Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders

Romit Maulik[a], Bethany Lusch[a], Prasanna Balaprakash[b,a]

[a]*Argonne Leadership Computing Facility, Argonne National Laboratory, Lemont - IL, 60439*
[b]*Mathematics & Computer Science Division, Argonne National Laboratory, Lemont - IL, 60439*

Abstract

A common strategy for the dimensionality reduction of nonlinear partial differential equations relies on the use of the proper orthogonal decomposition (POD) to identify a reduced subspace and the Galerkin projection for evolving dynamics in this reduced space. However, advection-dominated PDEs are represented poorly by this methodology since the process of truncation discards important interactions between higher-order modes during time evolution. In this study, we demonstrate that an encoding using convolutional autoencoders (CAEs) followed by a reduced-space time evolution by recurrent neural networks overcomes this limitation effectively. We demonstrate that a truncated system of only two latent-space dimensions can reproduce a sharp advecting shock profile for the viscous Burgers equation with very low viscosities, and a six-dimensional latent space can recreate the evolution of the inviscid shallow water equations. Additionally, the proposed framework is extended to a parametric reduced-order model by directly embedding parametric information into the latent space to detect trends in system evolution. Our results show that these advection-dominated systems are more amenable to low-dimensional encoding and time evolution by a CAE and recurrent neural network combination than the POD-Galerkin technique.

Keywords: ROMs, Autoencoders, Recurrent neural networks

---

1. Introduction

High-fidelity simulations of systems characterized by nonlinear partial differential equations (PDEs) represent large compute costs and are prohibitive for decision-making tasks for many fast-query applications. In order to reduce costs, there has recently been significant interest in the reduced-order modeling (ROM) of such systems [1, 2, 3, 4, 5, 6, 7, 8, 9]. As such, this field finds extensive application in control [10, 11, 12, 13, 14, 15], optimization [16, 17], uncertainty quantification [18, 19] and data-assimilation [20, 21, 22] among others. However, ROMs are limited in how they handle nonlinear dependence and perform poorly for complex physical phenomena, which are inherently multiscale in space and time [23, 24, 25, 26]. Researchers continue to search for efficient and reliable ROM techniques for such transient nonlinear systems [27, 28, 29, 30, 31, 6]. The identification of a reduced basis to ensure a compressed representation that is minimally *lossy* is a core component of most ROM development strategies (some examples include [32, 33, 34]). Once this basis is identified, we need a cost-effective strategy for accurate nonlinear dynamical system evolution to reproduce the full-order spatiotemporal complexity of the problem in the reduced basis. For example, we could use intrusive methods (which project the governing equations onto the reduced-basis), as seen in [35, 36], which use a Galerkin projection or [1, 37, 38], which use the Petrov-Galerkin method (see [39] for the comparison of these two methods). Finally, reconstruction of the compressed representation is required for full-order space post-processing and visualization. In this study, we utilize convolutional autoencoders (CAE) and long short-term memory neural networks (LSTM) [40] for parametric ROMs of advection-dominated and inviscid systems. The former are used to identify reduced-representations of the full-order fields and the latter are a type of recurrent neural network used for the temporal evolution of these compressed representations. LSTMs have recently become popular for the latent-space temporal modeling of dynamical systems [41, 42, 43, 44, 45, 46] although most such studies obtain reduced representations through the use of linear embeddings such as the proper orthogonal decomposition (POD). Additionally, we propose a parametric extension of the CAE-LSTM for exploring parametric search spaces through training on multiple offline simulations. In contrast with studies outlined in [47, 48], we deploy our framework on problems requiring shock capturing mechanisms as well as for fully inviscid system simulations. Ref. [49] also uses a CAE to nonlinearly embed states, but solves the governing equations on the nonlinear manifold defined by the autoencoder. We reduce computation by evolving with an LSTM network instead. An important difference with another similar study outlined in [50] is that our framework allows for the explicit embedding of control parameters such as viscosity or a parameterization of the initial condition into the LSTM. This allows for the independent training of our CAE and LSTM neural networks. A similar implementation has been demonstrated in [51] where another neural network is used to link global parameters with the latent space representations. We simplify this by directly embedding the parameters into the latent space. In [52], the latent space was obtained using a CAE followed by which a Gaussian process regressor was utilized for a continous representation of the temporal evolution of the state. However, this method is limited when the number of snapshots is very large due to the inherent limitations of most Gaussian process regression techniques. We direct the readers to the recent work of Agostini [53] for an overview of basic concepts and algorithms for autoencoder applications to fluid dynamics.

Our forward models for the purpose of data generation and subsequent testing are given by parametric partial differential equations. For the rest of this article, we shall represent a generic partial differential equation using the following notation:

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

where $\Omega \subset \mathbb{R}^i$ is the spatial domain, $\mathcal{T} = [0,T]$ is the temporal domain, $\mathcal{P} \subset \mathbb{R}^p$ are physical parameters and $\mathcal{N}$, $\mathcal{L}$ are non-linear and linear operators, respectively. Our system is characterized by a solution field $\mathbf{q} : \Omega \times \mathcal{T} \times \mathcal{P} \to \mathbb{R}^d$ and appropriately chosen initial and boundary conditions, where i is the number of spatial dimensions, d is the number of dependent variables of the PDE, and p is the number of control parameters in the problem. We assume that our system of equations can be solved in space-time on a discrete grid resulting in the following system of parameterized ODEs:

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

where $\mathbf{q}_h : \mathcal{T} \times \mathcal{P} \to \mathbb{R}^{N_h}$ is a discrete solution and N_h is the number of spatial degrees of freedom. In this problem, our goal is to bypass the solution of Equation 1 by constructing a compression manifold and a time advancement technique on this manifold solely from training data. Such

ROMs hold great promise for characterizing the spatiotemporal dynamics of systems for which observations may be available, but little knowledge of the governing equations exist.

To summarize, the contributions of this article are:

- We propose a deep learning based emulation strategy for nonlinear partial differential equations. We introduce a convolutional autoencoder architecture that obtains nonlinear embeddings with high compression ratios and we make use of long short-term memory networks for the evolution of the state in embedded space.
- We make our emulator parametric by passing the parameters into the latent space. This allows generalization across, for example, a range of viscosities.
- We demonstrate the performance of the proposed formulation for non-intrusive modeling of advection dominated physics obtained from the viscous Burgers and inviscid shallow water equations.

2. The POD-Galerkin projection

The POD-Galerkin reduced-order modeling technique (POD-GP) relies on the construction of a dynamical system with far fewer degrees of freedom. It accomplishes this by performing a Galerkin projection of the original (PDE) system onto a reduced basis which is optimal for linearly compressing the spatiotemporal dynamics of the system. This reduced basis (or latent space) may be obtained by using the POD compression technique. Here the number of POD modes obtained by the method of snapshots [54, 55] can be truncated to capture a large fraction of the total variance in the system. Further details of the POD basis computation can be found in Appendix B.

The POD basis may be leveraged for a Galerkin projection of each partial differential equation forming the coupled system onto its corresponding reduced basis. We start by revisiting Equation (1) as follows,

$$\dot{\mathbf{q}}_h(\mathbf{x}, t, \mathbf{p}) + \boldsymbol{N}_h[\mathbf{q}_h(\mathbf{x}, t, \mathbf{p})] + \boldsymbol{L}_h[\mathbf{q}_h(\mathbf{x}, t, \mathbf{p}); \mathbf{p}] = 0, \tag{3}$$

which can be expressed in the reduced basis as

$$\boldsymbol{\psi}\dot{\mathbf{q}}_r(t, \mathbf{p}) + \boldsymbol{N}_h[\boldsymbol{\psi}\mathbf{q}_r(t, \mathbf{p})] + \boldsymbol{L}_h[\boldsymbol{\psi}\mathbf{q}_r(t, \mathbf{p}); \mathbf{p}] = 0, \tag{4}$$

where $\mathbf{q}_r \in \mathbb{R}^{N_r}$ corresponds to the temporal coefficients at one time instant of the system evolution and $\boldsymbol{\psi}$ are the truncated POD bases. The orthogonal nature of the reduced basis can be leveraged to obtain

$$\dot{\mathbf{q}}_r(t, \mathbf{p}) + \boldsymbol{N}_h[\boldsymbol{\psi}\mathbf{q}_r(t, \mathbf{p})] + \boldsymbol{\mathcal{L}}_r[\mathbf{q}_r(t, \mathbf{p}); \mathbf{p}] = 0, \tag{5}$$

where $\boldsymbol{\mathcal{L}}_r$ is a precomputed linear operator in reduced space. This equation is denoted the POD Galerkin-projection formulation. Note that the nonlinear operator can also be modified to aid in fast computation in the reduced space if the nature of the nonlinear term is quadratic. In this article, we keep our treatment general. We have assumed that the residuals generated by the truncated representation of the full-order model are orthogonal to the reduced basis. A significant source of error in the forward evolution of this system of equations is due to the absence of higher-basis nonlinear interactions as shown in in Section 4.1. Also, POD-GP essentially consists of N_r coupled ODEs and is solved by a standard fourth-order accurate Runge-Kutta method. The reduced degrees of freedom lead to very efficient forward solves of the problem even though accuracy is limited. This transformed problem has initial conditions given by

$$\mathbf{q}_r(t = 0) = \left(\boldsymbol{\psi}^T \mathbf{q}_h(t = 0)\right). \tag{6}$$

3. Deep neural networks

In the following section, we introduce our deep neural network architectures for establishing a viable emulation strategy for data obtained from nonlinear partial differential equations.

3.1. Convolutional autoencoders

Autoencoders are neural networks that learn a new representation of the input data, usually with lower dimensionality. The initial layers, called the *encoder*, map the input $\mathbf{m} \in \mathbb{R}^m$ to a new representation $\mathbf{k} \in \mathbb{R}^k$ with $k << m$. The remaining layers, called the *decoder*, map $\mathbf{k}$ back to $\mathbb{R}^m$ with the goal of reconstructing $\mathbf{m}$. The objective is to minimize the reconstruction error. Autoencoders are unsupervised; the data $\mathbf{m}$ is given, but the representation $\mathbf{k}$ must be learned.

More specifically, we use autoencoders that have some convolutional layers. In a convolutional layer, instead of learning a matrix that connects all

m neurons of layer's input to all n neurons of the layer's output, we learn a set of filters. Each filter $\mathbf{f_i}$ is convolved with patches of the layer's input. Suppose a 1-d convolutional layer has filters of length m_{f_i}. Then each of the layer's output neurons corresponding to filter $\mathbf{f_i}$ is connected to a patch of m_{f_i} of the layer's input neurons. In particular, a 1-d convolution of filter $\mathbf{f}$ and patch $\mathbf{p}$ is defined as $\mathbf{f} * \mathbf{p} = \sum_j f_j p_j$ (For neural networks, convolutions are usually technically implemented as cross-correlations). Then, for a typical 1-d convolutional layer, the layer's output neuron $y_{ij} = \varphi(\mathbf{f_i} * \mathbf{p_j} + B_i)$ where φ is an activation function, and B_i are the entries of a bias term. As j increases, patches are shifted by stride s. For example, a 1-d convolutional layer with a filter f_0 of length $m_f = 3$ and stride $s = 1$ could be defined so that y_{0j} involves the convolution of f_0 and inputs $j-1, j$, and $j+1$. To calculate the convolution, it is common to add zeros around the inputs to a layer, which is called *zero padding*. In the decoder, we use deconvolutional layers to return to the original dimension. These layers upsample with nearest-neighbor interpolation.

Two-dimensional convolutions are defined similarly, but each filter and each patch are two-dimensional. A 2-d convolution sums over both dimensions, and patches are shifted both ways. For a typical 2-d convolutional layer, the output neuron $y_{hij} = \varphi(\mathbf{f_h} * \mathbf{p_{ij}} + B_h)$. Input data can also have a "channel" dimension, such as RGB for images. The convolutional operator sums over channel dimensions, but each patch contains all of the channels. The filters remain the same size as patches, so they can have different weights for different channels. It is common to follow a convolutional layer with a *pooling* layer, which outputs a sub-sampled version of the input. In this paper, we specifically use max-pooling layers. Each output of a max-pooling layer is connected to a patch of the input, and it returns the maximum value in the patch.

3.2. Long short-term memory networks

The LSTM network is a special type of a class of neural networks termed recurrent neural networks (or RNNs). It was introduced to consider time-delayed processes where events further 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 arbitrary input variable $\mathbf{a}$ are given by

$$\begin{aligned}
\text{input gate: } & \boldsymbol{G}_i = \boldsymbol{\varphi}_S \circ \mathcal{F}_i^{N_c}(\mathbf{a}), \\
\text{forget gate: } & \boldsymbol{G}_f = \boldsymbol{\varphi}_S \circ \mathcal{F}_f^{N_c}(\mathbf{a}), \\
\text{output gate: } & \boldsymbol{G}_o = \boldsymbol{\varphi}_S \circ \mathcal{F}_o^{N_c}(\mathbf{a}), \\
\text{internal state: } & \boldsymbol{s}_t = \boldsymbol{G}_f \odot \boldsymbol{s}_{t-1} + \boldsymbol{G}_i \odot \left(\boldsymbol{\varphi}_T \circ \mathcal{F}_{\mathbf{a}}^{N_c}(\mathbf{a})\right), \\
\text{output: } & \mathbf{h}_t = \boldsymbol{G}_o \circ \boldsymbol{\varphi}_T\left(\boldsymbol{s}_t\right),
\end{aligned} \tag{7}$$

where $\mathbf{a}$ is a vector of inputs comprising a snapshot of information in time. Within this study, this vector is generally the encoded representation after either the POD or CAE embedding. Also, $\boldsymbol{\varphi}_S$ and $\boldsymbol{\varphi}_T$ refer to tangent sigmoid and tangent hyperbolic activation functions respectively, N_c is the number of hidden layer units in the LSTM network. Here, $\mathcal{F}^n$ refers to a linear operation given by a matrix multiplication and subsequent bias addition i.e,

$$\mathcal{F}^n(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{B}, \tag{8}$$

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 $\mathbf{a} \odot \mathbf{b}$ refers to a Hadamard product of two vectors. The LSTM implementation is used to advance $\mathbf{a}$ as a function of time. The LSTM network's primary utility is the ability to control information flow through time with the use of the gating mechanisms. A quantity that preserves information of past inputs and predictions is the internal state $\mathbf{s}_t$ which is updated using the result of the input and forget gates every time the LSTM operations are performed. 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. Details of our LSTM deployments for the different experiments utilized in this article are provided in Section 4.

3.3. Combining CAE and LSTM for surrogate modeling

Our data-driven emulation strategy shall rely on the use of CAE for dimensionality reduction and LSTM for latent space temporal evolution of the state. The benefit of this formulation, in comparison to POD-GP, is the improved compression ratios obtained by the nonlinear embedding of the CAE and the equation-free evolution of the state using the LSTM. Here, the compression ratio is defined as the ratio of the degrees of freedom in the Cartesian space to that in the reduced space, given either by a POD-based truncation or a neural network based autoencoding.. The basic schematic for this formulation is shown in Figure 1 where a one-dimensional field is compressed to a low-dimensional latent space and then evolved non-intrusively. Our training framework is *separate* in that the snapshot data of the flow-field is first used to obtain a low dimensional embedding before a data-driven time-series forecast technique is used for evolving the state in this space. As mentioned previously, this is in contrast to previous studies where latent space embedding and temporal evolution have been performed in a simultaneous optimization [47, 50, 48]. We pursue this direction for our emulation strategy to allow for greater flexibility in modeling the evolution of the latent space. In particular, the choice for a novel state evolution mechanism will not require retraining a nonlinear embedding. In addition, uneven samples of snapshot data (for instance when new snapshots become available at arbitrary locations in time) may be deployed with time-series methods that are customized for irregular data without retraining an embedding [56]. The deployment of a concurrent optimization for an embedding and a time series forecast strategy usually relies on the construction of a loss-function that penalizes reconstruction and forecast accuracy together. The joint optimization can result in slower training and requires deciding how to weight the two loss functions. Specific details of the CAE and LSTM combinations for our test cases shall be described in Section 4.

4. Experiments

In the following, we introduce the two representative problems used to assess the proposed framework. We demonstrate framework performance for the viscous Burgers equation, which is characterized by an advecting shock and the conservative inviscid shallow water equations with varying initial conditions. While the first problem requires that our framework is able to capture the advection of a shock profile accurately in time, the second problem requires interpolation in initial condition space. These varying initial conditions are given by different locations of a Gaussian blob at the starting time. Note that all of our experiments are performed for non-dimensional variables only.

4.1. Burgers

Our first problem is given by the one-dimensional viscous Burgers' equation with Dirichlet boundary conditions which can be represented as

$$\begin{gathered} \dot{u} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \\ u(x,0) = u_0, \quad x \in [0,L], \quad u(0,t) = u(L,t) = 0. \end{gathered} \tag{9}$$

It is well known that the above equation is capable of generating discontinuous solutions even if initial conditions are smooth and ν is sufficiently small due to advection-dominated behavior. We specifically consider the initial condition

$$u(x,0) = \frac{x}{1 + \sqrt{\frac{1}{t_0}} \exp\left(Re\frac{x^2}{4}\right)}, \tag{10}$$

and we set $L = 1$ and maximum time $t_{max} = 2$. An analytical solution exists and is given by

$$u(x,t) = \frac{\frac{x}{t+1}}{1 + \sqrt{\frac{t+1}{t_0}} \exp\left(Re\frac{x^2}{4t+4}\right)}, \tag{11}$$

where $t_0 = \exp(Re/8)$ and $Re = 1/\nu$.

4.1.1. Convolutional autoencoder

We proceed by detailing the architecture of our CAE for effective compression of the full-order solution field. We use a one-dimensional convolutional framework with multiple strided filters to obtain a low-dimensional representation of the solution field. Figure 1 is a schematic of the architecture. We utilize several pairs of convolutional and max-pooling layers to reduce dimensionality of the input field to a size of *solely* two degrees of freedom in the encoded space. Following this, the two-dimensional state is convolved and upsampled several times to return to the dimensionality of the full-order field. Each one-dimensional convolutional layer utilizes a filter kernel of width 3 and a subsequent max-pooling reduces the grid degrees of freedom by a factor of two. Each layer also consists of rectified linear unit (ReLU) activations and utilizes a zero-padding at the edges of the domain for the purpose of convolution. The dynamics studied in this test case are not critically affected by the absence of accurate padding at the boundaries. Our network is trained by using a standard mean-squared error loss with a batch size of 10, a learning rate of 0.001 and the Adam optimizer. The choice of hyperparameters for this architecture (i.e., the number of layers, channels, latent-space dimension, learning rate and batch-size) were manually tuned to obtain the current performance. Also, each convolutional layer in the autoencoder utilized a ReLU activation function, with the exception of the output layer and the final layer of the encoder. No regularization was used in the process of training this model and approximately 10% of the total (non-test) data was randomly chosen and set aside for the purpose of validation. The validation data set in this, and subsequent, neural network trainings has been used for the purpose of early-stopping. Early-stopping is a technique to prevent the overfitting of the neural network model on a specific training data set. Overfitting may lead to poor generalization (on unseen data) when using deep neural network architectures. Early-stopping tracks the error on a validation data set and terminates the training after a pre-specified number of iterations (set at 10 in this study) for which this validation error is not seen to improve on its previous best value. The validation data is not used for updating the parameters of the neural network during the gradient-based optimization.

4.1.2. LSTM

In this section, we introduce architectural details of the LSTM used to advance latent space representations obtained by the CAE for the Burgers problem. We shall be outlining results from two different LSTM architectures: one that is valid for only one choice of ν and one that is valid for parametric interpolation. We observe that, in general, the latter requires more complex models.

Our basic LSTM architecture for this test case is two cells that are stacked on top of a windowed input of latent space representations. This leads to a windowed-input advancement of dynamics with the output being the prediction of the latent space representation at the next time step. This prediction is then fed back into the framework in an autoregressive manner. Our learning rate for the LSTM is the default 0.001, and we use the Adam optimizer for training. As in the case of the CAE, our cost function is the mean-squared error between predictions and targets. The LSTM hidden cells contain 20 neurons, and the batch size is set to 64 samples. Like in the case of the CAE, we do not employ any regularization and 10% of the snapshot data is

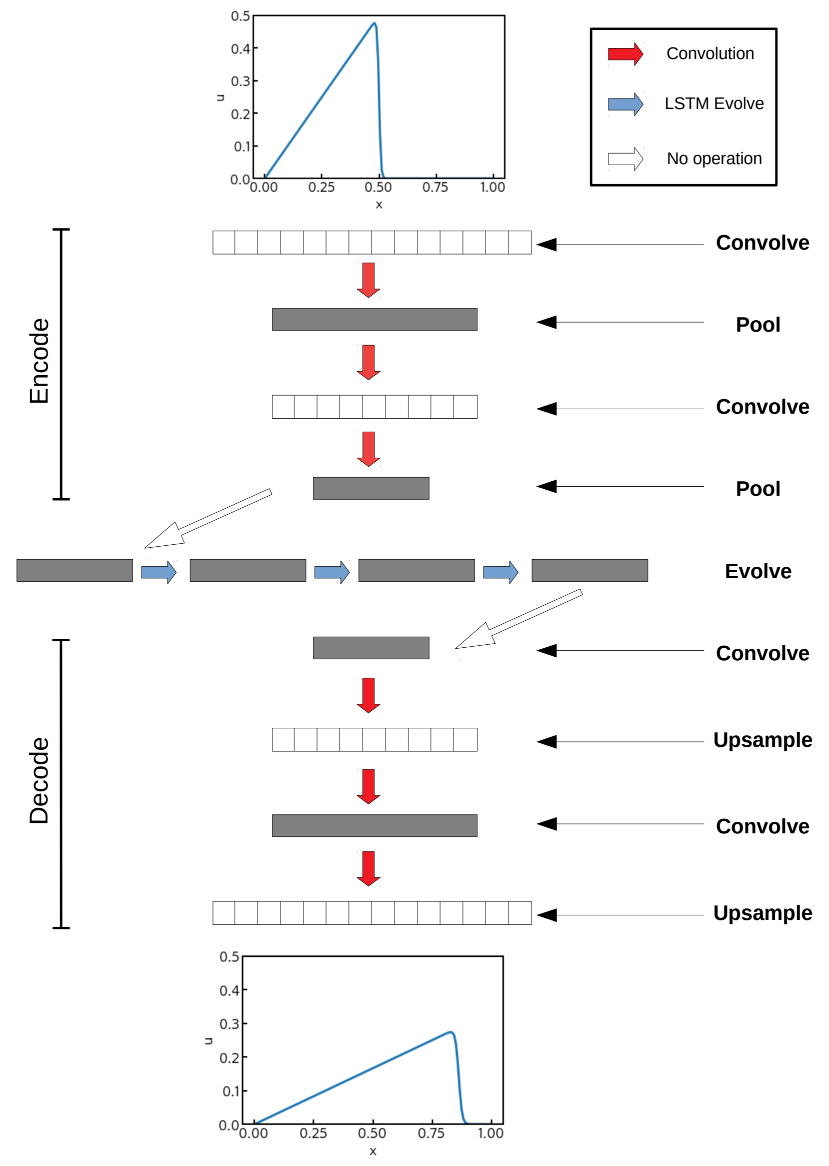


Figure 1: A schematic of the one-dimensional CAE-LSTM for the viscous Burgers equation. The nonlinear autoencoder embeds the data into latent space, and then the recurrent network can be used for time-series advancement.

set aside for the purpose of validation. The trained LSTM is deployed for emulating the evolution of the same data in a recursive fashion (i.e., outputs from the LSTM are used as inputs at the next time step).

4.1.3. CAE-LSTM modeling

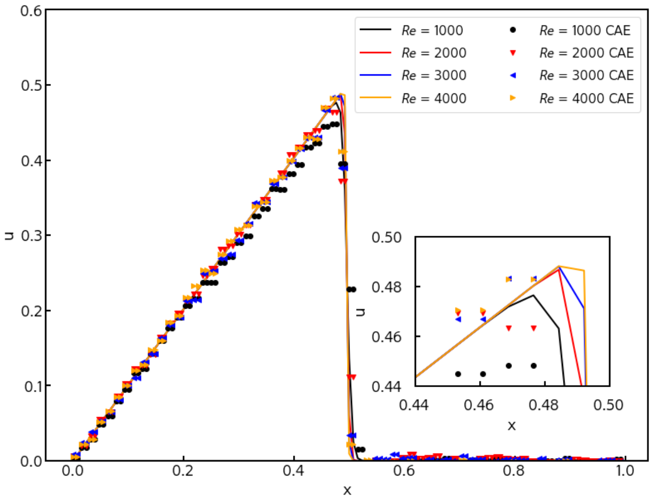
We assess the proposed framework on multiple datasets, each with a single value of ν. Solution fields that vary in time are generated using the analytical solution described in Equation 11. In this set of tests, we check the accuracy for different physics ranging from more dissipative at high values of viscosity to more advective at lower values. Error metrics and latent space visualization are provided to evaluate if any trends emerge that generalize in physics. We select four values of $Re = 1000, 2000, 3000, 4000$, each with 400 snapshots of the solution field (360 training and 40 validation) uniformly distributed in time. For the purpose of comparison we also provide results from the POD-Galerkin projection methodology. We highlight that this first set of assessments precludes testing and merely establishes the overall modeling paradigm. Testing for parameteric interpolation is performed subsequent to these assessments.

Figure 2 outlines the reconstruction performance of the CAE for various values of Re. As the value of Re increases, a sharper profile is obtained for the discontinuity as it advects in the positive x direction. The inset figures show that some remnant oscillations are still observed in the vicinity of the discontinuity but the behavior of the reconstruction in that region is consistent across variations of the control parameter. In other words, the reconstruction is smoother if the value of Re is relatively low. In addition, the behavior of the 2-dimensional encoding is also shown for varying control parameter values in Figure 3. The latent space consisting of two variables has a consistent trend in time which is repeated for other parameters. The y-axes of these plots, denoted by 'latent space magnitude,' indicate the value of the latent space encoding for a particular dimension in the bottleneck layer of the CAE. Notice the difference in magnitude of the latent space variables at the final snapshot. Trajectories at the highest values of Re behave in a similar manner indicating an asymptotic behavior as the diffusive behavior of the viscous Burgers equation reaches a high-Re limit. A thorough investigation of interpretability in latent space, though important, is beyond the scope of this article and is an active area of research. Note that at this point, we have not deployed any latent space model and these assessments are purely related to the CAE. In the following, we incorporate a latent space time-series model to obtain a 2 degree-of-freedom dynamical model of the advecting shock profile.

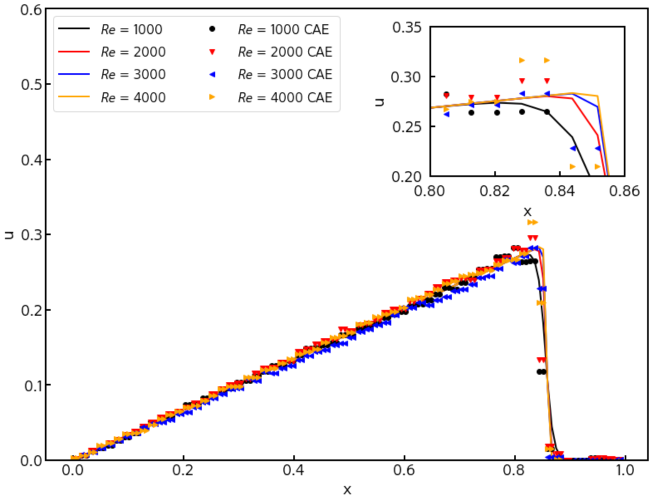
We now assess the ability of the proposed framework to mimic a standard reduced-order model. We start with an assessment of the POD-GP implementations at different values of Re as shown in Figure 4. The linear encoding leads to slow convergence of ROM representations to the shock profile. In addition, we also observe high frequency instabilities as the number of retained POD modes is increased for higher values of Re. This manifests itself in a solution that fails to converge at $Re = 4000$ for 30 retained modes and highlights a critical issue with the reduced-order modeling of advection-dominated problems. While the increasing frequencies of the Gibbs oscillations with modal retention represent the slow convergence to the true solution in the global POD basis, the failure of the method at 30 retained modes is due to the use of central differences in the calculation of the quadratic nonlinearity. This may be mitigated through the use of gradient calculation with shock capturing schemes, although these are more computationally expensive. The results in Figure 4 highlight the limitation of 'energy-based' truncation of the system in the POD basis for advection-dominated problems. The greater retention requirement diminishes compression ratios and surrogate effectiveness. In contrast, for problems where the physics is diffusion dominated, POD-GP is a superior alternative. Each POD-GP deployment utilized basis vectors from their respective full-order models.

In comparison, we show results from the CAE-LSTM implementation in Figure 5 which shows the ability of the proposed framework to capture the sharp profile advection with only two degrees of freedom. While Gibbs' phenomenon may still be visible (for instance at $t = 2.0$), the solution profile is captured much more accurately. Figure 6 shows the prediction of the latent-space model in comparison to the latent space representation by compressing each of the true snapshots. The evolution in the encoded space is recursive, in that the outputs of the LSTM are fed back into the input layer through a windowed input to obtain single time step output. The window is initialized with the true values of the first 10 time steps which implies that, in practice, a short duration of the simulation must be deployed with a full-order model following which the CAE-LSTM can take over non-intrusively. Research is underway to bypass this limitation by appending ghost-points in time to the training data in latent space to mimic a *burn-in* for the windowed input.

We perform assessments for the CAE-LSTM (as outlined for $Re = 4000$ above) for other parameter choices and report error metrics (given by L_2-norms at the final time step) in Table 1. These show the accuracy of the



(a) $t = 0$



(b) $t = 2.0$

Figure 2: Reconstruction ability of the CAE for initial condition (top) and the final field (bottom) for varying values of Re in the viscous Burgers equation.

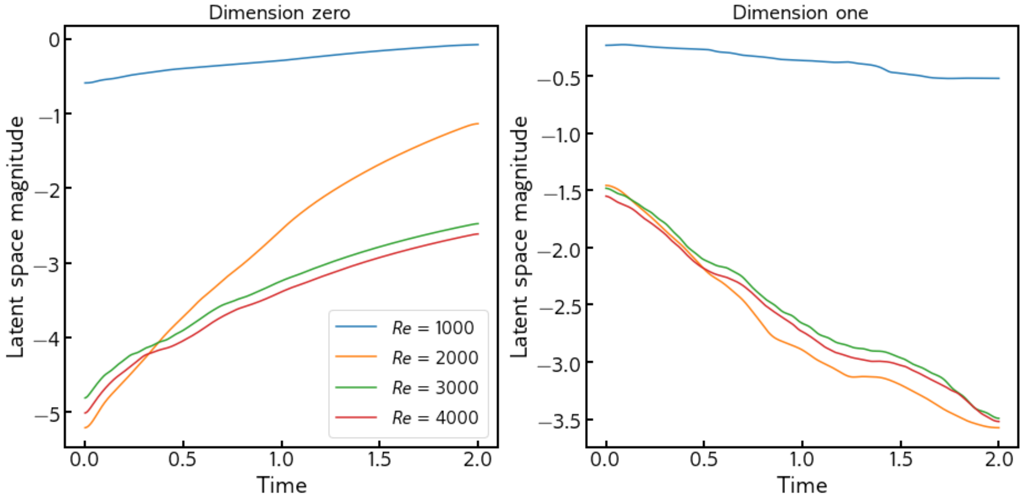
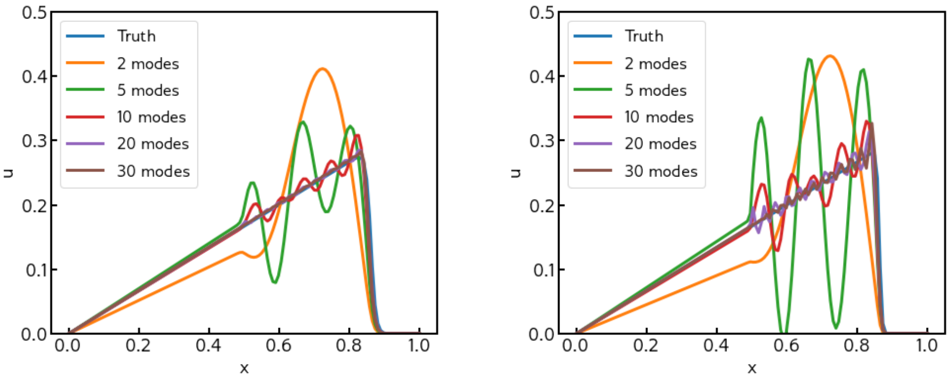


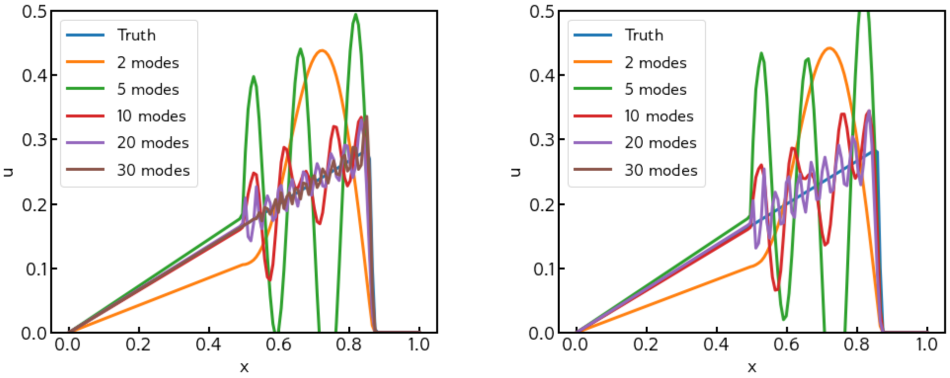
Figure 3: Evolution of the latent space for the CAE applied to the viscous Burgers equations for varying values of Re. The y-axes of both these plots refer to the value of the latent space encoding in the bottleneck layer of the CAE.

framework when compared to the POD-GP method for different POD mode retentions. In general, when dynamics are more advective, the CAE-LSTM has lower errors due to the self-similarity in the advecting shock profile. In comparison, the POD-GP method shows an order of magnitude greater errors at a comparable compression of 2 modes and displays trouble in dealing with strong advective physics for $Re = 4000$. Also, the CAE-LSTM, while unable to match POD-GP accuracies at greater mode retentions and lower Re, obtains an order of magnitude lower error across different Re for the same latent space dimensions (two degrees of freedom only). This establishes, empirically, that advective physics benefits from nonlinear encoding in space and nonlinear modeling in time for effective surrogates. Table 1 is complementary to Figure 4. While POD-GP shows greater oscillations even at high modal coefficient retention, overall L_2-error metrics are comparable (if not better) to the proposed framework. This is elaborated in Figure 7, which shows that the CAE results in noise in the reconstructed fields even if the oscillations due to the POD-GP implementation are stabilized.

We now extend the CAE-LSTM for parametric interpolation. By training the framework for full-order datasets generated for different Re, our framework can interpolate in a physical regime for quick generation of full-order

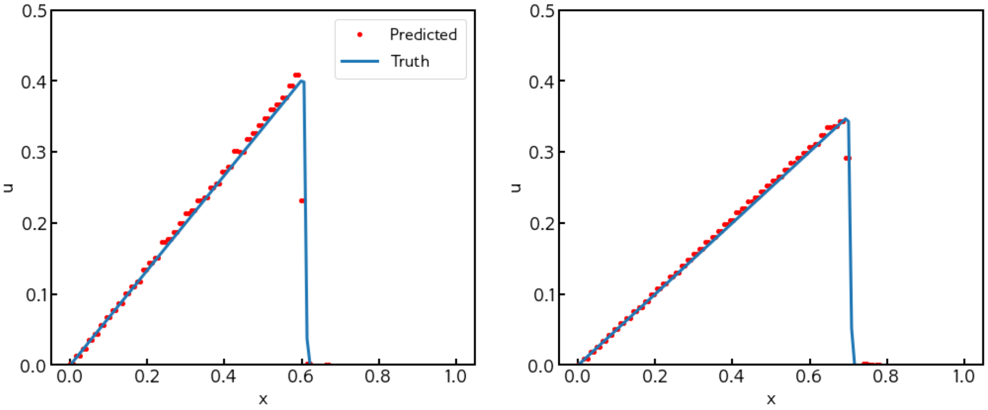


(a) $Re = 1000$ (b) $Re = 2000$

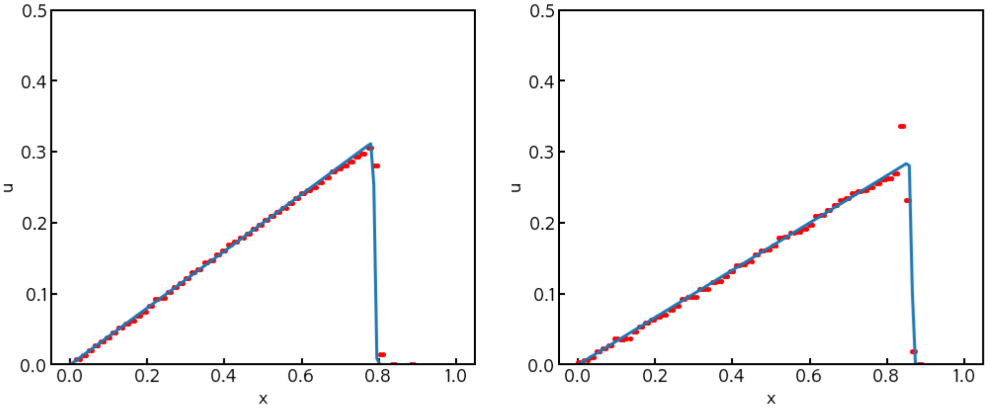


(c) $Re = 3000$ (d) $Re = 4000$

Figure 4: A demonstration of the limitations of the POD-Galerkin methods for building surrogates of advection dominated partial differential equations. Convergence to the true solution is slow and often limited by numerical instability.



(a) $t = 0.5$ (b) $t = 1.0$



(c) $t = 1.5$ (d) $t = 2.0$

Figure 5: Reduced-order modeling capability of the CAE for $Re = 4000$ showing evolution in physical space. We remind the reader that the system evolution has been performed using an LSTM in latent space, and these images are reconstructed from two degrees of freedom representations.

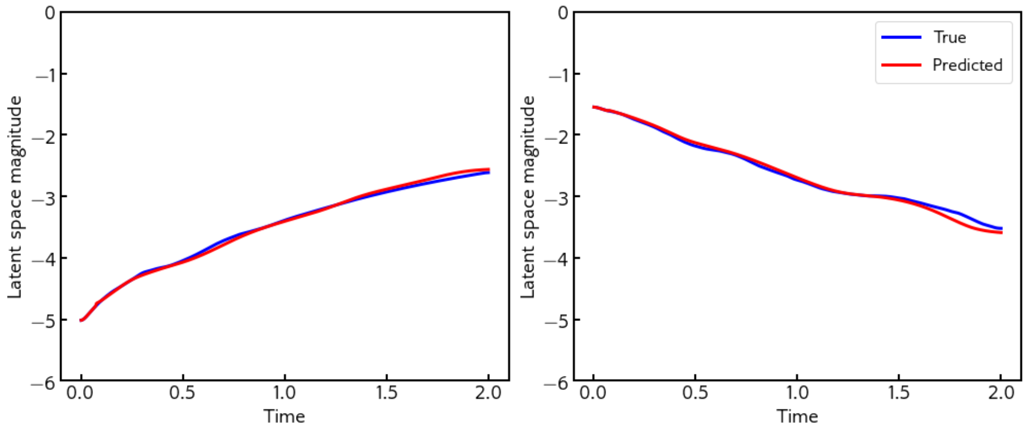


Figure 6: Learning dynamics in latent space obtained using CAE for $Re = 4000$. The y-axes indicate the magnitudes of the first (left) and second (right) latent space encoding.

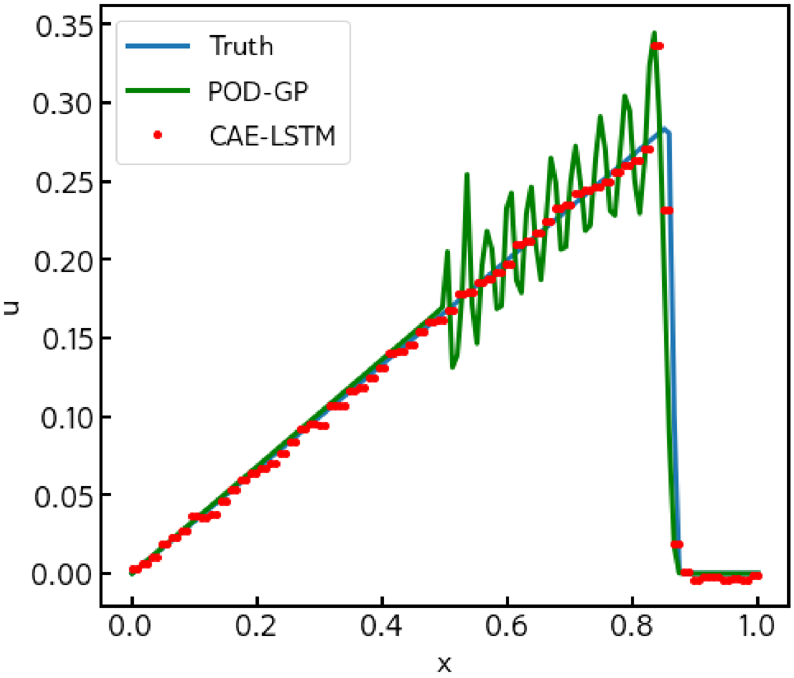


Figure 7: A direct comparison of the POD-GP and CAE-LSTM methods for $Re = 4000$at $t = 2.0$ where one can observe noise in the CAE-LSTM predictions even if a large portion of the Gibbs' oscillations are stabilized.

	Re = 1000	Re = 2000	Re = 3000	Re = 4000
GP 2 modes	4.197e-3	5.558e-3	6.12e-3	6.418e-3
GP 5 modes	1.57e-3	7e-3	1.244e-2	1.65e-2
GP 10 modes	1.497e-4	5.047e-4	1.063e-3	1.525e-3
GP 20 modes	4.607e-5	1.679e-4	4.099e-4	7.336e-4
GP 30 modes	4.938e-5	1.102e-4	8.333e-5	NaN
CAE LSTM	4.181e-4	3.912e-4	1.409e-4	1.551e-4

Table 1: L_2-norm error metrics for the final time reconstructions of the CAE-LSTM compared against POD-GP. This table outlines results where the CAE-LSTM and POD-GP deployments are trained anew for each Re. The CAE-LSTM error is lower for comparable compression (two degrees of freedom).

dynamics at novel parameter choices. We achieve this by appending another scalar component, the viscosity, to the latent space dimension. For training, we obtain snapshots from 19 simulations (i.e., with uniformly varying values of Re) and train a common CAE for all of the simulations. This lets us obtain a sequence of latent space representations for each full-order model concatenated with their respective viscosities. We then train an LSTM, also common across all of the simulations, on these sequences in the same manner as the previous experiments. Inferences can then be performed at a novel parameter choice with ease.

Our parametric LSTM has a similar architecture to the one we used for single-parameter data in the previous sections. The differences include 40 neurons in the hidden cells and a smaller batch size of 32. We remark that the CAE is identical to the one used previously. The performance of the CAE to reconstruct fields with varying dissipation on the shocked profiles is shown in Figure 8. The latent space representation of 2 degrees of freedom is expressive enough to capture the difference in the sharpness of the discontinuity for different viscosities. A parametric LSTM is then trained on these compressed representations with results as shown in Figure 9. We observe that the trends are reproduced appropriately for parameters that were not a part of the training data set. Finally, in Figure 10, we demonstrate that reconstructing full-order dynamics for a novel testing parameter accurately adheres to the true solution over time. The final time reconstruction mean-squared errors averaged across different testing viscosities was found to be $1.17e-4$, which was comparable to the cases where training was performed for solely one viscosity. Training and validation mean-squared-errors were $9.79e-05$ and

$1.38e-4$ respectively. We remark that the similarity between training errors in the single-parameter cases introduced previously and the testing error in the multiparameter case also indicates that our early stopping criterion is successfully preventing overfitting.

4.2. Shallow water equations

Our two-dimensional assessments utilize the inviscid shallow water equations which are a prototypical system of equations for geophysical flows. The governing equations are hyperbolic in nature and are

$$\begin{aligned}
\frac{\partial(\rho\eta)}{\partial t} + \frac{\partial(\rho\eta v_x)}{\partial x} + \frac{\partial(\rho\eta v_y)}{\partial y} &= 0, \\
\frac{\partial(\rho\eta v_x)}{\partial t} + \frac{\partial}{\partial x}\left(\rho\eta v_x^2 + \frac{1}{2}\rho g\eta^2\right) + \frac{\partial(\rho\eta v_x v_y)}{\partial y} &= 0, \\
\frac{\partial(\rho\eta v_y)}{\partial t} + \frac{\partial(\rho\eta v_x v_y)}{\partial x} + \frac{\partial}{\partial y}\left(\rho\eta v_y^2 + \frac{1}{2}\rho g\eta^2\right) &= 0.
\end{aligned} \tag{12}$$

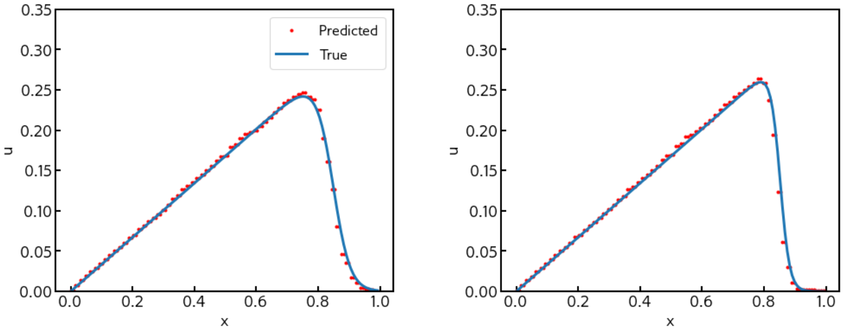
In the above set of equations, η corresponds to the total fluid column height, and (v_x, v_y) is the fluid's horizontal flow velocity, averaged across the vertical column. Further g is acceleration due to gravity, and ρ is the fluid density, which we fix at 1.0. The first equation captures the law of mass conservation whereas the second two denote the conservation of momentum. Our initial conditions are

$$\rho\eta(x, y, t=0) = e^{-\left(\frac{(x-\bar{x})^2}{2(5e+4)^2} + \frac{(y-\bar{y})^2}{2(5e+4)^2}\right)}, \tag{13}$$

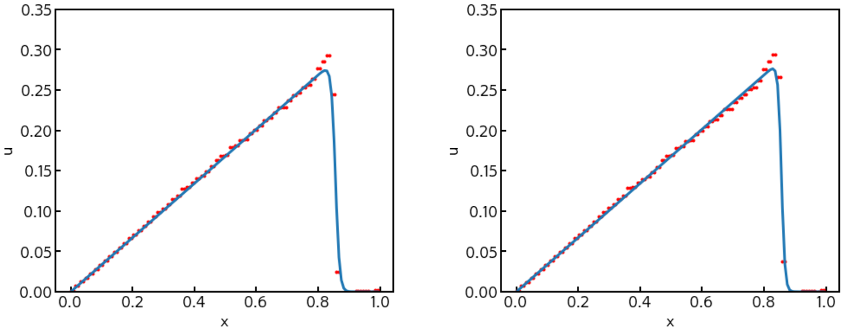
$$\rho\eta u(x, y, t=0) = 0, \tag{14}$$

$$\rho\eta v(x, y, t=0) = 0, \tag{15}$$

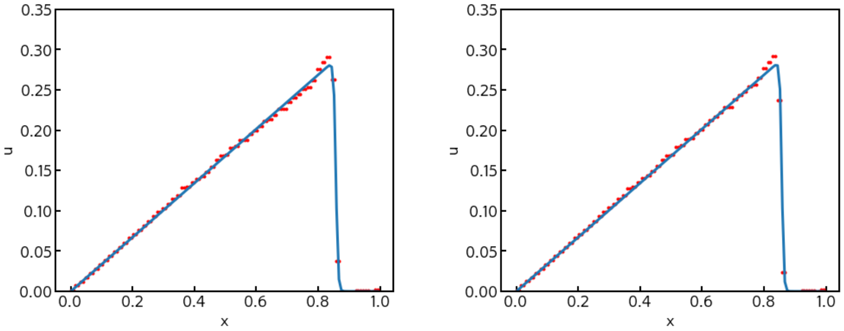
while our two-dimensional domain is a square with periodic boundary conditions. We generate data with full-order solves of the above system of equations until $t = 0.5$ with a time step of 0.001. Our full-order model uses a 4^{th}-order accurate Runge-Kutta temporal integration scheme and a fifth-order accurate weighted essentially non-oscillatory scheme (WENO) [57] for computing state reconstructions at cell faces. The Rusanov Reimann solver is utilized for flux reconstruction after cell-face quantities are calculated. The reader is directed to [58] for greater discussion of the temporal integration scheme and [59] for details on WENO and the Riemann solver implementation in two-dimensional problems. For ease of notation we denote $\rho\eta$ as q_1,



(a) $Re = 250$ (b) $Re = 450$



(c) $Re = 650$ (d) $Re = 850$



(e) $Re = 1050$ (f) $Re = 1250$

Figure 8: The ability of a CAE to reconstruct fields sampled from different parameters (Reynolds numbers) showing different sharpness in shock profiles. These snapshots are for parameters that were not included in the training dataset and are obtained by evolving only in the latent space.

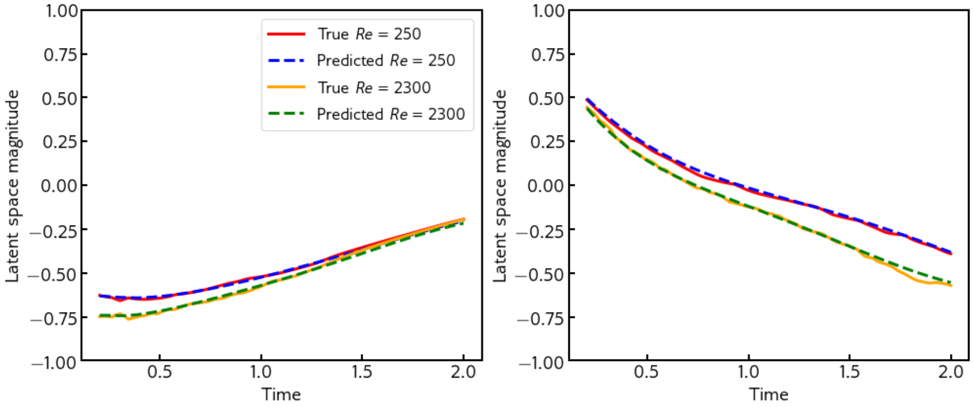
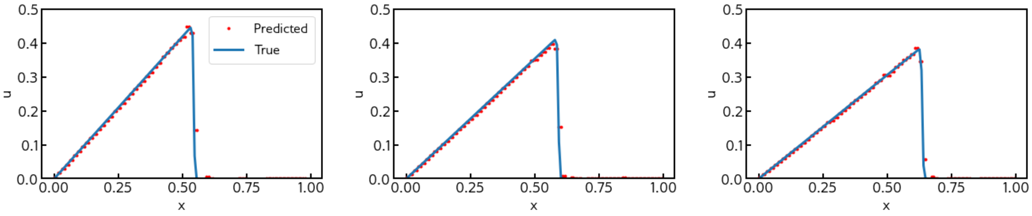


Figure 9: The ability for the parametric LSTM to learn latent space trends for different parameters that are not a part of the training data set. The y-axes indicate the magnitudes of the latent space encoding.

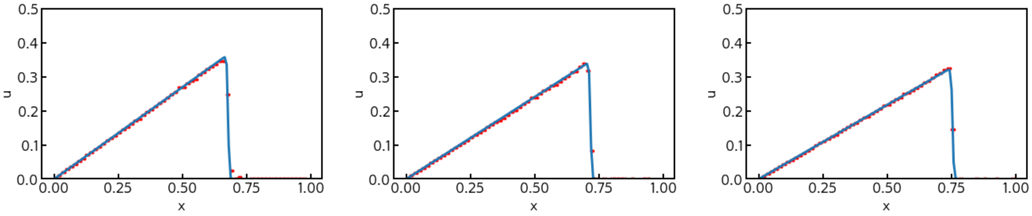
$\rho\eta u$ as q_2 and $\rho\eta v$ as q_3 in our subsequent discussions. The control parameters in the case of the shallow water equations are $\bar{x}$ and $\bar{y}$, which control the initial location of the Gaussian pulse in the domain. Our goal is to obtain a reduced-basis evolution of a new choice for these control parameters given *a priori* snapshots from full-order forward solves at pre-selected control parameter choices. We use 90 full-order simulations for the training and validation and 10 test simulations for *a posteriori* assessments. One hundred snapshots are utilized for each simulation, i.e., a snapshot is saved every five steps of the time integrator.

4.2.1. Convolutional autoencoder

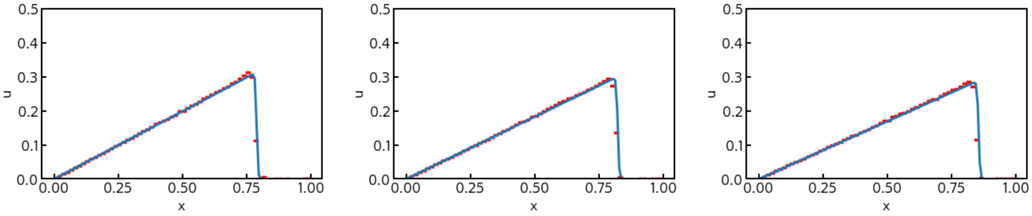
For the nonlinear encoding of the shallow water equations, we use the two-dimensional CAE detailed in the schematic in Figure 11. Our three conserved variables are encoded using three input and output channels in our autoencoder. We scaled the data to zero mean and unit variance to ensure that losses due to inaccurate reconstruction were weighted fairly across the different variables. We use an architecture that is similar to the Burgers' example in that a bottlenecked framework ensures the compression of the full-order field. A key difference is that the "bottleneck" layers are supplemented with fully connected layers to allow for an arbitrary latent dimensionality. We choose a latent space of 6 degrees of freedom for this problem which represents an approximate compression ratio of 680. Also, a batch size of



(a) $t = 0.2$ (b) $t = 0.4$ (c) $t = 0.6$



(d) $t = 0.8$ (e) $t = 1.0$ (f) $t = 1.2$



(g) $t = 1.4$ (h) $t = 1.6$ (i) $t = 1.8$

Figure 10: An example ROM characterized by the CAE-LSTM compared to the full-order solution. This parameter was not a part of the training dataset for either the CAE or the parametric LSTM.

24 with a learning rate of 0.001 was utilized to train the framework. Each two-dimensional convolutional layer and densely connected bottleneck layer utilized the Swish activation function, which has been shown to be superior to ReLU for significantly deep networks [60]. In contrast, the output layer of the network is a linear layer. 9000 snapshots are randomly partitioned into 8100 for training and 900 for validation, with the latter used for an early-stopping criterion. The trained CAE is tested on 1000 snapshots from 10 held-out simulations. The training and validation mean-squared-error losses for the CAE optimization were 0.1016 and 0.1124 respectively when training was terminated by the early stopping criterion. Note that these metrics were obtained with the data being scaled as mentioned previously. Specific details of the architecture such as the number of channels in each layer of the CAE and the size of the pooling may be found in our supporting code. Figure 12 shows the ability of the decoder to reconstruct from the latent space.

4.2.2. LSTM

We again couple the CAE with an LSTM that is conditioned on the control parameters. In this set of experiments, our control parameter affects the location of a Gaussian pulse applied to $\rho\eta$ at $t = 0$. Our goal is to replicate trends of field evolution for a novel initial condition given examples of full-order forward solves to train from. To create training data for the LSTM, we apply the trained CAE to compress the data, then concatenate the parameter information. Our LSTM architecture is 3 cells with 50 neurons in each cell. A batch size of 64 is used with the default learning rate of 0.001 for the Adam optimizer. As outlined previously, 10% of the total non-test data is set aside for the purpose of validation and early-stopping. A time-window of 10 points, utilized for the LSTM forecasts, provided adequately accurate trajectory predictions in the latent space. The training and validation mean-squared-error losses for the LSTM optimization were 0.0153 and 0.0298 respectively when training was terminated by the early stopping criterion.

4.2.3. CAE-LSTM modeling

Figure 13 shows the ability of the LSTM module to reconstruct dynamical trends in the latent space for a sample test simulation. The reference truth for these curves has been obtained by reconstructing (with use of the CAE) full-order solutions for a test control parameter that was not utilized during training. One can observe that dynamical trends are replicated by the parameterized LSTM. Evolutionary trends towards the end of the dy-

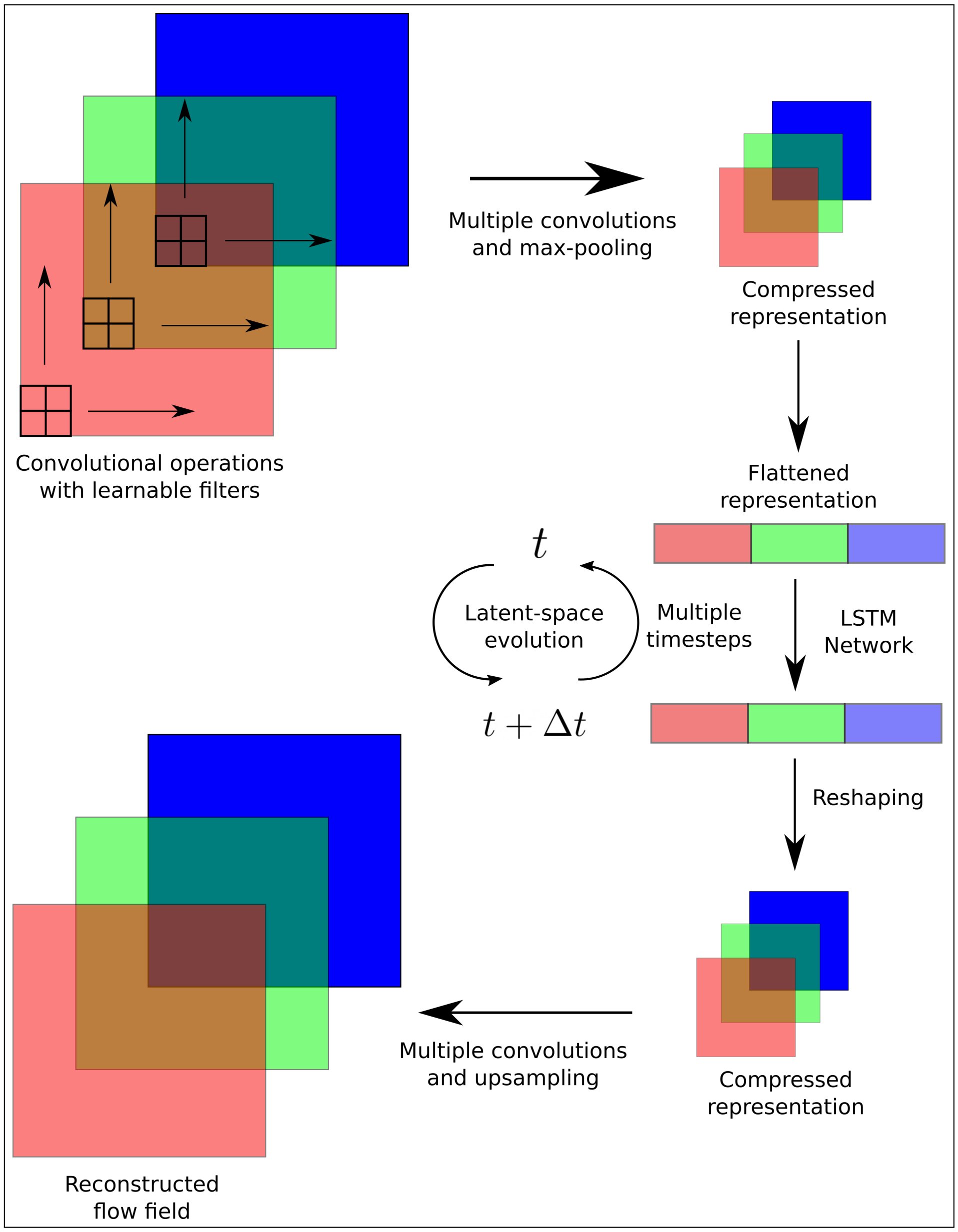


Figure 11: A schematic of the two-dimensional CAE-LSTM for the shallow water equations. The nonlinear autoencoder embeds the data into latent space, and then the LSTM can be used for time-series advancement of a flattened representation of the multidimensional system. “Flattening” refers to the conversion of the coarsened (and nonlinear transformed) representation of the solution field to a one-dimensional state vector.

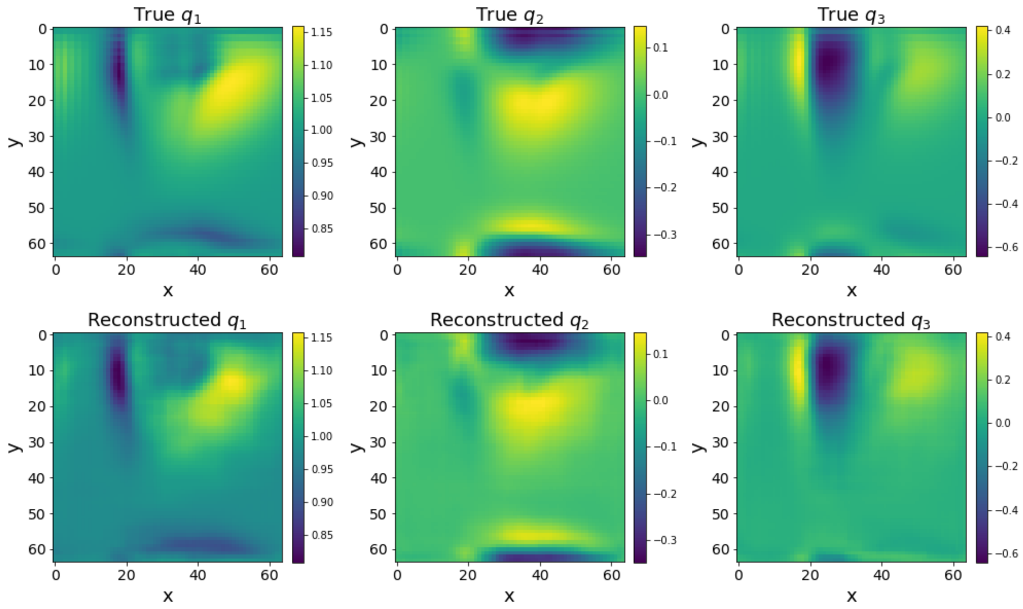
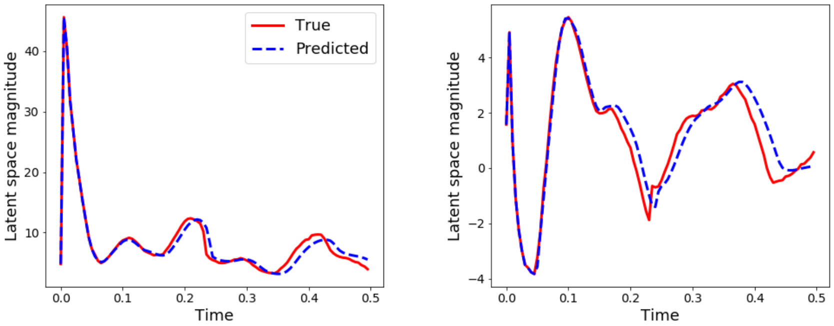


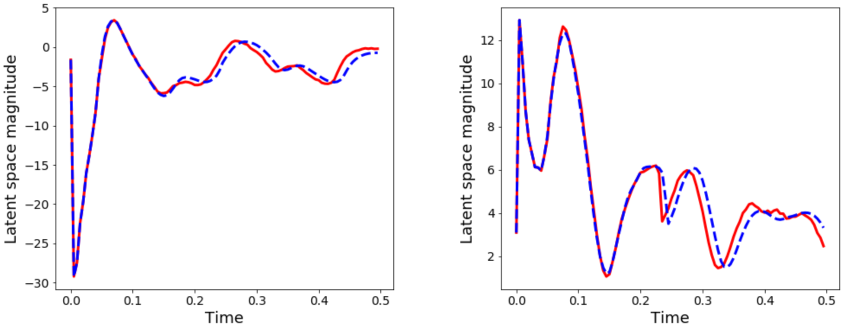
Figure 12: The reconstruction ability of the CAE for the conserved variables in the shallow water equations. This snapshot is from a representative test simulation starting from an unseen initial condition.

namics suggest that the dissipation of energy in the system by the numerical method is captured adequately. Figure 14 shows the ability of the CAE-LSTM surrogate model to identify coherent spatial features in a sample test simulation. For comparison, we show results from benchmark POD-GP deployments for 6 and 40 retained modes. At an equivalent compression ratio, the CAE-LSTM is able to represent the solution well. At 40 retained modes, the severe truncation of dynamics in POD space still leads to Gibbs' phenomena by POD-GP, which demonstrates the robustness of our proposed method. Contour plots at two representative times are shown in Figure 15 and 16 where one can clearly observe that the coherent structures in the flow fields are adequately recovered by the CAE-LSTM in comparison to both 6 and 40 mode POD-GP deployments. However, one can also discern that the POD-GP method gradually converges to the true dynamics with increasing modal retention. As shown in Figure 16, at a later time in the evolution of the trajectory, the POD-GP solution with 40 retained modes outperforms the CAE-LSTM result.

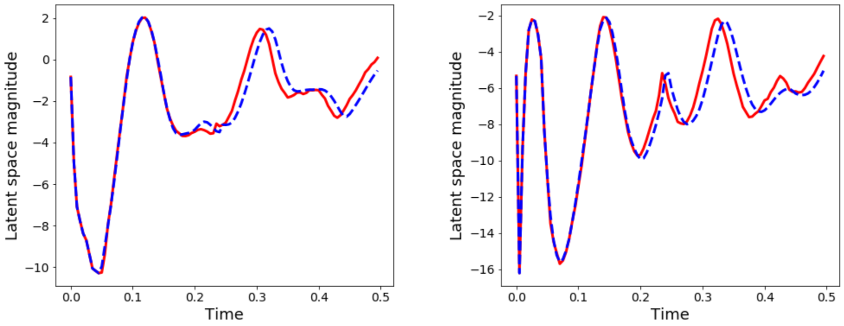
In terms of computational costs, the CAE-LSTM was able to provide an LSTM-based latent space forecast at 1.746 seconds per simulation. Reconstruction from latent space for a 100 snapshot simulation required 0.167 seconds. In comparison a POD-GP ROM deployment (using either 6 or 40 retained modes) required an average 24.67 seconds per simulation. The primary cost in POD-GP deployments is the reconstruction of the nonlinear term for the numerical calculation of fluxes which is independent of the number of latent degrees of freedom. The nonlinear term computation for this test case was performed using a fifth-order WENO scheme just like its full-order counterpart and thus is a memory and compute cost that the machine learned model bypasses. In terms of quantitative error metrics, the q_1 mean-squared error for all of the testing data was 4.8e-4 for CAE-LSTM, 5.6e-4 for POD-GP (6 modes) and 1.7e-4 for POD-GP (40 modes). Similar trends were observed for q_2 (4.8e-4, 7.8e-4, 2.6e-4) and q_3 (3e-3, 3.3e-3, 1.1e-3). Although the mean-squared error metrics support superiority for POD-GP at 40 retained modes, coherent structure reproduction is more accurate via the CAE-LSTM as demonstrated in the contour plots above. Mean-squared error metrics were affected by the greater amount of fine-scale noise in CAE-LSTM reconstructions. A possible avenue for addressing this limitation is to use intelligent loss functions or embed physics-inspired regularization in the optimization problem.



(a) Latent Dimension 1 (b) Latent Dimension 2

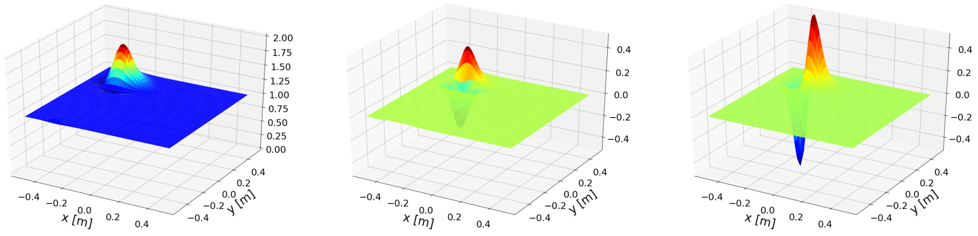


(c) Latent Dimension 3 (d) Latent Dimension 4

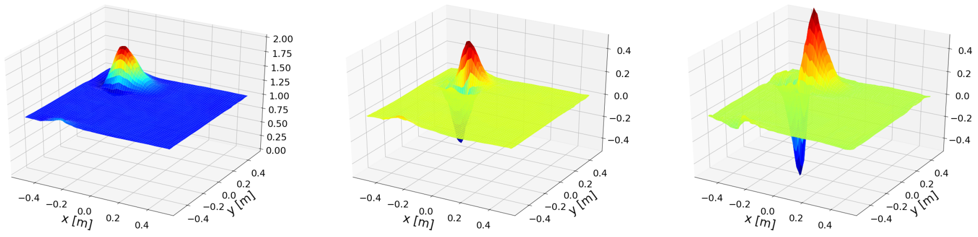


(e) Latent Dimension 5 (f) Latent Dimension 6

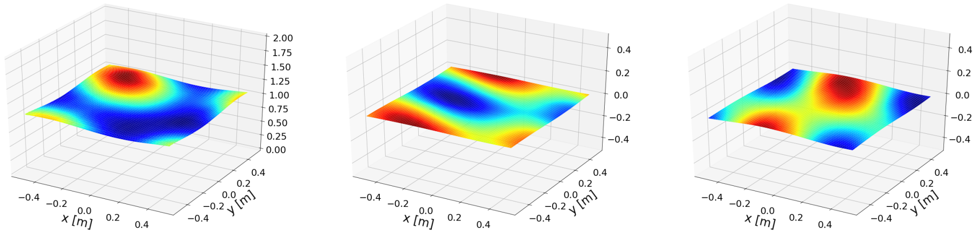
Figure 13: Hidden space evolution of a testing simulation using a parametric LSTM. The curves, here, indicate the individual degrees of freedom of a 6-dimensional latent space with the y-axes indicating their magnitudes in various latent space dimensions.



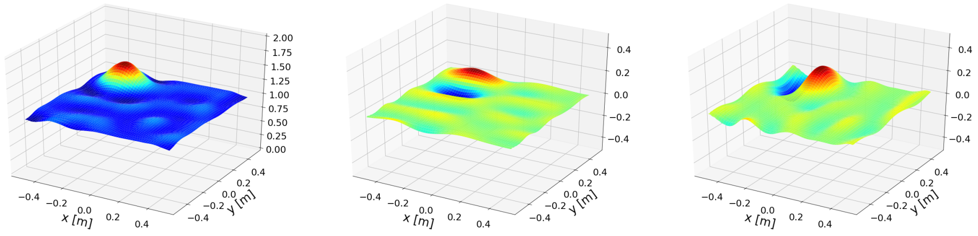
(a) True q_1 (b) True q_2 (c) True q_3



(d) CAE-LSTM (e) CAE-LSTM (f) CAE-LSTM



(g) POD-GP (6 modes) (h) POD-GP (6 modes) (i) POD-GP (6 modes)



(j) POD-GP (40 modes) (k) POD-GP (40 modes) (l) POD-GP (40 modes)

Figure 14: A qualitative assessment of reconstructed dynamics using the Galerkin projection methodology for a test simulation. The superiority of the CAE reconstruction over POD-GP at the same compression ratio (latent dimension 6) is evident. POD-GP performance improves as we capture more of the variance in the data set by increasing the number of modes.

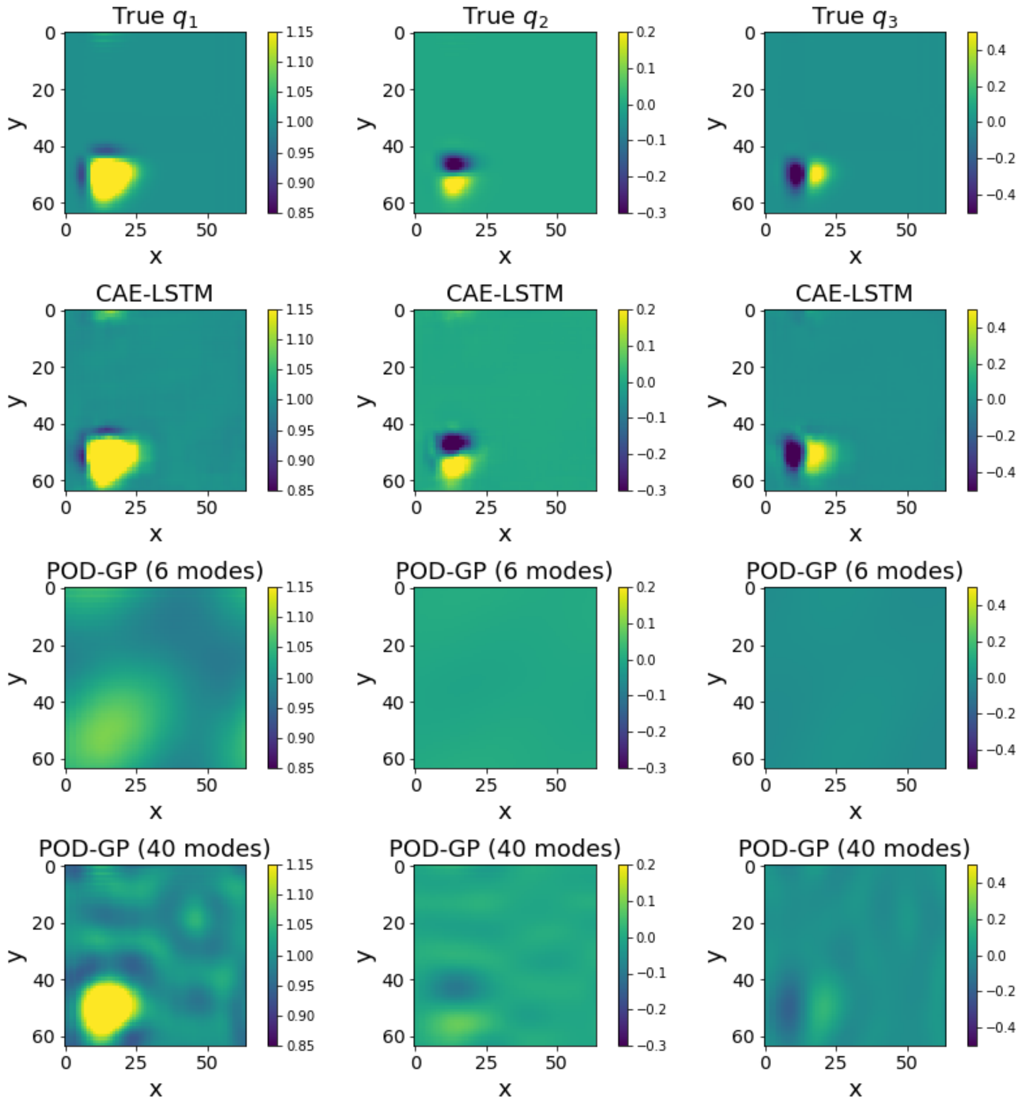


Figure 15: Contour plots showing true, CAE-LSTM and GP obtained results for the three conserved variables at time $t = 0.005$. This corresponds to one quarter of the simulation completed. The CAE-LSTM is seen to capture full-order spatial structures accurately in comparison to the POD-GP method (at 6 latent space dimensions).

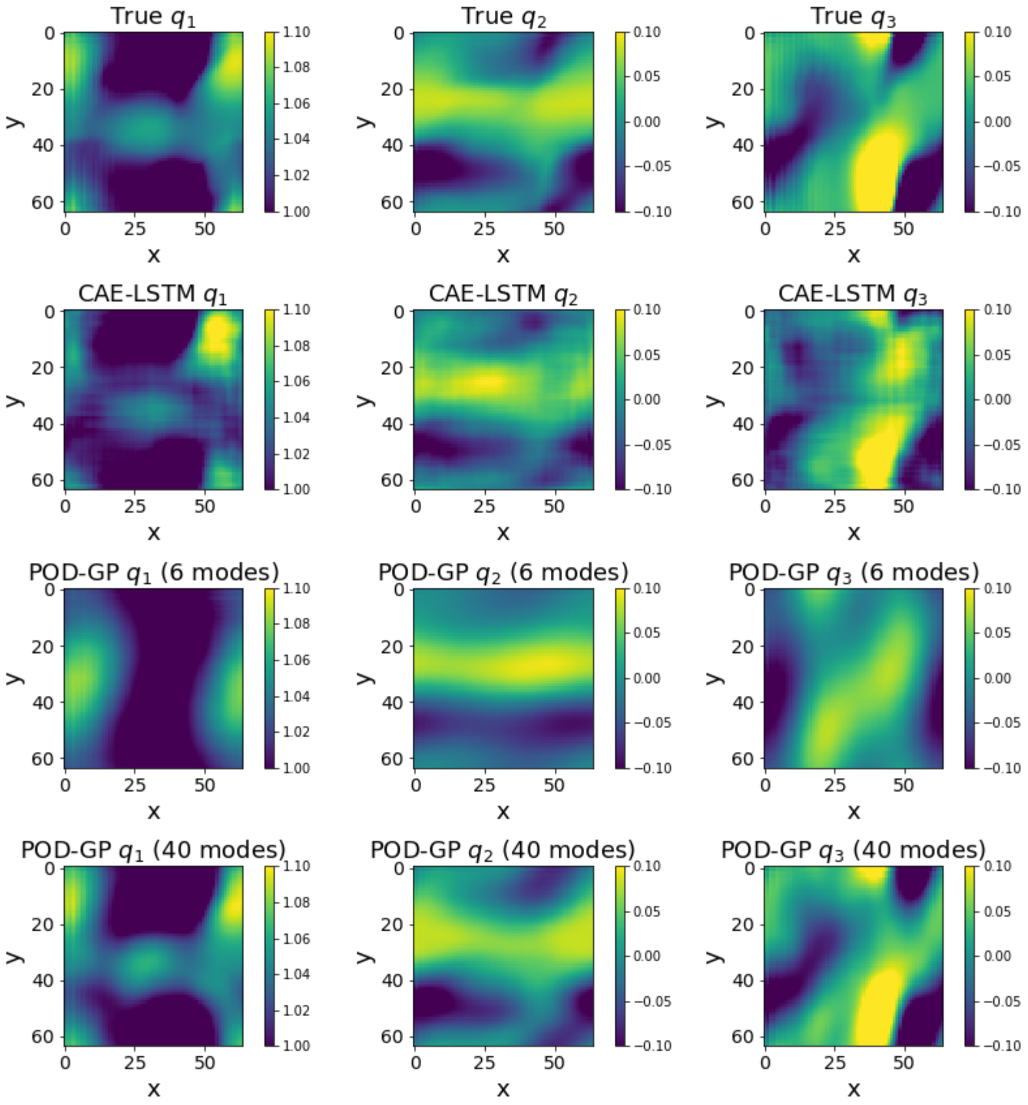


Figure 16: Contour plots showing true, CAE-LSTM and GP obtained results for the three conserved variables at time $t = 0.15$. This corresponds to 30% of the simulation completed. The CAE-LSTM is seen to capture full-order spatial structures accurately in comparison to the POD-GP method (at 6 latent space dimensions).

5. Discussion and Conclusions

In this study, we propose using a recurrent CAE framework for the reduced-order modeling of systems that are inherently advective and, therefore, high-dimensional. These systems suffer from slow convergence and instability in a linear reduced-basis space given by the POD and a Galerkin projection of the governing equations onto this space. In contrast, we demonstrate that the nonlinear embedding obtained by the CAE and the equation-free dynamics characterization by the LSTM network leads to stable reconstructions of high-dimensional physics in both space and time. We extend our machine learning framework to a parametric formulation where we concatenate the low-dimensional embedding with control parameter information to interpolate between full-order sample points in the data generation phase. Our results indicate that the proposed framework can be used for rapid exploration of a design space conditioned on a set of control parameters. Our framework utilizes a *burn-in* period for the LSTM that necessitates a short compute of less than 10% of the full-order compute. This is necessary to create a windowed input to the LSTM network. Results on test datasets show a good ability to recover physical trends on unseen control parameter choices.

We are currently extending the framework by exploring couplings with active learning wherein we adaptively learn control parameters during training in order to characterize parametric variations optimally. In addition, we are also exploring data-augmentation strategies to preclude the initial compute required for the initial LSTM window in latent space. The former will rely on the generation of so-called *ghost* points to serve as a burn-in to the ROM. Some key challenges also include the ability to incorporate unstructured grid information particularly for problems where there is significant anisotropy in spatial field. There is some promising work in this direction using generalized moving least squares methods [61] and point-cloud networks [62]. Another unanswered question is how one may obtain explicit bounds on the performance of the deep learning surrogate model for testing deployment scenarios which require extrapolation. Preliminary results in other scientific machine learning studies have demonstrated the use of stochastic neighbor embedding [63, 64] to compare test and training data *a priori*. Such assessments may be integrated into the aforementioned active learning framework for training data generation. In terms of interpretability, while autoencoder architectures are very responsive from the point of view of obtaining a reduced representation of the flow-field using nonlinear transformations, they are limited due to the opaque nature of this reduced space. While some techniques from the causal study of dynamical systems may be used to extract some meaning from the time-series of the latent space [65], their correlation with structures in the Cartesian space is unclear and a subject of active research. The final goal will be to incorporate these surrogate models in design frameworks that may utilize derivative-based or derivative-free optimization.

Acknowledgments

The authors acknowledge helpful comments from Dr. Sandeep Madireddy and Dr. Arvind Mohan. 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 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.

Data Availability

The data and source code that support the findings of this study are openly available in `https://github.com/Romit-Maulik/CAE_LSTM_ROMS`.

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] S. A. Renganathan, R. Maulik, V. Rao, Machine learning for nonintrusive model order reduction of the parametric inviscid transonic flow past an airfoil, Physics of Fluids 32 (2020) 047110.

[9] J. Ren, H. Wang, J. Xing, K. Luo, J. Fan, A lower-dimensional approximation model of turbulent flame stretch and its related quantities with machine learning approaches, Physics of Fluids 32 (2020) 115113.

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

[11] S. Peitz, S. Ober-Blöbaum, M. Dellnitz, Multiobjective optimal control methods for the navier-stokes equations using reduced order modeling, Acta Applicandae Mathematicae 161 (2019) 171–199.

[12] B. R. Noack, M. Morzynski, G. Tadmor, Reduced-order modelling for flow control, volume 528, Springer Science & Business Media, 2011.

[13] C. W. Rowley, S. T. Dawson, Model reduction for flow analysis and control, Annual Review of Fluid Mechanics 49 (2017) 387–417.

[14] C. Raibaudo, P. Zhong, B. R. Noack, R. J. Martinuzzi, Machine learning strategies applied to the control of a fluidic pinball, Physics of Fluids 32 (2020) 015108.

[15] F. Ren, H.-b. Hu, H. Tang, Active flow control using machine learning: A brief review, Journal of Hydrodynamics 32 (2020) 247–253.

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

[17] D. Fan, B. Zhang, Y. Zhou, B. R. Noack, Optimization and sensitivity analysis of active drag reduction of a square-back ahmed body using machine learning control, Physics of Fluids 32 (2020) 125117.

[18] 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.

[19] 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).

[20] R. Arcucci, L. Mottet, C. Pain, Y.-K. Guo, Optimal reduced space for variational data assimilation, J. Comp. Phys. 379 (2019) 51–69.

[21] M. Tang, Y. Liu, L. J. Durlofsky, A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems, Journal of Computational Physics (2020) 109456.

[22] C. Q. Casas, R. Arcucci, P. Wu, C. Pain, Y.-K. Guo, A reduced order deep data assimilation model, Physica D: Nonlinear Phenomena 412 (2020) 132615.

[23] 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.

[24] 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.

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

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

[27] M. Hess, A. Alla, A. Quaini, G. Rozza, M. Gunzburger, A localized reduced-order modeling approach for pdes with bifurcating solutions, Computer Methods in Applied Mechanics and Engineering 351 (2019) 379–403.

[28] B. Kramer, K. E. Willcox, Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition, AIAA J. 57 (2019) 2297–2307.

[29] R. Swischuk, L. Mainini, B. Peherstorfer, K. Willcox, Projection-based model reduction: Formulations for physics-based machine learning, Comput. Fluids 179 (2019) 704–717.

[30] B. Hamzi, E. H. Abed, Local modal participation analysis of nonlinear systems using poincaré linearization, Nonlinear Dynamics (2019) 1–9.

[31] S. M. Rahman, S. E. Ahmed, O. San, A dynamic closure modeling framework for model order reduction of geophysical flows, Phys. Fluids 31 (2019) 046602.

[32] O. San, J. Borggaard, Basis selection and closure for pod models of convection dominated boussinesq flows, in: 21st International Symposium on Mathematical Theory of Networks and Systems, volume 5, 2014.

[33] M. Korda, M. Putinar, I. Mezić, Data-driven spectral analysis of the koopman operator, Applied and Computational Harmonic Analysis (2018).

[34] V. L. Kalb, A. E. Deane, An intrinsic stabilization scheme for proper orthogonal decomposition based low-dimensional models, Phys. Fluids 19 (2007) 054106.

[35] I. Kalashnikova, M. Barone, On the stability and convergence of a galerkin reduced order model (rom) of compressible flow with solid wall and far-field boundary treatment, International journal for numerical methods in engineering 83 (2010) 1345–1375.

[36] 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.

[37] D. Xiao, F. Fang, J. Du, C. Pain, I. Navon, A. Buchan, A. H. Elsheikh, G. Hu, Non-linear petrov–galerkin methods for reduced order modelling of the navier–stokes equations using a mixed finite element pair, Computer Methods In Applied Mechanics and Engineering 255 (2013) 147–157.

[38] F. Fang, C. C. Pain, I. Navon, A. H. Elsheikh, J. Du, D. Xiao, Non-linear petrov–galerkin methods for reduced order hyperbolic equations and discontinuous finite element methods, Journal of Computational Physics 234 (2013) 540–559.

[39] K. Carlberg, M. Barone, H. Antil, Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction, Journal of Computational Physics 330 (2017) 693–734.

[40] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780.

[41] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, P. Koumoutsakos, Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 474 (2018) 20170844.

[42] S. E. Ahmed, O. San, A. Rasheed, T. Iliescu, A long short-term memory embedding for hybrid uplifted reduced order models, arXiv preprint arXiv:1912.06756 (2019).

[43] R. Maulik, A. Mohan, B. Lusch, S. Madireddy, P. Balaprakash, Time-series learning of latent-space dynamics for reduced-order model closure, arXiv preprint arXiv:1906.07815 (2019).

[44] 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).

[45] 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).

[46] 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.

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

[48] N. B. Erichson, M. Muehlebach, M. W. Mahoney, Physics-informed autoencoders for lyapunov-stable fluid flow prediction, arXiv preprint arXiv:1905.10866 (2019).

[49] K. Lee, K. T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comp. Phys. 404 (2020) 108973.

[50] F. J. Gonzalez, M. Balajewicz, Learning low-dimensional feature dynamics using deep convolutional recurrent autoencoders, arXiv preprint arXiv:1808.01346 (2018).

[51] J. Xu, K. Duraisamy, Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics, arXiv preprint arXiv:1912.11114 (2019).

[52] R. Maulik, T. Botsas, N. Ramachandra, L. R. Mason, I. Pan, Latent-space time evolution of non-intrusive reduced-order models using gaussian process emulation, Physica D: Nonlinear Phenomena (2020) 132797.

[53] L. Agostini, Exploration and prediction of fluid dynamical systems using auto-encoder technology, Physics of Fluids 32 (2020) 067103.

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

[55] 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.

[56] Y. Rubanova, R. T. Chen, D. Duvenaud, Latent odes for irregularly-sampled time series, arXiv preprint arXiv:1907.03907 (2019).

[57] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, J. Comp. Phys. 115 (1994) 200–212.

[58] E. Hairer, S. P. Nørsett, G. Wanner, Solving ordinary differential equations. 1, Nonstiff problems, Springer-Vlg, 1991.

[59] R. Maulik, O. San, Resolution and energy dissipation characteristics of implicit les and explicit filtering models for compressible turbulence, Fluids 2 (2017) 14.

[60] P. Ramachandran, B. Zoph, Q. V. Le, Searching for activation functions, arXiv preprint arXiv:1710.05941 (2017).

[61] N. Trask, R. G. Patel, B. J. Gross, P. J. Atzberger, Gmls-nets: A framework for learning from unstructured data, arXiv preprint arXiv:1909.05371 (2019).

[62] A. Kashefi, D. Rempe, L. J. Guibas, A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries, arXiv preprint arXiv:2010.09469 (2020).

[63] G. E. Hinton, S. Roweis, Stochastic neighbor embedding, Advances in neural information processing systems 15 (2002) 857–864.

[64] R. Maulik, N. A. Garland, J. W. Burby, X.-Z. Tang, P. Balaprakash, Neural network representability of fully ionized plasma fluid model closures, Physics of Plasmas 27 (2020) 072106.

[65] A. K. Seth, A. B. Barrett, L. Barnett, Granger causality analysis in neuroscience and neuroimaging, Journal of Neuroscience 35 (2015) 3293–3297.

[66] 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).

Appendix A. Nomenclature

$\bar{x}$	x -coordinate of Gaussian pulse in shallow water equations
$\bar{y}$	y -coordinate of Gaussian pulse in shallow water equations
$\boldsymbol{\psi}$	POD basis functions
$\boldsymbol{L}_h$	Discretized linear term in PDE
$\boldsymbol{N}_h$	Discretized nonlinear term in PDE
$\mathbf{p}$	Parameter vector
$\mathbf{q}$	Dependent variable in PDE
$\mathbf{q}_h$	Discretized dependent variable in PDE
$\mathbf{q}_r$	Dependent variable in reduced-space
$\mathbf{x}$	Position vector
η	Fluid column height in shallow water equations
$\mathcal{L}$	Linear PDE term
$\mathcal{L}_r$	Linear term in POD-GP reduced-order model
$\mathcal{N}$	Nonlinear PDE term
$\mathcal{P}$	Parameter domain
$\mathcal{T}$	Temporal domain
ν	Viscosity
Ω	Spatial domain
ρ	Fluid density in shallow water equations
d	Dimensionality of $\mathbf{q}$
g	Acceleration due to gravity in shallow water equations
i	Dimensionality of $\mathbf{x}$
L	Spatial domain length for Viscous Burgers equation
N_j	Number of spatial degrees of freedom
p	Dimensionality of p
q_1	Conserved variable $\rho\eta$ in shallow water equations
q_2	Conserved variable $\rho\eta v_x$ in shallow water equations
q_3	Conserved variable $\rho\eta v_y$ in shallow water equations
Re	Reynolds number (inverse of viscosity in Viscous Burgers equation)
t	Time
t_{max}	Maximum simulation time for Viscous Burgers equation
$u(x,t)$	Solution to Viscous Burgers equation
v_x	Fluid velocity x -component in shallow water equations
v_y	Fluid velocity y -component in shallow water equations
CAE	Convolutional autoencoder

GP	Galerkin-projection
LSTM	Long short-term memory network
POD	Proper orthogonal decomposition
ROM	Reduced-order model

Appendix B. Proper orthogonal decomposition

In this section, we review the POD technique for the construction of a reduced basis [54, 55]. The interested reader may also find an excellent explanation of POD and its relationship with other dimension-reduction techniques in [66]. The POD procedure is tasked with identifying a space

$$\mathbf{X}^f = \text{span}\left\{\boldsymbol{\vartheta}^1, \dots, \boldsymbol{\vartheta}^f\right\}, \tag{B.1}$$

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

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

where N_s is the number of snapshots, and $\hat{\mathbf{q}}_h^i \in \mathbb{R}^{N_h}$ corresponds to an individual snapshot in time of the discrete solution domain. Our POD bases can then be extracted efficiently through the method of snapshots where we solve the eigenvalue problem on the correlation matrix $\mathbf{C} = \mathbf{S}^T\mathbf{S} \in \mathbb{R}^{N_s \times N_s}$. Then

$$\mathbf{C}\mathbf{W} = \mathbf{W}\Lambda, \tag{B.3}$$

where $\Lambda = \text{diag}\left\{\lambda_1, \lambda_2, \cdots, \lambda_{N_s}\right\} \in \mathbb{R}^{N_s \times N_s}$ 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

$$\boldsymbol{\vartheta} = \mathbf{S}\mathbf{W} \in \mathbb{R}^{N_h \times N_s}. \tag{B.4}$$

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

$$\mathbf{X}^r = \text{span}\left\{\boldsymbol{\psi}^1, \dots, \boldsymbol{\psi}^{N_r}\right\}. \tag{B.5}$$

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

$$\mathbf{A} = \boldsymbol{\psi}^T \mathbf{S} \in \mathbb{R}^{N_r \times N_s}. \tag{B.6}$$

The POD approximation of our solution is then obtained via

$$\hat{\mathbf{S}} = \left[\ \tilde{\mathbf{q}}_h^1 \ \middle|\ \tilde{\mathbf{q}}_h^2 \ \middle|\ \cdots \ \middle|\ \tilde{\mathbf{q}}_h^{N_s}\ \right] \approx \boldsymbol{\psi}\mathbf{A} \in \mathbb{R}^{N_h \times N_s}, \tag{B.7}$$

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

$$\frac{\sum_{i=1}^{N_s} \|\hat{\mathbf{q}}_h^i - \tilde{\mathbf{q}}_h^i\|_{\mathbb{R}^{N_h}}^2}{\sum_{i=1}^{N_s} \|\hat{\mathbf{q}}_h^i\|_{\mathbb{R}^{N_h}}^2} = \frac{\sum_{i=N_r+1}^{N_s} \lambda_i^2}{\sum_{i=1}^{N_s} \lambda_i^2}, \tag{B.8}$$

which exhibits that with increasing retention of POD bases, increasing reconstruction accuracy may be obtained. We remark that for dimension $d > 1$, the solution variables may be stacked to obtain this set of bases that are utilized for the reduction of each PDE within the coupled system. Another approach may be to obtain reduced bases for each dependent variable within the coupled system and evolve each PDE on a different manifold. Each dependent variable is projected onto bases constructed from its snapshots alone. This affects the computation of $\mathcal{N}$ for computing the updates for each dimension in $\mathbf{q}$. In practice, this operation manifests itself in the concatenation of reduced bases to obtain one linear operation for reconstruction of all field quantities.