# Improving on DLDMD with Convolutional Layers

Susana Munguia and Joseph Diaz
*SDSU Department of Mathematics and Statistics*
(Dated: May 11, 2022)

## I. INTRODUCTION

In the study dynamical systems a central problem is how to derive models from measured data to facilitate the prediction of future states. Many approaches and techniques exist in the literature, from deriving sets of governing equations via application of the simple physical principles to the statistically meaningful Principal Component Analysis and other modal decompositions. The main goal of many of these methods is to come up with a generalized framework so that the dynamics from data can be more easily studied and understood.

Another category that we can add to this collection, one that uses many modern advancements in computational resources, is that of Machine Learning (ML) models; which leverage the power and expressiveness of Neural Networks (NNs) and abundance of data to forge a new data-driven paradigm of model discovery. [9] showed that feed forward networks can act as universal approximators of continuous functions between Euclidean spaces to any arbitrary degree of accuracy using as few as one hidden layer and ML models of all stripes have been used for tasks as diverse as image classification and network flow optimization.

In the last 10 years, many researchers have made attempts to bring the many tools of ML to bear on the topic of dynamical systems. This paper explores a very recent result by [1] that builds off of years of work in pursuit of data-driven model discovery for prediction and control. The major idea at the core of this work is the Koopman operator [10], and the numerical method of approximating it's spectra: DMD [11]. In the space of modal decompositions, this method focuses on exploiting the Koopman operators ability to generate flows from observables that are connected to the otherwise unknown dynamics that generate the data. Choice of how to represent data for DMD is paramount and, unfortunately, some natural choices of representation can lead to utterly useless results.

This is where ML comes in. The Universal Approximator property can be used to come up with an optimal set of observables to use with DMD. This is the key innovation of [1], but they are not the first to consider bringing ML to bear on DMD. [12] attempted a similar approach but overcomplicated the matter by introducing an auxiliary network to effectively parameterize the eigen-spectrum of Koopman operator. Even their results build off of those of [2], [14], and [5], which demonstrates a rich stable of methods and approaches for improving on Koopman style analysis like DMD and it's extensions.

DMD is chronically contrasted with other methods like Sparse Identification of Nonlinear Dynamics (SINDy) [6], which seeks to construct the governing equations from data by supposing that a sparse set of basis functions can appropriately represent the dynamics; while this gives a more direct form of investigation by seeking analytical objects, the trade-off is that you must test many combinations sets of bases to be confident that you've found a accurate representation.

Further work in this field also exists, many results using topology and the idea of space conjugacy have shown promise with respect to discovering optimal embeddings for chaotic time series. In some sense, these flow embedding results [3, 8, 12] are also used in [1] as part of the key innovation that justify the expectation for flow representation to map between spaces without being destroyed or otherwise ruined.

## II. METHODS

As it says on the tin, we seek a predictive model for a time series $\{\boldsymbol{y}_j\}_{j=1}^{N^T+1}$, which are the measurements of a uniformly sampled unknown dynamical system of the form

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

As established in the introduction, there is an old and venerable literature dedicated to deriving system using classical methods. What this literature will admit is that such a process is not trivial and even difficult or impossible to do in practice; particularly for nonlinear phenomena worth investigating. We want something that is more easily generalized and algorithmic.

One method that can be leveraged for this is via the Koopman Operator $\mathcal{K}$. If we denote $\boldsymbol{\varphi}(t;\boldsymbol{x}) = \boldsymbol{y}(t)$ to be the flow map affiliated with the initial condition $\boldsymbol{x}$ and denote $g : \mathcal{M} \to \mathbb{C}$ to be a square integrable observable of the system then, according to [10], there exists a linear representation of the flow map given by $\mathcal{K}$;

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

which means that $\mathcal{K}$ is a linear time-advancement operator of the dynamics. This might seem like it trivializes the problem, given that we now have a linear system, but it does not. The Koopman operator is an *infinite* dimensional operator, so we've traded a potentially nonlinear problem for an infinite dimensional linear one.

In order to actually use this new representation, it suffices to find the eigenvalues $\{\lambda_\ell\}$, and their affiliated

eigenfunctions $\{\boldsymbol{\phi}_\ell\}$, of the Koopman Operator such that

$$\mathcal{K}^t\boldsymbol{\phi}_\ell = \exp(t\lambda_\ell)\boldsymbol{\phi}_\ell \implies g(\boldsymbol{x}) = \sum_{\ell\in\mathbb{N}} c_\ell\boldsymbol{\phi}_{\boldsymbol{\ell}}(\boldsymbol{x}),$$

where we can essentially construct a modal decomposition of $g$. From here advancing the dynamics to time $t$ is equivalent to writing

$$\mathcal{K}^t g(\boldsymbol{x}) = \sum_{\ell\in\mathbb{N}} a_\ell\boldsymbol{\phi}_\ell(\boldsymbol{\varphi}(t,\boldsymbol{x}))$$

The most useful property of this framing is that we have a global linearization of the flow, with a major caveat: Generally, finding these eigenvalues and eigenfunctions is impossible. This led to the development of the already mentioned and much much lauded Dynamic Mode Decomposition (DMD) and it's extensions, which seeks to numerically approximate a finite number of these modes. We'll focus on the particular extension that [1] implements: Extended DMD (EDMD) [13]. With EDMD, we suppose that

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{N_O} a_\ell\boldsymbol{\psi}_\ell(\boldsymbol{x})$$

which is to say that the observable $g$ exists in a finite dimensional subspace of $L^2(\mathcal{M})$, applying the Koopman operator implies that

$$\begin{aligned}\mathcal{K}^{\delta t}g(\boldsymbol{x}) &= \sum_{\ell=1}^{N_O} a_\ell\boldsymbol{\phi}_\ell(\boldsymbol{\varphi}(\delta t,\boldsymbol{x})) \\ &= \sum_{\ell=1}^{N_O} \boldsymbol{\phi}_\ell(\boldsymbol{x})(\boldsymbol{K}_O^T\boldsymbol{a})_\ell + r(\boldsymbol{x};\boldsymbol{K}_O)\end{aligned}$$

for discrete time step $\delta t$. $K_O$ is the $N_O \times N_O$ matrix that minimizes

$$\boldsymbol{K}_O = \underset{K}{\operatorname{argmin}}\,||\Psi_+ - K\Psi_-||_F^2$$

and $r(\boldsymbol{x},\boldsymbol{K}_O)$ is a residual that represents the total error due to DMD. If the ansatz that $g$ lives in a finite space holds, then $r$ is identically 0. We define $\Psi_\pm$ to be

$$\Psi_- = (\Psi_1\ \Psi_2\ \cdots\Psi_{N_T}), \quad \Psi_+ = (\Psi_2\ \Psi_3\ \cdots\Psi_{N_T+1})$$

where $\{\Psi_j\}$ is an observable of the time series of interest $\{\boldsymbol{y}_j\}$. What the expression for $\boldsymbol{K}_O$ tells us is that, we are trying to find a one-step mapping from each data point to the next. Practically speaking, $\boldsymbol{K}_O$ is found using an SVD, with which we can write

$$\Psi_- = U\Sigma W^\dagger \implies \boldsymbol{K}_O = \Psi_+ W\Sigma^{-P}U^\dagger$$

where $-P$ denotes the Moore-Penrose pseudo-inverse and $\dagger$ denotes the conjugate transpose. This gives us an expression for $r$ in terms of the observables $\Psi$:

$$E_r(\boldsymbol{K}_O) = \left|\left|\Psi_+(I - WW^\dagger)\right|\right|_F$$

Finding the eigenvalues, eigenfunctions, and Koopman modes comes down to an eigen-decomposition, from which the dynamics can be approximated as

$$y(t;\boldsymbol{x}) \approx V\exp(t\Lambda)V^{-1}\boldsymbol{\Psi}(\boldsymbol{x})$$

where $\boldsymbol{K}_O = VTV^{-1}$, $\Lambda_{\ell\ell} = \ln(T_{\ell\ell})/\delta t$ is a diagonal matrix and $\boldsymbol{\Psi}$ is the representation of the initial condition in terms of the observables. The key innovation of [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. This is implemented by defining an encoder $\mathcal{E} : N_S \to N_O$ and decoder $\mathcal{D} : N_O \to N_S$ composed of Dense layers such that

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

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. As such, a loss function that correctly identifies and prioritizes the desired properties is a necessary condition for the DLDMD to function as needed. A natural choice considering these constraints is given by

$$\mathcal{L} = \alpha_1\mathcal{L}_{\text{recon}} + \alpha_2\mathcal{L}_{\text{dmd}} + \alpha_3\mathcal{L}_{\text{pred}} + \alpha_4\,||\boldsymbol{W}_g||_2^2$$

where

$$\begin{aligned}\mathcal{L}_{\text{recon}} &= \frac{1}{N_T+1}\sum_{j=1}^{N_T+1}||\boldsymbol{y}_j - (\mathcal{D}\circ\mathcal{E})(\boldsymbol{y}_j)||_2\,, \\ \mathcal{L}_{\text{dmd}} &= E_r(\boldsymbol{K}_O), \\ \mathcal{L}_{\text{pred}} &= \frac{1}{N_T}\sum_{j=1}^{N_T}\left|\left|\boldsymbol{y}_{j+1} - \mathcal{D}\left(VT^jV^{-1}\mathcal{E}(x)\right)\right|\right|_2,\end{aligned}$$

Each component guides the machine to a particular outcome:

1. $\mathcal{L}_{\text{recon}}$ is the Mean Squared Error (MSE) of each time step with respect to the reconstruction from the composition of $\mathcal{E}$ and $\mathcal{D}$. This component ensures that, under training, the network effectively acts as a near identity transformation for data that is fed into it. This quality allows the DMD advanced trajectories to be recovered from the higher dimensional latent space back to the original dimension of the data.

2. $\mathcal{L}_{\text{DMD}}$ is the error associated with DMD. Consequently, this component is the one that is most responsible for finding the optimal set of observables for DMD. $\mathcal{E}$ immerses the data into a higher dimensional latent space, in effect acting as our set of observable; minimizing this gives us greater flexibility in the latent space.

3. $\mathcal{L}_{\text{pred}}$ is the (MSE) for each forward time prediction due to DMD and immersion/submersion due to $\mathcal{E}/\mathcal{D}$. In addition to balancing the last conditions, this condition ensures that the DMD step in the latent dimension is consistent with the next time-step from the time series after encoding, advancing, and decoding.

4. $\boldsymbol{W}_g$ is the vectorized quantity that represents all weights in both $\mathcal{E}$ and $\mathcal{D}$. This is really only a regularization condition to keep the coefficients of the weight matrices from blowing up in value as the model trains, which can be a concern for ML models.

5. $\alpha_1$, $\alpha_2$, $\alpha_3$, and $\alpha_4$ are 4 positive constants that allow us to assign a weighting to each component of the loss. This allows the loss function to be dynamically weighted to prioritize some conditions over others. In [1], $\alpha_1 = \alpha_2 = \alpha_3 = 1$ and $\alpha_4 < 10^{-10}$.

This details the methods used in their paper, and we'll now move on to the details of our changes and our results.

## III. RESULTS

In [1], the DLDMD demonstrates high performance of point-wise prediction and reconstruction of system structure for non-chaotic dynamics. The metric for point-wise prediction is evaluated using the average MSE (Mean Squared Error), while reconstruction of the system is visually evaluated against the observed structure. A diverse set of phase-plane geometries were studied to fully test the capabilities of the DLDMD. The systems studied along with their phase-plane dynamics and average MSE follow:

1. Harmonic Oscillator - Nonlinear Center, planar system - $3.69 \cdot 10^{-3}$

2. Duffing Equation - Two Nonlinear Centers surrounded by Homoclinic orbits, planar system - $3.47 \cdot 10^{-3}$

3. Van Der Pol Oscillator - One globally attracting Limit Cycle, with slow/fast dynamics , planar system - $2.87 \cdot 10^{-2}$

4. Lorenz-63 - Chaotic attractor (globally attracting), three-dimensional system - $1.79 \cdot 10^{0}$

As we can see, chaotic systems produce less accuracy in point-wise predictions. This is one of many drawbacks that [1] highlighted to the reader. Delay embeddings during the EDMD step is suggested as an alternative method for the reconstruction of chaotic systems and noisy observations. The most optimal latent dimension is determined by user trial and error, choosing the latent dimension, $N_0$, that produces the smallest final loss. An improvement to the DLDMD would be to integrate finding the optimal latent space dimension into the learning of the NN, which would require a more complicated algorithm.

Our focus is to demonstrate how the DLDMD performs on various scales of noise added to time series of three dynamical systems and modifications to improve the model's robustness to noise. The three systems we will study include 2 and 3 stated above as well as the Rössler chaotic attractor, a three-dimensional system. The implementation details for Duffing and Van der Pol are the same as [1] and thus, is available to those interested. Since Rössler was not studied previously in [1], we will state it's implementation details. To generate the test data for the Rossler system, we choose a time step of $\delta t = 0.15$ and ran simulations till $t_f = 60$, then ran until $t_f = 120$ for the prediction time. Due to computing limitation, we instead used 1,500 trajectories for each system, in which 1,000 is reserved for training, 300 for validation and 200 for testing. Initial conditions were randomly sampled from $x \in (-20, 20)$, $y \in (-20, 20)$, and $z \in (0, 35)$.

Noisy data is created by adding uniform random samples from the Gaussian distribution with mean 0 and a fixed standard deviation to each entry of the time series. The standard deviation is increased in value until we notice a "Break-point" (B.P.) of the method, the smallest magnitude in noise in which the reconstruction fails to capture the behavior of the denoised data. In Figures 1 - 3, we first plot the data with Gaussian noise scale of 0.001 and it's reconstruction. The plots that follow are the B.P. value along with it's reconstruction. This is done for 1) Duffing, 2) Van Der Pol, and c) Rössler. Notice, that the value of the B.P. is increasing from system to system. So, Van der pol and Rossler are more robust to noise than Duffing. Global attractors *should be* robust to noise, if a trajectory is pushed away for some small value, in the long-term, we expect it to land on the chaotic attractor. On the other hand, the dynamics of the Duffing oscillator is separated by nonlinear centers and homoclinic orbits around those centers. That system *should be* sensitive to a trajectory which is perturbed between the sections of nonlinear centers and homoclinic orbits. Overall, the DLDMD surpassed our expectation in its robustness to noise, as well as the authors of the paper.

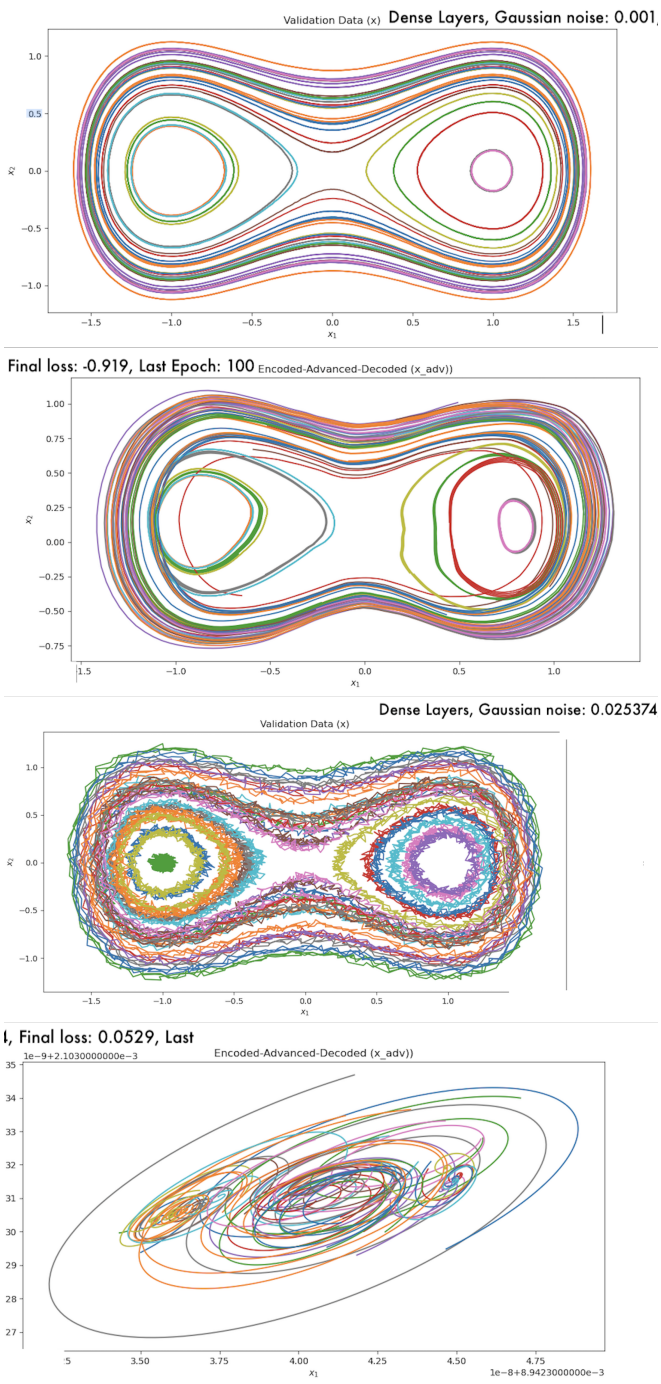


FIG. 1: Noisy Duffing Time Series with DLDMD. a) Duffing with Gaussian noise scale 0.001. b)Predicted Future Dynamics c) Duffing with Gaussian noise scale B.P. = 0.025374 d) Predicted Future Dynamics.

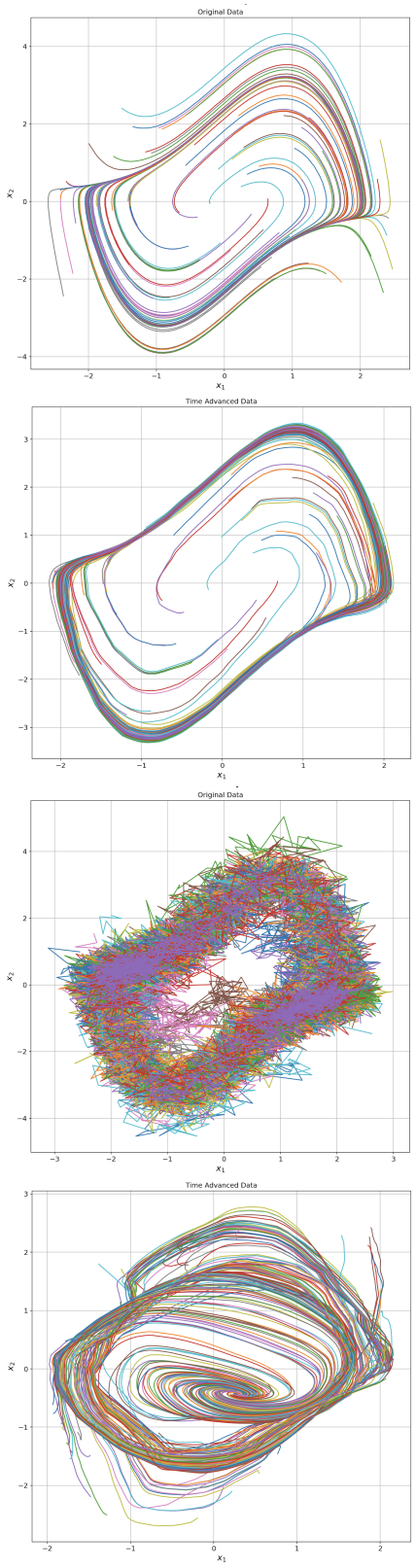


FIG. 2: Noisy Van Der Pol Time Series. a) Duffing with Gaussian noise scale 0.001. b) Predicted Future Dynamics c) Duffing with Gaussian noise scale B.P. = 0.25625 d) Predicted Future Dynamics.

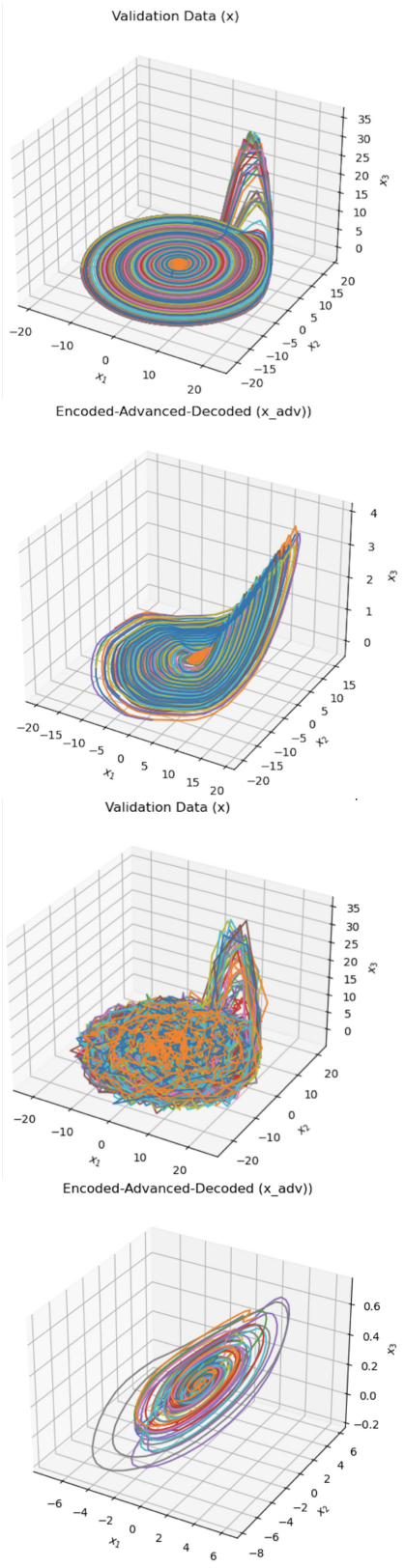


FIG. 3: Noisy Rossler Time Series. a) Duffing with Gaussian noise scale 0.001. b) Predicted Future Dynamics c) Duffing with Gaussian noise scale B.P. = 1.25 d) Predicted Future Dynamics.

To improve the DLDMD's performance with noisy data, modifications to the loss function as well as the AutoEncoder NN can be made. In [7], Ghosh et al. compared the most widely used loss functions amongst diverse data sets. The loss functions evaluated include Cross-Entropy (CCE), Mean Squared Error (MSE), and Mean Absolute Error (MAE). MSE and MAE produced similar results amongst all the data outperforming CCE. MAE had the best results. Thus, MSE and MAE are robust to noise. The Loss function for the DLDMD has terms that are defined by the MSE, specifically $\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{pred}}$, thus this explains the already good results to noisy data we observed.

Modifications can be made to the AutoEncoder, currently, it has fully connected layers or Dense layers. These are the standard layers implemented for the simplest of NNs. The nodes in the preceding layer connects to each node in the next layer via tunable weights. Once the weights are optimized, the next layers is defined by inputting the dot product of the weight vector with the input vector into a nonlinear function. However, in Convolutional layers (C.L.s), the relationship between the next node and previous nodes is more complicated. C.L.s require an additional parameter, kernel size, which defines the size of the filter applied to the inputs. If you take the vector of inputs as a 1D vector, ie $1 \times N$, and define a kernel size of 3, the NN will produce a $1 \times 3$ vector of randomly generated values. Then, the Discrete Convolution is applied to the input vector with this created filter vector. The Discrete Convolution produces a new vector, of shape $1 \times (N - (\text{kernel size}) + 1)$, defined as outputs of dot products of the input vector with the filter as we traverse the input vector. In this application, the three Dense layers used to define the Encoder and Decoder are replaced by 1D Convolutional layers, thus we have a fully Convolutional AutoEncoder (C.A.E).

In figure 4-6, the results for implementing the CAE within the DLDMD method is shown. Each figure consists of noisy data at the value of the previous B.P. along with a much larger value to show the improved robustness to noise.

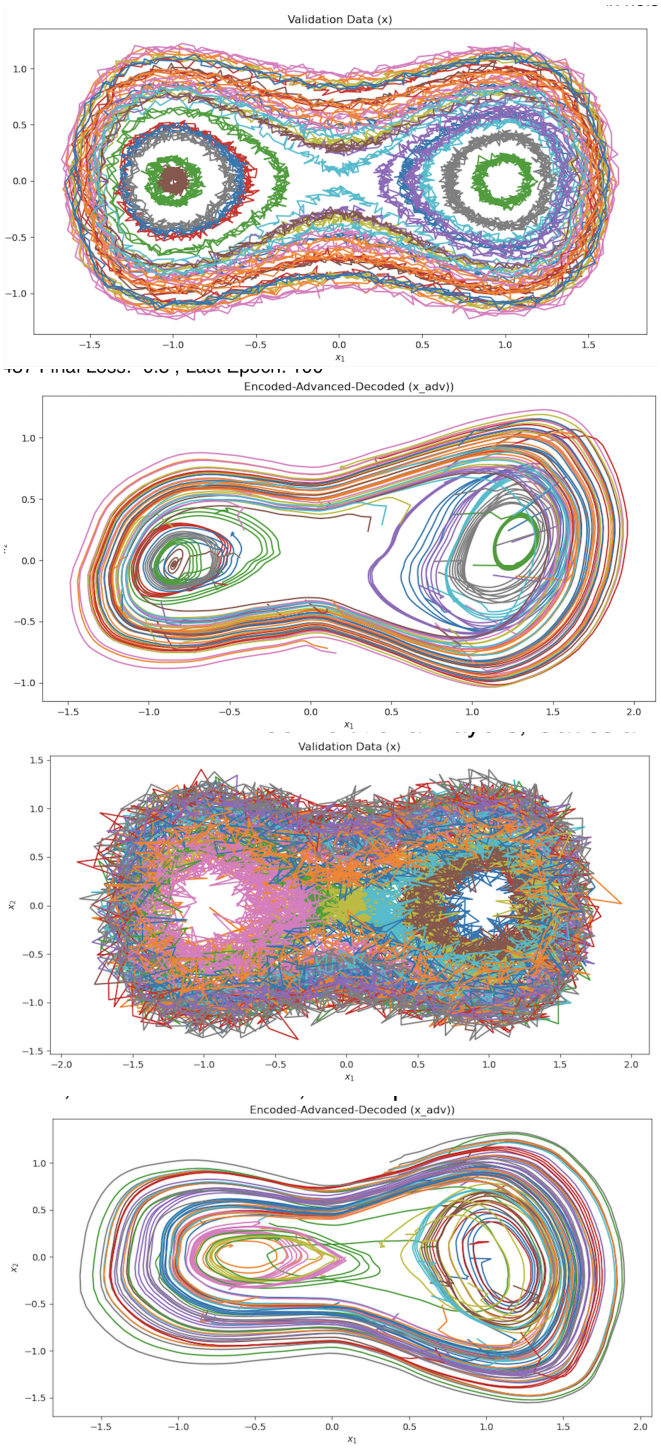


FIG. 4: Noisy Duffing Time Series with DLDMD + CAE. a) Duffing with Gaussian noise scale previous B.P. = 0.025374 b) Predicted Future Dynamics c) Duffing with Gaussian noise scale larger than B.P., 0.1 d) Predicted Future Dynamics.

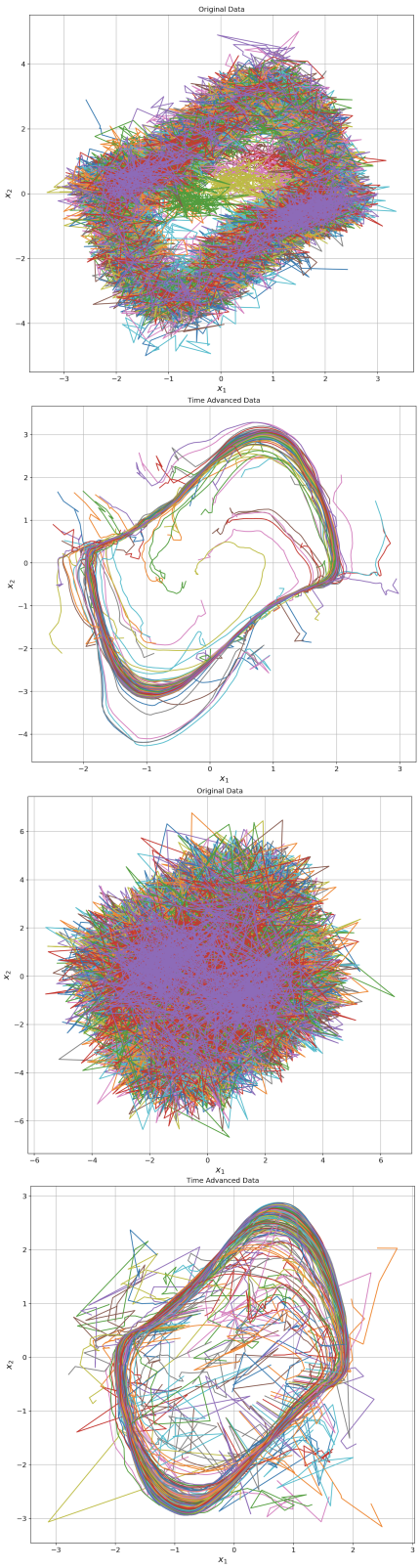


FIG. 5: Noisy Van Der Pol Time Series with DLDMD + CAE. a) Van Der Pol with Gaussian noise scale previous B.P. = 0.25625 b) Predicted Future Dynamics c) Duffing with Gaussian noise scale larger than B.P. 1, d) Predicted Future Dynamics.

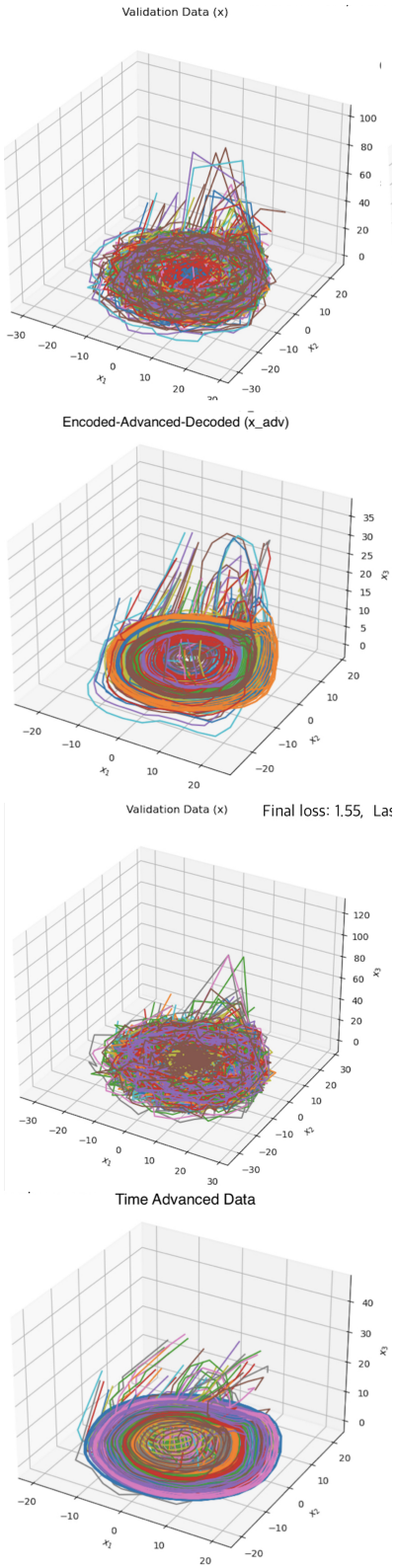


FIG. 6: Noisy Rössler Time Series with DLDMD + CAE. a) Rossler with Gaussian noise scale B.P. = 1.25 b) Predicted Future Dynamics c) Duffing with Gaussian noise scale larger than B.P., 2 d) Predicted Future Dynamics.

## IV. DISCUSSION

Before discussing future work and other important points it must be said that while [1] mentions their belief that noise should ruin the predictive power of DLDMD, they don't even experimentally verify this. As a matter of course, our first objective was to test this and the model performed much better than expected. Being wrong about an ultimately positive quality of their method is an excellent problem to have, but this assumption should have been tested. Further, due to the definition of the $\boldsymbol{K}_O$ matrix that is used to push the dynamics forward temporally, it should have been anticipated that some small magnitude stationary noise would not significantly ruin the methods. Such a fitting from time-step to time-step can effectively see the larger trend through the noise, bolstering the robust loss function even before the introduction of de-noising convolutional layers. This criticism notwithstanding, DLDMD is already quite effective on noisy data, so much so that the author of the paper was surprised when we shared our results with them. A future line of inquiry would be to investigate the effects and countermeasures for non-stationary noise sources and non-uniform sampling intervals.

While we managed to get reasonably good results with the Rössler attractor, getting good results for other chaotic systems like Lorenz63 and the Chua circuit will require more sophisticated techniques, since (DL)DMD performs poorly on them even with perfect data. Some approaches like [4] consider using Hankel matrices and Takens' Embedding to handle such cases, and integrating such an approach with convolutional layers for de-noising is a natural next step given our results.

A caveat to all things ML is the crucial and sometimes arbitrary choice of hyper parameters. When attempting to train the machine for larger noise scales, we ocassionally ran into issues where ill-conditioned matrices would cause SVD and Eigen decomposition failures after which the code would error out. Even more strangely, the diagnostic plots would tell us that the machine was successfully predicting the future dynamics and reconstructing the time series data. Changing hyper parameters could alleviate this, but there no is method for choosing them appropriately to solve any one issue and it's unclear how any single change will affect the model's evolution. The interpretability of the model with respect to how it manages to predict future dynamics is an intractable problem that comes with the territory of ML. Using Koopman for the actual dynamics and relegating the ML to an ancillary role helps mightily with the interpretation problem, but future work with respect to how NNs do their work more concretely is forthcoming.

## V. CONCLUSION

In this paper, we have extended the utility of a recent result in the field of ML and data-driven modeling and

shown that DLDMD is a robust numerical method for non-chaotic systems that only requires data and a rather arbitrary set of hyper parameters to produce meaningful results. While not as scientifically satisfying as classical methods, data-driven methods like DMD fit nicely into the modern inquiry into dynamical systems from a data scientific perspective and with the still ongoing data revolution it's unlikely that such methods will fall into disuse over more classic methods. The simple extension that we implemented here is evidence of how modular and polymorphic these algorithms can be, and is a natural extension of the DLDMD that [1] implemented.

Using machine learning to extend and improve on data-driven numerical methods like DMD has born fruit in our heavily data dependent world and these models have proven their validity from a practical point of view. The innovation of new architectures will only lead to more nuanced ways to model problems given by real world data and the future is bright for the marriage of ML with dynamical systems.


[1] D.J. Alford-Lago, C. W. Curtis, A. T. Ihler, and O. Issan. Deep Learning Enhanced Dynamic Mode Decomposition. 2022.

[2] E. Bollt, Q. Li, F. Dietrich, and I. Kevrekidis. On matching, and even rectifying, dynamical systems through koopman operator eigenfunctions. *SIAM J. Appl. Dyn. Sys.*, 17.2:1925–1960, 2018.

[3] Jason J. Bramburger, Steven L. Brunton, and J. Nathan Kutz. Deep learning of conjugate mappings. *arXiv*, 2104.01874, 2021.

[4] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz. Chaos as an intermittently forced system. *Nature Comm.*, 8(1), 2017.

[5] S.L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. *Ann. Rev. Fluid Mech.*, 52:477–508, 2020.

[6] S.L. Brunton, J.L. Proctor, and J.N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 113:3932–3937, 2016.

[7] A. Ghosh, H. Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks. 2017.

[8] W. Gilpin. Deep reconstruction of strange attractors from time series. In *34th Conference on NeurIPS*, 2020.

[9] K. Hornik, M. Stinchcombe, and H. White. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 1989.

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

[11] J.N. Kutz, S.L. Brunton, B.W. Brunton, and J.L. Proctor. *Dynamic Mode Decomposition: Data-driven modeling of complex systems*. SIAM, Philadelphia, PA, 2016.

[12] B. Lusch, J. N. Kutz, and S. L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Comm.*, 9:4950, 2018.

[13] M.O. Williams, I. G. Kevrekidis, and C. W. Rowley. A data-driven approximation of the Koopman operator: extending dynamic mode decomposition. *J. Nonlin. Sci.*, 25:1307–1346, 2015.

[14] E. Yeung, S. Kundu, and N.O. Hodas. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In *American Control Conference*, 2019.