LATENT DIFFEOMORPHIC DYNAMIC MODE DECOMPOSITION

Willem Diepeveen
Department of Mathematics
University of California, Los Angeles
Los Angeles, CA 90095, USA
wdiepeveen@math.ucla.edu

Jon Schwenk
Earth and Environmental Sciences Division
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
jschwenk@lanl.gov

Andrea L. Bertozzi
Department of Mathematics
University of California, Los Angeles
Los Angeles, CA 90095, USA
bertozzi@ucla.edu

ABSTRACT

We present Latent Diffeomorphic Dynamic Mode Decomposition (LDDMD), a new data reduction approach for the analysis of non-linear systems that combines the interpretability of Dynamic Mode Decomposition (DMD) with the predictive power of Recurrent Neural Networks (RNNs). Notably, LDDMD maintains simplicity, which enhances interpretability, while effectively modeling and learning complex non-linear systems with memory, enabling accurate predictions. This is exemplified by its successful application in streamflow prediction.

1 Introduction

Predicting the next element in a sequence is rarely a straightforward, one-to-one process. Instead, it often relies on retaining and utilizing information from earlier in the sequence. For example, in language, the choice of the next word depends on more than just the preceding word. Similarly, in meteorology, identical weather conditions do not always lead to the same outcomes across different locations or times. Therefore, a crucial aspect of effective prediction is memory: the ability to learn and maintain relevant information over extended periods.

In many such applications, understanding the reasoning behind predictions – that is, understanding what information has been learned and maintained – is as important as making accurate forecasts. Streamflow prediction, crucial for flood preparedness and infrastructure design, illustrates this. Long Short-Term Memory networks (LSTMs) [7] – a special type of Recurrent Neural Network (RNN) [15] – have revolutionized this field by outperforming traditional hydrological models for the streamflow prediction problem [10]. However, they often lack interpretability, which is a common issue among similar machine learning methods [2, 5].

In scenarios without memory where a one-to-one mapping exists, Dynamic Mode Decomposition (DMD) [16] has proven highly effective as an interpretable data reduction method. Its success has led to numerous extensions [19, 3, 6]. Recent advancements have focused on adapting DMD to non-linear cases, primarily through Koopman operator theory [14]. This approach typically follows one of two paths: using non-linear dictionaries [20, 11] or employing non-linear mappings [17, 12, 13, 8], with the latter being more popular. A key challenge in the non-linear mapping approach is the need for an (approximate) inverse function to recover the actual state from the dynamics in the mapped domain. While early attempts used approximations [17, 12], more recent research has shown that diffeomorphisms [13, 8] offer improved results.

Currently, all versions of Dynamic Mode Decomposition (DMD) rely on the assumption of a one-to-one mapping. This work aims to address this limitation by developing a novel approach that combines the high performance of RNN-like

Our code is available at https://github.com/wdiepeveen/Latent-Diffeomorphic-Dynamic-Mode-Decomposition.

methods with the interpretability of DMD. Specifically, we aim to construct a non-linear DMD-like framework that mimics RNNs and can operate effectively in scenarios where the one-to-one mapping assumption does not hold. This would bridge the gap between the predictive power of RNN-like methods and the explanatory capabilities of traditional DMD, potentially opening up new avenues for interpretable machine learning for data reduction in complex, non-linear systems.

Contributions In this work we: (i) first propose Diffeomorphic Dynamic Mode Decomposition (DDMD) as a new non-linear DMD extension; (ii) then use it to construct Latent Diffeomorphic Dynamic Mode Decomposition (LDDMD) and discuss why this method effectively combines the strengths of DMD and RNNs while avoiding their respective limitations; and (iii) finally show that LDDMD shows promising results on toy and real data for streamflow prediction.

2 General Diffeomorphic Dynamic Mode Decomposition

Dynamic Mode Decomposition (DMD) assumes that the evolution of a time series $\{\mathbf{x}^i\}_{i=0}^N \subset \mathbb{R}^m$ is governed by an unknown matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$ through

$$\mathbf{x}^i = \mathbf{A}\mathbf{x}^{i-1}, \quad i = 1, \dots, N. \tag{1}$$

In addition, assuming that m is even, and $\mathbf{A} - \mathbf{I}$ is diagonalizable with all eigenvalues having a non-zero imaginary component, we can always write (1) as

$$\mathbf{x}^i = \mathbf{W}^{-1}\mathbf{K}\mathbf{W}\mathbf{x}^{i-1}, \quad i = 1, \dots, N, \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{m\times m}$ is an invertible matrix and $\mathbf{K} \in \mathbb{R}^{m\times m}$ is a block diagonal matrix with $m/2$ blocks of the form

$$e^{-\mu_j \Delta t} \begin{bmatrix} \cos(\omega_j \Delta t) & -\sin(\omega_j \Delta t) \\ \sin(\omega_j \Delta t) & \cos(\omega_j \Delta t) \end{bmatrix}, \quad \text{for } j = 1, \dots, m/2, \tag{3}$$

with $\mu_j, \omega_j \in \mathbb{R}$ and $\Delta t > 0$ being the time between observations in the time series.

For the above assumptions on $\mathbf{A}$ to hold, the time series needs (at least) to be centered, i.e., $\frac{1}{N+1}\sum_{i=0}^N \mathbf{x}^i = \mathbf{0}$. For a non-centered time series, we can repeat the above for $\{\mathbf{x}^i - \mathbf{b}\}_{i=0}^N$ with $\mathbf{b} = \frac{1}{N+1}\sum_{i=0}^N \mathbf{x}^i$, which gives

$$\mathbf{x}^i = \mathbf{W}^{-1}\mathbf{K}\mathbf{W}(\mathbf{x}^{i-1} - \mathbf{b}) + \mathbf{b}, \quad i = 1, \dots, N. \tag{4}$$

We propose *Diffeomorphic Dynamic Mode Decomposition (DDMD)*, which directly extends (4) by replacing the invertible linear mapping $\mathbf{x} \mapsto \mathbf{W}(\mathbf{x} - \mathbf{b})$ by a diffeomorphism $\varphi : \mathbb{R}^m \to \mathbb{R}^m$. In particular, DDMD assumes that

$$\mathbf{x}^i = (\varphi^{-1} \circ \mathbf{K} \circ \varphi)(\mathbf{x}^{i-1}) = (\varphi^{-1} \circ (\mathbf{K})^i \circ \varphi)(\mathbf{x}^0), \quad i = 1, \dots, N. \tag{5}$$

Remark 1. *We note that this approach can be seen as: (i) a combination of the approaches taken in [12] – where a similar parametrization for $\mathbf{K}$ is used, but φ^{-1} and φ are replaced by (non-invertible) neural networks – and state-of-the-art approaches taken in [13, 8] – where diffeomorphisms are used, but no block diagonal assumptions are made on $\mathbf{K}$, and (ii) a special case of [1] where the eigenvectors of the Koopman operator are only assumed to generate a local diffeomorphism.*

3 Latent Diffeomorphic Dynamic Mode Decomposition

The actual problem we are interested is an adaptation of the DDMD problem. That is, given a time series $\{\mathbf{x}^i\}_{i=0}^N \subset \mathbb{R}^m$, we aim to predict the corresponding sequence element in $\{\mathbf{v}^i\}_{i=0}^N \subset \mathbb{R}^n$, assuming that there is no one-to-one mapping $\mathbf{x} \mapsto \mathbf{v}$, i.e., there is memory in the system. To address the one-to-one mapping assumption still inherent in DDMD, we introduce additional variables to predict the next sequence element. In particular, we assume that the evolution of a time series $\{(\mathbf{x}^i, \mathbf{v}^i)\}_{i=0}^N \subset \mathbb{R}^m \times \mathbb{R}^n$ is governed by dynamics

$$(\mathbf{x}^i, \mathbf{p}^i) = (\Phi^{-1} \circ \mathcal{K} \circ \Phi)(\mathbf{x}^{i-1}, \mathbf{p}^{i-1}), \quad \mathbf{v}^i = g(\mathbf{p}^i). \tag{6}$$

Here, $\Phi : \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^m \times \mathbb{R}^d$ is a diffeomorphism defined as

$$\Phi(\mathbf{x}, \mathbf{p}) := (\varphi_1(\mathbf{x}), f(\mathbf{x}) + \varphi_2(\mathbf{p})), \tag{7}$$

where $\varphi_1 : \mathbb{R}^m \to \mathbb{R}^m$ and $\varphi_2 : \mathbb{R}^d \to \mathbb{R}^d$ are diffeomorphisms and $f : \mathbb{R}^m \to \mathbb{R}^d$ is a coupling mapping. The inverse of Φ is given by

$$\Phi^{-1}(\mathbf{y}, \mathbf{q}) = (\varphi_1^{-1}(\mathbf{y}), \varphi_2^{-1}(\mathbf{q} - f(\varphi_1^{-1}(\mathbf{y})))). \tag{8}$$

Furthermore,

$$\mathcal{K} = \begin{bmatrix} \mathbf{K}_1 & \\ & \mathbf{K}_2 \end{bmatrix}, \tag{9}$$

where $\mathbf{K}_1 : \mathbb{R}^{m\times m}$ and $\mathbf{K}_2 : \mathbb{R}^{d\times d}$ are block-diagonal matrices with blocks of the form (3)[1], $g : \mathbb{R}^d \to \mathbb{R}^n$ is a mapping, and the latent variable $\mathbf{p}$ is initialized by some $\mathbf{p}^0 \in \mathbb{R}^d$.

Remark 2. *The dynamical system (6) encodes a state defined by* $\mathbf{x}$ *and* $\mathbf{p}$*, where* $\mathbf{x}$ *is independent of* $\mathbf{p}$*, but* $\mathbf{p}$ *is partially driven by* $\mathbf{x}$*, and* $\mathbf{v}$ *depends on* $\mathbf{p}$*, which incorporates both* $\mathbf{x}$ *and system memory, thus breaking the one-to-one assumption of previous DMD versions.*

The goal of *Latent Diffeomorphic Dynamic Mode Decomposition (LDDMD)* is slightly different than DDMD. That is, we are not interested in explicitly representing the underlying dynamics of the $\mathbf{x}^i$, but want to predict $\mathbf{v}^i$ given $\mathbf{x}^i$ and the current time index i. In other words, we aim to represent the dynamics of the $\mathbf{p}^i$, which allows us to focus on the $\mathbf{p}^i$ part in (6), i.e.,

$$\mathbf{p}^i = \varphi_2^{-1}(\mathbf{K}_2(f(\mathbf{x}^{i-1}) + \varphi_2(\mathbf{p}^{i-1})) - f(\mathbf{x}^i)) = \varphi_2^{-1}((\mathbf{K}_2)^i(f(\mathbf{x}^0) + \varphi_2(\mathbf{p}^0)) - f(\mathbf{x}^i)). \tag{10}$$

Since $\mathbf{p}^0$ is some initialization, we are free to write $\mathbf{q}_0$ for $f(\mathbf{x}^0) + \varphi_2(\mathbf{p}^0)$. Also dropping all the subscripts gives us the LDDMD problem, where we assume that

$$\mathbf{v}^i = g(\mathbf{p}^i) = g(\varphi^{-1}((\mathbf{K})^i\mathbf{q}_0 - f(\mathbf{x}^i))), \tag{11}$$

and aim to learn the mapping g, the diffeomorphism φ, the coupling f, and the initialization $\mathbf{q}^0$ along with the μ_j, ω_j that parametrize $\mathbf{K}$ through (3).

Remark 3. *The latent dynamics* $(\mathbf{K})^i\mathbf{q}^0$ *encode the memory of the system and is evolving independently of* $\mathbf{x}^i$*. In other words, we really have a non-one-to-one mapping from* $\mathbf{x}^i$ *to* $\mathbf{v}^i$*, where the memory is encoded in a more interpretable way than for RNN-like methods, with the key difference that we explicitly have to solve for an initial condition* $\mathbf{q}_0$*. Having said that, standard RNNs famously have trouble with maintaining information for longer terms and can be unstable. Both of these issues can be attributed to vanishing or exploding gradients, which LDDMD can also suffer from for* $\mu_j \neq 0$ *in* $\mathbf{K}$*, see (3). So in practice, we assume for LDDMD that* $\mu_j = 0$*.*

4 Training

To learn an LDDMD representation of a data set $\{(\mathbf{x}^i, \mathbf{v}^i)\}_{i=0}^N \subset \mathbb{R}^m \times \mathbb{R}^n$, we need loss functions, parametrizations and initializations thereof.

We propose the following training loss to solve for the LDDMD approximation of a time series $\{(\mathbf{x}^i, \mathbf{v}^i)\}_{i=0}^N$ with an even dimension d for the latent space:

$$\mathcal{L}_{LDDMD}(\theta, \eta, \omega, \mathbf{q}_0, \xi) := \sum_{i=0}^{N} \|\mathbf{v}^i - g_\xi(\varphi_\theta^{-1}((\mathbf{K}_\omega)^i\mathbf{q}_0 - f_\eta(\mathbf{x}^i)))\|_2. \tag{12}$$

Next, although the specific neural networks used to parametrize the mappings might vary from application to application, there are some general choices for the case $n = 1$ – which is the case for streamflow prediction – that come with well-motivated initializations.

In this case, we found the following parameterization suitable: (i) the diffeomorphism as an invertible neural network composed of a single additive coupling layer [4] with learnable polynomial activation on either the even or odd indices, initialized such that $\varphi_{\theta^{(0)}}(\mathbf{x}) = \mathbf{x}$; (ii) the coupling as a feedforward network with learnable polynomial activation functions, initialized such that $f_{\eta^{(0)}}(\mathbf{x}) \approx \mathbf{0}$; (iii) the frequencies ω_j as learnable parameters, initialized as the $d/2$ most prevalent frequencies in the Fourier spectrum[2] of the 1D signal $\{\mathbf{v}^i\}_{i=0}^N$; (iv) the latent initialization as a learnable vector, initialized as $\mathbf{q}_0^{(0)} = \mathbf{0}$; (v) and the mapping g_ξ as a feedforward network with softplus activation functions and random initialization. Limiting the diffeomorphism to only use additive coupling layers serves as a form of regularization, helping to prevent the mapping from producing excessively large or small output values.

[1]Note that for this to be well-defined, we need both m and d to be even.

[2]Factoring in the time Δt between observations.

5 Numerical experiments

In this section, we demonstrate the performance of LDDMD through experiments on data reduction for streamflow data $\{v^i\}_{i=0}^N \subset \mathbb{R}$ from synthetic and real meteorological data $\{\mathbf{x}^i\}_{i=0}^N \subset \mathbb{R}^m$. Specifically, we show that LDDMD effectively reduces the dataset using a low-dimensional latent space and extrapolates beyond its training data, both in nearly ideal settings with noisy data and in real-world scenarios.

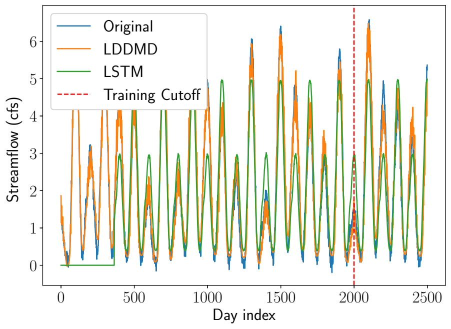
The key metric that will be used to quantify performance is the *Nash–Sutcliffe model efficiency coefficient (NSE)*:

$$\mathrm{NSE} = 1 - \frac{\sum_{i=0}^N (v^i - g_\xi(\varphi_\theta^{-1}((\mathbf{K}_\omega)^i \mathbf{q}_0 - f_\eta(\mathbf{x}^i))))^2}{\sum_{i=0}^N (v^i - \bar{v})^2}, \tag{13}$$

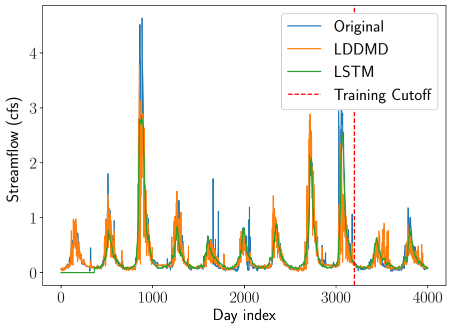
where $\bar{v} := \frac{1}{N}\sum_{i=0}^N v^i$. For intuition, a higher NSE is better, where $\mathrm{NSE} = 1$ is perfect and a model having $\mathrm{NSE} = 0.5$ or over indicates that the dynamics are captured well. For reference, we also compare our results to LSTMs using the standard implementation in `NeuralHydrology` [9], which is a widely used package that is optimized for streamflow prediction.

For all experiments in this section, details on the data sets are provided in *Supplementary Material 1*, detailed LDDMD-training configurations in *Supplementary Material 2* and additional results in *Supplementary Material 3*.

Finally, all of the experiments are implemented using `PyTorch` in Python 3.8 and run on a 2 GHz Quad-Core Intel Core i5 with 16 GB RAM.



(a) Synthetic data results



(b) Real-world data results

Figure 1: For both synthetic and real-world data LDDMD (proposed) is not only able to learn the underlying dynamics from the training data, but also to extrapolate far beyond the training horizon. The `NeuralHydrology` LSTM model performs well on data that it is optimized for (b), but is doing worse on non-typical streamflow data (a).

Synthetic data For the synthetic data set, 2D data and a 2D latent process govern the streamflow through dynamics of the form (6). The data used for training the LDDMD model comes from this underlying dynamics model, but noise has been added to it. The dimensions $m = 2$ and $d = 2$ are also used for parametrizing the mappings in the LDDMD model. The results in Figure 4a suggest that despite the noise we are able to find a model to fit the data, predict the signal well-beyond its training window and outperform the LSTM: for LDDMD we have $\mathrm{NSE} = 0.96$ on training data and $\mathrm{NSE} = 0.94$ on validation data, whereas for the LSTM[3] we have $\mathrm{NSE} = 0.80$ on training data and $\mathrm{NSE} = 0.80$ on validation data. Notably, the model has arguably almost recovered the ground truth. In particular, the learned frequency $\omega_1 = 0.0097$ is almost exactly the ground truth frequency $\omega^* = 0.0099$ and despite over-parametrization of the coupling mapping f_η, even the latent dynamics look similar to the ground truth up to rigid body transformation and rescaling[4], from which we retrieve the information that the the process that governs the streamflow oscillates between two regions of the latent space (see Figure 2 in *Supplementary Material 3*).

Real-world data For the real-world data, we have $m = 14$ for the meteorological data and use $d = 10$ for the latent space to approximate the streamflow. We parametrize the mappings in the LDDMD model accordingly. The

[3]We do not take the 365 day sequence length of the LSTM into account here for the NSE.

[4]Note that this is not expected at all as there is just a 1-dimensional variable v^i to reconstruct each two-dimensional variable $\mathbf{p}^i$, but the fact that we are able to get a similar shape reminisces of Takens' theorem [18]. We leave exploration of this connection for future research.

results in Figure 4b indicate that the model effectively fits the data, albeit with lower performance compared to the numericaly optimized LSTM, and provides (almost) practically useful streamflow predictions that extend well beyond the training window: for LDDMD we have $\mathrm{NSE} = 0.75$ on training data and $\mathrm{NSE} = 0.46$ on validation data, whereas for the LSTM we have $\mathrm{NSE} = 0.89$ on training data and $\mathrm{NSE} = 0.89$ on validation data. Despite the performance gap between our general-purpose LDDMD and the numerically optimized LSTM in terms of NSE, the results remain encouraging and highlight key areas for further improvement of the base method. Although a thorough analysis of the performance gap is left for future research, we expect that possible explanations involve that LSTMs are better at removing noise from the signal (see Figure 3 in *Supplementary Material 3*), which would significantly increase the NSE. Finally, as there is no ground truth process available, we also leave matching these results to possible underlying processes for future research.

6 Conclusions

In conclusion, this work presents Latent Diffeomorphic Dynamic Mode Decomposition (LDDMD), a novel method that combines Dynamic Mode Decomposition and Recurrent Neural Networks for data reduction and nonlinear time series prediction. LDDMD offers both interpretability and strong performance, as shown in streamflow forecasting, though further improvements are needed for state-of-the-art results.

Acknowledgments

All authors are supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under award DE-SC0025589 and under Triad National Security, LLC ('Triad') contract grant 89233218CNA000001 [FWP: LANLE2A2]. JS is also supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project numbers 20170668PRD1 and 20210213ER. ALB is also supported by NSF grant DMS-2152717.

References

[1] Erik M Bollt, Qianxiao Li, Felix Dietrich, and Ioannis Kevrekidis. On matching, and even rectifying, dynamical systems through koopman operator eigenfunctions. *SIAM Journal on Applied Dynamical Systems*, 17(2):1925–1960, 2018.

[2] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

[3] Scott TM Dawson, Maziar S Hemati, Matthew O Williams, and Clarence W Rowley. Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Experiments in Fluids*, 57:1–19, 2016.

[4] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[5] Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. Backprop kf: Learning discriminative deterministic state estimators. *Advances in neural information processing systems*, 29, 2016.

[6] Maziar S Hemati, Clarence W Rowley, Eric A Deem, and Louis N Cattafesta. De-biasing the dynamic mode decomposition for applied koopman spectral analysis of noisy datasets. *Theoretical and Computational Fluid Dynamics*, 31:349–368, 2017.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[8] Xiao Hou, Jin Zhang, and Le Fang. Invertible neural network combined with dynamic mode decomposition applied to flow field feature extraction and prediction. *Physics of Fluids*, 36(9), 2024.

[9] Frederik Kratzert, Martin Gauch, Grey Nearing, and Daniel Klotz. Neuralhydrology—a python library for deep learning research in hydrology. *Journal of Open Source Software*, 7(71):4050, 2022.

[10] Frederik Kratzert, Daniel Klotz, Claire Brenner, Karsten Schulz, and Mathew Herrnegger. Rainfall–runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022, 2018.

[11] Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10), 2017.

[12] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[13] Yuhuang Meng, Jianguo Huang, and Yue Qiu. Koopman operator learning using invertible neural networks. *Journal of Computational Physics*, 501:112795, 2024.

[14] Igor Mezić. Koopman operator, geometry, and learning of dynamical systems. *Not. Am. Math. Soc*, 68(7):1087–1105, 2021.

[15] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning internal representations by error propagation, 1985.

[16] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[17] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. *Advances in neural information processing systems*, 30, 2017.

[18] Floris Takens. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Warwick 1980: proceedings of a symposium held at the University of Warwick 1979/80*, pages 366–381. Springer, 2006.

[19] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[20] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25:1307–1346, 2015.

A Data set details

A.1 Synthetic data

The synthetic data in Section 5 of the main article is of the form

$$(\mathbf{x}^i, \mathbf{p}^i) = (\Phi^{-1} \circ \mathcal{K} \circ \Phi)(\mathbf{x}^{i-1}, \mathbf{p}^{i-1}), \quad \mathbf{v}^i = g(\mathbf{p}^i).$$

with

$$\Phi(\mathbf{x}, \mathbf{p}) := (\varphi_1(\mathbf{x}), f(\mathbf{x}) + \varphi_2(\mathbf{p})),$$

and

$$\mathcal{K} = \begin{bmatrix} \mathbf{K}_1 & \\ & \mathbf{K}_2 \end{bmatrix}.$$

The diffeomorphisms are given by

$$\varphi_1(\mathbf{x}) := (2(\mathbf{x}_1 - \sin(\mathbf{x}_2)), \frac{1}{4}\mathbf{x}_2),$$

and

$$\varphi_2(\mathbf{x}) := (2(\mathbf{x}_1 - \mathbf{x}_2^2 - 3), 3\mathbf{x}_2),$$

the coupling is given by

$$f(\mathbf{x}) := (\mathbf{x}_1^2 + \mathbf{x}_2^2, \mathbf{x}_1 - \mathbf{x}_2),$$

and the matrices $\mathbf{K}_1$ and $\mathbf{K}_2$ are parametrized as

$$e^{-\mu_j \Delta t} \begin{bmatrix} \cos(\omega_j \Delta t) & -\sin(\omega_j \Delta t) \\ \sin(\omega_j \Delta t) & \cos(\omega_j \Delta t) \end{bmatrix},$$

with $\mu_1 = \mu_2 = 0, \omega_1 = \frac{\pi}{100}, \omega_2 = \frac{\pi}{100\sqrt{10}}$ and $\Delta t = 1$.

The system is initialized as

$$(\mathbf{x}^0, \mathbf{p}^0) := \Phi^{-1}([0, 1]^\top, [1, 1]^\top).$$

Finally,

$$g(\mathbf{p}) := \text{softplus}(-\mathbf{p}_1 - \frac{3}{4}\mathbf{p}_2 + 1\frac{1}{2}).$$

As the $\mathbf{x}$ and $\mathbf{p}$-variables are 2-dimensional, we can visualize them separately (see Figure 2).

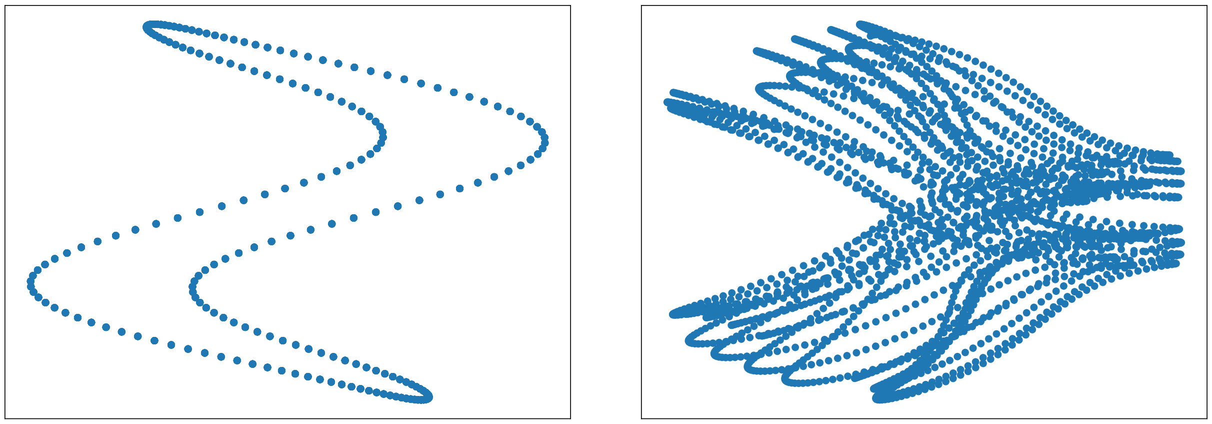

(a) Clean $\mathbf{x}$-variable data (b) Clean $\mathbf{p}$-variable data

Figure 2: A visualization of the synthetic data: whereas the $\mathbf{x}$-variable follows a periodic orbit, the latent $\mathbf{p}$-variable data are much more complicated due to the interplay with the memory of the system.

A.2 Real data

Streamflow observations ($\mathbf{v}$ variable) were provided by the United States Geological Survey (USGS). The example here is from Muddy Creek near Emery, Utah (USGS id 09330500). These data are available at https://waterdata.usgs.gov/monitoring-location/09330500/#dataTypeId=continuous-00065-0&period=P7D. Streamflow data are provided as daily averages in cubic feet per second. These data were normalized by the drainage area of the watershed of the gage, or 271.9 km^2 in this case, and converted to millimeters per day to arrive at $\mathbf{v}$.

The $\mathbf{x}$-variables were sampled over the watershed of the Muddy Creek gage by spatial averaging (or summing in the case of total precipitation). These variables include:

Variable Name	**Variable Meaning**
dewpoint_temperature_2m__mean__era5l_daily	Daily mean dewpoint temperature at 2 meters
potential_evaporation__sum__era5l_daily	Daily sum of potential evaporation
snow_depth_water_equivalent__mean__era5l_daily	Daily mean snow depth water equivalent
surface_net_solar_radiation__mean__era5l_daily	Daily mean surface net solar radiation
surface_net_thermal_radiation__mean__era5l_daily	Daily mean surface net thermal radiation
surface_pressure__mean__era5l_daily	Daily mean surface atmospheric pressure
temperature_2m__mean__era5l_daily	Daily mean air temperature at 2 meters
total_precipitation__sum__era5l_daily	Daily sum of total precipitation
u_component_of_wind_10m__mean__era5l_daily	Daily mean east-west wind component at 10 meters
v_component_of_wind_10m__mean__era5l_daily	Daily mean north-south wind component at 10 meters
volumetric_soil_water_layer_1__mean__era5l_daily	Daily mean volumetric soil water content, Layer 1 (0–7 cm)
volumetric_soil_water_layer_2__mean__era5l_daily	Daily mean volumetric soil water content, Layer 2 (7–28 cm)
volumetric_soil_water_layer_3__mean__era5l_daily	Daily mean volumetric soil water content, Layer 3 (28–100 cm)
volumetric_soil_water_layer_4__mean__era5l_daily	Daily mean volumetric soil water content, Layer 4 (100–289 cm)

B Training details

Based on our experience, both shallow and deep networks delivered strong performance, provided the activation functions were regular. For instance, non-regular ReLU activation performed poorly. We prioritized simplicity in our models, which primarily guided the selection of the parameters reported below.

Common parameters

- **Batch size**: 256
- **Optimizer**: Adam with `betas` = (0.9, 0.99) and learning rate 10^{-3}.
- **Model Architecture**:
  - **Diffeomorphisms**: Additive coupling layer [4], which adds the mapping of the sum of two adjacent parity-0 inputs through a learnable order-2 polynomials to corresponding parity-1 entries (i.e., the parity-1 entry in between the two parity-0 entries).
  - **Couplings**: Multi layer perceptron network with ℓ_f hidden layers with dimension m_f (different per data set) and learnable polynomial activation functions of order 2 (unique polynomial per neuron).
  - **Mappings**: Multi layer perceptron with one hidden layer with dimension m_g and softplus activation.

Data set-specific parameters The remaining parameters are summarized below:

Data set	ℓ_f	m_f	m_g	**Epochs**
Synthetic	2	2	4	1000
Real	1	40	4	200

C Additional numerical results

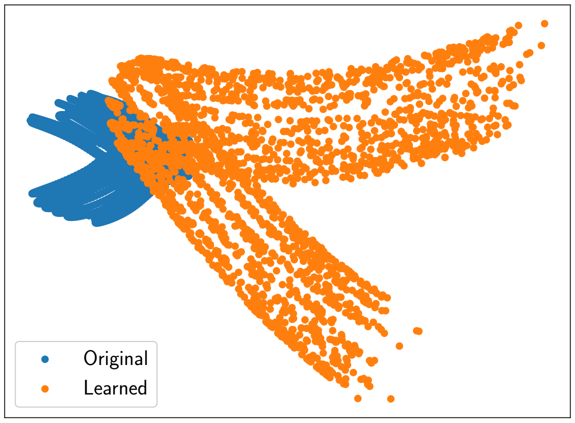
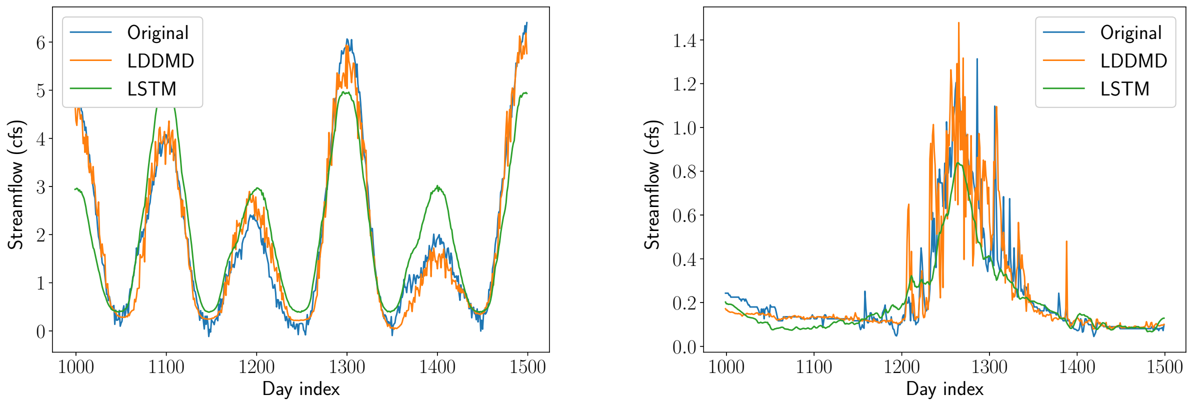


Figure 3: The ground truth and learned latent ($\mathbf{p}$-variable) dynamics look very similar, but differ roughly by a rigid body transformation and rescaling. This indicates that the latent space does carry information for the synthetic data, i.e., LDDMD has an interpretable latent space.



(a) Syntethic data results

(b) Real-world data results

Figure 4: When examining a specific interval in the plots in Figure 1, it is evident that the LSTM predictions are considerably less noisy than those of LDDMD. This difference is likely due to the inherent averaging in LSTMs, where predictions at each time step incorporate information from many preceding time points, effectively smoothing out noise. In contrast, LDDMD lacks a comparable denoising mechanism.