}_t - \mathbf{\Phi} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \kappa \sum_i \rho(p_i).$$

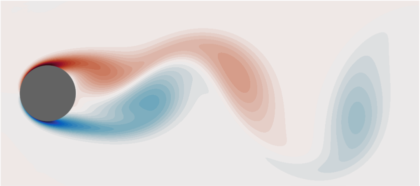
- The prior p can take various forms. We use the following in our experiments:

$$\rho(p) := \begin{cases} \exp\left(-\frac{|p-1|}{\gamma}\right) & \text{if } p < 0 \\ 0 & \text{otherwise.} \end{cases}$$

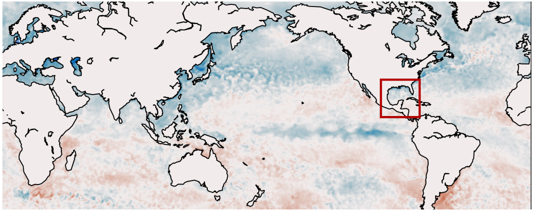


Examples that we consider

- Flow past the cylinder.

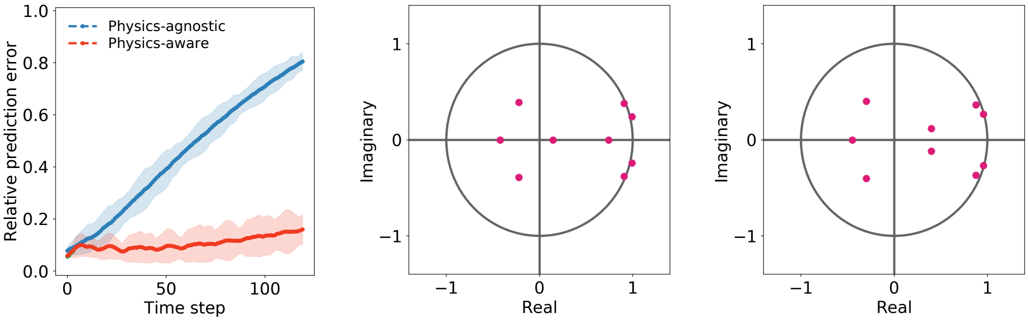


- Daily sea surface temperature data of the gulf of Mexico over a period of 6 years.



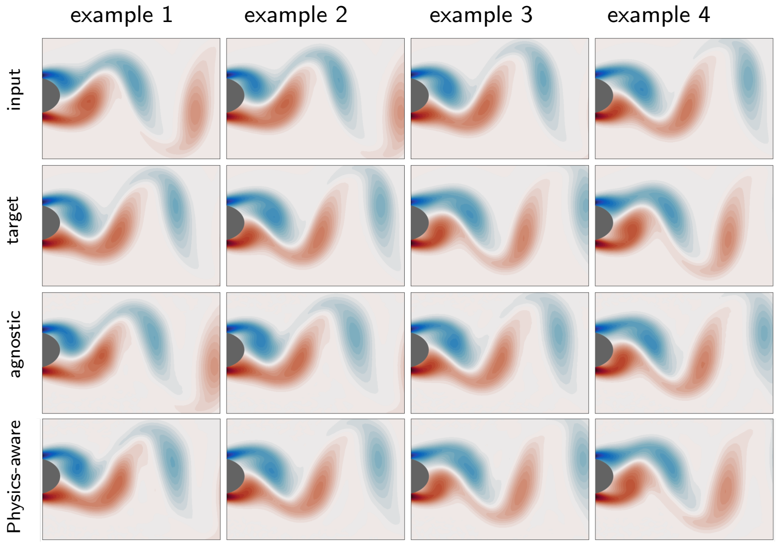
Prediction performance for flow past the cylinder (without weight decay)

$$\min \frac{1}{T}\sum_{t=0}^{T} \|\mathbf{y}_{t+1}-\boldsymbol{\Phi}\circ\boldsymbol{\Omega}\circ\boldsymbol{\Psi}(\mathbf{y}_t)\|_2^2+\|\mathbf{y}_{t+2}-\boldsymbol{\Phi}\circ\boldsymbol{\Omega}\circ\boldsymbol{\Omega}\circ\boldsymbol{\Psi}(\mathbf{y}_t)\|_2^2+\lambda\|\mathbf{y}_t-\boldsymbol{\Phi}\circ\boldsymbol{\Psi}(\mathbf{y}_t)\|_2^2+\kappa\sum_i \rho(p_i).$$

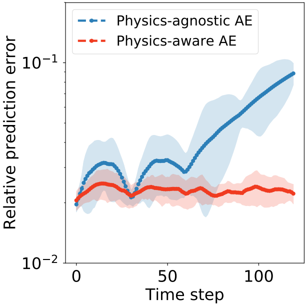


(a) With LR $1\text{e}{-2}$.   (b) Physics-agnostic (blue).   (c) Physics-aware (red).

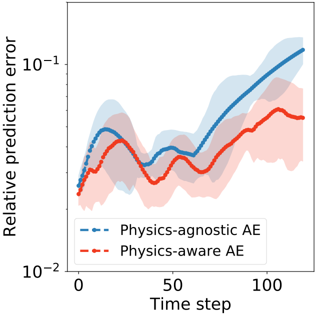
Visual results for flow past the cylinder (100 time steps)



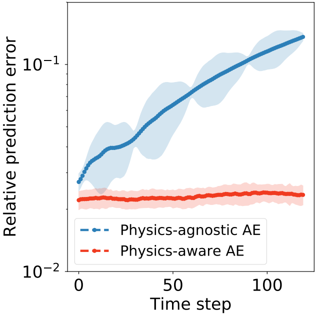
More results for the flow past the cylinder (with weight decay)



(a) With LR $1e{-}2$ and WD $1e{-}6$.



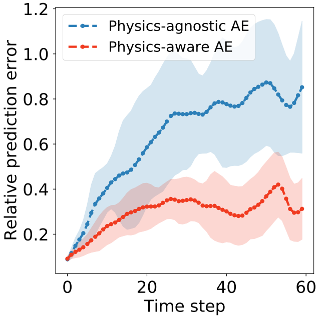
(b) With LR $1e{-}2$ and WD $1e{-}8$.



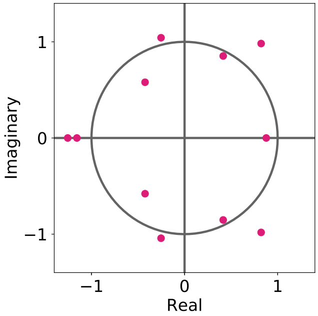
(c) With LR $5e{-}3$ and WD $1e{-}6$.

Results for the sea surface temperature data

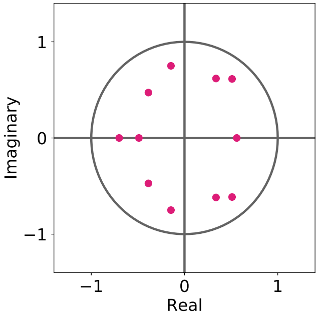
$$\min \frac{1}{T} \sum_{t=0}^{T} \|\mathbf{y}_{t+1} - \mathbf{\Phi} \circ \mathbf{\Omega} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \lambda \|\mathbf{y}_t - \mathbf{\Phi} \circ \mathbf{\Psi}(\mathbf{y}_t)\|_2^2 + \kappa \sum_i \rho(p_i).$$



(a) With LR $1\mathrm{e}{-2}$.

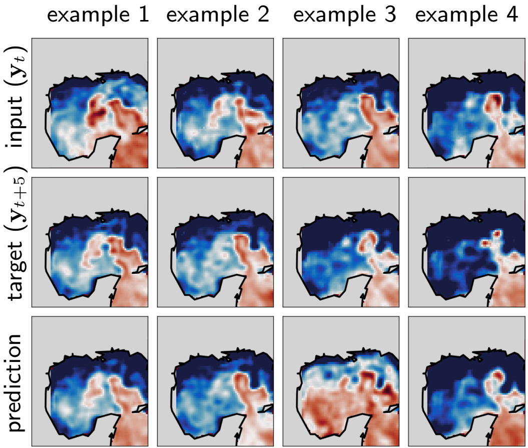


(b) Physics-agnostic (blue).

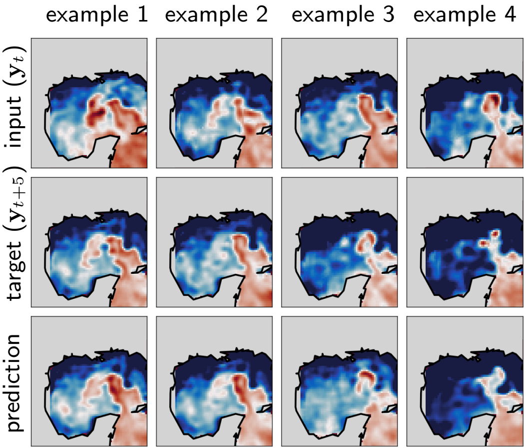


(c) Physics-aware (red).

Visual results for the sea surface temperature data



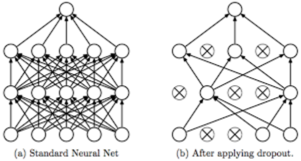
(a) Physics-agnostic model.



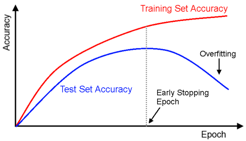
(b) Physics-aware model.

Have we just proposed a new regularizer?

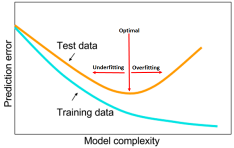
Every adjustable knob and switch — and there are many[2] — is regularization.



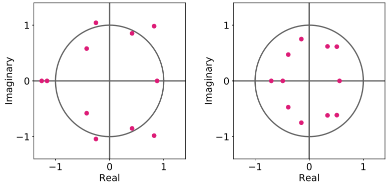
(a) Dropout.



(b) Early stopping.



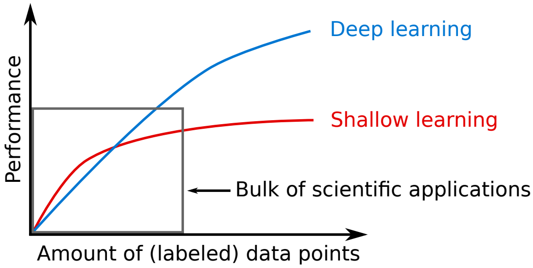
(c) Bottleneck.

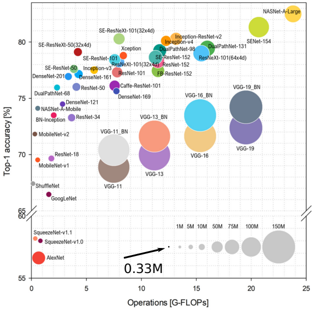


(d) Stability.

[2]https://arxiv.org/pdf/1710.10686.pd

We use shallow networks...





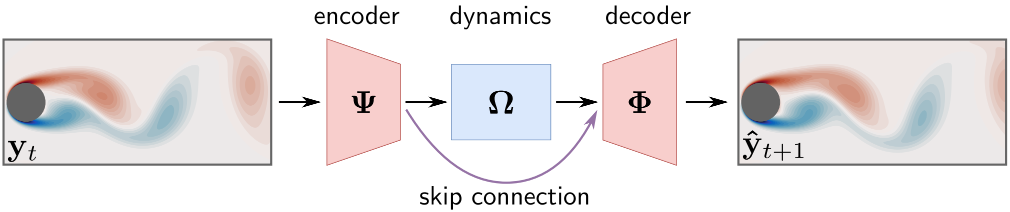
[3]

	very shallow	(our) shallow	deeper
Computational demands:	low	☺☺☺	high
Time for hyper-parameter tuning:	low	☺☺☺	high
Complexity of architecture design:	low	☺☺☺	high
Inference time:	low	☺☺☺	high
Carbon footprint:	low	☺☺☺	high

[3] Adapted from `https://arxiv.org/pdf/1810.00736.pdf`

Summary

- *Physics-informed* autoencoders can help to improve the generalization performance.
- Caveat of physics-informed learning are complicated loss functions: $\mathcal{L}_1 + \gamma\mathcal{L}_2 + \kappa\mathcal{L}_3 + \ldots$.
- Next steps: non-linear dynamics, recurrent networks and parameterized layers.



Outline

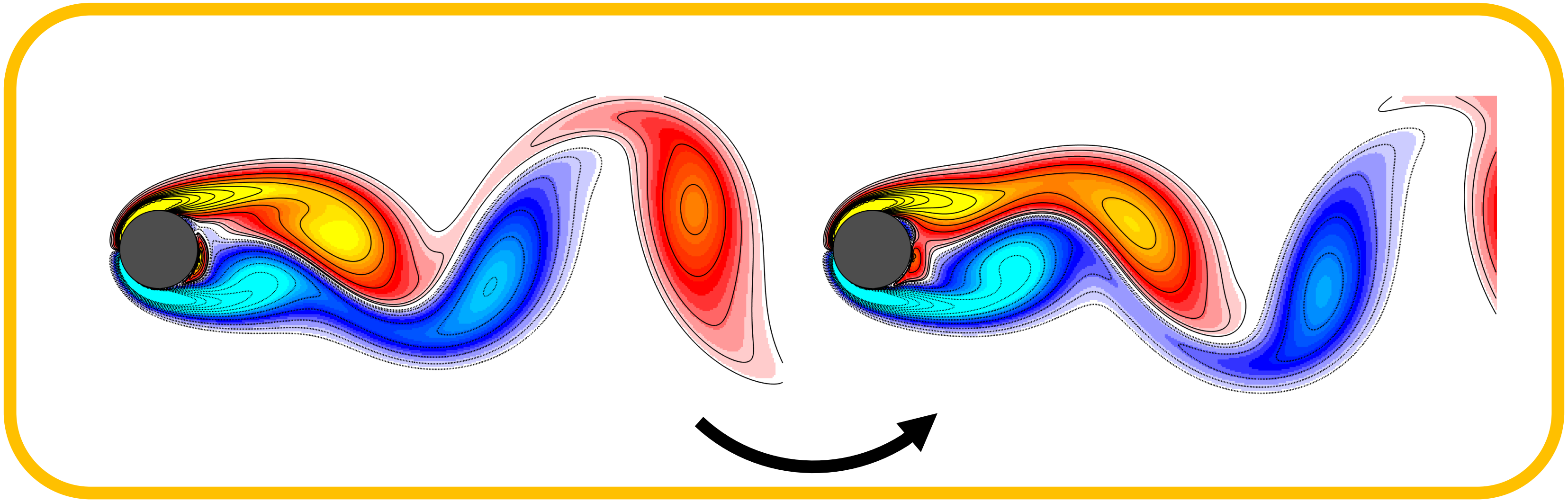
Introduction and Overview

Physics-informed Autoencoders for Lyapunov-stable Fluid Flow Prediction (Benjamin Erichson and Michael Muehlebach)

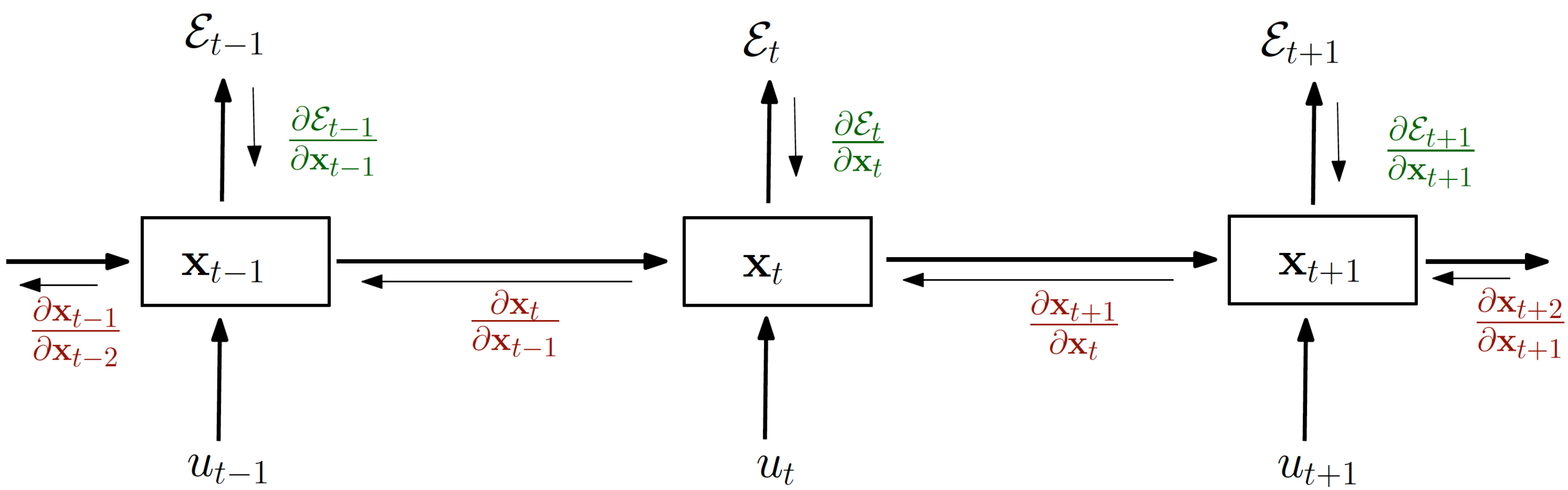
Forecasting Sequential Data using Consistent Koopman Autoencoders (Omri Azencot, Benjamin Erichson, and Vanessa Lin)

Conclusions

Task: Prediction of Future Data



Recurrent Neural Networks



Pascanu et al., ICML '13

Vanilla RNN

state nonlinearity input

$$x_{t+1} = \sigma(Wx_t + Vu_{t+1})$$

hidden-to-hidden input-to-hidden

Advantages of Vanilla RNN

$$x_{t+1} = \sigma(Wx_t + Vu_{t+1})$$

- Weights are shared across time
- RNN can simulate a universal Turing machine (?)
  Siegelmann and Sontag, ‘91
- Accomodates all systems in table (?)

Recursive expansion

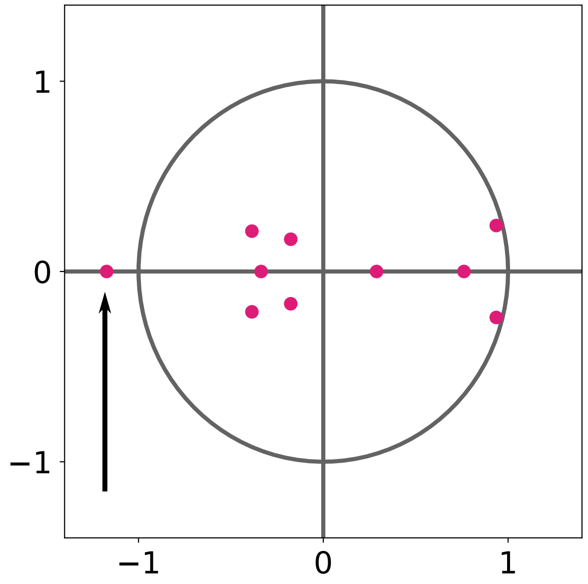
$$
\begin{aligned}
x_{t+1} &= \sigma(Wx_t + Vu_{t+1}) \\
&= \sigma(W\sigma(Wx_{t-1} + Vu_t) + Vu_{t+1}) \\
&= \sigma(W\sigma(W\sigma(Wx_{t-2} + Vu_{t-1}) + Vu_t) + Vu_{t+1})
\end{aligned}
$$

Increasing “nonlinear” powers of W!

Practical Challenges

- Exploding/Vanishing gradients

- Analyzed via the spectrum of W: Arjovsky et al. '16



Practical Challenges

- (too) Constrained hidden-to-hidden weights

$$x_{t+1} = \cdots + \sigma W \sigma V u_t + (\sigma W)^2 \sigma V u_{t-1} + \cdots$$

zero hidden state

- Powers of σW range from tens to hundreds!

Physics-based “RNN”

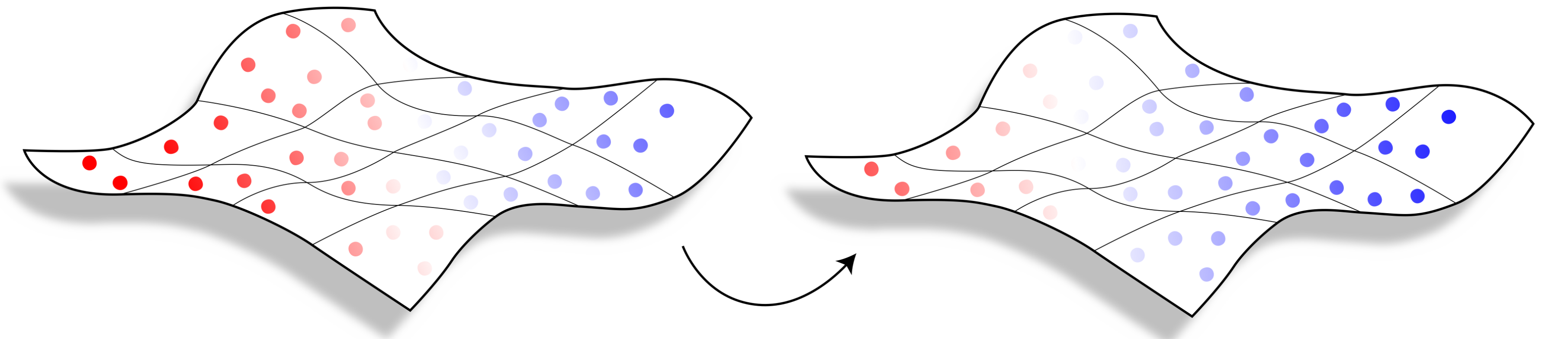
Lagrangian mechanics, Lutter et al., ’19

Hamiltonian dynamics, Greydanus et al., ‘19, …

Koopman methods, Takeishi et al. ‘17, …

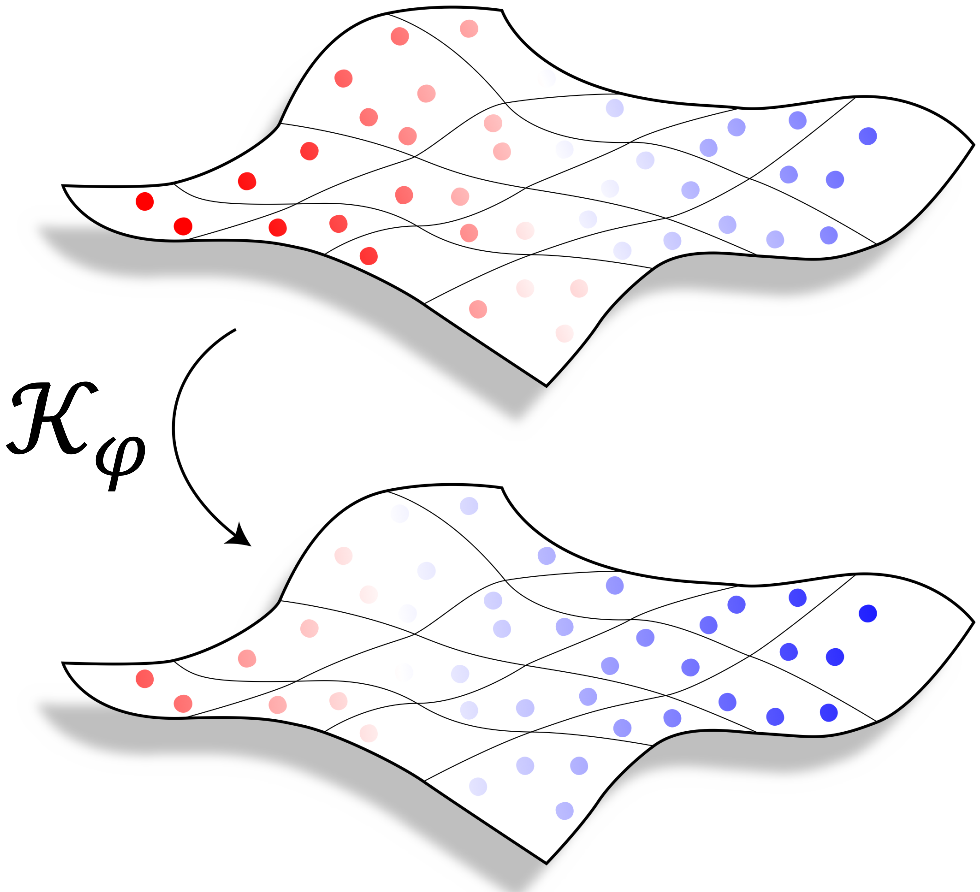
Dynamical Systems via Koopman

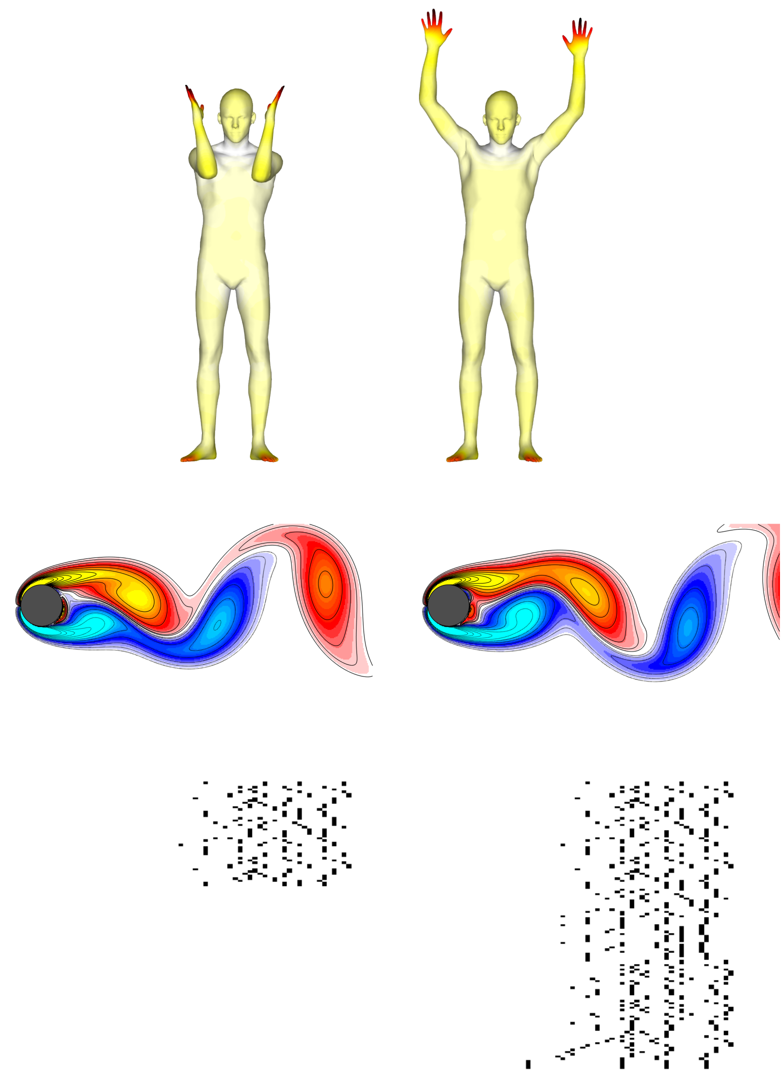
$$z_{k+1} = \varphi(z_k) \quad \Longrightarrow \quad \mathcal{K}_\varphi f(z_k) = f\big(\varphi(z_k)\big)$$



$\mathcal{K}_\varphi$

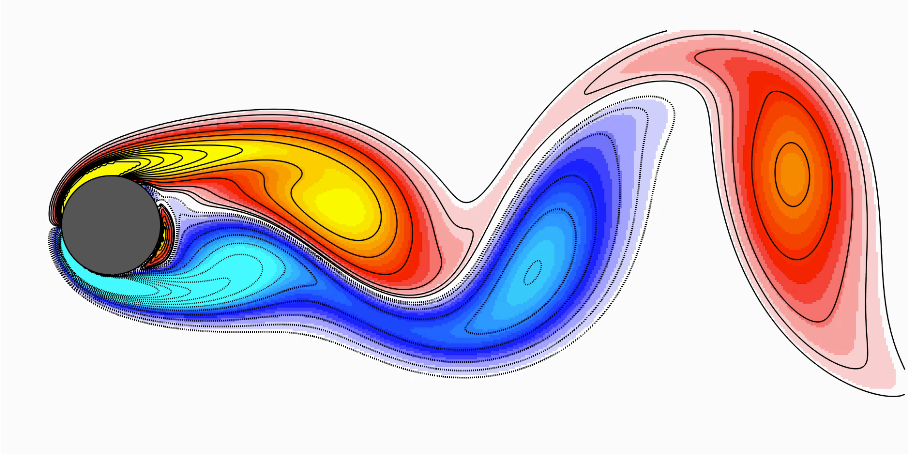
Dynamical Systems via Koopman

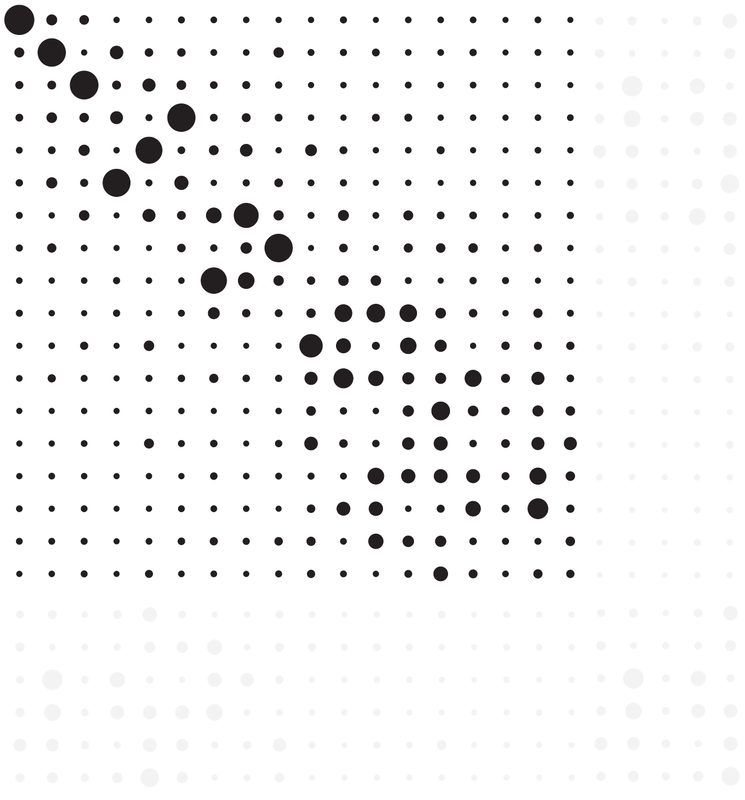




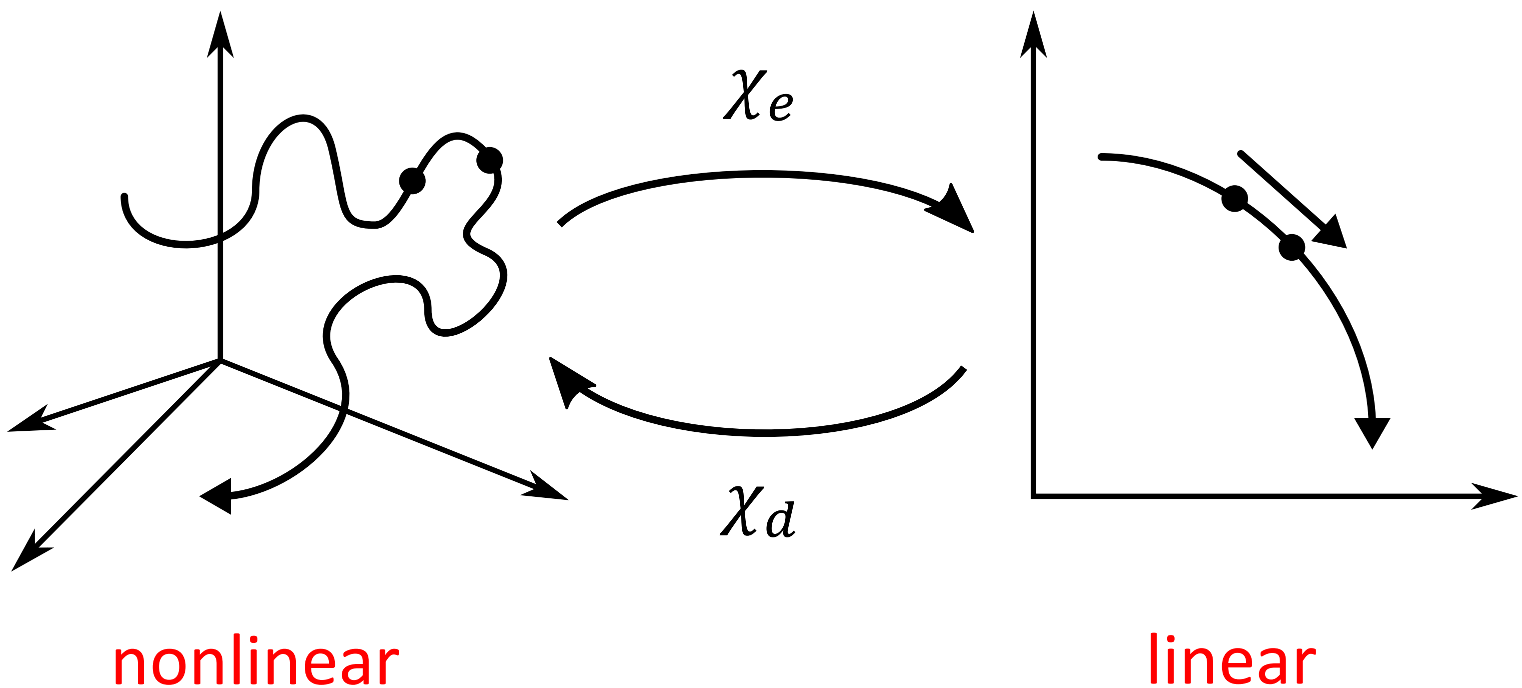
Koopman Operators

$$\mathcal{K}_{\varphi} f(z_k) = f\big(\varphi(z_k)\big)$$





Linearizing Data Transformation



Dynamic Mode Decomposition

1. Time series data in matrices

$$F = [f_j], \qquad G = [g_j],$$

2. Compute PCA\POD modes

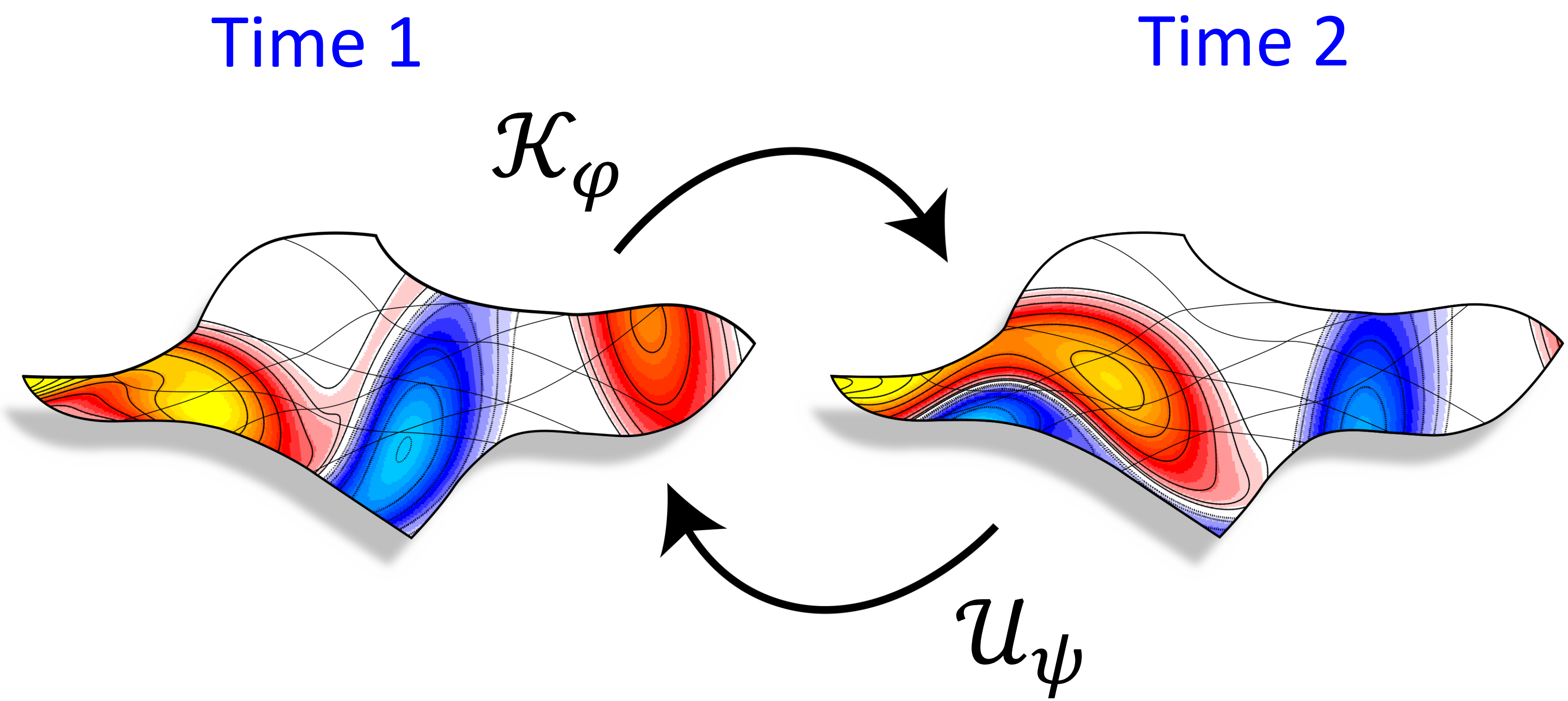
$$F = U_F S_F V_F^*, \qquad G = U_G S_G V_G^*$$

3. Solve

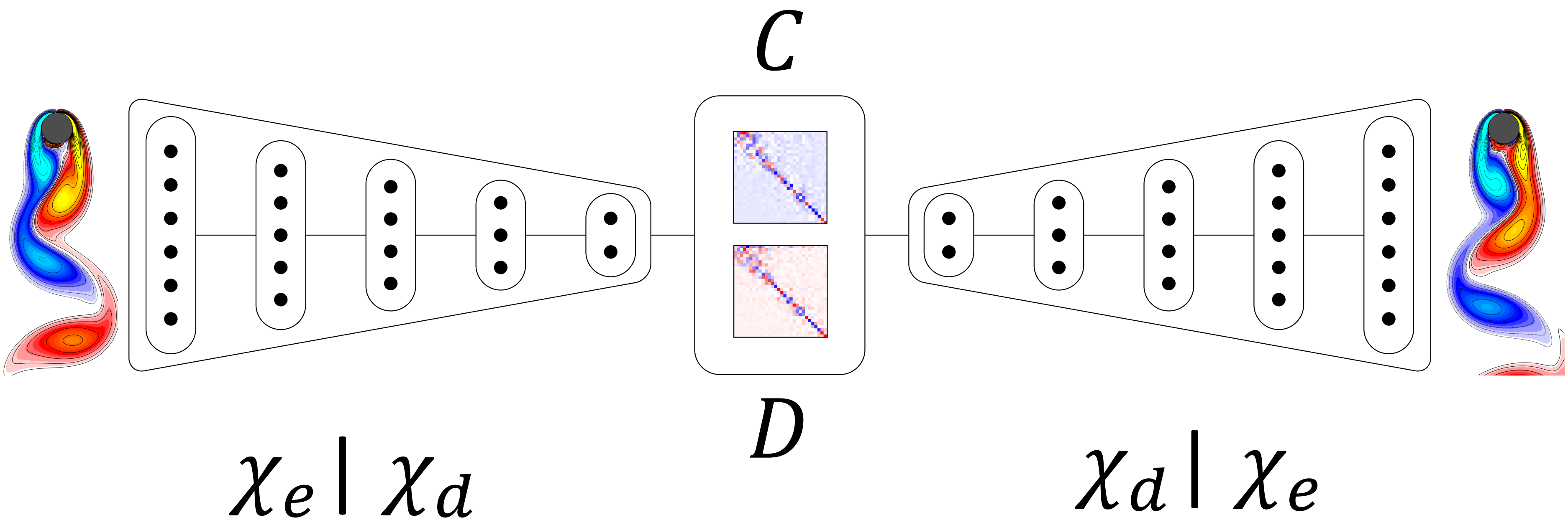
$$\min_C \left| C U_F^T F - U_G^T G \right|_F^2$$

Schmid, JFM '10

Our Approach



Deep Koopman Autoencoders



Azencot*, Erichson*, Lin, and Mahoney, '19

Deep Koopman Autoencoders

Reconstruction/fwd prediction/bwd prediction:

$$\begin{aligned}\tilde{u}_t &= \chi_d \circ \chi_e(u_t)\\ \hat{u}_{t+1} &= \chi_d \circ C \circ \chi_e(u_t)\\ \check{u}_{t-1} &= \chi_d \circ D \circ \chi_e(u_t)\end{aligned}$$

Our hidden state: $x_t = \chi_e(u_t)$

short-term dependencies

Forward Prediction in Linear Space

Prediction over l steps = Apply l times C:

$$\hat{u}_{t+l} = \chi_d \circ C^l \circ \chi_e(u_t)$$

given that $\chi_d \circ \chi_e = \mathrm{id}$!

Loss Function Terms

Long-term (fwd+bwd) predictions:

$$\mathcal{E}_{fwd} = \frac{1}{2\lambda_s n} \sum_{l=1}^{\lambda_s} \sum_{t=1}^{n} |u_{t+l} - \hat{u}_{t+l}|_2^2$$

$$\mathcal{E}_{bwd} = \frac{1}{2\lambda_s n} \sum_{l=1}^{\lambda_s} \sum_{t=1}^{n} |u_{t-l} - \check{u}_{t-l}|_2^2$$

Reconstruction:

$$\mathcal{E}_{id} = \frac{1}{2n} \sum_{t=1}^{n} |u_t - \tilde{u}_t|_2^2$$

Bijections and invertible Koopman

Theorem:

The mapping φ is *bijective,* if and only if the associated Koopman operators satisfy

$$\langle \xi_i, \mathcal{U}\mathcal{K}\xi_j \rangle_{\mathcal{M}} = \delta_{ij}$$

Consistent Maps

Theorem:

The discrete map φ is *consistent,* if and only if the following condition holds

$$\sum_k |D_{k*}C_{*k} - I_k|_F^2 = 0$$

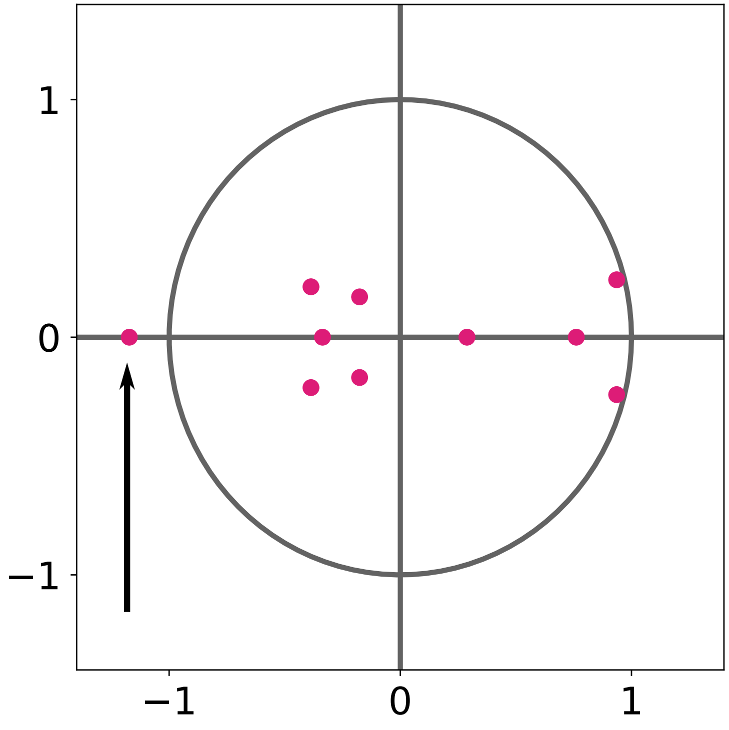
Important: C and D must come from point-to-point maps

Consistency Loss Term

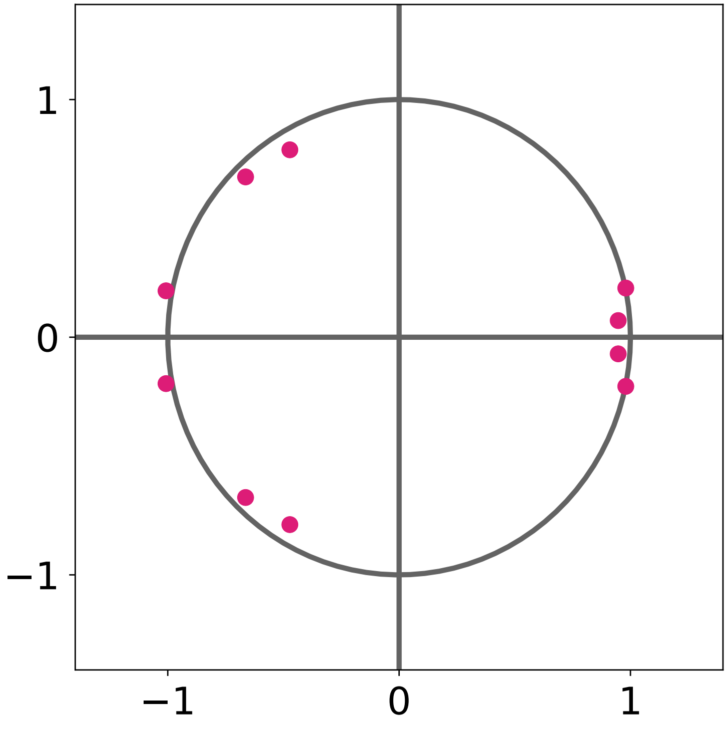
Penalize symmetrically:

$$\begin{aligned}\mathcal{E}_{con} \quad &= \quad \sum_{k=1}^{\kappa} \frac{1}{2k} |D_{k*}C_{*k} - I_k|_F^2 \\ &+ \quad \sum_{k=1}^{\kappa} \frac{1}{2k} |C_{k*}D_{*k} - I_k|_2^2\end{aligned}$$

Soft Unitary Weights



RNN

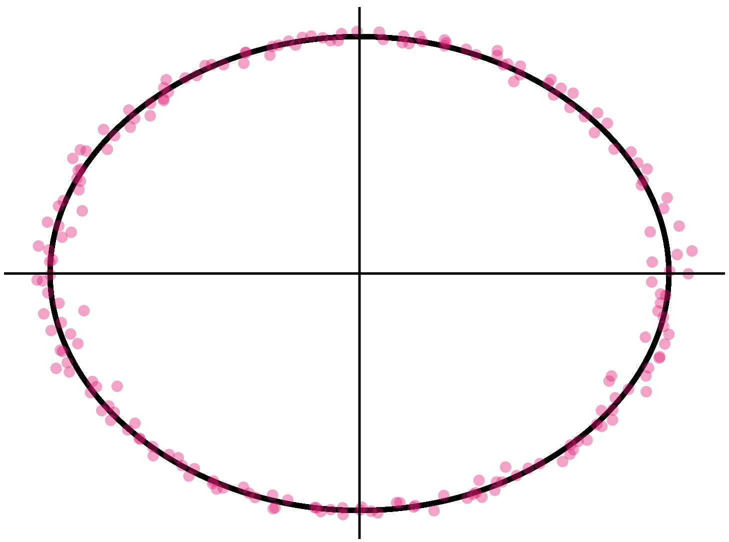


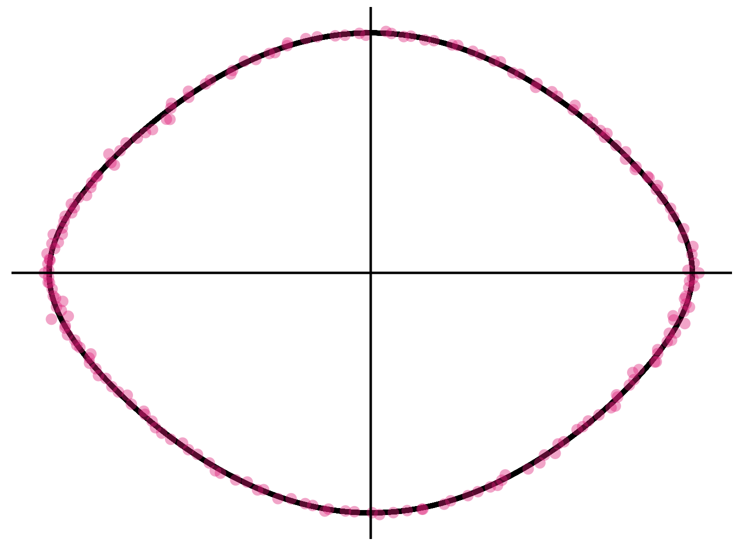
Ours

Results

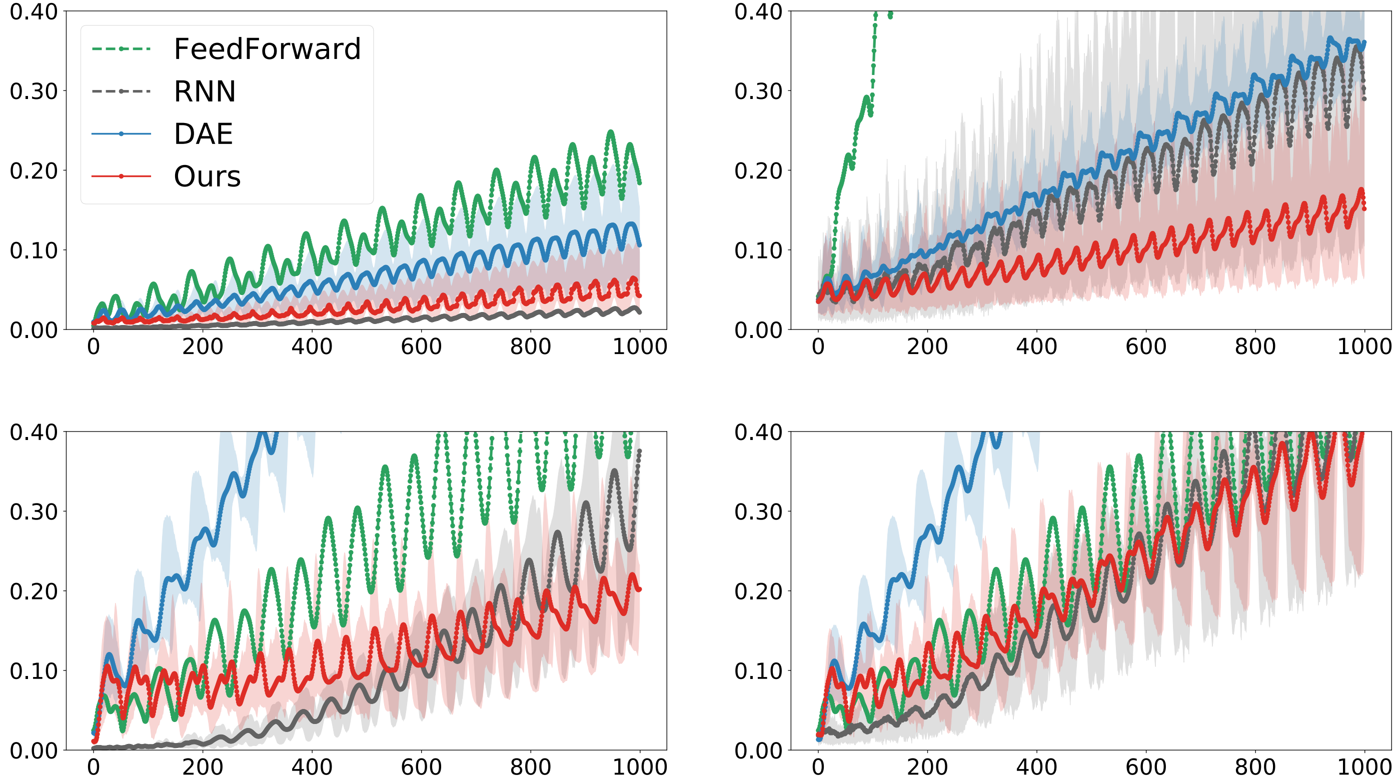
Nonlinear Pendulum

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0, \qquad \theta(0) = \theta_0, \frac{d\theta}{dt}(0) = 0$$





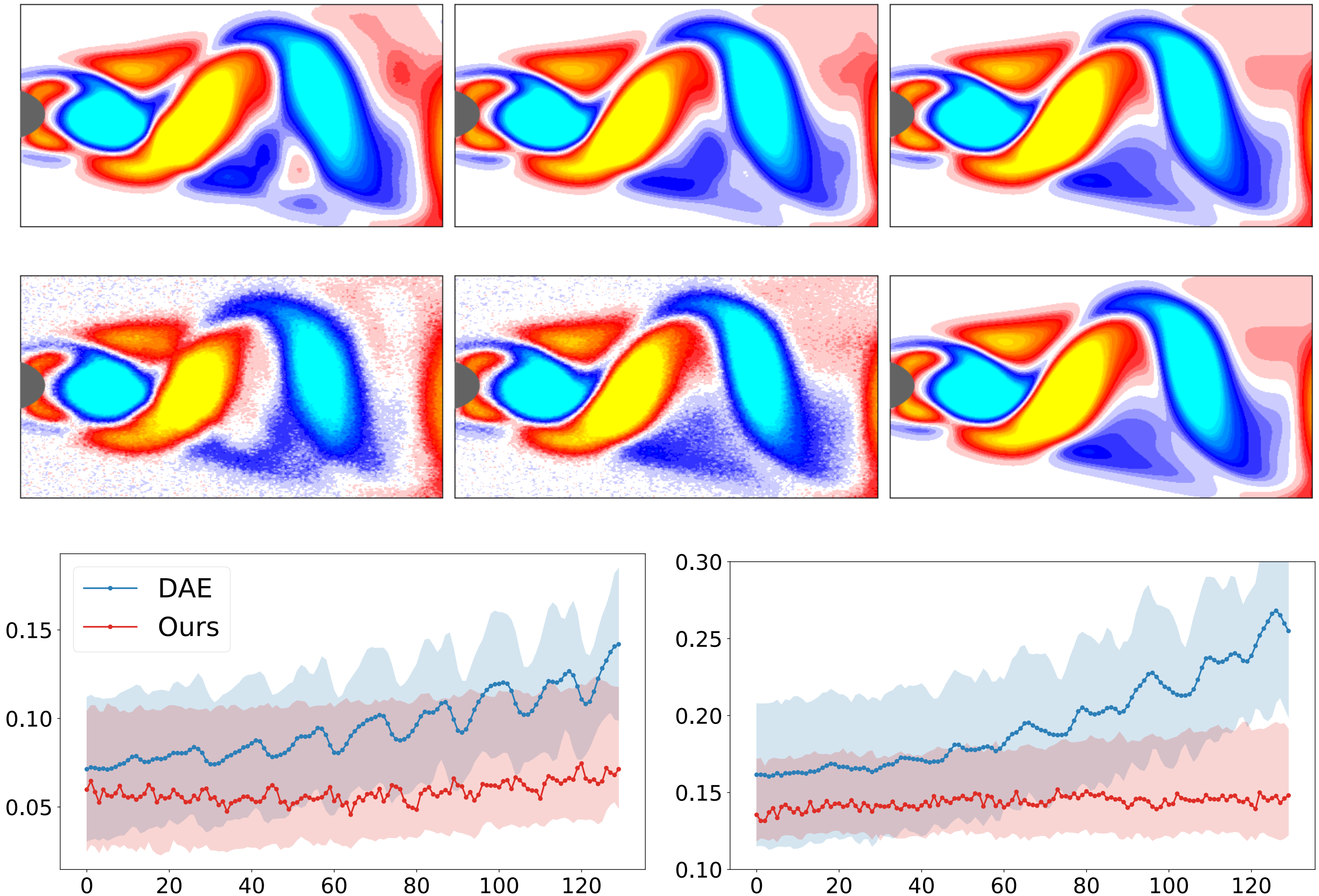
Nonlinear Pendulum



Flow Past a Cylinder

$$\partial_t \omega = -\langle v, \nabla \omega \rangle + \nu \Delta \omega$$

Flow Past a Cylinder



Outline

Summary

- Ideas from dynamical systems theory can help to develop novel algorithmic tools.
- We need to rethink DNNs in order to improve interpretability and explainability.
- Should we expect rigorous mathematical analysis of deep learning? Maybe, but...

We also wish to allow the possibility than an engineer or team of engineers may construct a machine which works, but whose manner of operation cannot be satisfactorily described by its constructors because they have applied a method which is largely experimental – Alan M. Turing