SIAM J. SCI. COMPUT.
Vol. 45, No. 3, pp. B283–B313

PHYSICS GUIDED MACHINE LEARNING FOR VARIATIONAL MULTISCALE REDUCED ORDER MODELING*

SHADY E. AHMED†, OMER SAN†, ADIL RASHEED‡, TRAIAN ILIESCU§, AND ALESSANDRO VENEZIANI¶

Abstract. We propose a new physics guided machine learning (PGML) paradigm that leverages the variational multiscale (VMS) framework and available data to dramatically increase the accuracy of reduced order models (ROMs) at a modest computational cost. The hierarchical structure of the ROM basis and the VMS framework enable a natural separation of the resolved and unresolved ROM spatial scales. Modern PGML algorithms are used to construct novel models for the interaction among the resolved and unresolved ROM scales. Specifically, the new framework builds ROM operators that are closest to the true interaction terms in the VMS framework. Finally, machine learning is used to reduce the projection error and further increase the ROM accuracy. Our numerical experiments for a two-dimensional vorticity transport problem show that the novel PGML-VMS-ROM paradigm maintains the low computational cost of current ROMs while significantly increasing the ROM accuracy.

Key words. reduced order modeling, variational multiscale method, physics guided machine learning, nonlinear proper orthogonal decomposition, autoencoder, Galerkin projection

MSC codes. 37N10, 62M10, 62M20, 62M45, 65M06, 68T05, 76M30, 76F20

DOI. 10.1137/22M1496360

1. Introduction.

The behavior of physical systems can be generally described by physical principles (e.g., conservation of mass, momentum, and energy) together with constitutive laws. The resulting models are often mathematically formulated as partial differential equations (PDEs) (e.g., the Navier–Stokes equations). Solving them allows prediction and analysis of the system's dynamics. The applicability of analytical methods for solving PDEs is usually limited to simple cases with special geometry and under severe assumptions. In practice, numerical approaches (e.g., finite difference, finite volume, spectral, and finite element methods) are utilized to discretize the governing equations and approximate the values of the unknowns corresponding to a given grid. For turbulent flows, we need to deal with an exceedingly large number of degrees of freedom due to the existence of a wide range of spatio-temporal scales to be resolved. Although such models, called here *full order models* (FOMs), are capable of providing very accurate results, they can be computationally demanding. Therefore, FOMs become impractical for applications that require

* Submitted to the journal's Computational Methods in Science and Engineering section May 16, 2022; accepted for publication (in revised form) January 17, 2023; published electronically June 6, 2023.

https://doi.org/10.1137/22M1496360

Funding: This research was supported by National Science Foundation grants DMS-2012253 (fourth author), DMS-2012255 (second author), and DMS-2012286 (fifth author). The second author also gratefully acknowledges support through the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under award DE-SC0019290.

† School of Mechanical & Aerospace Engineering, Oklahoma State University, Stillwater, OK 74078 USA (shady.ahmed@okstate.edu, osan@okstate.edu).

‡ Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7465, Trondheim, Norway (adil.rasheed@ntnu.no).

§ Department of Mathematics, Virginia Tech, Blacksburg, VA 24061 USA (iliescu@vt.edu).

¶ Department of Mathematics, Department of Computer Science, Emory University, Atlanta, GA 30322 USA (avenez2@emory.edu).

multiple forward evaluations with varying inputs (e.g., flow control [42, 53, 64], optimization [9, 12, 16, 22, 46, 61, 89], and digital twinning [14, 30, 31, 44, 63, 82]) or studies requiring several simulations like computational-aided clinical trials [83].

Reduced order models (ROMs) are defined as computationally light surrogates that can mimic the behavior of FOMs with sufficient accuracy [2, 48, 65, 80, 81]. Projection-based ROMs have gained significant popularity in the past few decades due to the increased amounts of collected data (either from actual experiments or numerical simulations) as well as the development of system identification tools [17, 79]. Of particular interest is the combination of proper orthogonal decomposition (POD) and Galerkin projection, which has been a powerful driver for ROM progress [2]. The process comprises an *offline* stage and an *online* stage. The offline stage starts with the collection of data corresponding to system realizations (called *snapshots*) at different time instants and/or parameter values. With these data sets, POD provides a hierarchy of basis functions (or modes) that capture the maximum amount of the underlying system's energy (defined by the data variance). The offline stage is concluded by performing a Galerkin projection of the FOM operators onto the subspace spanned by a truncated set of POD modes to obtain a system of ordinary differential equations (ODEs) representing the Galerkin ROM (GROM). Although this offline stage can be extremely expensive, the resulting GROM can be utilized during the online deployment phase to efficiently predict the system's behavior at parameter values and/or time instants different from those in the data preparation process.

The GROM framework has been successful in many applications (e.g., [2, 7, 13, 27, 32, 41, 52, 66, 88]), especially those dominated by diffusion mechanisms or periodic dynamics. Those are often referred to as systems with a solution manifold that is characterized by a small Kolmogorov n-width [3, 60]. In the POD context, this means that the dynamics can be accurately represented by a few modes. However, for convection-dominated flows with strong nonlinearity, the Kolmogorov n-width is often large with a slow decay, which hinders the linear reducibility of the underlying system.

The repercussions of a Galerkin truncation and projection are twofold. First, the span of the retained POD basis functions does not necessarily provide an accurate representation of the solution, and it gives rise to the *projection error* [4, 8, 73]. Second, the interactions between the truncated and the retained modes can be significant. These interactions are ignored in the Galerkin projection step, and consequently the GROM cannot in general capture the dynamics of the resolved modes accurately. This introduces a *closure error* [1, 29, 40, 55, 62, 67, 68, 69, 71, 86, 87]. A variety of approaches have been proposed to quantify the error incurred by the approximate solution from different ROM techniques [24, 25, 56].

Several efforts have been devoted to address the closure problem. A recent survey covering a plethora of physics-based and data-driven ROM closure methodologies can be found in [2]. The closure problem has been historically related to the stabilization of the ROM solution, drawing roots from large eddy simulation (LES) studies, where the truncated small scales are thought of as having diffusive effects on the larger scales. Balajewicz and Dowell [10] proposed a subspace rotation technique where a new set of basis functions is constructed as a superposition of a larger set of POD basis functions (i.e., a mixture of large energetic and small dissipative modes). This technique has been applied to the incompressible [10] and compressible [11] Navier–Stokes equations.

Another approach to effectively accounting for the dissipative effects of the truncated low energy-containing modes is through the eddy viscosity-based frameworks [33]. Nonetheless, it was found that introducing eddy viscosity to *all* resolved scales

can unnecessarily contaminate the dynamics of the *largest* scales. To mitigate this problem, the *variational multiscale* (VMS) method, which was proposed by Hughes's group [35, 36, 37] in the finite element setting (see, e.g., [21, 43] for a survey), was utilized to add eddy viscosity dissipation to only a portion of the ROM resolved scales in [38, 39, 86]. A data-driven version of VMS (DD-VMS) has been recently proposed in [50], where the effects of the truncated modes on the GROM dynamics are not restricted to be diffusive.

In the present study, we transform the DD-VMS [45, 50] and provide an alternative modular framework by utilizing machine learning (ML) capabilities. We stress that this is a fundamental change in which the standard DD-VMS regression is replaced by ML in order to better account for closure effects. Therefore, the proposed neural network approach is essentially different from the regression-based DD-VMS [50]. In particular, the DD-VMS ansatz of a quadratic polynomial closure model is relieved by utilizing the deep neural network (DNN) functionality with memory embedding. We also leverage the long short-term memory (LSTM) variant of recurrent neural networks (RNNs) to approximate scale-aware closures. In essence, the use of LSTM encompasses a non-Markovian closure, supported by the Mori–Zwanzig formalism [18, 19, 20, 49, 90].

Moreover, we adopt the physics guided machine learning (PGML) framework introduced in [57, 58, 59] to reduce the uncertainty of the output results. In particular, we exploit concatenation layers informed by the VMS-ROM arguments to enrich the neural network architecture and constrain the learning algorithm to the manifold of physically consistent solutions. Finally, for problems with a large Kolmogorov n-width, we utilize the nonlinear POD (NLPOD) methodology [5] to reduce the projection error without affecting the computational efficiency, by learning the correlations among the small unresolved scales to provide far fewer latent space variables. We also perform a numerical investigation of the proposed strategies (ML-VMS-ROM, PGML-VMS-ROM, and PGML-VMS-NLPOD-ROM), with a particular focus on the locality of scale interactions, which is a cornerstone of the VMS framework.

The rest of the paper is organized as follows: We briefly describe the reduced order modeling methodology by the nexus of POD and Galerkin projection in section 2. The relevant background information and notation for the VMS approach are given in section 3. The use of the PGML methodology to provide reliable predictions is explained in section 4, while the NLPOD approach is discussed in section 5. The proposed PGML-VMS-NLPOD framework is tested for the parametric unsteady vortex-merger problem, which exemplifies convection-dominated flow systems. Results and discussions are presented in section 6, followed by the concluding remarks in section 7.

2. Reduced order modeling. A Newtonian incompressible fluid flow in a domain $\Omega \subset \mathbb{R}^d$, where d defines the spatial dimension (i.e., $d \in \{2, 3\}$), can be described by the Navier–Stokes equations (NSE). We note that the improper treatment of the pressure term has been shown to yield inaccuracies and instabilities in the resulting ROM [54]. In order to eliminate the pressure term, we consider the NSE in the vorticity-vector potential formulation. This formulation is widely popular for modeling vortex transport phenomena (e.g., wake modeling [15]). In particular, we consider the two-dimensional (2D) case where the vector potential is reduced to the streamfunction as follows:

$$\begin{aligned} \partial_t \omega - \nu \Delta \omega + (\boldsymbol{u} \cdot \nabla)\omega = 0 \qquad & \text{in } \Omega \times [0, T], \\ \Delta \psi + \omega = 0 \qquad & \text{in } \Omega \times [0, T], \end{aligned} \tag{2.1}$$

where $\omega(\boldsymbol{x},t)$ and $\psi(\boldsymbol{x},t)$ denote the vorticity and streamfunction fields, respectively, for $\boldsymbol{x}\in\Omega$ and $t\in[0,T]$, while ν stands for the kinematic viscosity (diffusion coefficient). In dimensionless form, ν represents the reciprocal of the Reynolds number, Re. The velocity vector field $\boldsymbol{u}(\boldsymbol{x},t)$ is related to the streamfunction as follows:

$$\boldsymbol{u}=\nabla^{\perp}\psi,\quad \nabla^{\perp}=[\partial_y,-\partial_x]^T. \tag{2.2}$$

By using (2.2), equation (2.1) can be further rewritten as follows:

$$\partial_t\omega-\nu\Delta\omega+J(\omega,\psi)=0\qquad \text{in } \Omega\times[0,T], \tag{2.3}$$

where $J(\cdot,\cdot)$ denotes the Jacobian operator, which is defined as follows:

$$J(\omega,\psi)=\frac{\partial\omega}{\partial x}\frac{\partial\psi}{\partial y}-\frac{\partial\omega}{\partial y}\frac{\partial\psi}{\partial x}. \tag{2.4}$$

The vorticity-streamfunction of the NSE, also known as the vorticity transport equation, inherently satisfies the incompressibility constraint. In addition, it mitigates the odd-even decoupling problem of the NSE when a collocated grid is used. Equation (2.3) is equipped with an initial condition and boundary conditions on $\Gamma:=\partial\Omega$. For convenience and simplicity of presentation, we shall assume the following conditions:

$$\begin{aligned} &IC:\ \omega(\boldsymbol{x},0)=\omega_0(\boldsymbol{x}) && \text{in } \Omega,\\ &BC:\ \psi(\boldsymbol{x},t)=0,\quad \frac{\partial\psi}{\partial\boldsymbol{n}}=0 && \text{in } \Gamma\times[0,T]. \end{aligned} \tag{2.5}$$

In the remainder of this section, we describe the construction of the projection-based ROM of the vorticity transport equation. This includes the use of POD to approximate the solution (subsection 2.1), followed by the Galerkin method, where the FOM operators in (2.1) are projected onto the POD subspace to define the GROM (subsection 2.2).

2.1. Proper orthogonal decomposition. We consider a collection of system realizations defined by an ensemble of vorticity fields $\{\omega(\boldsymbol{x},t_0),\omega(\boldsymbol{x},t_1),\dots,\omega(\boldsymbol{x},t_{M-1})\}$. These are often called *snapshots* and come from either experimental measurements or numerical simulations of (2.1) or (2.3) using any of the standard discretization schemes (e.g., finite element, finite difference, or finite volume methods). Without loss of generality, we assume that these snapshots are sampled at equidistant M (>1) time instants with $t_m=m\Delta t$, where $m=0,1,\dots,M-1$ and $\Delta t=\frac{T}{M-1}$. We note that, in general, these snapshots can correspond to different types of parameters (e.g., operating conditions, physical properties, and geometry).

In POD, we seek a low-dimensional basis $\{\phi_1,\phi_2,\dots,\phi_R\}$ that optimally approximates the space spanned by the snapshots in the following sense [33]:

$$\begin{aligned} &\min\ \left\langle\left\|\omega(\cdot,\cdot)-\sum_{k=1}^{R}\big(\omega(\cdot,\cdot),\phi_k(\cdot)\big)\phi_k(\cdot)\right\|^2\right\rangle,\\ &\text{subject to}\qquad \|\phi\|=1,\qquad \big(\phi_i(\cdot),\phi_j(\cdot)\big)=\delta_{ij}, \end{aligned} \tag{2.6}$$

where $\langle\cdot\rangle$ denotes an average operation with respect to the parametrization, $(\cdot,\cdot)$ is an inner product, and $\|\cdot\|$ is the corresponding norm. For example, an ensemble average based on temporal snapshots can be defined as follows:

$$\langle\omega\rangle=\frac{1}{M}\sum_{m=0}^{M-1}\omega(\cdot,t_m). \tag{2.7}$$

The snapshots represent the approximation of the quantity of interest on a specific grid. For example, a realization of the vorticity field at a given time can be arranged in a column vector $\boldsymbol{\omega} \in \mathbb{R}^N$, where N is the number of grid points. It can be shown that solving the optimization problem (2.6) amounts to solving the following eigenvalue problem [84]:

$$\mathbf{D}\Phi = \Phi\Lambda, \tag{2.8}$$

where the entries of the diagonal matrix Λ and the columns of Φ represent the eigenpairs of the spatial autocorrelation matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ with entries defined as

$$\big[\mathbf{D}\big]_{ij} = \Big\langle \boldsymbol{\omega}(\boldsymbol{x}_i, \cdot)\boldsymbol{\omega}(\boldsymbol{x}_j, \cdot) \Big\rangle, \tag{2.9}$$

where $\boldsymbol{\omega}(\boldsymbol{x}_i, \cdot)$ is the ith entry of $\boldsymbol{\omega}$. For fluid flow problems, the length of the vector $\boldsymbol{\omega}$ is often large, which makes the eigenvalue problem in (2.8) computationally challenging.

Sirovich [74, 75, 76] proposed a numerical procedure, known as the *method of snapshots*, to reduce the computational cost of solving (2.8). This approach is efficient, especially when the number of collected snapshots M is much smaller than the number of degrees of freedom (i.e., $M \ll N$), as it reduces the $N \times N$ eigenvalue problem in (2.8) to an $M \times M$ problem. The spatial autocorrelation matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ is replaced by the temporal snapshot correlation matrix $\mathbf{K} \in \mathbb{R}^{M\times M}$ with entries defined as follows:

$$\big[\mathbf{K}\big]_{ij} = \frac{1}{M}\Big(\omega(\cdot, t_i), \omega(\cdot, t_j)\Big). \tag{2.10}$$

The following eigenvalue problem is thus considered:

$$\mathbf{K}\mathbf{v}_k = \lambda_k \mathbf{v}_k, \tag{2.11}$$

where $\mathbf{v}_k$ is the kth eigenvector of $\mathbf{K}$ and λ_k is the associated eigenvalue. To obtain the hierarchy of the POD basis, the eigenpairs are sorted in descending order by their eigenvalues (i.e., $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_M \geq 0$). Finally, the POD basis functions can be computed as a linear superposition of the collected snapshots as follows [84]:

$$\phi_k(\cdot) = \frac{1}{\sqrt{\lambda_k}} \sum_{m=0}^{M-1} [\mathbf{v}_k]_m \omega(\cdot, t_m), \tag{2.12}$$

where $[\mathbf{v}_k]_m$ denotes the mth component of $\mathbf{v}_k$. It can be verified that the basis functions in (2.12) are orthonormal (i.e., $(\phi_i(\cdot), \phi_j(\cdot)) = \delta_{ij}$), where δ_{ij} is the Kronecker delta.

The POD eigenvalues define the contribution of each mode toward the total variance in the given snapshots. In practice, the selection of the number of POD modes is often informed by an analysis of the eigenvalue spectrum. In this regard, a metric that evaluates the quality of a given set of retained modes in representing the system is the relative information content (RIC) [2], defined as follows:

$$\mathrm{RIC}(k) = \frac{\sum_{l=1}^{k} \lambda_l}{\sum_{l=1}^{M} \lambda_l}, \tag{2.13}$$

where k is the POD index at which modal truncation takes place. The selection of truncation level (i.e., value of R) is defined in such a way that the corresponding RIC

value is within an acceptable range (e.g., larger than 90%). We also note that the same POD algorithm and analysis can be applied considering parameters other than time. In this case, the temporal correlation matrix is substituted by a generalized parameter correlation matrix.

2.2. Galerkin projection. The GROM starts by the Galerkin truncation step, making use of the optimality criterion in (2.6) as follows:

$$\omega(\boldsymbol{x}, t_m) \approx \omega_R(\boldsymbol{x}, t_m) = \sum_{k=1}^{R} a_k(t_m)\phi_k(\boldsymbol{x}), \tag{2.14}$$

where $\{a_k\}_{k=1}^{R}$ are the time-varying modal coefficients (weights), known as *generalized coordinates*. The optimal values of these coefficients are defined by the true projection of the FOM trajectory onto the corresponding POD basis function as follows:

$$a_k(t_m) = \big(\omega(\cdot, t_m), \phi_k(\cdot)\big). \tag{2.15}$$

Since the vorticity and streamfunction are related to each other by the kinematic relationship $\Delta\psi = -\omega$ (see (2.1)), the basis functions $(\theta_k(x, y))$ for the streamfunction can be obtained from those of the vorticity as follows:

$$\nabla^2\theta_k(\boldsymbol{x}) = -\phi_k(\boldsymbol{x}), \quad k = 1, 2, \ldots, R. \tag{2.16}$$

Moreover, the reduced order approximations of the vorticity and streamfunction can share the same temporal coefficients $a_k(t)$,

$$\psi(\boldsymbol{x}, t_m) \approx \psi_R(\boldsymbol{x}, t_m) = \sum_{k=1}^{R} a_k(t_m)\theta_k(\boldsymbol{x}). \tag{2.17}$$

We note that the resulting set of streamfunction basis functions from (2.16) are not necessarily orthonormal.

Next, the vorticity ω and streamfunction ψ fields in (2.3) are replaced by their approximation ω_R and ψ_R from (2.14) and (2.17) as follows:

$$\begin{aligned} \partial_t \left[\sum_{k=1}^{R} a_k(t)\phi_k(\boldsymbol{x})\right] - \nu\Delta \left[\sum_{k=1}^{R} a_k(t)\phi_k(\boldsymbol{x})\right] \\ + J\left(\left[\sum_{k=1}^{R} a_k(t)\phi_k(\boldsymbol{x})\right], \left[\sum_{k=1}^{R} a_k(t)\theta_k(\boldsymbol{x})\right]\right) = 0. \end{aligned} \tag{2.18}$$

Since the Laplacian Δ and Jacobian $J(\cdot, \cdot)$ are spatial operators, and the summation and differentiation operations commute, (2.18) can be rewritten as follows (with a slight change of indices):

$$\sum_{i=1}^{R} \dot{a}_i(t)\phi_i(\boldsymbol{x}) - \nu\sum_{i=1}^{R} a_i(t)\Delta\phi_i(\boldsymbol{x}) + \sum_{i=1}^{R}\sum_{j=1}^{R} a_i(t)a_j(t)J\big(\phi_i(\boldsymbol{x}), \theta_j(\boldsymbol{x})\big) = 0. \tag{2.19}$$

Now, the Galerkin projection step comes into play by defining the POD test subspace $\boldsymbol{X}_R$ as follows:

$$\boldsymbol{X}_R := \operatorname{span}\{\phi_1, \phi_2, \ldots, \phi_R\}. \tag{2.20}$$

Then, (2.3) with ω and ψ replaced by ω_R and ψ_R, respectively, is projected onto the POD space $\boldsymbol{X}_R$. This yields the GROM of the vorticity transport equation: Find $\omega_R \in \boldsymbol{X}_R$ such that

$$(\partial_t \omega_R, \phi) - \nu(\Delta\omega_R, \phi) + \big(J(\omega_R, \psi_R), \phi\big) = 0 \qquad \forall \phi \in \boldsymbol{X}_R. \tag{2.21}$$

Next, taking the inner product of (2.19) with an arbitrary basis function ϕ_k yields the following (we drop the independent variable for clarity):

$$\underbrace{\sum_{i=1}^{R} \dot{a}_i \Big(\phi_i, \phi_k\Big)}_{\dot{a}_k} \underbrace{-\nu \sum_{i=1}^{R} a_i \Big(\Delta\phi_i, \phi_k\Big)}_{\text{linear term}} + \underbrace{\sum_{i=1}^{R}\sum_{j=1}^{R} a_i a_j \Big(J\big(\phi_i, \theta_j\big), \phi_k\Big)}_{\text{nonlinear term}} = 0. \tag{2.22}$$

Note that the first term reduces $\dot{a}_k$ thanks to the orthonormality property of the POD basis functions. Equation (2.22) can be rearranged as follows:

$$\dot{a}_k = \nu \sum_{i=1}^{R} A_{k,i} a_i + \sum_{i=1}^{R}\sum_{j=1}^{R} B_{k,i,j} a_i a_j, \tag{2.23}$$

which can be written in the so-called tensorial form

$$\dot{\boldsymbol{a}} = A\boldsymbol{a} + \boldsymbol{a}^\top B \boldsymbol{a}, \tag{2.24}$$

where $\boldsymbol{a}(t) \in \mathbb{R}^R$ is the vector of unknown coefficients $\{a_k\}_{k=1}^R$, while $A \in \mathbb{R}^{R\times R}$ and $B \in \mathbb{R}^{R\times R\times R}$ are the matrix and tensor corresponding to the linear and nonlinear terms, respectively. It should be noted that the tensorial form eliminates the dependence of GROM on N (the number of FOM degrees of freedom). Due to the quadratic nonlinearity, the cost of solving (2.24) scales with R^3. Other techniques for projection-based ROMs often employ hyper-reduction techniques to enable quick simulations. For further discussion on benchmarking tensorial ROM and hyper-reduction methods for various problems with different complexities, we refer the interested reader to [23, 77].

The Galerkin truncation step restricts the approximation of the vorticity field to live in a low-rank subspace $\boldsymbol{X}_R$ ($R \ll N$), which might not capture all the relevant flow structures. Therefore, a projection error is introduced. Furthermore, the Galerkin projection step enforces the dynamics of the ROM to be defined using only the scales supported by $\boldsymbol{X}_R$. Nonetheless, due to the coupling between different modes, the unresolved scales (i.e., the scales modeled by $\{\phi_k\}_{k\geq R+1}$) influence the dynamics of the resolved scales (i.e., the scales modeled by $\{\phi_k\}_{k\leq R}$). By neglecting these mutual interactions, the GROM becomes incapable of accurately describing the dynamics of the retained modes, which is usually referred to as the *closure* problem [2].

The projection error and closure error are illustrated in Figure 1 for a toy system whose full-rank linear expansion can be represented with 3 modes as follows:

$$\omega(x,t) = a_1(t)\phi_1(x) + a_2(t)\phi_2(x) + a_3(t)\phi_3(x). \tag{2.25}$$

Assuming that the FOM is written in the form

$$\dot{\omega} = F(\omega), \tag{2.26}$$

the dynamics of $\{a_k\}_{k=1}^3$ can be described as $\dot{a}_k = (F(\omega), \phi_k)$. Thus, the FOM trajectory can be written as follows:

$$\begin{bmatrix} \dot{a}_1 \\ \dot{a}_2 \\ \dot{a}_3 \end{bmatrix} = \begin{bmatrix} f_1(a_1, a_2, a_3) \\ f_2(a_1, a_2, a_3) \\ f_3(a_1, a_2, a_3) \end{bmatrix}. \tag{2.27}$$

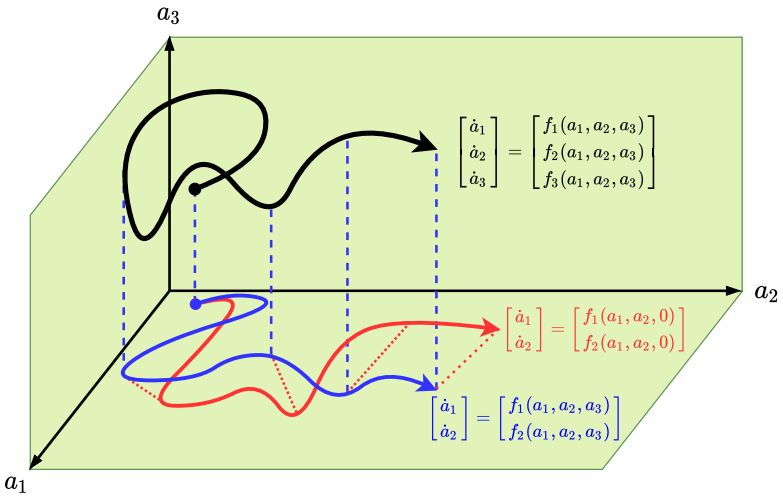

FIG. 1. *Representation of the repercussions of modal truncation onto the ROM solution. The solid black curve denotes the FOM trajectory, assuming that the full rank expansion is defined by a_1, a_2, and a_3. The solid blue curve defines the projection of the FOM trajectory onto a 2D subspace. The vertical dashed blue lines refer to the* projection *or* representation error*. Note that evaluating a_1 and a_2 still requires the knowledge of the FOM trajectory (i.e., a_1, a_2, and a_3) at every point. In practice, we only have information regarding the resolved variables (i.e., a_1 and a_2), so the contribution of a_3 towards the dynamics of a_1 and a_2 is neglected. This yields a* closure error*, denoted by the dashed red lines.*

In other words, evolving $\{a_k\}_{k=1}^3$ using (2.27) and reconstructing ω with (2.25) recovers the FOM field (equivalent to solving (2.26) using standard discretization schemes). For the sake of demonstration, we suppose that we retain only 2 modes in the ROM approximation. This corresponds to removing the third row in (2.27) as follows:

$$\begin{bmatrix}\dot{a}_1\\ \dot{a}_2\end{bmatrix} = \begin{bmatrix}f_1(a_1,a_2,a_3)\\ f_2(a_1,a_2,a_3)\end{bmatrix}. \tag{2.28}$$

Approximating ω with just two modes results in losing the flow structures that are contained in the truncated mode (the vertical direction in Figure 1), which yields the projection error. Furthermore, we note that f_1 and f_2 are usually functions of a_1, a_2, and a_3 for systems with strong nonlinearity and coupling between different modes. However, during ROM deployment, we do not usually have information regarding the unresolved dynamics (a_3 in this example). Thus, in GROM, the effects of the truncated scales on the resolved scales are assumed to be negligible, as follows:

$$\begin{bmatrix}\dot{a}_1\\ \dot{a}_2\end{bmatrix} = \begin{bmatrix}f_1(a_1,a_2,0)\\ f_2(a_1,a_2,0)\end{bmatrix}. \tag{2.29}$$

We denote the reference trajectory described by (2.28) as the true projection, which is related to (2.15). This defines the best low-rank approximation that can be obtained for a given number of modes, assuming we have access to the whole set of FOM scales. The difference between the GROM trajectory (corresponding to solving (2.29)) and the true projection trajectory represents the closure error. In the present study, we address both the closure error and the projection error. First, to tackle the closure problem, we leverage the VMS framework outlined in section 3 to develop the PGML methodology in section 4. Then, we utilize the NLPOD approach in section 5 to

reduce the projection error by learning a compressed latent space that encapsulates some of the truncated flow structures.

3. Variational multiscale method. The variational multiscale (VMS) methods are general numerical discretizations that significantly increase the accuracy of classical Galerkin approximations in under-resolved simulations, e.g., on coarse meshes or when not enough basis functions are available. The VMS framework, which was proposed by Hughes and coworkers [35, 36, 37], has made a profound impact in many areas of computational mechanics (see, e.g., [21, 43] for a survey).

To illustrate the standard VMS methodology, we consider a general nonlinear PDE as follows:

$$\dot{\omega} = F(\omega), \tag{3.1}$$

whose weak (variational) form is

$$(\dot{\omega}, \phi) = (F(\omega), \phi) \quad \forall \phi \in \boldsymbol{X}, \tag{3.2}$$

where F is a general nonlinear function and $\boldsymbol{X}$ is an appropriate test space. To build the VMS framework, we start with a sequence of hierarchical spaces of increasing resolutions: $\boldsymbol{X}_1$, $\boldsymbol{X}_1 \oplus \boldsymbol{X}_2$, $\boldsymbol{X}_1 \oplus \boldsymbol{X}_2 \oplus \boldsymbol{X}_3$, ... Next, we project system (3.1) onto each of the spaces $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, $\boldsymbol{X}_3$, ..., which yields a separate equation for each space. From a computational efficiency point of view, the goal is to solve for the ω component that lives in the coarsest space (i.e., $\boldsymbol{X}_1$), since this yields the lowest-dimensional system:

$$(\dot{\omega}, \phi) = (F(\omega), \phi) \quad \forall \phi \in \boldsymbol{X}_1. \tag{3.3}$$

However, system (3.3) is *not* closed since its right-hand side involves ω components that do not belong to $\boldsymbol{X}_1$ (i.e., $\omega_2 \in \boldsymbol{X}_2$, $\omega_3 \in \boldsymbol{X}_3$, ...):

$$(F(\omega), \phi) = (F(\omega_1, \omega_2, \omega_3, \dots), \phi), \quad \forall \phi \in \boldsymbol{X}_1. \tag{3.4}$$

Thus, the VMS closure problem needs to be solved. That is, (3.4) needs to be replaced with an equation that involves only terms that belong to $\boldsymbol{X}_1$. In general, the VMS system in (3.3) equipped with an appropriate closure model (i.e., a model with components in $\boldsymbol{X}_1$ that captures the interaction between ω_1 and the scales in $\boldsymbol{X}_2, \boldsymbol{X}_3, \dots$) yields an accurate approximation of the $\boldsymbol{X}_1$ component of ω.

The POD procedure in subsection 2.1 yields a hierarchy of orthogonal basis functions, sorted by their contribution to the total energy. Therefore, it provides a natural fit to the VMS framework. Next, we illustrate the adoption of VMS in GROM settings to define a multilevel VMS ROM. In particular, we detail the two-scale and the three-scale VMS ROMs, while further extensions become straightforward.

3.1. Two-scale VMS ROM. The two-scale VMS (VMS-2) ROM utilizes two orthogonal spaces, $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, defined as follows:

$$\begin{aligned} \boldsymbol{X}_1 &:= \operatorname{span}\{\phi_1, \phi_2, \dots, \phi_R\}, \\ \boldsymbol{X}_2 &:= \operatorname{span}\{\phi_{R+1}, \phi_{R+2}, \dots, \phi_N\}, \end{aligned} \tag{3.5}$$

where $\boldsymbol{X}_1$ represents the span of the resolved ROM scales and $\boldsymbol{X}_2$ is the span of the unresolved scales. Thus, ω can be written as follows:

$$\omega = \sum_{k=1}^{R} a_k \phi_k + \sum_{k=R+1}^{N} a_k \phi_k = \underbrace{\omega_R}_{\text{resolved}} + \underbrace{\omega'}_{\text{unresolved}}, \tag{3.6}$$

where $\omega_R \in \boldsymbol{X}_1$ is the resolved ROM component of ω, while $\omega' \in \boldsymbol{X}_2$ is the unresolved component of ω. Using this decomposition, (3.3) can be rewritten as follows:

$$\big(\dot{\omega}_R, \phi_k\big) = \big(F(\omega_R), \phi_k\big) + \underbrace{\Big[\big(F(\omega), \phi_k\big) - \big(F(\omega_R), \phi_k\big)\Big]}_{\text{VMS-2 closure term}} \quad \forall k \in \{1, \dots, R\}. \tag{3.7}$$

The bracketed term in (3.7) is the VMS-2 closure term, which models the interaction between the ROM modes and the discarded modes. Since the unresolved component of ω, ω', is not available during the online deployment stage, it is not possible to exactly compute the closure term in practical settings. Instead, the closure term can be approximated using a generic function $G(\omega_R)$ as follows:

$$\big(G(\omega_R), \phi_k\big) \approx \big(F(\omega), \phi_k\big) - \big(F(\omega_R), \phi_k\big), \tag{3.8}$$

and the VMS-2 ROM can be written as

$$\big(\dot{\omega}_R, \phi_k\big) = \big(F(\omega_R), \phi_k\big) + \big(G(\omega_R), \phi_k\big). \tag{3.9}$$

The form and parameters of G will be defined in section 4.

3.2. Three-scale VMS ROM. The *locality of modal interactions* is a cornerstone of the VMS framework. It states that neighboring modes have more mutual interactions than those who are far apart in the energy spectrum. For this reason, it is natural to distinguish between neighboring and far modes when closure modeling is performed. To this end, the flexibility of the hierarchical structure of the ROM space is leveraged to perform a three-scale decomposition of ω, leading to a three-scale VMS (VMS-3) ROM, which aims at increasing the VMS-2 ROM accuracy.

To construct the VMS-3 ROM, we first build three orthogonal spaces, $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, and $\boldsymbol{X}_3$, as follows:

$$\begin{aligned} \boldsymbol{X}_1 &:= \operatorname{span}\{\phi_1, \phi_2, \dots, \phi_r\}, \\ \boldsymbol{X}_2 &:= \operatorname{span}\{\phi_{r+1}, \phi_{r+2}, \dots, \phi_R\}, \\ \boldsymbol{X}_3 &:= \operatorname{span}\{\phi_{R+1}, \phi_{R+2}, \dots, \phi_N\}. \end{aligned} \tag{3.10}$$

Compared to the decomposition into resolved and unresolved scales in subsection 3.1, $\boldsymbol{X}_1$ now represents the *large resolved* ROM scales, $\boldsymbol{X}_2$ represents the *small resolved* ROM scales, and $\boldsymbol{X}_3$ denotes the unresolved ROM scales. With these definitions, ω can be written as follows:

$$\begin{aligned} \omega &= \sum_{k=1}^{r} a_k \phi_k + \sum_{k=r+1}^{R} a_k \phi_k + \sum_{k=R+1}^{N} a_k \phi_k \\ &= \underbrace{\omega_L}_{\text{large resolved}} + \underbrace{\omega_S}_{\text{small resolved}} + \underbrace{\omega'}_{\text{unresolved}}. \end{aligned} \tag{3.11}$$

This is similar to (3.6) with $\omega_R = \omega_L + \omega_S$. To construct the VMS-3 ROM, we project system (3.1) onto each of the spaces $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, as follows:

$$\big(\dot{\omega}_L, \phi_k\big) = \big(F(\omega_L + \omega_S), \phi_k\big) + \Big[\big(F(\omega), \phi_k\big) - \big(F(\omega_L + \omega_S), \phi_k\big)\Big], \ k = 1, \dots, r, \tag{3.12}$$

$$\big(\dot{\omega}_S, \phi_k\big) = \big(F(\omega_L + \omega_S), \phi_k\big) + \Big[\big(F(\omega), \phi_k\big) - \big(F(\omega_L + \omega_S), \phi_k\big)\Big], \ k = r+1, \dots, R. \tag{3.13}$$

Although the two bracketed terms in (3.12) and (3.13) defining the VMS-3 closure terms look similar, they have different roles. To understand this, let us consider the different types of modal interactions involved in these equations. For example, the (low-index: $k = 1, \dots, r$) large scales (ω_L) interact with the (medium-index: $k = r+1, \dots, R$) small resolved scales (ω_S) and the (high-index: $k = R+1, \dots$) unresolved scales (ω'). However, the interactions between the resolved large scales and the unresolved scales are assumed to be negligible as compared to those between the resolved large scales and the resolved small scales (according to the VMS principle of locality of modal interactions). Therefore, the bracketed term in (3.12) (for $k = 1, \dots, r$) basically models the contribution of the interactions between the large resolved scales and the small resolved modes, neglecting the contribution from the higher index truncated modes. On the other hand, the bracketed term in (3.13) (for $k = r+1, \dots, R$) models the interaction between the (medium-index) small resolved and the (high-index) unresolved ROM modes. This allows great flexibility in choosing the structure of the different VMS ROM closure terms. This concept is investigated numerically in section 6.

4. Physics guided machine learning. In this section, the VMS-2 and VMS-3 closure terms defined in section 3 are approximated using only the information in the resolved scales. Specifically, we utilize a purely data-driven approach to compute the parameters of the closure models. However, instead of relying on heuristics or ad hoc arguments to define the specific structure of the closure model (as in the standard DD-VMS [50]), we exploit the capabilities of a deep neural network (DNN) in approximating arbitrary functions. In particular, we use the long short-term memory (LSTM) variant of recurrent neural networks (RNNs), which has shown substantial success in data-driven modeling of time series [26, 34, 72]. We emphasize that, to mitigate well-known drawbacks of data-driven modeling (e.g., sensitivity to noise in input data), the VMS ROM framework utilizes data to model only the VMS ROM closure operators, but all the other ROM operators are built by using classical Galerkin projection. Thus, our VMS ROM framework incorporates "data-driven closure" rather than "data-driven modeling" for the resolved scales.

4.1. ML-VMS ROM. The VMS-2 ROM in (3.9) can be rewritten as follows:

$$\dot{\boldsymbol{a}} = \boldsymbol{f}(\boldsymbol{a}) + \boldsymbol{c}(\boldsymbol{a}), \tag{4.1}$$

where $\boldsymbol{a} = [a_1, a_2, \dots, a_R]^T \in \mathbb{R}^R$ is the vector of coefficients for the resolved POD modes, $\boldsymbol{f}(\boldsymbol{a}) = [(F(\omega_R), \phi_1), (F(\omega_R), \phi_2), \dots, (F(\omega_R), \phi_R)]$ represents the Galerkin projection of the FOM operators onto the POD subspace, and $\boldsymbol{c}(\boldsymbol{a}) = [c_1, c_2, \dots, c_R]^R \in \mathbb{R}^R$ is the vector of the closure (correction) terms, i.e., $c_k = (G(\omega_R), \phi_k)$. In the present study, we use DNN to represent the closure model, i.e., $\boldsymbol{c}(\cdot) \approx \pi_\theta(\boldsymbol{a})$, where θ denotes the parameterization of the LSTM. The general functional form of the DNN models used for temporal forecasting can be written as follows:

$$\begin{aligned} \mathbf{h}^{(n)} &= f_h^h(\boldsymbol{a}^{(n)}, \mathbf{h}^{(n-1)}), \\ \boldsymbol{c}^{(n)} &= f_h^o(\mathbf{h}^{(n)}), \end{aligned} \tag{4.2}$$

where $\boldsymbol{a}^{(n)} := \boldsymbol{a}(t_n) \in \mathbb{R}^R$ is the vector of modal coefficients at time t_n and $\boldsymbol{c}^{(n)} \in \mathbb{R}^R$ is the corresponding closure term, defining the input and output of the DNN, respectively. In (4.2), $\mathbf{h} \in \mathbb{R}^H$ represents the hidden-state of the neural network, f_h^h and f_h^o the hidden-to-hidden and hidden-to-output mappings, respectively, and H the dimension of the hidden state.

The Mori–Zwanzig formulation [28, 55, 78, 85, 91] shows that non-Markovian terms are required to account for the effects of the unresolved scales on the resolved scales. Thus, the closure operators are modeled as functions of the time history of the resolved scales. We emphasize that employing a non-Markovian closure model is a key feature of the proposed PGML-VMS-ROM that is in stark contrast with the DD-VMS in [45, 50], which considers only the Markovian effects.

For memory embedding, we let $\boldsymbol{c}$ be a function of the short time history of the resolved POD coefficients, i.e., $\boldsymbol{c}^{(n)}(\cdot) \approx \pi_\theta(\boldsymbol{a}^{(n)}, \boldsymbol{a}^{(n-1)}, \ldots, \boldsymbol{a}^{(n-\tau)}) = \pi_\theta(\boldsymbol{a}^{(n):(n-\tau)})$, where τ defines the length of the time history of $\boldsymbol{a}$ that is required for estimating the closure term. The LSTM allows modeling non-Markovian processes while mitigating the issue with vanishing (or exploding) gradient by employing gating mechanisms. In particular, the hidden-to-hidden mapping f_h^h is defined using the following equations:

$$
\begin{aligned}
\mathbf{g}_f^{(n)} &= \sigma_f(\mathbf{W}_f[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_f),\\
\mathbf{g}_i^{(n)} &= \sigma_i(\mathbf{W}_i[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_i),\\
\tilde{\mathbf{s}}^{(n)} &= \tanh\left(\mathbf{W}_s[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_s\right),\\
\mathbf{s}^{(n)} &= \mathbf{g}_f^{(n)} \odot \mathbf{s}^{(n-1)} + \mathbf{g}_i^{(n)} \odot \tilde{\mathbf{s}}^{(n)},\\
\mathbf{g}_o^{(n)} &= \sigma_o(\mathbf{W}_o[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_o),\\
\mathbf{h}^{(n)} &= \mathbf{g}_o^{(n)} \odot \tanh\left(\mathbf{s}^{(n)}\right),
\end{aligned}
\tag{4.3}
$$

where $\mathbf{g}_f, \mathbf{g}_i, \mathbf{g}_o \in \mathbb{R}^H$ are the forget gate, input gate, and output gate, respectively, with the corresponding $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o \in \mathbb{R}^{H\times(H+R)}$ weight matrices, and $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o \in \mathbb{R}^H$ bias vectors. $\mathbf{s} \in \mathbb{R}^H$ is the cell state with a weight matrix $\mathbf{W}_s \in \mathbb{R}^{H\times(H+R)}$ and bias vector $\mathbf{b}_s \in \mathbb{R}^H$. Finally, σ is the sigmoid activation function, and $\odot$ denotes the elementwise multiplication.

We stack l LSTM layers to define the hidden states, followed by a fully connected layer with a linear activation function to represent the hidden-to-output mapping. Thus, the ML-VMS-2 closure model can be written as

$$
\boldsymbol{c}^{(n)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \mathbf{h}_{l-1}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}),
\tag{4.4}
$$

where $\mathcal{L}(\cdot)$ represents the output layer with linear activation, and $\mathcal{I}(\cdot)$ denotes the input layer. Note that each of the internal LSTM layers ($i = 1, 2, \ldots, l-1$) produces a sequence of hidden states $\mathbf{h}_i^{(n):(n-\tau)}$, while the the lth layer passes only the hidden state at the final time $\mathbf{h}_l^{(n)}$ to the output layer. A mean squared error loss function can be defined to evaluate the performance of LSTM as follows:

$$
loss = \frac{1}{N_{batch}} \sum_{n=1}^{N_{batch}} \sum_{k=1}^{R} \left\| \left(F(\omega^{(n)}), \phi_k\right) - \left(F(\omega_R^{(n)}), \phi_k\right) - \boldsymbol{c}_k^{(n)} \right\|_2^2,
\tag{4.5}
$$

and a minimization algorithm is used to optimize the weight and biases described in (4.3).

In order to make use of the locality of modal interactions, the VMS-3 ROM is written as

$$
\begin{bmatrix} \dot{\boldsymbol{a}_L} \\ \dot{\boldsymbol{a}_S} \end{bmatrix} = \boldsymbol{f}(\boldsymbol{a}) + \begin{bmatrix} \boldsymbol{c}_L(\boldsymbol{a}) \\ \boldsymbol{c}_S(\boldsymbol{a}) \end{bmatrix},
\tag{4.6}
$$

where two separate terms are dedicated to model the closure for the resolved large scales and resolved small scales. For the ML-VMS-3, the closure terms are defined as follows:

$$
\begin{aligned}
\boldsymbol{c}_L^{(n)} &\approx \pi_{L,\theta}(\boldsymbol{a}^{(n):(n-\tau)}) \\
&\approx \mathcal{L}_{_L}(\cdot) \circ \mathbf{h}_{l_L}^{(n)}(\cdot) \circ \mathbf{h}_{l-1_L}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_{1_L}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}), \\
\boldsymbol{c}_S^{(n)} &\approx \pi_{S,\theta}(\boldsymbol{a}^{(n):(n-\tau)}) \\
&\approx \mathcal{L}_{_S}(\cdot) \circ \mathbf{h}_{l_S}^{(n)}(\cdot) \circ \mathbf{h}_{l-1_S}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_{1_S}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}).
\end{aligned} \tag{4.7}
$$

Since we are using two separate LSTM neural networks to model $\boldsymbol{c}_L$ and $\boldsymbol{c}_S$, the corresponding loss functions are defined as follows:

$$
\begin{aligned}
loss_L &= \frac{1}{N_{batch}} \sum_{n=1}^{N_{batch}} \sum_{k=1}^{r} \left\| \big(F(\omega), \phi_k\big) - \big(F(\omega_L + \omega_S), \phi_k\big) - \boldsymbol{c}_{L,k}^{(n)} \right\|_2^2, \\
loss_S &= \frac{1}{N_{batch}} \sum_{n=1}^{N_{batch}} \sum_{k=r+1}^{R} \left\| \big(F(\omega), \phi_k\big) - \big(F(\omega_L + \omega_S), \phi_k\big) - \boldsymbol{c}_{S,k}^{(n)} \right\|_2^2.
\end{aligned} \tag{4.8}
$$

We note that we have more flexibility in ML-VMS-3 than in ML-VMS-2. Hence, it is possible to make richer descriptions of the interactions between large resolved, small resolved, and unresolved scales.

4.2. PGML-VMS ROM. Critical aspects that should be considered during the adoption of ML-based approaches include their reliability, robustness, and trustworthiness. Previous studies [57, 58, 59] have reported high levels of uncertainty in the predictions of vanilla-type ML methods, especially for sparse data and incomplete governing equations regimes. In order to mitigate this issue, we utilize the physics-guided machine learning (PGML) paradigm to incorporate known physical arguments and constraints into the learning process. In particular, we exploit a modular approach to modify the neural network architectures through layer concatenation to inject physical information at different points in the latent space of the given DNN. This adaptation improves the performance during both the training and the deployment phases and results in significant reduction in the uncertainty levels of the model prediction, as we demonstrate in section 6.

In the PGML framework, the features extracted from the physics-based model are embedded into the generic ith intermediate hidden layer along with the latent variables. In order to build the PGML-VMS framework, we consider the Galerkin projection of the governing equations onto different POD modes to define the physics-based features (since they are derived from physical principles). Thus, the PGML-VMS-2 closure model can be written as

$$
\begin{aligned}
\boldsymbol{c}^{(n)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_i^{(n):(n-\tau)}(\cdot), \boldsymbol{f}^{(n):(n-\tau)} \right) \circ \mathbf{h}_{i-1}^{(n):(n-\tau)}(\cdot) \\
\circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}),
\end{aligned} \tag{4.9}
$$

where $\mathcal{C}(\cdot,\cdot)$ represents the concatenation operation, and $\boldsymbol{f}^{(n):(n-\tau)}$ is the time history of projecting the FOM operators onto the truncated POD subspace. We highlight that there is no significant computational load for the calculation of $\boldsymbol{f} := A\boldsymbol{a} + \boldsymbol{a}^\top B\boldsymbol{a}$, since A and B are already precomputed.

A schematic illustration of the PGML adaptation of the standard LSTM architecture is depicted in Figure 2. In this figure, 3 LSTM layers are used (i.e., $l = 3$), followed by a dense layer to provide the mapping from hidden state to the closure terms. The physics-based features are injected into the LSTM latent space after two

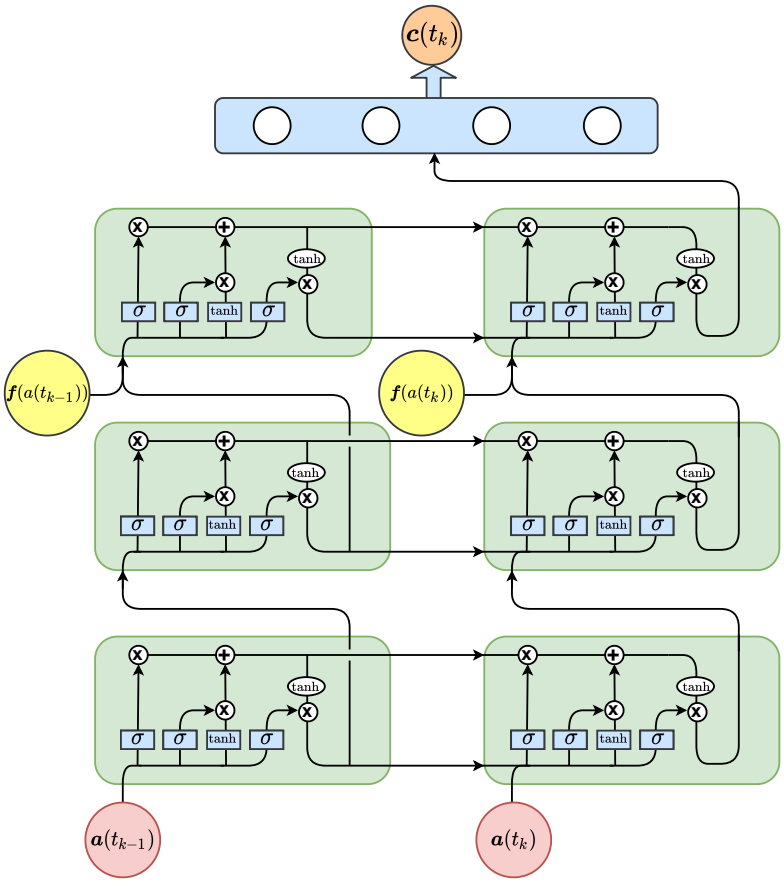

FIG. 2. *Illustration of the PGML methodology with concatenated LSTM layers. In this figure, a time history of* 2 *time-steps is used while physics-based features (yellow circles in the figure) are injected into the LSTM latent space after the second hidden layer* ($i = 2$).

hidden layers. One of the main advantages of the novel PGML framework in Figure 2 is its modularity and simplicity. For example, based on the level of fidelity and our confidence in the injected features, we can promptly change the layer at which we embed them.

Finally, the PGML-VMS-3 closure models can be written as

$$
\begin{aligned}
\boldsymbol{c}_L^{(n)} \approx \mathcal{L}_{_L}(\cdot) \circ \mathbf{h}_{l_L}^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_{i_L}^{(n):(n-\tau)}(\cdot), \boldsymbol{f}_{_L}^{(n):(n-\tau)}\right) \circ \mathbf{h}_{i-1_L}^{(n):(n-\tau)}(\cdot) \\
\circ \cdots \circ \mathbf{h}_{1_L}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}), \\
\boldsymbol{c}_S^{(n)} \approx \mathcal{L}_{_S}(\cdot) \circ \mathbf{h}_{l_S}^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_{i_S}^{(n):(n-\tau)}(\cdot), \boldsymbol{f}_{_S}^{(n):(n-\tau)}\right) \circ \mathbf{h}_{i-1_S}^{(n):(n-\tau)}(\cdot) \\
\circ \cdots \circ \mathbf{h}_{1_S}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}).
\end{aligned}
\tag{4.10}
$$

Note that in (4.10), we enjoy higher flexibility in choosing the physics-based features injected for each of the large- and small-scale closure models. For instance, in the present study, we benefit from the locality of modal interactions by embedding the

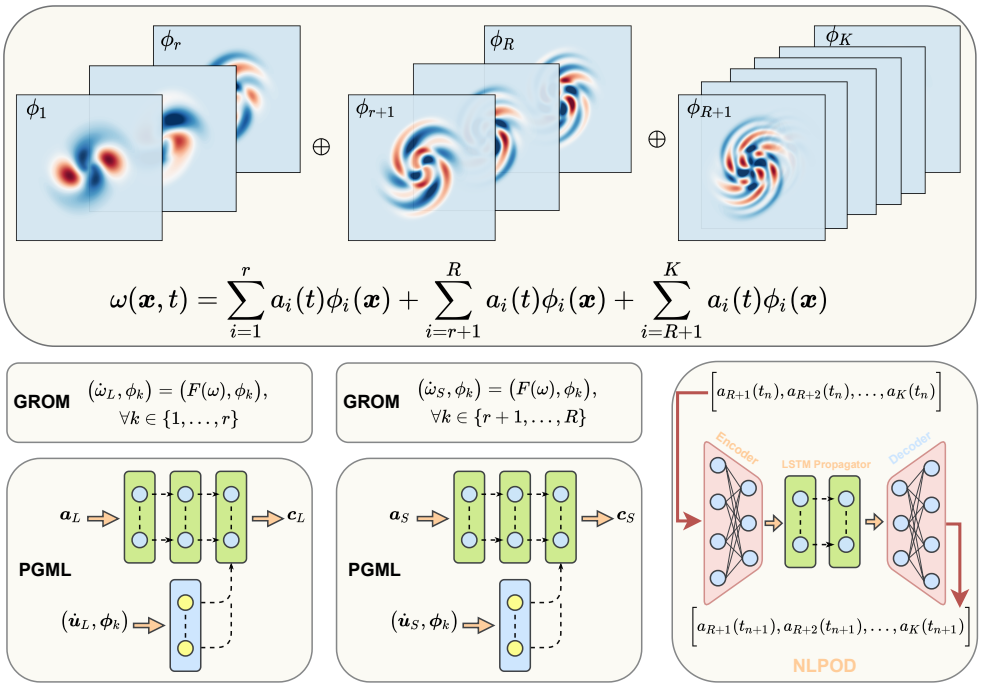

FIG. 3. *Schematic representation of the PGML-VMS-3 model for the large and small resolved scales, combined with NLPOD for enhanced field reconstruction. We note that PGML-VMS-3 is built upon a GROM for the first R modes and mitigates the closure error (i.e., the effect of the truncated scales on the resolved scales). In a complementary fashion, NLPOD implements an equation-free model for the truncated scales to reduce the projection error (i.e., the effect of the truncated scales on the flow field reconstruction).*

Galerkin propagator of only a few relevant neighboring modes (i.e., $\boldsymbol{f}_L$ and $\boldsymbol{f}_S$ in (4.10)), rather than including all of them in the LSTM learning (i.e., $\boldsymbol{f}$ in (4.9)).

5. Nonlinear POD. In sections 3 and 4, we have addressed the closure problem. That is, we corrected the GROM dynamics by approximating the effects of modal interactions between resolved and unresolved scales to accurately predict the dynamics of the retained ROM modes. However, the reconstructed flow field is still restricted to the span of the first R POD basis functions, as given in (2.14). Nonetheless, for convection-dominated flows, the important flow structures generally span a large number of modes. Thus, truncating the solution beyond a small number of modes results in a large projection error. In other words, the component $\omega' = \sum_{k=R+1}^{N} a_k \phi_k$ that cannot be approximated by the resolved POD basis becomes significant.

In this section, we adapt the nonlinear POD (NLPOD) framework, introduced in [5], to model the unresolved part of the field. Figure 3 presents a schematic representation of the PGML-VMS-3 model for the large and small resolved scales combined with NLPOD for enhanced field reconstruction. Note that, although both the PGML-VMS-3 and the NLPOD aim at increasing the ROM accuracy, they target different error sources: the PGML-VMS-3 aims at mitigating the closure error, whereas the NLPOD aims at alleviating the projection error.

The NLPOD methodology is based on combining POD with autoencoder (AE) techniques from ML to learn a latent representation of the POD expansion. It lever-

ages the predefined hierarchy of POD basis functions, which satisfy the conservation laws and physical constraints, together with the capabilities of DNN to reveal the nonlinear correlations between the modes. Rather than using the NLPOD for the compression of the total set of POD coefficients, we constrain it to learn a few latent variables, which represent only the unresolved scales. To construct the NLPOD, we first define $\boldsymbol{b} = \{a_k\}_{k=R+1}^{K}$ corresponding to an almost full-rank POD expansion, where $K \leq N$ can be defined using the RIC spectrum (e.g., $\mathrm{RIC}(K) \geq 99.99\%$). The goal is utilize the nonlinear mapping capabilities of DNNs to learn a compressed representation of those $K - R$ POD coefficients. In what follows, $\boldsymbol{z} = \{z_k\}_{k=1}^{q}$ refers to the learned compression, where q denotes the dimension of the AE latent space and $q \ll K$ so that an auto-regressive model can be constructed efficiently to evolve $\boldsymbol{z}$ in time.

The AE starts with an encoding process that involves applying a series of nonlinear mappings onto the input data to shrink the dimensionality down to a bottleneck layer representing the low rank or latent space embedding. An inverse mapping from the latent space variables to the same input is performed by another set of nonlinear mappings, defining the decoding part. For the NLPOD, the encoder and decoder can be represented as follows:

$$\text{Encoder } \eta : \boldsymbol{b} \in \mathbb{R}^{K-R} \mapsto \boldsymbol{z} \in \mathbb{R}^{q}, \qquad \text{Decoder } \zeta : \boldsymbol{z} \in \mathbb{R}^{q} \mapsto \boldsymbol{b} \in \mathbb{R}^{K-R}, \tag{5.1}$$

and they are trained jointly to minimize the following objective function:

$$\mathcal{J} = \sum_{n=1}^{N_{train}} \|\boldsymbol{b}^{(n)} - (\eta \circ \zeta)(\boldsymbol{b}^{(n)})\|, \tag{5.2}$$

where N_{train} is the number of training samples.

In order to temporally propagate $\boldsymbol{z}$, we can use any of the regression tools, including sparse regression, Gaussian process regression, Seq2seq algorithms, temporal fusion transformers, and auto-regression methods. In the present study, we use LSTM architectures that are similar to the ones used in section 4 to learn the one time-step mapping from $\boldsymbol{z}^{(n)}$ to $\boldsymbol{z}^{(n+1)}$, as follows:

$$\boldsymbol{z}^{(n+1)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \mathbf{h}_{l-1}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{z}^{(n):(n-\tau)}). \tag{5.3}$$

Note that the number of layers, l, and the length of time history, τ, are not necessarily equal to those in section 4. Moreover, the LSTM and AE can be trained either jointly or separately. In the present study, we train them separately for the sake of simplicity and to facilitate the NLPOD combination with other time series prediction tools.

Before moving to the numerical experiments, we summarize the proposed framework, which has two main objectives: (1) to reduce the closure error, which rises due to the difference between $(F(\omega), \phi_k)$ and $(F(\omega_R), \phi_k)$; and (2) to reduce the projection error, which rises due to the difference between ω and ω_R.

1. For the first objective (the closure error), we use ML-based models to learn the correction term (i.e., $(F(\omega), \phi_k) - (F(\omega_R), \phi_k)$) as a function of the coefficients of the POD expansion. The training data for input are (ω, ϕ_k), and those for the output are $(F(\omega), \phi_k) - (F(\omega_R), \phi_k)$.
   Recently, there has been an increase in discourse regarding the trustworthiness of ML-based approaches in physics-based computations, and it is commonly accepted that incorporating physical knowledge into the ML models is essential to ensure

their reliability. For example, the governing equations can be embedded by adding physics-based penalty terms to the data-based loss function, and certain symmetries and invariances can be enforced through the neural network by customized architectures. The current work explores another approach (i.e., PGML) that we believe is simpler, flexible, and effective, especially when the known physics is incomplete. In particular, we identify certain features and information from physics and perform feature engineering to enrich the neural network with this information. We refer to the GROM as a physics-based model since it employs an orthogonal projection of the PDE operators onto the POD basis functions. Thus, the resulting GROM ($\dot{\boldsymbol{a}} = A\boldsymbol{a} + \boldsymbol{a}^\top B\boldsymbol{a}$) inherits the underlying dynamics from the FOM and has a polynomial structure. The linear and quadratic terms correspond to the dissipation and convective terms in the NSE, respectively. We take advantage of this physics-based ROM and use the projected propagator ($A\boldsymbol{a} + \boldsymbol{a}^\top B\boldsymbol{a}$) as an additional feature in the PGML-based closure. In addition, we find that adding these features at intermediate layers (rather than the first layer) of the neural network (similar to skip-connection architectures) yields improved results. For training the PGML closure, we now have both (ω, ϕ_k) and $(F(\omega_R), \phi_k)$ as inputs. However, the data-based part (i.e., (ω, ϕ_k)) is fed to the first layer while the the physics-guided part $(F(\omega_R), \phi_k)$ is skip-connected to an intermediate layer. The output of the neural network is the correction term $(F(\omega), \phi_k) - (F(\omega_R), \phi_k)$.

2. For the second objective (the projection error), we learn the POD expansion coefficients in $\omega_K = \sum_{i=1}^{K} a_i\phi_i$ for the terms beyond $i = R$ (where $K > R$). When $K \gg R$, building an autoregressive model to evolve $a_i(t)_{i=R+1}^{K}$ to $a_i(t + \Delta t)_{i=R+1}^{K}$ is found to be fragile in practice, and the training process becomes cumbersome. Therefore, we first use an autoencoder model to learn a compressed representation $z_i{}_{1=1}^{q}$ for $a_i{}_{i=R+1}^{K}$, where $q \ll K$. Then, we build an LSTM-based autoregressive model to evolve $z_i(t)_{1=1}^{q}$ to $z_i(t + \Delta t)_{1=1}^{q}$, where the decoder part can be finally used to recover $a_i(t + \Delta t)_{i=R+1}^{K}$ from $z_i(t + \Delta t)_{1=1}^{q}$.

Finally, it is worth noting that our proposed framework maintains the Galerkin POD at its heart and constructs nonlinear ML-based corrections to separately address the associated closure and representation errors. This is in contrast to other methodologies that learn a nonlinear low-dimensional map altogether (e.g., [47]). Our choice has been motivated by the following:

- It is more feasible to perform rigorous analysis of the accuracy, consistency, convergence, and stability of linear space methods (e.g., Galerkin methods) than nonlinear space techniques (e.g., ML approaches). Since our framework is based on POD and Galerkin projection in its core, it is still possible to apply similar mathematical tools to analyze it (see [45] for a first step in this direction).
- Galerkin POD models have been widely accepted in industry, which makes the introduction of correction techniques more appealing than replacing existing methodologies altogether. This is particularly important given the rapid advancement in ML algorithms as it would be impractical to keep replacing the whole framework to adopt a new algorithm. Furthermore, our framework is quite modular in the sense that the core Galerkin POD model is fixed while the complimentary ML techniques can be easily replaced and/or combined with other tools. For instance, a transformer can be used as a drop-in replacement for the LSTM without changing its POD core.
- The VMS framework has strong support from computational mechanics studies, and the present study is only a small step towards leveraging the richness of

VMS algorithms to boost ROM developments. In particular, the VMS framework equips the ROM practitioner with a high level of flexibility in adopting different correction schemes to address various error sources.

6. Results and discussion. In this section, we perform a numerical investigation of the proposed PGML-VMS-ROM methodologies (with and without the NLPOD extension) using the 2D vortex merger problem [70], governed by the following vorticity transport equation:

$$\partial_t \omega + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \Delta \omega \qquad \text{in } \Omega \times [0, T]. \tag{6.1}$$

We consider a spatial domain of dimensions $(2\pi \times 2\pi)$ with periodic boundary conditions. The flow is initialized with a pair of corotating Gaussian vortices with equal strengths centered at $(x_1, y_1) = (5\pi/4, \pi)$ and $(x_2, y_2) = (3\pi/4, \pi)$ as follows:

$$\omega(x, y, 0) = \exp\left(-\rho\left[(x - x_1)^2 + (y - y_1)^2\right]\right) + \exp\left(-\rho\left[(x - x_2)^2 + (y - y_2)^2\right]\right), \tag{6.2}$$

where ρ is a parameter that controls the mutual interactions between the two vortical motions, set at $\rho = \pi$ in the present study. For the FOM simulations, we consider a regular Cartesian grid resolution of 256×256 (i.e., $\Delta x = \Delta y = 2\pi/256$), with a time-step size of 0.001. Vorticity snapshots are collected every 100 time-steps for $t \in [0, 30]$, totaling 300 snapshots. The evolution of the vortex merger problem at selected values of the Reynolds number is depicted in Figure 4, which illustrates the convective and interactive mechanisms affecting the transport and development of the two vortices.

In terms of POD analysis, we use $R = 6$, which captures more than 90% of the total variance in the snapshot data, to define the total number of resolved scales. For the three-scale VMS investigation, we split the resolved modes into 2 resolved large scales (i.e., $r = 2$) and 4 resolved small scales. For the NLPOD study, we find that $K = 20$ corresponds to near full-rank approximation of the flow field at all values of the Reynolds number. This is illustrated by the plot of the RIC values as a function of the number of POD modes at $\mathrm{Re} = 3000$ in Figure 5.

Following a systematic approach, in subsection 6.1, we first present our computational results for ML-VMS-2 and PGML-VMS-2 to quantitatively demonstrate the benefit of incorporating the physics guided machine learning approach. We then present the results for PGML-VMS-3 to highlight the flexibility and accuracy gain of the three-scale approach. Finally, in subsection 6.2, we reveal the additional role of the NLPOD approach by illustrating the performance of the PGML-VMS-3+NLPOD approach. The codes to reproduce the results in this section are publicly available as a GitHub repository [6].

6.1. Multilevel VMS closure for resolved scales. We store data corresponding to $\mathrm{Re} \in \{500, 750, 1000, \ldots, 3000\}$ (in increments of 250), but we use only the data collected at $\mathrm{Re} \in \{500, 750, 1000\}$ for neural network training, while the remaining data are reserved for testing purposes. For each value of the Reynolds number, we store 300 snapshots, which results in 900 samples for the offline training phase. In addition, 20% of these samples (randomly selected) are excluded from the training for validation and comparison of neural network architecture designs (e.g., number of layers and LSTM cells). For the ML-VMS frameworks, we use two LSTM layers with a hidden state ($\mathbf{h}$) dimensionality of 20 and hyperbolic tangent activation. For the PGML-VMS cases, we add an extra LSTM layer (i.e., a total of 3 layers) and

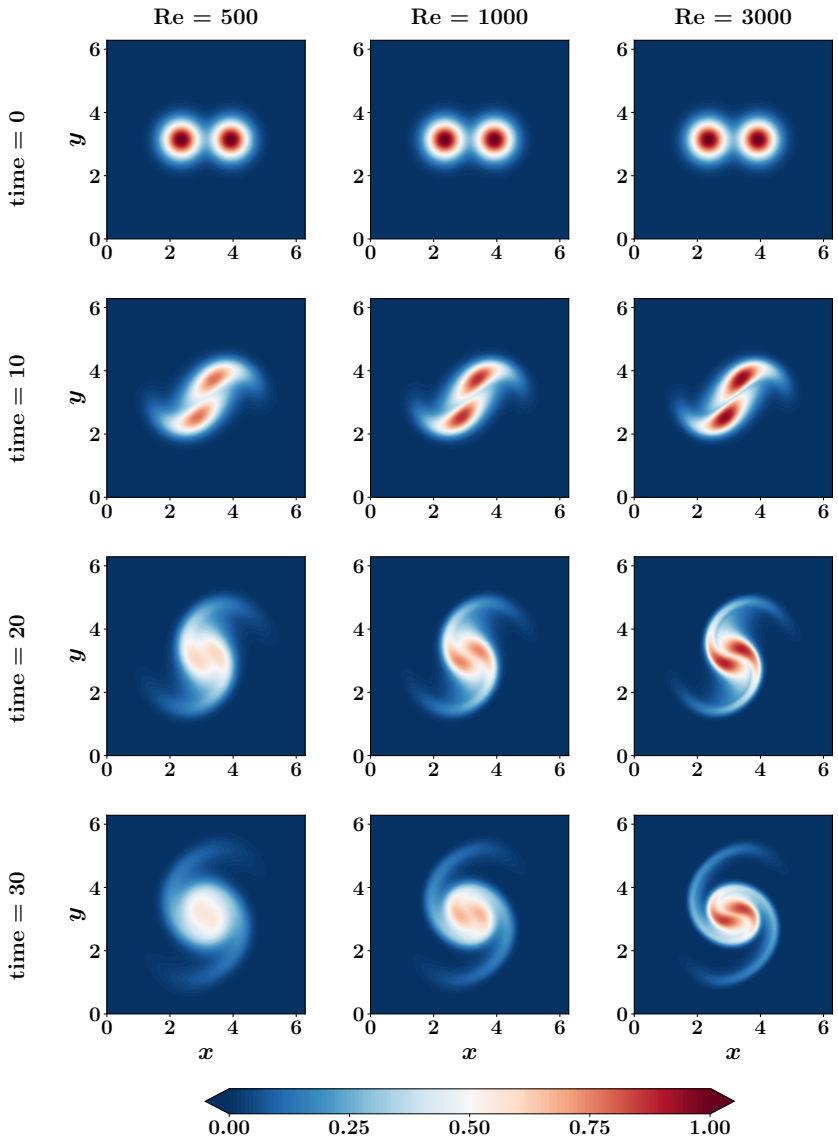

FIG. 4. *Samples of temporal snapshots of the vorticity field for the vortex merger problem at different values of the Reynolds number.*

the physics-based features are passed to the first hidden layer. The Adam optimizer with default settings (e.g., learning rate $= 10^{-3}$) and batch size of 32 is used for the training.

First, we explore the combination of multilevel variational multiscale methods with machine learning. Figure 6 displays the results of applying the ML-VMS-2 framework to model the closure term at Re $= 3000$. In particular, we run a group of 10 LSTMs with different initializations of the neural network weights and utilize the deep ensemble method to quantify the uncertainty in the predictions. On the average, the ML-VMS-2 method provides accurate results compared to the GROM results. However, the uncertainty levels, described by the standard deviation in the

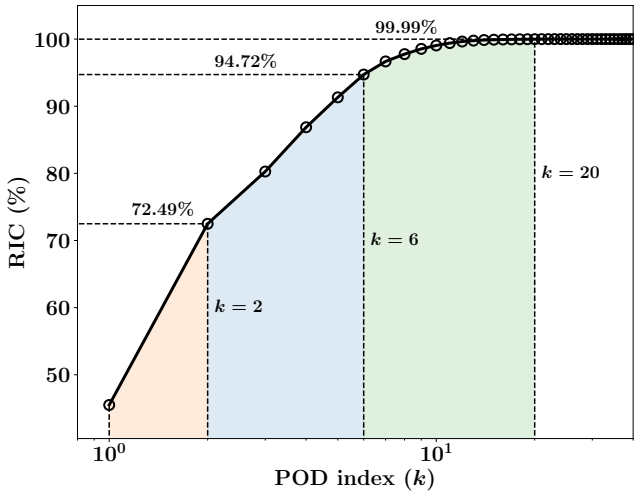

FIG. 5. *RIC values as a function of the modal truncation for the vortex merger problem at* $Re = 3000$.

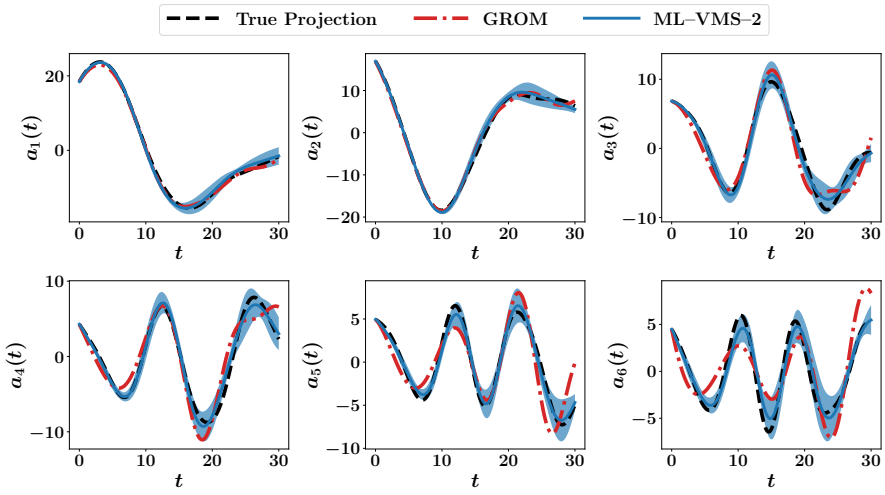

FIG. 6. *The time evolution of the first* 6 *modes of the vortex merger problem for* $Re = 3000$ *with the two-level VMS using ML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values* (μ) *from an ensemble of* 10 *different LSTM neural networks trained with different weight initializations, while the shaded area defines the uncertainty bounds using standard deviation* (σ) *values. For better visualization, the shaded band is plotted with* $\mu \pm 5\sigma$.

ensemble predictions, are high. This is especially evident at the late time instants as the uncertainty propagates and grows with time.

In order to increase the closure model robustness and reduce the uncertainty levels, we apply the PGML to inject physics-based features, as detailed in section 4. Figure 7 shows the evolution of the first 6 POD modal coefficients using the PGML-VMS-2. We can observe a significant reduction in the uncertainty levels as depicted by the shaded area, compared to the ML-VMS-2. It is also clear that the GROM yields inaccurate predictions. Moreover, we can observe that the deviations of the

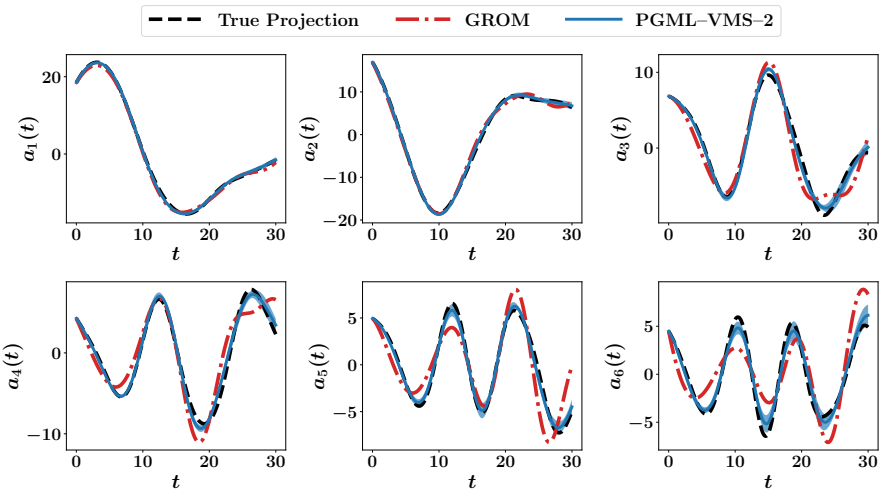

FIG. 7. *The time evolution of the first* 6 *modes of the vortex merger problem for* $Re = 3000$ *with the two-level VMS using PGML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values* (μ) *from an ensemble of* 10 *different LSTM neural networks trained with different weight initializations, while the shaded area defines the uncertainty bounds using standard deviation* (σ) *values. For better visualization, the shaded band is plotted with* $\mu \pm 5\sigma$.

GROM trajectory from the true projections are larger for the high-index portion of the resolved modes. In fact, this observation also applies to the ML-VMS-2 and PGML-VMS-2, which provide better results for the first two or three modes than the remaining ones.

In Figure 8, we plot the ROM propagator $\dot{\boldsymbol{a}}$ computed by the Galerkin method (i.e., with truncation, with no access to the unresolved scales, and without correction) against the true propagator (assuming access to all the flow scales). We find that the GROM equations can adequately describe the dynamics of the first modes, but fail to do so for the last modes. This can be explained by locality of information transfer, which is one of the main concepts used in the VMS development. Such locality indicates that the neighboring modes exhibit larger mutual interactions than the modes that are far apart. Thus, describing the dynamics of the leading modes requires more information from the first few scales than from the remaining scales. In other words, the resolved scales become almost sufficient to define the propagator of the leading modes. On the other hand, the last modes are adjacent to the unresolved scales. Thus, the mode truncation considerably affects the dynamics of the last modes.

In order to improve the quality of the closure model, we leverage the locality of modal interactions and apply the three-level VMS closure to correct the ROM dynamics. The selection of r (i.e., the index at which the resolved scales are divided into resolved large and resolved small scales) is still an open research question for VMS-ROMs (and for the VMS framework, in general). Proof-of-concept studies show that even an arbitrary splitting yields more accurate results. A ROM practitioner can also follow an energy-based criterion for choosing r. For instance, by considering the POD eigenvalue distribution, the spectrum can be divided into parts with distinct decay rates (slopes). Furthermore, different definitions of ROM length-scales [51] can be also considered to select r, R, and K (see Figure 3). In the present study, we split the resolved scales into two parts: the first 2 modes represent the largest

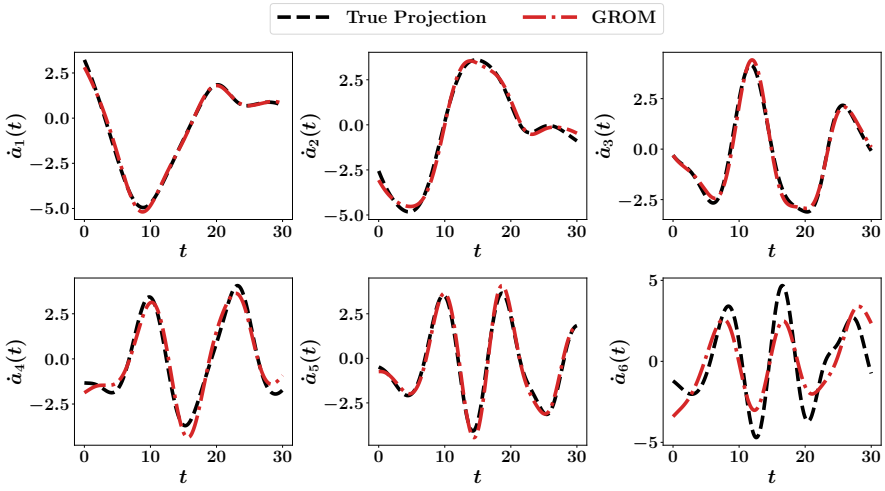

FIG. 8. *Comparison between the ROM propagator computed by Galerkin projection (with truncation, i.e.,* $\dot{a_k} = (-J(\omega_R, \psi_R) + \nabla^2\omega_R, \phi_k)$*, against the true (FOM projection) propagator (i.e.,* $\dot{a_k} = (-J(\omega, \psi) + \nabla^2\omega, \phi_k)$ *at* $Re = 3000$ *and for* $R = 6$*. We notice that the Galerkin projection accurately captures the dynamics of the first modes, but a discrepancy appears at the higher-index modes, which motivates the use of multilevel VMS closure.*

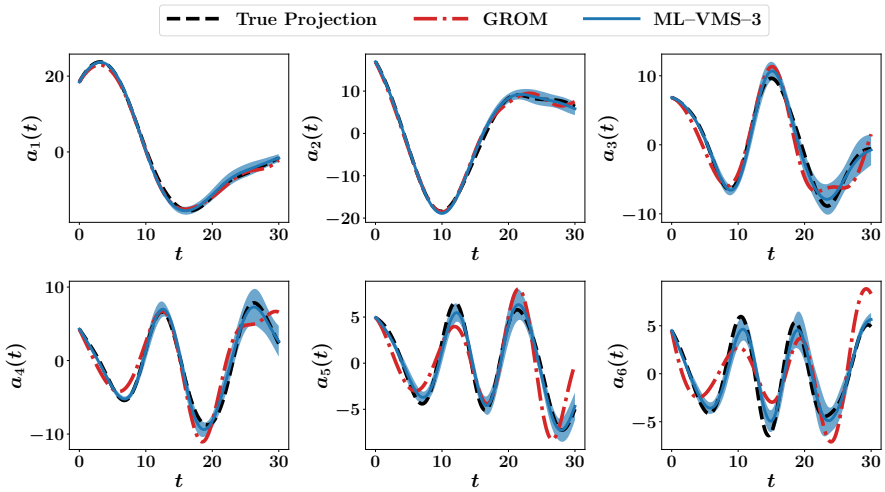

FIG. 9. *The time evolution of the first* 6 *modes of the vortex merger problem for* $Re = 3000$ *with the three-level VMS using ML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values* (μ) *from an ensemble of* 10 *different LSTM neural networks trained with different weight initializations, while the shaded area defines the uncertainty bounds using standard deviation* (σ) *values. For better visualization, the shaded band is plotted with* $\mu \pm 5\sigma$*.*

resolved scales, while the remaining 4 modes represent the small resolved scales. The ML-VMS-3 predictions of the temporal dynamics for the first 6 modes are shown in Figure 9. Compared to Figure 6, the ML-VMS-3 provides more accurate results than the ML-VMS-2, also in terms of uncertainty levels.

Finally, the PGML-VMS-3 results are shown in Figure 10, where we can see improved results across all the resolved scales with very low levels of uncertainty.

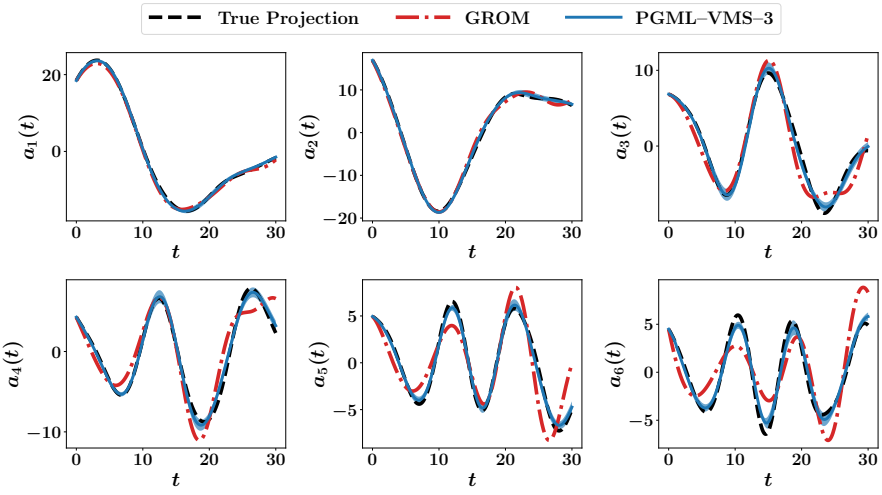

FIG. 10. *The time evolution of the first* 6 *modes of the vortex merger problem for* $Re = 3000$ *with the three-level VMS using PGML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values* (μ) *from an ensemble of* 10 *different LSTM neural networks trained with different weight initializations, while the shaded area defines the uncertainty bounds using standard deviation* (σ) *values. For better visualization, the shaded band is plotted with* $\mu \pm 5\sigma$.

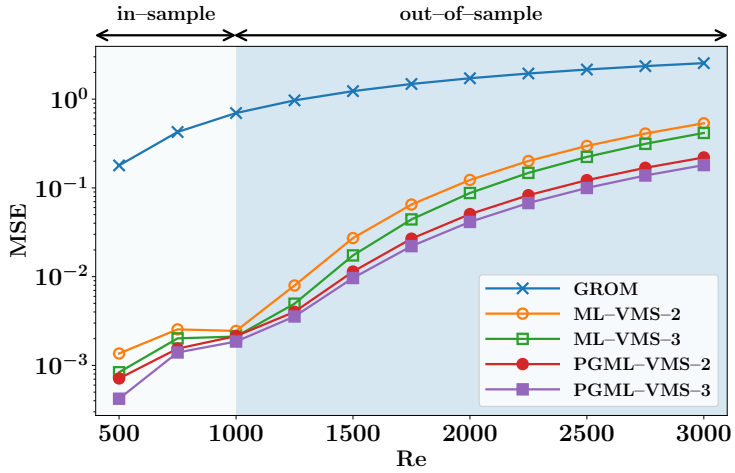

FIG. 11. *Mean squared error (MSE) between the true values of modal coefficients and the predictions of GROM, ML-VMS-*2*, ML-VMS-*3*, PGML-VMS-*2*, and PGML-VMS-*3.

The mean squared error (MSE) between the true projection values of the resolved scales and the prediction of the ROM with and without various closure models is shown in Figure 11. We can see that the VMS closure provides at least one order of magnitude better predictions than the baseline GROM. Moreover, the PGML-VMS is superior to the ML-VMS, especially for Reynolds numbers that are not included in the LSTM training. This can be attributed to the fact that PGML employs physics-based features derived from the governing equations, resulting in improved extrapolatory capabilities of the overall model. Finally, the three-level variant of VMS provides

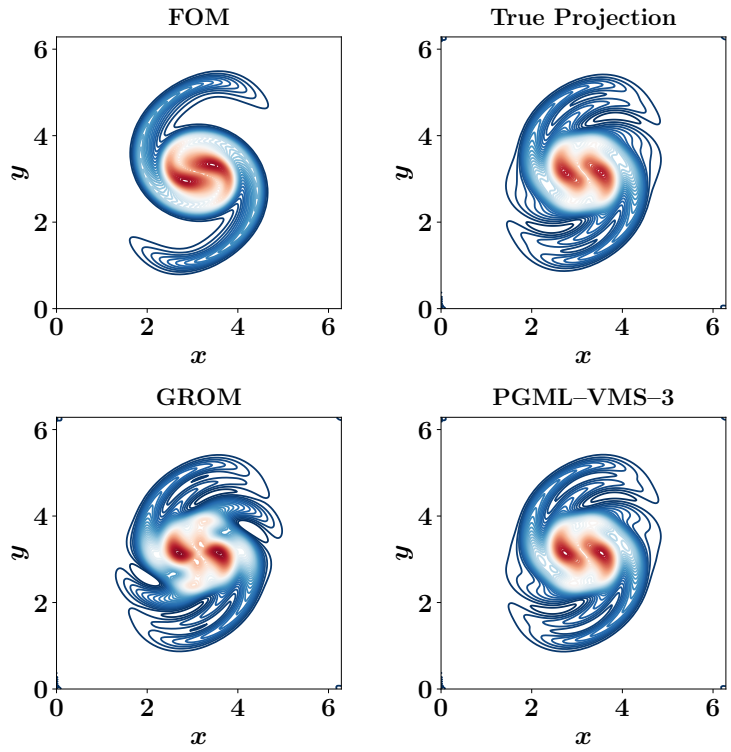

FIG. 12. *Comparison between the FOM vorticity field at the final time (i.e., $t = 30$) and the reconstruction from true projection (i.e., optimal reconstruction), GROM, and PGML-VMS-3. Note that the PGML-VMS-3 field is very similar to the true projection field, which implies that the closure error is minimized. However, there are clear differences between the FOM and PGML-VMS-3 results, which suggest a significant projection error in the PGML-VMS-3 model.*

more accurate ROMs than VMS-2, making use of the locality of information transfer to build more localized closure models.

6.2. NLPOD for unresolved scales. The reconstructed vorticity fields from GROM, true projection, and PGML-VMS-3 at the final time (i.e., $t = 30$) for $\mathrm{Re} = 3000$ are visualized in Figure 12. We can see that the GROM field is significantly inaccurate. In contrast, the PGML-VMS-3 vorticity field is very close to the true projection field. This suggests that the PGML-VMS-3 is successful in providing accurate closure terms in such a way that the resulting ROM trajectory converges to the best linear approximation with 6 modes. Nonetheless, compared to the FOM solution, it is clear that 6 POD modes are not enough to capture all the relevant flow structures, especially at large Reynolds numbers. On the other hand, building a projection-based ROM with an increased number of modes will result in an undesired higher computational burden.

In order to tackle this limitation, we apply the NLPOD methodology from section 5 to learn a latent space representation of important unresolved scales. We find that the value $K = 20$ corresponds to RIC $\geq 99.99\%$, so we consider $\boldsymbol{b} = \{a_k\}_{k=7}^{20} \in \mathbb{R}^{14}$ in the NLPOD extension. We use the NLPOD to learn a rank-2 compression of the resolved scales, i.e., $\boldsymbol{z} = \{z_k\}_{k=1}^{2} \in \mathbb{R}^2$. We use a total of 9 hidden feedforward layers to define the autoencoder. The first 4 layers with a hyperbolic tangent activation

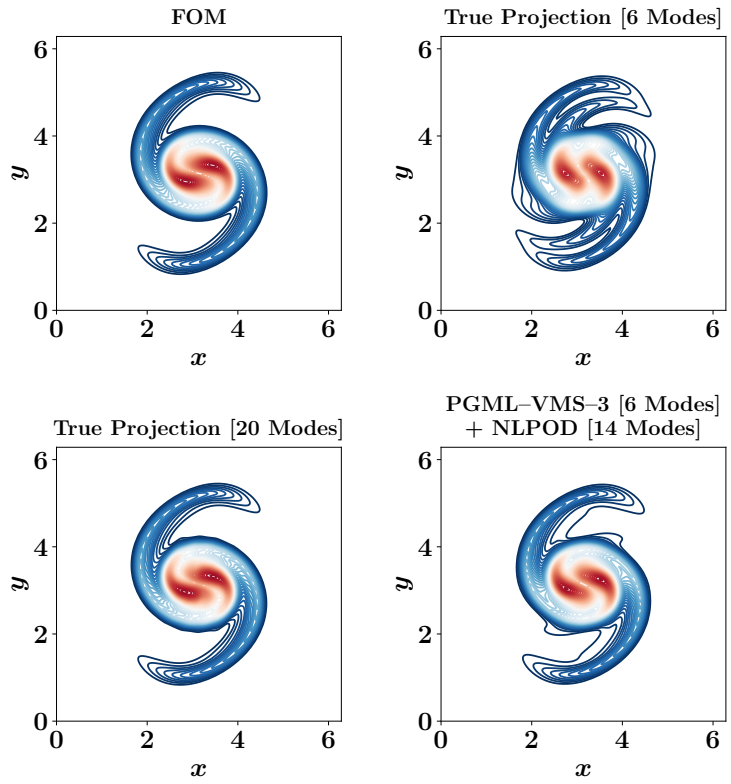

FIG. 13. *Comparison between the FOM vorticity field at the final time (i.e., $t = 30$) for $Re = 3000$ and the reconstruction from true projection (i.e., optimal reconstruction) at two different values of modal truncation, as well as the predictions of the PGML-VMS-3 for the dynamics of the first 6 modes, augmented with NLPOD for the following 14 modes (i.e., a total of $K = 20$ modes) to reduce the projection error.*

function define the encoder starting with 128 neurons in the first layer, followed by 64 neurons in the second layer, then 32 neurons in the third layer, and 8 neurons in the fourth layer. The fifth layer represents the bottleneck, corresponding to $\boldsymbol{z}$, with 2 neurons and a linear activation function. The decoder architecture is the same as the encoder, but in reverse order (i.e., starting from 8 neurons in the sixth layer up to 128 neurons in the ninth layer). The input and output layers have the same dimension, K. Figure 13 displays the reconstructed vorticity fields at the final time from the true projection of the FOM field onto the first 6 and the first 20 POD modes. We notice that the FOM flow scales can be adequately captured by the subspace spanned by the first 20 POD modes. Furthermore, the plots clearly show that the combination of PGML-VMS-3 for the first 6 modes and NLPOD for the subsequent 14 modes (i.e., a total of 20 modes) provides improved field reconstruction. We highlight that the computational overhead of the online deployment of the PGML-VMS closure and NLPOD is negligible compared to solving the projection-based ROM with 6 modes.

The CPU times for different portions of the FOM and ROMs are listed in Table 1. For the ROMs, we can see that the majority of the time is spent training the neural networks during the offline stage. We note that this time can be significantly reduced by considering parallel training algorithms that make use of distributed hardware facilities. We also notice that the three-level VMS framework takes about twice the

TABLE 1

Comparison of the CPU times for the offline and online stages for FOM and ROMs. Note that the PGML-VMS-3+NLPOD model yields a level of accuracy which is comparable to the GROM ($R = 20$) model with only a fraction of computational overhead (i.e., with a total computational online execution time of 63.876 s for the PGML-VMS-3+NLPOD model).

Offline CPU time [s]		Online CPU time [s]	
POD basis	0.646	FOM	1860.056
GROM operators	0.246	GROM ($R = 6$)	20.226
ML-VMS-2 training	71.641	ML-VMS-2 ($R = 6$)	32.289
ML-VMS-3 training	148.057	ML-VMS-3 ($R = 6$)	45.055
PGML-VMS-2 training	65.324	PGML-VMS-2 ($R = 6$)	33.358
PGML-VMS-3 training	139.863	PGML-VMS-3 ($R = 6$)	51.545
NLPOD training (AE)	111.543	NLPOD ($R = 6, K = 20$)	12.331
NLPOD training (LSTM)	85.234	GROM ($R = 20$)	604.427

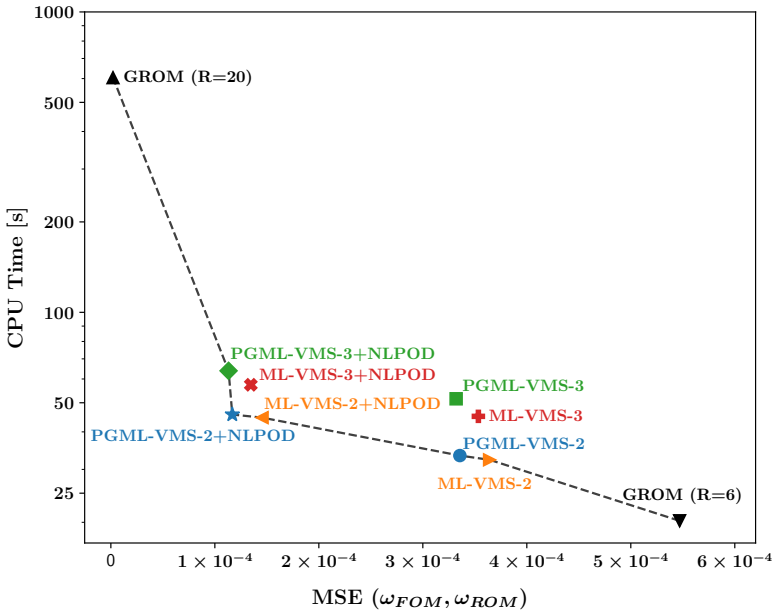

FIG. 14. *Pareto front plot for the mean squared error in the reconstructed vorticity field from different ROM approaches (compared to the FOM snapshots) versus the online CPU time.*

time taken by the two-level VMS due to the use of two distinct neural networks, which doubles the training and testing time. Nonetheless, we see that considerable computational gains are achieved compared to the FOM, by offloading most of the expensive computations to the offline stage, which results in computationally light models that can be used efficiently in the online stage. In addition, Figure 14 depicts the Pareto front diagram of the MSE of reconstructed vorticity field from different ROM approaches (compared to the FOM snapshots) versus the online CPU time. We can see that the GROMs with ($R = 6$) and ($R = 20$) correspond to the tails on the plot with the highest MSE and lowest CPU time on the right and the lowest MSE and highest CPU time on the left, respectively. We also observe that the costs of the ML and PGML frameworks are of the same order, which implies that incorporating physics-based features into the neural network latent space comes with negligible overheads.

7. Conclusions and future work. We propose a hybrid hierarchical learning approach for the reduced order modeling of nonlinear fluid flow systems. The core component of the proposed method comprises a multilevel variational multiscale (VMS) framework for the natural separation of the resolved modes of different length scales and unresolved modes. We develop a modular physics-guided machine learning (PGML) paradigm through the concatenation of neural network layers to enable the convergence of the ROM trajectory of resolved scales to the optimal low-rank approximation. We use the projection of the governing equations onto the POD modes as physics-based features to constrain the output to a manifold of the physically realizable solutions.

For a vorticity transport problem with high Reynolds numbers, we numerically demonstrate that this injection of physical information yields more robust and reliable ROM closures with reduced uncertainty levels. Moreover, we showcase the benefits of exploiting the locality of information transfer by building a three-level VMS, which centers around the scale-separation of the resolved modes into large resolved scales and small resolved scales. The numerical results show that the VMS-3 provides significant flexibility in defining the closure terms and is superior to the classical VMS-2 model used in previous studies. Finally, to decrease the projection error, we adapt the nonlinear proper orthogonal decomposition approach to learn a latent space representation of the unresolved ROM scales that yield a near-full rank approximation of the flow field.

Further investigations are required to optimize the layer(s) at which physics-based features are injected in the PGML framework. For example, we can add the injection at multiple points in the latent space, rather than at a single point. Moreover, we may fuse various information from different models by repeating the concatenation operator for each piece of information. It is worth noting that advanced hyperparameter tuning approaches for the automated design of neural network architectures (e.g., using genetic algorithms) can be utilized to find the optimal layer(s) to inject the physics into the PGML architectures. In the present study, the ML-VMS, PGML-VMS, and NLPOD components of the hybrid framework are treated separately. In other words, the training of each neural network takes place independently of other neural networks in the framework. In a follow-up study, we plan to explore the simultaneous training of these neural networks to ensure that these models are integrated seamlessly in the computational workflow. Finally, the truncated scales that are recovered by NLPOD can be further embedded in the PGML-VMS architecture to improve the approximation of the closure model.

REFERENCES

[1] S. E. Ahmed, S. Pawar, O. San, and A. Rasheed, *Reduced order modeling of fluid flows: Machine learning, Kolmogorov barrier, closure modeling, and partitioning*, in AIAA Aviation 2020 Forum, AIAA, 2020, 2946.

[2] S. E. Ahmed, S. Pawar, O. San, A. Rasheed, T. Iliescu, and B. R. Noack, *On closures for reduced order models—a spectrum of first-principle to machine-learned avenues*, Phys. Fluids, 33 (2021), 091301.

[3] S. E. Ahmed and O. San, *Breaking the Kolmogorov barrier in model reduction of fluid flows*, Fluids, 5 (2020), 26.

[4] S. E. Ahmed, O. San, A. Rasheed, and T. Iliescu, *A long short-term memory embedding for hybrid uplifted reduced order models*, Phys. D, 409 (2020), 132471.

[5] S. E. Ahmed, O. San, A. Rasheed, and T. Iliescu, *Nonlinear proper orthogonal decomposition for convection-dominated flows*, Phys. Fluids, 33 (2021), 121702.

[6] S. E. Ahmed, O. San, A. Rasheed, T. Iliescu, and A. Veneziani, *Physics Guided Machine Learning for Variational Multiscale Reduced Order Modeling: Python Codes*, https://github.com/Shady-Ahmed/PGML-VMS-NLPOD, 2022.

[7] I. Akhtar, A. H. Nayfeh, and C. J. Ribbens, *On the stability and extension of reduced-order Galerkin models in incompressible flows*, Theoret. Comput. Fluid Dyn., 23 (2009), pp. 213–237.

[8] D. Amsallem and B. Haasdonk, *PEBL-ROM: Projection-error based local reduced-order models*, Adv. Model. Simul. Eng. Sci., 3 (2016), pp. 1–25.

[9] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat, *Design optimization using hyper-reduced-order models*, Struct. Multidiscip. Optim., 51 (2015), pp. 919–940.

[10] M. Balajewicz and E. H. Dowell, *Stabilization of projection-based reduced order models of the Navier-Stokes*, Nonlinear Dynam., 70 (2012), pp. 1619–1632.

[11] M. Balajewicz, I. Tezaur, and E. Dowell, *Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier-Stokes equations*, J. Comput. Phys., 321 (2016), pp. 224–241.

[12] P. Benner, E. Sachs, and S. Volkwein, *Model order reduction for PDE constrained optimization*, in Trends in PDE Constrained Optimization, Springer, Cham, 2014, pp. 303–326.

[13] L. Bertagna and A. Veneziani, *A model reduction approach for the variational estimation of vascular compliance by solving an inverse fluid-structure interaction problem*, Inverse Problems, 30 (2014), 055006.

[14] S. Boschert and R. Rosen, *Digital twin—the simulation aspect*, in Mechanic Futures, Springer, New York, 2016, pp. 59–74.

[15] R. E. Brown and A. J. Line, *Efficient high-resolution wake modeling using the vorticity transport equation*, AIAA J., 43 (2005), pp. 1434–1443.

[16] T. Bui-Thanh, K. Willcox, O. Ghattas, and B. van Bloemen Waanders, *Goal-oriented, model-constrained optimization for reduction of large-scale systems*, J. Comput. Phys., 224 (2007), pp. 880–896.

[17] K. Carlberg, M. Barone, and H. Antil, *Galerkin v. least-squares Petrov-Galerkin projection in nonlinear model reduction*, J. Comput. Phys., 330 (2017), pp. 693–734.

[18] A. J. Chorin and O. H. Hald, *Stochastic Tools in Mathematics and Science*, Surveys Tutorials Appl. Math. Sci. 1, Springer-Verlag, New York, 2009.

[19] A. J. Chorin, O. H. Hald, and R. Kupferman, *Optimal prediction and the Mori-Zwanzig representation of irreversible processes*, Proc. Natl. Acad. Sci. USA, 97 (2000), pp. 2968–2973.

[20] A. J. Chorin, O. H. Hald, and R. Kupferman, *Optimal prediction with memory*, Phys. D, 166 (2002), pp. 239–257.

[21] R. Codina, S. Badia, J. Baiges, and J. Principe, *Variational multiscale methods in computational fluid dynamics*, in Encyclopedia Computational Mechanics, 2nd ed., Wiley, New York, 2018, pp. 1–28.

[22] J. Degroote, J. Vierendeels, and K. Willcox, *Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis*, Internat. J. Numer. Methods Fluids, 63 (2010), pp. 207–230.

[23] G. Dimitriu, R. Ştefănescu, and I. M. Navon, *Comparative numerical analysis using reduced-order modeling strategies for nonlinear large-scale systems*, J. Comput. Appl. Math., 310 (2017), pp. 32–43.

[24] M. Drohmann and K. Carlberg, *The ROMES method for statistical modeling of reduced-order-model error*, SIAM/ASA J. Uncertain. Quantif., 3 (2015), pp. 116–145, https://doi.org/10.1137/140969841.

[25] B. A. Freno and K. T. Carlberg, *Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations*, Comput. Methods Appl. Mech. Engrg., 348 (2019), pp. 250–296.

[26] F. A. Gers, D. Eck, and J. Schmidhuber, *Applying LSTM to time series predictable through time-window approaches*, in Neural Nets WIRN Vietri-01, Springer, New York, 2002, pp. 193–200.

[27] M. Girfoglio, A. Quaini, and G. Rozza, *A POD-Galerkin reduced order model for a LES filtering approach*, J. Comput. Phys., 436 (2021), 110260.

[28] A. Gouasmi, E. J. Parish, and K. Duraisamy, *A priori estimation of memory effects in reduced-order models of nonlinear systems using the Mori-Zwanzig formalism*, Proc. R. Soc. A Math. Phys. Eng. Sci., 473 (2017), 20170385.

[29] A. Gupta and P. F. Lermusiaux, *Neural closure models for dynamical systems*, Proc. R. Soc. A, 477 (2021), 20201004.

[30] S. Haag and R. Anderl, *Digital twin-proof of concept*, Manuf. Lett., 15 (2018), pp. 64–66.

[31] D. Hartmann, M. Herz, and 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.
[32] S. Hijazi, G. Stabile, A. Mola, and G. Rozza, *Data-driven POD-Galerkin reduced order model for turbulent flows*, J. Comput. Phys., 416 (2020), 109513.
[33] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge University Press, Cambridge, UK, 2012.
[34] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, *Deep learning with long short-term memory for time series prediction*, IEEE Commun. Mag., 57 (2019), pp. 114–119.
[35] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy, *The variational multiscale method—a paradigm for computational mechanics*, Comput. Methods Appl. Mech. Engrg., 166 (1998), pp. 3–24.
[36] T. J. Hughes, L. Mazzei, and K. E. Jansen, *Large eddy simulation and the variational multiscale method*, Comput. Vis. Sci., 3 (2000), pp. 47–59.
[37] T. J. Hughes, A. A. Oberai, and L. Mazzei, *Large eddy simulation of turbulent channel flows by the variational multiscale method*, Phys. Fluids, 13 (2001), pp. 1784–1799.
[38] T. Iliescu and Z. Wang, *Variational multiscale proper orthogonal decomposition: Convection-dominated convection-diffusion-reaction equations*, Math. Comp., 82 (2013), pp. 1357–1378.
[39] T. Iliescu and Z. Wang, *Variational multiscale proper orthogonal decomposition: Navier-stokes equations*, Numer. Methods Partial Differential Equations, 30 (2014), pp. 641–663.
[40] H. Imtiaz and I. Akhtar, *Nonlinear closure modeling in reduced order models for turbulent flows: A dynamical system approach*, Nonlinear Dynam., 99 (2020), pp. 479–494.
[41] A. Iollo, S. Lanteri, and J.-A. Désidéri, *Stability properties of POD-Galerkin approximations for the compressible Navier-Stokes equations*, Theoret. Comput. Fluid Dyn., 13 (2000), pp. 377–396.
[42] K. Ito and S. S. Ravindran, *A reduced-order method for simulation and control of fluid flows*, J. Comput. Phys., 143 (1998), pp. 403–425.
[43] V. John, *Finite Element Methods for Incompressible Flow Problems*, Springer, Cham, 2016.
[44] M. G. Kapteyn, J. V. Pretorius, and K. E. Willcox, *A probabilistic graphical model foundation for enabling predictive digital twins at scale*, Nature Comput. Sci., 1 (2021), pp. 337–347.
[45] B. Koc, C. Mou, H. Liu, Z. Wang, G. Rozza, and T. Iliescu, *Verifiability of the data-driven variational multiscale reduced order model*, J. Sci. Comput., 93 (2022), pp. 1–26.
[46] Y.-d. Lang, A. Malacina, L. T. Biegler, S. Munteanu, J. I. Madsen, and S. E. Zitney, *Reduced order model based on principal component analysis for process simulation and optimization*, Energy Fuels, 23 (2009), pp. 1695–1706.
[47] K. Lee and K. T. Carlberg, *Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders*, J. Comput. Phys., 404 (2020), 108973.
[48] D. J. Lucia, P. S. Beran, and W. A. Silva, *Reduced-order modeling: New approaches for computational physics*, Progr. Aerosp. Sci., 40 (2004), pp. 51–117.
[49] H. Mori, *Transport, collective motion, and Brownian motion*, Progr. Theoret. Phys., 33 (1965), pp. 423–455.
[50] C. Mou, B. Koc, O. San, L. G. Rebholz, and T. Iliescu, *Data-driven variational multiscale reduced order models*, Comput. Methods Appl. Mech. Engrg., 373 (2021), 113470.
[51] C. Mou, E. Merzari, O. San, and T. Iliescu, *An Energy-Based Lengthscale for Reduced Order Models of Turbulent Flows*, preprint, https://arxiv.org/abs/2211.04404, 2022.
[52] B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, and F. Thiele, *A hierarchy of low-dimensional models for the transient and post-transient cylinder wake*, J. Fluid Mech., 497 (2003), pp. 335–363.
[53] B. R. Noack, M. Morzynski, and G. Tadmor, *Reduced-Order Modelling for Flow Control*, CSIM Internat. Centre Mech. Sci. 528, Springer, Vienna, 2011.
[54] B. R. Noack, P. Papas, and P. A. Monkewitz, *The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows*, J. Fluid Mech., 523 (2005), pp. 339–365.
[55] S. Pan and K. Duraisamy, *Data-driven discovery of closure models*, SIAM J. Appl. Dyn. Syst., 17 (2018), pp. 2381–2413, https://doi.org/10.1137/18M1177263.
[56] E. J. Parish and K. T. Carlberg, *Time-series machine-learning error models for approximate solutions to parameterized dynamical systems*, Comput. Methods Appl. Mech. Engrg., 365 (2020), 112990.
[57] S. Pawar, O. San, B. Aksoylu, A. Rasheed, and T. Kvamsdal, *Physics guided machine learning using simplified theories*, Phys. Fluids, 33 (2021), 011701.

[58] S. Pawar, O. San, A. Nair, A. Rasheed, and T. Kvamsdal, *Model fusion with physics-guided machine learning: Projection-based reduced-order modeling*, Phys. Fluids, 33 (2021), 067123.
[59] S. Pawar, O. San, P. Vedula, A. Rasheed, and T. Kvamsdal, *Multi-Fidelity Information Fusion with Concatenated Neural Networks*, preprint, https://arxiv.org/abs/2110.04170, 2021.
[60] B. Peherstorfer, *Breaking the Kolmogorov barrier with nonlinear model reduction*, Notices Amer. Math. Soc., 69 (2022), pp. 725–733.
[61] B. Peherstorfer, K. Willcox, and M. Gunzburger, *Survey of multifidelity methods in uncertainty propagation, inference, and optimization*, SIAM Rev., 60 (2018), pp. 550–591, https://doi.org/10.1137/16M1082469.
[62] S. M. Rahman, S. E. Ahmed, and O. San, *A dynamic closure modeling framework for model order reduction of geophysical flows*, Phys. Fluids, 31 (2019), 046602.
[63] A. Rasheed, O. San, and T. Kvamsdal, *Digital twin: Values, challenges and enablers from a modeling perspective*, IEEE Access, 8 (2020), pp. 21980–22012.
[64] S. Ravindran, *Reduced-order adaptive controllers for fluid flows using POD*, J. Sci. Comput., 15 (2000), pp. 457–478.
[65] C. W. Rowley and S. T. M. Dawson, *Model reduction for flow analysis and control*, Ann. Rev. Fluid Mech., 49 (2017), pp. 387–417.
[66] E. W. Sachs and S. Volkwein, *POD-Galerkin approximations in PDE-constrained optimization*, GAMM-Mitt., 33 (2010), pp. 194–208.
[67] O. San and T. Iliescu, *Proper orthogonal decomposition closure models for fluid flows: Burgers equation*, Int. J. Numer. Anal. Model. Ser. B, 5 (2014), pp. 217–237.
[68] O. San and T. Iliescu, *A stabilized proper orthogonal decomposition reduced-order model for large scale quasigeostrophic ocean circulation*, Adv. Comput. Math., 41 (2015), pp. 1289–1319.
[69] O. San and R. Maulik, *Extreme learning machine for reduced order modeling of turbulent geophysical flows*, Phys. Rev. E, 97 (2018), 042322.
[70] O. San and A. E. Staples, *A coarse-grid projection method for accelerating incompressible flow computations*, J. Comput. Phys., 233 (2013), pp. 480–508.
[71] T. P. Sapsis and A. J. Majda, *Blending modified Gaussian closure and non-Gaussian reduced subspace methods for turbulent dynamical systems*, J. Nonlinear Sci., 23 (2013), pp. 1039–1071.
[72] S. Siami-Namini, N. Tavakoli, and A. S. Namin, *The performance of LSTM and BiLSTM in forecasting time series*, in 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019, pp. 3285–3292.
[73] J. R. Singler, *New POD error expressions, error bounds, and asymptotic results for reduced order models of parabolic PDEs*, SIAM J. Numer. Anal., 52 (2014), pp. 852–876, https://doi.org/10.1137/120886947.
[74] L. Sirovich, *Turbulence and the dynamics of coherent structures.* I. *Coherent structures*, Quart. Appl. Math., 45 (1987), pp. 561–571.
[75] L. Sirovich, *Turbulence and the dynamics of coherent structures.* II. *Symmetries and transformations*, Quart. Appl. Math., 45 (1987), pp. 573–582.
[76] L. Sirovich, *Turbulence and the dynamics of coherent structures.* III. *Dynamics and scaling*, Quart. Appl. Math., 45 (1987), pp. 583–590.
[77] R. Ştefănescu, A. Sandu, and I. M. Navon, *Comparison of POD reduced order strategies for the nonlinear* 2*D shallow water equations*, Internat. J. Numer. Methods Fluids, 76 (2014), pp. 497–521.
[78] P. Stinis, *Renormalized Mori-Zwanzig-reduced models for systems without scale separation*, Proc. R. Soc. A Math. Eng. Sci., 471 (2015), 20140446.
[79] R. Swischuk, L. Mainini, B. Peherstorfer, and K. Willcox, *Projection-based model reduction: Formulations for physics-based machine learning*, Comput. Fluids, 179 (2019), pp. 704–717.
[80] K. Taira, S. L. Brunton, S. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley, *Modal analysis of fluid flows: An overview*, AIAA J., 55 (2017), pp. 4013–4041.
[81] K. Taira, M. S. Hemati, S. L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S. T. Dawson, and C.-A. Yeh, *Modal analysis of fluid flows: Applications and outlook*, AIAA J., 58 (2020), pp. 998–1022.
[82] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, *Digital twin in industry: State-of-the-art*, IEEE Trans. Ind. Inform., 15 (2018), pp. 2405–2415.

[83] M. Viceconti, F. Pappalardo, B. Rodriguez, M. Horner, J. Bischoff, and F. M. Musuamba Tshinanu, *In silico trials: Verification, validation and uncertainty quantification of predictive models used in the regulatory evaluation of biomedical products*, Methods, 185 (2021), pp. 120–127.

[84] S. Volkwein, *Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling*, Lecture Notes, University of Konstanz, 2013.

[85] Q. Wang, N. Ripamonti, and J. S. Hesthaven, *Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism*, J. Comput. Phys., 410 (2020), 109402.

[86] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu, *Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison*, Comput. Methods Appl. Mech. Engrg., 237 (2012), pp. 10–26.

[87] X. Xie, M. Mohebujjaman, L. G. Rebholz, and T. Iliescu, *Data-driven filtered reduced order modeling of fluid flows*, SIAM J. Sci. Comput., 40 (2018), pp. B834–B857, https://doi.org/10.1137/17M1145136.

[88] H. Yang and A. Veneziani, *Efficient estimation of cardiac conductivities via POD-DEIM model order reduction*, Appl. Numer. Math., 115 (2017), pp. 180–199.

[89] M. J. Zahr and C. Farhat, *Progressive construction of a parametric reduced-order model for PDE-constrained optimization*, Internat. J. Numer. Methods Engrg., 102 (2015), pp. 1111–1135.

[90] R. Zwanzig, *Problems in nonlinear transport theory*, in Systems Far from Equilibrium, Springer, Berlin, 1980, pp. 198–225.

[91] R. Zwanzig, *Nonequilibrium Statistical Mechanics*, Oxford University Press, Oxford, UK, 2001.