Physica D 409 (2020) 132471

Contents lists available at ScienceDirect

Physica D

journal homepage: www.elsevier.com/locate/physd

A long short-term memory embedding for hybrid uplifted reduced order models

Shady E. Ahmed [a], Omer San [a,*], Adil Rasheed [b], Traian Iliescu [c]

[a] School of Mechanical and Aerospace Engineering, Oklahoma State University, Stillwater, OK 74078, USA
[b] Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7465, Trondheim, Norway
[c] Department of Mathematics, Virginia Tech, Blacksburg, VA 24061, USA

ARTICLE INFO

Article history:
Received 11 December 2019
Received in revised form 15 March 2020
Accepted 16 March 2020
Available online 1 April 2020

Keywords:
Hybrid analysis and modeling
Supervised machine learning
Long short-term memory
Model reduction
Galerkin projection
Grassmann manifold

ABSTRACT

In this paper, we introduce an uplifted reduced order modeling (UROM) approach through the integration of standard projection based methods with long short-term memory (LSTM) embedding. Our approach has three modeling layers or components. In the first layer, we utilize an intrusive projection approach to model the dynamics represented by the largest modes. The second layer consists of an LSTM model to account for residuals beyond this truncation. This closure layer refers to the process of including the residual effect of the discarded modes into the dynamics of the largest scales. However, the feasibility of generating a low rank approximation tails off for higher Kolmogorov n-width systems due to the underlying nonlinear processes. The third uplifting layer, called super-resolution, addresses this limited representation issue by expanding the span into a larger number of modes utilizing the versatility of LSTM. Therefore, our model integrates a physics-based projection model with a memory embedded LSTM closure and an LSTM based super-resolution model. In several applications, we exploit the use of Grassmann manifold to construct UROM for unseen conditions. We perform numerical experiments by using the Burgers and Navier–Stokes equations with quadratic nonlinearity. Our results show the robustness of the proposed approach in building reduced order models for parameterized systems and confirm the improved trade-off between accuracy and efficiency.

1. Introduction

Physical models are often sought because of their reliability, interpretability, and generalizability being derived from basic principles and physical intuition. However, accurate solution of these models for complex systems usually requires the use of very high spatial and temporal resolutions and/or sophisticated discretization techniques. This limits their applications to offline simulations over a few sets of parameters and short time intervals since they can be excessively computationally-demanding. Although those are valuable in understanding physical phenomena and gaining more insight, realistic applications often require near real-time and multi-query responses [1,2]. Therefore, cheaper numerical approximations using "adequate-fidelity" models are usually acceptable [3]. In this regard, reduced order modeling offers a viable technique to address systems characterized by underlying patterns [4–14]. This is especially true for fluid flows dominated by coherent structures (e.g., atmospheric and oceanic flows) [15–23].

Reduced order models (ROMs) have shown great success for prototypical problems in different fields. In particular, Galerkin projection (GP) coupled with the capability of proper orthogonal decomposition (POD) to extract the most energetic modes has been used to build ROMs for linear and nonlinear systems [24–31]. These ROMs preserve sufficient interpretability and generalizability since they are constructed by projecting the full order model (FOM) operators (from governing equations) on a reduced subspace. Despite that, Galerkin projection ROMs (GROMs) have severe limitations in practice, especially for systems with strong nonlinearity. Most fluid flows exhibit quadratic nonlinearity, which makes the computational cost of the resulting GROMs $\sim O(R^3)$, where R is the number of retained modes in the ROM approximation. As a result, R should be kept as low as possible (e.g., $O(5)$) through modal truncation for practical purposes; however, this has two main consequences. First, the solution is enforced to live in a smaller subspace which might not contain enough information to accurately represent complex realistic systems. Examples include advection-dominated flows and parametric systems where the decay of Kolmogorov n-width is

* Corresponding author.
E-mail addresses: shady.ahmed@okstate.edu (S.E. Ahmed), osan@okstate.edu (O. San), adil.rasheed@ntnu.no (A. Rasheed), iliescu@vt.edu (T. Iliescu).

https://doi.org/10.1016/j.physd.2020.132471
0167-2789/

slow [32–34]. Second, due to the inherent system's nonlinearity, the truncated modes interact with the retained modes. In GROM, these interactions are simply eliminated by modal truncation, which often generates instabilities in the approximation [35–37]. Several efforts have been devoted to introduce stabilization and closure techniques [38–55] to account for the effects of truncated modes on ROM's dynamics.

In the present study, we aim to address the above problems while preserving considerable interpretability, and generalizability at the core of our uplifted ROM (UROM) approach. In UROM, we present a three-modeling layer framework. In the first layer, we use a standard Galerkin projection method based on the governing equations to model the large scales of the flow (represented by the first few R POD modes) and provide a predictor for the temporal evolution. In the next layer, we introduce a corrector step to correct the Galerkin approximation and make up for the interactions of the truncated modes (or scales) with the large ones. These large scales contribute most to the total system's energy. That is why we dedicate two layers of our approach to correctly resolve them, one of which (i.e., the Galerkin projection) is totally physics-based to promote framework generality. In the third layer, we uplift our approximation and expand the solution subspace to recover some of the flow's finer details by learning a map between the large scales (predicted using the first two layers) and smaller scales.

In particular, we choose the first $R \approx O(5)$ modes to represent the resolved largest scales and the next $Q - R$ modes to represent the resolved smaller scales, where Q is about 4 to 8 times larger than R. For the second and third layers, we incorporate memory embedding through the use of long short-term memory (LSTM) neural network architecture [56,57]. Machine learning (ML) tools (of which neural network is a subclass) have been gaining popularity in fluid mechanics community and analysis of dynamical systems [58–66]. In particular, LSTMs have shown great success in learning maps of sequential data and time-series [67–72]. It should be noted here that UROM can be thought of as a way of augmenting physical models with data-driven tools [73] and vice versa. For the former, an LSTM closure model (second component) is developed to correct GROM and an LSTM super-resolution model (third component) is constructed to uplift GROM and resolve smaller scales. This relaxes the computational cost of GROM to account only for a few R modes. On the other hand, LSTMs in UROM framework are fed with inputs coming from a physics-based approach. This is one way of utilizing physical information rather than full dependence on ML results.

To illustrate the UROM framework, we consider two convection-dominated flows as test cases. The first is the one-dimensional (1D) Burgers equation, which is a simplified benchmark problem for fluid flows with strong nonlinearity. As the second test case, we investigate the two-dimensional (2D) Navier–Stokes equations for a flow with interacting vortices, namely the vortex merger problem. We compare the UROM approach with the standard GROM approach using R and Q modes. We also investigate a fully non-intrusive ROM (NIROM) approach in these flow problems. We perform a comparison in terms of solution accuracy and computational time to show the pros and cons of UROM with respect to either GROM or NIROM approaches. The rest of the paper is outlined here. In Section 2, we introduce the POD technique for data compression and constructing lower-dimensional subspaces to approximate the solution. For an out-of-sample control parameter (e.g., Reynolds number), we describe basis interpolation via a Grassmann manifold approach in Section 3. As a standard physics-informed technique for building ROMs, we introduce Galerkin projection in Section 4 with a brief description of the governing equations of the test cases as well as their corresponding GROM structures. In Section 5, we present the proposed UROM framework with a description of its main features. We give results and corresponding discussions in Section 6. Finally, we draw concluding remarks and insights in Section 7.

2. Proper orthogonal decomposition

Proper orthogonal decomposition (POD) is one of the most popular techniques for dimensionality reduction and data compression [15,74–77]. Given data sets, POD provides a linear subspace that minimizes the error between the true data and its projection compared to all possible linear subspaces with the same dimension. If a number of N_s data snapshots, $\mathbf{u}(\mathbf{x}, t_n)$, where $n \in \{1, 2, \ldots, N_s\}$, $\mathbf{x} \in \mathbb{R}^N$ (N being the spatial resolution), are collected in a snapshot matrix $\mathbf{A} \in \mathbb{R}^{N \times N_s}$ (where N is much larger than N_s for typical flow problems), then a reduced (or thin) singular value decomposition (SVD) can be applied to this matrix as

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T, \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{N \times N_s}$ is a matrix with orthonormal columns representing the left-singular vectors of $\mathbf{A}$, also known as spatial basis, and $\mathbf{V} \in \mathbb{R}^{N_s \times N_s}$ is also a matrix with orthonormal columns which represent the right-singular vectors, sometimes referred to as temporal basis. $\boldsymbol{\Sigma} \in \mathbb{R}^{N_s \times N_s}$ is a diagonal matrix whose entries are the singular values of $\mathbf{A}$ (square-roots of the largest N_s eigenvalues of $\mathbf{A}\mathbf{A}^T$ or $\mathbf{A}^T\mathbf{A}$). In $\boldsymbol{\Sigma}$, the singular values σ_i are sorted in a descending order such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{N_s} \geq 0$.

For dimensionality reduction purposes, only the first R columns of $\mathbf{U}$ (denoted as $\widehat{\mathbf{U}}$), the first R columns of $\mathbf{V}$ (denoted as $\widehat{\mathbf{V}}$), and the upper-left $R \times R$ block sub-matrix of $\boldsymbol{\Sigma}$ (denoted as $\widehat{\boldsymbol{\Sigma}}$) are retained to provide a reduced order approximation $\widehat{\mathbf{A}}$ of $\mathbf{A}$ as

$$\widehat{\mathbf{A}} = \widehat{\mathbf{U}}\widehat{\boldsymbol{\Sigma}}\widehat{\mathbf{V}}^T. \tag{2}$$

It can be easily shown that this approximation $\widehat{\mathbf{A}}$ satisfies the following equalities [78]

$$\|\mathbf{A} - \widehat{\mathbf{A}}\|_2 = \inf_{\substack{\mathbf{B} \in \mathbb{R}^{N \times N_s} \\ rank(\mathbf{B}) \leq R}} \|\mathbf{A} - \mathbf{B}\|_2 \tag{3}$$

$$\|\mathbf{A} - \widehat{\mathbf{A}}\|_2 = \sigma_{R+1}, \tag{4}$$

where $\|\cdot\|_2$ refers to the matrix 2-norm. Eq. (3) means that across all possible matrices $\mathbf{B} \in \mathbb{R}^{N \times N_s}$ of rank R (or less), $\widehat{\mathbf{A}}$ provides the closest one to $\mathbf{A}$ in the ℓ_2 sense. Moreover, the singular values $\{\sigma_i\}$ provide a measure of the quality of this approximation as Eq. (4) shows that the ℓ_2 norm between the matrix $\mathbf{A}$ and its R-rank approximation equals σ_{R+1}. From now on, the first R columns of $\mathbf{U}$ will be referred to as the POD modes or basis functions, denoted as $\boldsymbol{\Phi} = [\phi_1, \phi_2, \ldots, \phi_R]$.

3. Grassmann manifold interpolation

In recent years, Grassmann manifold has attracted great interest in various applications including reduced order modeling for parametric systems [79–83]. The Grassmann manifold, $\mathcal{G}(q, N)$, is a set of all q-dimensional subspaces in $\mathbb{R}^N$, where $0 \leq q \leq N$. A point $[\boldsymbol{\Phi}] \in \mathcal{G}(q, N)$ is given as [84]

$$[\boldsymbol{\Phi}] = \{\boldsymbol{\Phi}Q \;\big|\; \boldsymbol{\Phi}^T\boldsymbol{\Phi} = I_q, Q \in \mathcal{O}(q)\}, \tag{5}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{N \times q}$ and $\mathcal{O}(q)$ is the group of all $q \times q$ orthogonal matrices. This point represents a q-dimensional subspace $\mathcal{S}$ in $\mathbb{R}^N$ spanned by the columns of $\boldsymbol{\Phi}$. At each point $[\boldsymbol{\Phi}] \in \mathcal{G}(q, N)$, a tangent space $\mathcal{T}([\boldsymbol{\Phi}])$ of the same dimension, $N \times q$, can be defined as follows [85,86]

$$\mathcal{T}([\boldsymbol{\Phi}]) = \{\mathcal{X} \in \mathbb{R}^{N \times q} \;\big|\; \boldsymbol{\Phi}^T\mathcal{X} = \mathbf{0}\}. \tag{6}$$

Similarly, each point $[\mathbf{\Gamma}]$ on $\mathcal{T}$ represents a subspace spanned by the columns of $\mathbf{\Gamma}$. This tangent space is a vector space with its origin at $[\mathbf{\Phi}]$. An exponential mapping from a point $[\mathbf{\Gamma}] \in \mathcal{T}([\mathbf{\Phi}])$ to $[\mathbf{\Psi}] \in \mathcal{G}(q, N)$ can be defined as

$$\mathbf{\Psi} = \left(\mathbf{\Phi}\mathbf{V}\cos(\mathbf{\Sigma}) + \mathbf{U}\sin(\mathbf{\Sigma})\right)\mathbf{V}^T, \tag{7}$$

where $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V}$ are obtained from the reduced SVD of $\mathbf{\Gamma}$ as $\mathbf{\Gamma} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Inversely, a logarithmic map can be defined from a point $[\mathbf{\Psi}]$ in the neighborhood of $[\mathbf{\Phi}]$ to $[\mathbf{\Gamma}] \in \mathcal{T}([\mathbf{\Phi}])$ as

$$\mathbf{\Gamma} = \mathbf{U}\tan^{-1}(\mathbf{\Sigma})\mathbf{V}^T, \tag{8}$$

where $(\mathbf{\Psi} - \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{\Psi})(\mathbf{\Phi}\mathbf{\Psi})^{-1} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. We would like to note here that the trigonometric functions are applied element-wise to the diagonal entries.

To demonstrate the procedure in our ROM context, for a number N_p of control parameters $\{\mu_i\}_{i=1}^{N_p}$, different sets of POD basis functions are computed corresponding to each parameter, denoted as $\{\mathbf{\Phi}_i\}_{i=1}^{N_p}$. These bases correspond to a set of points on the Grassmann manifold $\mathcal{G}(R, N)$. To perform an out-of-sample testing, the basis functions $\mathbf{\Phi}_{\text{Test}}$ for the test parameter μ_{Test} should be computed through interpolation. However, direct interpolation of the POD bases is not effective since it is an interpolation on a non-flat space and it does not guarantee that the resulting point would lie on $\mathcal{G}(R, N)$. Moreover, the optimality and orthonormality characteristics of POD are not necessarily conserved. Alternatively, the tangent space $\mathcal{T}$ is a flat space where standard interpolation can be performed effectively. First, a reference point at the Grassmann manifold is selected, corresponding to $\mathbf{\Phi}_{\text{Ref}}$. The tangent plane at this point is thus defined using Eq. (6). Then, the neighboring points on Grassmann manifold corresponding to the subspaces spanned by $\{\mathbf{\Phi}_i\}_{i=1}^{N_p}$ are mapped onto that tangent plane using the logarithmic map, defined in Eq. (8) to calculate $\{\mathbf{\Gamma}_i\}_{i=1}^{N_p}$. Standard Lagrange interpolation can be performed to compute $\mathbf{\Gamma}_{\text{Test}}$ as follows

$$\mathbf{\Gamma}_{\text{Test}} = \sum_{i=1}^{N_p}\left(\prod_{\substack{j=1\\ j\neq i}}^{N_p}\frac{\mu_{\text{Test}} - \mu_j}{\mu_i - \mu_j}\right)\mathbf{\Gamma}_i. \tag{9}$$

Finally, the point $[\mathbf{\Gamma}_{\text{Test}}] \in \mathcal{T}([\mathbf{\Phi}_{\text{Ref}}])$ is mapped to the Grassmann manifold $\mathcal{G}(R, N)$ to obtain the set of POD basis functions at the test parameter, $\mathbf{\Phi}_{\text{Test}}$, using the exponential map given in Eq. (7). Hence, an interpolation on the tangent plane to Grassmann manifold provides a basis of the same dimension (i.e., $[\mathbf{\Phi}_{\text{Test}}] \in \mathcal{G}(R, N)$). Moreover, it preserves the orthonormality of the basis (i.e., the columns of $\mathbf{\Phi}_{\text{Test}}$ are orthonormal to each other). Those properties are not guaranteed if conventional interpolation techniques are used directly to interpolate basis. The procedure for Grassmann manifold interpolation is summarized in Algorithm 1. Note that this approach can be generalized to consider more than one varying parameter (instead of just μ), where higher dimensional interpolation schemes can be used in lieu of the adopted one-dimensional Lagrange interpolation.

Algorithm 1 Grassmann manifold interpolation

1: Given N_p sets of basis functions $\mathbf{\Phi}_1, \mathbf{\Phi}_2, \ldots, \mathbf{\Phi}_{N_p}$ corresponding to the offline simulations parameterized by $\mu_1, \mu_2, \ldots, \mu_{N_p}$.

2: Select a point $[\mathbf{\Phi}_{\text{Ref}}] \leftarrow [\mathbf{\Phi}_i] \in \{[\mathbf{\Phi}_1], \ldots, [\mathbf{\Phi}_{N_p}]\}$ corresponding to the basis function set $\mathbf{\Phi}_{\text{Ref}} \leftarrow \mathbf{\Phi}_i \in [\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_{N_p}]$ as the reference point.

3: Map each point $[\mathbf{\Phi}_i] \in \mathcal{G}(N, R)$ to $[\mathbf{\Gamma}_i] \in \mathcal{T}([\mathbf{\Phi}_{\text{Ref}}])$ using logarithmic map

$$(\mathbf{\Phi}_i - \mathbf{\Phi}_{\text{Ref}}\mathbf{\Phi}_{\text{Ref}}^T\mathbf{\Phi}_i)(\mathbf{\Phi}_{\text{Ref}}^T\mathbf{\Phi}_i)^{-1} = \mathbf{U}_i\mathbf{\Sigma}_i\mathbf{V}_i^T, \tag{10}$$

$$\mathbf{\Gamma}_i = \mathbf{U}_i\tan^{-1}(\mathbf{\Sigma}_i)\mathbf{V}_i^T. \tag{11}$$

4: Construct the matrix $\mathbf{\Gamma}_{\text{Test}}$ corresponding to the test parameter μ_{Test} using Lagrange interpolation of matrices $\mathbf{\Gamma}_i$, corresponding to $\mu_1, \ldots, \mu_{N_p}$

$$\mathbf{\Gamma}_{\text{Test}} = \sum_{i=1}^{N_p}\left(\prod_{\substack{j=1\\ j\neq i}}^{N_p}\frac{\mu_{\text{Test}} - \mu_j}{\mu_i - \mu_j}\right)\mathbf{\Gamma}_i. \tag{12}$$

5: Compute the POD basis functions $\mathbf{\Phi}_{\text{Test}}$ corresponding to the test parameter μ_{Test} using the exponential map

$$\mathbf{\Gamma}_{\text{Test}} = \mathbf{U}_{\text{Test}}\mathbf{\Sigma}_{\text{Test}}\mathbf{V}_{\text{Test}}^T, \tag{13}$$

$$\mathbf{\Phi}_{\text{Test}} = \left(\mathbf{\Phi}_{\text{Ref}}\mathbf{V}_{\text{Test}}\cos(\mathbf{\Sigma}_{\text{Test}}) + \mathbf{U}_{\text{Test}}\sin(\mathbf{\Sigma}_{\text{Test}})\right)\mathbf{V}_{\text{Test}}^T, \tag{14}$$

where the trigonometric operators apply only to the diagonal elements.

4. Galerkin projection

To emulate the system's dynamics in ROM context, a Galerkin projection is usually performed. In Galerkin projection-based ROM (GROM), the solution $\mathbf{u}(\mathbf{x}, t_n)$ is constrained to lie in a trial subspace $\mathcal{S}$ spanned by the basis $\mathbf{\Phi}$. In our study, this basis is computed using the POD method presented in Section 2. Then, the full-order operators are projected onto the same subspace $\mathcal{S}$. In other words, the residual of the governing ODE is enforced to be orthogonal to $\mathcal{S}$. Galerkin projection can be viewed as a special case of Petrov–Galerkin method [87–90], by utilizing the same trial subspace as a test subspace. In the following, we present the governing equations for our test cases, namely Burgers equation and Navier–Stokes equations as well as their low-order approximations.

4.1. 1D Burgers equation

The one-dimensional (1D) viscous Burgers equation represents a standard benchmark for the analysis of nonlinear advection–diffusion problems in a 1D setting with similar quadratic nonlinear interaction and Laplacian dissipation. The evolution of the velocity field $u(x, t)$, in a dimensionless form, is given by

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{1}{\text{Re}}\frac{\partial^2 u}{\partial x^2}, \tag{15}$$

where Re is the dimensionless Reynolds number, defined as the ratio of inertial effects to viscous effects. Eq. (15) can be rewritten as

$$\frac{\partial u}{\partial t} = \frac{1}{\text{Re}}\frac{\partial^2 u}{\partial x^2} - u\frac{\partial u}{\partial x}. \tag{16}$$

Then, the reduced-rank approximation $u(x, t) \approx \sum_{k=1}^{R} a_k(t)\phi_k(x)$ (where ϕ_k are the constructed POD modes and a_k are the corresponding coefficients) is plugged into this equation and an inner product with an arbitrary basis ϕ_k is performed to give the following dynamical ODE, which represents the GROM for the Burgers equation

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \sum_{i=1}^{R}\mathfrak{L}_{i,k}a_i + \sum_{i=1}^{R}\sum_{j=1}^{R}\mathfrak{N}_{i,j,k}a_i a_j, \qquad k = 1, 2, \ldots, R, \tag{17}$$

where $\mathfrak{L}$ and $\mathfrak{N}$ are the matrix and tensor of predetermined model coefficients corresponding to linear and nonlinear terms, respectively. They are precomputed as

$$\mathfrak{L}_{i,k} = \left\langle \frac{1}{\mathrm{Re}} \frac{\partial^2 \phi_i}{\partial x^2}; \phi_k \right\rangle,$$
$$\mathfrak{N}_{i,j,k} = \left\langle -\phi_i \frac{\partial \phi_j}{\partial x}; \phi_k \right\rangle,$$

where the angle-parentheses refer to the Euclidean inner product defined as $\langle \mathbf{x}; \mathbf{y} \rangle = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^{N} x_i y_i$.

We note here that the basis functions ϕ_k are spatial functions and thus, standard discretization techniques (e.g., finite difference methods) can be used to compute the required derivatives. Moreover, to compute the inner product in Euclidean space, the basis functions can be treated as regular vectors (i.e., values of $\phi_k(x)$ can be arranged in a vector as $[\phi_k(x_1), \phi_k(x_2), \ldots, \phi_k(x_{N_x})]^T$, where N_x is the number of grid points.

4.2. 2D Navier–Stokes equations

The vorticity-streamfunction formulation of the two-dimensional (2D) Navier–Stokes equations can be written as [91]

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \nabla^2 \omega, \tag{18}$$

where ω is the vorticity and ψ is the streamfunction. The vorticity-streamfunction formulation prevents the odd–even decoupling issues that might arise between pressure and velocity components and enforces the incompressibility condition. The kinematic relationship between vorticity and streamfunction is given by the following Poisson equation,

$$\nabla^2 \psi = -\omega. \tag{19}$$

Eqs. (18) and (19) include two operators, the Jacobian ($J(f, g)$) and the Laplacian ($\nabla^2 f$) defined as

$$J(f, g) = \frac{\partial f}{\partial x}\frac{\partial g}{\partial y} - \frac{\partial f}{\partial y}\frac{\partial g}{\partial x}, \tag{20}$$
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \tag{21}$$

Similar to the 1D Burgers problem, Eq. (18) can be rearranged as

$$\frac{\partial \omega}{\partial t} = \frac{1}{\mathrm{Re}} \nabla^2 \omega - J(\omega, \psi). \tag{22}$$

The reduced-rank approximations of the vorticity and streamfunction fields can be written as follows

$$\omega(x, y, t) \approx \sum_{k=1}^{R} a_k(t) \phi_k^{\omega}(x, y), \tag{23}$$
$$\psi(x, y, t) \approx \sum_{k=1}^{R} a_k(t) \phi_k^{\psi}(x, y). \tag{24}$$

We note that the vorticity and streamfunction share the same time-dependent coefficients ($a_k(t)$) since they are related through the kinematic relationship given by Eq. (19) (i.e., streamfunction is not a prognostic variable). Moreover, as POD preserves linear properties, the spatial modes for streamfunction can be obtained from the vorticity modes by solving the following Poisson equations

$$\nabla^2 \phi_k^{\psi}(x, y) = -\phi_k^{\omega}(x, y), \quad k = 1, 2, \ldots, R. \tag{25}$$

The GROM for the 2D Navier–Stokes equations is given by the same ODE (Eq. (17)) with the following coefficients

$$\mathfrak{L}_{i,k} = \left\langle \frac{1}{\mathrm{Re}} \nabla^2 \phi_i^{\omega}; \phi_k^{\omega} \right\rangle,$$
$$\mathfrak{N}_{i,j,k} = \left\langle -J(\phi_i^{\omega}, \phi_j^{\psi}); \phi_k^{\omega} \right\rangle.$$

Similar to the case of 1D Burgers problem, the basis functions are spatial functions and standard discretization techniques can be used to compute the required derivatives. Moreover, they might be arranged in a vector form to compute the inner product in Euclidean space. In 2D case, a reshaping might be necessary to form the vector version of $\phi_k(x, y)$, e.g., $[\phi_k(x_1, y_1), \phi_k(x_2, y_1), \ldots \phi_k(x_{N_x}, y_1), \phi_k(x_1, y_2), \phi_k(x_2, y_2), \ldots, \phi_k(x_{N_x}, y_2), \ldots, \phi_k(x_1, y_{N_y}), \ldots, \phi_k(x_{N_x}, y_{N_y})]^T$, where N_x and N_y are the number of grid points in the x and y directions, respectively.

Due to the modal truncation and inherent nonlinearity in Eq. (17), GROM no longer represents the same system (i.e., it solves a different problem). As a result, the obtained trajectory from solving the ROM deviates from the projected trajectory, as shown in Fig. 1. Therefore, the optimality of the POD basis is lost. Moreover, due to the triadic nonlinear interactions, instabilities can occur in GROMs. To mitigate these problems, closure and/or stabilization techniques are usually required to obtain accurate results. Increasing the ROM dimension can improve the results. However, due to the nonlinearity of the resulting ROM, the computational cost of GROM is $O(R^3)$, which severely constrains the ROM dimension used in practical applications.

We note here that we are adopting the tensorial GROM approach [93], where the coefficients $\mathfrak{L}$ and $\mathfrak{N}$ are computed offline. Other approaches can include online computations while incorporating the empirical interpolation method (EIM) [94] or its discrete version, the discrete empirical interpolation method (DEIM) [95] to reduce the online computational cost.

Although, we focus on the standard GROM with POD in the present study, several studies have been devoted to addressing the computational cost and stability/accuracy trade-off for advection-dominated flows. For example, decomposing the time domain via a principal interval decomposition approach can produce localized basis functions and tailor more representative compact subspaces [96,97], which improves the ROM accuracy and keeps the online computational cost minimal. Rather than being restricted to a linear basis, auto-encoders can be used to compute nonlinear subspaces to approximate the solution manifold [98,99]. Also, Grimberg et al. [100] recently demonstrated that many ROM instabilities can be attributed to the standard Galerkin projection, and the Petrov–Galerkin approach can mitigate these instabilities and provide more accurate ROM [87–90].

5. Uplifted reduced order modeling

As noted in Section 4, the computational cost of GROM is $O(R^3)$, which limits the number of modes to be used in the ROM. This modal truncation has two major consequences. First, the flow field variable is constrained to lie in a small subspace, spanned by the very first few modes. For convection-dominated flows or parametric problems characterized by slow decay of the Kolmogorov n-width, these few modes may be less representative of the true physical system, which might significantly reduce the accuracy of the resulting ROM. This is shown as the projection error $\mathcal{E}_{\Phi^\perp}$ in Fig. 1, since the truncated modes are orthogonal to the subspace spanned by $\mathbf{\Phi}$. Second, due to the inherent nonlinearity, the truncated modes (or scales) interact with the retained ones. Thus, this truncation simply ignores these interactions, often giving rise to numerical instabilities of solution. This error is represented as $\mathcal{E}_{\Phi}$ in Fig. 1 since it lies in the same subspace $\mathbf{\Phi}$. In our uplifted reduced order modeling (UROM) framework (presented in Fig. 2), we try to address these two problems.

We extend our reduced order approximation to include the first Q modes, where $Q > R$, assuming that the first R modes account for the resolved large scales, and the next $(Q - R)$ modes

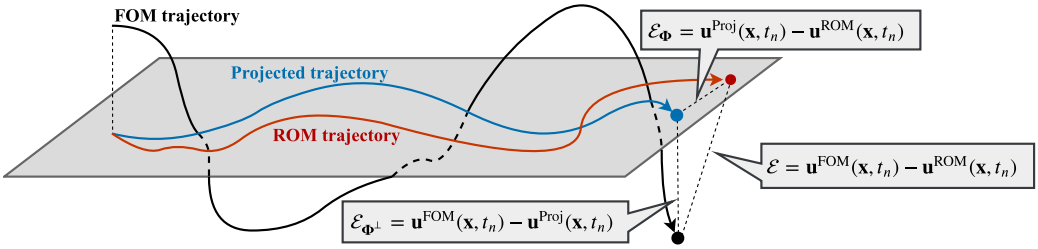

Fig. 1. A representation of error sources in ROM (e.g., see [77,92] for further details). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

represent the resolved small scales (while the remaining truncated modes account for the unresolved scales). So, the UROM approximation of the true field $\mathbf{u}(\mathbf{x}, t_n)$ can be expanded as

$$\mathbf{u}(\mathbf{x}, t_n) \approx \underbrace{\sum_{k=1}^{R} a_k(t)\phi_k(\mathbf{x})}_{\substack{\text{core} \\ \text{(resolved large scales)}}} + \underbrace{\sum_{k=R+1}^{Q} a_k(t)\phi_k(\mathbf{x})}_{\substack{\text{uplift} \\ \text{(resolved small scales)}}}, \tag{26}$$

where $\mathbf{u}$ is a general notation for the flow field of interest, and ϕ_k denotes the POD modes. a_k represents the corresponding temporal coefficients (sometimes called the generalized coordinates), defined as the projection of the field $\mathbf{u}$ onto the basis function ϕ_k,

$$a_k(t) = \langle \mathbf{u}(\mathbf{x}, t); \phi_k(\mathbf{x}) \rangle. \tag{27}$$

Before presenting the UROM framework, we first briefly revisit the Galerkin ROM (GROM). Here, we denote the GROM solution as $\tilde{a}_k(t_n)$, where the initial condition $\tilde{a}_k(t_0)$ can be computed from the projection of the initial field (at t_0) onto the POD modes (by using Eq. (27)). Then, GROM is used to evolve $\tilde{a}$ in time as

$$\frac{\mathrm{d}\tilde{a}_k}{\mathrm{d}t} = G(\tilde{a}_k), \tag{28}$$

which can be numerically solved using a time-stepping integrator,

$$\tilde{a}_k(t_{n+1}) = \tilde{a}_k(t_n) + \Delta t \sum_{q=0}^{s} \beta_q G(\tilde{a}_k(t_{n-q})), \tag{29}$$

where s and β_q depend upon the numerical scheme used for the time integration. In the present study, we use the third-order Adams–Bashforth (AB3) method, for which $s = 2$, $\beta_0 = 23/12$, $\beta_1 = -16/12$, and $\beta_2 = 5/12$. Here, $G(\tilde{a}_k)$ is obtained by Galerkin projection (e.g., see Section 4) as

$$G(\tilde{a}_k(t_n)) = \sum_{i=1}^{R} \mathfrak{L}_{i,k}\tilde{a}_i(t_n) + \sum_{i=1}^{R}\sum_{j=1}^{R} \mathfrak{N}_{i,j,k}\tilde{a}_i(t_n)\tilde{a}_j(t_n). \tag{30}$$

However, due to the modal truncation and incurred errors and instabilities of GROM (as discussed in Section 4), the resulting predictions $\tilde{a}_k$ from Eqs. (28)–(30) become erroneous.

UROM builds on the GROM, but considers the output of GROM at each time step as a first predictor of a_k rather than the final approximation. In other words, starting from an initial condition $a_k(t_0)$, we use the same GROM structure (being physics-inspired) to evolve one time step. In standard GROM, this would be considered the prediction at t_1. Instead, we denote this as $\hat{a}_k(t_1)$ and treat it as just an initial guess. Then, a correction term is introduced to steer $\hat{a}_k(t_1)$ to better approximate $a_k(t_1)$. This corrected value is subsequently fed back into GROM structure to evaluate $\hat{a}_k(t_2)$, which is then corrected (and so on). Thus, at any time t_n, the best-known value of temporal coefficients (after corrections) is denoted as $a_k(t_n)$. This is used to compute an initial guess $\hat{a}_k(t_{n+1})$ for $a_k(t_{n+1})$ for $k = 1, 2, \ldots, R$ (i.e., the large scales) using

$$\hat{a}_k(t_{n+1}) = a_k(t_n) + \Delta t \sum_{q=0}^{s} \beta_q G(a_k(t_{n-q})). \tag{31}$$

After $\hat{a}_k(t_{n+1})$ is computed from Eq. (31), a correction might be introduced before evolving GROM to the next time step, i.e., $a_k(t_{n+1}) = \hat{a}_k(t_{n+1}) + c_k(t_{n+1})$, where c_k can be considered as the difference between the physics-based model estimate and the true projection. In other words, the corrected temporal coefficient is assumed to be the true value of a_k (or at least the best-known value), which is therefore used as input to GROM to evolve one time step further. That is why the corrected a_k values are used to compute the right hand side of Eq. (31).

In order to correct GROM results for the first R modes, closure and/or stabilization are required. In our framework, we propose the use of LSTM architecture to learn a correction term to steer the GROM prediction of the modal coefficients $\{\hat{a}_k(t_n)\}_{k=1}^{R}$ to the true values $\{a_k(t_n)\}_{k=1}^{R}$ at each time step. In other words, an LSTM is trained to learn the map from $\{\hat{a}_k(t_n)\}_{k=1}^{R}$ as input to $\{c_k(t_n)\}_{k=1}^{R}$ as output, where c is a correction (closure) term defined as

$$c_k(t_n) = a_k(t_n) - \hat{a}_k(t_n). \tag{32}$$

It should be noted here that the introduced data driven closure takes into account the interactions of *all* fine scales ($k = R+1, \ldots, N_s$) with the resolved large scales ($k = 1, \ldots, R$), as manifested in the data snapshots. Finally, to account for small scales, we train a second super-resolution LSTM neural network to predict the modal coefficients of the next $(Q-R)$ modes, where the input of the LSTM is $\{a_k(t_n)\}_{k=1}^{R}$ and the output is $\{a_k(t_n)\}_{k=R+1}^{Q}$. To improve the parametric performance of the UROM architecture and promote generality, the LSTMs' inputs are augmented with the control parameter. Therefore, the LSTM maps f and g corresponding to the closure and super-resolution models, respectively, can be written as

$$f : \begin{bmatrix} \mu \\ \hat{a}_1(t_n) \\ \vdots \\ \hat{a}_R(t_n) \end{bmatrix} \mapsto \begin{bmatrix} c_1(t_n) \\ \vdots \\ c_R(t_n) \end{bmatrix}, \qquad g : \begin{bmatrix} \mu \\ a_1(t_n) \\ \vdots \\ a_R(t_n) \end{bmatrix} \mapsto \begin{bmatrix} a_{R+1}(t_n) \\ \vdots \\ a_Q(t_n) \end{bmatrix}. \tag{33}$$

In brief, we first steer the red line in Fig. 1 to the blue one (i.e., introduce data-driven closure by LSTM). Then, we reduce the projection error (difference between the blue and black lines) by expanding our solution subspace to span Q modes rather than only R. Fig. 2 demonstrates the building blocks and workflow of our proposed UROM framework in both the offline and online phases, which can be described as follows.

During offline training, we suppose that we have access to the true fields at different time instants (those can come from experiments or numerical simulations). Thus, we can obtain the

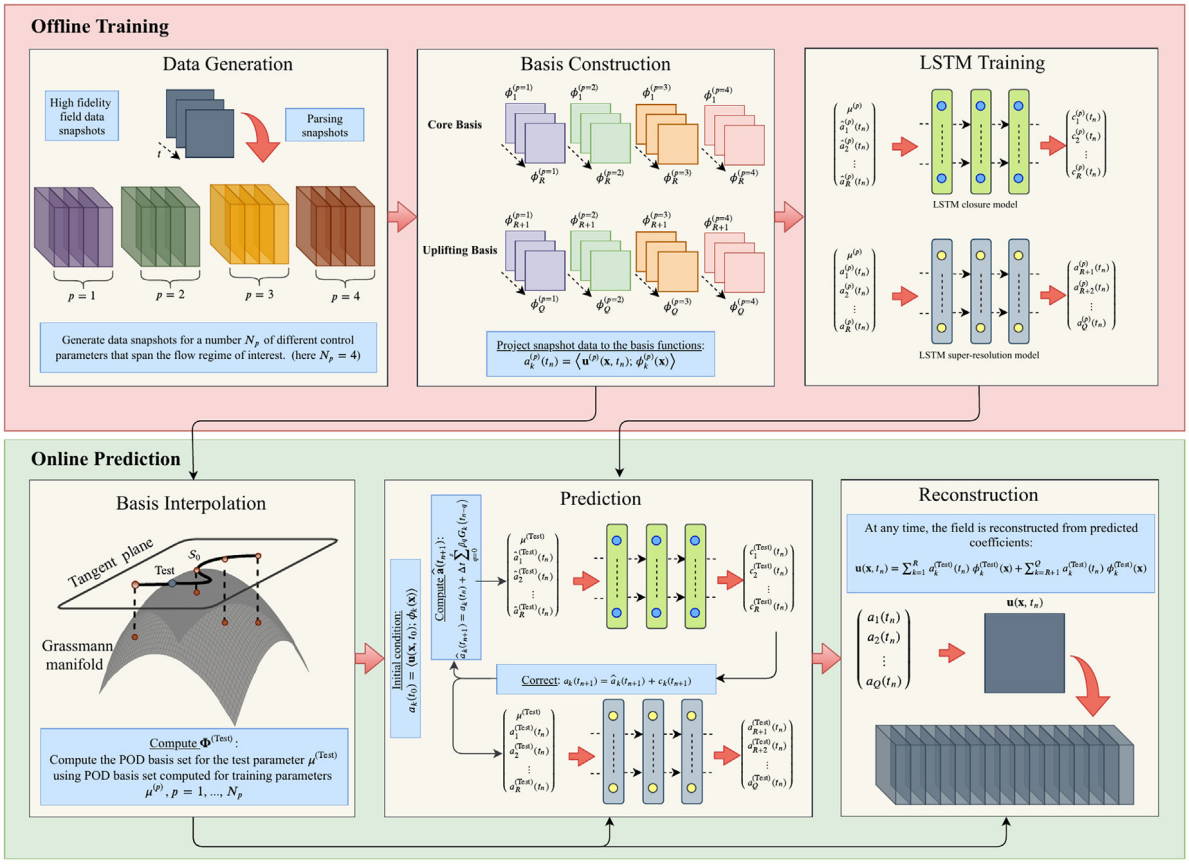

Fig. 2. A schematic diagram for the workflow of UROM framework.

true modal coefficients a_k (see Eq. (27)). Moreover, we use the GROM equations to evolve the modal coefficients for one time step (in that sense, GROM can be viewed as a mapping from $a_k(t_n)$ to $\hat{a}_k(t_{n+1})$). Then, a correction term can be computed as the difference between $a_k(t_{n+1})$ and $\hat{a}_k(t_{n+1})$ for $k = 1, 2, \ldots, R$. Therefore, a corrector LSTM is trained to learn the mapping from $\hat{a}_k$ to c_k. Also, a super-resolver LSTM is trained to map a_k for $k = 1, 2, \ldots, R$ to a_k for $k = R+1, R+2, \ldots, Q$.

During online deployment (actual testing), we start with the initial field (at time zero) and project it onto the first R modes, to obtain the true $a_k(t_0)$. Then, GROM is used to evolve these R coefficients for one time step to obtain $\hat{a}_k(t_1)$ for $k = 1, 2, \ldots, R$. At this point, the corrector LSTM is fed by $\hat{a}_k(t_1)$ to output $c_k(t_1)$, and the corrected modal coefficients are computed as $a_k(t_1) = \hat{a}_k(t_1) + c_k(t_1)$ for $k = 1, 2, \ldots, R$. Those values are again fed to GROM to compute $\hat{a}_k(t_2)$, which are subsequently corrected by the corrector LSTM, and so on. Finally, before we reconstruct the full order field at any time instant t_n of interest, we utilize a super-resolver LSTM to recover the finer field scales. This super-resolver LSTM is fed with the corrected values $(a_k(t_n))$ for $k = 1, 2, \ldots, R$, representing the large scale dynamics resulting from the GROM and the LSTM corrector. Then, $a_k(t_n)$ for $k = R+1, R+2, \ldots, Q$, are obtained as output from this super-resolver. The workflow for online deployment is shown in Fig. 3. Note the recursive nature of the deployment, where the corrected values are fed back to GROM to advance one more time step. We also emphasize that the super-resolver is only used at instants of interest (i.e., it is not necessarily required at each time step), which makes further computational savings possible.

Few merits of the proposed UROM approach can be listed as follows.

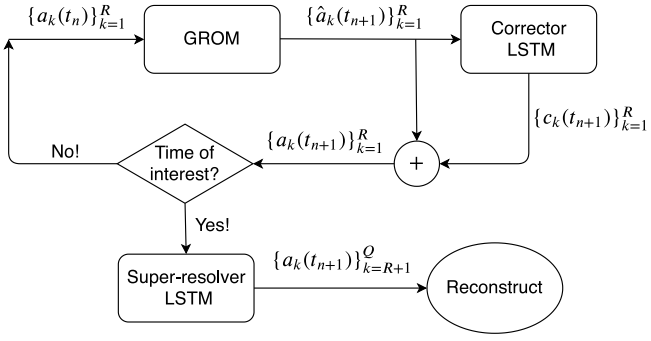

Fig. 3. A schematic diagram for the online deployment of UROM approach. Note that $\{a_k(t_n)\}_{k=1}^R$ is a short-hand notation for $a_1(t_n), a_2(t_n), \ldots, a_R(t_n)$.

- The physics-constrained GROM is maintained to account for the large scales. This enriches the framework interpretability and generalizability across a wide range of control parameters.
- GROM acts on a few modes, minimizing the online computational cost (i.e., $O(R^3)$, where $R < Q$).
- GROM, being physics-informed, can be used as a sanity check to decide whether or not the LSTM predictions should be considered.
- Data-driven closure/correction encapsulates information from *all* interacting modes and mechanisms.
- Since both LSTMs are fed with input from a physics-based approach, UROM can be considered as a way of enforcing physical knowledge to enhance data-driven tools.

Table 1
A list of hyperparameters utilized to train the LSTM network for all numerical experiments.

Variables	1D Burgers	2D Navier–Stokes
Number of hidden layers	3	3
Number of neurons in each hidden layer	60	80
Number of lookbacks	3	3
Batch size	64	64
Epochs	200	200
Activation functions in the LSTM layers	tanh	tanh
Validation data set	20%	20%
Loss function	MSE	MSE
Optimizer	ADAM	ADAM
Learning rate	0.001	0.001
First moment decay rate	0.9	0.9
Second moment decay rate	0.999	0.999

- LSTMs' inputs are augmented with the control parameter to provide a more accurate mapping (sometimes also called physics-guided mapping).

5.1. Long short-term memory embedding

To learn the maps f and g in UROM, we incorporate memory embedding through the use of LSTM architecture. LSTM is a variant of recurrent neural networks capable of learning and predicting the temporal dependencies between given data sequences based on the input information and previously acquired information. Recurrent neural networks have been used successfully in ROM community to enhance standard projection ROMs [101] and build fully non-intrusive ROMs [96,102–106]. In the present study, we use LSTMs to augment the standard physics-informed ROM by introducing closure as well as super-resolution data-driven models. We utilize Keras API [107] to build the LSTMs used in our UROM approach. Details about the LSTM architecture can be found in [96,104]. A summary of the adopted hyperparameters is presented in Table 1. We also found that the constructed neural networks are not very sensitive to hyperparameters. Meanwhile, for optimal hyperparameter selection, different techniques (e.g., gridsearch) can be used to tune them.

6. Results

In order to demonstrate the features and merits of UROM, we present results for the two test cases at out-of-sample control parameters (interpolatory and extrapolatory). For the number of modes, we use $R = 4$ for the core dynamics and $Q = 16$ for super-resolution. We compare the accuracy of UROM prediction with the FOM results as well as the true projection of the FOM snapshots on the POD subspace (denoted as 'True' in our results), where

$$a_k^{\text{True}}(t_n) = \left\langle \mathbf{u}(\mathbf{x}, t_n); \phi_k(\mathbf{x}) \right\rangle, \tag{34}$$

$$\mathbf{u}^{\text{True}}(\mathbf{x}, t_n) = \sum_{k=1}^{Q} a_k^{\text{True}}(t_n)\phi_k(\mathbf{x}). \tag{35}$$

Since UROM can be considered as a hybrid approach between fully intrusive and fully non-intrusive ROMs, we compare it with standard Galerkin projection ROM with 4 and 16 modes, denoted as GROM(4) and GROM(16), respectively. Moreover, we show the results of a fully non-intrusive ROM approach using 16 modes (denoted as NIROM). For this NIROM, we use the same LSTM architecture presented in Section 5.1 as a time-stepping integrator. In particular, we learn a map between the values of modal coefficients at current time step and their values at the following time step. Also, we augment our input with the control parameter to enhance the mapping accuracy. In other words, the NIROM map h can be represented as follows

$$h : \begin{bmatrix} \mu \\ a_1(t_n) \\ \vdots \\ a_Q(t_n) \end{bmatrix} \mapsto \begin{bmatrix} a_1(t_{n+1}) \\ \vdots \\ a_Q(t_{n+1}) \end{bmatrix}. \tag{36}$$

Finally, we present the CPU time for UROM, GROM(4), GROM(16), and NIROM to demonstrate the computational gain. For interested readers, we also provide a GitHub repository (https://github.com/Shady-Ahmed/UROM) describing a Python implementation of UROM as well as the reproduction of the numerical experiments discussed in the present study.

6.1. 1D Burgers problem

For 1D Burgers simulation, we consider the initial condition [108]

$$u(x, 0) = \frac{x}{1 + \exp\left(\frac{\text{Re}}{16}\left(4x^2 - 1\right)\right)}, \tag{37}$$

with $x \in [0, 1]$. The 1D Burgers equation with the above initial condition and Dirichlet boundary conditions has the following analytic solution representing a traveling wave [108]

$$u(x, t) = \frac{\frac{x}{t+1}}{1 + \sqrt{\frac{t+1}{t_0}}\exp\left(\text{Re}\frac{x^2}{4t+4}\right)}, \tag{38}$$

where $t_0 = \exp(\text{Re}/8)$. For offline training, we obtain solutions for different Reynolds numbers ($\text{Re} \in \{200, 400, 600, 800\}$). Data generation is performed using the analytic solution given in Eq. (38) after dividing the spatial domain $[0, 1]$ into 1024 equally-spaced spatial intervals (i.e., $N_x = 1025$). For each case, we collect 1000 snapshots for $t \in [0, 1]$ (i.e., $\Delta t = 0.001$). That is, a snapshot matrix of $\{\mathbf{u}(t_0), \mathbf{u}(t_2), \ldots, \mathbf{u}(t_{1000})\}$ is formed, where $\mathbf{u}(t_n)$ is the velocity field $u(x, t_n)$ collected as a column vector. Then, the POD basis functions are computed using the technique presented in Section 2. For online deployment, we obtain the POD basis at $\text{Re} = 500$ and $\text{Re} = 1000$ using Grassmann manifold interpolation as discussed in Section 3.

6.1.1. $Re = 500$: demonstrating interpolatory capability

A Reynolds number of 500 represents an interpolatory case, where we use the POD basis at $\text{Re} = 600$ as our reference point for basis interpolation. The evolution of the first 4 POD temporal coefficients using different frameworks is shown in Fig. 4. It is clear that GROM(4) is incapable of capturing the true dynamics due to the severe modal truncation. On the other hand, both GROM(16) and UROM show very good results; however, GROM(16) is more computationally expensive as will be shown in Section 6.3.

For field reconstruction, we present the temporal field evolution in Fig. 5 for FOM snapshots, true projection, UROM, GROM, and NIROM. It can be seen that UROM gives very good predictions for field reconstruction compared with GROM(4) and NIROM, which yield less accurate results. For better visualizations, we show the final field (i.e., at $t = 1$) in Fig. 6 with a close-up view on the region characterizing the wave-shock.

6.1.2. $Re = 1000$: demonstrating extrapolatory capability

In order to investigate the extrapolatory performance of UROM, we test the approach at $\text{Re} = 1000$, with the basis at $\text{Re} = 800$ as reference point for interpolation. The POD modal coefficients are shown in Fig. 7, where we can see that both UROM and GROM(16)

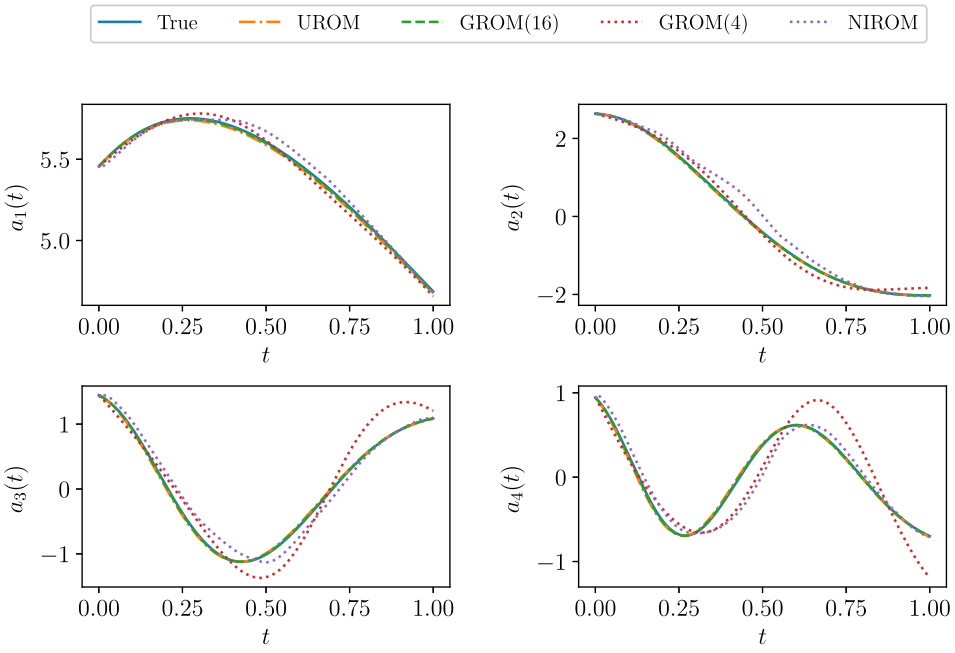

Fig. 4. Temporal evolution of the first 4 POD modal coefficients for Burgers problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re = 500. Note that GROM(4) and NIROM yield inaccurate results.

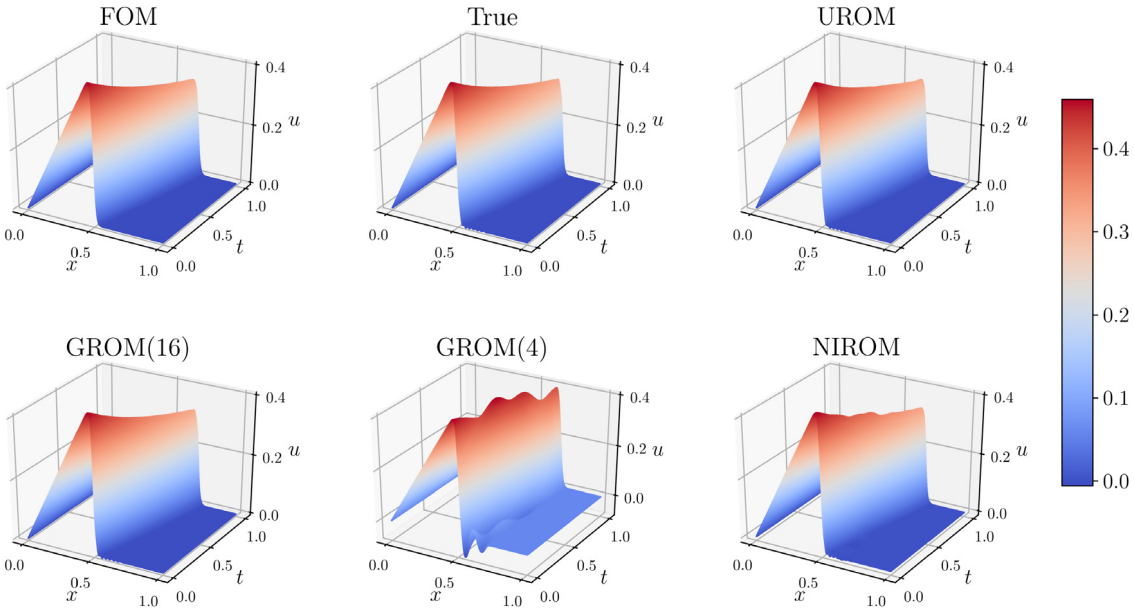

Fig. 5. Temporal evolution of velocity fields for Burgers problem at Re = 500 with $R = 4$ and $Q = 16$.

are still capable of capturing the true projected trajectory. Interestingly, the NIROM predictions are less satisfactory, giving non-physical behavior at some time instants. This suggests that the physical core of UROM promotes its generality, compared to the totally data-driven NIROM approach. However, we note that the deficient behavior of NIROM can be partly due to the sub-optimal architecture of our network as we only use the same hyperparameters (except for the size of input and output layers) for all simulations (as given in Table 1). More sophisticated architectures and further tuning of hyperparameters would probably improve NIROM predictions.

The temporal field evolution of flow field is shown in Fig. 8, which illustrates the non-physical and unstable behavior of both GROM(4) and NIROM approaches. The final field is plotted in Fig. 9 with a close-up view at the right. It can be seen that even the true projected fields do not match the FOM and show some fluctuations at the shock region. For this type of behavior, a larger subspace is required to capture most of the dynamics of the flow at Re = 1000.

As indicated in the previous discussion, a more sophisticated architecture for LSTM and elegant tuning of the hyperparameters would be needed to get acceptable performance for NIROM. It is quite common that NIROM suffers in long time predictions. In other words, during training and validation, the LSTM learns a map from the *true* modal coefficients to their values after one time step. Therefore, the network is always supposed to be fed with the true values. However, during actual deployment in the testing phase, the network is fed with true values *only* at the

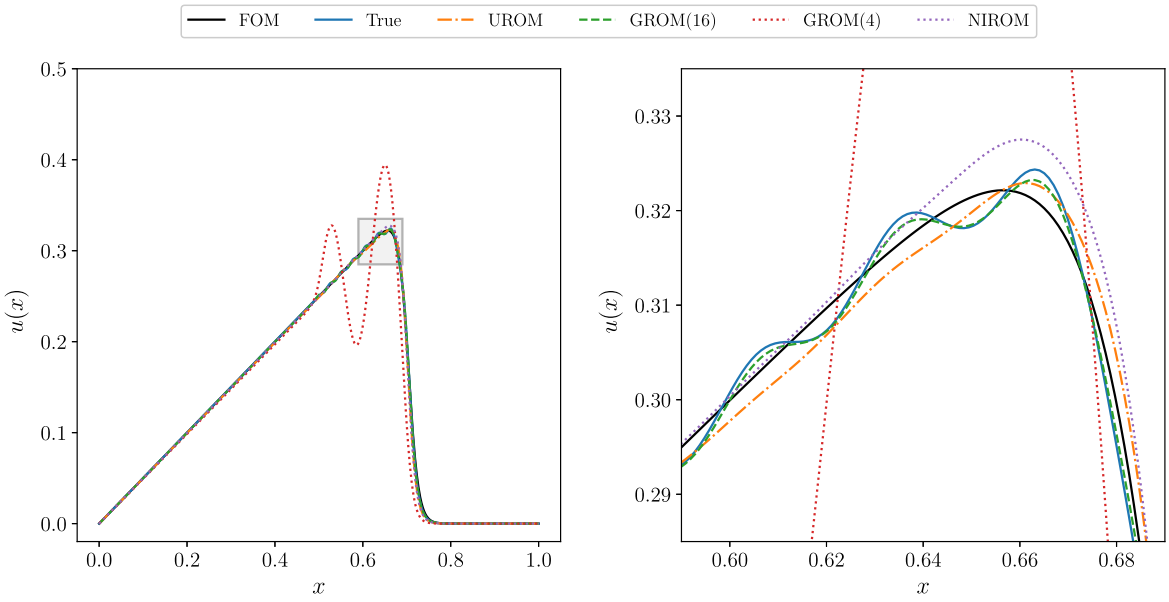

Fig. 6. Final velocity fields (at $t = 1$) for Burgers problem at Re $= 500$ with a zoom-in view at the right using $R = 4$ and $Q = 16$. Note that UROM is giving smooth predictions while GROM(4) is showing significant oscillations.

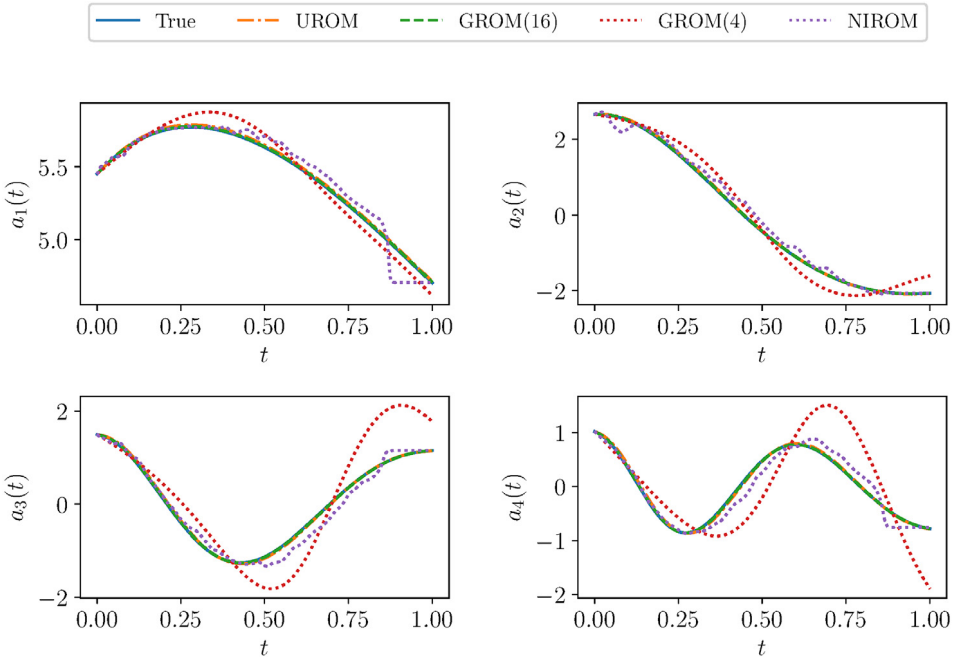

Fig. 7. Temporal evolution of the first 4 POD modal coefficients for Burgers problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re $= 1000$. GROM(4) deviates from true trajectory while NIROM gives non-physical predictions.

initial time (t_0). Then, the output of the network is returned back as input in a recursive manner to provide long time predictions. Thus, when the network encounters any error in the output (which is to be expected for testing at different parameters/regions), this error is amplified in the subsequent time steps. Being fully non-intrusive, the network has no way to account for errors. As a result, after a few time steps, the output of the LSTM might blow-up giving non-physical results, unless the network is stabilized. As a demonstration, we show in Fig. 10 the predictions for NIROM at Re $= 1000$ with a simple variation of architectures (different number of layers and neurons). We note that all these combinations give a converging performance during training/validation, where the training and validation losses fall below 1×10^{-5}. On the other hand, they are not doing very well during actual testing, in the presence of numerical errors and instabilities. In that sense, a hybridization between physics-based and data-driven models helps to stabilize the predictions during the online deployment phase. In the rest of the paper, we use the same architecture for NIROM and UROM, but the reader should be aware of these issues, which suggest the need for the development of more involved architectures and/or more elegant training and validation.

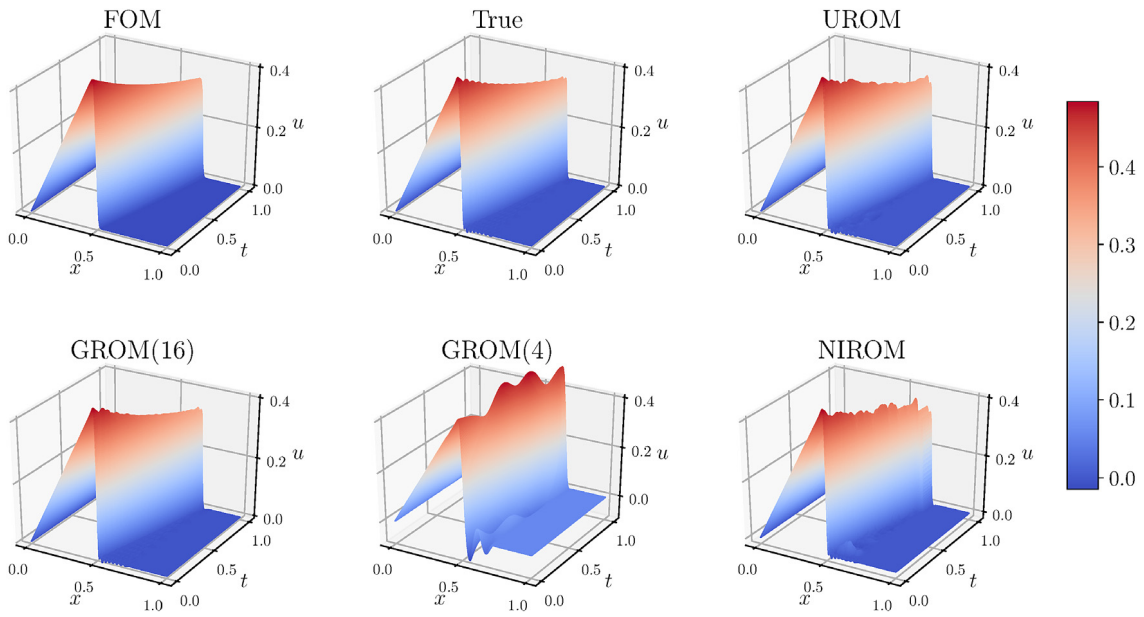

Fig. 8. Temporal evolution of velocity fields for Burgers problem at Re = 1000 with $R = 4$ and $Q = 16$.

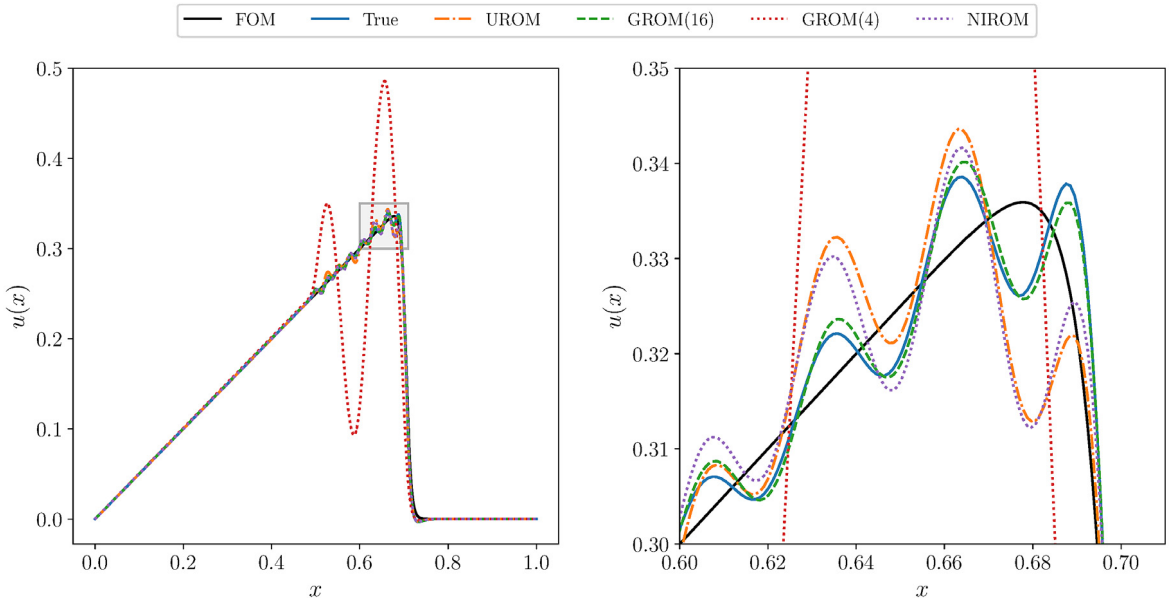

Fig. 9. Final velocity fields (at $t = 1$) for Burgers problem at Re = 1000 with a zoom-in view at the right using $R = 4$, and $Q = 16$. Oscillations in UROM and GROM(16) occur mainly because a subspace spanning the first 16 modes is insufficient to capture the dynamics at this Reynolds number.

6.2. 2D vortex merger problem

As an application for 2D Navier–Stokes equations, we examine the vortex merger problem (i.e., the merging of co-rotating vortex pair) [109]. The merging process occurs when two vortices of the same sign with parallel axes are within a certain critical distance from each other, ending as a single, nearly axisymmetric, final vortex [110]. We consider an initial vorticity field of two Gaussian-distributed vortices with a unit circulation as follows,

$$\omega(x, y, 0) = \exp\left(-\rho\left[(x - x_1)^2 + (y - y_1)^2\right]\right) + \exp\left(-\rho\left[(x - x_2)^2 + (y - y_2)^2\right]\right), \tag{39}$$

where ρ is an interacting constant set as $\rho = \pi$ and the vortices centers are initially located at $(x_1, y_1) = \left(\frac{3\pi}{4}, \pi\right)$ and $(x_2, y_2) = \left(\frac{5\pi}{4}, \pi\right)$. We use a Cartesian domain $(x, y) \in [0, 2\pi] \times [0, 2\pi]$ over a spatial grid of 256×256, with periodic boundary conditions. For this 2D problem, we collect 200 snapshots for $t \in [0, 20]$, while varying Reynolds number as Re $\in \{200, 400, 600, 800\}$. For solving the full order model equations, we use a third-order Arakawa scheme [111] for spatial derivatives, and a third-order total variation diminishing Runge–Kutta scheme (TVD-RK3) [112] for temporal integration. Similar to the 1D Burgers problem, we test our framework at Re = 500 and Re = 1000. Also, for basis interpolations, we use reference points at Re = 600 and Re = 800, respectively.

6.2.1. Re = 500: demonstrating interpolatory capability

Fig. 11 shows the temporal evolution of the first 4 POD coefficients for the vorticity field. Recall that the temporal coefficients

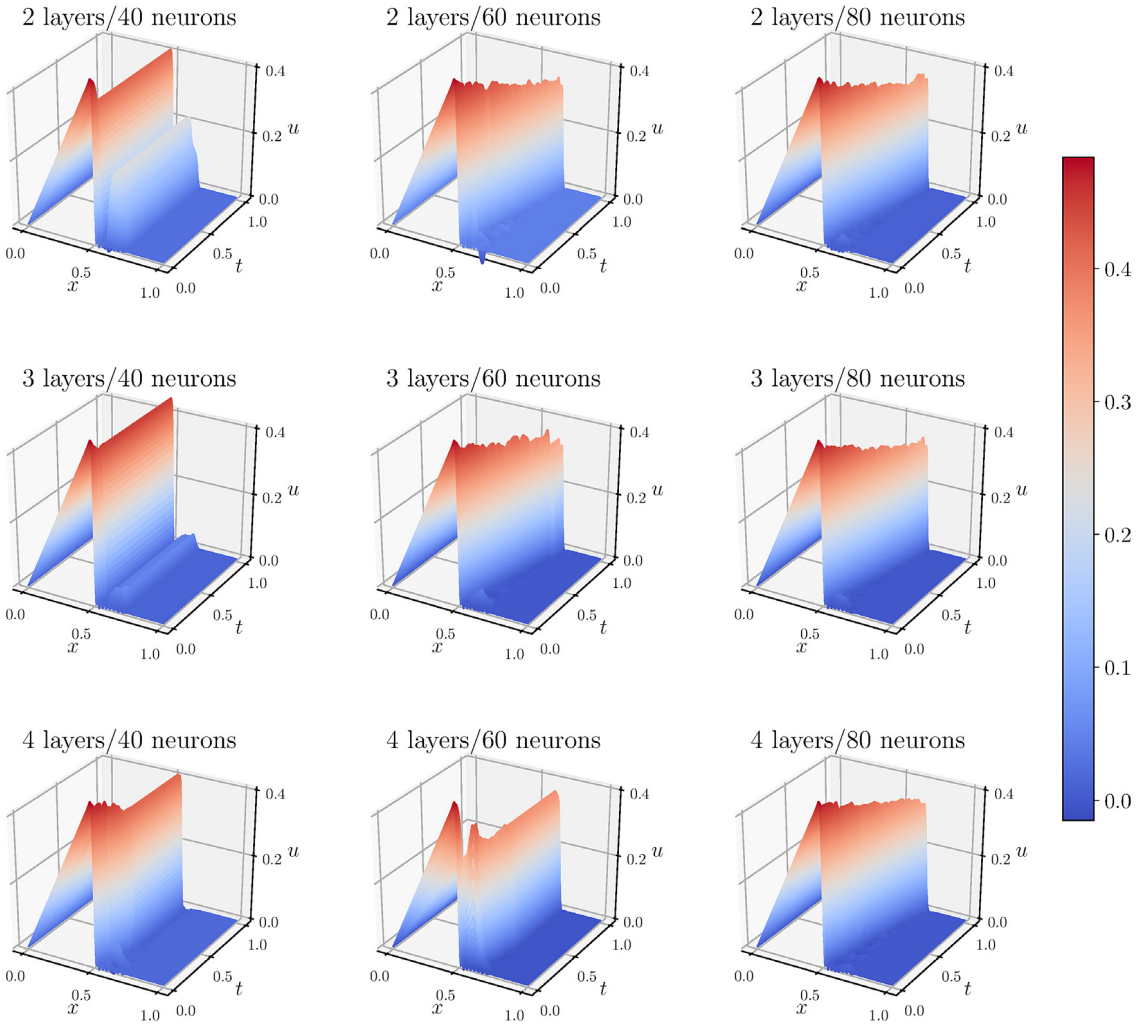

Fig. 10. NIROM predictions for Burgers problem at Re = 1000 with $Q = 16$, and different numbers of layers and neurons.

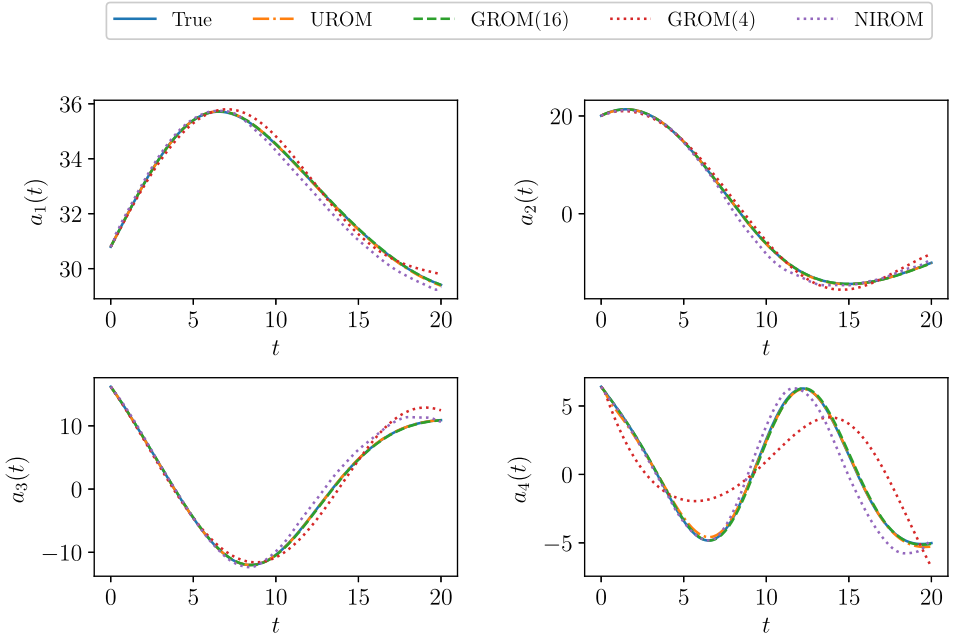

Fig. 11. Temporal evolution of the first 4 POD modal coefficients of vorticity field for 2D vortex merger problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re = 500.

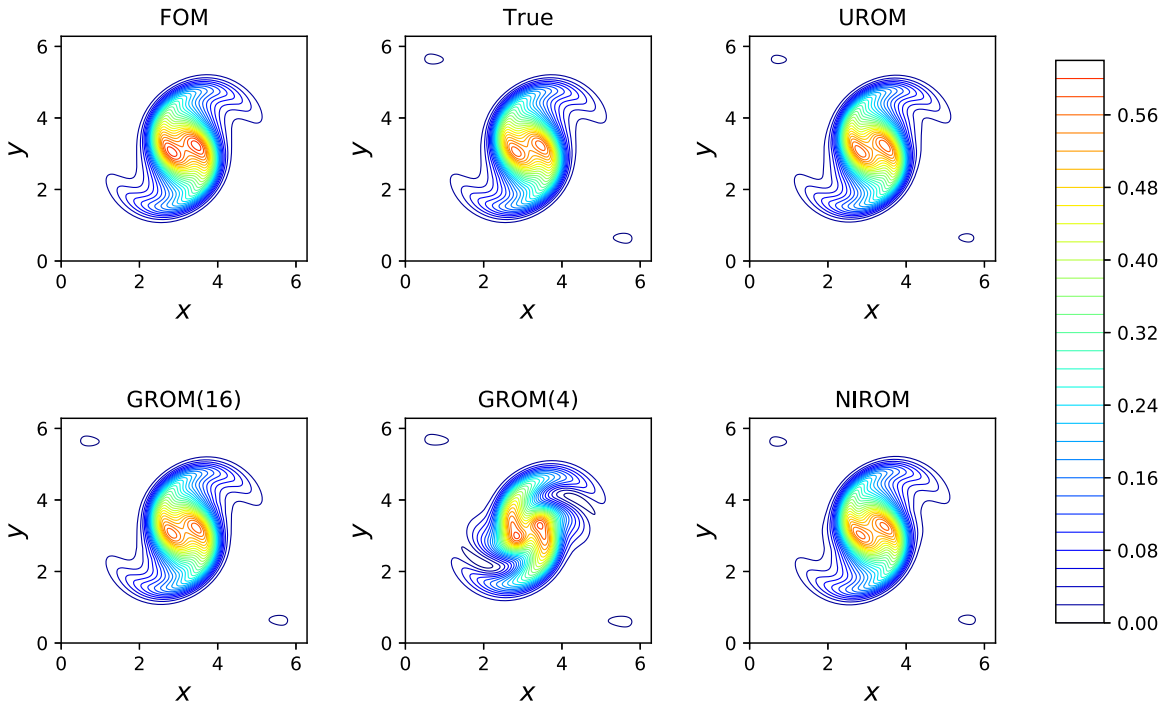

Fig. 12. Final vorticity fields (at $t = 20$) for 2D vortex merger problem at $\mathrm{Re} = 500$ with $R = 4$, and $Q = 16$.

for vorticity and streamfunction fields are the same, as discussed in Section 4.2. We can see that both UROM and GROM(16) accurately predict the true modal dynamics. For better visualizations, the final vorticity field at $t = 20$ is given in Fig. 12, where GROM(4) is showing instabilities manifested in the reconstructed field. For this interpolatory case, NIROM is providing *acceptable* results.

6.2.2. Re = 1000: demonstrating extrapolatory capability

The investigated approaches, namely GROM, UROM and NIROM, are tested at $\mathrm{Re} = 1000$ as a case that is out-of-range compared to the training set. The POD modal coefficients predicted by these approaches are given in Fig. 13. It can be easily seen that as time increases, the predictions of GROM(4) and NIROM become poor. Using a neural network as time-stepping integrator in NIROM increases its sensitivity to computational noise and this recursive deployment accumulates the error until predictions totally depart from the true trajectory. This is even clearer in the reconstructed field shown in Fig. 14, where the orientation of the merging vortices is not matching the true orientation. GROM(4) prediction is also suffering from severe deformation of the true flow topology. On the other hand, the field reconstruction via UROM is accurate compared to the true projection and GROM(16).

6.3. Computing time

Finally, we report the "online" computing time for the investigated approaches. In particular, we show the computational time as well as the root-mean squared error (RMSE) of reconstructed fields at final time for the two test cases at $\mathrm{Re} = 1000$ in Table 2. The reported RMSE is computed as

$$\mathrm{RMSE}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{u}^{FOM}(\mathbf{x},t) - \mathbf{u}^{ROM}(\mathbf{x},t)\right)^2}, \tag{40}$$

where N represents the spatial resolution (i.e., $N = N_x \times N_y$). In this table, we also report the NIROM results using 4 modes in the input and output layers. Although GROM(4) is the fastest, its predictions are very poor and further corrections and stabilization might be required. Also, a subspace spanned by only the first four POD modes might be insufficient in complex applications. On the other hand, GROM(16) is the slowest. We can also observe that computing time of UROM is close to that of NIROM and much lower than GROM(16). In Fig. 15, we present a bar chart for both the computing time and RMSE of reconstructed fields at final time to illustrate the time-accuracy trade-off.

We note that Table 2 and Fig. 15 document the performance of our implementation rather than that of the approaches. We should emphasize that, in this paper, we are not aiming at benchmarking the computational performance of these approaches. Instead, our main objective is to demonstrate the feasibility of hybrid approaches fusing physics-based and machine learning models. Nonetheless, the runtimes in Table 2 indicate that GROM approximately (not exactly due to various other loading/writing abstractions in our Python implementations) scales with R^3. Therefore, combining NIROM and GROM, UROM yields better computational performance. We also note that, if written more optimally, we would also expect that the execution time of UROM (with 16 modes) can be reduced to the sum of the execution times of GROM (with 4 modes) and NIROM (with 16 modes). Indeed, we remark that the second LSTM in UROM (representing the map g) need not be used at all times and can be deployed only at the instant of interest. In that case, the UROM computing times become 1.31 s for Burgers case and 0.30 s for vortex merger (which are very close to NIROM computing time).

In a nutshell, our investigation, for the considered test cases, shows that GROM with 4 modes provides inaccurate results, while GROM with 16 modes gives good predictions. However, the computing cost of the latter is significantly higher than the computing cost of the former. Adopting the UROM approach, we are able to get an accuracy similar to the GROM(16) accuracy, but with a minimal computing cost. Moreover, UROM is more stable than NIROM for moderate LSTM architectures, since the UROM is always constrained in its core by the GROM update step. Conversely, NIROM is totally non-intrusive and the output is fed

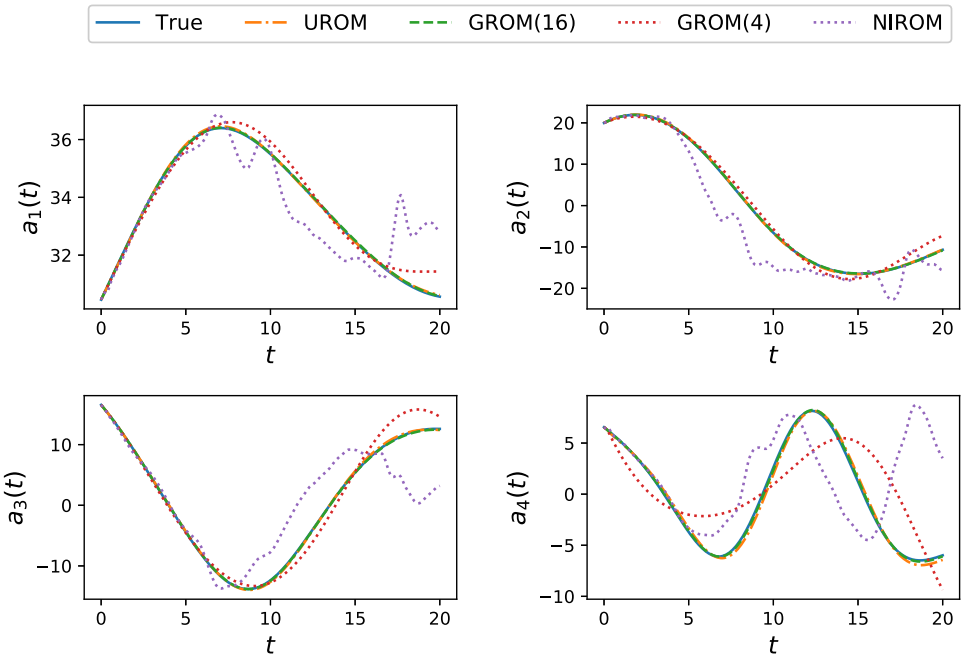

Fig. 13. Temporal evolution of the first 4 POD modal coefficients of vorticity field for 2D vortex merger problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re = 1000.

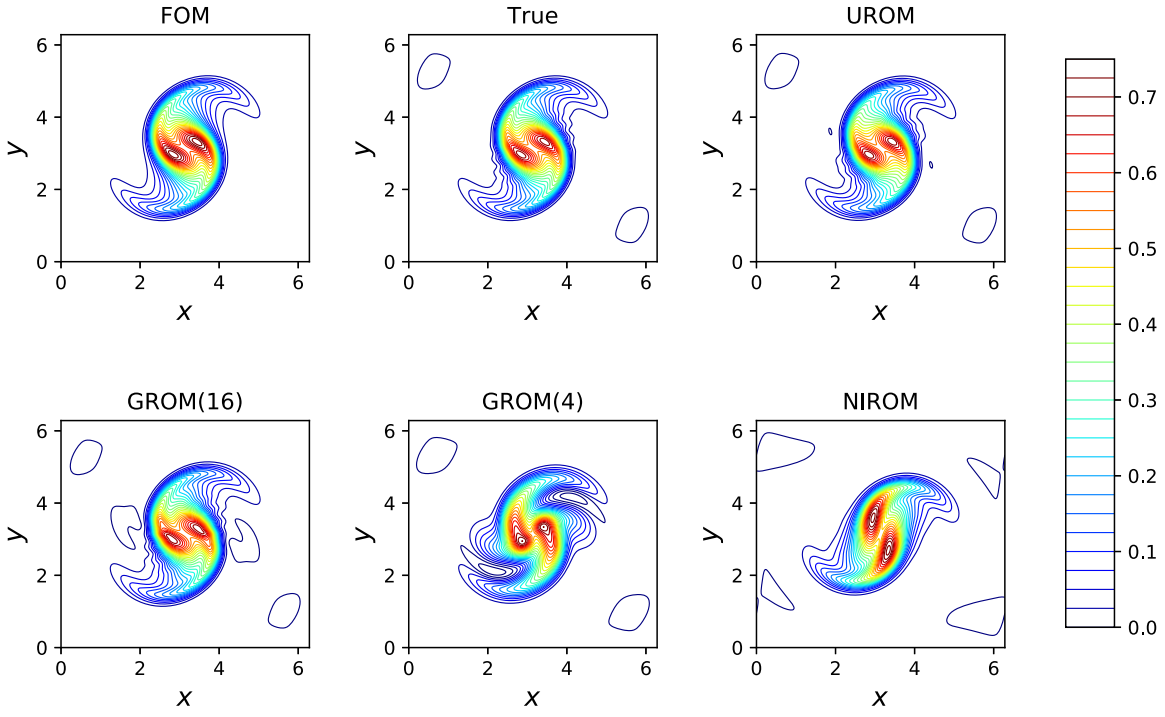

Fig. 14. Final vorticity fields (at $t = 20$) for 2D vortex merger problem at Re = 1000 with $R = 4$, and $Q = 16$.

back to the LSTM recursively, resulting in amplification of error for long time predictions. This framework can also be generalized to get higher accuracy or address more complex problems by increasing R and/or Q.

7. Concluding remarks

In the present study, we have proposed an uplifted reduced order modeling (UROM) approach to elevate the standard Galerkin projection reduced order modeling (GROM). This approach can be considered as a hybrid approach between physics-based and purely data-driven techniques. With GROM at the core of the framework, UROM (with three modeling layers) enhances the model generalizability and interpretability. Moreover, large scales (represented by the first few modes) are given due attention since they control most of the bulk mass, momentum, and energy transfers. Therefore, two out of a total of three layers in UROM aim at predicting the dynamics of these modes as accurately as possible. Then, an uplifting layer is designed to enhance the prediction resolution (i.e., super-resolution). Performance of

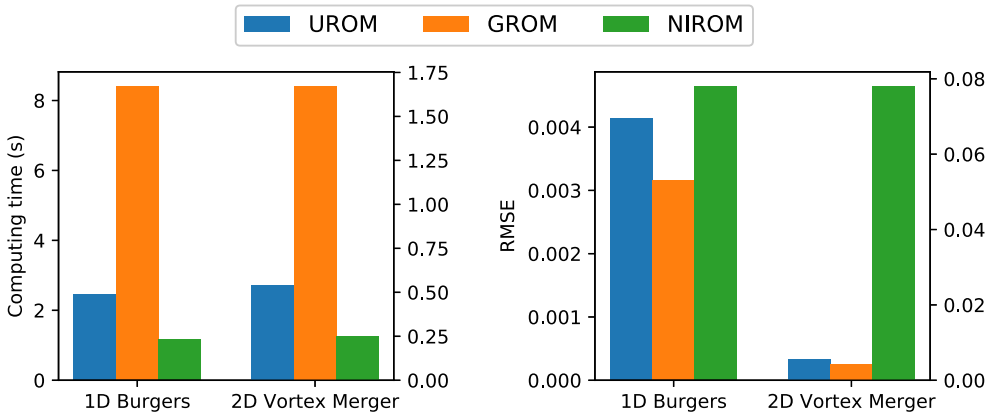

Fig. 15. Computing time of testing (online) stage at Re = 1000 (left) and RMSE of reconstructed fields at final time (right) for UROM(4 + 12) (i.e., $R = 4$ and $Q = 16$), GROM(16), and NIROM(16).

Table 2
Computing time (in seconds) and RMSE of UROM, GROM(4), GROM(16), NIROM(4), and NIROM(16) for Re = 1000. We note that the computing time assessments documented in this table are based on Python executions.

Framework	1D Burgers		2D vortex-merger	
	Time (s)	RMSE	Time (s)	RMSE
UROM	2.46	4.13E−3	0.54	5.44E−3
GROM(16)	8.40	3.17E−3	1.67	4.17E−3
GROM(4)	0.17	5.17E−2	0.06	3.99E−2
NIROM(16)	1.16	4.64E−3	0.25	7.80E−2
NIROM(4)	1.07	3.14E−2	0.23	8.58E−2

UROM has been compared against standard GROM and fully non-intrusive ROM (NIROM) approaches.

Two test cases, representing convection-dominated flows in 1D and 2D settings, have been used to evaluate the UROM. For testing, two out-of-sample control parameters have been investigated to study the interpolatory and extrapolatory performances. In all cases, UROM has showed very good results, compared to GROM and NIROM. In particular, UROM(4 + 12) has been demonstrated to provide more accurate results than both GROM(4) and NIROM. In contrast to NIROM where the deployment is fully data-driven, the LSTMs in UROM take their inputs from a physics-based approach. This can be considered as one way of leveraging physical information and intuition into LSTM. On the other hand, UROM has provided significant speed-ups compared to GROM(16) with comparable accuracy. Although we have presented the results for $Q = 16$, more complex flows can require much larger Q, which makes GROM(Q) unfeasible. Finally, this UROM approach is thought to open new avenues to utilize data-driven tools to enhance existing physical models as well as use physics to inform data-driven approaches to maximize the pros of both approaches and mitigate their cons.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award Number DE-SC0019290. O.S. gratefully acknowledges their support. The work of T.I. was supported by the National Science Foundation grant DMS-1821145. Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] V. Singh, K.E. Willcox, Engineering design with digital thread, AIAA J. 56 (11) (2018) 4515–4528.
[2] A. Rasheed, O. San, T. Kvamsdal, Digital twin: Values, challenges and enablers from a modeling perspective, IEEE Access 8 (2020) 21980–22012.
[3] R. Arcucci, L. Mottet, C. Pain, Y.-K. Guo, Optimal reduced space for variational data assimilation, J. Comput. Phys. 379 (2019) 51–69.
[4] Z. Bai, Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems, Appl. Numer. Math. 43 (1–2) (2002) 9–44.
[5] D.J. Lucia, P.S. Beran, W.A. Silva, Reduced-order modeling: new approaches for computational physics, Prog. Aerosp. Sci. 40 (1–2) (2004) 51–117.
[6] M. Hess, A. Alla, A. Quaini, G. Rozza, M. Gunzburger, A localized reduced-order modeling approach for PDEs with bifurcating solutions, Comput. Methods Appl. Mech. Engrg. 351 (2019) 379–403.
[7] B. Kramer, K.E. Willcox, Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition, AIAA J. 57 (6) (2019) 2297–2307.
[8] R. Swischuk, L. Mainini, B. Peherstorfer, K. Willcox, Projection-based model reduction: Formulations for physics-based machine learning, Comput. & Fluids 179 (2019) 704–717.
[9] J. Bouvrie, B. Hamzi, Kernel methods for the approximation of nonlinear systems, SIAM J. Control Optim. 55 (4) (2017) 2460–2492.
[10] B. Hamzi, E.H. Abed, Local modal participation analysis of nonlinear systems using Poincaré linearization, Nonlinear Dynam. (2019) 1–9.
[11] M. Korda, M. Putinar, I. Mezić, Data-driven spectral analysis of the Koopman operator, Appl. Comput. Harmon. Anal. (2018) http://dx.doi.org/10.1016/j.acha.2018.08.002.
[12] M. Korda, I. Mezić, Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control, Automatica 93 (2018) 149–160.
[13] D. Hartmann, M. Herz, U. Wever, Model order reduction a key technology for digital twins, in: Reduced-Order Modeling (ROM) for Simulation and Optimization, Springer, New York, 2018, pp. 167–179.
[14] S. Peitz, S. Ober-Blöbaum, M. Dellnitz, Multiobjective optimal control methods for the Navier-Stokes equations using reduced order modeling, Acta Appl. Math. 161 (1) (2019) 171–199.
[15] P. Holmes, J.L. Lumley, G. Berkooz, C.W. Rowley, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University Press, Cambridge, 2012.

[16] K. Taira, S.L. Brunton, S.T. Dawson, C.W. Rowley, T. Colonius, B.J. McKeon, O.T. Schmidt, S. Gordeyev, V. Theofilis, L.S. Ukeiley, Modal analysis of fluid flows: An overview, AIAA J. (2017) 4013–4041.
[17] K. Taira, M.S. Hemati, S.L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S.T. Dawson, C.-A. Yeh, Modal analysis of fluid flows: Applications and outlook, AIAA J. (2019) 1–25.
[18] B.R. Noack, M. Morzynski, G. Tadmor, Reduced-Order Modelling for Flow Control, Vol. 528, Springer, Berlin, 2011.
[19] C.W. Rowley, S.T. Dawson, Model reduction for flow analysis and control, Annu. Rev. Fluid Mech. 49 (2017) 387–417.
[20] N.J. Nair, M. Balajewicz, Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent shocks, Internat. J. Numer. Methods Engrg. 117 (12) (2019) 1234–1262.
[21] E. Kaiser, B.R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Östh, S. Krajnović, R.K. Niven, Cluster-based reduced-order modelling of a mixing layer, J. Fluid Mech. 754 (2014) 365–414.
[22] B. Haasdonk, M. Dihlmann, M. Ohlberger, A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space, Math. Comput. Model. Dyn. Syst. 17 (4) (2011) 423–442.
[23] M.A. Dihlmann, B. Haasdonk, Certified PDE-constrained parameter optimization using reduced basis surrogate models for evolution problems, Comput. Optim. Appl. 60 (3) (2015) 753–787.
[24] K. Ito, S.S. Ravindran, A reduced-order method for simulation and control of fluid flows, J. Comput. Phys. 143 (2) (1998) 403–425.
[25] A. Iollo, S. Lanteri, J.-A. Désidéri, Stability properties of POD-Galerkin approximations for the compressible Navier-Stokes equations, Theor. Comput. Fluid Dyn. 13 (6) (2000) 377–396.
[26] C.W. Rowley, T. Colonius, R.M. Murray, Model reduction for compressible flows using POD and Galerkin projection, Physica D 189 (1–2) (2004) 115–129.
[27] R. Milk, S. Rave, F. Schindler, PyMOR–generic algorithms and interfaces for model order reduction, SIAM J. Sci. Comput. 38 (5) (2016) S194–S216.
[28] V. Puzyrev, M. Ghommem, S. Meka, PyROM: A computational framework for reduced order modeling, J. Comput. Sci. 30 (2019) 157–173.
[29] M. Bergmann, C.-H. Bruneau, A. Iollo, Enablers for robust POD models, J. Comput. Phys. 228 (2) (2009) 516–538.
[30] M. Couplet, C. Basdevant, P. Sagaut, Calibrated reduced-order POD-Galerkin system for fluid flow modelling, J. Comput. Phys. 207 (1) (2005) 192–220.
[31] K. Kunisch, S. Volkwein, Galerkin proper orthogonal decomposition methods for parabolic problems, Numer. Math. 90 (1) (2001) 117–148.
[32] A. Kolmogoroff, Über die beste Annäherung von Funktionen einer gegebenen Funktionenklasse, Ann. of Math. 37 (1) (1936) 107–110.
[33] A. Pinkus, N-widths in Approximation Theory, Vol. 7, Springer-Verlag, Berlin, 1985.
[34] S.E. Ahmed, O. San, Breaking the Kolmogorov barrier in model reduction of fluid flows, Fluids 5 (1) (2020) 26.
[35] T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza, Model order reduction in fluid dynamics: challenges and perspectives, in: Reduced Order Methods for Modeling and Computational Reduction, Springer, 2014, pp. 235–273.
[36] D. Rempfer, On low-dimensional Galerkin models for fluid flow, Theor. Comput. Fluid Dyn. 14 (2) (2000) 75–88.
[37] B.R. Noack, K. Afanasiev, M. Morzynski, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, J. Fluid Mech. 497 (2003) 335–363.
[38] S.M. Rahman, S.E. Ahmed, O. San, A dynamic closure modeling framework for model order reduction of geophysical flows, Phys. Fluids 31 (4) (2019) 046602.
[39] S. Sirisup, G.E. Karniadakis, A spectral viscosity method for correcting the long-term behavior of POD models, J. Comput. Phys. 194 (1) (2004) 92–116.
[40] O. San, T. Iliescu, Proper orthogonal decomposition closure models for fluid flows: Burgers equation, Int. J. Numer. Anal. Model. B 5 (2014) 217–237.
[41] 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, Vol. 5, 2014.
[42] B. Protas, B.R. Noack, J. Östh, Optimal nonlinear eddy viscosity in Galerkin models of turbulent flows, J. Fluid Mech. 766 (2015) 337–367.
[43] L. Cordier, B.R. Noack, G. Tissot, G. Lehnasch, J. Delville, M. Balajewicz, G. Daviller, R.K. Niven, Identification strategies for model-based control, Exp. Fluids 54 (8) (2013) 1580.
[44] J. Östh, B.R. Noack, S. Krajnović, D. Barros, J. Borée, On the need for a nonlinear subscale turbulence term in POD models as exemplified for a high-Reynolds-number flow over an Ahmed body, J. Fluid Mech. 747 (2014) 518–544.
[45] M. Couplet, P. Sagaut, C. Basdevant, Intermodal energy transfers in a proper orthogonal decomposition–Galerkin representation of a turbulent separated flow, J. Fluid Mech. 491 (2003) 275–284.
[46] V.L. Kalb, A.E. Deane, An intrinsic stabilization scheme for proper orthogonal decomposition based low-dimensional models, Phys. Fluids 19 (5) (2007) 054106.
[47] 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, Internat. J. Numer. Methods Engrg. 83 (10) (2010) 1345–1375.
[48] X. Xie, M. Mohebujjaman, L.G. Rebholz, T. Iliescu, Data-driven filtered reduced order modeling of fluid flows, SIAM J. Sci. Comput. 40 (3) (2018) B834–B857.
[49] M. Mohebujjaman, L.G. Rebholz, T. Iliescu, Physically constrained data-driven correction for reduced-order modeling of fluid flows, Internat. J. Numer. Methods Fluids 89 (3) (2019) 103–122.
[50] Z. Wang, I. Akhtar, J. Borggaard, T. Iliescu, Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison, Comput. Methods Appl. Mech. Engrg. 237 (2012) 10–26.
[51] I. Akhtar, Z. Wang, J. Borggaard, T. Iliescu, A new closure strategy for proper orthogonal decomposition reduced-order models, J. Comput. Nonlinear Dyn. 7 (3) (2012) 034503.
[52] M. Balajewicz, E.H. Dowell, Stabilization of projection-based reduced order models of the Navier–Stokes, Nonlinear Dynam. 70 (2) (2012) 1619–1632.
[53] D. Amsallem, C. Farhat, Stabilization of projection-based reduced-order models, Internat. J. Numer. Methods Engrg. 91 (4) (2012) 358–377.
[54] O. San, T. Iliescu, A stabilized proper orthogonal decomposition reduced-order model for large scale quasigeostrophic ocean circulation, Adv. Comput. Math. 41 (5) (2015) 1289–1319.
[55] M. Gunzburger, T. Iliescu, M. Mohebujjaman, M. Schneier, An evolve-filter-relax stabilized reduced order stochastic collocation method for the time-dependent Navier–Stokes equations, SIAM/ASA J. Uncertain. Quantif. 7 (4) (2019) 1162–1184.
[56] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[57] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: Continual prediction with LSTM, Neural Comput. 12 (1999) 2451–2471.
[58] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. 52 (1) (2020) 477–508.
[59] J.N. Kutz, Deep learning in fluid dynamics, J. Fluid Mech. 814 (2017) 1–4.
[60] P.A. Durbin, Some recent developments in turbulence closure modeling, Annu. Rev. Fluid Mech. 50 (2018) 77–103.
[61] K. Duraisamy, G. Iaccarino, H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. 51 (2019) 357–377.
[62] J.C.B. Gamboa, Deep learning for time-series analysis, 2017, arXiv preprint arXiv:1701.01887.
[63] B. Lusch, J.N. Kutz, S.L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nature Commun. 9 (1) (2018) 4950.
[64] S.E. Otto, C.W. Rowley, Linearly recurrent autoencoder networks for learning dynamics, SIAM J. Appl. Dyn. Syst. 18 (1) (2019) 558–593.
[65] K. Lee, K.T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comput. Phys. (2019) 108973.
[66] J. Pathak, A. Wikner, R. Fussell, S. Chandra, B.R. Hunt, M. Girvan, E. Ott, Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model, Chaos 28 (4) (2018) 041101.
[67] K. Yeo, I. Melnyk, Deep learning algorithm for data-driven simulation of noisy dynamical system, J. Comput. Phys. 376 (2019) 1212–1231.
[68] H. Jaeger, H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science 304 (5667) (2004) 78–80.
[69] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[70] 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, Proc. R. Soc. A 474 (2213) (2018) 20170844.
[71] H. Sak, A. Senior, F. Beaufays, Long short-term memory recurrent neural network architectures for large scale acoustic modeling, in: Fifteenth Annual Conference of the International Speech Communication Association, 2014.
[72] Z.Y. Wan, P. Vlachas, P. Koumoutsakos, T. Sapsis, Data-assisted reduced-order modeling of extreme events in complex dynamical systems, PLOS ONE 13 (5) (2018) e0197704.
[73] S. Pawar, S.E. Ahmed, O. San, A. Rasheed, Data-driven recovery of hidden physics in reduced order modeling of fluid flows, Phys. Fluids 32 (3) (2020) 036602.

[74] L. Sirovich, Turbulence and the dynamics of coherent structures. I. Coherent structures, Quart. Appl. Math. 45 (3) (1987) 561–571.
[75] G. Berkooz, P. Holmes, J.L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annu. Rev. Fluid Mech. 25 (1) (1993) 539–575.
[76] A. Chatterjee, An introduction to the proper orthogonal decomposition, Current Sci. (2000) 808–817.
[77] M. Rathinam, L.R. Petzold, A new look at proper orthogonal decomposition, SIAM J. Numer. Anal. 41 (5) (2003) 1893–1925.
[78] L.N. Trefethen, D. Bau III, Numerical Linear Algebra, Vol. 50, SIAM, Philadelphia, 1997.
[79] D. Amsallem, C. Farhat, Interpolation method for adapting reduced-order models and application to aeroelasticity, AIAA J. 46 (7) (2008) 1803–1813.
[80] D. Amsallem, J. Cortial, K. Carlberg, C. Farhat, A method for interpolating on manifolds structural dynamics reduced-order models, Internat. J. Numer. Methods Engrg. 80 (9) (2009) 1241–1258.
[81] R. Zimmermann, B. Peherstorfer, K. Willcox, Geometric subspace updates with applications to online adaptive nonlinear model reduction, SIAM J. Matrix Anal. Appl. 39 (1) (2018) 234–261.
[82] R. Zimmermann, Manifold interpolation and model reduction, 2019, arXiv preprint arXiv:1902.06502.
[83] M. Oulghelou, C. Allery, A fast and robust sub-optimal control approach using reduced order model adaptation techniques, Appl. Math. Comput. 333 (2018) 416–434.
[84] M. Oulghelou, C. Allery, Non intrusive method for parametric model order reduction using a bi-calibrated interpolation on the Grassmann manifold, 2018, arXiv preprint arXiv:1901.03177.
[85] A. Edelman, T.A. Arias, S.T. Smith, The geometry of algorithms with orthogonality constraints, SIAM J. Matrix Anal. Appl. 20 (2) (1998) 303–353.
[86] P.-A. Absil, R. Mahony, R. Sepulchre, Riemannian geometry of Grassmann manifolds with a view on algorithmic computation, Acta Appl. Math. 80 (2) (2004) 199–220.
[87] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations, Internat. J. Numer. Methods Engrg. 86 (2) (2011) 155–181.
[88] K. Carlberg, M. Barone, H. Antil, Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction, J. Comput. Phys. 330 (2017) 693–734.
[89] Y. Choi, K. Carlberg, Space–time least-squares Petrov–Galerkin projection for nonlinear model reduction, SIAM J. Sci. Comput. 41 (1) (2019) A26–A58.
[90] 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, Comput. Methods Appl. Mech. Engrg. 255 (2013) 147–157.
[91] M. Gunzburger, Finite Element Methods for Viscous Incompressible Flows, in: Computer Science and Scientific Computing, Academic Press Inc., Boston, MA, 1989, p. xviii+269, A guide to theory, practice, and algorithms.
[92] D. Amsallem, Interpolation on Manifolds of CFD-Based Fluid and Finite Element-Based Structural Reduced-Order Models for On-Line Aeroelastic Predictions (Ph.D. thesis), Stanford University, 2010.
[93] R. Ştefănescu, A. Sandu, I.M. Navon, Comparison of POD reduced order strategies for the nonlinear 2D shallow water equations, Internat. J. Numer. Methods Fluids 76 (8) (2014) 497–521.
[94] M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera, An empirical interpolation method: Application to efficient reduced-basis discretization of partial differential equations, C. R. Math. 339 (2004) 667–672.
[95] S. Chaturantabut, D.C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764.
[96] S.E. Ahmed, S.M. Rahman, O. San, A. Rasheed, I.M. Navon, Memory embedded non-intrusive reduced order modeling of non-ergodic flows, Phys. Fluids 31 (12) (2019) 126602.
[97] M. Dihlmann, M. Drohmann, B. Haasdonk, Model reduction of parametrized evolution problems using the reduced basis method with adaptive time-partitioning, in: International Conference on Adaptive Modeling and Simulation, ADMOS 2011, 2011, pp. 156—167.
[98] R. Maulik, B. Lusch, P. Balaprakash, Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders, 2020, arXiv preprint arXiv:2002.00470.
[99] K. Lee, K.T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comput. Phys. 404 (2020) 108973.
[100] S. Grimberg, C. Farhat, N. Youkilis, On the stability of projection-based model order reduction for convection-dominated laminar and turbulent flows, 2020, arXiv preprint arXiv:2001.10110.
[101] J. Kani, A.H. Elsheikh, Reduced-order modeling of subsurface multi-phase flow models using deep residual recurrent neural networks, Transp. Porous Media 126 (3) (2019) 713–741.
[102] A.T. Mohan, D.V. Gaitonde, A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks, 2018, arXiv preprint arXiv:1804.09269.
[103] Z. Wang, D. Xiao, F. Fang, R. Govindan, C.C. Pain, Y. Guo, Model identification of reduced order fluid dynamics systems using deep learning, Internat. J. Numer. Methods Fluids 86 (4) (2018) 255–268.
[104] S.M. Rahman, S. Pawar, O. San, A. Rasheed, T. Iliescu, Nonintrusive reduced order modeling framework for quasigeostrophic turbulence, Phys. Rev. E 100 (2019) 053306.
[105] S. Wiewel, M. Becher, N. Thuerey, Latent space physics: Towards learning the temporal evolution of fluid flow, in: Computer Graphics Forum, Vol. 38, Wiley Online Library, 2019, pp. 71–82.
[106] D. Xiao, F. Fang, J. Zheng, C. Pain, I. Navon, Machine learning-based rapid response tools for regional air pollution modelling, Atmos. Environ. 199 (2019) 463–473.
[107] F. Chollet, et al., Keras, 2015, https://keras.io.
[108] M. Maleewong, S. Sirisup, On-line and off-line POD assisted projective integral for non-linear problems: A case study with Burgers' equation, Int. J. Math. Comput. Phys. Electr. Comput. Eng. 5 (7) (2011) 984–992.
[109] J.D. Buntine, D. Pullin, Merger and cancellation of strained vortices, J. Fluid Mech. 205 (1989) 263–295.
[110] J. Von Hardenberg, J. McWilliams, A. Provenzale, A. Shchepetkin, J. Weiss, Vortex merging in quasi-geostrophic flows, J. Fluid Mech. 412 (2000) 331–353.
[111] A. Arakawa, Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I, J. Comput. Phys. 1 (1) (1966) 119–143.
[112] S. Gottlieb, C.W. Shu, Total variation diminishing Runge-Kutta schemes, Math. Comp. 67 (221) (1998) 73–85.