DATA-DRIVEN REDUCED ORDER MODELING OF ENVIRONMENTAL HYDRODYNAMICS USING DEEP AUTOENCODERS AND NEURAL ODES

SCIPEDIA

DATA-DRIVEN REDUCED ORDER MODELING OF ENVIRONMENTAL HYDRODYNAMICS USING DEEP AUTOENCODERS AND NEURAL ODES

Sourav Dutta, Peter Rivera-Casillas, Orie Cecil, Matthew Farthing, Emma Perracchione, Mario Putti

INFORMATION

Keywords:
Data-driven model order reduction
Autoencoder
Neural ordinary differential equations
Proper orthogonal decomposition
Dynamic mode decomposition
Radial basis function interpolation

DOI: 10.23967/coupled.2021.017

Published: 12/07/2021

IX International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2021
E. Oñate, M. Papadrakakis and B. Schrefler (Eds)

DATA-DRIVEN REDUCED ORDER MODELING OF ENVIRONMENTAL HYDRODYNAMICS USING DEEP AUTOENCODERS AND NEURAL ODES

SOURAV DUTTA[1†], PETER RIVERA-CASILLAS[1], ORIE M. CECIL[1], MATTHEW W. FARTHING[1], EMMA PERRACCHIONE[2], AND MARIO PUTTI[3]

[1] USACE Engineer Research Development Center, 3909 Halls Ferry Rd, Vicksburg MS 39180, USA,

[2] University of Genoa, Via Dodecaneso 35, 16146 Genova, Italy,

[3] University of Padua, Via Trieste, 63, 35131 Padova, Italy,

[†] email: sourav.dutta@erdc.dren.mil

Key words: Data-Driven Model Order Reduction, Autoencoder, Neural Ordinary Differential Equations, Proper Orthogonal Decomposition, Dynamic Mode Decomposition, Radial Basis Function Interpolation

Abstract. Model reduction for fluid flow simulation continues to be of great interest across a number of scientific and engineering fields. In a previous work [1], we explored the use of Neural Ordinary Differential Equations (NODE) as a non-intrusive method for propagating the latent-space dynamics in reduced order models. Here, we investigate employing deep autoencoders for discovering the reduced basis representation, the dynamics of which are then approximated by NODE. The ability of deep autoencoders to represent the latent-space is compared to the traditional proper orthogonal decomposition (POD) approach, again in conjunction with NODE for capturing the dynamics. Additionally, we compare their behavior with two classical non-intrusive methods based on POD and radial basis function interpolation as well as dynamic mode decomposition. The test problems we consider include incompressible flow around a cylinder as well as a real-world application of shallow water hydrodynamics in an estuarine system. Our findings indicate that deep autoencoders can leverage nonlinear manifold learning to achieve a highly efficient compression of spatial information and define a latent-space that appears to be more suitable for capturing the temporal dynamics through the NODE framework.

INTRODUCTION

The computational challenges faced during *high-fidelity* numerical simulations of engineering systems governed by nonlinear partial differential equations (PDEs), especially in appli-

cations involving control [2], optimal design and multi-fidelity optimization [3], can often be mitigated by the development of *reduced order models* (ROMs)[4].

Proper orthogonal decomposition (POD) [5, 6] is a well known method for extracting a solution-dependent reduced basis space from a set of well-resolved, high-fidelity snapshots, and is most effective when the coherent structures of the dataset can be ranked in terms of their energy content. The POD method has been successfully applied in statistics [7], signal analysis and pattern recognition [8], ocean models [9], air pollution models [10], convective Boussinesq flows [11], and Shallow Water Equation (SWE) models [12, 13]. Alternatively, nonlinear dimension reduction techniques such as kernel POD [14] or deep learning-based approaches like autoencoders [15, 16] have also been used for extracting a reduced basis. Combining autoencoder-generated bases with various specialized machine learning algorithms for time series modeling result in fully non-intrusive reduced order models [17, 18, 19]. Hybrid methods [20, 21] can also be obtained by combining a nonlinear manifold learning technique like autoencoder for discovering the latent space with an intrusive method for the temporal dynamics.

Following the identification of the latent space, a reduced representation of the dynamical system is obtained by a Galerkin or Petrov-Galerkin projection on to the latent space [13, 22], which typically involves *intrusive* modifications of the high-fidelity system operators. This work focuses on non-intrusive reduced order models (NIROMs) that do not require any knowledge of the high-fidelity simulator. In such a framework, the evolution of the expansion coefficients in the latent space is usually computed by the application of several regression-based methods directly on the high-fidelity data. These include artificial neural networks (ANNs), in particular multi-layer perceptrons [23], Gaussian process regression (GPR) [24], and radial basis function (RBF) [25] interpolation. RBF interpolation in particular has been shown to be quite successful for nonlinear, time-dependent partial differential equations (PDEs) [26], nonlinear, parametrized PDEs [25], and aerodynamic shape optimization [27].

Alternatively, in deep neural networks (DNN) such as ResNet, the evolution of features over the depth of the network is equivalent to solving an ordinary differential equation (ODE) of the form $\frac{dz}{dt} = F(z, \theta)$ with the forward Euler method [28]. With this connection in mind, [29] proposed a 'continuous-depth' neural network called ODE-Net that effectively replaced the layers in ResNet-like architectures with a trainable ODE solver. This neural ordinary differential equation approach (NODE) was further improved in [30, 31] and [32] proposed a NODE generative model that can be efficiently trained on large-scale datasets. Some applications of the NODE framework include latent space closure modeling [33], ODE/PDE model identification [34], modeling of irregularly spaced time series data [35], and modeling of spatio-temporal information in video signals [36].

Dynamic mode decomposition (DMD) is yet another method for obtaining a reduced order model. DMD represents the temporal dynamics of a complex, nonlinear system [37, 38] as the combination of a few linearly evolving, spatially coherent modes that oscillate at a fixed frequency, and which are closely related to the eigenvectors of the infinite-dimensional Koopman operator [39]. Several variants of the DMD algorithm have been proposed [2, 40, 41, 42] and

have been successfully applied as efficient ROM techniques for determining the optimal global basis modes for nonlinear, time-dependent problems [43, 44]. For non-parametrized PDEs, DMD presents an efficient framework that combines all the three stages of ROM development to learn a linear operator in an optimal least square sense. However, this approach cannot be directly applied to parametrized problems [45].

In [1], we explored propagating the dynamics of a latent space formed from POD modes with neural ODEs. The present work investigates substituting the latent-space described by POD modes with one learned by an autoencoder. Our results for the combined autoencoder-NODE approach are compared to other methods like - a) dimension reduction by POD modes with latent space temporal dynamics captured by Neural ODEs (POD-NODE), b) dimension reduction via POD and temporal evolution of the latent space with Radial Basis Functions (POD-RBF), and c) Dynamic Mode Decomposition (DMD), which serve as benchmarks in our numerical experiments. The performance of each approach will be evaluated on sample problems based on incompressible flow around a cylinder and shallow water hydrodynamics in the context of fast replay applications for complex fluid-dynamics problems.

METHODOLOGY

The standard ROM development framework can be divided into three stages:

1. identification of a low-dimensional latent (or reduced-order) space,
2. representation of the nonlinear dynamical system in terms of the reduced basis and modeling the evolution of the system of modal coefficients, and
3. reconstruction in the high-fidelity space for validation and analysis.

Dimension reduction

In this work, the dominant features of the unsteady flow-field have been extracted using a linear modal decomposition technique, POD, and a nonlinear manifold learning method that relies on deep, fully-connected, multi-layer perceptron (MLP) autoencoders that are highly expressive and scalable [46]. POD is a popular technique for dimension reduction of the solution manifold of a dynamical system by determining a linear reduced space spanned by an orthogonal basis with an associated energetic hierarchy, and which represents an optimal approximation of the solution manifold with respect to the L^2-norm. Given a matrix of high-fidelity system snapshots $\mathbf{S} \in \mathbb{R}^{N \times M}$ and a matrix of orthonormal POD basis vectors $\boldsymbol{\theta}$, the modal coefficient matrix $\mathbf{Z} = \boldsymbol{\Theta}^T \mathbf{S}$ constitutes our training data for the latent space learning methods. [47] provides an excellent overview of POD as well as a comparison with other dimension-reduction techniques.

Autoencoders

An autoencoder is a type of feedforward neural network that is designed to learn the identity mapping, $\mathbf{h} : \mathbf{v} \mapsto \widetilde{\mathbf{v}}$ such that $\widetilde{\mathbf{v}} \approx \mathbf{v}$ and $\mathbf{h} : \mathbb{R}^N \mapsto \mathbb{R}^N$. This is accomplished using a two-part architecture. The first part is called an encoder, $\mathbf{h}_E$, defined by $\mathbf{z} = \mathbf{h}_E(\mathbf{v}; \boldsymbol{\theta}_E)$ where $\mathbf{z} \in \mathbb{R}^m$ ($m \ll N$), which maps a high-dimensional input vector $\mathbf{v}$ to a low-dimensional latent vector $\mathbf{z}$. The second part is called a decoder, $\mathbf{h}_D$, defined as $\widetilde{\mathbf{v}} = \mathbf{h}_D(\mathbf{z}; \boldsymbol{\theta}_D)$, which maps the latent vector $\mathbf{z}$ to an approximation $\widetilde{\mathbf{v}}$ of the high-dimensional input vector $\mathbf{v}$. The combination of the two parts yields an autoencoder of the form

$$\mathbf{h} : \mathbf{v} \mapsto \mathbf{h}_D \circ \mathbf{h}_E(\mathbf{v}). \tag{1}$$

This autoencoder is trained by computing the optimal values of the parameters $(\boldsymbol{\theta}_E^*, \boldsymbol{\theta}_D^*)$ that minimize the reconstruction error over all the training data

$$\boldsymbol{\theta}_E^*,\ \boldsymbol{\theta}_D^* = \underset{\boldsymbol{\theta}_E, \boldsymbol{\theta}_D}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{v}, \widetilde{\mathbf{v}}), \tag{2}$$

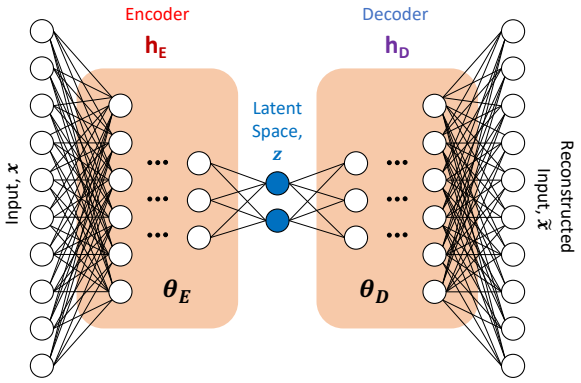


Figure 1: The architecture of the autoencoder network

where $\mathcal{L}(\mathbf{v}, \widetilde{\mathbf{v}})$ is a chosen measure of discrepancy between $\mathbf{v}$ and its approximation $\widetilde{\mathbf{v}}$. The restriction $\dim(\mathbf{z}) = m \ll N = \dim(\mathbf{v})$ forces the autoencoder model to learn the salient features of the input data via compression into a low-dimensional space and then reconstructing the input, instead of directly learning the identity function. It is worth noting that with the choice of a linear, single-layer encoder of the form $\mathbf{z} = H_E\mathbf{v}$, and a linear, single-layer decoder of the form $\widetilde{\mathbf{v}} = \mathbf{H}_D\mathbf{z}$, where $\mathbf{H}_E \in \mathbb{R}^{m \times N}$, $\mathbf{H}_D \in \mathbb{R}^{N \times m}$, and a squared reconstruction error as the loss function $\mathcal{L}(\widetilde{\mathbf{v}}, \mathbf{v}) = \|\mathbf{v} - \widetilde{\mathbf{v}}\|_2^2$, the autoencoder model has been shown to learn the same subspace as that spanned by the first m POD modes if $\mathbf{H} = \mathbf{H}_E = \mathbf{H}_D$. However, additional constraints are necessary to ensure that the columns of $\mathbf{H}$ form an orthonormal basis and follow an energy-based hierarchical ordering [48].

In this work, autoencoders are employed to generate separate low-dimensional latent representations of the pressure (depth) and the velocity snapshot data on the computational grid points obtained from the high-fidelity simulation of the numerical examples. The encoder and decoder neural networks are constructed using fully-connected MLP architectures, as depicted in Figure 1. As the high-fidelity simulation data is usually available on a two-dimensional spatial grid, the data is first flattened and then fed to the autoencoder model.

Latent space evolution

In this section, we outline the non-intrusive framework for modeling the evolution of time-series data in the latent space. RBF interpolation is a classical, data-driven, kernel-based method

for computing an approximate continuous response surface that aligns with the given multivariate data. More details about the POD-RBF NIROM framework can be found in [26]. The second technique called NODE is a neural-network based method to predict the continuous evolution of a vector $\mathbf{c}$ over time, that is designed to preserve memory effects within the architecture.

Neural ordinary differential equations

We assume that the time evolution of the modal coefficients of the high-fidelity dynamical system in the latent space can be modeled using a (first-order) ODE,

$$\frac{d\,\mathbf{z}}{dt} = \mathcal{F}(t, \mathbf{z}(t)), \text{ with } \mathbf{z}(0) = \mathbf{z}^0, \ \mathbf{z} \in \mathbb{R}^d, d \geq 1. \tag{3}$$

The goal is to obtain a NN approximation $\widehat{\mathcal{F}}$ of the dynamics function $\mathcal{F}$ such that $\frac{d\mathbf{z}}{dt} \approx \text{net}(t, \mathbf{z}) = \widehat{\mathcal{F}}(t, \mathbf{z}, \boldsymbol{\omega})$. The full procedure can be outlined as follows:

1. Compute the time series of modal coefficients $[\mathbf{z}^0, \ldots, \mathbf{z}^{M-1}]$ for $t \in \{0, \ldots, M-1\}$ where $\mathbf{z}^k \in \mathbb{R}^m$.

2. Initialize a NN approximation for the dynamics function $\widehat{\mathcal{F}}(t, \mathbf{z}, \boldsymbol{\omega})$ where $\boldsymbol{\omega}$ represents the initial NN parameters.

3. The NN parameters are optimized iteratively through the following steps.

   (a) Compute the approximate forward time trajectory of the modal coefficients by solving eq. (3) using a standard ODE integrator as,

   $$\hat{\mathbf{z}}^{M-1} = ODESolve(\widehat{\mathcal{F}}, \boldsymbol{\omega}, \mathbf{z}^0, t^0, t^{M-1}) \tag{4}$$

   (b) The free parameters of the NODE model are $\{\boldsymbol{\omega}, t^0, t^{M-1}\}$. Evaluate the differentiable loss function $\mathcal{L}\left(ODESolve(\widehat{\mathcal{F}}, \boldsymbol{\omega}, \mathbf{z}^0, t^0, t^{M-1}) - \mathbf{z}^{M-1}\right)$.

   (c) To optimise the loss, compute gradients with respect to the free parameters. Similar to the usual backpropagation algorithm, this can be achieved by first computing the gradient $\partial\mathcal{L}/\partial\widehat{\mathbf{z}}(t)$, and then a performing a reverse traversal through the intermediate states of the ODE integrator. For a memory-efficient implementation, the adjoint method [29] can be used to backpropagate the errors by solving an adjoint system for the augmented state vector $\mathbf{b} = [\frac{\partial\mathcal{L}}{\partial\widehat{\mathbf{z}}}, \frac{\partial\mathcal{L}}{\partial\boldsymbol{\omega}}, \frac{\partial\mathcal{L}}{\partial t}]^T$ backwards in time from t^{M-1} to t^0.

   (d) The gradient $\frac{\partial\mathcal{L}}{\partial\boldsymbol{\omega}}(t=0)$ computed in the previous step is used to update the parameters $\boldsymbol{\omega}$ by using an optimization algorithm like RMSProp or Adam.

4. The trained NODE approximation of the dynamics function can be used to compute predictions for the time trajectory of the modal coefficients.

Following [1], we adopt the TFDiffEq (`https://github.com/titu1994/tfdiffeq`) library that runs on the Tensorflow Eager Execution platform to train the NODE models. RMSProp is adopted for loss minimization with an initial learning rate of 0.001, a staircase decay function with a range of variable decay schedules, and a momentum coefficient of 0.9.

As a final point of comparison, we consider the standard Dynamic mode decomposition (DMD)[37, 38] algorithm, which is a powerful data-driven method capable of providing an accurate decomposition of a complex system into spatiotemporal coherent structures that may be used for short-time future-state prediction.

NUMERICAL EXPERIMENTS

In this section, we first assess the use of autoencoders for building a reduced space in which the system dynamics are propagated by NODE for a benchmark flow problem characterized by the incompressible Navier Stokes equations (NSE). We compare the performance of this framework with a POD-NODE method where NODE is employed to capture the evolution of modal coefficients in a reduced space defined by a POD basis. Also, we examine the relative performance of different NIROM models for a real-world estuarine flow application governed by the shallow water equations (SWE). The POD-RBF and DMD NIROM training runs were performed on a Macbook Pro 2018 with a 2.9 GHz 6-Core Intel Core i9 processor and 32 GB 2400 MHz DDR4 RAM. The autoencoder latent-space representations were trained on Vulcanite, a high performance computer at the U.S. Army Engineer Research and Development Center DoD Supercomputing Resource Center (ERDC-DSRC). Vulcanite is equipped with NVIDIA Tesla V100 PCIe GPU accelerator nodes and has 32GBytes memory/node. Training for the NODE models was performed in serial on Jim, a high performance computer at the U.S. Army Engineer Research and Development Center Coastal and Hydraulics Lab (CHL), which is equipped with 2 Intel Xeon E5-2699 v3 CPUs and 128Gbytes of memory/node.

Flow around a cylinder

The problem of two-dimensional, incompressible flow arond a cylinder is a classical benchmark CFD example that simulates a time-periodic fluid flow through a 2D pipe with a circular obstacle. For further details about the problem setup please see [1]. High-fidelity simulation data is obtained with OpenFOAM using an unstructured mesh with 14605 nodes at $Re = 100$, such that the flow exhibits the periodic shedding of von Karman vertices. 313 training snapshots are collected for $t = [2.5, 5.0]$ seconds with $\Delta t = 0.008$ seconds, and the NIROM predictions are obtained for $t = [2.5, 6.0]$ seconds with $\Delta t = 0.002$ seconds.

Several different architectures were explored for the autoencoder model by varying the model parameters like the dimension of the latent space, the activation functions used, various configurations of the learning rate evolution during training, and Table 1 shows four of the best architectures for the cylinder flow example. The second column shows the size of the latent space for each of the solution variables p, v_x, v_y. The third and fourth columns list the activation functions used in the last hidden encoder and the last hidden decoder layers, respectively.

Id	Units	Encoder	Decoder	Scaling	MSE	Training
Range	(p, v_x, v_y) 2-8	linear, relu, ...	elu, tanh, ...			
AE1	$(5, 8, 7)$	linear	sigmoid	$[0, 1]$	2.398e-6	9.38 min
AE2	$(2, 2, 2)$	linear	sigmoid	$[0, 1]$	4.196e-6	9.56 min
AE3	$(2, 2, 2)$	linear	tanh	$[-1, 1]$	2.499e-6	9.29 min
AE4	$(2, 3, 3)$	linear	sigmoid	$[0, 1]$	3.107e-6	9.07 min

Table 1: Best Autoencoder architectures for the cylinder example. Models were trained for 1000 epochs using the Adam optimizer with initial learning rate = 1e-3 and momentum = 0.9.

The fifth column shows the scaling applied to the input data which is directly determined by the activation function used in the decoder output layer. In general, it was found that activation functions that required scaled input data like *sigmoid*, *tanh* performed better for the decoder output layer than some of the (semi-)unbounded ones like *linear*, *relu*, and *elu*. However, unbounded activation functions like *linear* and *elu* were seen to generate a more accurate and efficient latent space. The encoder and the decoder networks were made up of four hidden layers with gradually decreasing and gradually increasing size, respectively, and characterized by the *relu* activation function, which were found to generate the least noisy latent representations. The *Adam* optimizer is used for training with an initial learning rate=10^{-3} and momentum=0.9. An adaptive learning rate decay algorithm is employed that monitors the training loss and reduces the learning rate by a factor of 2 if no improvement is detected for 200 epochs. The sixth column of Table 1 lists the total mean square reconstruction error for all three solution variables, while the last column shows the training times for each model on two NVIDIA Tesla V100

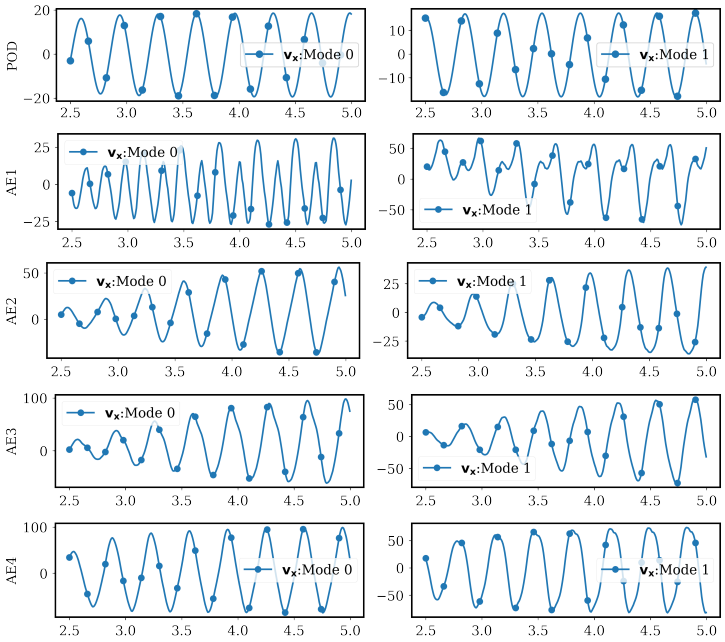


Figure 2: Visualization of the modal coefficients of the first two latent space modes for x-velocity, as generated by POD and the four chosen autoencoder models for the cylinder example

GPU nodes.

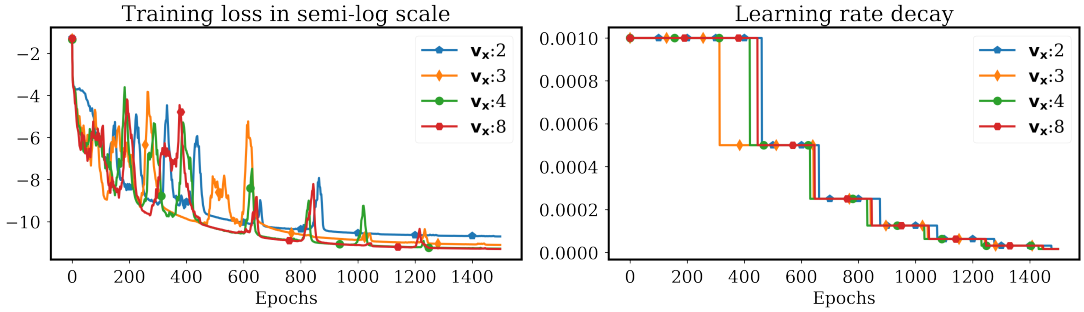


Figure 3: Training characteristics for a fixed autoencoder architecture while varying the latent space dimension

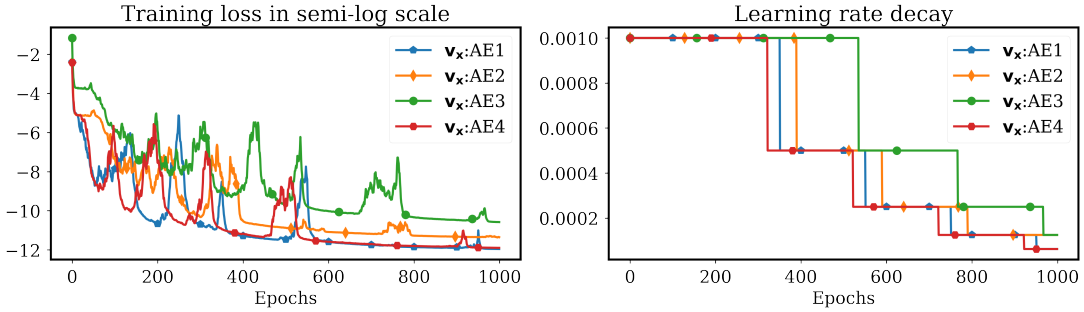


Figure 4: Training characteristics of the chosen four autoencoder architectures (see Table 1) over 1000 epochs

Figure 2 shows the temporal coefficients for the first two latent modes of v_x using a POD truncation that captures 99% of modal energy content, and the four chosen AE models. While the POD modal coefficients are arranged according to the decreasing order of amplitude, this cannot be guaranteed for the AE-generated spaces. AE2 and AE3 models define a latent space of dimension 2, whereas the AE4 model has 3 latent space units and the AE1 model is identical to the dimension of the POD bases for each variable - 5 (p), 8 (v_x), and 7(v_y). The richer quality of the AE4 latent space leads to better expressivity which is reflected in the subtler features captured in the modal coefficients (Figure 2 row 2) and also in the lowest reconstruction error (Table 1).

Figure 3 shows the evolution of the training loss and the adaptive decay of the learning rate while training autoencoder models for v_x using four gradually increasing latent space dimensions - $2, 3, 4, 8$. The AE3 architecture was adopted for these runs. The optimization of the hyperparameters with respect to the training loss becomes gradually harder as the dimension of the latent space increases. So the models with smaller latent spaces initially show a faster reduction in training losses. However, the enhanced expressivity of the models with higher latent

Id	Lyrs	Units	Act.	LR steps, rate	Scaling	Aug.	MSE	Training
Range	1-4	32-512	tanh, elu,...	5k-25k, 0.1-0.9				
AE1-NODE1	1	256	elu	10k, 0.3	No	No	2.30e-5	28.80 hrs
AE1-NODE2	1	256	tanh	5k, 0.7	Yes	No	1.34e-4	28.69 hrs
AE1-NODE3	1	512	elu	5k, 0.5	No	No	1.97e-5	29.17 hrs
AE1-NODE4	1	256	tanh	10k, 0.25	Yes	Yes	1.49e-4	28.27 hrs
AE1-NODE5	4	64	tanh	5k, 0.5	Yes	No	1.33e-4	33.08 hrs

Table 2: Results for the best NODE architectures for the cylinder example using the latent space defined by the AE1 autoencoder model. All NODE models were trained for 50000 epochs using the fourth-order Runge-Kutta solver and the RMSProp optimizer with initial learning rate = 1e-3 and momentum = 0.9.

dimensions allow them to reach lower values of training loss after sufficient epochs of training, whereas the losses for the models with smaller latent dimensions appear to stagnate. Figure 4 shows the training loss and the learning rate decay for the four chosen AE models. It is evident that the AE4 model with the richest latent spaces achieves the sharpest reduction in training loss, followed by AE4, whereas the latent space dimensions for models AE1 and AE2 appear to be significant barriers in their training efficiency.

An extensive exploration of the NODE hyperparameter and architecture space for the cylinder example was reported in [1]. Based on those inferences and some further numerical study, five architectures were selected, and the results after training for 50000 epochs using the latent space data generated by the AE1 model are presented in Table 2. All the models generate accurate predictions at a finer temporal resolution than the training data, and have excellent agreement with the high-fidelity solution even while extrapolating outside the training data ($5 \leq t \leq 6$ seconds).

The first row of Fig. 5 compares the time trajectory of the spatial root mean square errors (RMSE) in the high-fidelity space for NIROM solutions generated using the POD and the AE1 basis with the NODE3 and NODE5 configurations. It is encouraging to note that even though the latent-space sizes are identical between the POD and the AE1 basis, the AE1-NODE solutions are more accurate, confirming that the nonlinear AE basis generates a more accurate spatial compression than a linear POD basis of similar size. (IF space allows add reconstruction error values here). The second row of Fig. 5 compares the RMSE between the AE4-NODE3, POD-NODE3, and two DMD NIROM models with truncation levels of $r = 3$ and $r = 8$. The AE4-NODE3 model achieves better accuracy than all the other models even with the most highly compressed latent space (p, v_x, v_y:2,3,3). The DMD(8) model is roughly comparable with the POD-NODE3 model owing to the similarity in their latent space dimensions, whereas the DMD(3) model fares the worst due to the lack of hierarchical ordering in its latent basis.

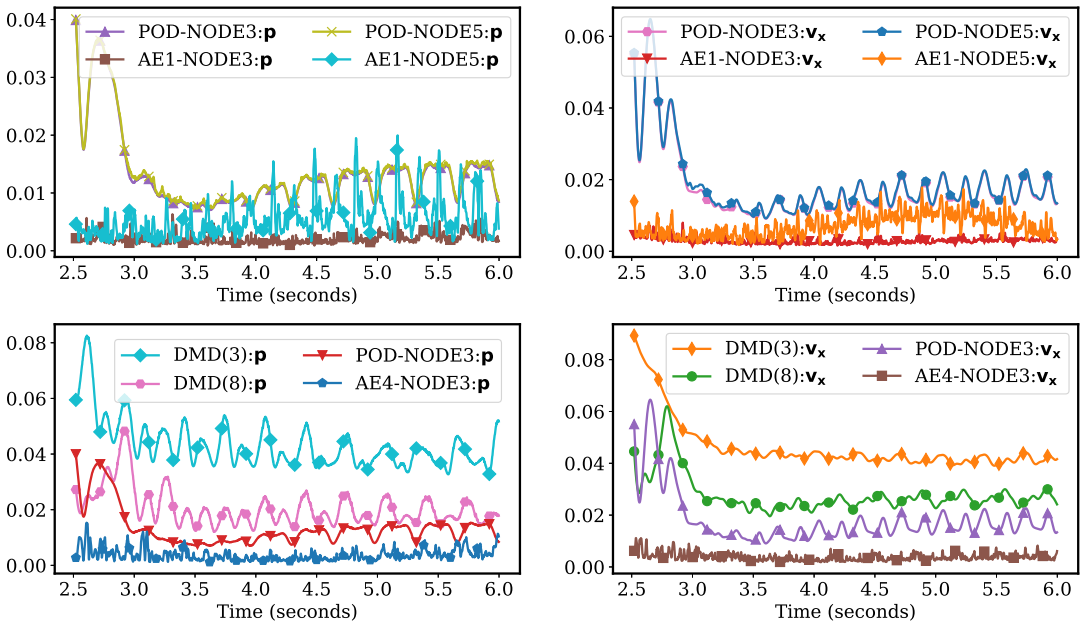


Figure 5: Row1 - Comparison of RMSE of the NODE3 and the NODE5 models using POD and the AE1 basis; Row2 - Comparison of RMSE of the POD-NODE and the AE-NODE models with two DMD models with comparable latent space dimensions for the cylinder example

Shallow water equations

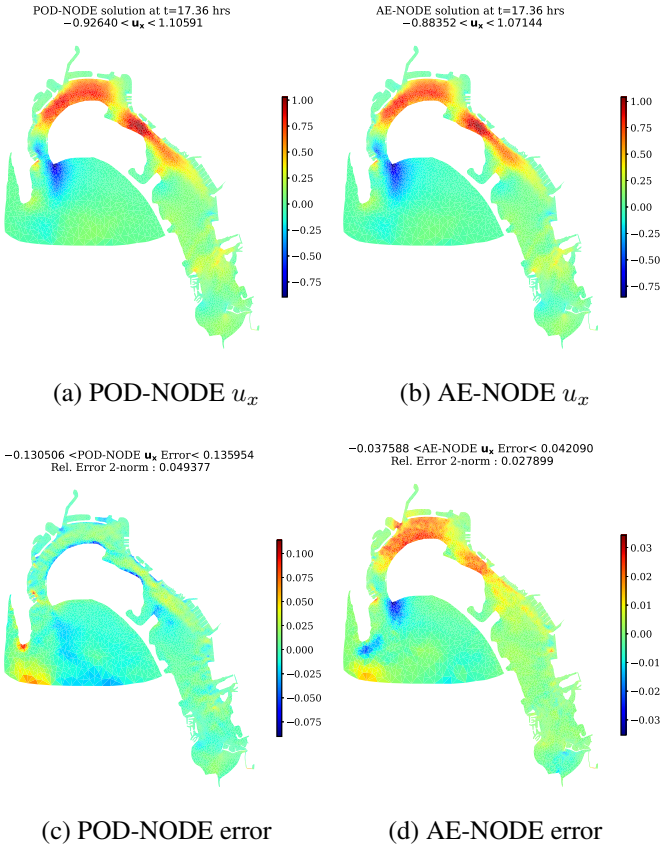
The final numerical example involves flows governed by the depth-averaged SWE which is written in a conservative residual formulation as

$$\mathfrak{R} \equiv \frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{p}_x}{\partial x} + \frac{\partial \mathbf{p}_y}{\partial y} + \mathbf{r} = 0, \tag{5}$$

where the state variable $\mathbf{q} = [h, u_x h, u_y h]^T$ consists of the flow depth, h, and the discharges in the x and y directions, given by $u_x h$ and $u_y h$, respectively. Further details about the flux vectors $\mathbf{p}_x$, $\mathbf{p}_y$ and the high-fidelity model equations are available in [26]. The high-fidelity numerical solutions of the SWE are obtained using the 2D depth-averaged module of the Adaptive Hydraulics (AdH) finite element suite, which is a U.S. Army Corps of Engineers (USACE) high-fidelity, finite element resource for 2D and 3D dynamics [49].

Tidal flow in San Diego bay

This numerical example involves the simulation of tide-driven flow in the San Diego Bay in California, USA. The AdH high-fidelity model consists of $N = 6311$ nodes, uses tidal data obtained from NOAA/NOS Co-Ops website at a tailwater elevation inflow boundary and has no flow boundary conditions everywhere else. Further details are available in [26].



(a) POD-NODE u_x (b) AE-NODE u_x

(c) POD-NODE error (d) AE-NODE error

Figure 6: NIROM solutions of u_x and errors at $t = 17.36$ hours for the San Diego example

The training space is generated using 1801 high-fidelity snapshots obtained between $t = 41$ minutes to $t = 50$ hours at a time interval of $\Delta t = 100$ seconds. The predicted ROM solutions are computed for the same time interval with $\Delta t = 50$ seconds. A latent space of dimension $265 (p, u_x, u_y : 36, 115, 113)$ is generated by using a POD truncation tolerance of $\tau_{POD} = 5 \times 10^{-7}$ for each solution component. The AE2 architecture designed for the cylinder example is modified by including *BatchNormalization* layers for each hidden layer, using full batches for training, and by enforcing a latent space of dimension 30 for each solution component. The RBF NIROM approximation is computed using a shape factor, $c = 0.01$. The simulation time points provided as input to the NODE model are normalized to lie in $t \in [0, 1]$. The 'dopri5' ODE solver is adopted for computing the hidden states both forward and backward in time. Learning from the conclusions of the cylinder example, a network consisting of a single hidden layer with 256 neurons is deployed and the RMSProp optimizer with an initial learning rate of 0.001, a staircase decay rate of 0.5 every 5000 epochs, and a momentum of 0.9 is utilized for training the model over 20000 epochs. For the DMD NIROM, the simulation time points are normalized to an unit time step, and a truncation level of $r = 115$ is used to compute the DMD eigen-spectrum.

Figure 6 compares the POD-NODE and AE-NODE NIROM solution fields (top row) for u_x at $t = 17.36$ hours and the corresponding error plots (bottom row). It can be seen that even while using a latent dimension that is three times smaller than POD, the relative errors for the AE-NODE solution (0.0279) is almost two times lower than that of the POD-NODE solution (0.0494). Figure 7 shows the spatial RMSE over time for the depth (left) and the x-velocity (right) NIROM solutions. The AE-NODE $(30$ modes$)$ solution has comparable accuracy to the POD-NODE $(36$ modes$)$ and the DMD $(115$ modes$)$ solution for the depth variable and actually outperforms the POD-NODE $(115$ modes$)$ solution for the x-velocity variable. Additionally,

unlike the RBF NIROM solution, the AE-NODE solution does not exhibit any appreciable accumulation of error over time due to the higher-order time-stepping scheme adopted for NODE.

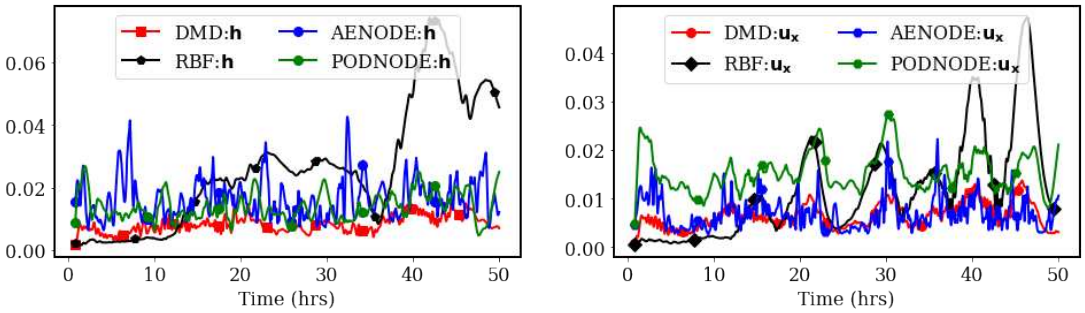


Figure 7: NIROM RMSEs for the San Diego example

CONCLUSION

We have studied the combined use of autoencoders as a data-driven method for identifying an efficient reduced latent space that approximates the solution manifold of a system of nonlinear, time-dependent PDEs and neural ODEs as a non-intrusive method for modeling the evolution of the reduced latent space coefficients for the aforementioned dynamical system. Numerical experiments were carried out with a benchmark periodic flow problem governed by the incompressible Navier Stokes equations and a real-world application of estuarine flow dynamics governed by the two-dimensional shallow water equations. The AE latent space representation was shown to provide a very high degree of efficient spatial compression, especially for the advection-dominated shallow water example. The POD-NODE NIROM formulation demonstrated a stable and accurate learning trajectory in modeling reduced basis dynamics, even in comparison to classical ROM techniques utilizing DMD, POD and RBF interpolation. The AE-NODE formulation also produced encouraging extrapolatory predictions for the flow around a cylinder example. This presents an exciting prospect for future exploration as even for an isolated system, unperturbed by unseen external forcings, truly extrapolative predictions of reduced order dynamics in flow regimes that do not correspond to the training data is a rare feature for most well-established ROM frameworks.

This study leads to several promising avenues of research. For instance, neural architecture search (NAS) tools can be adopted to perform an exhaustive exploration of the network architecture and model hyperparameter space for a wide range of flow dynamics in order to gain insight of the learning trajectory and to design more generalizable AE models with improved reconstruction accuracy. Moreover, integrating the process of identification of an AE-based latent space with the modeling of system dynamics using NODE may lead to significant performance improvements if the two independent learning problems can be designed to intelligently

inform each other. All the relevant data and codes for this study will be made available in a public repository at `https://github.com/erdc/aenode_nirom` upon publication.

ACKNOWLEDGMENTS

This research was supported in part by an appointment of the first author to the Postgraduate Research Participation Program at the U.S. Army Engineer Research and Development Center, Coastal and Hydraulics Laboratory (ERDC-CHL) administered by the Oak Ridge Institute for Science and Education through an interagency agreement between the U.S. Department of Energy and ERDC. Permission was granted by the Chief of Engineers to publish this information.

REFERENCES

[1] Sourav Dutta, Peter Rivera-Casillas, and Matthew W Farthing. Neural Ordinary Differential Equations for Data-Driven Reduced Order Modeling of Environmental Hydrodynamics. In *Proc. AAAI 2021 Spring Symp. Comb. Artif. Intell. Mach. Learn. with Phys. Sci.*, Stanford, CA, USA, 2021. CEUR-WS, cs.LG/2104.13962.

[2] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.*, 15(1):142–161, 2016.

[3] Benjamin Peherstorfer, Karen Willcox, and M A X Gunzburger. Optimal model management for multifidelity Monte Carlo estimation. *SIAM J. Sci. Comput.*, 38(5):A3163–A3194, 2016.

[4] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.*, 57(4):483–531, 2015.

[5] L. Sirovich. Turbulence and the dynamics of coherent structures. Part I:Coherent structures. *Quart. Appl. Math.*, 45:561–571, 1987.

[6] G. Berkooz, P. Holmes, and J.L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rev. Fluid Mech.*, 25(1):539–575, 1993.

[7] I.T. Jolliffe. *Principal Component Analysis*. Springer New York, USA, 1986.

[8] P. Deheuvels and G.V. Martynov. A Karhunen-Loeve decomposition of a Gaussian process generated by independent pairs of exponential random variables. *J. Func. Anal.*, 255(9):2363–2394, 2008.

[9] P.T.M. Vermeulen and A.W. Heemink. Model-reduced variational data assimilation. *Monthly Weather Review*, 134:2888–2899, 2006.

[10] F. Fang, T. Zhang, D. Pavlidis, C.C. Pain, A.G. Buchanan, and I.M. Navon. Reduced order modelling of an unstructured mesh air pollution model and application in 2D/3D urban street canyons. *Atmos. Env.*, 96:96–106, 2014.

[11] O. San and J. Borggaard. Principal interval decomposition framework for POD reduced-order modeling of convective Boussinesq flow. *Int. J. Numer. Methods Fluids*, 78(1):37–62, 2015.

[12] R. Stefanescu, A. Sandu, and I.M. Navon. Comparison of POD reduced order strategies for the nonlinear 2D shallow water equations. *Int. J. Numer. Methods Fluids*, 76(8):497–521, 2014.

[13] A. Lozovskiy, M.W. Farthing, C.E. Kees, and E. Gildin. POD-based model reduction for stabilized finite element approximations of shallow water flows. *J. Comput. Appl. Math.*, 302:50–70, 2016.

[14] Matteo Salvador, Luca Dede, and Andrea Manzoni. Non Instrusive Reduced Order Modeling of Parameterized PDEs by Kernel POD and Neural Networks, 2021, math.NA/2103.17152.

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

[16] Hojat Ghorbanidehno, Jonghyun Lee, Matthew Farthing, Tyler Hesser, Eric F. Darve, and Peter K. Kitanidis. Deep learning technique for fast inference of large-scale riverine bathymetry. *Advances in Water Resources*, 147:103715, 2021.

[17] Francisco J. Gonzalez and Maciej Balajewicz. Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems, 2018, 1808.01346.

[18] Hamidreza Eivazi, Hadi Veisi, Mohammad Hossein Naderi, and Vahid Esfahanian. Deep neural networks for nonlinear model order reduction of unsteady flows. *Phys. Fluids*, 32(10):1–19, 2020.

[19] Romit Maulik, Bethany Lusch, and Prasanna Balaprakash. Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Phys. Fluids*, 33(3), 2021.

[20] Kookjin Lee and Kevin T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *J. Comput. Phys.*, 404:108973, 2020, 1812.08373.

[21] Youngkyu Kim, Youngsoo Choi, David Widemann, and Tarek Zohdi. A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder, 2020, math.NA/2009.11990.

[22] A. Lozovskiy, M. Farthing, and C. Kees. Evaluation of Galerkin and Petrov-Galerkin model reduction for finite element approximations of the shallow water equations. *Comput. Methods Appli. Mech. Eng.*, 318:537–571, 2017.

[23] J.S. Hesthaven and S. Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *J. Comput. Phys.*, 363:55–78, 2018.

[24] M. Guo and J.S. Hesthaven. Data-driven reduced order modeling for time-dependent problems. *Comput. Methods Appl. Mech. Eng.*, 345:75–99, 2019.

[25] C. Audouze, F. De Vuyst, and P.B. Nair. Nonintrusive Reduced-Order Modeling of Parametrized Time-Dependent Partial Differential Equations. *Numer. Methods Partial Differ. Equation*, 29(5):1587–1628, 2013.

[26] Sourav Dutta, Matthew W. Farthing, Emma Perracchione, Gaurav Savant, and Mario Putti. A greedy non-intrusive reduced order model for shallow water equations. *J. Comput. Phys.*, page 110378, 2021.

[27] E. Iuliano and D. Quagliarella. Aerodynamic shape optimization via non-intrusive POD-based surrogate modelling. In *2013 IEEE Congr. Evol. Comput. CEC 2013*, pages 1467–1474. IEEE, 2013.

[28] Lars Ruthotto and Eldad Haber. Deep Neural Networks Motivated by Partial Differential Equations. *J. Math. Imaging Vis.*, pages 2–10, 2019.

[29] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations, 2018, 1806.07366.

[30] Amir Gholami, Kurt Keutzer, and George Biros. Anode: Unconditionally accurate memory-efficient gradients for neural odes, 2019, cs.LG/1902.10298.

[31] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented Neural ODEs, 2019, stat.ML/1904.01681.

[32] Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M Oberman. How to train your neural ODE: the world of Jacobian and kinetic regularization, 2020, stat.ML/2002.02798.

[33] Romit Maulik, Arvind Mohan, Bethany Lusch, Sandeep Madireddy, Prasanna Balaprakash, and Daniel Livescu. Time-series learning of latent-space dynamics for reduced-order model closure. *Phys. D Nonlinear Phenom.*, 405:132368, 2020.

[34] Yifan Sun, Linan Zhang, and Hayden Schaeffer. NeuPDE: Neural Network Based Ordinary and Partial Differential Equations for Modeling Time-Dependent Data. *Proc. Mach. Learn. Res.*, 107(2016):352–372, 2020.

[35] Yulia Rubanova, Ricky T. Q. Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 5320–5330. Curran Associates, Inc., 2019.

[36] David Kanaa, Vikram Voleti, Samira Kahou, and Christopher Pal. Simple Video Generation using Neural ODEs. In *Annu. Conf. Neural Inf. Process. Syst. 2019, NeurIPS 2019*, Vancouver, BC, Canada, 2019.

[37] P.J. Schmid. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.*, 656(July 2010):5–28, 2010.

[38] J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 2016.

[39] B O Koopman. Hamiltonian Systems and Transformation in Hilbert Space. *Proc. Natl. Acad. Sci. U. S. A.*, 17(5):315–8, 1931.

[40] J. Nathan Kutz, Xing Fu, and Steven L. Brunton. Multiresolution dynamic mode decomposition. *SIAM J. Appl. Dyn. Syst.*, 15(2):713–735, 2016.

[41] A.K. Alekseev, D.A. Bistrian, A.E. Bondarev, and I.M. Navon. On linear and nonlinear aspects of dynamic mode decomposition. *Int. J. Numer. Methods Fluids*, 82(6):348–371, 2016.

[42] Soledad Le Clainche and Jose M. Vega. Higher order dynamic mode decomposition. *SIAM J. Appl. Dyn. Syst.*, 16(2):882–925, 2017.

[43] D.A. Bistrian and I.M. Navon. An improved algorithm for the shallow water equations model reduction: Dynamic Mode Decomposition vs POD. *Int. J. Numer. Methods Fluids*, 2015.

[44] D.A. Bistrian and I.M. Navon. Randomized dynamic mode decomposition for nonintrusive reduced order modelling. *Int. J. Numer. Methods Eng.*, 112:3–25, 2017.

[45] Fahad Alsayyari, Zoltán Perkó, Marco Tiberga, Jan Leen Kloosterman, and Danny Lathouwers. A fully adaptive nonintrusive reduced-order modelling approach for parametrized time-dependent problems. *Comput. Methods Appl. Mech. Eng.*, 373:113483, 2021.

[46] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313:504–507, 2006.

[47] Kunihiko Taira, Maziar S. Hemati, Steven L. Brunton, Yiyang Sun, Karthik Duraisamy, Shervin Bagheri, Scott T.M. Dawson, and Chi An Yeh. Modal analysis of fluid flows: Applications and outlook. *AIAA J.*, 58(3):998–1022, 2020.

[48] Elad Plaut. From Principal Subspaces to Principal Components with Linear Autoencoders, 2018, stat.ML/1804.10253.

[49] C.J. Trahan, G. Savant, R.C. Berger, M. Farthing, T.O. McAlpin, L. Pettey, G.K. Choudhary, and C.N. Dawson. Formulation and application of the adaptive hydraulics three-dimensional shallow water and transport models. *J. Comput. Phys.*, 374:47–90, 2018.