Method Development for Efficient Training of Reduced Order Models

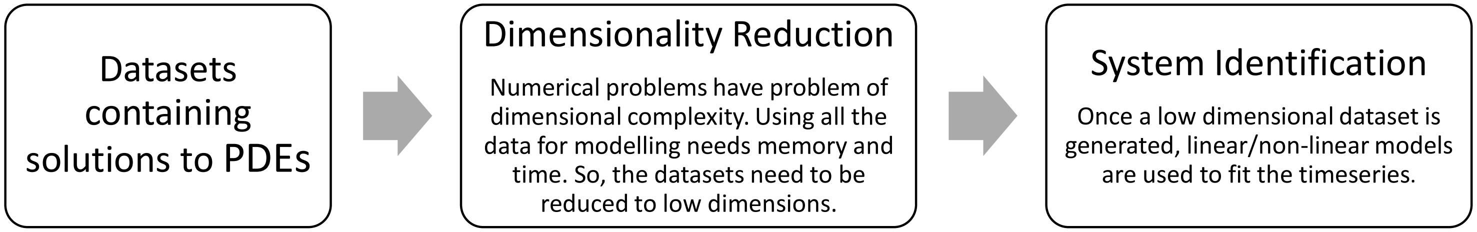
Khushant Khurana
Advised by William Anderson, Youngsoo Choi & Charles Stanley Wojnar
Strategic Deterrence
Lawrence Livermore National Laboratory

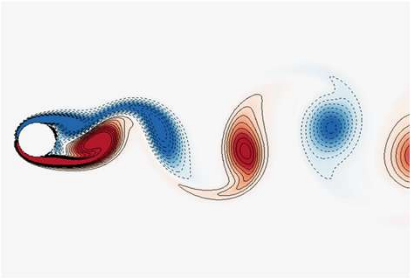
DSTI Exit Talk

August 8, 2025

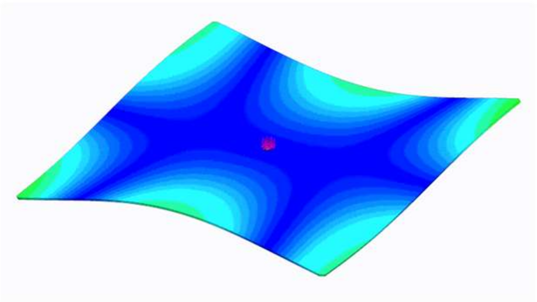
Motivation

- Numerically solving time-dependent partial differential equations (PDEs) can be challenging and computationally expensive. This has prompted the development of reduced order models (ROMs) for providing fast and accurate approximate solutions.





Air flow simulation from Shantanu Bailoor [1]

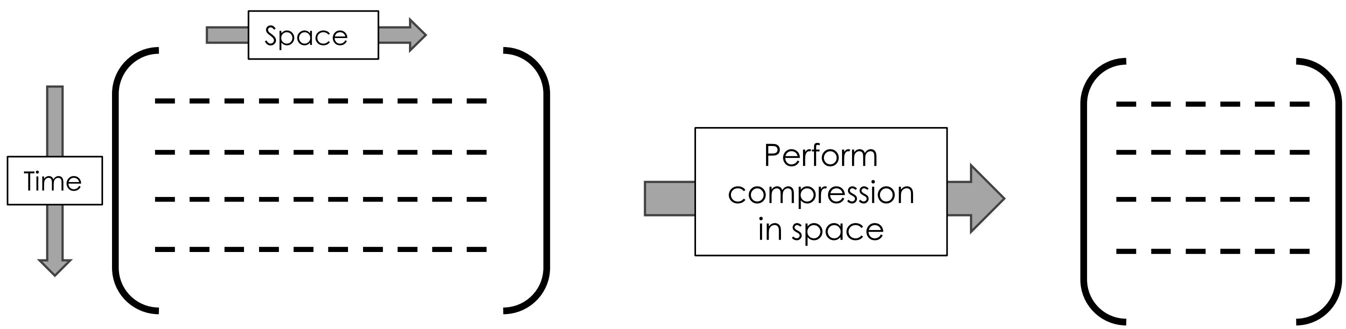


FEA simulation from Sentek Dynamics [2]

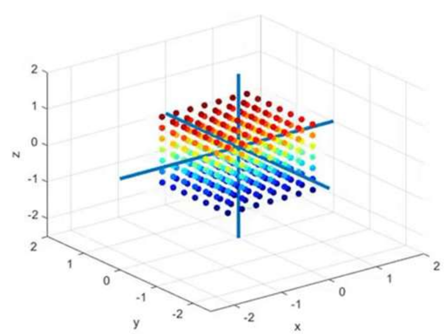
[1] https://www.shantanubailoor.com/carreau-cylinder
[2] https://www.sentekdynamics.com/finite-element-analysis-animations

Dimensionality Reduction

- Consider the following matrix transformation:



- Methods to perform compression: Singular Value Decomposition (SVD - linear) & machine learning (non-linear). While SVD does so using stretch, squeeze, and rotate, machine learning does so through minimizing a loss function.



Example of data transformation using SVD [3]

[3] https://angeloyeo.github.io/2019/08/01/SVD_en.html

SINDy - Sparse Identification of Non-Linear Dynamics

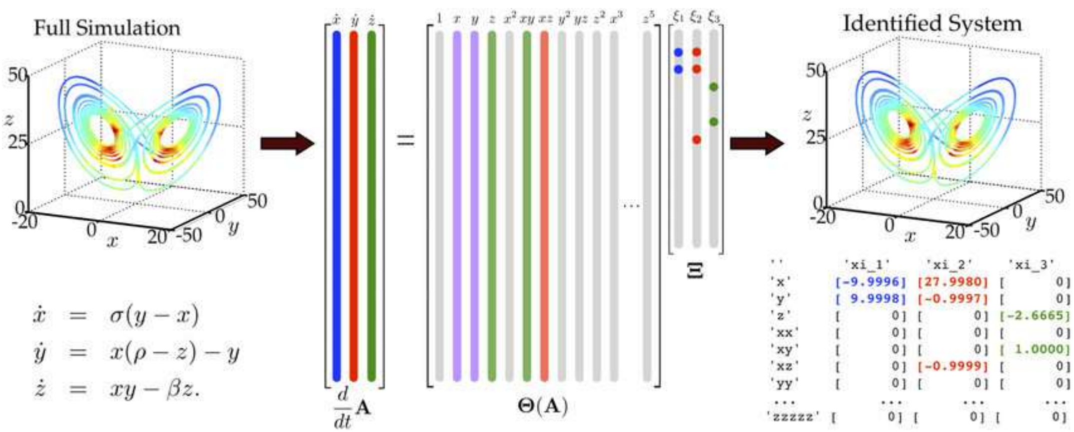
- SINDy is a data-driven algorithm for obtaining ordinary differential equations from timeseries data. Consider a system with dynamics:

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

- SINDy tries to estimate the ODE as a linear combination of candidate sets (could be linear or non-linear):

$$\underbrace{\dot{x}}_{\text{Estimated Derivative}} = \underbrace{\theta(x)}_{\text{Candidate Set}} \underbrace{\xi_{\mathbf{k}}}_{\text{Coefficients}}$$

where the coefficients are determined using a L2 norm: $\xi_{\mathbf{k}} = \arg\min_{\xi'_{\mathbf{k}}} \left\| \dot{\mathbf{X}}_{\mathbf{k}} - \mathbf{\Theta}(\mathbf{X})\xi'_{\mathbf{k}} \right\|_{2}$

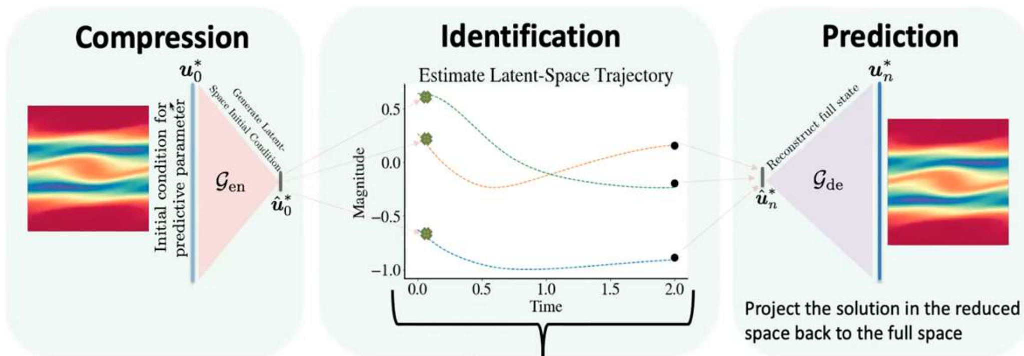


SINDy Computation Example from Brunton [4]

[4] Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Discovering governing equations from data: Sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences, 113(15), 3932-3937

LaSDI – Latent Space Dynamics Identification

- LaSDI [5] algorithm combines dimensionality reduction (using ML) and system identification to map full order solutions to latent space using autoencoders.



$$\frac{d\hat{\boldsymbol{u}}}{dt} = \hat{\boldsymbol{A}}(\boldsymbol{\mu})\Theta(\hat{\boldsymbol{u}})$$

Performed using ML:
- Multi-Layered Perceptron
- Convolution
- Transformers etc.

→ Learn time evolutionary dynamics → Decode low dimensional data back to full space using ML techniques.

- To train the model, the loss function is evaluated as:

$$\mathcal{L}(\omega_{\text{enc}}, \omega_{\text{dec}}, \theta) = \underbrace{\mathcal{L}_{\text{AE}}(\omega_{\text{enc}}, \omega_{\text{dec}})}_{\text{Compression Loss}} + \underbrace{\varepsilon_1 \mathcal{L}_{\text{DI}}(\theta)}_{\text{Dynamics Identification Loss}} + \underbrace{\varepsilon_2 \|\theta\|_2^2}_{\text{Penalty Term}}$$

[5] Kim, H., Lee, K., & Choi, Y. (2023). LaSDI: Parametric Latent Space Dynamics Identification. arXiv preprint arXiv:2301.00816.

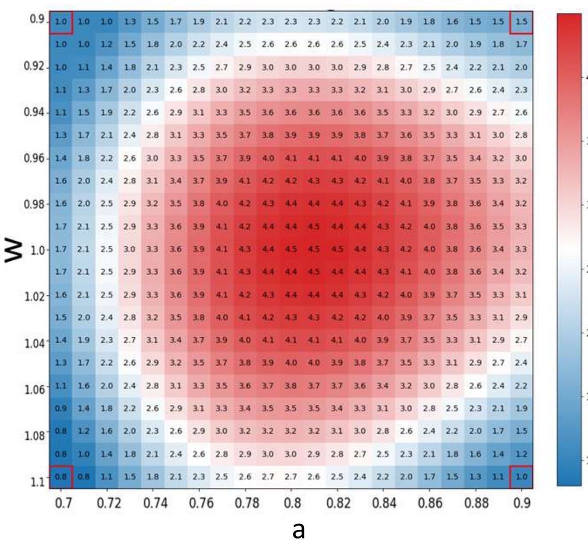
Introducing Higher Order ODEs in SINDy

- Integrated the ability to use higher order polynomials and non-linear functions in SINDy's candidate set and tested the updates on 1D Burgers.

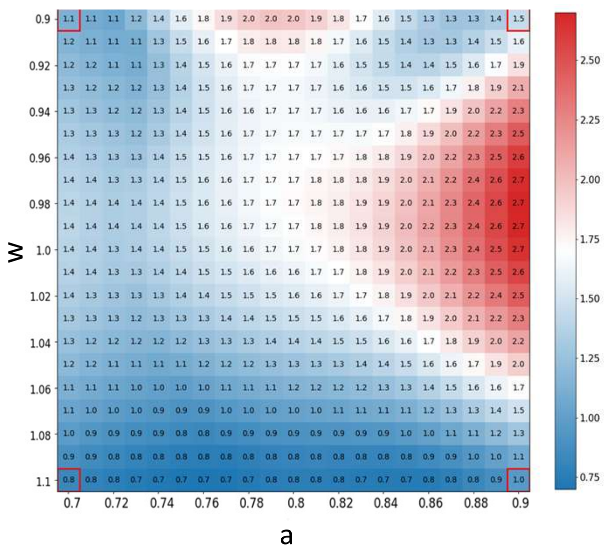
$$\begin{cases} \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \qquad u(t=0,x) = a\exp\left(-\frac{x^2}{2w^2}\right) \\ u(t, x=3) = u(t, x=-3) \end{cases}$$

- Autoencoder structure used: 1001 => 500 => 5 => 500 => 1001. The models are trained for 5000 iterations.

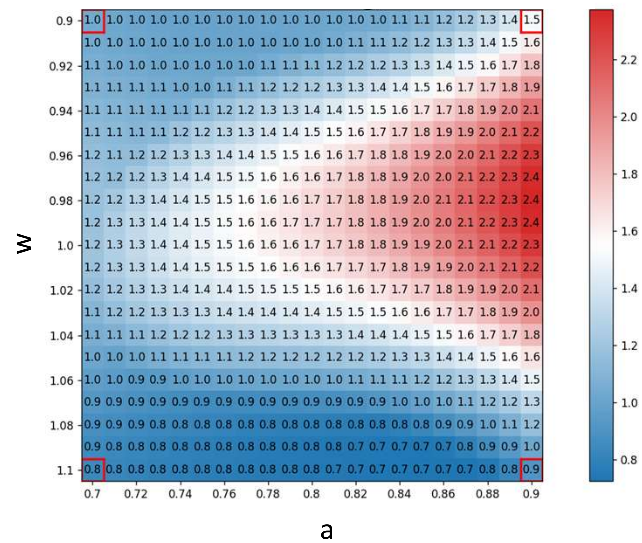
Relative Error (%) using Linear Terms



Relative Error (%) using Trig functions + linear terms



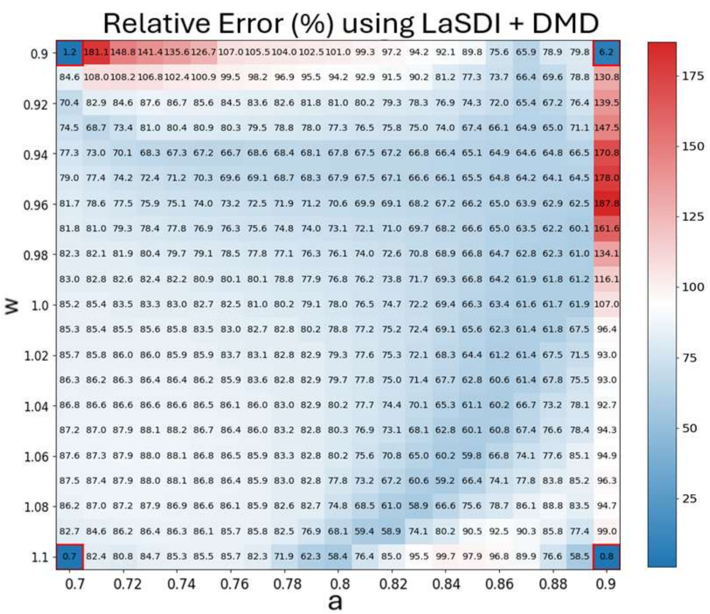
Relative Error (%) using Exponentials + linear terms



Using Dynamic Mode Decomposition (DMD) in Latent Space

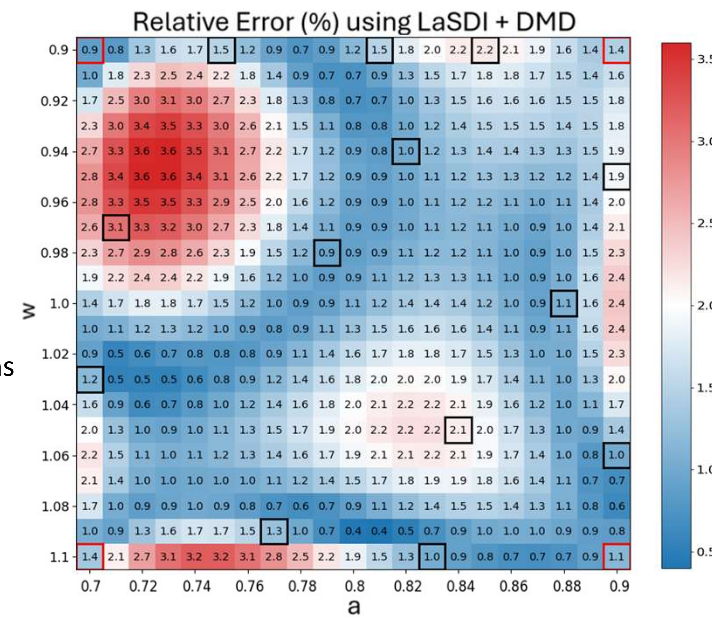
- DMD finds a linear mapping, called 'A', to propagate the system by one time step.

$$\begin{bmatrix} | & | & & | \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix} = \mathbf{A} \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix}$$



5000 Iterations

28000 Iterations + Greedy Sampling

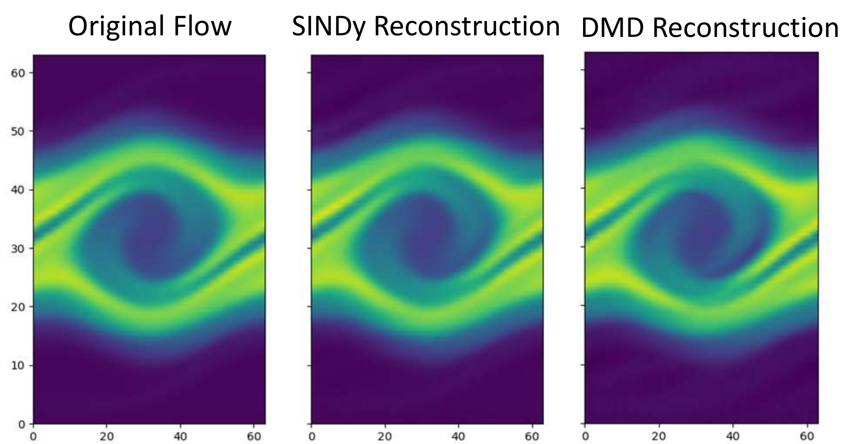


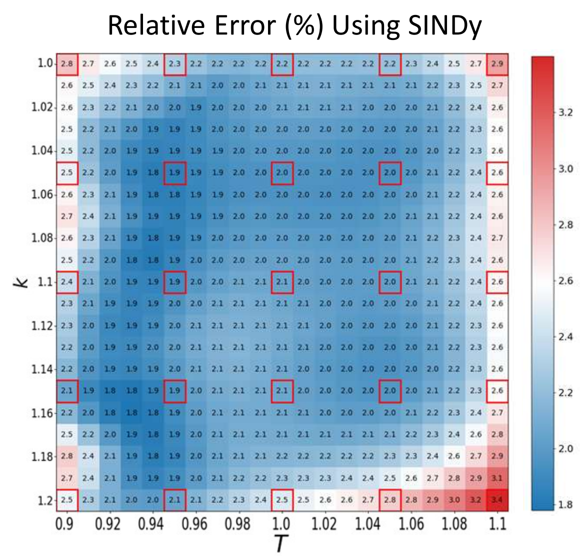
DMD Results for Two Stream Plasma Instability

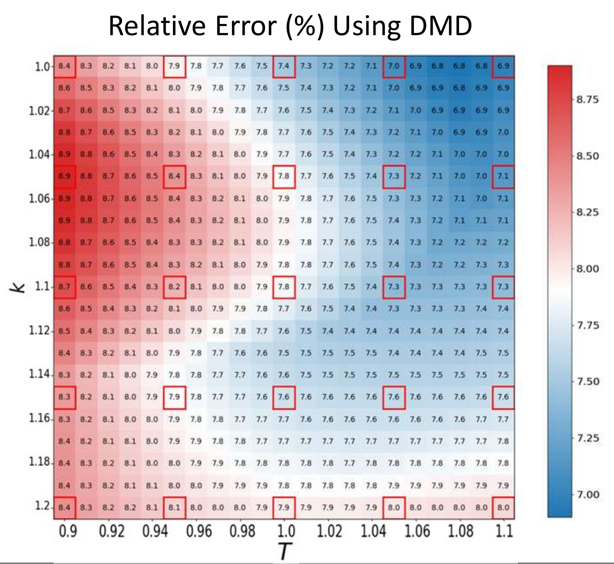
- The two stream plasma instability problem is given as:

$$\begin{cases} \dfrac{\partial f}{\partial t} + \dfrac{\partial}{\partial x}(vf) + \dfrac{\partial}{\partial v}\left(\dfrac{d\phi}{dx} f\right) = 0, \quad \dfrac{d^2\phi}{dx^2} = \displaystyle\int_v f\,dv \\ f(t=0,x,v) = \dfrac{4}{\pi T}\left[1 + \dfrac{1}{10}\cos(k\pi x)\right]\left[\exp\left(-\dfrac{(v-2)^2}{2T}\right) + \exp\left(-\dfrac{(v+2)^2}{2T}\right)\right] \end{cases}$$

- Autoencoder structure used here: 4096 => 550 => 5 => 550 => 4096. The model is trained for 28e3 iterations.



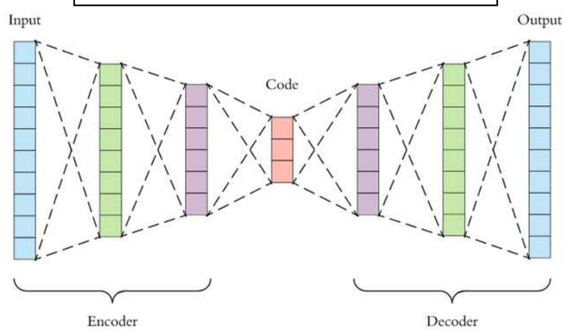




Using Fourier Neural Operator (FNO) for Compression

- We wanted to try a different compression method that makes the training faster while maintaining/increasing the accuracy.
- FNOs are used for mapping function spaces (inputs) to function spaces (solutions)

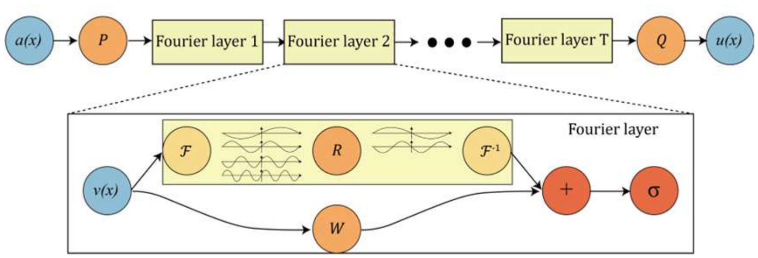
LaSDI + SINDy Structure



Autoencoder architecture [6]

- Uses **linear layers** for dimensionality reduction
- SINDy is applied in **time domain**
- The latent space is highly non-physical

FNO + SINDy Structure



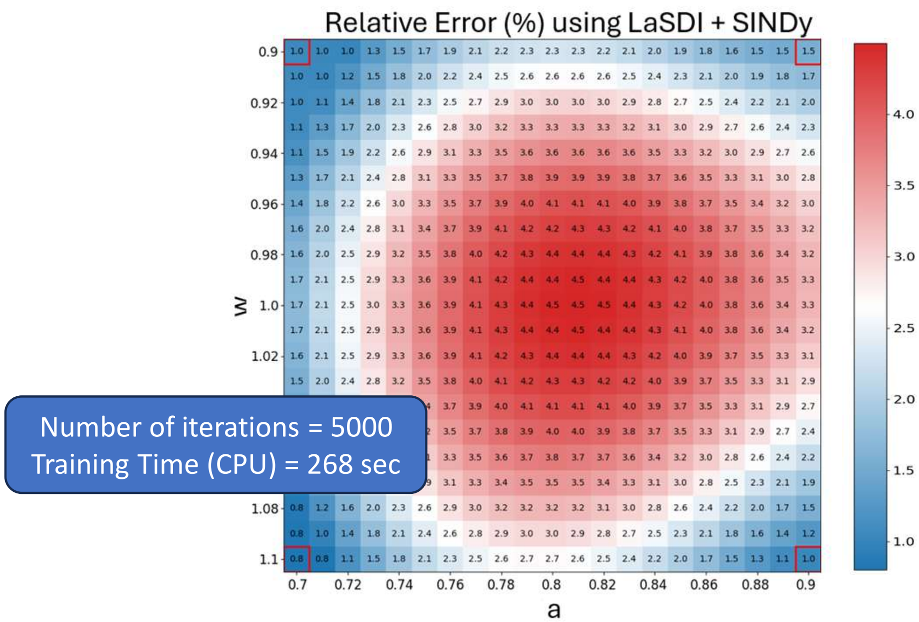
Fourier Neural Operator architecture [7]

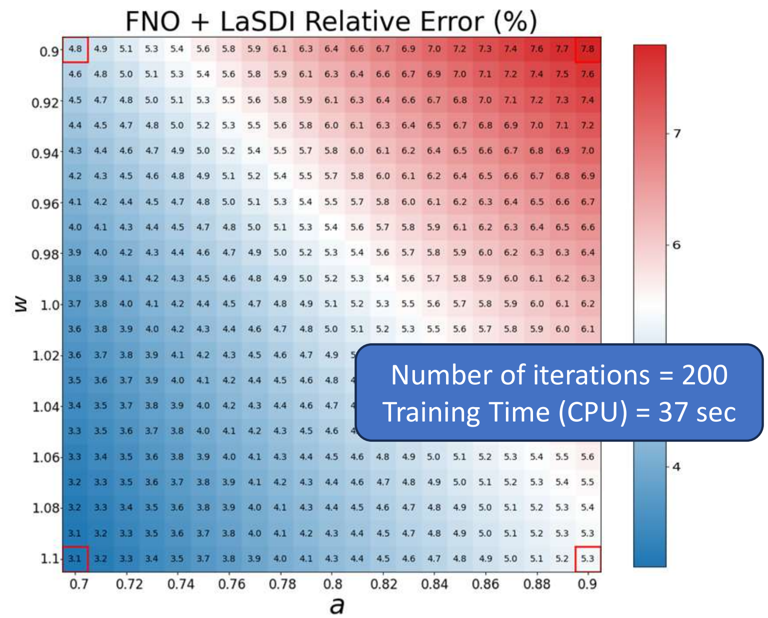
- Truncates **Fourier modes** for dimensionality reduction
- SINDy is applied in **frequency domain**.
- The latent space contains physical modes

[6] https://medium.com/data-science/applied-deep-learning-part-3-autoencoders-1c083af4d798
[7] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2021). Fourier Neural Operator for Parametric Partial Differential Equations. International Conference on Learning Representations (ICLR)

FNO + SINDy v/s LaSDI + SINDy

- Evaluated both techniques on 1D Burgers while using SINDy in latent space.
- LaSDI + SINDy structure: 1001 => 500 => 5 => 500 => 1001. Trainable parameters: 1e6
- Number of Modes kept for FNO => 9. Trainable parameters: 2.7e4





Conclusion & Future Work

- To wrap things up:
  - Integrated capabilities to use higher order terms and non-linear functions in latent space.
  - Using DMD in latent space requires more training samples and iterations.
  - FNO + SINDy works faster that LaSDI + SINDy on 1D Burgers.
- Currently, we are trying to work on implementing FNO + SINDy on the two-stream plasma instability problem.
- We are also working on different types of compression architectures and hyperparameter tuning to squeeze more accuracy out of the system.
- Also currently working on getting these simulations on Lassen and use GPUs.
- Future work would require more rigorous testing of the updated architectures/algorithms on different problems.