A study on quantifying effective training of DLDMD

Joseph A.G. Diaz

Master of Science in Applied Mathematics
with a Concentration in Dynamical Systems,
San Diego State University

August 1, 2022

Introduction

In the study of dynamical systems a central problem is how to derive models from measured data to facilitate the prediction of future states. The data-driven method Dynamic Mode Decomposition and it's extensions offer a compelling avenue in the problem of prediction from time-series data.

The marriage of these methods with Machine Learning and Neural Networks allows for the leveraging the power of these tools in the space.

If standard metrics of model training are unavailable, what other tools can we use to analyze the performance of a Machine Learning algorithm?

Introduction - Koopmanism

We seek a predictive model for a time series $\{\boldsymbol{y}_j\}_{j=1}^{N^T+1}$, which are the measurements of a dynamical system of the form

$$\frac{d\boldsymbol{y}}{dt} = f\left(\boldsymbol{y}(t)\right), \quad \boldsymbol{y}(0) = \boldsymbol{x} \in \mathcal{M} \subseteq \mathbb{R}^{N_s} \tag{1}$$

Denote $\varphi(t;\boldsymbol{x}) = \boldsymbol{y}(t)$ to be the flow map from $\boldsymbol{x}$ and $g : \mathcal{M} \rightarrow \mathbb{C}$, be a square integrable observable then, by [4], $\exists\mathcal{K}$ such that

$$\mathcal{K}^t g(\boldsymbol{x}) = g(\varphi(t;\boldsymbol{x})), \tag{2}$$

Finding the eigen-values $\{\lambda_\ell\}$ and eigen-functions $\{\phi_\ell\}$ of $\mathcal{K}$ yields

$$\mathcal{K}^t \phi_\ell = \exp(t\lambda_\ell)\phi_\ell \implies g(\boldsymbol{x}) = \sum_{\ell\in\mathbb{N}} a_\ell \phi_\ell(\boldsymbol{x}), \tag{3}$$

A modal decomposition of g.

Introduction - DMD and EDMD

From here advancing the dynamics to time t is equivalent to writing

$$\mathcal{K}^t g(\boldsymbol{x}) = \sum_{\ell \in \mathbb{N}} a_\ell \exp(t\lambda_\ell)\phi_\ell(\boldsymbol{x}) \tag{4}$$

If we suppose

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{N_O} a_\ell \psi_\ell(\boldsymbol{x}) \tag{5}$$

with basis $\{\psi_\ell\}_{\ell=1}^{N_O}$ of a subspace, then applying $\mathcal{K}$ for discrete time implies that

$$\mathcal{K}^{\delta t} g(\boldsymbol{x}) = \sum_{\ell=1}^{N_O} a_\ell \exp(\delta t \lambda_\ell)\psi_\ell(\boldsymbol{x}) = \sum_{\ell=1}^{N_O} \psi_\ell(\boldsymbol{x})(\boldsymbol{K}_O^T \boldsymbol{a})_\ell + r(\boldsymbol{x}; \boldsymbol{K}_O) \tag{6}$$

Introduction - Time advancement

Where

$$\boldsymbol{K}_O = \underset{K}{\text{argmin}} \, ||\boldsymbol{\Psi}_+ - K\boldsymbol{\Psi}_-||_F^2 \tag{7}$$

We define $\boldsymbol{\Psi}_\pm$ to be

$$\boldsymbol{\Psi}_- = \left(\Psi_1 \ \Psi_2 \ \cdots \Psi_{N_T}\right), \quad \boldsymbol{\Psi}_+ = \left(\Psi_2 \ \Psi_3 \ \cdots \Psi_{N_T+1}\right) \tag{8}$$

where $\{\Psi_j\}$ is an observable of the time series of interest $\{\boldsymbol{y}_j\}$. $\boldsymbol{K}_O$ is a one-step mapping from each data point to the next that is found using a singular value decomposition (SVD). Thus

$$\boldsymbol{\Psi}_- = \mathbf{U}\boldsymbol{\Sigma}\mathbf{W}^\dagger \implies \boldsymbol{K}_O = \boldsymbol{\Psi}_+\mathbf{W}\boldsymbol{\Sigma}^{-P}\mathbf{U}^\dagger. \tag{9}$$

Another eigendecomposition gives

$$\boldsymbol{K}_O = \mathbf{V}\mathbf{T}\mathbf{V}^{-1}, \tag{10}$$

with $\lambda_\ell = \ln((\mathbf{T})_{\ell\ell})/\delta t$.

Introduction - Flow reconstruction and advancement

With this the dynamics can be approximated and advanced as

$$y(t;\boldsymbol{x}) \approx \boldsymbol{K}_m \exp(t\boldsymbol{\Lambda})\mathbf{V}^{-1}\boldsymbol{\Psi}(\boldsymbol{x}) \qquad (11)$$

where $\boldsymbol{\Psi}$ is the representation of the initial condition in terms of the observables, $\boldsymbol{K}_m$ is the $N_S \times N_O$ matrix whose columns are the Koopman modes and $\boldsymbol{\Lambda}$ is the diagonal matrix whose elements are $\lambda_\ell = (\boldsymbol{\Lambda})_{\ell\ell}$.

This is the basis for Dynamic Mode Decomposition (DMD) when $N_O = N_S$ and the ultimate takeaway is that if an optimal set of observables is found, the error from DMD can be reduced.

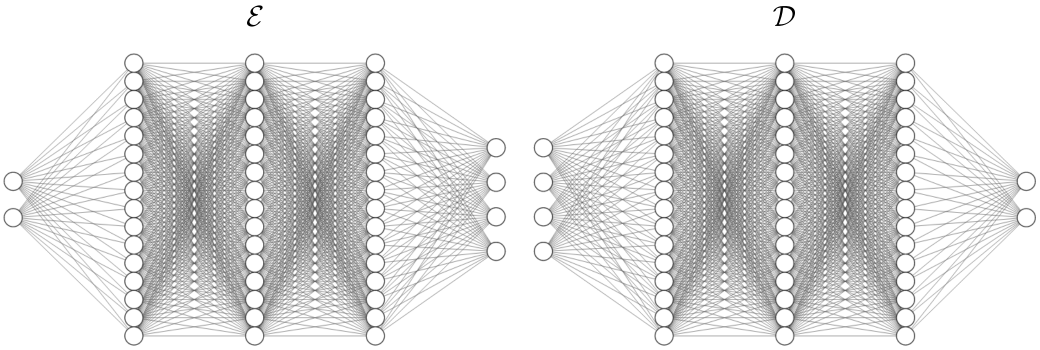
The key innovation of Lago [1] is to use a neural network to come up with the collection of observables on $\{\boldsymbol{y}_j\}$ that allow for the best prediction of future system states, we call this method Deep Learning Enhanced DMD (DLDMD). This is implemented by defining an encoder $\mathcal{E} : \mathbb{R}^{N_S} \rightarrow \mathbb{R}^{N_O}$ and decoder $\mathcal{D} : \mathbb{R}^{N_O} \rightarrow \mathbb{R}^{N_S}$ composed of dense layers such that

$$(\mathcal{D} \circ \mathcal{E})(\boldsymbol{x}) = \boldsymbol{x} \tag{12}$$

We choose $N_O \geq N_S$ and an appropriate loss function so that $\mathcal{E}$ and $\mathcal{D}$ give a richer space of observables, called the latent space, for EDMD to use when advancing the dynamics. The implementation of NNs for this purpose requires a method of tuning to allow $\mathcal{E}$ and $\mathcal{D}$ to learn the best representations possible.

Introduction - The Network Architecture

Example of DLDMD newtork with $N_S = 2$, $N_O = 4$, and $N_L = 3$ where every hidden layer has 16 neurons. The layers in the Figure are labeled, sequentially, left to right: Enc in, Enc 0, Enc 1, Enc 2, Enc out, Dec in, Dec 0, Dec 1, Dec 2, Dec out.



Introduction - The Dilemma

For dense layers, passing a vector of data $\boldsymbol{x} \in \mathbb{R}^d$ through a layer L can be written as

$$L(\boldsymbol{x}; \mathbf{A}, \sigma, \mathbf{b}) = \sigma(\mathbf{A}\boldsymbol{x} + \mathbf{b}) \tag{13}$$

for some matrix $\mathbf{A} \in \mathbb{R}^{N_O \times d}$, vector $\mathbf{b} \in \mathbb{R}^{N_O}$ and (typically nonlinear) activation function $\sigma : \mathbb{R}^{N_O} \to \mathbb{R}^{N_O}$. The next state for each layer is dependent on the previous, so a reasonable framing would be as one of a discrete dynamical system of the form

$$Q_{n+1} = P(Q_n) \tag{14}$$

In the interest of examining convergences, one might consider the following limit

$$\lim_{n\to\infty} ||Q_{n+1} - Q_n||_2 \tag{15}$$

for some two-norm on the space that Q_n inhabits. While this "one-step" Cauchy convergence will tell us whether the machine is approaching a particular configuration point-wise, this might not actually tell us much of anything else.

Kullback-Leibler Divergence - Entropy

A measure of the uncertainty was proposed in 1948 and this measure is known as *informational entropy* [6] or, simply, entropy. For a continuous distribution with density function $f(x)$, the entropy is given by

$$h[f] = \mathbb{E}\left[-\log(f(x))\right] = -\int_X f(x)\log(f(x))\;dx \tag{16}$$

For the normally distributed X defined above, we have the probability density function

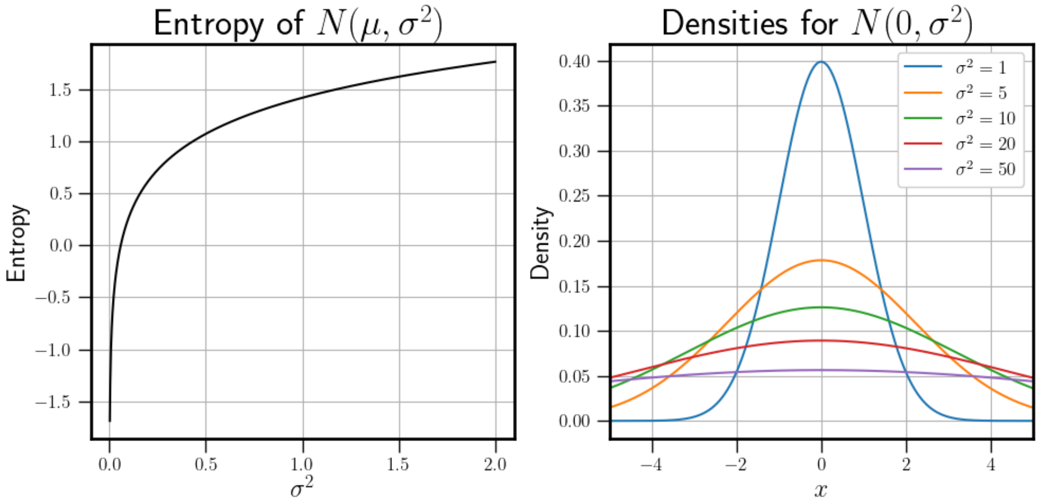
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) \tag{17}$$

and the entropy integral for this evaluates to

$$h[f] = \frac{1}{2}\left(\log(2\pi\sigma^2)+1\right) \tag{18}$$

which only depends on the variance.

Kullback-Leibler Divergence - Entropy example



Kullback-Leibler Divergence - Basic Definitions

The Kullback-Leibler Divergence (KLD) is a statistical distance between a pair of probability distributions which measures how different a distribution P is from a reference distribution Q. A simple interpretation of the divergence of P from Q is the expected excess surprise from using Q as a model when the actual distribution is P [5]. If P and Q are continuous probability distributions defined on X with probability density functions p and q, the KLD formula is

$$D_{KL}(P \parallel Q) = \int_X p(x) \log\left(p(x)/q(x)\right) \, dx \tag{19}$$

The reason we can consider this a "distance" is that the KLD is non-negative, $D_{KL}(P \parallel Q) \geq 0$.

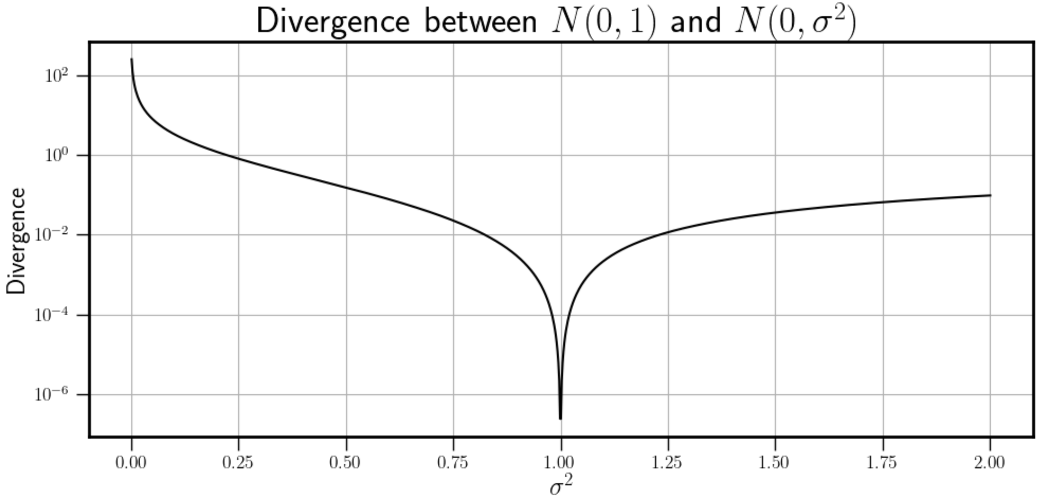
For example, consider the random variables $X_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $X_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ with corresponding density functions given by equation 17; the divergence between the distributions is

$$D_{KL}(p \parallel q) = \frac{1}{2}\left(\ln\left(\frac{\sigma_2^2}{\sigma_1^2}\right) + \frac{\sigma_1^2}{\sigma_2^2} + \frac{(\mu_1 - \mu_2)^2}{\sigma_2^2} - 1\right), \tag{20}$$

Letting $\mu_1 = \mu_2$, $\sigma_1^2 = 1$, and allowing σ_2^2 to vary, we can write

$$D_{KL}(p \parallel q) = \frac{1}{2}\left(\ln \sigma_2^2 + \frac{1}{\sigma_2^2} - 1\right), \tag{21}$$

Kullback-Leibler Divergence - normal distribution example



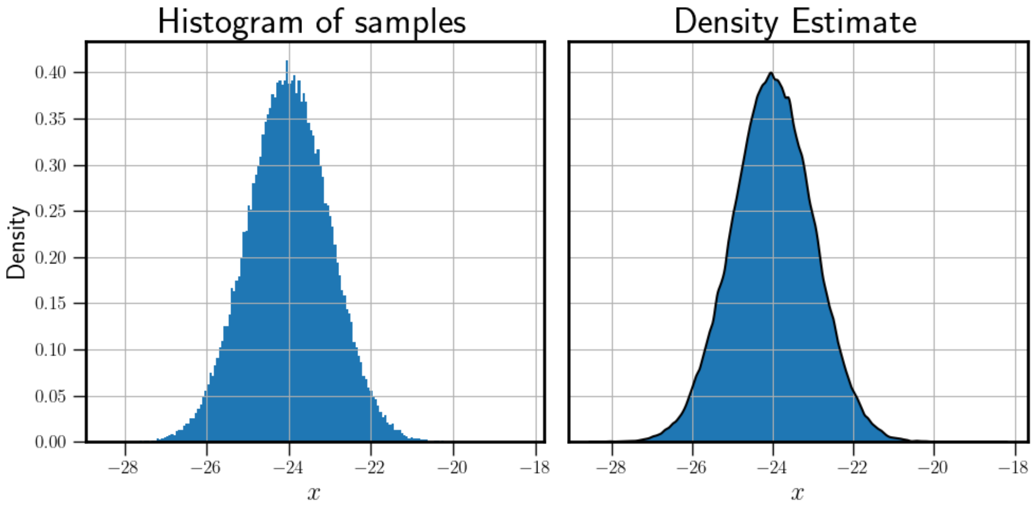
Kullback-Leibler Divergence - Kernel Density Estimation

The most basic idea behind KDE is using a bin centering method and a smoothing factor, called a kernel, to approximate a probability density f from measured data in the form of a histogram. This is accomplished with the *kernel density estimator* for f given by

$$\hat{f}(x; K, h) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right), \tag{22}$$

where K is the chosen kernel, h is what's called the bandwidth parameter, and $\{x_i\}_{i=1}^{n}$ is the data that generates our histogram. Per Epanechnikov, the choice of kernel does not provide much statistical significance [3]; but the choice of the bandwidth parameter h is very crucial for finding a density estimate that approximates the underlying density function appropriately and can be thought of as an analogue of the bin width for the affliated histogram. We choose h using the Improved Sheather-Jones algorithm [2], which algorithmically finds an optimal bandwidth using the data itself.

Kullback-Leibler Divergence - KDE example



Kullback-Leibler Divergence - Measuring Entropy Flow

For a converging machine, for each layer we should have that

$$\mathbf{W}_{l,\mathcal{E}}^{(n)}, \mathbf{W}_{l,\mathcal{D}}^{(n)} \rightarrow \mathbf{W}_{l,\mathcal{E}}^{*}, \mathbf{W}_{l,\mathcal{D}}^{*} \text{ as } n \rightarrow \infty \tag{23}$$

Consequently, we can charaterize any set of weights via the formulas

$$\mathbf{W}_{l,\mathcal{E}}^{(n)} = \mathbf{W}_{l,\mathcal{E}}^{*} + \mathbf{W}_{l,\mathcal{E}}^{(n),f}, \ \mathbf{W}_{l,\mathcal{D}}^{(n)} = \mathbf{W}_{l,\mathcal{D}}^{*} + \mathbf{W}_{l,\mathcal{D}}^{(n),f}, \tag{24}$$

where $\mathbf{W}_{l,\mathcal{E}}^{(n),f}$ and $\mathbf{W}_{l,\mathcal{D}}^{(n),f}$ are fluctuations from the steady state. First-order differencing gives us detrended matrices:

$$\delta\mathbf{W}_{l,\mathcal{E}}^{(n)} = \mathbf{W}_{l,\mathcal{E}}^{(n+1)} - \mathbf{W}_{l,\mathcal{E}}^{(n)} = \mathbf{W}_{l,\mathcal{E}}^{(n+1),f} - \mathbf{W}_{l,\mathcal{E}}^{(n),f} \tag{25}$$

Using KDE we generate an affliated empirical probability distribution $p_{l,\mathcal{E}}^{(n)}(w)$, which we can use to find consecutive divergences with KLD:

$$D = \left\{ D_{KL}\left(p_{l,\mathcal{E}}^{(n+1)} \middle\| p_{l,\mathcal{E}}^{(n)} \right) \right\}_{n=1}^{N_E-2} \tag{26}$$

Kullback-Leibler Divergence - Implementation details

After the training of each model for N_E epochs, the data that we have to play with is

$$\mathbf{W}_{\mathcal{E}} = \left\{ \mathbf{W}_{l,\mathcal{E}}^{(n)} \right\}_{n=1,l=1}^{N_E,N_L}, \mathbf{W}_{\mathcal{D}} = \left\{ \mathbf{W}_{l,\mathcal{D}}^{(n)} \right\}_{n=1,l=1}^{N_E,N_L}, \tag{27}$$

which are the sets of weights of the layers of the encoder and decoder. For each layer l, the set of detrended matrices are computed

$$\delta \mathbf{W}_{l,\mathcal{E}} = \left\{ \mathbf{W}_{l,\mathcal{E}}^{(n+1),f} - \mathbf{W}_{l,\mathcal{E}}^{(n),f} \right\}_{n=1}^{N_E-1}, \tag{28}$$

these matrices are flattened and their entries are used as the data for KDE. The kernel used is the Epanechnikov kernel, defined as

$$K(x) = \begin{cases} \frac{3(1-x^2)}{4}, & |x| < 1 \\ 0, & |x| \geq 1 \end{cases}, \tag{29}$$

which is optimal in the mean-square error sense.

Kullback-Leibler Divergence - Implementation details cont.

From this, we obtain the set of density estimates for each detrended matrix

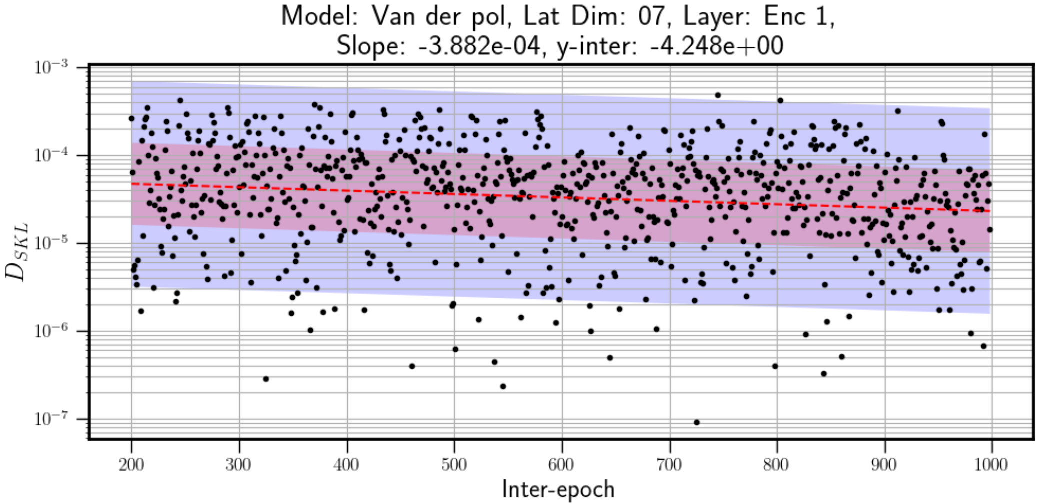
$$p_{l,\mathcal{E}} = \text{KDE}(\delta \mathbf{W}_{l,\mathcal{E}}) = \left\{ p_{l,\mathcal{E}}^{(n)}(w) \right\}_{n=1}^{N_E - 1} \tag{30}$$

where w is the weight value itself and $p_{l,\mathcal{E}}(w)$ is the approximate density associated with that weight value. Given these density approximations, we can now compute what we are truly interested in:

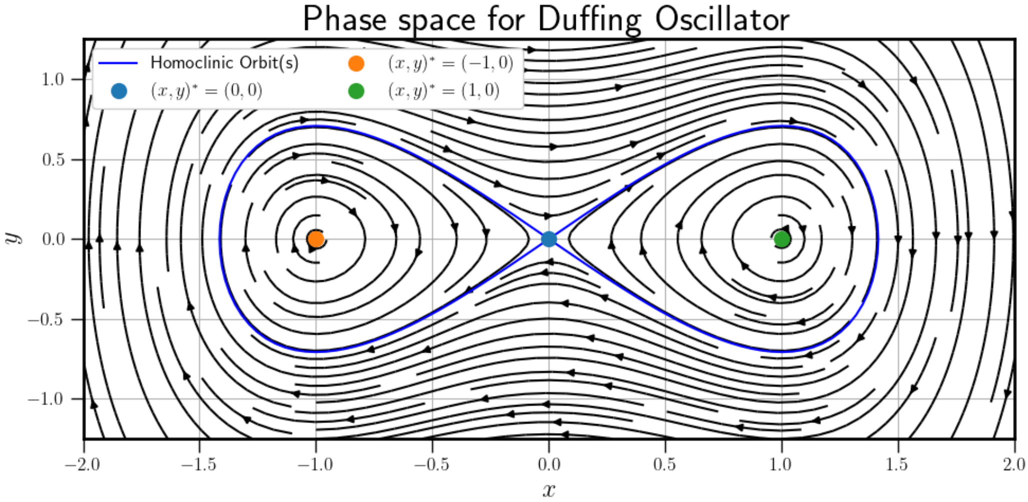
$$D_l = \left\{ D_{SKL}\left(p_{l,\mathcal{E}}^{(n+1)} \middle\| p_{l,\mathcal{E}}^{(n)} \right) \right\}_{n=1}^{N_E - 2} \tag{31}$$

Unlike in the first mention, we instead use the SKLD instead of the KLD. We are interested in what these divergences tell us about information flow in the model's weights and seek to identify specific classifiers of good training.

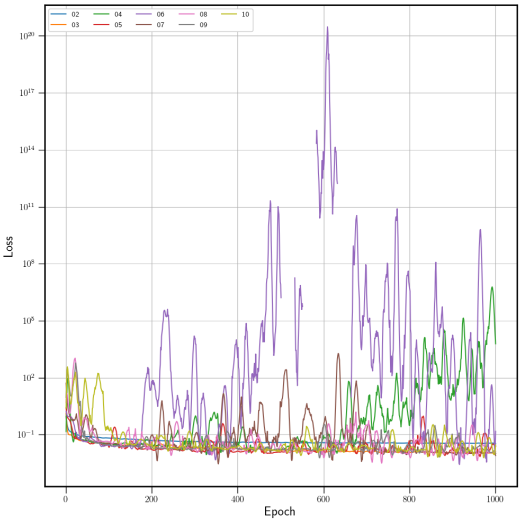
Kullback-Leibler Divergence - Example fitting

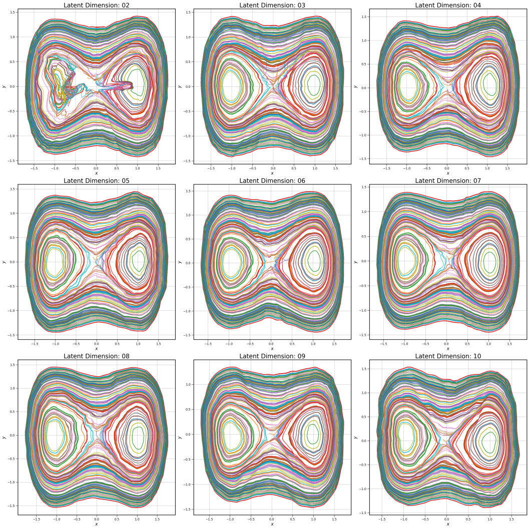


Results - Duffing

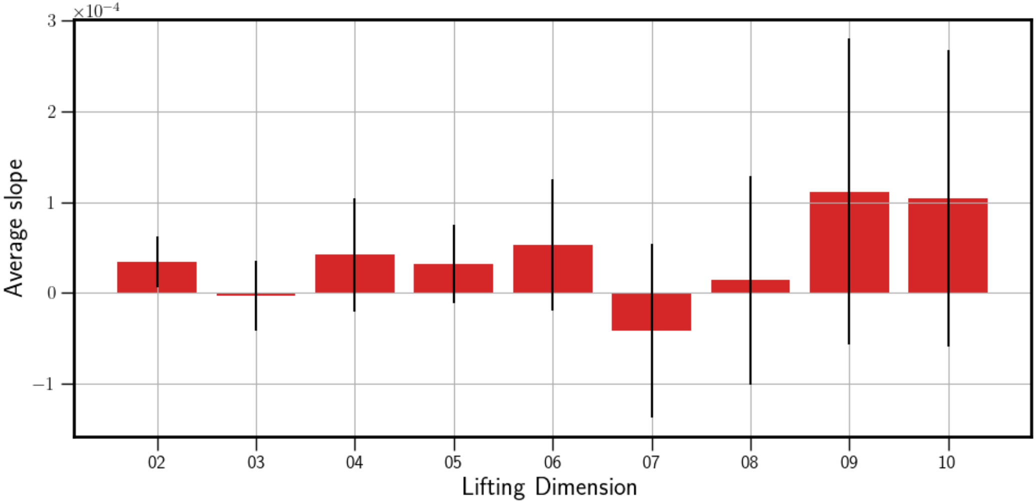


Results - Duffing loss curves and phase space

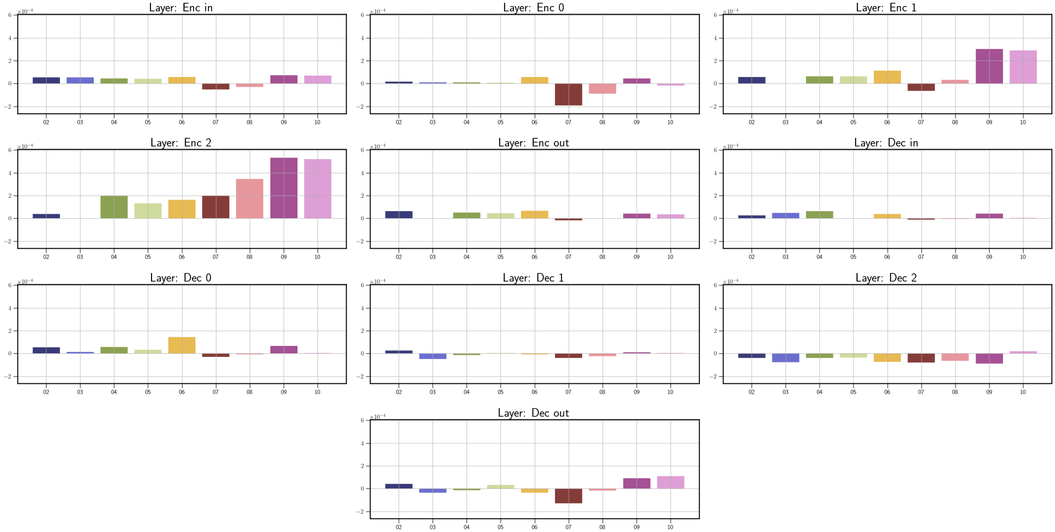




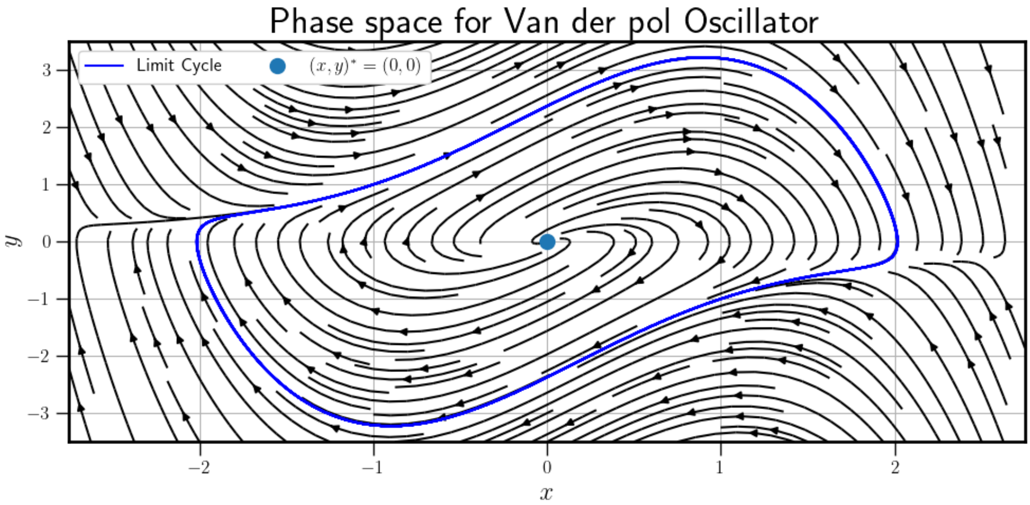
Results - Duffing linear fit slope averages and variances



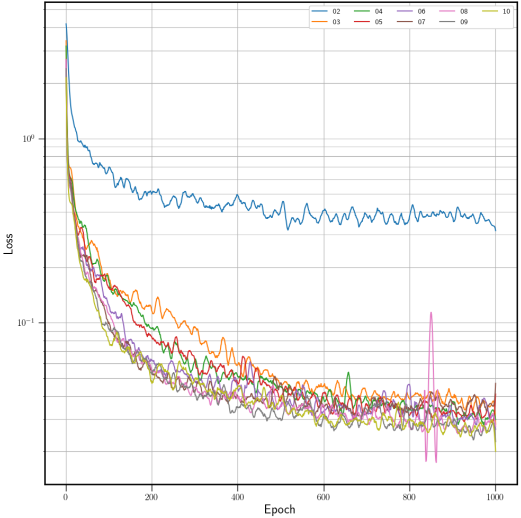
Results - Duffing linear fit slopes

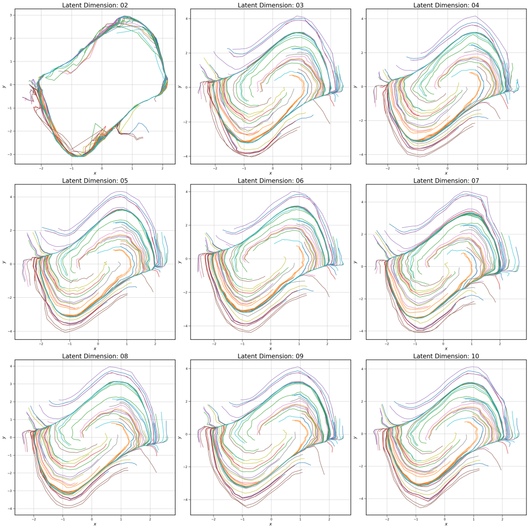


Results - Van der Pol

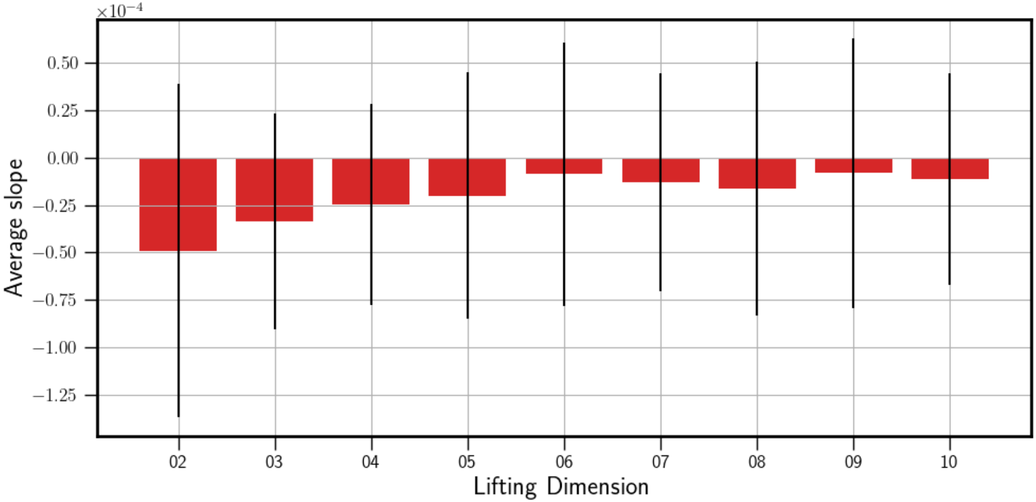


Results - Van der Pol loss curves and phase space

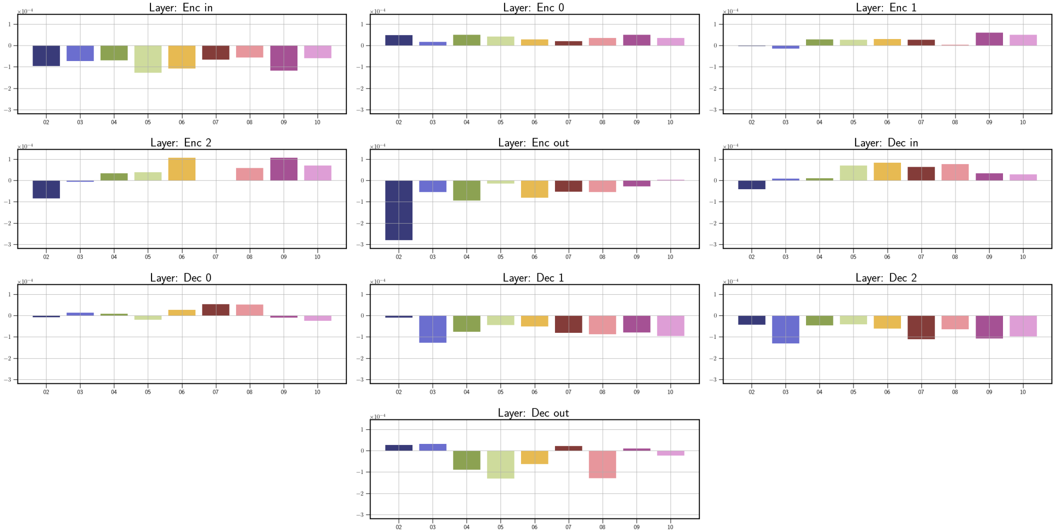




Results - Van der Pol linear fit slope averages and variances



Results - Van der Pol linear fit slopes



Discussion

- The Good
- Optimal Parameters
- Non-unitary densities
- Statistical issues with edge layers
- Future work

References

D.J. Alford-Lago, C. W. Curtis, A. T. Ihler, and O. Issan.
Deep Learning Enhanced Dynamic Mode Decomposition.
Chaos: An Interdisciplinary Journal of Nonlinear Science, 32, 2022.

Z. I. Botev, J. F. Grotowski, and D. P. Kroese.
Kernel density estimation via diffusion.
The Annals of Statistics, 38(5):2916 – 2957, 2010.

V. A. Epanechnikov.
Non-parametric estimation of a multivariate probability density.
Theory of Probability & Its Applications, 14(1):153–158, 1969.

B.O. Koopman.
Hamiltonian systems and transformations in Hilbert space.
Proc. Nat. Acad. Sci., 17:315–318, 1931.

S. Kullback and R. A. Leibler.
On Information and Sufficiency.
The Annals of Mathematical Statistics, 22(1):79 – 86, 1951.

The End!

Thank you for your time!

Questions?