Data-Driven Model-Free Approximation of Koopman Operator for Offline System Identification

Alexander Krolicki

akrolic@clemson.edu

Pierre-Yves Lavertu

plavert@clemson.edu

Overview

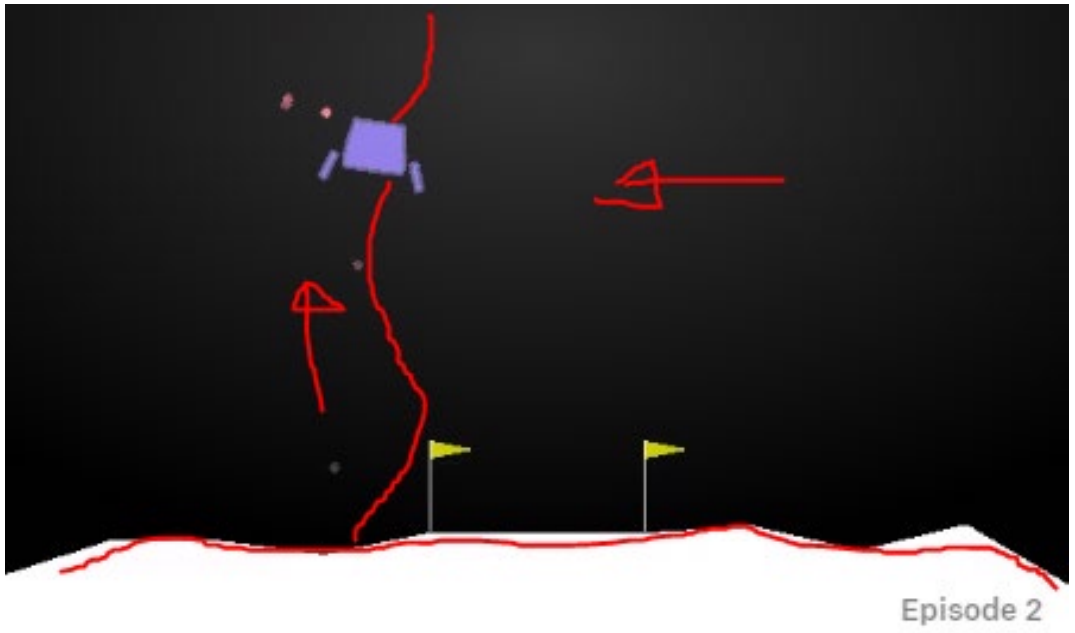
1. Applications
2. Theory
3. Data Collection and Pre-Processing
4. Learning Architecture
5. Results
6. Future Work

Theory: Non-Linear Dynamic System Example (easy example)

<u>Lunar lander:</u>

- Lander coming down
- Disturbances during the way down
- Goal to land in a specified zone

What will be the path based on the initial conditions provided?

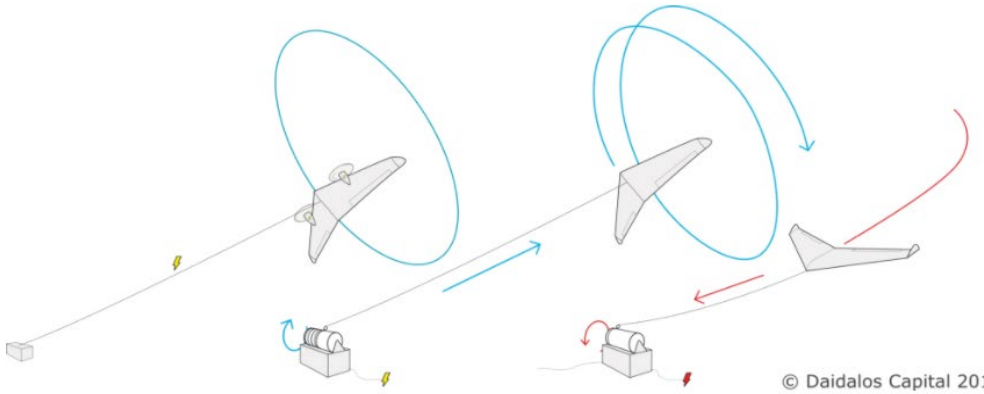


[REF]: https://gym.openai.com/envs/#box2d

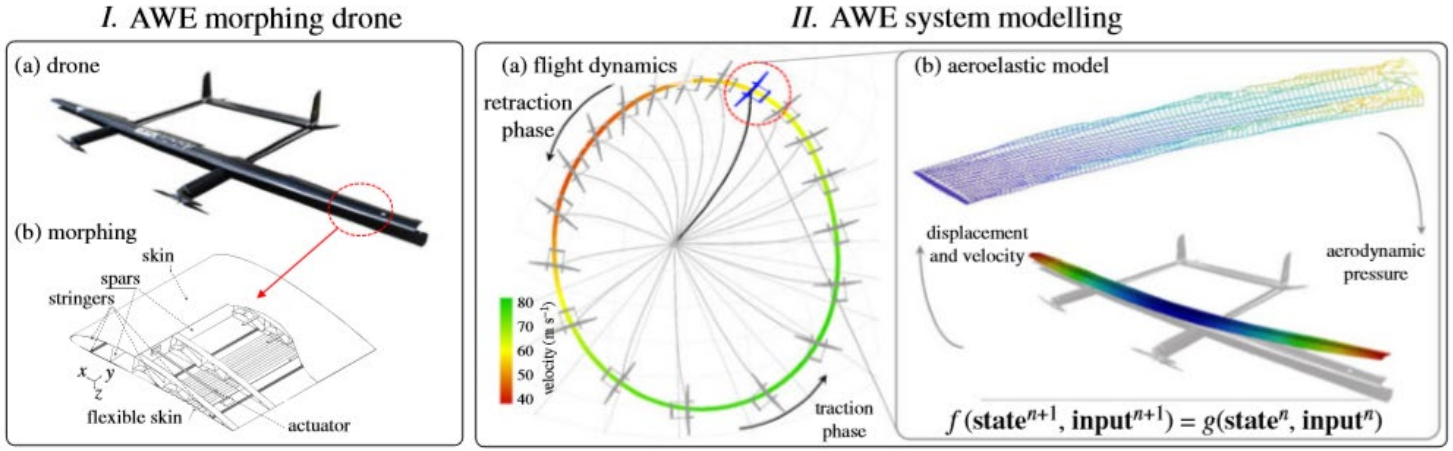
Theory: Existing Applications (complex example)

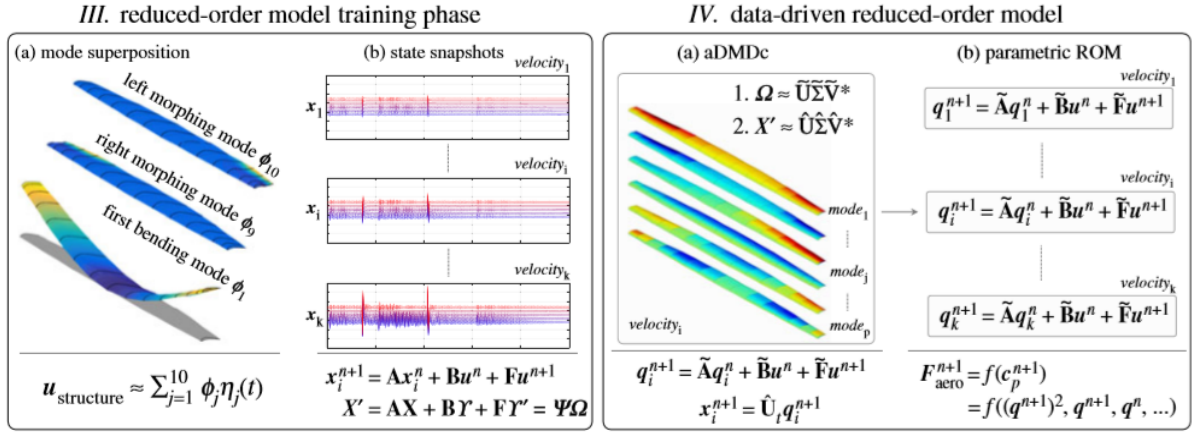
Airborne Wind Energy drones

- **Control of the wing shape is very complex**
- Quantifying all responses of the system based on all the physics involved is a very long endeavor.
- Usage of the Koopman operator theory to simplify the whole problem.



[Ref]: https://airbornewindeurope.org/producing-energy-with-drones/





(Fonzi et al. 2019)

Theory: Basic Concept

- At a high level, the Koopman operator maps the nonlinear dynamics from state space to linear dynamics in the higher-dimensional space of functions (lifted space)

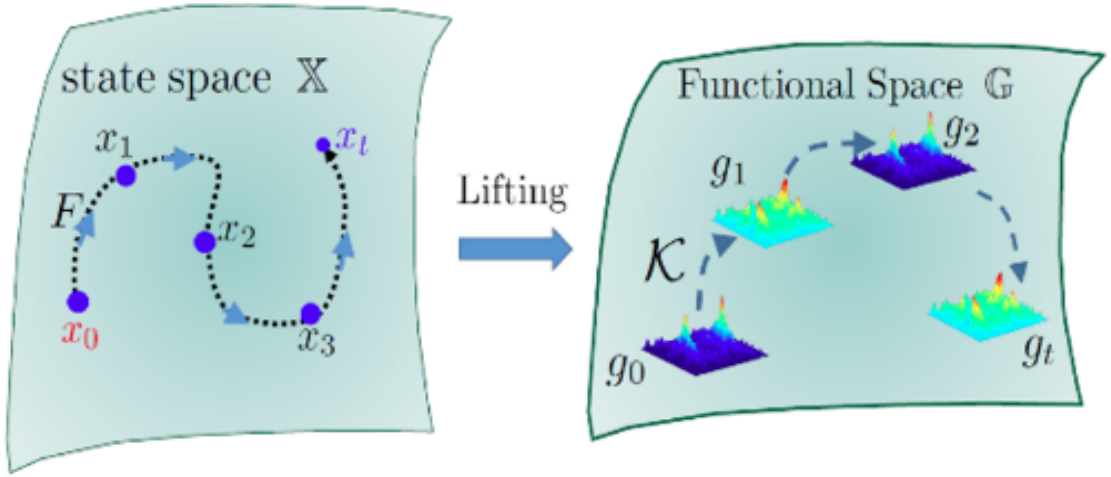


Figure 1: Nonlinear evolution of state in the state space is lifted to linear evolution of functions in the lifted space

[4]

Why Use a Deep Neural Network?

Challenge with current standard approach to approximate the Koopman Operator:

- Eigenfunctions of the Koopman operator can be arbitrarily complex
- Complex function will only be approximately represented in a finite basis
- Generally applied to single Dynamic systems
- Sensitive to noise in the data
- Can be unstable when interpolating between operating regime

Deep learning:

- Well-suited for representing arbitrarily complex functions
- Generalizable to scenarios with a variable number of objects

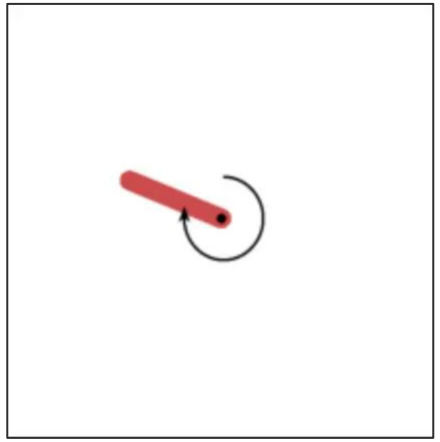
(Brunton et al. 2021)

Theory: Our Contribution

Data pre-processing	**Usage of spectrogram images** ➢ enrich the input dataset
Deep Neural Network Structure	**Learns the basis function instead of approximate them** ➢ Map the states forward in time (from initial non-linear to lifted linear dimension)
Neural Network Output	**Controllable linear state space** ➢ Simplified representation of the original nonlinear system
Error – Loss Measurement	**Find an adequate N-dimension lifted space** ➢ Minimize the predictive error for longer periods of time
Linear system usage	**Design a LQR controller** ➢ Usage of the Linear systems within its accurate prediction range

Data Collection and Pre-Processing

Any Non-Linear Dynamic System



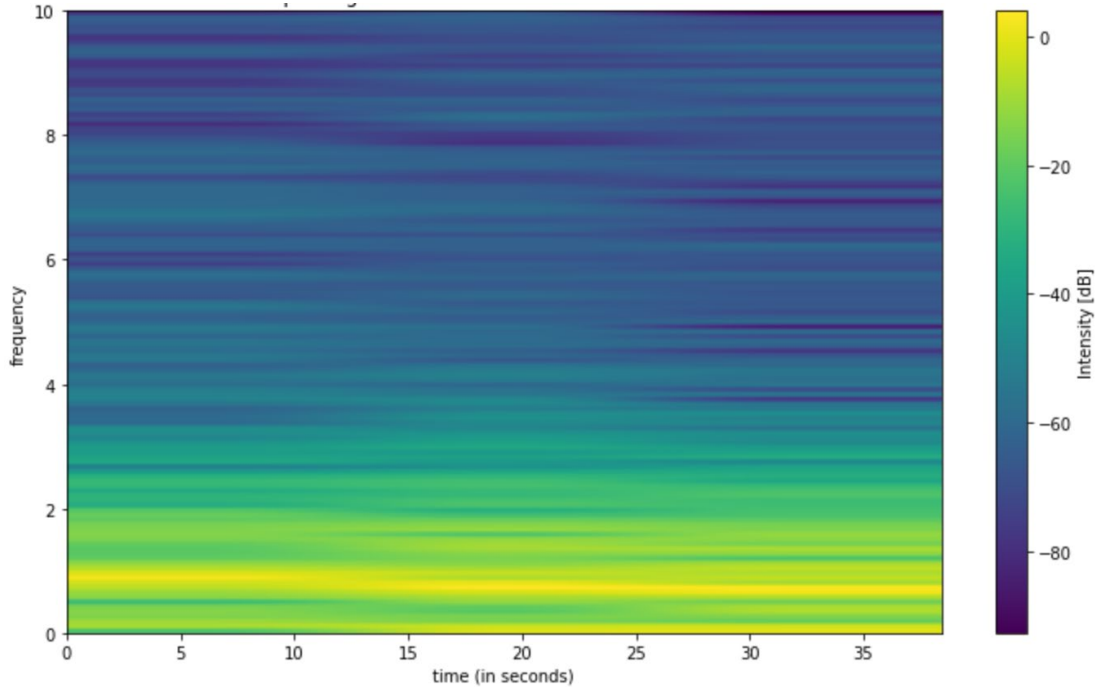
Simple Pendulum Example

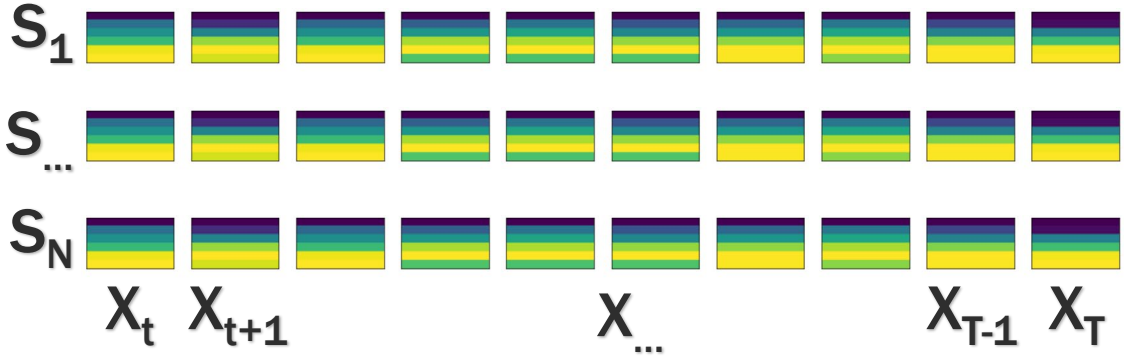
Collect Raw Data from time t to time T

State Data: $S_1 = \theta_{t:T},\ S_2 = \dot{\theta}_{t:T}$

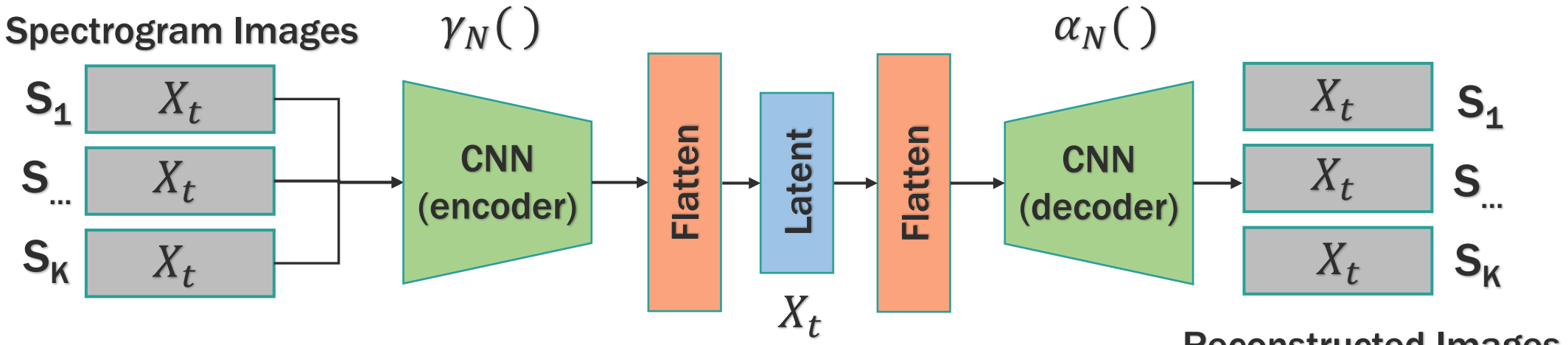
Control Data: $\mathrm{U}_1 = \tau_{t:T}$

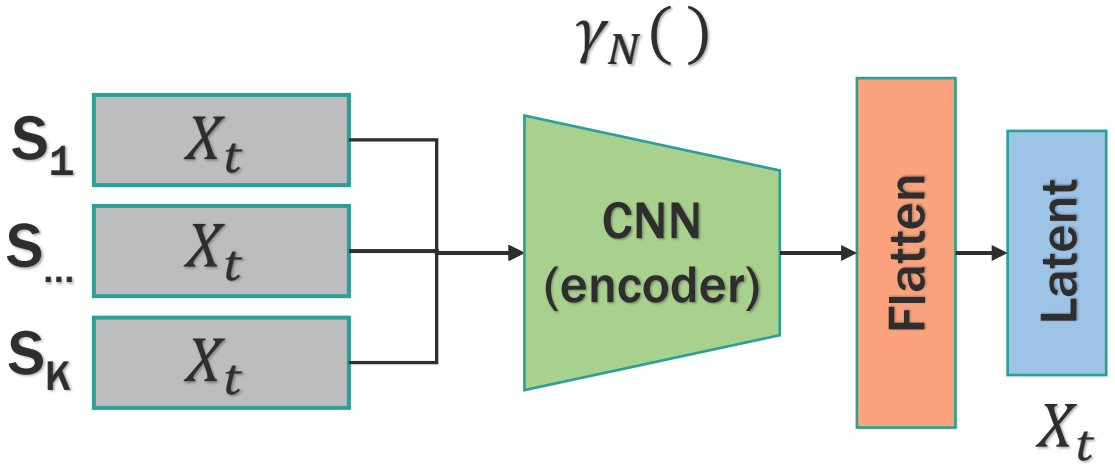
Generate Spectrogram Images





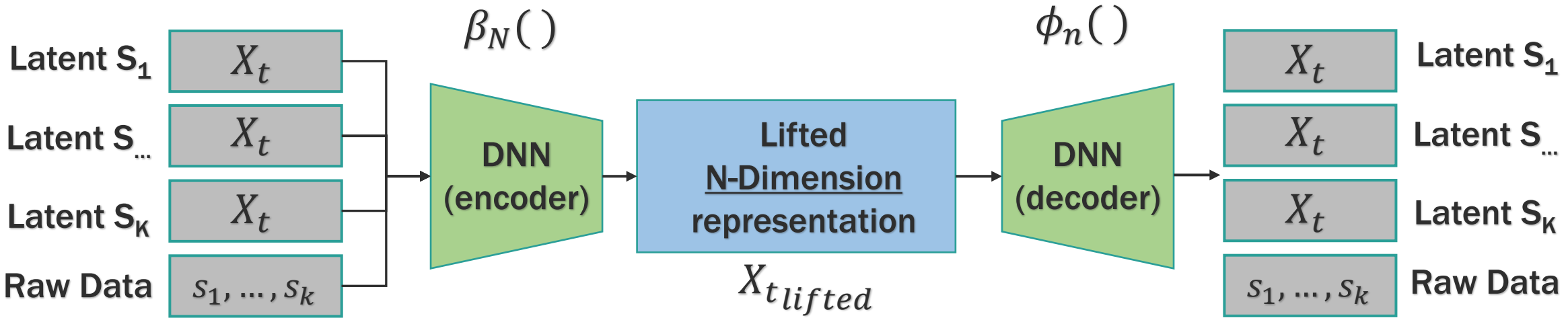
CNN Autoencoder Latent Representation Labeling

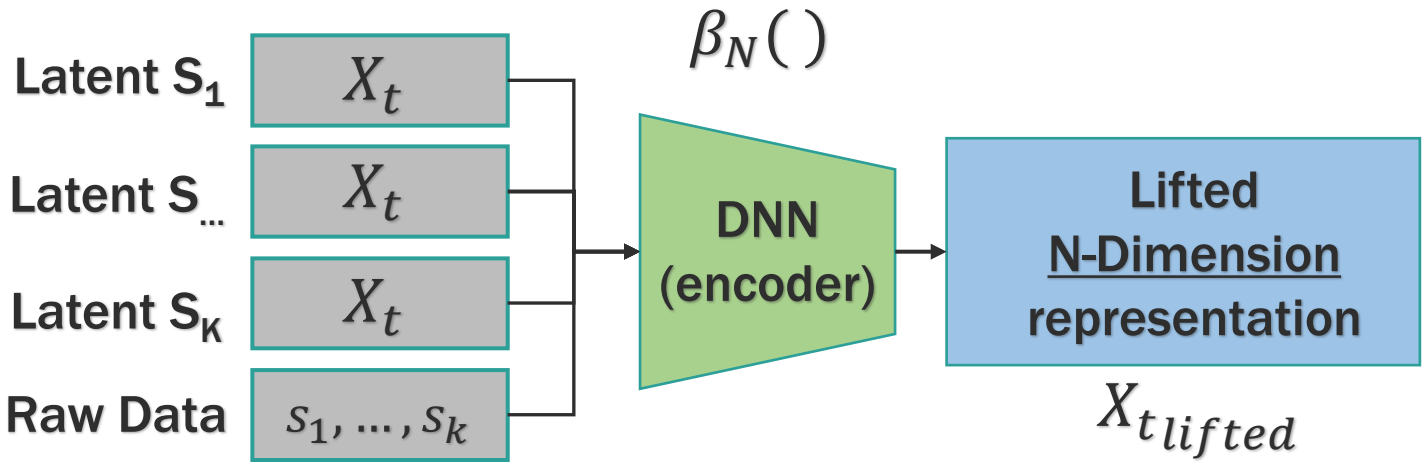




After training, remove decoder and use encoder to generate latent representation of our state data

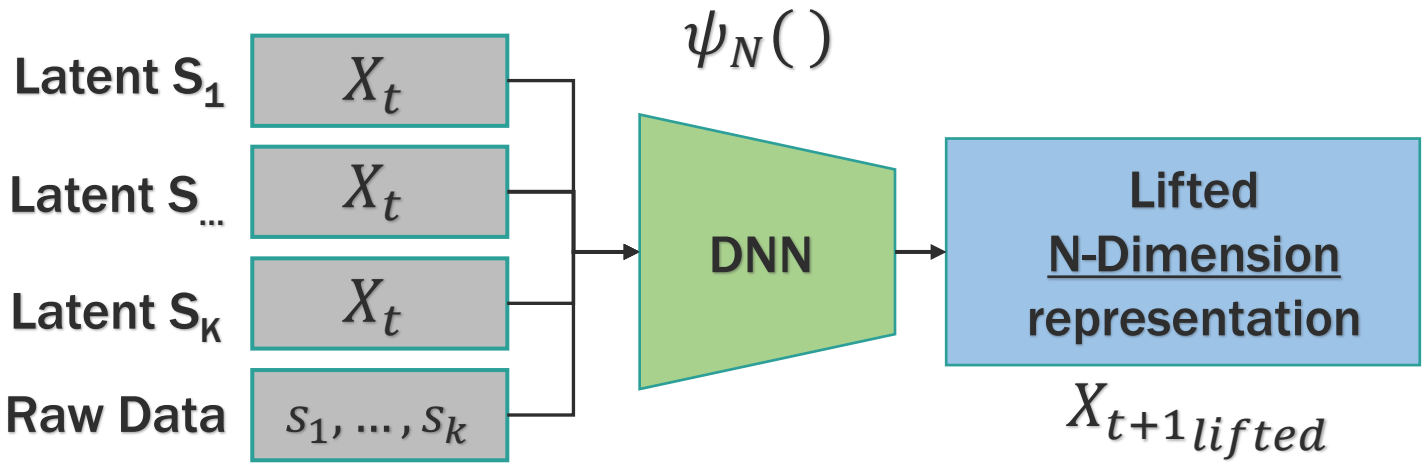
Fully Connected Lifted State Labeling





After training, remove decoder and use encoder to generate lifted state training and test datasets

Lifting DNN



Finally, the lifting basis function $\psi_N( )$ between time steps is learned

$$\psi_N(X_t) = X_{t+1_{lifted}}$$

Model Loss Functions

$$L_1 = X_{t+1_{lifted}} - [AX_{t_{lifted}} + BU_t]$$

$$L_2 = N_{lift} - rank(ctrb(A, B))$$

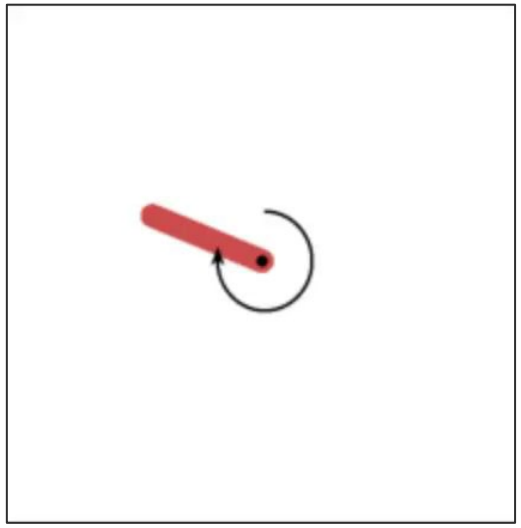
$$L_{Total} = L_1 + L_2$$

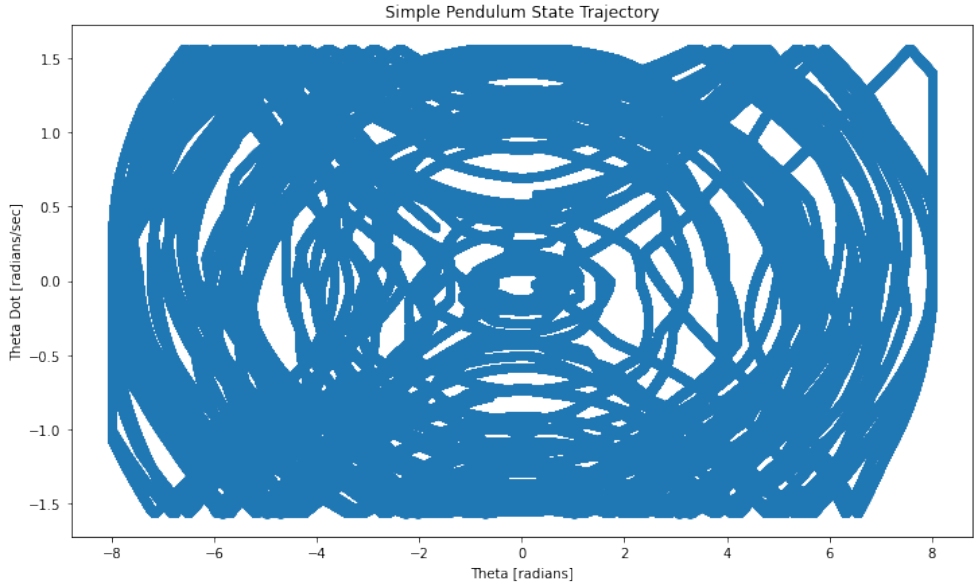
Linear State Space Model

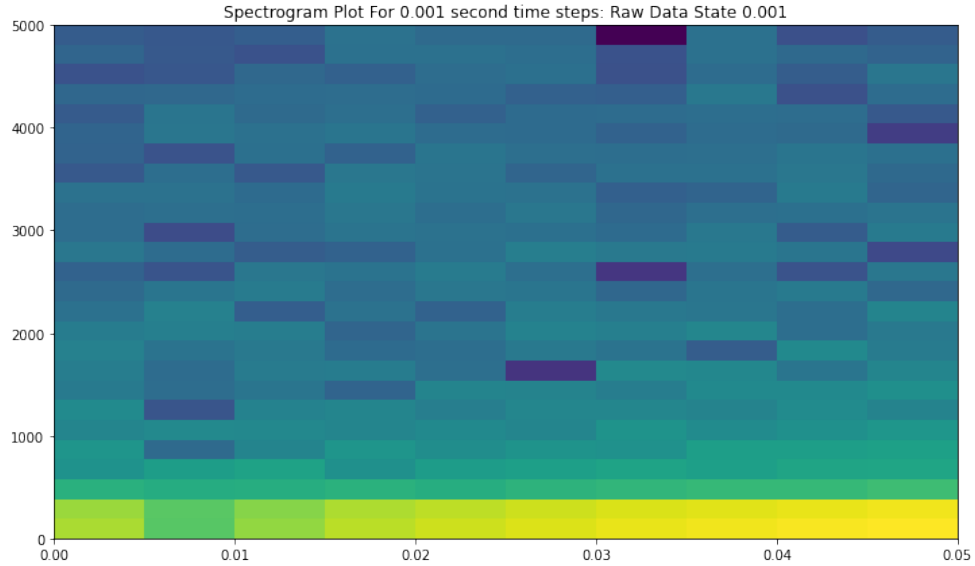
$$X_{t+1_{lifted}} = AX_{t_{lifted}} + BU_t$$

$$Y_t = CX_{t_{lifted}} + DU_t$$

Image + Raw Data Results: Simple Pendulum







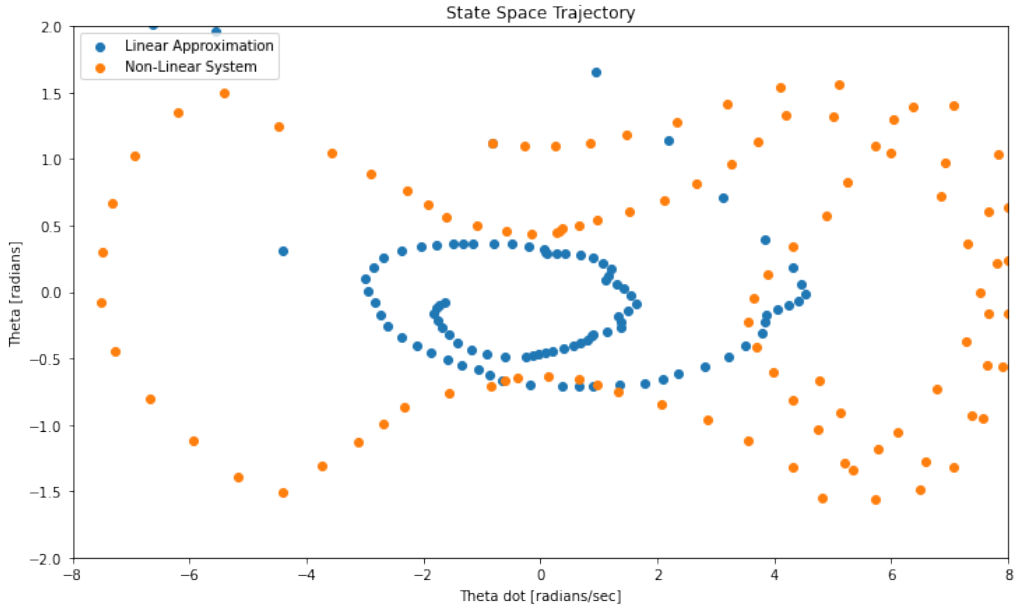
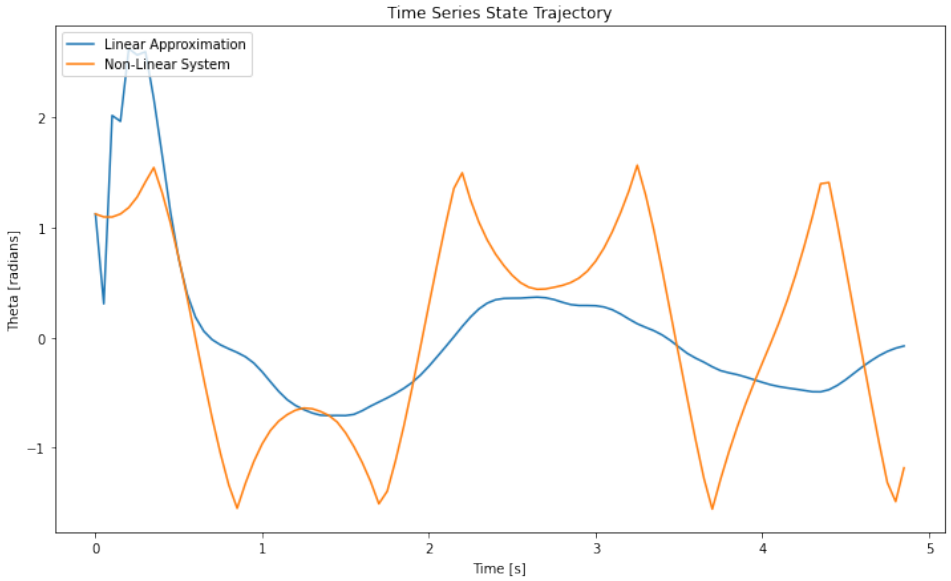
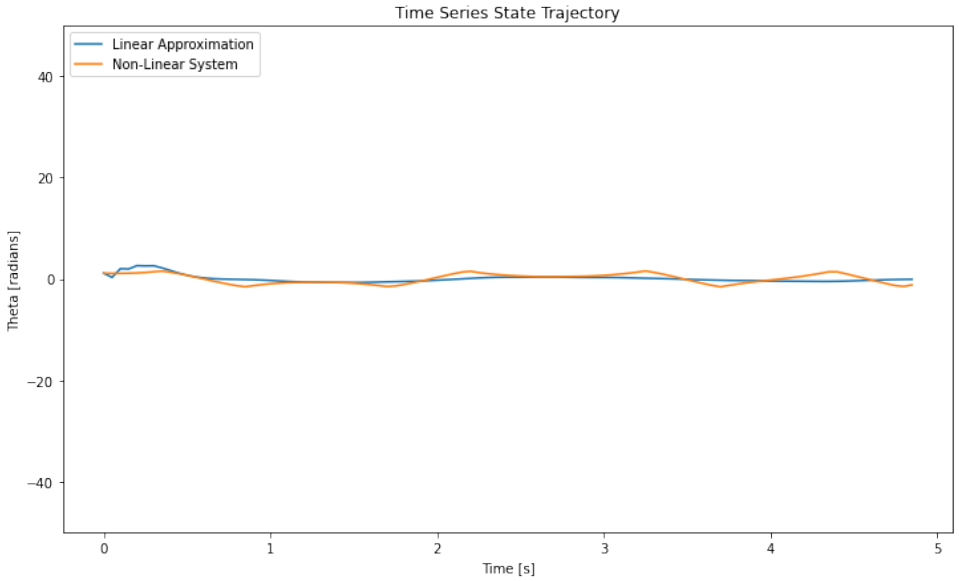
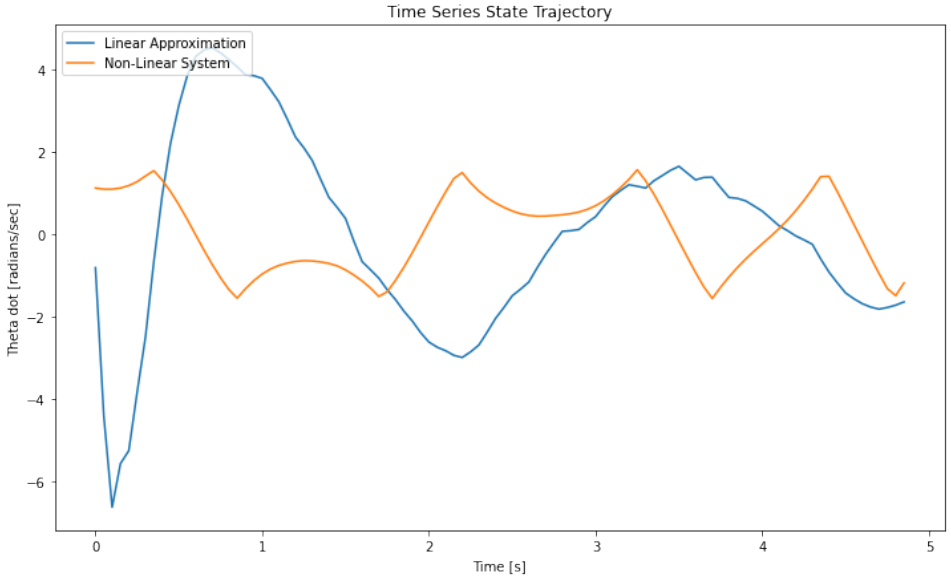
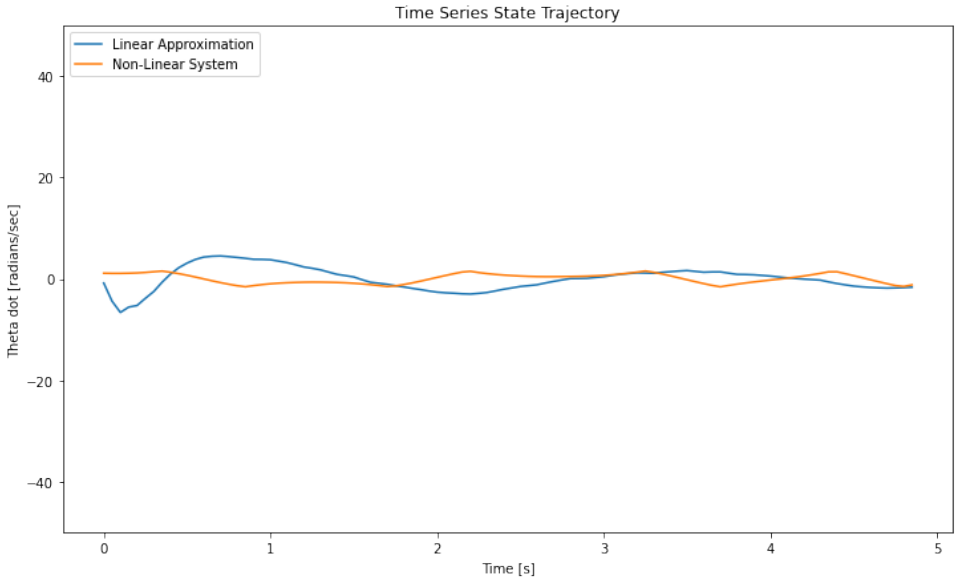


Image + Raw Data Results: Simple Pendulum









Future Work

Refine hyperparameters, Modify Architecture with VAE's

Evaluate methods on higher order non-linear systems

Deploy methods to physical systems

References

1. S. L. Brunton, M. Budiši´c, B. Budišibudiši´ Budiši´c, E. Kaiser, and J. N. Kutz, “MODERN KOOPMAN THEORY FOR DYNAMICAL SYSTEMS *.”
2. N. Fonzi, S. L. Brunton, and U. Fasel, “royalsocietypublishing.org/journal/rspa Research Data-driven nonlinear aeroelastic models of morphing wings for control,” doi: 10.1098/rspa.2020.0079.
3. B. Lusch, J. Nathan Kutz, and S. L. Brunton, “Deep learning for universal linear embeddings of nonlinear dynamics,” doi: 10.1038/s41467-018-07210-0.

Appendix

Theory: Mathematical Overview

Consider a discrete-time, non-linear dynamic system

$$x_{t+1} = F(x_t, u_t)$$

Given a finite set of states x_t and control inputs u_t we have

$$X = [x_1 \quad \dots \quad x_T] \qquad U = [u_1 \quad \dots \quad u_T]$$

We can create a set of labels for x_{t+1} such that $y = x_{t+1}$

$$Y = [y_1 = x_2 \quad \dots \quad y_T = x_T] \qquad X' = [x_1 \quad \dots \quad x_{T-1}]$$

The goal is to learn the linear mapping between states, which in our case is nonlinear and in some cases no equations exist to model the system of interest

Theory: Mathematical Overview

The Koopman operator, $\mathcal{K}$, is an infinite dimensional operator that maps the trajectory of nonlinear finite dimensional states to an infinite dimensional linear state space forward in time.

$$x_{t+1} = F(x_t, u_t)$$

$$\downarrow$$

$$\mathcal{K}g(\mathrm{x_t}) = \mathrm{g}(F(x_t, u_t)) = \mathrm{g}(x_{t+1})$$

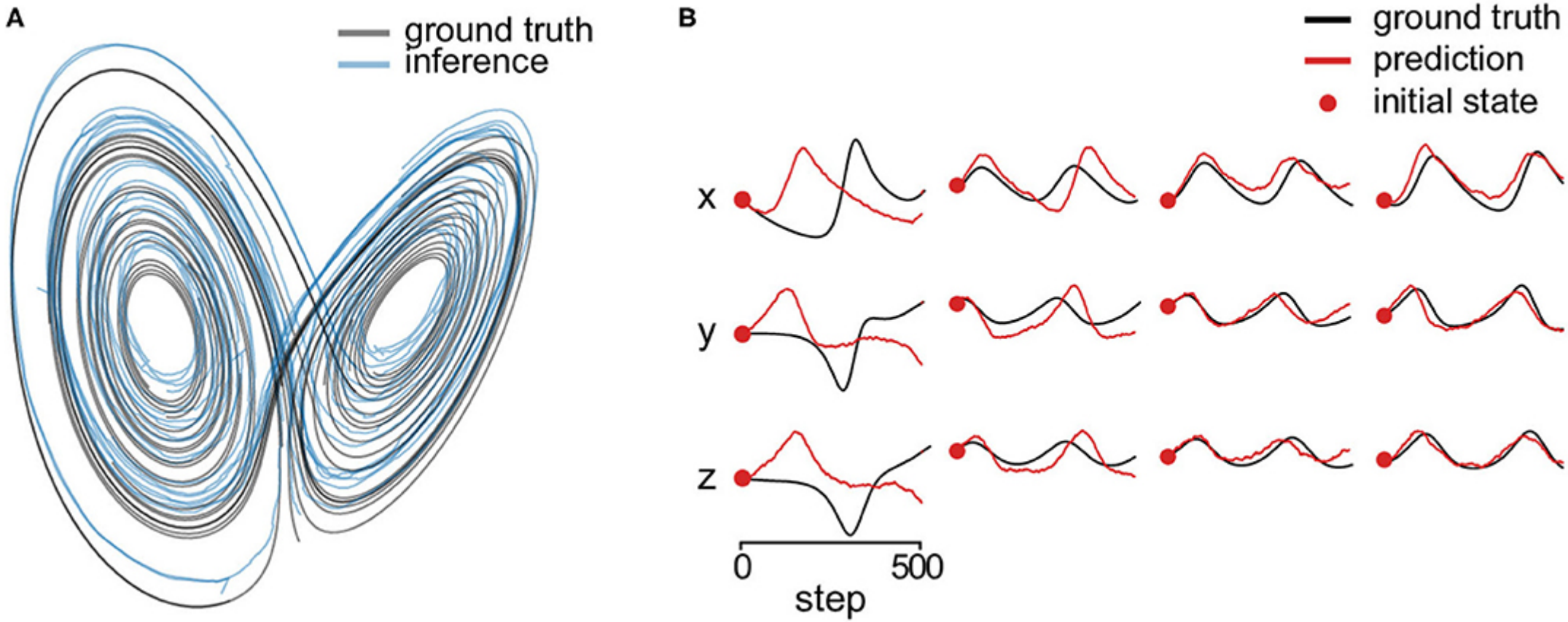
A truncated approximation can be found through the choice of a lifting basis function

$$\psi: \mathbb{R}^n \rightarrow \mathbb{R}^{\infty \approx N} \quad N \gg n$$

Thus, we can represent the nonlinear system as a lifted linear system

$$\psi(x_{t+1}) = A\psi(x_t) + Bu_t$$

Extremely Non-Linear - Lorenz System



Linear Model Criteria and Loss Functions

1. Use the the X_{t+1} outputs from DNN – lifted representations – to compute A, B state space matrices.

$$[A, B] = \left[X_{t+1_{lift}} \begin{bmatrix} X_{t_{lift}} \\ U_t \end{bmatrix}^T \right] \left[\begin{bmatrix} X_{t_{lift}} \\ U_t \end{bmatrix} \begin{bmatrix} X_{t_{lift}} \\ U_t \end{bmatrix}^T \right]^{-1}$$

2. Evaluate accuracy of prediction as loss function L_1

$$L_1 = X_{t+1_{lift}} - [AX_{t_{lifted}} + BU_t]$$

3. Evaluate lifted systems controllability as loss function L_2

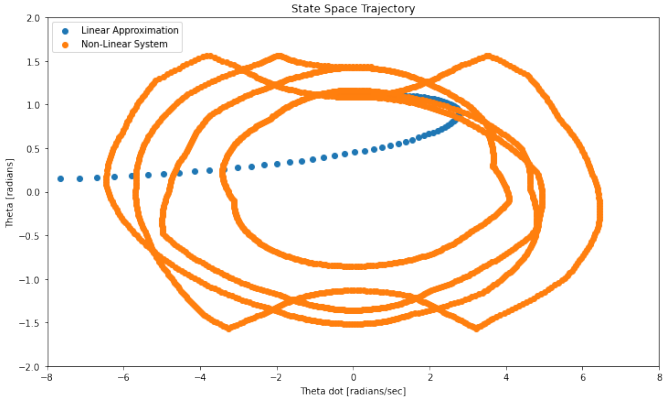
$$L_2 = N_{lift} - rank(ctrb(A, B))$$

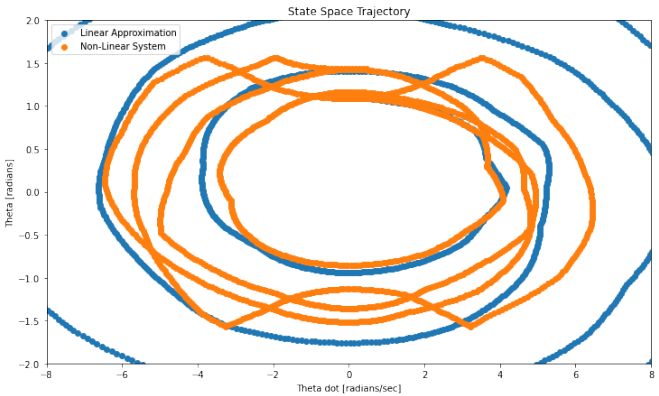
4. Loop training until $L_1 + L_2$ is minimized then compute C, D

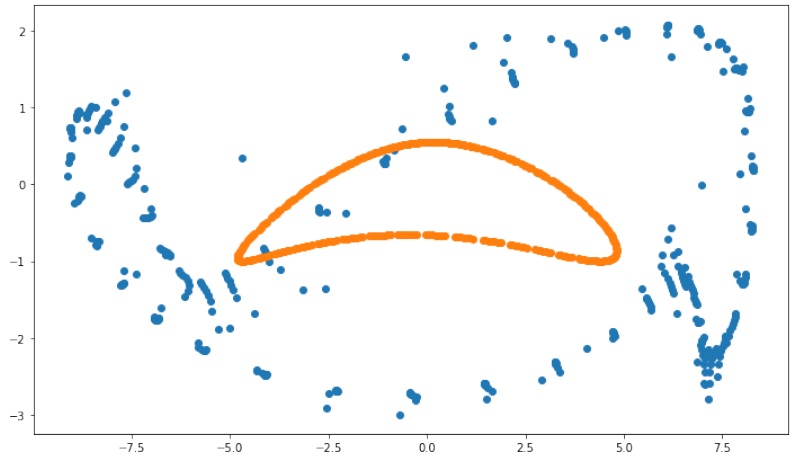
$$C = \mathrm{X_t} \left[X_{t_{lift}} \right]^T \qquad D = 0$$

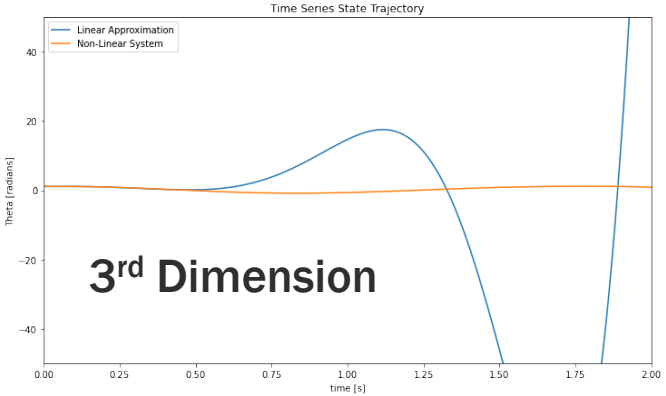
- Utilizing spectrogram images to enrich the input dataset (pre-processing)
- Learning the basis function that maps the states forward in time from the original nonlinear dimension to a lifted linear dimension (NN structure – Functioning)
- Generating a controllable linear state space representation of the original nonlinear system from raw state and control data. (initial - NN ouput)
- Minimizing the predictive error in the learned linear model for longer periods of time (Error measurement – Quality measurement)
- Demonstrating the ease of designing an optimal controller using linear system methods such as linear quadratic regulator methods.

Raw Data Only Results: Simple Pendulum

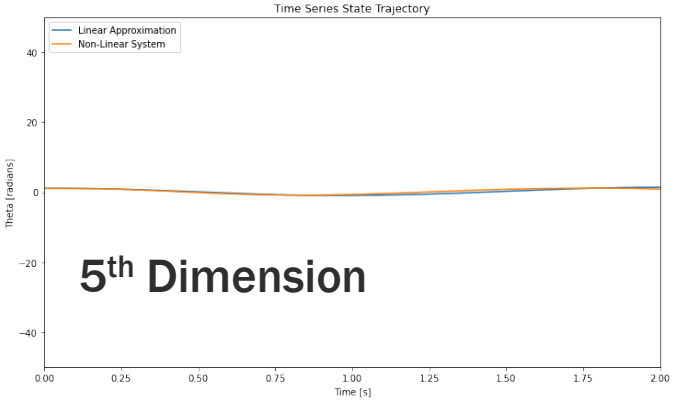








3rd Dimension



5th Dimension