Time-series learning of latent-space dynamics for reduced-order model closure

Romit Maulik[a,*], Arvind Mohan[b], Bethany Lusch[a], Sandeep Madireddy[c], Prasanna Balaprakash[a,c], Daniel Livescu[b]

[a] *Argonne Leadership Computing Facility, Argonne National Laboratory, Lemont, IL 60439, USA*
[b] *Center for Nonlinear Studies/CCS-2 Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*
[c] *Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL 60439, USA*

Abstract

We study the performance of long short-term memory networks (LSTMs) and neural ordinary differential equations (NODEs) in learning latent-space representations of dynamical equations for an advection-dominated problem given by the viscous Burgers equation. Our formulation is devised in a non-intrusive manner with an equation-free evolution of dynamics in a reduced space with the latter being obtained through a proper orthogonal decomposition. In addition, we leverage the sequential nature of learning for both LSTMs and NODEs to demonstrate their capability for closure in systems that are not completely resolved in the reduced space. We assess our hypothesis for two advection-dominated problems given by the viscous Burgers equation. We observe that both LSTMs and NODEs are able to reproduce the effects of the absent scales for our test cases more effectively than does intrusive dynamics evolution through a Galerkin projection. This result empirically suggests that time-series learning techniques implicitly leverage a memory kernel for coarse-grained system closure as is suggested through the Mori-Zwanzig formalism.

[*]Corresponding author
Email addresses: `rmaulik@anl.gov` (Romit Maulik), `arvindm@lanl.gov` (Arvind Mohan), `blusch@anl.gov` (Bethany Lusch), `smadireddy@anl.gov` (Sandeep Madireddy), `pbalapra@anl.gov` (Prasanna Balaprakash), `livescu@lanl.gov` (Daniel Livescu)

Keywords: ROMs, LSTMs, Neural ODEs, Closures

1. Introduction

High-fidelity simulations of systems characterized by nonlinear partial differential equations represent immense computational expenditure and are prohibitive for decision-making tasks for applications. Recently, researchers have expended significant effort in the reduced-order modeling (ROM) of such systems to reduce the degrees of freedom of the forward problem to manageable magnitudes [1, 2, 3, 4, 5, 6, 7, 8]. This field finds extensive application in control [9], multifidelity optimization [10], and uncertainty quantification [11, 12], among others. ROMs are limited in how they handle nonlinear dependence, however, and they perform poorly for complex physical phenomena that are inherently multiscale in space and time [13, 14, 15, 16]. To address this issue, researchers continue to search for efficient and reliable ROM techniques for transient nonlinear systems.

A common ROM development procedure may be described by the following tasks:

1. Reduced basis identification
2. Nonlinear dynamical system evolution in the reduced basis
3. Reconstruction in full-order space for assessments

The first two tasks individually constitute areas of extensive investigation, and studies have attempted to combine these into one optimization problem as well. In this investigation, we utilize conventional ideas for reduced basis identification with the use of the proper orthogonal decomposition (POD) for finding the optimal global basis. We consider a parameterized time-dependent partial differential equation given (in the full-order space) by

$$\dot{u}(x,t,\nu) + \mathcal{N}[u(x,t,\nu)] + \mathcal{L}[u(x,t,\nu);\nu] = 0, \quad (x,t,\nu) \in \Omega \times \mathcal{T} \times \mathcal{P}, \quad (1)$$

where $\Omega \subset \mathbb{R}^1, \mathcal{T} = [0,T], \mathcal{P} \subset \mathbb{R}^1$ and $\mathcal{N}$, $\mathcal{L}$ are nonlinear and linear operators, respectively. Our system is characterized by a solution field $u : \Omega \times \mathcal{T} \times \mathcal{P} \to \mathbb{R}^1$ and appropriately chosen initial as well as boundary conditions. We assume that our system of equations can be solved in space-time on a discrete grid resulting in the following systems of parameterized ordinary differential equations (ODEs),

$$\dot{\mathbf{u}}_h(t,\nu) + \mathbf{N}_h[\mathbf{u}_h(t,\nu)] + \mathbf{L}_h[\mathbf{u}_h(t,\nu);\nu] = 0 \quad (t,\nu) \in \mathcal{T} \times \mathcal{P}, \quad (2)$$

where $\mathbf{u}_h : \mathcal{T} \times \mathcal{P} \rightarrow \mathbb{R}^{N_h}$ is a discrete solution and N_h is the number of spatial degrees of freedom. Specifically, our problem is given by the viscous Burgers equation with periodic boundary conditions that can be represented as

$$\begin{gathered}\dot{u} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \\ u(x,0) = u_0, \quad x \in [0,L], \quad u(0,t) = u(L,t) = 0.\end{gathered} \tag{3}$$

These equations can generate discontinuous solutions even if initial conditions are smooth and ν is sufficiently small because of advection-dominated behavior. We can then project our governing equations onto a space of reduced orthonormal bases for inexpensive forward solves of the dynamics.

1.1. Proper orthogonal decomposition

In this section, we review the POD technique for the construction of a reduced basis [17, 18]. The interested reader may also find an excellent explanation of POD and its relationship with other dimension-reduction techniques in [19]. The POD procedure is tasked with identifying a space

$$\mathbf{X}^f = \operatorname{span}\left\{\boldsymbol{\vartheta}^1, \dots, \boldsymbol{\vartheta}^f\right\}, \tag{4}$$

which approximates snapshots optimally with respect to the L^2-norm. The process of $\boldsymbol{\vartheta}$ generation commences with the collections of snapshots in the *snapshot matrix*

$$\mathbf{S} = [\ \hat{\mathbf{u}}_h^1 \mid \hat{\mathbf{u}}_h^2 \mid \cdots \mid \hat{\mathbf{u}}_h^{N_s}\ ] \in \mathbb{R}^{N_h \times N_s}, \tag{5}$$

where $\hat{\mathbf{u}}_i : \mathcal{T} \times \mathcal{P} \rightarrow \mathbb{R}^{N_h}$ corresponds to an individual snapshot in time (for a total of N_s snapshots) of the discrete solution domain with mean value removed, namely,

$$\begin{gathered}\hat{\mathbf{u}}_h^i = \mathbf{u}_h^i - \bar{\mathbf{u}}_h, \\ \bar{\mathbf{u}}_h = \frac{1}{N_s}\sum_{i=1}^{N_s} \mathbf{u}_h^i.\end{gathered} \tag{6}$$

with $\overline{\mathbf{u}}_i : \mathcal{P} \rightarrow \mathbb{R}^{N_h}$ being the time-averaged solution field. Our POD bases can then be extracted efficiently through the method of snapshots where we solve an eigenvalue problem for a correlation matrix

$$\begin{gathered}\mathbf{CW} = \Lambda \mathbf{W}, \\ \mathbf{C} = \mathbf{S}^T\mathbf{S} \in \mathbb{R}^{N_s \times N_s},\end{gathered} \tag{7}$$

where $\Lambda = \operatorname{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_{N_s}\} \in \mathbb{R}^{N_s \times N_s}$ is the diagonal matrix of eigenvalues and $\mathbf{W} \in \mathbb{R}^{N_s \times N_s}$ is the eigenvector matrix. Our POD basis matrix can then be obtained by

$$\boldsymbol{\vartheta} = \mathbf{SW} \in \mathbb{R}^{N_h \times N_s}. \tag{8}$$

In practice, a reduced basis $\boldsymbol{\psi} \in \mathbb{R}^{N_h \times N_r}$ is built by choosing the first N_r columns of $\boldsymbol{\vartheta}$ for the purpose of efficient ROMs, where $N_r \ll N_s$. This reduced basis spans a space given by

$$\mathbf{X}^r = \operatorname{span}\left\{\boldsymbol{\psi}^1, \dots, \boldsymbol{\psi}^{N_r}\right\}. \tag{9}$$

The coefficients of this reduced basis (which capture the underlying temporal effects) may be extracted as

$$\mathbf{A} = \boldsymbol{\psi}^T \mathbf{S} \in \mathbb{R}^{N_r \times N_s}. \tag{10}$$

The POD approximation of our solution is then obtained via

$$\hat{\mathbf{S}} = [\ \tilde{\mathbf{u}}_h^1 \mid \tilde{\mathbf{u}}_h^2 \mid \cdots \mid \tilde{\mathbf{u}}_h^{N_s}\ ] \approx \boldsymbol{\psi}\mathbf{A} \in \mathbb{R}^{N_h \times N_s}, \tag{11}$$

where $\tilde{\mathbf{u}}_h^i : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^{N_h}$ corresponds to the POD approximation to $\hat{\mathbf{u}}_h^i$. The optimal nature of reconstruction may be understood by defining the relative projection error

$$\frac{\sum_{i=1}^{N_s} \|\hat{\mathbf{u}}_h^i - \tilde{\mathbf{u}}_h^i\|_{\mathbb{R}^{N_h}}^2}{\sum_{i=1}^{N_s} \|\hat{\mathbf{u}}_h^i\|_{\mathbb{R}^{N_h}}^2} = \frac{\sum_{i=N_r+1}^{N_s} \lambda_i^2}{\sum_{i=1}^{N_s} \lambda_i^2}, \tag{12}$$

which indicates that with increasing retention of POD bases, increasing reconstruction accuracy may be obtained. As will be explained later, the coefficient matrix $\mathbf{A}$ forms our training data for time-series learning.

1.2. Galerkin projection onto reduced space

The orthogonal nature of the POD basis may be leveraged for a Galerkin projection onto the reduced basis. We start by revisiting Equation (1) written in the form of an evolution equation for fluctuation components:

$$\dot{\hat{\mathbf{u}}}_h(x,t,\nu) + \mathcal{N}_h[\hat{\mathbf{u}}_h(x,t,\nu)] + \mathcal{L}_h[\hat{\mathbf{u}}_h(x,t,\nu);\nu] = 0, \tag{13}$$

which can expressed in the reduced basis as

$$\boldsymbol{\psi}\dot{\mathbf{a}}_r(t,\nu) + \mathcal{N}_h[\boldsymbol{\psi}\mathbf{a}_r(t,\nu)] + \mathcal{L}_h[\boldsymbol{\psi}\mathbf{a}_r(t,\nu);\nu] = 0, \tag{14}$$

where $\mathbf{a}_r : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^{N_r}, \mathbf{a}_r \in \alpha$ corresponds to the temporal coefficients at one time instant of the system evolution (i.e., equivalent to a particular column of $\mathbf{A}$). The orthogonal nature of the reduced basis can be leveraged to obtain

$$\dot{\mathbf{a}_r}(t,\nu) + \mathcal{N}_r[\mathbf{a}_r(t,\nu)] + \mathcal{L}_r[\mathbf{a}_r(t,\nu);\nu] = 0, \tag{15}$$

which we denote the POD-Galerkin projection formulation (POD-GP). Note that we have assumed that the residual generated by the truncated representation of the full-order model is orthogonal to the reduced basis. It is precisely this assumption that necessitates closure. From the point of view of the Burgers equations given in Equation (3), our POD-GP implementation is

$$\frac{da_k}{dt} = b_k^1 + b_k^2 + \sum_{i=1}^{N_r} \left(L_{ik}^1 + L_{ik}^2\right) a_i + \sum_{i=1}^{N_r}\sum_{j=1}^{N_r} N_{ijk} a_i a_j, \quad \text{for} \quad k = 1, 2, \dots, N_r, \tag{16}$$

where $a_k : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^1$ is one component of $\mathbf{a}_r$ and where

$$\begin{aligned}
b_k^1 &= \left(\nu L[\overline{\mathbf{u}}_h], \boldsymbol{\psi}^k\right), \\
b_k^2 &= \left(N[\overline{\mathbf{u}}_h; \overline{\mathbf{u}}_h], \boldsymbol{\psi}^k\right), \\
L_{ik}^1 &= \left(\nu L\left[\boldsymbol{\psi}^i\right], \boldsymbol{\psi}^k\right), \\
L_{ik}^2 &= \left(N\left[\overline{\mathbf{u}}_h; \boldsymbol{\psi}^i\right] + N\left[\boldsymbol{\psi}^i; \overline{\mathbf{u}}_h\right], \boldsymbol{\psi}^k\right), \\
N_{ijk} &= \left(N\left[\boldsymbol{\psi}^i; \boldsymbol{\psi}^j\right], \boldsymbol{\psi}^k\right),
\end{aligned} \tag{17}$$

are operators that can be computed offline (i.e., $b_k^1, L_{ik}^1 : \mathcal{P} \to \mathbb{R}^1; b_k^2, L_{ik}^2, N_{ijk} \in \mathbb{R}^1$) and where we have defined an inner product by

$$(f, g) = \int_\Omega f g d\Omega. \tag{18}$$

with $L[f] = \frac{\partial^2 f}{\partial x^2}$ and $N[f; g] = -f\frac{\partial g}{\partial x}$, the operators stemming from the Burgers equation. Later we will demonstrate that the absence of higher-basis nonlinear interactions causes errors in the forward evolution of this system of equations. We note that the POD-GP process essentially consists of N_r coupled ODEs and is solved by a standard total-variation-diminishing third-order Runge-Kutta method. The reduced degrees of freedom lead to efficient forward solves of the problem. Note that this transformed problem has initial conditions given by

$$\mathbf{a}_r(t=0) = \left(\boldsymbol{\psi}^T \hat{\mathbf{u}}_h(t=0)\right). \tag{19}$$

1.3. Contribution

In this article, we investigate strategies to bypass the POD-GP process with an intrusion-free (or equation-free) learning of dynamics in reduced space. We deploy machine learning strategies devised for sequential data learning on time-series samples extracted from the true dynamics of the problems considered. In recent literature, considerable interest has been expressed in the utility of machine learning techniques for effective ROM construction. Data-driven techniques have been used in various phases of ROM techniques, such as in reduced basis construction [20], augmented dynamics evolution [21, 22, 15, 5, 23, 16, 24, 6, 25, 26], and system identification [27, 28, 29, 30, 31].

Here we study the utility of data-driven techniques to make *a posteriori* predictions for state evolution in reduced space, with assessments made of their ability to reconstruct transient characteristics of the full-order solution. We observe that the ability to learn a time series (possible through the inbuilt design of memory and an assumption of non-i.i.d. sequential data in the learning) leads to an implicit closure whereby the effects of uncaptured frequencies are retained, drawing parallels to a Mori-Zwanzig formalism. In related work, the study in [32] utilizes recurrent neural networks to explicitly specify a subgrid stress model for large eddy simulation with the lower-frequency evolution controlled by coarse-grained partial differential equations (PDEs). Our framework is also similar to [33], where a long short-term memory is utilized to learn a *parametric* memory kernel for explicit closure of nonlinear PDEs. The present study can be considered a *nonintrusive* counterpart of these investigations. In addition, we detail a formalism for efficient machine learning architecture selection using scalable Bayesian optimization. Our test problems are given by the advection-dominated viscous Burgers equation [15] with a moving shock as well as a pseudo-turbulence test case denoted "Burgulence" [34, 35] showing the characteristic k^{-2} scaling in wavenumber (k) space.

2. Latent-space learning

In this section, we outline our machine learning techniques for latent-space time-series learning. We study two techniques built around the premise of preserving memory effects within their architecture: neural ordinary differential equations (NODE) and long short-term memory networks (LSTMs). Both frameworks are tasked with predicting the evolution of $\mathbf{a}_r$ over time.

2.1. Neural ordinary differential equations

In recent times, several studies have interpreted residual neural networks using dynamical systems theory [36, 37, 38, 39]. The framework of the NODE [40] envisions the learning of $\mathbf{a}_r$ over time as

$$\frac{d\mathbf{a}_r}{dt} = f(\mathbf{a}_r, \theta), \quad (\theta) \in \Theta, \tag{20}$$

where $\Theta \subset \mathbb{R}^{N_w}$ is a space of N_w user-defined model parameters. The learning can be thought to be through a continuous backpropagation through time, that is, where there an infinite number of layers in the network with the input layer given by $\mathbf{a}_r(t = 0)$ and the output layer given by $\mathbf{a}_r(t = T)$. Therefore, the NODE approximates the latent-space evolution as an ordinary differential equation that is continuous through time in a manner similar to the Galerkin projection. The function $f : \alpha \times \Theta \to \mathbb{R}^{N_r}$ in this study is represented by a neural network with a single 40-neuron hidden layer and a tan-sigmoid activation where $\alpha \subset \mathbb{R}^{N_r}, \Theta \subset \mathbb{R}^{N_w}$, and N_w is the number of parameters of the neural network architecture. Note that the assumption of a single hidden layer architecture for the right-hand side of the latent-space ODE allows for upper-bound guarantees given by the universal approximation theorem (Barron, 1993) although more complicated dynamics may require deeper architectures. Readers are referred to the work of Chen et al. [40], for a detailed discussion of the neural ODE and its utility in learning sequential data.

The forward propagation of information through time (i.e., from $t = 0$ to $t = T$) is performed through a standard ODE solver (in this case a first-order accurate Euler method) whereas backpropagation of errors is performed through the backward solve of an adjoint system given by

$$\frac{d\mathbf{b}_g}{dt} = -\mathbf{b}_g^{\mathrm{T}} \frac{\partial f(\mathbf{a}_r, \theta)}{\partial \mathbf{a}_r}, \tag{21}$$

where $\mathbf{b}_g : \mathcal{T} \times \alpha \times \mathcal{E} \rightarrow \mathbb{R}^{N_r+N_w+1}$ is the augmented state vector given by

$$\mathbf{b}_g = [\frac{\partial E}{\partial \mathbf{a}_r}, \frac{\partial E}{\partial \theta}, \frac{\partial E}{\partial t}]^T, \quad E \in \mathcal{E}, \tag{22}$$

with scalar loss at final time $\mathcal{E} \subset \mathbb{R}^1$ obtained at the $t = T$ following forward propagation. Each calculation of E is followed by the backward solve of Equation (21) (which may be interpreted as continuous backpropagation in time) to calculate $\mathbf{b}_g(t = 0)$, which can then be used to determine $\frac{\partial E}{\partial \theta}(t = 0)$. This value of the gradient can then be used to update the parameters θ by using an optimization algorithm. In this article, we utilize RMSProp for our loss minimization with a learning rate of 0.01 and a momentum parameter of 0.9. Instead of performing the forward deployment of the NODE and backpropagation of the model errors for the entire domain, we utilize 1,000 samples of our total data as our training and validation dataset using the technique detailed in the original article to speed up training. Each sample is given by a sequence of 10 timesteps. The training, for each epoch, is performed by using 10 randomly chosen samples (i.e., our batch size is 10) for the calculation of parameter gradients. The final gradient deployed for model training is averaged across this batch. A set of samples (20% of the total 1,000), chosen randomly before training, is kept aside from the learning process to assess validation errors. Note that validation errors are also characterized by final timestep loss (i.e., at timestep 10 of each batch), thereby incorporating the degree of error accummulation due to an inaccurately trained model at that epoch. The best model corresponds to the lowest validation loss (averaged across all validation samples). We do not utilize a separate dataset for testing.

All assessments for the problems are through forward (or *a posteriori*) deployment. In other words, the NODE is specified an initial condition and then deployed to obtain state vectors using an ODE forward solve until the final time. The prediction at each timestep is obtained by the Euler integration, which requires the knowledge of previous state alone. Note that apart from the first prediction by NODE (which utilizes the initial condition), state predictions are recursively utilized for predicting the future. Therefore, testing may be assumed to be a long-term predictive test of the model learning in the presence of deployment error.

2.2. Long short-term memory networks

LSTM networks were introduced to consider time-delayed processes where events farther back in the past may potentially affect predictions for the current location in the sequence. The basic equations of the LSTM in our context for an input variable $\mathbf{z}$ are

$$\begin{aligned}
\text{input gate: } & \boldsymbol{G}_i = \boldsymbol{\varphi}_S \circ \mathcal{F}_i^{N_c}(\mathbf{z}), \\
\text{forget gate: } & \boldsymbol{G}_f = \boldsymbol{\varphi}_S \circ \mathcal{F}_f^{N_c}(\mathbf{z}), \\
\text{output gate: } & \boldsymbol{G}_o = \boldsymbol{\varphi}_S \circ \mathcal{F}_o^{N_c}(\mathbf{z}), \\
\text{internal state: } & \boldsymbol{s}_t = \boldsymbol{G}_f \odot \boldsymbol{s}_{t-1} + \boldsymbol{G}_i \odot \left(\boldsymbol{\varphi}_T \circ \mathcal{F}_{\mathbf{z}}^{N_c}(\mathbf{z})\right), \\
\text{output: } & \mathbf{h}_t = \boldsymbol{G}_o \circ \boldsymbol{\varphi}_T\left(\boldsymbol{s}_t\right),
\end{aligned} \tag{23}$$

where $\mathbf{z}$ is a fixed sequence of inputs $\mathbf{a}_r$ comprising past history. Also, $\boldsymbol{\varphi}_S$ and $\boldsymbol{\varphi}_L$ refer to tangent sigmoid and tangent hyperbolic activation functions, respectively, and N_c is the number of hidden layer units in the LSTM network. Note that $\mathcal{F}^n$ refers to a linear operation given by a matrix multiplication and subsequent bias addition, namely,

$$\mathcal{F}^n(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{B}, \tag{24}$$

where $\boldsymbol{W} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{B} \in \mathbb{R}^n$ for $\mathbf{x} \in \mathbb{R}^m$ and where $\mathbf{a} \circ \mathbf{b}$ refers to a Hadamard product of two vectors. The LSTM implementation will be used to advance $\mathbf{a}_r$ as a function of time in the reduced space. The LSTM network's primary utility is the ability to control information flow through time with the use of the gating mechanisms. A greater value of the forget gate (post-sigmoidal activation) allows for a greater preservation of past state information through the sequential inference of the LSTM, whereas a smaller value suppresses the influence of the past. Our LSTM deployment utilized 32 neurons in its input, forget, output, and state calculation operations each and utilized a learning rate of 0.001. It used a sequence to sequence prediction utilized as a rolling window for predicting the output at the next timestep. We utilized a batch size of 16 samples, with each sample having a sequence of 10 timesteps for all of our LSTM deployments. As in the previous learning approach, a set of data was kept aside for validation. This validation loss is used to make decisions about model selection. We note that the total number of samples (1,000) is the same as for the NODE deployment.

2.3. Connection with Mori-Zwanzig formalism

Next, we outline the Mori-Zwanzig formalism [41, 42] for the viscous Burgers equation and connect it to time-series learning in POD space. We frame the (full-order) dynamics evolution in latent space using the following formulation derived from the first step of the Mori-Zwanzig, treatment

$$\frac{d\mathbf{a}_r}{dt} = e^{\mathcal{W}t}\mathcal{W}\mathbf{a}_r, \tag{25}$$

where $\mathcal{W}$ is the viscous Burgers operator given by

$$\mathcal{W} = -\mathcal{N}_r\left[.\right] - \mathcal{L}_r\left[.\right]. \tag{26}$$

We define two self-adjoint projection operators into orthogonal subspaces given by

$$P\mathbf{a}_h = \frac{(\mathbf{a}_r^0, \mathbf{a}_h^0)}{(\mathbf{a}_r^0, \mathbf{a}_r^0)}\mathbf{a}_r, \quad Q = I - P, \tag{27}$$

with $QP = 0$ and $\mathbf{a}^0 = \mathbf{a}(t = 0)$. Therefore, P may be assumed to be a projection of our full-order representation in POD space ($\mathbf{a}_h$ living in $\mathbf{X}^f$) onto the reduced basis ($\mathbf{a}_r$ living in $\mathbf{X}^r$). We can further expand our system as

$$\frac{d\mathbf{a}_r}{dt} = e^{\mathcal{W}t}(Q + P)\mathcal{W}\mathbf{a}_r, \tag{28}$$

which may further be decoupled to a Markov-like projection operator $\mathcal{M}$ given by

$$e^{\mathcal{W}t}P\mathcal{W}a_r = \frac{(\mathbf{a}_r^0, \mathbf{a}_h^0)}{(\mathbf{a}_r^0, \mathbf{a}_r^0)}e^{\mathcal{W}t}a_r = \mathcal{M}a_r \tag{29}$$

and a memory operator $\mathcal{G}$ given by

$$e^{\mathcal{W}t}Q\mathcal{W}\mathbf{a}_r^0 = e^{Q\mathcal{W}t}Q\mathcal{W}\mathbf{a}_r^0 + \int_0^t e^{\mathcal{W}(t-t_1)}P\mathcal{W}e^{Q\mathcal{W}t_1}Q\mathcal{W}\mathbf{a}_r^0 dt_1 = \mathcal{G}a_r, \tag{30}$$

for which we have used Dyson's formula [43] and where t_1 corresponds to a hyperparameter that specifies the length of memory retention. The second relationship may be assumed to be a combination of memory effects and noise. The final evolution of the system can then be bundled into a linear combination of these two kernels, namely,

$$\frac{d\mathbf{a}_r}{dt} = \mathcal{G}a_r + \mathcal{M}a_r. \tag{31}$$

The reader may compare this expression with that of the internal state update within an LSTM,

$$\boldsymbol{s}_t = \boldsymbol{G}_f \odot \boldsymbol{s}_{t-1} + \boldsymbol{G}_i \odot \left(\boldsymbol{\varphi}_T \circ \mathcal{F}_{\mathbf{a}_r}^{N_c}(\mathbf{a}_r)\right), \tag{32}$$

where a linear combination of a nonlinearly transformed input vector at time t with the gated result of a hidden state at a previous time $t-1$ is used to calculate the result vector at the current time. The process of carrying a state through time via gating may be assumed to be a representation of the memory integral (as well as the noise), whereas the utilization of the current input may be assumed to be the Markovian component of the map. In contrast, from the point of view of the NODE implementation, the goal is to learn $e^{\mathcal{W}t}\mathcal{W}\mathbf{a}_r$ directly through a neural network.

3. Experiments

In this section, we assess the performance of both NODE and LSTM frameworks in representing latent-space dynamics appropriately. We investigate two problems given by the viscous Burgers equation in a periodic domain. Both problems are advection dominated: the first has a moving discontinuity over time (which we designate the *advecting shock* problem), and the second is characterized by standing shocks of various magnitudes (which we designate *Burgulence*). Their problem statement and results are shown below.

3.1. Advecting shock

Our first problem is given by the following initial and boundary conditions:

$$u(x,0) = \frac{x}{1+\sqrt{\frac{1}{t_0}}\exp\left(Re\frac{x^2}{4}\right)}, \tag{33}$$

$$u(0,t) = 0, \tag{34}$$

$$u(L,t) = 0, \tag{35}$$

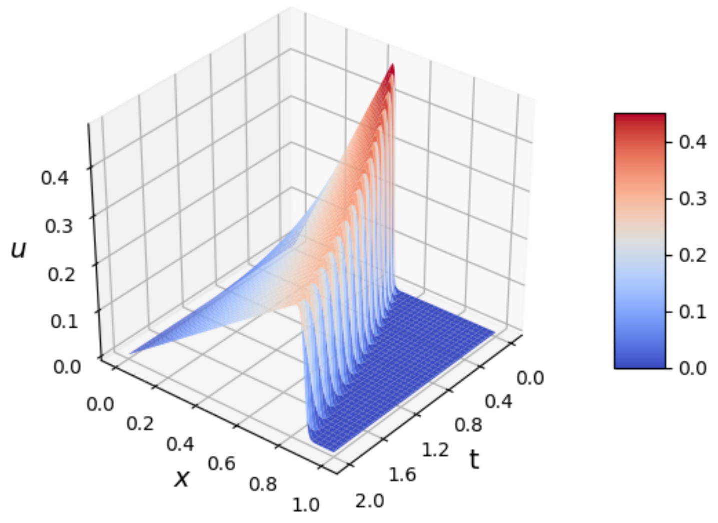


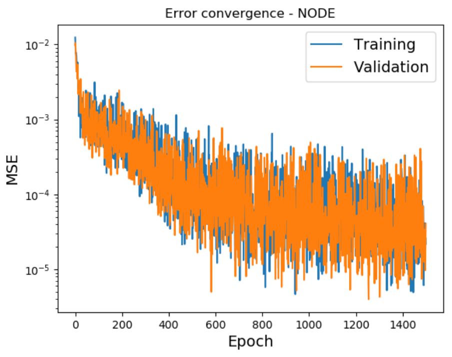
Figure 1: Full-order solution of the advecting shock problem. Note the presence of a moving discontinuity.

where we specify $L = 1$ and maximum time $t_{max} = 2$. An analytical solution for this set of equations is given by

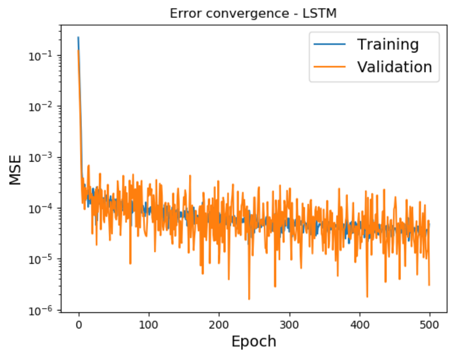
$$u(x,t) = \frac{\frac{x}{t+1}}{1 + \sqrt{\frac{t+1}{t_0}} \exp\left(Re \frac{x^2}{4t+4}\right)}, \tag{36}$$

where $t_0 = \exp(Re/8)$ and $Re = 1/\nu$ is kept fixed at 1,000. We directly utilize this expression to generate our snapshot data for ROM assessments. A visualization of the time evolution of the initial condition is shown in Figure 1. As outlined in a previous assessment of this problem [15], a reduced basis constructed of 20 basis vectors retains 99.93% of the energy of the system. For our assessments, we retain only three modes, resulting in an unresolved ROM that corresponds to only 86.71 % of the total energy, thus necessitating closure.

We perform an optimization for learning the modal coefficient time series using the LSTM and NODE frameworks. To recap model specifics, we deploy NODE (using 40 neurons) and LSTM (using 32 hidden layer neurons)



(a) LSTM



(b) NODE

Figure 2: Training and validation loss convergence with epochs for time-series predictions of the advecting shock case.

for learning the sequential nature of the modal coefficient evolution in POD space. We utilize the RMSprop optimizer using a learning rate of 0.01 for the former and 0.001 for the latter and a momentum coefficient of 0.9 for both. Batch sizes of 10 and 16 respectively are also used at each epoch of the learning phase. We use 1,000 randomly chosen sequence lengths of 10 for learning and validation through time, with 20% of the total data kept aside for the latter. We note that the best validation loss (aggregated over all validation samples) is utilized for model selection. Figure 2 shows the progress to convergence for both LSTM and NODE architectures during training for the first three modal coefficients. Both NODE and LSTM trainings are run until validation loss magnitudes hover around a magnitude of 10^{-4}. We observe that the LSTM framework reaches convergence more quickly although the oscillating losses of the NODE potentially indicate better exploration. The oscillations may also indicate the requirement of a lower learning rate.

The time-series predictions for the trained frameworks are shown in Figure 3, where a_0, a_1, and a_2 correspond to the first three retained modes. For comparison, we also show predictions from GP and the true modal coefficients, the latter of which are utilized for training our time-series predictions. We observe that both LSTM and NODE deployments capture coefficient trends accurately indicating that sequential behavior has been learned. The GP predictions can be seen to show unphysical growth in coefficient amplitude due to the lack of presence of the finer modes. However, LSTM and NODE deployments embed memory into their formulation in the form of a hidden state or through explicit learning of a latent-space ODE respectively. The memory-based nature of their learning leads to excellent agreement with the true behavior of the resolved scales.

The final time reconstructions for the true as well as the GP, LSTM and NODE time-series predictions are shown in Figure 4. One can observe that at this severely truncated state, the discontinuity is not completely resolved. The GP reconstructions show the manifestation of large oscillations (now in the physical domain) whereas NODE and LSTM are able to recover the true solution well. Figures 5 and 6 show a validation of our learning in an ensemble sense, where multiple architectures (with slight differences in the hidden layer neurons) are able to recover similar trends in accuracy as examined through final time reconstruction ability. This reinforces our assumption that an implicit closure is being learned through time-series trends in a statistical manner. The corresponding training losses for the LSTM and NODE architectures are shown in Figures 7 and 8, where similar learning trends are obtained with slight variations in the number of trainable parameters.

3.2. Burgers' turbulence

Our next test case is given by the challenging Burgers turbulence or *Burgulence* test case that leads to multiscale behavior in wavenumber space. Our problem domain is given by a length $L = 2\pi$, and the initial condition is specified by an initial energy spectrum (in wavenumber space) given by

$$E(k) = Ak^4 \exp\left(-\left(k/k_0\right)^2\right), \tag{37}$$

where k is the wavenumber and $k_0 = 10$ is the parameter at which the peak value of the energy spectrum is obtained. The constant A is set to

$$A = \frac{2k_0^{-5}}{3\sqrt{\pi}}, \tag{38}$$

in order to ensure a total energy of $\int E(k)dk = 1/2$ at the initial condition. The initial velocity magnitudes can be expressed in wavenumber space by the following relation with our previously defined spectrum,

$$\hat{u}(k) = \sqrt{2E(k)}\exp(i2\pi\Psi(k)), \tag{39}$$

where $\Psi(k)$ is a uniform random number generated between 0 and 1 at each wavenumber. Note that this distribution is constrained by $\Psi(k) = -\Psi(k)$

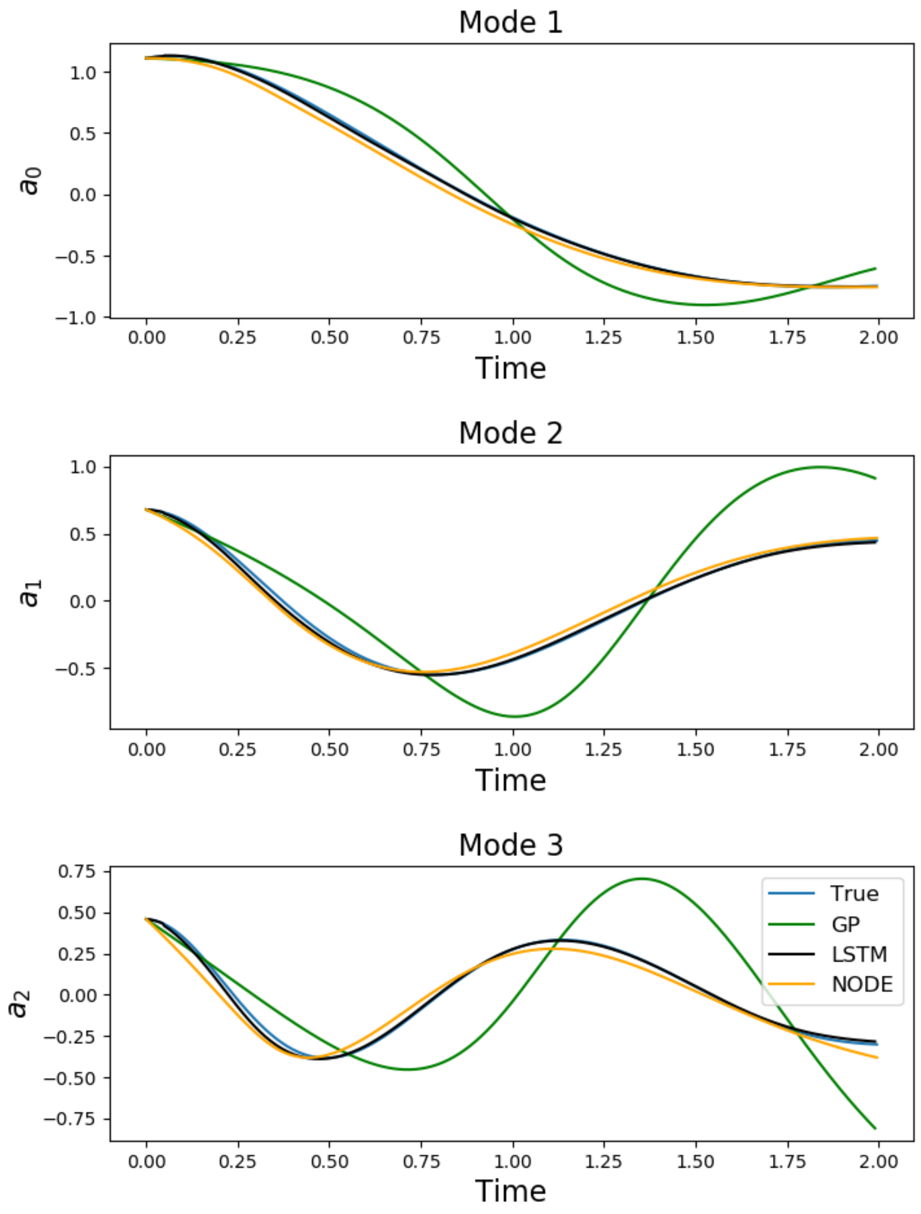


Figure 3: POD-space coefficient evolution for the advecting shock case.

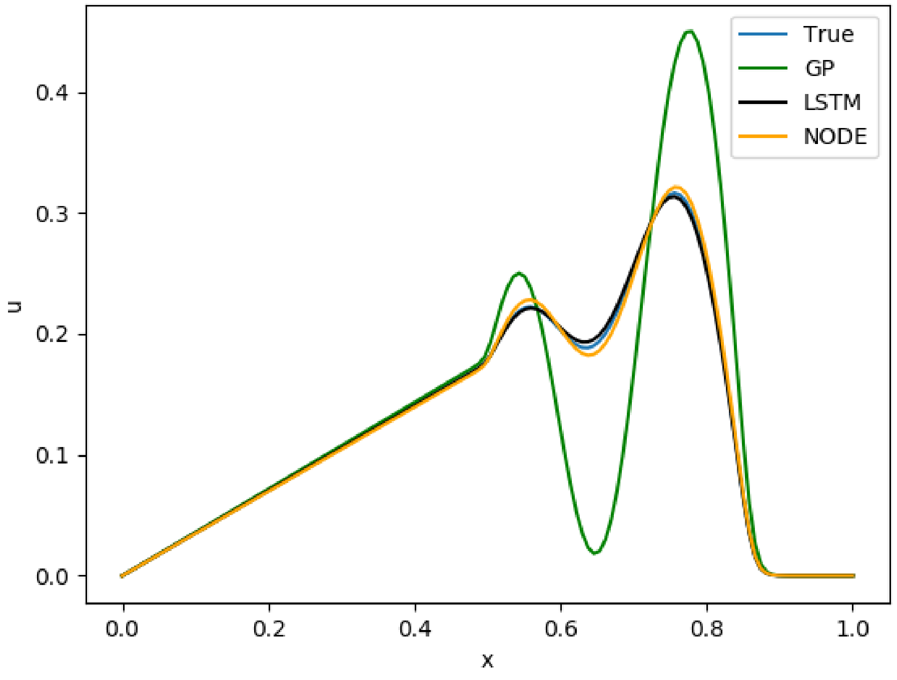
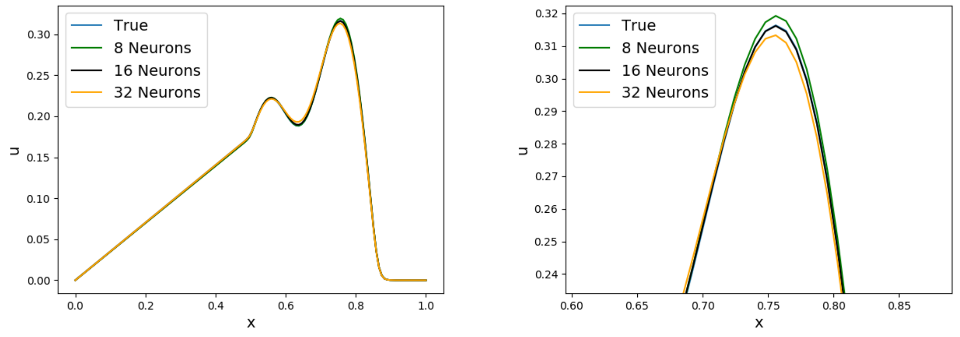


Figure 4: Field reconstruction ability for NODE and LSTM for the advecting shock case.



(a) Field (b) Zoomed

Figure 5: Comparison of three different LSTM predictions for the advecting shock case.

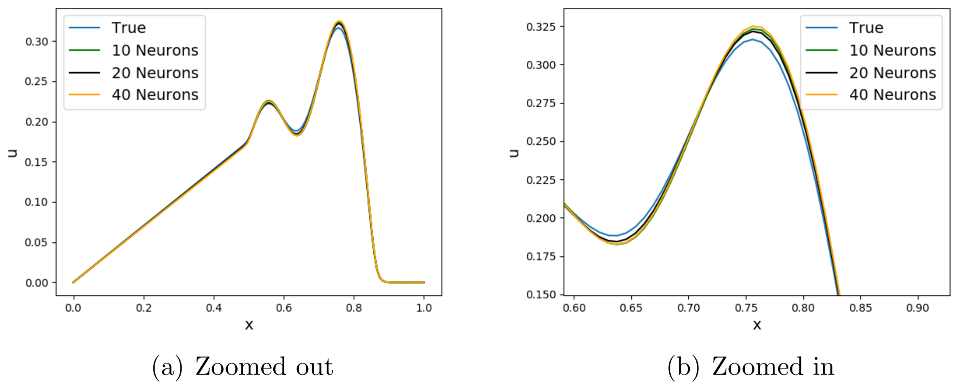


Figure 6: Comparison of three different NODE predictions for the advecting shock case. The deployment with 16 neurons coincides with the true solution.

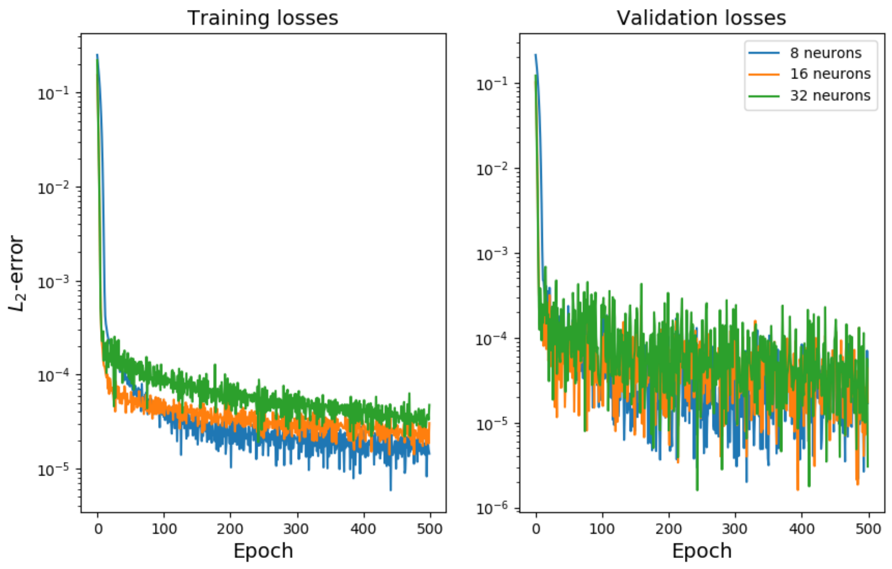


Figure 7: Ensemble training and validation losses for the LSTM architecture for the advecting shock case.

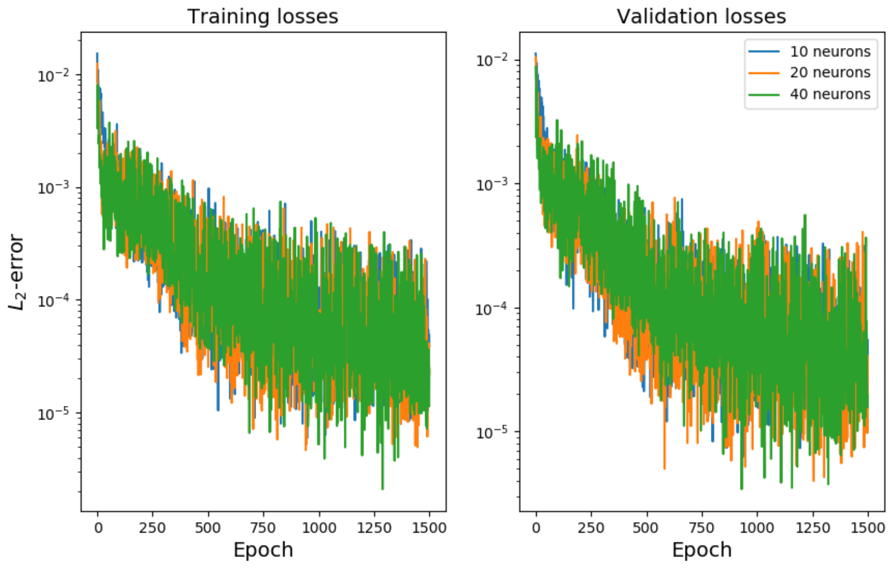


Figure 8: Ensemble training and validation losses for the NODE architecture for the advecting shock case.

to ensure that a real initial condition in physical space is obtained. For our assessment, we use energy spectra given by

$$E(k,t) = \frac{1}{2}|\hat{u}(k,t)|^2. \tag{40}$$

The aforementioned initial conditions are solved for the viscous Burgers equation in wavenumber space by a Runge-Kutta Crank-Nicolson scheme as described in [44]. Note that our ν is chosen to be 2.5×10^{-3} to ensure that sharp discontinuities emerge from the smooth initial condition. Our NODE and LSTM hyperparameters are identical to the previous test case. Our investigations here are performed for the initial condition (and its corresponding time evolution) as shown in Figure 9. We observe that the solution has a considerable multiscale element that makes this a challenging problem for POD-ROM techniques.

Figure 10 shows reduced-space time-series evolutions of the three retained modal coefficients for the frameworks we are comparing. We observe that the LSTM and NODE techniques are successful in coefficient evolution stabilization in comparison with GP, although the LSTM adds an element of phase

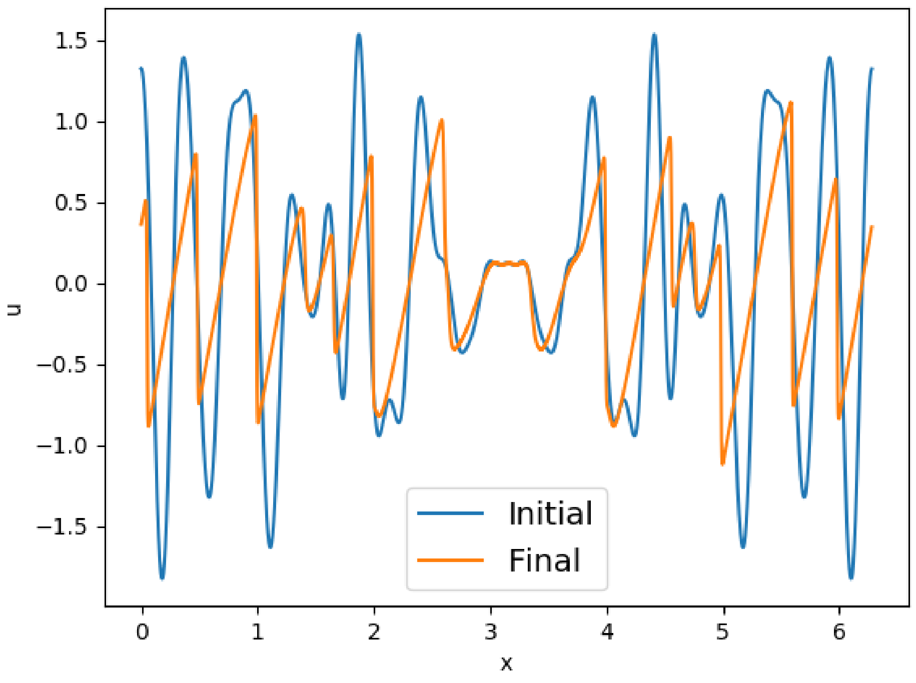


Figure 9: Initial and final conditions for the *Burgulence* case showing multiple standing discontinuities decaying in strength over time.

error. The NODE, however, captures latent-space trends exceptionally well. The performance of these time-series learning models is further assessed by their reconstruction in physical space as shown in Figure 11, where one can see that the LSTM and NODE perform well in preventing spurious oscillations near discontinuities as exhibited by an unclosed GP evolution. A further validation of this hypothesis is observed in Figure 12, where kinetic energy spectra in wavenumber space show that the high residuals of the GP method are controlled effectively by the LSTM and NODE deployments. The LSTM results in slightly higher residuals for this particular test case and choice of hyperparameters and optimizer.

3.3. Improving performance through hyperparameter search

While the results presented in the preceding sections indicate an acceptable choice for hyperparameters, we utilize DeepHyper [45] to improve the test performance of our frameworks. This choice is motivated by the comparatively poorer performance of STM in the *Burgulence* experiment. DeepHyper relies on an asynchronous-model-based search (i.e., a dynamically updated surrogate model $\mathcal{S}$, which is inexpensive to evaluate) for obtaining hyperparameters with the lowest validation losses. To ensure an expressive surrogate model that is still computationally tractable, we utilize random forest (RF). This results in a superior search algorithm as compared with both a random-search and a genetic-algorithm-based search. We note that RF also enables us to handle discrete and nonordinal parameters directly without the need for any encoding. DeepHyper is configured for searching the hyperparameter space using a standard Bayesian optimization framework. In other words, for each sampled configuration s, $\mathcal{S}$ predicts a mean value for the validation loss $\mu(s)$ and standard deviation $\sigma(s)$. This information is utilized recursively to improve the approximation to the loss surface as predicted by $\mathcal{S}$. In terms of exploring the hyperparameter search space, evaluation points with small values of $\mu(s)$ indicate that s can potentially result in the reduction of validation error subject to the accuracy of $\mathcal{S}$. Evaluation of points with large values of $\sigma(s)$ improves $\mathcal{S}$ since these locations are areas where $\mathcal{S}$ is least confident about the approximation surface. The choice for the selection of a configuration s is utilized by minimizing an acquisition function given by

$$\mathcal{A}(s) = \mu(s) - \lambda * \sigma(s), \tag{41}$$

where $\lambda = 1.96$ for encouraging exploration.

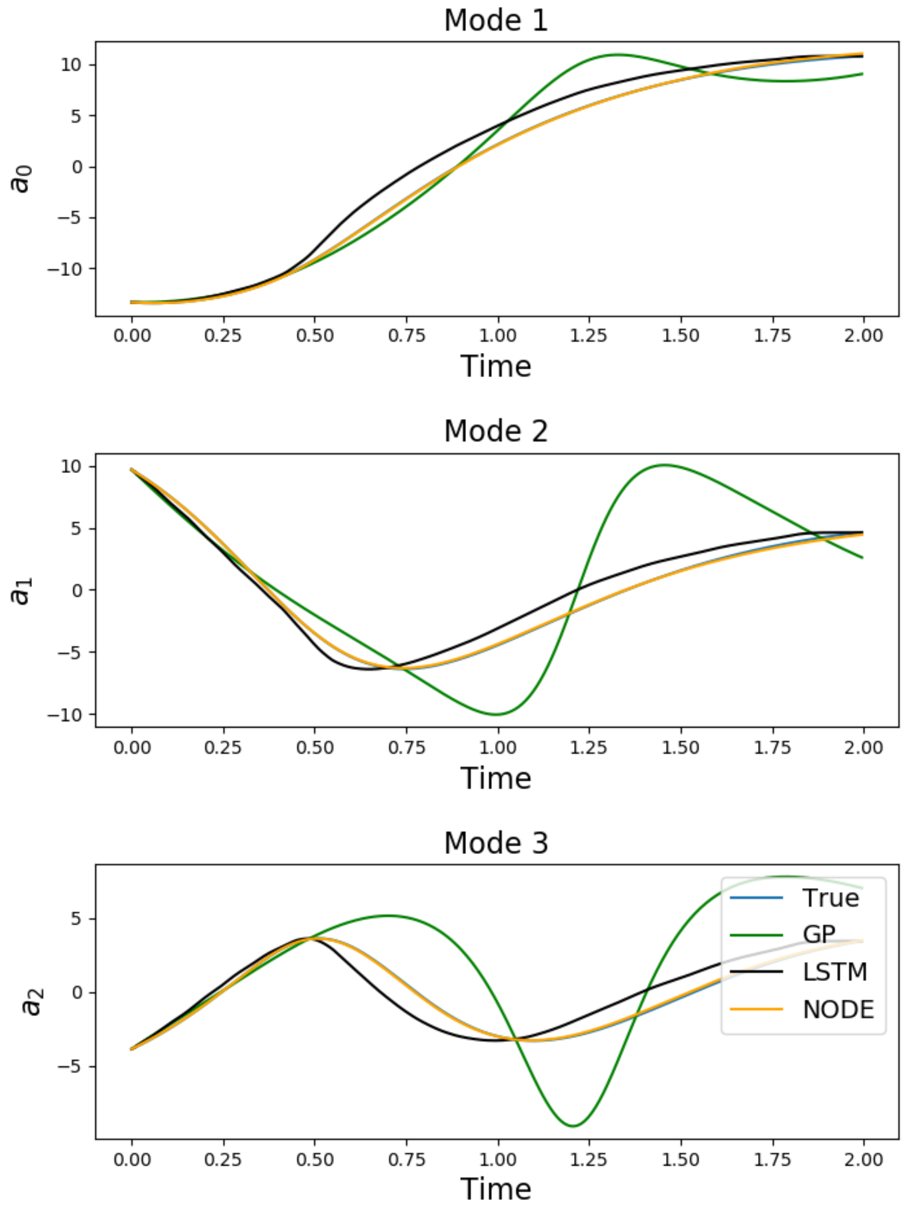
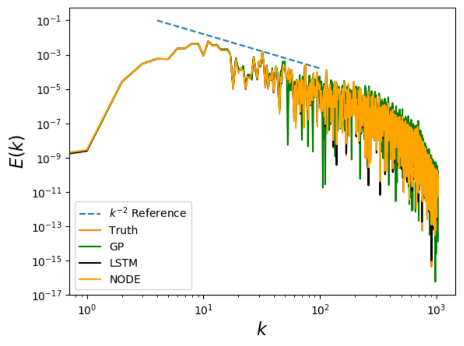
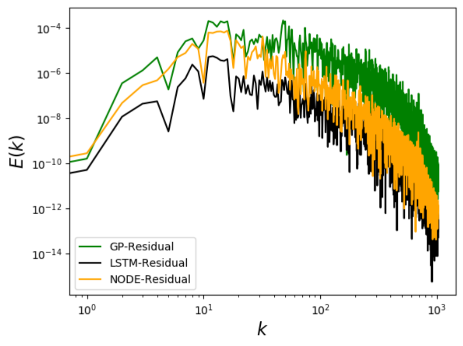


Figure 10: POD-space coefficient evolution for the *Burgulence* case.

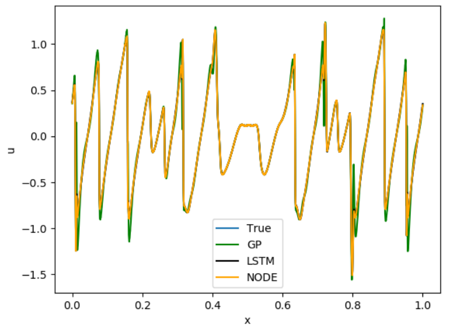


(a) Spectra

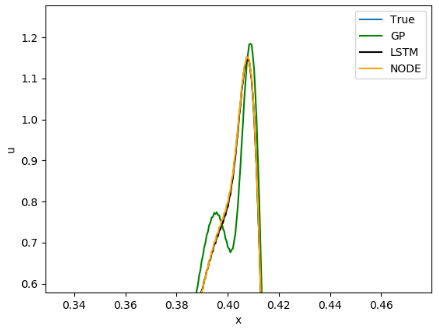


(b) Residual

Figure 11: Kinetic-energy spectra predictions (left) and their residuals (right) as predicted by NODE and LSTM.



(a) Zoomed out



(b) Zoomed in

Figure 12: Field reconstruction abilities for the NODE and LSTM frameworks showing superior performance as compared with GP.

Table 1: Search range for LSTM hyperparameters and their optimal values deployed for the Burgers' turbulence test case.

Hyperparameter	Type	Starting value	Ending value	Optimal
Sequence size	Integer	5	30	30
Neurons	Integer	5	100	73
Learning rate	Real	0.0001	0.1	0.0005
Momentum	Real	0.99	0.999	0.9988
Epochs	Integer	100	1000	317
Batch Size	Integer	5	30	8

DeepHyper requires a range specification for real variables and a list of possible choices for discrete hyperparameters. Table 1 outlines the range of hyperparameters for the LSTM architecture utilized for the Burgers turbulence test case as well as the optimal hyperparameters obtained. A summary of the distribution of sampled hyperparameters and pairwise dependencies is also shown in Figure 13. Note that loss is encoded as negative since the hyperparameter search is based on objective function maximization. Hyperparameter correlations are summarized in Figure 14, where we observe that most hyperparameters are weakly correlated with each other. We note, however, that these results are problem specific. In total, 2,151 hyperparameter combinations were evaluated during this search.

For comparison, we also show results from a similar hyperparameter search experiment for the NODE but for the advecting shock experiment. The optimal parameters and ranges of this search are listed in Table 2. A summary of the distribution of sampled hyperparameters and pairwise dependencies is also shown in Figure 15. Correlation plots between hyperparameters are shown in Figure 16. In total, 734 hyperparameter combinations were evaluated during this search.

We also deployed the optimal hyperparameter configuration for an *a posteriori* assessment. The results are shown in Figure 17. We can see that by using LStm, an improved performance has been obtained that now matches NODE and true observations. In addition, an analysis of the spectra and residuals in Figure 18 confirms the superior performance as well. DeepHyper has successfully led to an improved LSTM architecture.

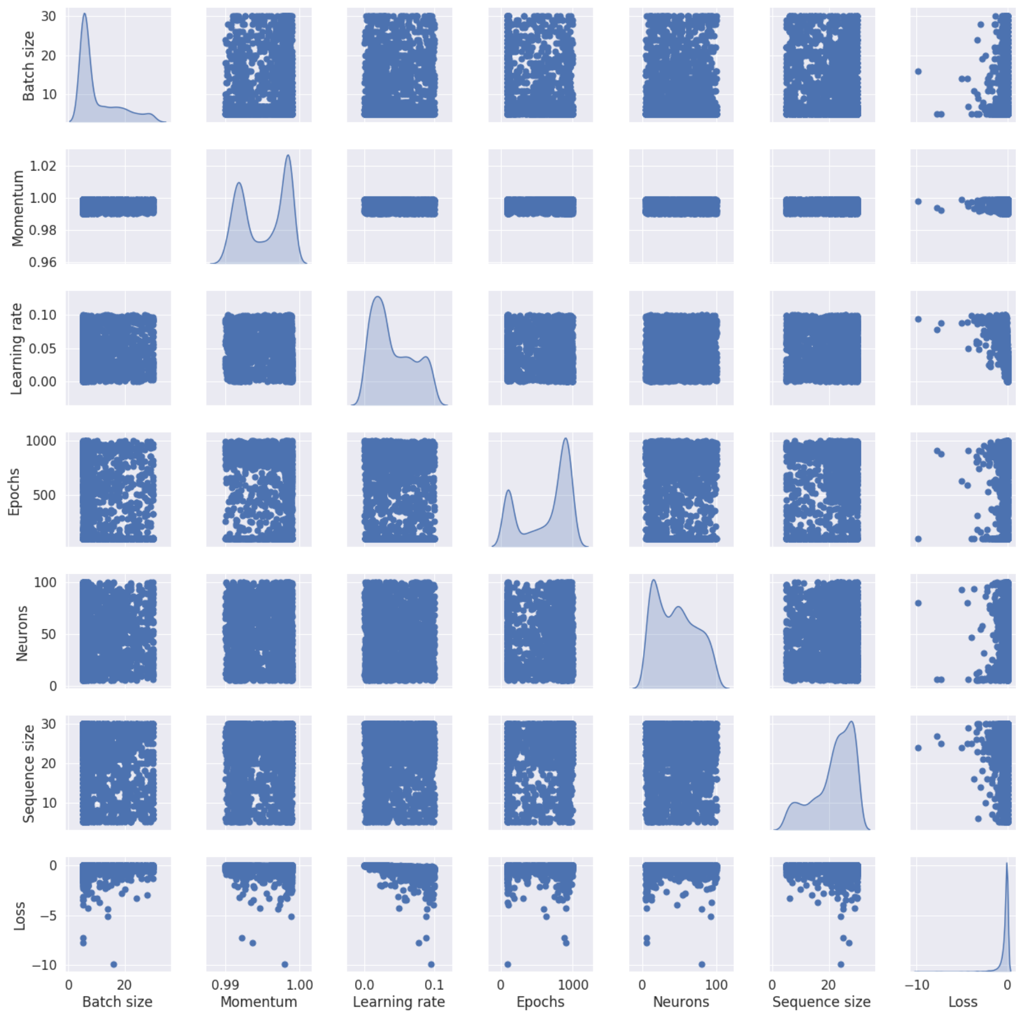


Figure 13: Pairwise dependency plots for LSTM hyperparameter search using DeepHyper. Diagonal entries show distributions of configurations sampled. Note that loss is encoded as negative since the hyperparameter search is based on objective function maximization.

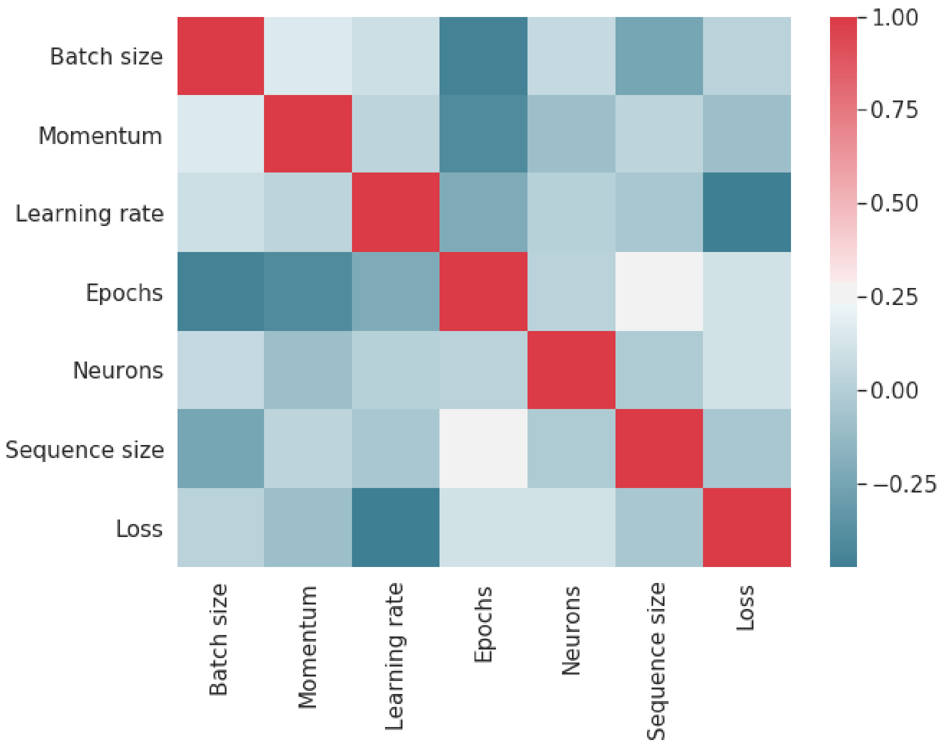


Figure 14: Pairwise LSTM hyperparameter correlations for the Burgers turbulence case.

Table 2: Search range for NODE hyperparameters and their optimal values deployed for the Burgers' turbulence test case.

Hyperparameter	Type	Starting value	Ending value	Optimal
Sequence size	Integer	5	30	5
Neurons	Integer	10	100	82
Learning rate	Real	0.0001	0.1	0.0074
Momentum	Real	0.99	0.999	0.9983
Epochs	Integer	200	1200	546
Batch Size	Integer	5	30	21

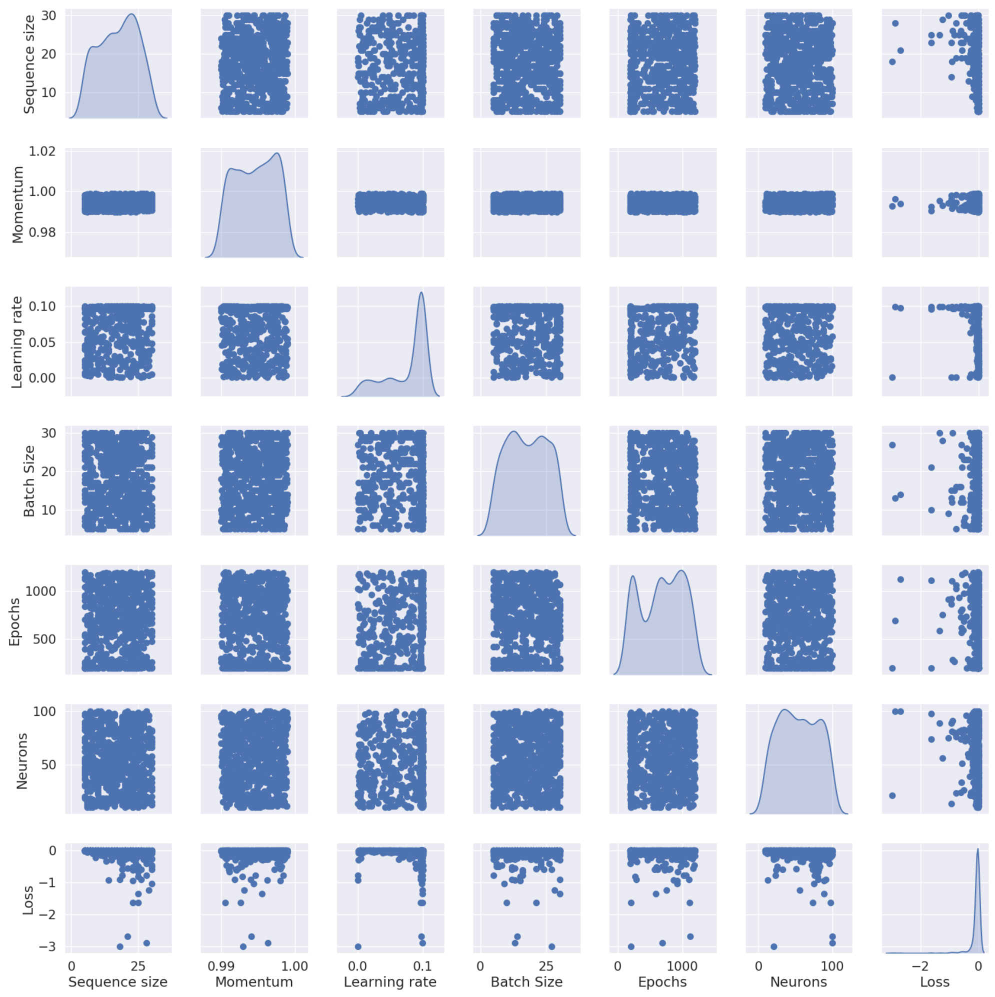


Figure 15: Pairwise dependency plots for NODE hyperparameter search using DeepHyper. Diagonal entries show distributions of configurations sampled. Note that loss is encoded as negative since the hyperparameter search is based on objective function maximization.

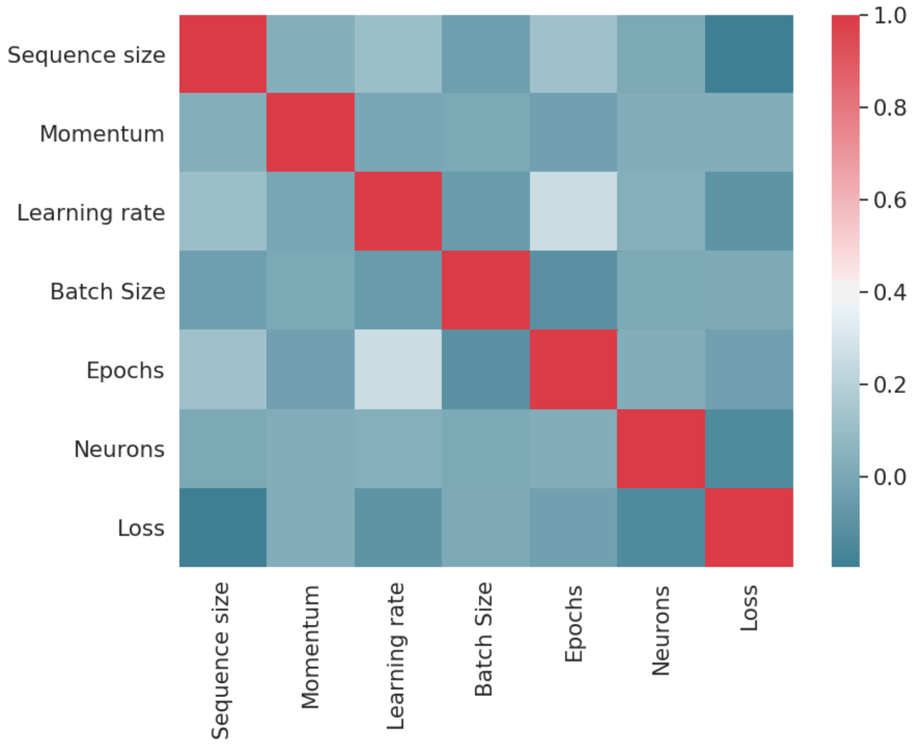


Figure 16: Pairwise NODE hyperparameter correlations for the Burgers turbulence case.

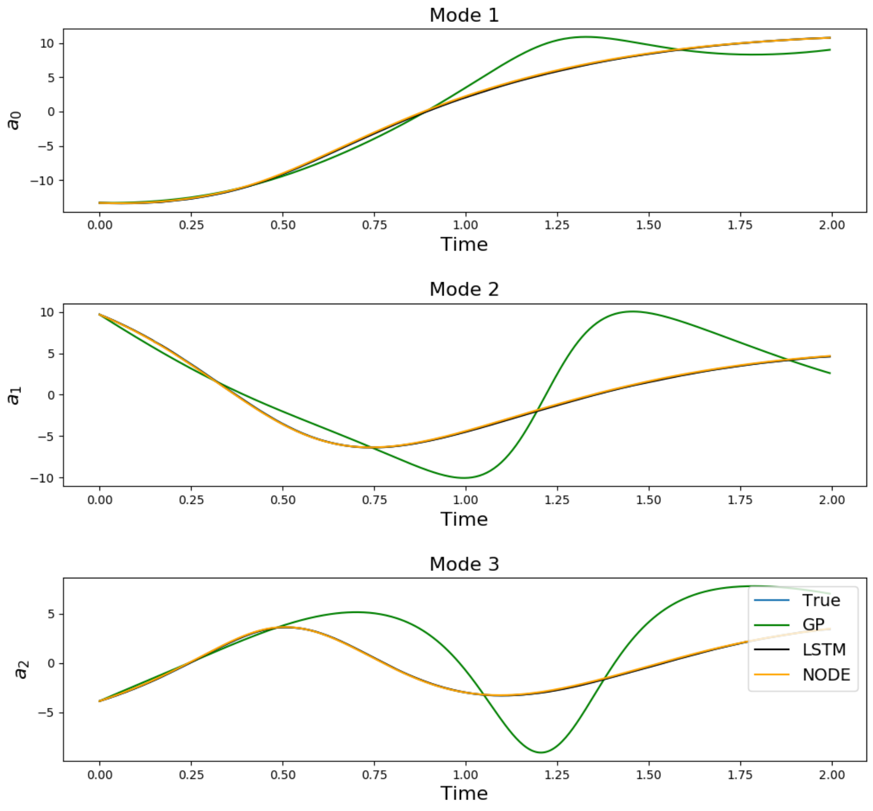
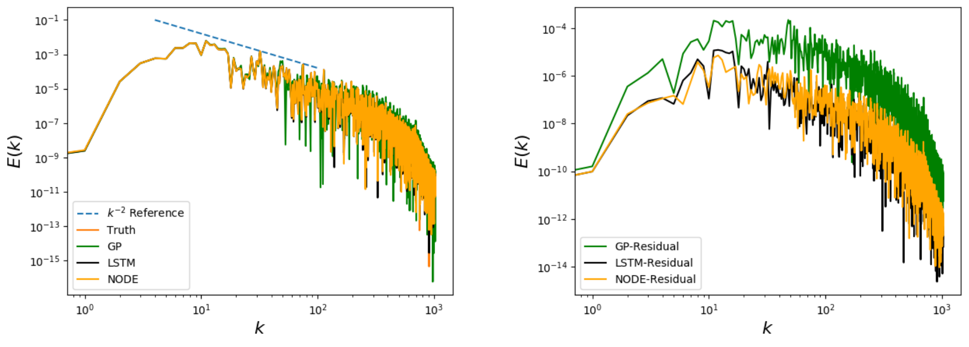


Figure 17: POD-space coefficient evolution for the *Burgulence* case with improved hyper-parameter choices. The LSTM performance is significantly improved.



(a) Spectra (b) Residual

Figure 18: Kinetic-energy spectra predictions (left) and their residuals (right) as predicted by NODE and LSTM deployed with optimal hyperparameters.

4. Discussion and conclusions

We have investigated using LSTMs and NODEs as nonintrusive learning models for the projections of nonlinear partial differential equations to a latent-space spanned by severely truncated POD modes. We note that the choice of the POD modes (which form a linear subspace) also ensures the applicability of the symmetries of the PDE-governed solution on the machine-learned predictions.

We tested our ideas on two test cases governed by the viscous Burgers equation, with the first exhibiting an advecting shock and the second displaying a multiscale nature in full-order space. Both LSTM and NODE formulations are seen to learn the transient nature of our systems in the reduced space: they exploit the sequential nature of data and provide an implicit closure. In the second case, we also utilize DeepHyper, a scalable Bayesian optimization package for an improved hyperparameter configuration choice, in order to obtain superior performance for LSTM. The non-i.i.d. assumption of the data and associated learning allows for the embedding of a memory effect that provides for accurate coarse-grained evolution of modal coefficients in a manner similar to the Mori-Zwanzig formalism. Our assessments reveal that the machine learning techniques studied here are able to provide stable evolutions of the modal coefficients in comparison with their intrusive and unclosed counterpart (i.e., GP).

These methods have several potential limitations that provide a focus for future studies. For example, we have observed that nonintrusive ROMs constructed by using autoregressive machine learning models (such as the LSTM formulation presented here) are particularly susceptible to error accumulation during deployment. Extending these methods to long prediction horizons remains an important unanswered question. The current study fixes this length to be a 400 timestep simulation and avoids this issue. Moving beyond this range to somewhere around $O(4000)$ timesteps while retaining stable predictions is a difficult task to accomplish. The long-term stability of these machine learning methods may intuitively be linked with the underlying dynamics of the full-order model, but thorough empirical and rigorous studies need to be performed to establish these links. We also note that, in the absence of strong *a priori* physics constraints, these frameworks are unreliable in an extrapolatory sense (for instance, predicting in the future for previously unseen dynamics) and for parameteric regimes that do not correspond to the training data. With this point in mind, embedding uncertainty estimates into such methods thus is important. However, quantifying the uncertainty of a nonintrusive ROM remains computationally and mathematically challenging. The current training data were generated by using numerical simulations and were noise-free. However, the practical deployment of such ROMs needs to account for learning from noisy data in the presence of measurement uncertainty. In order to deal with this issue, machine learning methods are complemented by strategies such as the incorporation of artificial noise into the training data or the use of weight regularization during training. We note, however, that these strategies rarely behave in a universal manner, and significant problem-specific tuning may be needed. An avenue for exploration is to move beyond the use of POD and employ nonlinear embeddings to characterize the reduced space. This may aid with high-dimensional dynamics that are insufficiently captured by a set of truncated POD basis vectors; some early results are shown in [20]. Another interesting direction to improve shock resolution with training neural networks is by using wavelet bases of the data [46], due to their adaptive nature and suitability for equations with strong non-stationarity and discontinuities. oh We conclude by noting that ROM developments that incorporate history explicitly (such as in the LSTM) or implicitly (such as through a NODE) represent an attractive avenue for exploration for efficient reduced basis dynamics learning of systems that are advection-dominated.

Data availability

All the relevant data and codes for this study are provided in a public repository at `https://github.com/Romit-Maulik/ML_ROM_Closures`.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357. This research was funded in part and used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357. This project was also funded by the Los Alamos National Laboratory, 2019 LDRD grant "Machine Learning for Turbulence". This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. DOE or the United States Government.

References

[1] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations, Int. J. Numer. Meth. Eng. 86 (2011) 155–181.

[2] Z. Wang, I. Akhtar, J. Borggaard, T. Iliescu, Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison, Comput. Meth. Appl. M. 237 (2012) 10–26.

[3] O. San, J. Borggaard, Principal interval decomposition framework for POD reduced-order modeling of convective Boussinesq flows, Int. J. Numer. Meth. Fl. 78 (2015) 37–62.

[4] F. Ballarin, A. Manzoni, A. Quarteroni, G. Rozza, Supremizer stabilization of POD-Galerkin approximation of parametrized steady incompressible Navier-Stokes equations, Int. J. Numer. Meth. Eng. 102 (2015) 1136–1161.

[5] O. San, R. Maulik, Extreme learning machine for reduced order modeling of turbulent geophysical flows, Phys. Rev. E 97 (2018) 042322.

[6] Q. Wang, J. S. Hesthaven, D. Ray, Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem, J. Comp. Phys. 384 (2019) 289–307.

[7] Y. Choi, K. Carlberg, Space-time least-squares Petrov–Galerkin projection for nonlinear model reduction, SIAM J. Sci. Comput. 41 (2019) A26–A58.

[8] R. Maulik, V. Rao, S. Madireddy, B. Lusch, P. Balaprakash, Using recurrent neural networks for nonlinear component computation in advection-dominated reduced-order models, Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada (2019).

[9] J. L. Proctor, S. L. Brunton, J. N. Kutz, Dynamic mode decomposition with control, SIAM J. Appl. Dyn. Syst. 15 (2016) 142–161.

[10] B. Peherstorfer, K. Willcox, M. Gunzburger, Optimal model management for multifidelity Monte Carlo estimation, SIAM J. Sci. Comput. 38 (2016) A3163–A3194.

[11] T. P. Sapsis, A. J. Majda, Statistically accurate low-order models for uncertainty quantification in turbulent dynamical systems, P. Natl. Acad. Sci. USA 110 (2013) 13705–13710.

[12] M. J. Zahr, K. T. Carlberg, D. P. Kouri, An efficient, globally convergent method for optimization under uncertainty using adaptive model reduction and sparse grids, arXiv preprint arXiv:1811.00177 (2018).

[13] D. Wells, Z. Wang, X. Xie, T. Iliescu, An evolve-then-filter regularized reduced order model for convection-dominated flows, Int. J. Numer. Meth. Fl. 84 (2017) 598–615.

[14] X. Xie, M. Mohebujjaman, L. G. Rebholz, T. Iliescu, Data-driven filtered reduced order modeling of fluid flows, SIAM J. Sci. Comput 40 (2018) B834–B857.

[15] O. San, R. Maulik, Neural network closures for nonlinear model order reduction, Adv. Comput. Math. 44 (2018) 1717–1750.

[16] O. San, R. Maulik, M. Ahmed, An artificial neural network framework for reduced order modeling of transient flows, Commun. Nonlinear Sci. (2019).

[17] D. Kosambi, Statistics in function space, J Indian Math. Soc. 7 (1943) 76–88.

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

[19] K. Taira, M. S. Hemati, S. L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S. Dawson, C.-A. Yeh, Modal analysis of fluid flows: Applications and outlook, arXiv preprint arXiv:1903.05750 (2019).

[20] K. Lee, K. T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, Journal of Computational Physics (2019) 108973.

[21] B. Lusch, J. N. Kutz, S. L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nature Comm. 9 (2018) 4950.

[22] A. T. Mohan, D. V. Gaitonde, A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks, arXiv preprint arXiv:1804.09269 (2018).

[23] M. Guo, J. S. Hesthaven, Data-driven reduced order modeling for time-dependent problems, Comput. Meth. Appl. M. 345 (2019) 75–99.

[24] M. Mohebujjaman, L. G. Rebholz, T. Iliescu, Physically constrained data-driven correction for reduced-order modeling of fluid flows, Int. J. Numer. Meth. Fl. 89 (2019) 103–122.

[25] K. Yeo, I. Melnyk, Deep learning algorithm for data-driven simulation of noisy dynamical system, J. Comp. Phys. 376 (2019) 1212–1231.

[26] A. Mohan, D. Daniel, M. Chertkov, D. Livescu, Compressed Convolutional LSTM: An Efficient Deep Learning framework to Model High Fidelity 3D Turbulence, arXiv preprint arXiv:1903.00033 (2019).

[27] S. L. Brunton, J. L. Proctor, J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, P. Natl. Acad. Sci. USA 113 (2016) 3932–3937.

[28] J. N. Kutz, S. L. Brunton, B. W. Brunton, J. L. Proctor, Dynamic mode decomposition: data-driven modeling of complex systems, SIAM, 2016.

[29] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Adv. 3 (2017) e1602614.

[30] K. Champion, B. Lusch, J. N. Kutz, S. L. Brunton, Data-driven discovery of coordinates and governing equations, arXiv preprint arXiv:1904.02107 (2019).

[31] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics informed deep learning (Part II): Data-driven discovery of nonlinear partial differential equations, arXiv preprint arXiv:1711.10566 (2017).

[32] C. Ma, J. Wang, et al., Model reduction with memory and the machine learning of dynamical systems, arXiv preprint arXiv:1808.04258 (2018).

[33] Q. Wang, N. Ripamonti, J. S. Hesthaven, Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism, Technical Report, 2019.

[34] J. Bec, K. Khanin, Burgers turbulence, Phys. Rep. 447 (2007) 1–66.

[35] R. Maulik, O. San, Explicit and implicit LES closures for Burgers turbulence, J. Comp. Appl. Math 327 (2018) 12–40.

[36] E. Haber, L. Ruthotto, Stable architectures for deep neural networks, Inverse Probl. 34 (2017) 014004.

[37] L. Ruthotto, E. Haber, Deep neural networks motivated by partial differential equations, arXiv preprint arXiv:1804.04272 (2018).

[38] J. Behrmann, D. Duvenaud, J.-H. Jacobsen, Invertible residual networks, arXiv preprint arXiv:1811.00995 (2018).

[39] V. Reshniak, C. Webster, Robust learning with implicit residual networks, arXiv preprint arXiv:1905.10479 (2019).

[40] T. Q. Chen, Y. Rubanova, J. Bettencourt, D. K. Duvenaud, Neural ordinary differential equations, in: Adv. Neur. In., pp. 6571–6583.

[41] H. Mori, Transport, collective motion, and Brownian motion, Prog. Theor. Phys. 33 (1965) 423–455.

[42] R. Zwanzig, Nonlinear generalized Langevin equations, J. Stat. Phys. 9 (1973) 215–220.

[43] D. J. Evans, G. Morriss, Statistical mechanics of nonequilibrium liquids, Cambridge University Press, 2008.

[44] O. San, A. E. Staples, Stationary two-dimensional turbulence statistics using a Markovian forcing scheme, Compu. Fluids 71 (2013) 1–18.

[45] P. Balaprakash, M. Salim, T. Uram, V. Vishwanath, S. Wild, DeepHyper: Asynchronous hyperparameter search for deep neural networks, in: 2018 IEEE 25th International Conference on High Performance Computing (HiPC), IEEE, pp. 42–51.

[46] A. T. Mohan, D. Livescu, M. Chertkov, Wavelet-powered neural networks for turbulence, Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada (2019).