MODEL-DATA FUSION IN DIGITAL TWINS OF LARGE SCALE DYNAMICAL SYSTEMS

By

SHADY E. AHMED

Bachelor of Science in Mechanical Power Engineering,
Mansoura University,
Mansoura, Egypt,
2013

Master of Science in Mechanical Power Engineering,
Mansoura University,
Mansoura, Egypt,
2017

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
July, 2022

MODEL-DATA FUSION IN DIGITAL TWINS OF LARGE SCALE DYNAMICAL SYSTEMS

Dissertation Approved:

Omer San, PhD

Dissertation Advisor

Jamey Jacob, PhD

Kursat Kara, PhD

JaEun Ku, PhD

ACKNOWLEDGMENTS

Alhamdulillah, all praises are due to Allah the Almighty for the opportunities and success I have been blessed with. The past four years as a graduate student at Oklahoma State University (OSU) have been some of the most challenging yet rewarding experiences of my life. I am leaving with good memories and I am grateful to everyone who has enriched my graduate school experience.

I would first like to thank my advisor, Dr. Omer San, for taking a chance on me in the fall of 2018. I appreciate all his contributions of time and ideas to make my PhD journey productive and stimulating. I would also like to thank the members of my dissertation committee, Dr. Jamey Jacob, Dr. Kursat Kara, and Dr. JaEun Ku, for their time and support in providing constructive feedback on my research. In addition, I have learnt a lot from the OSU faculty members whom I have interacted with, both in the classroom and outside it.

Much of what is in this document was made possible by continuous interactions with research collaborators from outside OSU as well. A special thanks goes to Dr. Adil Rasheed from the Norwegian University of Science and Technology (NTNU) for our discussions on digital twins technologies and Dr. Traian Iliescu from Virginia Tech for his insights on developing the variational multi-scale reduced order modeling framework. I am also grateful for the guidance and mentoring provided by Dr. Sivaramakrishnan Lakshmivarahan from the University of Oklahoma on adopting and analyzing data assimilation techniques. In no particular order, I would like to thank

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

Dr. Rami Younis from the University of Tulsa, Dr. Bernd Noack from TU Berlin and Harbin Institute of Technology, Dr. Alessandro Veneziani from Emory University and Dr. Mandar Tabib from NTNU for our fruitful discussions and creating a welcoming collaborative environment. I am also grateful to Dr. Ahmed Attia from Argonne National Laboratory for hosting me as summer intern in 2021 and introducing me to the national laboratories work environment.

I am indebted to my fellow graduate students in the school of mechanical and aerospace engineering (MAE) who have contributed immensely to my personal and professional time at OSU: Suraj Pawar, Mashfiqur Rahman, Harsha Vaddireddy, Mansoor Ahmed, Romit Maulik, Furkan Oz, and Rohit Vuppala. A special shout-out goes to Suraj Pawar who has been a huge inspiration, both technically and personally, by his hard work and friendship.

I gratefully acknowledge the funding sources that made my PhD work possible. I would like to extend thanks to the US DOE Office of Science and the NSF for their generous financial support. I was also funded by the OSU John Brammer graduate research fellowship, the college of engineering, architecture and technology initiative for special graduate student support, the OSU graduate and professional student government association travel award, the American physical society (APS) travel grant, and the society for industrial and applied mathematics (SIAM) travel award.

Last but not least, words cannot describe how lucky I am to have the family and loved ones that I have in my life, including those who are back home in Egypt and my small family (Aya, Fayrouza and Yaseen) who could make it to Stillwater. Your unconditional love, prayers and encouragement have made all the difference and helped me through the hard times. It is because of all of you that I am where I am today and I am who I am today. Thank you for everything!

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

To my loving parents, dearest sister, charming wife, and adorable kids.

Name: Shady E. Ahmed

Date of Degree: July, 2022

Title of Study: Model-Data Fusion in Digital Twins of Large Scale Dynamical Systems

Major Field: Mechanical & Aerospace Engineering

Abstract: Digital twins (DTs) are virtual entities that serve as the *real-time* digital counterparts of actual physical systems across their *life-cycle*. In a typical application of DTs, the physical system provides sensor measurements and the DT should incorporate the incoming data and run different simulations to assess various scenarios and situations. As a result, an informed decision can be made to alter the physical system or at least take necessary precautions, and the process is repeated along the system's life-cycle. Thus, the effective deployment of DTs requires fulfilling multi-queries while communicating with the physical system in real-time. Nonetheless, **DTs of large-scale dynamical systems, as in fluid flows, come with three grand challenges that we address in this dissertation.**

First, the high dimensionality makes full order modeling (FOM) methodologies unfeasible due to the associated computational time and memory costs. In this regard, reduced order models (ROMs) can potentially accelerate the forward simulations by orders of magnitude, especially for systems with recurrent spatial structures. However, traditional ROMs yield inaccurate and unstable results for turbulent and convective flows. Therefore, we propose a hybrid variational multi-scale framework that benefits from the locality of modal interactions to deliver accurate ROMs. Furthermore, we adopt a novel physics guided machine learning technique to provide on-the-fly corrections and elevate the trustworthiness of the resulting ROM in the sparse data and incomplete governing equations regimes.

Second, complex natural or engineered systems are characterized by multi-scale, multi-physics, and multi-component nature. The efficient simulation of such systems requires quick communication and information sharing between several heterogeneous computing units. In order to address this challenge, we pioneer an interface learning (IL) paradigm to ensure the seamless integration of hierarchical solvers with different scales, physics, abstractions, and geometries without compromising the integrity of the computational setup. We demonstrate the IL paradigm for non-iterative domain decomposition and the FOM-ROM coupling in multi-fidelity computations.

Third, fluid flow systems are continuously evolving and thus the validity of the DT should be warranted across varying operating conditions and flow regimes. To do so, we embed data assimilation (DA) techniques to enable the DT to self-adapt based on in-situ observational data and efficiently replicate the physical system. In addition, we combine DA algorithms with machine learning models to build a robust framework that collectively addresses the model closure problem, the error in prior information, and the measurement noise.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Motivation and Scope of Current Work

With the advent in modeling and simulation capabilities as well as the availability of modern data acquisition facilities, it is desired nowadays to seek ways that help understand the world in a more dynamic and interactive way than ever before. Defined as the virtual counterpart of a physical system across its life-cycle, digital twin (DT) technologies have placed themselves as an appealing solution in what is called the fourth wave of industrial revolution (also known as Industry 4.0). DTs are not only useful during the conceptualization and prototyping phases, but also during the operation phase. Typically, the physical system sends real-time measurements and observations to its DT which in turn runs multiple simulations to make predictions for candidate what-if scenarios. These predictions are eventually embedded in a decision-making process for active control, risk management, and monitoring purposes. Since the physical system and its DT need to be communicated and coupled continuously, efficient model-data fusion techniques are necessary for the reliable implementation of DTs.

The objective of this dissertation is to develop efficient, accurate, and reliable model-data fusion frameworks that address key challenges associated with large scale dynamical systems, like those encountered in geophysical fluid flow systems. In particular, we focus on three major aspects of such systems: (1) dimensionality, (2) complexity, and (3) evolution as shown in Fig. 1.1. The higher dimensionality dictates the need to develop lightweight proxy models that are capable of providing reliable predictions in near real-time. Common reduced order modeling techniques are briefly outlined in Section 1.2 and our developments along these lines are presented in Part I. In addition, the complexity of the system poses significant challenges for monolithic computing environments due to the disparity of resolved temporal and spatial scales. Alternatively, different solvers can be dedicated to different scales, physics, geometries, and components and eventually information should be passed among these

computing units. Section 1.3 and Part II respectively introduce and showcase the new interface learning (IL) paradigm that we design to address various modeling and data interfaces that appear in many forms in modern computational workflows and can threaten computational efficiency, data integrity, or physical coupling. Finally, Section 1.4 highlight the need to build self-adaptive models as the actual system undergoes changes in the operation conditions or flow regimes. Part III details how we combine machine learning, modal analysis, physical arguments, and data assimilation to build efficient models where the underlying parameters and model outputs are updated using new data streams.

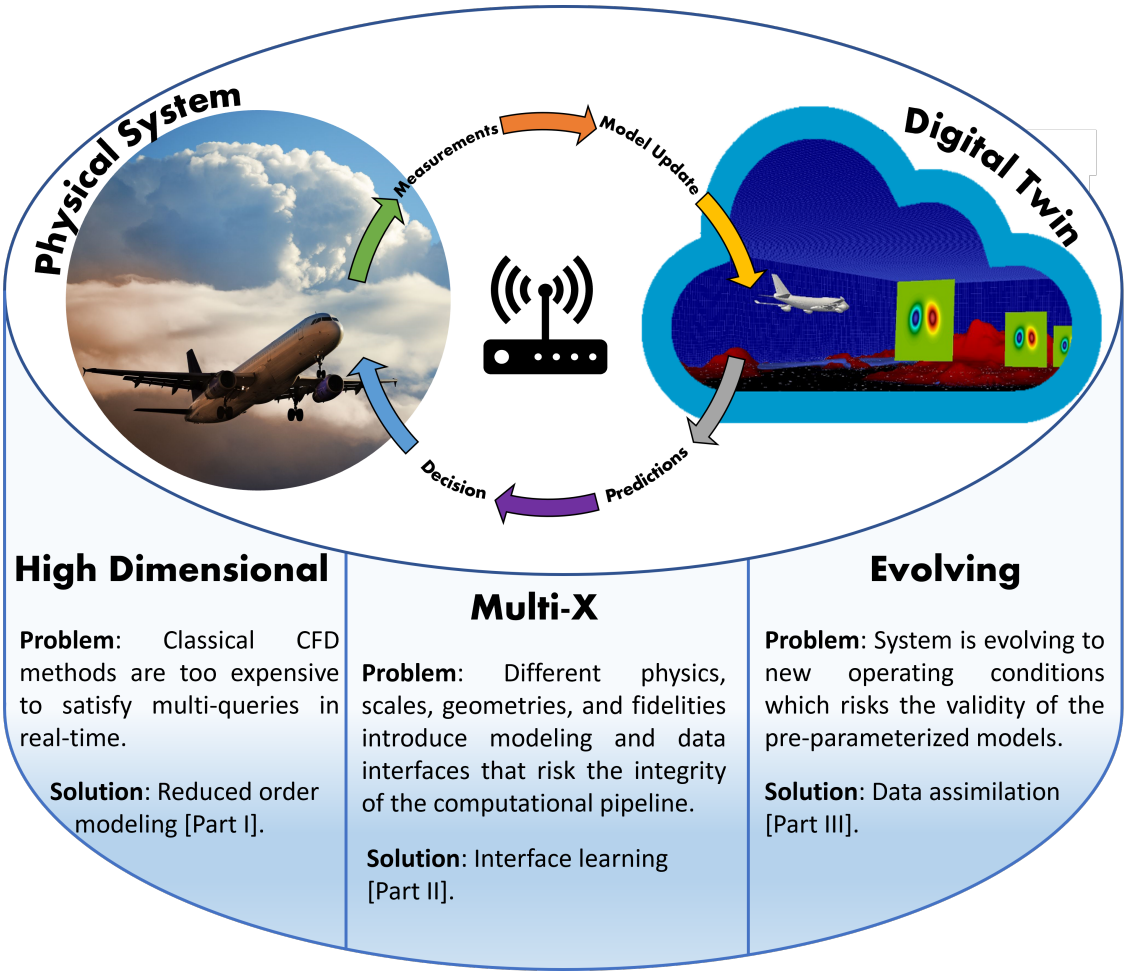


Figure 1.1: A schematic representation of digital twin frameworks and the scope of this dissertation.

1.2 Fluid Flows Are High-Dimensional

Classical computational fluid dynamics (CFD) solving the full order model (FOM) of the underlying system cannot fulfill the multi-query nature of DTs due to the associated computational time and memory costs. Fluid flows are often extremely high-dimensional once a standard discretization approach is applied. While the DT is required to operate in real-time, a single run of the FOM simulations can take hours, or even days, which makes the decision making process too delayed. In this regard, reduced order models (ROMs) offer a key enabler for next-generation DT frameworks. In general, ROMs present approximate substitutes to the original system that are cheaper to solve and analyze while retaining the essential characteristics of the original system. Significant successes of ROM efforts have largely relied on the fact that high dimensional systems are often dominated by a few number of underlying structures controlling most of the bulk mass, momentum, and energy transfer. ROMs can be built by using available *data* to identify and rank these structures, choosing the most effective few of them, and tracking their dynamical behavior in order to approximate the evolution of the underlying flow. The computational cost of the relatively low-dimensional ROMs is dramatically lower than the computational cost of a direct numerical simulation that aims at capturing all the flow scales.

Reduced order modeling techniques can be widely sorted into three classes: (a) intrusive methods that require knowledge of the underlying governing equations to derive the ROM, (b) non-intrusive techniques that rely solely on the data to identify approximate flow maps, and (c) hybrid techniques that aim at drawing from both approaches. A brief overview of these methodologies is presented here while a detailed discussion can be found in our recent review article on ROM techniques [1].

Intrusive reduced order models Classical ROM efforts have been mainly focused on projection-based techniques, where the governing equations are projected onto a set of basis functions that define a reduced order representation (ROR). Physically interpretable RORs are possible when dominant coherent structure are present. This is clearly ubiquitous in the fluid flows that we encounter in our daily life as well as in large-scale and industrial settings. For spatio-temporal dynamical systems, the rank-r ROR of the state $\mathbf{u}(\mathbf{x}, t)$ can be simply written as

$$\mathbf{u}((\mathbf{x}, t) = \sum_{i=1}^{r} a_i(t)\psi_i((\mathbf{x}), \tag{1.1}$$

where $\mathbf{x}$ refers to the spatial coordinates, t is the time, ψ_i denotes the i-th mode in the ROR, and a_i is the corresponding amplitude or coefficient. In fluid dynamics community, proper orthogonal decomposition (POD) is, generally speaking, the most popular and effective technique to produce hierarchically ordered solution-adapted basis functions (or modes) that provide the optimal basis to represent a given collection of field data or snapshots.

To emulate system's dynamics and make predictions for the time-dependent coefficients a_i, a surrogate model is often built by performing a Galerkin projection of the FOM operators onto a reduced rank subspace spanned by the selected POD modes. For computational efficiency, Galerkin ROMs (GROMs) usually work in *under-resolved* regimes. This means that a severe modal truncation is performed, keeping only the first few leading POD modes. The repercussions of the Galerkin truncation and projection is two-fold (see Fig. 1.2). First, the representability of the resulting subspace (i.e., approximating the state as a linear superposition of the retained POD basis functions) is limited, giving rise to the projection error. Second, the interactions between the truncated and the retained modes can be significant and those interactions are often discarded in the Galerkin projection step. Thus, the GROM cannot capture the dynamics of the resolved modes accurately, introducing a closure error. As a result, traditional ROMs yield inaccurate and unstable predictions for stiff systems and convection-dominated flows, which affects the reliability of ROMs for realistic applications and hinders their applicability in DT frameworks.

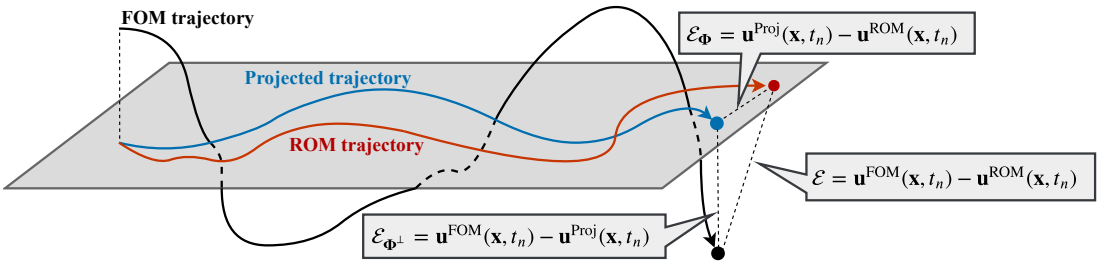


Figure 1.2: An illustration of error sources in GROM as example of intrusive ROMs (adapted from Ahmed et al. [2]).

Nonintrusive reduced order models The emergence and popularity of data-driven tools and open-source software libraries have given rise to a new class of ROM techniques that rely solely on data to define a surrogate model. These are often denoted as nonintrusive ROMs (NIROM) or physics-agnostic ROMs since they do not require access to the governing equations or modification to the source codes of

available solvers [3, 4]. Nonintrusive approaches are attractive due to their portability since they do not necessarily require the exact form of the equations or the methods used to generate the data. In addition, nonintrusive models offer a unique advantage in multidisciplinary collaborative environments, where only specific data can be shared without revealing the proprietary or sensitive information. Nonintrusive approaches are also useful when the detailed governing equations of the problem are unknown. This modeling approach can benefit from the enormous amount of data collected from experiments, sensor measurements, and high-fidelity simulations to build NIROMs based on the assumption that data is a manifestation of *all* the underlying dynamics and processes.

Machine learning (ML) tools, in particular artificial neural networks (ANNs) equipped with the universal approximation theorem [5], have been widely used in this regard. A typical feed-forward neural network is depicted in Fig. 1.3, where a mapping $\boldsymbol{\mathcal{M}}$ from the input $\boldsymbol{\mathcal{X}}$ to the output $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{X}})$ is inferred through a learning algorithm. For example, the Galerkin projection of intrusive ROM in Section 1.2 can be bypassed and a NIROM can be constructed by using ANNs to learn the mapping from the current values of $\mathbf{a} = [a_1, a_2, \ldots, a_r]$ to their future values in a sequential implementation. The application of variants of ANNs as regression models for the dynamics of low-order states (e.g., POD amplitudes) has gained substantial popularity [6–8]. In addition, ANNs can be utilized to identify a latent space with lower dimensionality than POD using auto-encoder architectures, see Fig. 1.4. This is a hot topic and dozens of new papers appear every week in different journals and conferences all over the world, dealing with different aspects of NIROM based on ANNs (e.g., different architectures, test bed problem, and error bounds).

Another family of NIROM that has gained special attention in recent years is the dynamic mode decomposition (DMD) based on the Koopman approximation theory. This approximation relies on a transformation that lifts the dynamics from state space where dynamics might be nonlinear to the observable space where the dynamics are linear but infinite-dimensional (see Fig. 1.5). DMD can be viewed as a data-driven approximation of the Koopman operator spectrum and it can be applied to both experimental and numerical data. In DMD, each mode is associated with a unique growth/decay rate and oscillation frequency. Therefore, DMD not only reveals information about the system's spatial characteristics, but also delivers knowledge about the system's temporal evolution, resulting in a purely data-driven ROM [9].

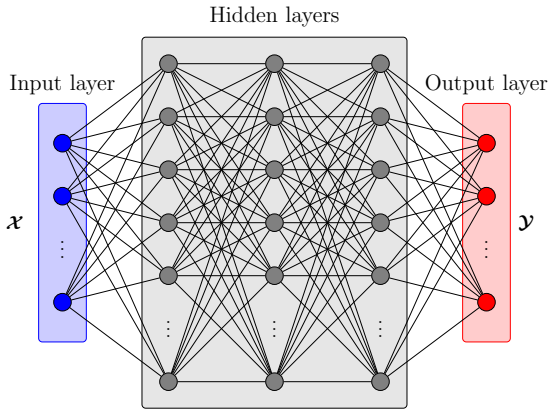


Figure 1.3: A schematic diagram of a typical feed-forward neural network with an input layer, hidden layers, and an output layer (adapted from Ahmed et al. [1]).

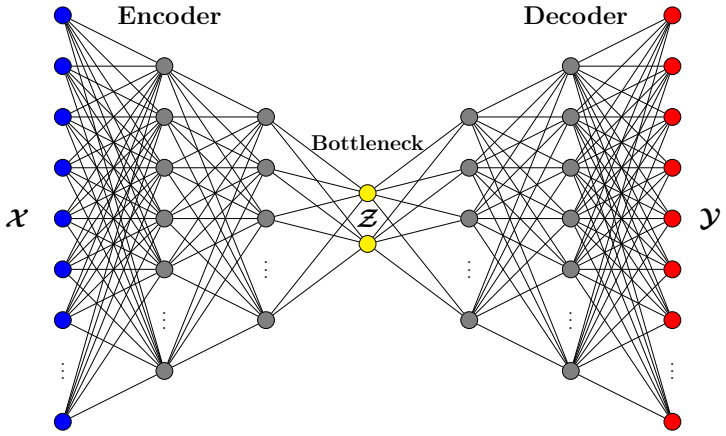


Figure 1.4: A schematic diagram for an autoencoder (AE) for latent space construction, where the input $\mathcal{X}$ is the full field, the output $\mathcal{Y}$ designates its reconstruction, and the bottleneck $\mathcal{Z}$ represents the latent space (compressed) variables (adapted from Ahmed et al. [1]).

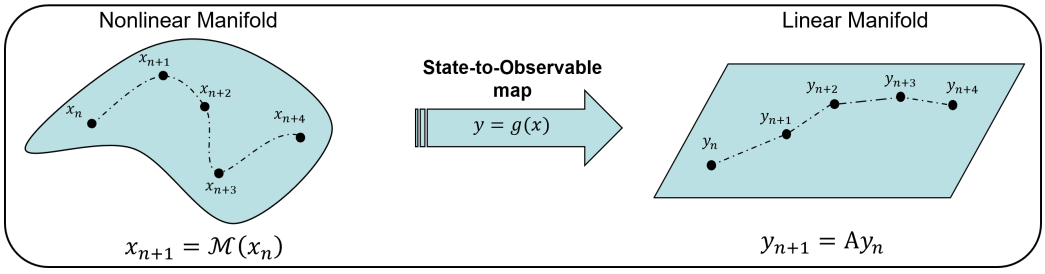


Figure 1.5: The Koopman viewpoint: while the original system with state x evolves according to $\mathcal{M}$ on a nonlinear manifold (left), the selection of appropriate observables $y = g(x)$ provides a substitute system where the dynamics can be approximated using a linear flow map (right) (adapted from Ahmed et al. [10]).

Hybrid reduced order models While physics-based GROMs are generally more interpretable and generalizable than NIROM, the latter has the capability to efficiently assimilate different sources of data and make use of advances in ML and data analytic tools. The preference of one approach over the other is not often a clear-cut choice. Therefore, hybrid ROMs aim to maintain the advantages of both intrusive and nonintrusive ROM methodologies and mitigate their limitations by combining key ideas and principles from physics-based modeling with the capabilities of modern data-driven tools. In this regard, we propose two hybrid frameworks that leverage the generalizability, interpretability and robustness of GROMs while complementing them with data-driven capabilities to improve their accuracy and efficiency.

- In the first framework, we introduce an uplifted reduced order modeling (UROM) approach through the integration of standard projection based methods with long short-term memory (LSTM) embedding. UROM has three modeling layers or components to enable accurate and efficient predictive capabilities. In the core layer, we utilize an intrusive projection approach to model the dynamics represented by the largest modes. Then, the second layer consists of an LSTM model to account for the residual effect of the discarded modes onto the dynamics of the largest scales. Finally, the third layer employs an uplifting operation that expands the span of the ROR by learning the correlations between the leading and trailing modes in the POD expansion. This layer introduces a super-resolution effect by recovering some of the truncated flow scales at desired times. Therefore, our model integrates a physics-based projection model with a memory embedded LSTM closure and an LSTM based super-resolution model. Moreover, the generalizability of UROM for unseen conditions is enhanced by employing efficient Grassmann manifold interpolation techniques.

- In the second approach, we propose a hybrid variational multi-scale (VMS) framework that benefits from the locality of energy transfer and modal interactions to deliver an accurate ROM trajectory. In particular, we leverage the VMS framework to characterize the hierarchical structure of the ROM basis. In the first step, ROM projection is used to naturally separate the scales into three categories: (i) resolved large scales, (ii) resolved small scales, and (iii) unresolved scales. In the second step, the terms representing the interactions among the three types of scales are explicitly identified and we employ the ANN to provide on-the-fly corrections to different ROM scales.

Furthermore, we adopt a novel physics-guided machine learning (PGML) methodology to elevate the trustworthiness of traditional ML models such as ANNs in real physical situations. We modify the ANN architectures by utilizing layer concatenation to inject physical information at different points in the ANN latent space. This adaptation improves the performance during both the training and deployment phases and results in significant reduction in the uncertainty levels of the model prediction, especially in the sparse data and incomplete governing equations regimes. We extend the PGML algorithms to model the interaction among the three types of ROM scales and build ROM operators that are closest to the true terms evaluated with FOM. Moreover, we develop a modular nonlinear POD (NLPOD) methodology to reduce the projection error by considering the correlations amongst the small unresolved scales. In particular, a feedforward autoencoder is employed to to learn a latent space representation of the unresolved ROM scales that yield a near-full rank approximation of the flow field. The evolution of the latent space variables for the unresolved scales is emulated using an LSTM neural network. Therefore, we address different errors resulting from the Galerkin POD approach in a hybrid framework that benefits from the locality of energy transfer, which is one of the cornerstone of the VMS method.

1.3 Complex Systems Are Multi-X

The second thrust of our model-data fusion developments is related to the system complexity. In particular, complex systems that constitute the target of the next generation of digital twins are often characterized by a multi-scale, multi-physics and multi-component (multi-X for short) nature. In a naive implementation, the stiffest part of the system dictates the overall spatial mesh resolution and time stepping requirements, making the numerical simulations computationally daunting. Instead, an ensemble of different solvers should be considered to address different scales, physics, domains and/or geometries. Nonetheless, efficient simulations of such systems require quick communication and information sharing between several heterogeneous computing units. In order to address this challenge, we establish the interface learning (IL) paradigm shift to allow the seamless integration of hierarchical solvers with different levels of complexity, fidelity and abstractions. We foresee that the development of IL techniques will have far reaching impacts on a large variety of problems as

shown in Fig. 1.6. We also highlight that IL enables us to focus certain computational resources on the region or scales of interest.

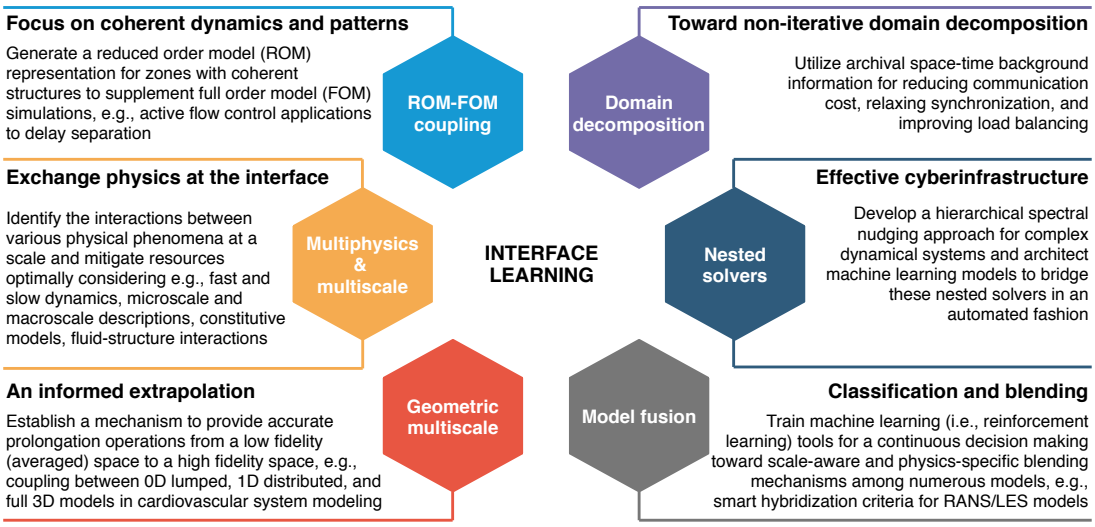


Figure 1.6: Overview of the interface learning paradigm considering numerous scientific and engineering interpretations.

We demonstrate the success of IL in two potential applications of particular interest in multi-scale and multi-physics problems as follows:

- We propose an IL framework that allows for system partitioning and non-iterative domain decomposition by utilizing time history of data to infer physically-consistent boundary conditions at the interface to reduce the communication costs between different computing units. Moreover, we put forth the concept of *upwind learning* towards a physics-informed domain decomposition to enable IL for hyperbolic systems by considering the domain of influence and wave structures into account. We highlight that high-performance computing environments can benefit from this methodology to reduce communication costs among processing units in emerging ML-ready heterogeneous platforms toward exascale era.

- We devise an IL methodology that enable the efficient coupling of multi-fidelity solvers. In particular, we consider scenarios where parts of the domain and physics are emulated using a lower fidelity ROM while the rest of the computational domain is resolved using higher fidelity FOM as shown in Fig. 1.7. We demonstrate the advantages of IL by developing a series of data-augmented correction, uplifting and prolongation mappings that communicate the information between different solution spaces and varying fidelities.

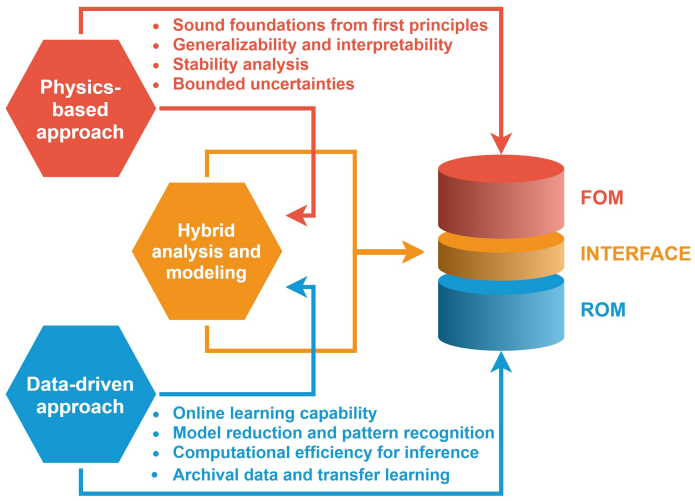


Figure 1.7: Hybrid analysis and modeling as a key enabler for FOM-ROM coupling problems toward predictive digital twins.

1.4 Dynamical Systems Are Evolving

Surrogate models embedded in DT frameworks are often parameterized using pre-recorded data or prior knowledge about the system. However, fluid flow systems evolve continuously on high-dimensional nonlinear solution manifold, spanning a wide range of flow regimes and operating conditions. Unless the system exhibits a limit cycle behavior or quasi-steady state dynamics, it is highly possible that a model calibrated at some conditions becomes invalid as the system evolves to other conditions. Therefore, the reliability of the DT is highly dependent on the ability of the underlying models to efficiently self-adapt to new situations. With measurements and observational data being continuously collected, efficient model-data fusion are desired to maintain the correspondence and matching between the physical system and its digital counterpart. Sensor are often sparse and subject to external disruptions (e.g., noise). In addition, we might not be able to measure the model's state directly, but only a function of it. We extend data assimilation (DA) and Bayesian inversion methodologies to update the model's parameters based on the live streams of sensor data. DA is a family of algorithms that combine dynamical models with observational data to account for missing processes in the governing equations, errors in model's parameters and incorrect initial and boundary conditions. DA has a long history of decades in numerical weather predictions and is being used on a daily basis in operational centers to make reliable forecasts. We refer interested readers to

our introductory article on popular DA techniques with step-by-step implementation tutorials [11].

In order to enable the efficient communication of DT with the physical system and augment the reliability of the resulting predictions, the following frameworks are developed:

- First, we employ a variational approach based on the forward sensitivity methodology to improve the surrogate model's prediction by correcting its parameters. In particular, we utilize physical arguments based on energy transfer analysis to define scale-aware closure models that counteract the effect of ROM truncation. Streams of sparse and noisy measurements are then assimilated in a hybrid model-data fusion framework to dynamically update components of these closure models.

- Second, we design a deep learning methodology that complement classical nudging approaches to estimate the discrepancy between the model's forecast and the target values. In particular, an LSTM-Nudging approach is implemented to correct the GROM predictions using new observational data that are sparse both in space and time. This framework collectively addresses the model closure problem, the uncertainty in prior model's structure and initialization, and the measurement noise.

1.5 Dissertation Outline

The rest of this dissertation is split into three main parts. Part I is dedicated to building accurate surrogate using reduced order modeling methodologies to fulfill the multi-query and real-time response requirements of DTs. In Chapter 2, we develop the uplifted reduced order model (UROM) framework to learn the discrepancy between Galerkin ROM predictions and their optimal values that yield the lowest reduced order representation (ROR) error. In addition, the uplifting component of UROM aims at using the relationships between different ROM scales to recover some of the small scales in the reconstructed flow fields. In Chapter 3, we propose a hybrid variational multi-scale (VMS) framework that leverage the physics guided machine learning (PGML) algorithm to construct novel and robust models for the interaction among the ROM scales. Moreover, we describe the nonlinear proper orthogonal decomposition (NLPOD) methodology to learn a compressed representation of the unresolved

scales to reduce the projection error and improve the flow field predictions.

In Part II, we put forth the interface learning (IL) paradigm to account for different modeling and data interfaces that appear among computational units. In Chapter 4, we motivate the need for IL with a discussion on potential areas or applications that can benefit from the IL methodologies. Demonstrations for non-iterative domain decomposition are provided for a set of flow problems governed by hyperbolic dynamics. We extend the IL for the efficient coupling of FOMs and ROMs in Chapter 5, with application to multi-component and multi-physics problems.

The adaptability of DT to evolving flow regimes is addressed in Part III. In particular, Chapter 6 shows how the stream of sparse and noisy measurements can be embedded in a forward sensitivity analysis framework to learn and update scale-aware closure terms. In addition, the combination of machine learning capabilities with data assimilation algorithms is depicted in Chapter 7 where an LSTM-Nudging framework is developed for the wake-vortex transport and decay problem. Concluding remarks and outlook regarding the proposed hybrid physics-based and data-driven frameworks to enable next-generation of digital twins are drawn in Chapter 8.

Part I

Reduced Order Modeling and Variational Multi-Scale Framework

CHAPTER 2

A Long Short-Term Memory Embedding for Hybrid Uplifted Reduced Order Models

2.1 Abstract

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

2.2 Introduction

Physical models are often sought because of their reliability, interpretability, and generalizability being derived from basic principles and physical intuition. However,

This chapter is adapted from: *Ahmed, S. E., San, O., Rasheed, A., & Iliescu, T. (2020). A long short-term memory embedding for hybrid uplifted reduced order models. Physica D: Nonlinear Phenomena, 409, 132471.*

accurate solution of these models for complex systems usually requires the use of very high spatial and temporal resolutions and/or sophisticated discretization techniques. This limits their applications to offline simulations over a few set of parameters and short time intervals since they can be excessively computationally-demanding. Although those are valuable in understanding physical phenomena and gaining more insight, realistic applications often require near real-time and multi-query responses [12, 13]. Therefore, cheaper numerical approximations using "adequate-fidelity" models are usually acceptable [14]. In this regard, reduced order modeling offers a viable technique to address systems characterized by underlying patterns [15–23]. This is especially true for fluid flows dominated by coherent structures (e.g., atmospheric and oceanic flows) [24–32].

Reduced order models (ROMs) have shown great success for prototypical problems in different fields. In particular, Galerkin projection (GP) coupled with the capability of proper orthogonal decomposition (POD) to extract the most energetic modes has been used to build ROMs for linear and nonlinear systems [33–40]. These ROMs preserve sufficient interpretability and generalizability since they are constructed by projecting the full order model (FOM) operators (from governing equations) on a reduced subspace. Despite that, Galerkin projection ROMs (GROMs) have severe limitations in practice, especially for systems with strong nonlinearity. Most fluid flows exhibit quadratic nonlinearity, which makes the computational cost of the resulting GROMs $\sim O(R^3)$, where R is the number of retained modes in the ROM approximation. As a result, R should be kept as low as possible (e.g., $O(5)$) through modal truncation for practical purposes; however, this has two main consequences. First, the solution is enforced to live in a smaller subspace which might not contain enough information to accurately represent complex realistic systems. Examples include advection-dominated flows and parametric systems where the decay of Kolmogorov n-width is slow [41–43]. Second, due to the inherent system's nonlinearity, the truncated modes interact with the retained modes. In GROM, these interactions are simply eliminated by modal truncation, which often generates instabilities in the approximation [44–46]. Several efforts have been devoted to introduce stabilization and closure techniques [47–64] to account for the effects of truncated modes on ROM's dynamics.

In the present study, we aim to address the above problems while preserving considerable interpretability, and generalizability at the core of our uplifted ROM (UROM) approach. In UROM, we present a three-modeling layer framework. In the

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

In particular, we choose the first $R \approx O(5)$ modes to represent the resolved largest scales and the next $Q - R$ modes to represent the resolved smaller scales, where Q is about 4 to 8 times larger than R. For the second and third layers, we incorporate memory embedding through the use of long short-term memory (LSTM) neural network architecture [65, 66]. Machine learning (ML) tools (of which neural networks is a subclass) have been gaining popularity in fluid mechanics community and analysis of dynamical systems [67–75]. In particular, LSTMs have shown great success in learning maps of sequential data and time-series [76–81]. It should be noted here that UROM can be thought of as a way of augmenting physical models with data-driven tools [82] and vice versa. For the former, an LSTM closure model (second component) is developed to correct GROM and an LSTM super-resolution model (third component) is constructed to uplift GROM and resolve smaller scales. This relaxes the computational cost of GROM to account only for a few R modes. On the other hand, LSTMs in UROM framework are fed with inputs coming from a physics-based approach. This is one way of utilizing physical information rather than full dependence on ML results.

To illustrate the UROM framework, we consider two convection-dominated flows as test cases. The first is the one-dimensional (1D) Burgers equation, which is a simplified benchmark problem for fluid flows with strong nonlinearity. As the second test case, we investigate the two-dimensional (2D) Navier-Stokes equations for a flow with interacting vortices, namely the vortex merger problem. We compare the UROM approach with the standard GROM approach using R and Q modes. We also investigate a fully non-intrusive ROM (NIROM) approach in these flow problems. We perform a comparison in terms of solution accuracy and computational

time to show the pros and cons of UROM with respect to either GROM or NIROM approaches. The rest of the chaptre is outlined here. In Section 2.3, we introduce the POD technique for data compression and constructing lower-dimensional subspaces to approximate the solution. For an out-of-sample control parameter (e.g., Reynolds number), we describe basis interpolation via a Grassmann manifold approach in Section 2.4. As a standard physics-informed technique for building ROMs, we introduce Galerkin projection in Section 2.5 with a brief description of the governing equations of the test cases as well as their corresponding GROM structures. In Section 2.6, we present the proposed UROM framework with a description of its main features. We give results and corresponding discussions in Section 2.7. Finally, we draw concluding remarks and insights in Section 2.8.

2.3 Proper Orthogonal Decomposition

Proper orthogonal decomposition (POD) is one of the most popular techniques for dimensionality reduction and data compression [83, 84, 24, 85, 86]. Given datasets, POD provides a linear subspace that minimizes the projection error between the true data and its projection compared to all possible linear subspaces with the same dimension. If a number of N_s data snapshots, $\mathbf{u}(\mathbf{x}, t_n)$, where $n \in \{1, 2, \ldots, N_s\}$, $\mathbf{x} \in \mathbb{R}^N$ (N being the spatial resolution), are collected in a snapshot matrix $\mathbf{A} \in \mathbb{R}^{N \times N_s}$ (where N is much larger than N_s for typical flow problems), then a reduced (or thin) singular value decomposition (SVD) can be applied to this matrix as

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{2.1}$$

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

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

$\widehat{\mathbf{A}}$ of $\mathbf{A}$ as

$$\widehat{\mathbf{A}} = \widehat{\mathbf{U}}\widehat{\mathbf{\Sigma}}\widehat{\mathbf{V}}^T. \tag{2.2}$$

It can be easily shown that this approximation $\widehat{\mathbf{A}}$ satisfies the following infimum [87]

$$\|\mathbf{A} - \widehat{\mathbf{A}}\|_2 = \inf_{\substack{\mathbf{B}\in\mathbb{R}^{N\times N_s} \\ rank(\mathbf{B})\leq R}} \|\mathbf{A} - \mathbf{B}\|_2 \tag{2.3}$$

$$\|\mathbf{A} - \widehat{\mathbf{A}}\|_2 = \sigma_{R+1}, \tag{2.4}$$

where $\|\cdot\|_2$ refers to the matrix 2-norm. Equation (2.3) means that across all possible matrices $\mathbf{B} \in \mathbb{R}^{N\times N_s}$ of rank R (or less), $\widehat{\mathbf{A}}$ provides the closest one to $\mathbf{A}$ in the ℓ_2 sense. Moreover, the singular values $\{\sigma_i\}$ provide a measure of the quality of this approximation as Eq. (2.4) shows that the ℓ_2 norm between the matrix $\mathbf{A}$ and its R-rank approximation equals to σ_{R+1}. From now on, the first R columns of $\mathbf{U}$ will be referred to as the POD modes or basis function, denoted as $\mathbf{\Phi} = [\phi_1, \phi_2, \ldots, \phi_R]$.

2.4 Grassmann Manifold Interpolation

In recent years, Grassmann manifold has attracted great interest in various applications including model order reduction for parametric systems [88–92]. The Grassmann manifold, $\mathcal{G}(q, N)$, is a set of all q-dimensional subspaces in $\mathbb{R}^N$, where $0 \leq q \leq N$. A point $[\mathbf{\Phi}] \in \mathcal{G}(q, N)$ is given as [93]

$$[\mathbf{\Phi}] = \{\mathbf{\Phi}Q \;\big|\; \mathbf{\Phi}^T\mathbf{\Phi} = I_q, Q \in \mathcal{O}(q)\}, \tag{2.5}$$

where $\mathbf{\Phi} \in \mathbb{R}^{N\times q}$ and $\mathcal{O}(q)$ is the group of all $q \times q$ orthogonal matrices. This point represents a q-dimensional subspace $\mathcal{S}$ in $\mathbb{R}^N$ spanned by the columns of $\mathbf{\Phi}$. At each point $[\mathbf{\Phi}] \in \mathcal{G}(q, N)$, a tangent space $\mathcal{T}([\mathbf{\Phi}])$ of the same dimension, $N \times q$, can be defined as follows [94, 95]

$$\mathcal{T}([\mathbf{\Phi}]) = \{\mathcal{X} \in \mathbb{R}^{N\times q} \;\big|\; \mathbf{\Phi}^T\mathcal{X} = \mathbf{0}\}. \tag{2.6}$$

Similarly, each point $[\mathbf{\Gamma}]$ on $\mathcal{T}$ represents a subspace spanned by the columns of $\mathbf{\Gamma}$. This tangent space is a vector space with its origin at $[\mathbf{\Phi}]$. An exponential mapping

from a point $[\mathbf{\Gamma}] \in \mathcal{T}([\mathbf{\Phi}])$ to $[\mathbf{\Psi}] \in \mathcal{G}(q, N)$ can be defined as

$$\mathbf{\Psi} = \Big(\mathbf{\Phi}\mathbf{V}\cos(\mathbf{\Sigma}) + \mathbf{U}\sin(\mathbf{\Sigma})\Big)\mathbf{V}^T, \tag{2.7}$$

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

$$\mathbf{\Gamma} = \mathbf{U}\tan^{-1}(\mathbf{\Sigma})\mathbf{V}^T, \tag{2.8}$$

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

To demonstrate the procedure in our ROM context, for a number N_p of control parameters $\{\mu_i\}_{i=1}^{N_p}$, different sets of POD basis functions are computed corresponding to each parameter, denoted as $\{\mathbf{\Phi}_i\}_{i=1}^{N_p}$. These bases correspond to a set of points on the Grassmann manifold $\mathcal{G}(R, N)$. To perform an out-of-sample testing, the basis functions $\mathbf{\Phi}_{\text{Test}}$ for the test parameter μ_{Test} should be computed through interpolation. However, direct interpolation of the POD bases is not effective since it is an interpolation on a non-flat space and it does not guarantee that the resulting point would lie on $\mathcal{G}(R, N)$. Moreover, the optimality and orthonormality characteristics of POD are not necessarily conserved. Alternatively, the tangent space $\mathcal{T}$ is a flat space where standard interpolation can be performed effectively. First, a reference point at the Grassmann manifold is selected, corresponding to $\mathbf{\Phi}_{\text{Ref}}$. The tangent plane at this point is thus defined using Eq. (2.6). Then, the neighboring points on Grassmann manifold corresponding to the subspaces spanned by $\{\mathbf{\Phi}_i\}_{i=1}^{N_p}$ are mapped onto that tangent plane using the logarithmic map, defined in Eq. (2.8) to calculate $\{\mathbf{\Gamma}_i\}_{i=1}^{N_p}$. Standard Lagrange interpolation can be performed to compute $\mathbf{\Gamma}_{\text{Test}}$ as follows

$$\mathbf{\Gamma}_{\text{Test}} = \sum_{i=1}^{N_p}\Bigg(\prod_{\substack{j=1\\ j\neq i}}^{N_p}\frac{\mu_{\text{Test}} - \mu_j}{\mu_i - \mu_j}\Bigg)\mathbf{\Gamma}_i. \tag{2.9}$$

Finally, the point $[\mathbf{\Gamma}_{\text{Test}}] \in \mathcal{T}([\mathbf{\Phi}_{\text{Ref}}])$ is mapped to the Grassmann manifold $\mathcal{G}(R, N)$ to obtain the set of POD basis functions at the test parameter, $\mathbf{\Phi}_{\text{Test}}$, using the exponential map given in Eq. (2.7). Hence, an interpolation on the tangent plane to Grassmann manifold provides a basis of the same dimension (i.e., $[\mathbf{\Phi}_{\text{Test}}] \in \mathcal{G}(R, N)$).

Moreover, it preserves the orthonormality of the basis (i.e., the columns of $\mathbf{\Phi}_{\text{Test}}$ are orthonormal to each other). Those properties are not guaranteed if conventional interpolation techniques are used directly to inpterpolate basis.

2.5 Galerkin Projection

To emulate the system's dynamics in ROM context, a Galerkin projection is usually performed. In Galerkin projection-based ROM (GROM), the solution $\mathbf{u}(\mathbf{x}, t_n)$ is constrained to lie in a trial subspace $\mathcal{S}$ spanned by the basis $\mathbf{\Phi}$. In our study, this basis is computed using the POD method presented in Section 2.3. Then, the full-order operators are projected onto the same subspace $\mathcal{S}$. In other words, the residual of the governing ODE is enforced to be orthogonal to $\mathcal{S}$. Galerkin projection can be viewed as a special case of Petrov-Galerkin method [96–99], by utilizing the same trial subspace as a test subspace. In the following, we present the governing equations for our test cases, namely Burgers equation and Navier-Stokes equations as well as their low-order approximations.

2.5.1 1D Burgers equation

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

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{1}{\text{Re}}\frac{\partial^2 u}{\partial x^2}, \tag{2.10}$$

where Re is the dimensionless Reynolds number, defined as the ratio of inertial effects to viscous effects. Equation (2.10) can be rewritten as

$$\frac{\partial u}{\partial t} = \frac{1}{\text{Re}}\frac{\partial^2 u}{\partial x^2} - u\frac{\partial u}{\partial x}. \tag{2.11}$$

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

the following dynamical ODE, which represents the GROM for the Burgers equation

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \sum_{i=1}^{R} \mathfrak{L}_{i,k} a_i + \sum_{i=1}^{R}\sum_{j=1}^{R} \mathfrak{N}_{i,j,k} a_i a_j, \qquad k = 1, 2, \ldots, R, \tag{2.12}$$

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

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

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

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

2.5.2 2D Navier-Stokes equations

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

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \nabla^2 \omega, \tag{2.13}$$

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

$$\nabla^2 \psi = -\omega. \tag{2.14}$$

Equations (2.13) and (2.14) include two operators, the Jacobian ($J(f,g)$) and the Laplacian ($\nabla^2 f$) defined as

$$J(f,g) = \frac{\partial f}{\partial x}\frac{\partial g}{\partial y} - \frac{\partial f}{\partial y}\frac{\partial g}{\partial x}, \tag{2.15}$$

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \tag{2.16}$$

Similar to the 1D Burgers problem, Eq. (2.13) can be rearranged as

$$\frac{\partial \omega}{\partial t} = \frac{1}{\mathrm{Re}}\nabla^2 \omega - J(\omega, \psi). \tag{2.17}$$

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

$$\omega(x,y,t) \approx \sum_{k=1}^{R} a_k(t)\phi_k^{\omega}(x,y), \tag{2.18}$$

$$\psi(x,y,t) \approx \sum_{k=1}^{R} a_k(t)\phi_k^{\psi}(x,y). \tag{2.19}$$

We note that the vorticity and streamfunction share the same time-dependent coefficients ($a_k(t)$) since they are related through the kinematic relationship given by Eq. (2.14) (i.e., streamfunction is not a prognostic variable). Moreover, as POD preserves linear properties, the spatial POD modes for streamfunction can be obtained from the vorticity modes by solving the following Poisson equations

$$\nabla^2 \phi_k^{\psi}(x,y) = -\phi_k^{\omega}(x,y), \quad k = 1, 2, \ldots, R. \tag{2.20}$$

The GROM for the 2D Navier-Stokes equations is given by the same ODE (Eq. (2.12)) with the following coefficients

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

Similar to the case of 1D Burgers problem, the basis functions are spatial function and standard discretization techniques can be used to compute the required derivatives. Moreover, they might be arranged in a vector form to compute the inner product in

Euclidean space. In 2D case, a reshaping might be necessary to form the vector version of $\phi_k(x, y)$, e.g., $[\phi_k(x_1, y_1), \phi_k(x_2, y_1), \ldots \phi_k(x_{N_x}, y_1), \phi_k(x_1, y_2), \phi_k(x_2, y_2), \ldots, \phi_k(x_{N_x}, y_2), \ldots, \phi_k(x_1, y_{N_y}), \ldots, \phi_k(x_{N_x}, y_{N_y})]^T$, where N_x and N_y are the number of grid points in the x and y directions, respectively.

Due to the modal truncation and inherent nonlinearity in Eq. (2.12), GROM no longer represents the same system (i.e., it solves a different problem). As a result, the obtained trajectory from solving the ROM deviates from the projected trajectory, as shown in Fig. 2.1. Therefore, the optimality of the POD basis is lost. Moreover, due to the triadic nonlinear interactions, instabilities can occur in GROMs. To mitigate these problems, closure and/or stabilization techniques are usually required to obtain accurate results. Increasing the ROM dimension can improve the results. However, due to the nonlinearity of the resulting ROM, the computational cost of GROM is $O(R^3)$, which severely constrains the ROM dimension used in practical applications.

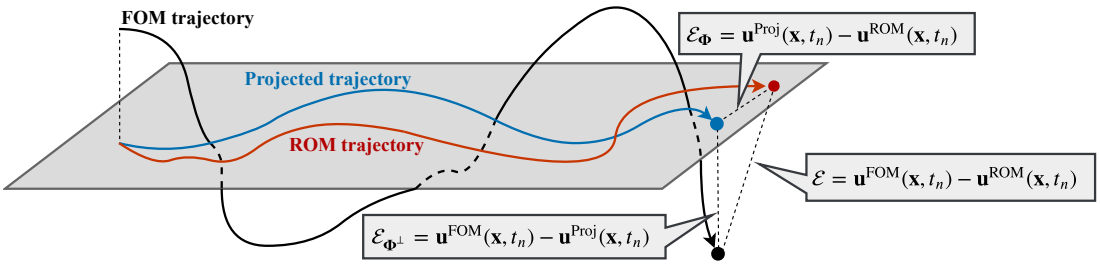


Figure 2.1: A representation of error sources in Galerkin ROM (e.g., see [86, 100] for further details).

We note here that we are adopting the tensorial GROM approach [101], where the coefficients $\mathfrak{L}$ and $\mathfrak{N}$ are computed offline. Other approaches can include online computations while incorporating the empirical interpolation method (EIM) [102] or its discrete version, the discrete empirical interpolation method (DEIM) [103] to reduce the online computational cost.

Although, we focus on the standard GROM with POD in the present study, several studies have been devoted to address the computational cost and stability/accuracy trade-off for advection-dominated flows. For example, decomposing the time domain via a principal interval decomposition approach can produce localized basis functions and tailor more representative compact subspaces [104, 105], which improves the ROM accuracy and keeps the online computational cost minimal. Rather than being restricted to a linear basis, auto-encoders can be used to compute nonlinear subspaces to approximate the solution manifold [106, 107]. Also, Grimberg et al. [108] recently

demonstrated that many ROM instabilities can be attributed to the standard Galerkin projection, and the Petrov-Galerkin approach can mitigate these instabilities and provide more accurate ROM [96–99].

2.6 Uplifted Reduced Order Modeling

As noted in Section 2.5, the computational cost of GROM is $O(R^3)$, which limits the number of modes to be used in the ROM. This modal truncation has two major consequences. First, the flow field variable is constrained to lie in a small subspace, spanned by the very first few modes. For convection-dominated flows or parametric problems characterized by slow decay of the Kolmogorov n-width, these few modes may be less representative of the true physical system, which might significantly reduce the accuracy of the resulting ROM. This is shown as the projection error $\mathcal{E}_{\mathbf{\Phi}^\perp}$ in Fig. 2.1, since the truncated modes are orthogonal to the subspace spanned by $\mathbf{\Phi}$. Second, due to the inherent nonlinearity, the truncated modes (or scales) interact with the retained ones. Thus, this truncation simply ignores these interactions, often giving rise to numerical instabilities of solution. This error is represented as $\mathcal{E}_{\mathbf{\Phi}}$ in Fig. 2.1 since it lies in the same subspace $\mathbf{\Phi}$. In our uplifted reduced order modeling (UROM) framework (presented in Fig. 2.2), we try to address these two problems.

We extend our reduced-order approximation to include the first Q modes, where $Q > R$, assuming that the first R modes account for the resolved large scales, and the next $(Q-R)$ modes represent the resolved small scales (while the remaining truncated modes account for the unresolved scales). So, the UROM approximation of the true field $\mathbf{u}(\mathbf{x}, t_n)$ can be expanded as

$$\mathbf{u}(\mathbf{x}, t_n) \approx \underbrace{\sum_{k=1}^{R} a_k(t)\phi_k(\mathbf{x})}_{\substack{\text{core} \\ \text{(resolved large scales)}}} + \underbrace{\sum_{k=R+1}^{Q} a_k(t)\phi_k(\mathbf{x})}_{\substack{\text{uplift} \\ \text{(resolved small scales)}}}, \tag{2.21}$$

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

$$a_k(t) = \langle \mathbf{u}(\mathbf{x}, t); \phi_k(\mathbf{x}) \rangle. \tag{2.22}$$

Before presenting the UROM framework, we first briefly revisit the Galerkin ROM

(GROM). Here, we denote the GROM solution as $\tilde{a}_k(t_n)$, where the initial condition $\tilde{a}_k(t_0)$ can be computed from the projection of the initial field (at t_0) onto the POD modes (by using Eq. (2.22)). Then, GROM is used to evolve $\tilde{a}$ in time as

$$\frac{\mathrm{d}\tilde{a}_k}{\mathrm{d}t} = G(\tilde{a}_k), \tag{2.23}$$

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

$$\tilde{a}_k(t_{n+1}) = \tilde{a}_k(t_n) + \Delta t \sum_{q=0}^{s} \beta_q G(\tilde{a}_k(t_{n-q})), \tag{2.24}$$

where s and β_q depend upon the numerical scheme used for the time integration. In the present study, we use the third-order Adams-Bashforth (AB3) method, for which $s = 2,\ \beta_0 = 23/12,\ \beta_1 = -16/12$, and $\beta_2 = 5/12$. Here, $G(\tilde{a}_k)$ is obtained by Galerkin projection (e.g., see Section 2.5) as

$$G(\tilde{a}_k(t_n)) = \sum_{i=1}^{R} \mathfrak{L}_{i,k}\tilde{a}_i(t_n) + \sum_{i=1}^{R}\sum_{j=1}^{R} \mathfrak{N}_{i,j,k}\tilde{a}_i(t_n)\tilde{a}_j(t_n). \tag{2.25}$$

However, due to the modal truncation and incurred errors and instabilities of GROM (as discussed in Section 2.5), the resulting predictions $\tilde{a}_k$ from Eqs. (2.23) to (2.25) become erroneous.

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

$$\hat{a}_k(t_{n+1}) = a_k(t_n) + \Delta t \sum_{q=0}^{s} \beta_q G(a_k(t_{n-q})), \tag{2.26}$$

After $\hat{a}_k(t_{n+1})$ is computed from Eq. (2.26), a correction might be introduced before

evolving GROM to the next time step, i.e., $a_k(t_{n+1}) = \hat{a}_k(t_{n+1}) + c_k(t_{n+1})$, where c_k can be considered as the difference between the physics-based model estimate and the true projection. In other words, the corrected temporal coefficient is assumed to be the true value of a_k (or at least the best-known value), which is therefore used as input to GROM to evolve one time step further. That is why the corrected a_k values are used to compute the right hand side of Eq. (2.26).

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

$$c_k(t_n) = a_k(t_n) - \hat{a}_k(t_n). \tag{2.27}$$

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

$$f : \begin{bmatrix} \mu \\ \hat{a}_1(t_n) \\ \vdots \\ \hat{a}_R(t_n) \end{bmatrix} \mapsto \begin{bmatrix} c_1(t_n) \\ \vdots \\ c_R(t_n) \end{bmatrix}, \qquad g : \begin{bmatrix} \mu \\ a_1(t_n) \\ \vdots \\ a_R(t_n) \end{bmatrix} \mapsto \begin{bmatrix} a_{R+1}(t_n) \\ \vdots \\ a_Q(t_n) \end{bmatrix}. \tag{2.28}$$

In brief, we first steer the red line in Fig. 2.1 to the blue one (i.e., introduce data-driven closure by LSTM). Then, we reduce the projection error (difference between the the blue and black lines) by expanding our solution subspace to span Q modes rather than only R. Figure 2.2 demonstrates the building blocks and workflow of our proposed UROM framework in both the offline and online phases, which can be

described as follows.

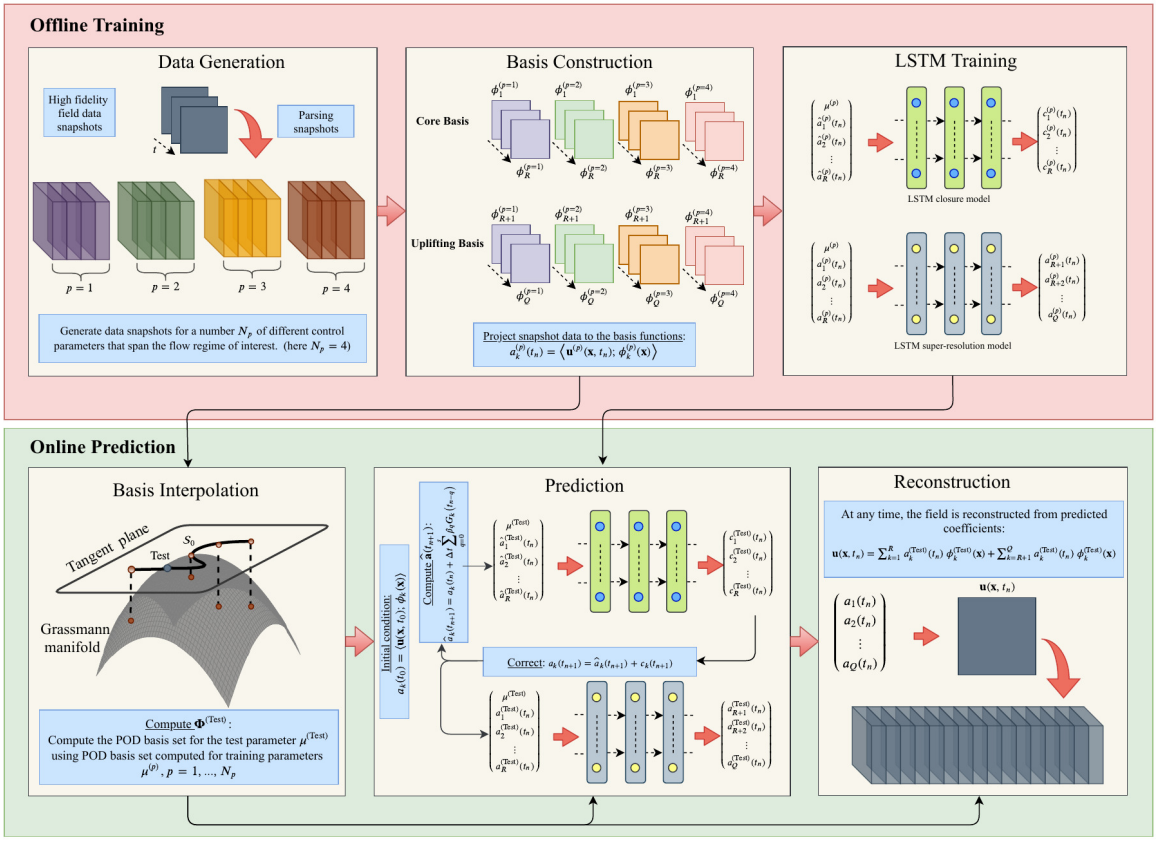


Figure 2.2: A schematic diagram for the workflow of UROM framework.

During offline training, we suppose that we have access to the true fields at different time instants (those can come from experiments or numerical simulations). Thus, we can obtain the true modal coefficients a_k (see Eq. (2.22)). Moreover, we use the Galerkin ROM equations to evolve the modal coefficients for one time step (in that sense, GROM can be viewed as a mapping from $a_k(t_n)$ to $\hat{a}_k(t_{n+1})$. Then, a correction term can be computed as the difference between $a_k(t_{n+1})$ and $\hat{a}_k(t_{n+1})$ for $k = 1, 2, \ldots, R$. Therefore, a corrector LSTM is trained to learn the mapping from $\hat{a}_k$ to c_k. Also, a super-resolver LSTM is trained to map a_k for $k = 1, 2, \ldots, R$ to a_k for $k = R+1, R+2, \ldots, Q$.

During online deployment (actual testing), we start with initial field (at time zero) and project it onto the first R modes, to obtain the true $a_k(t_0)$. Then, GROM is used to evolve these R coefficients for one time step to obtain $\hat{a}_k(t_1)$ for $k = 1, 2, \ldots, R$. At this point, the corrector LSTM is fed by $\hat{a}_k(t_1)$ to output $c_k(t_1)$, and the corrected

modal coefficients are computed as $a_k(t_1) = \hat{a}_k(t_1) + c_k(t_1)$ for $k = 1, 2, \ldots, R$. Those values are again fed to GROM to compute $\hat{a}_k(t_2)$, which are subsequently corrected by the corrector LSTM, and so on. Finally, before we reconstruct the full order field at any time instant t_n of interest, we utilize a super-resolver LSTM to recover the finer field scales. This super-resolver LSTM is fed with the corrected values $(a_k(t_n))$ for $k = 1, 2, \ldots, R$, representing the large scale dynamics resulting from the GROM and the LSTM corrector. Then, $a_k(t_n)$ for $k = R+1, R+2, \ldots, Q$, are obtained as output from this super-resolver. The workflow for online deployment is shown in Fig. 2.3. Note the recursive nature of the deployment, where the corrected values are fed back to GROM to advance one more time step. We also emphasize that the super-resolver is only used at instants of interest (i.e., it is not necessarily required at each time step), which makes further computational savings possible.

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

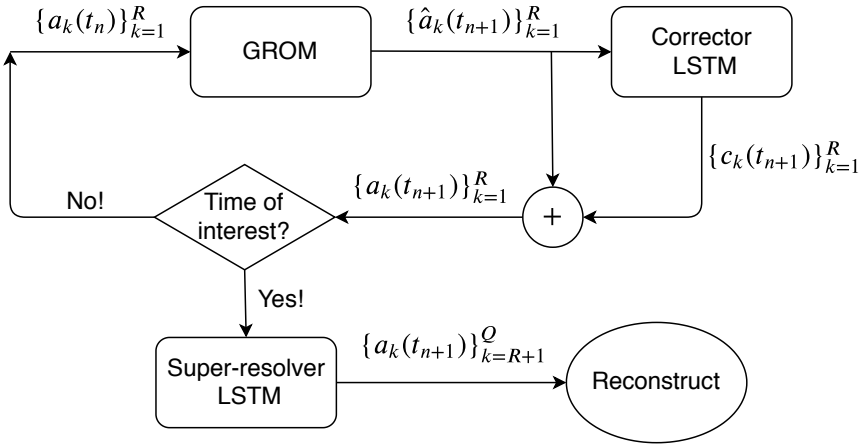


Figure 2.3: A schematic diagram for the online deployment of UROM approach. Note that $\{a_k(t_n)\}_{k=1}^R$ is a short-hand notation for $a_1(t_n), a_2(t_n), \ldots, a_R(t_n)$.

- The physics-constrained GROM is maintained to account for the large scales. This enriches the framework interpretability and generalizability across a wide range of control parameters.
- GROM acts on a few modes, minimizing the online computational cost (i.e., $O(R^3)$, where $R < Q$).
- GROM, being physics-informed, can be used as a sanity check to decide whether or not the LSTM predictions should be considered.

- Data-driven closure/correction encapsulates information from *all* interacting modes and mechanisms.
- Since both LSTMs are fed with input from a physics-based approach, UROM can be considered as a way of enforcing physical knowledge to enhance data-driven tools.
- LSTMs' inputs are augmented with the control parameter to provide a more accurate mapping (sometimes also called physics-guided mapping).

2.6.1 Long short-term memory embedding

To learn the maps f and g in UROM, we incorporate memory embedding through the use of LSTM architecture. LSTM is a variant of recurrent neural networks capable of learning and predicting the temporal dependencies between given data sequences based on the input information and previously acquired information. Recurrent neural networks have been used successfully in ROM community to enhance standard projection ROMs [109] and build fully non-intrusive ROMs [110–112, 104, 113, 114]. In the present study, we use LSTMs to augment the standard physics-informed ROM by introducing closure as well as super-resolution data-driven models. We utilize Keras API [115] to build the LSTMs used in our UROM approach. Details about the LSTM architecture can be found in [104, 112]. A summary of the adopted hyperparameters is presented in Table 2.1. We also found that the constructed neural networks are not very sensitive to hyperparameters. Meanwhile, for optimal hyperparameter selection, different techniques (e.g., gridsearch) can be used to tune them.

2.7 Results

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

Table 2.1: A list of hyperparameters utilized to train the LSTM network for all numerical experiments.

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

'True' in our results), where

$$a_k^{\text{True}}(t_n) = \big\langle \mathbf{u}(\mathbf{x}, t_n); \phi_k(\mathbf{x}) \big\rangle, \tag{2.29}$$

$$\mathbf{u}^{\text{True}}(\mathbf{x}, t_n) = \sum_{k=1}^{Q} a_k^{\text{True}}(t_n)\phi_k(\mathbf{x}). \tag{2.30}$$

Since UROM can be considered as a hybrid approach between fully intrusive and fully non-intrusive ROMs, we compare it with standard Galerkin projection ROM with 4 and 16 modes, denoted as GROM(4) and GROM(16), respectively. Moreover, we show the results of a fully non-intrusive ROM approach using 16 modes (denoted as NIROM). For this NIROM, we use the same LSTM architecture presented in Section 2.6.1 as a time-stepping integrator. In particular, we learn a map between the values of modal coefficients at current time step and their values at the following time step. Also, we augment our input with the control parameter to enhance the mapping accuracy. In other words, the NIROM map h can be represented as follows

$$h : \begin{bmatrix} \mu \\ a_1(t_n) \\ \vdots \\ a_Q(t_n) \end{bmatrix} \mapsto \begin{bmatrix} a_1(t_{n+1}) \\ \vdots \\ a_Q(t_{n+1}) \end{bmatrix}. \tag{2.31}$$

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

2.7.1 1D Burgers problem

For 1D Burgers simulation, we consider the initial condition [116]

$$u(x,0) = \frac{x}{1+\exp\left(\frac{\mathrm{Re}}{16}\left(4x^2-1\right)\right)}, \tag{2.32}$$

with $x \in [0,1]$. The 1D Burgers equation with the above initial condition and Dirichlet boundary conditions accepts the following analytic solution representing a traveling wave [116]

$$u(x,t) = \frac{\frac{x}{t+1}}{1+\sqrt{\frac{t+1}{t_0}}\exp\left(\mathrm{Re}\frac{x^2}{4t+4}\right)}, \tag{2.33}$$

where $t_0 = \exp(\mathrm{Re}/8)$. For offline training, we obtain solution for different Reynolds numbers ($\mathrm{Re} \in \{200, 400, 600, 800\}$). Data generation is performed using the analytic solution given in Eq. (2.33) after dividing the spatial domain $[0,1]$ into 1024 equally-spaced spatial intervals (i.e., $N_x = 1025$). For each case, we collect 1000 snapshots for $t \in [0,1]$ (i.e., $\Delta t = 0.001$). That is, a snapshot matrix of $\{\mathbf{u}(t_0), \mathbf{u}(t_2), \ldots, \mathbf{u}(t_{1000})\}$ is formed, where $\mathbf{u}(t_n)$ is the velocity field $u(x,t_n)$ collected as a column vector. Then, the POD basis functions are computed using the technique presented in Section 2.3. For online deployment, we obtain the POD basis at $\mathrm{Re} = 500$ and $\mathrm{Re} = 1000$ using Grassmann manifold interpolation as discussed in Section 2.4.

2.7.1.1 Re = 500: demonstrating interpolatory capability

A Reynolds number of 500 represents an interpolatory case, where we use the POD basis at $\mathrm{Re} = 600$ as our reference point for basis interpolation. The evolution of the first 4 POD temporal coefficients using different frameworks is shown in Fig. 2.4. It is clear that GROM(4) is incapable of capturing the true dynamics due to the severe modal truncation. On the other hand, both GROM(16) and UROM show very good results; however, GROM(16) is more computationally expensive as will be shown in

Section 2.7.3.

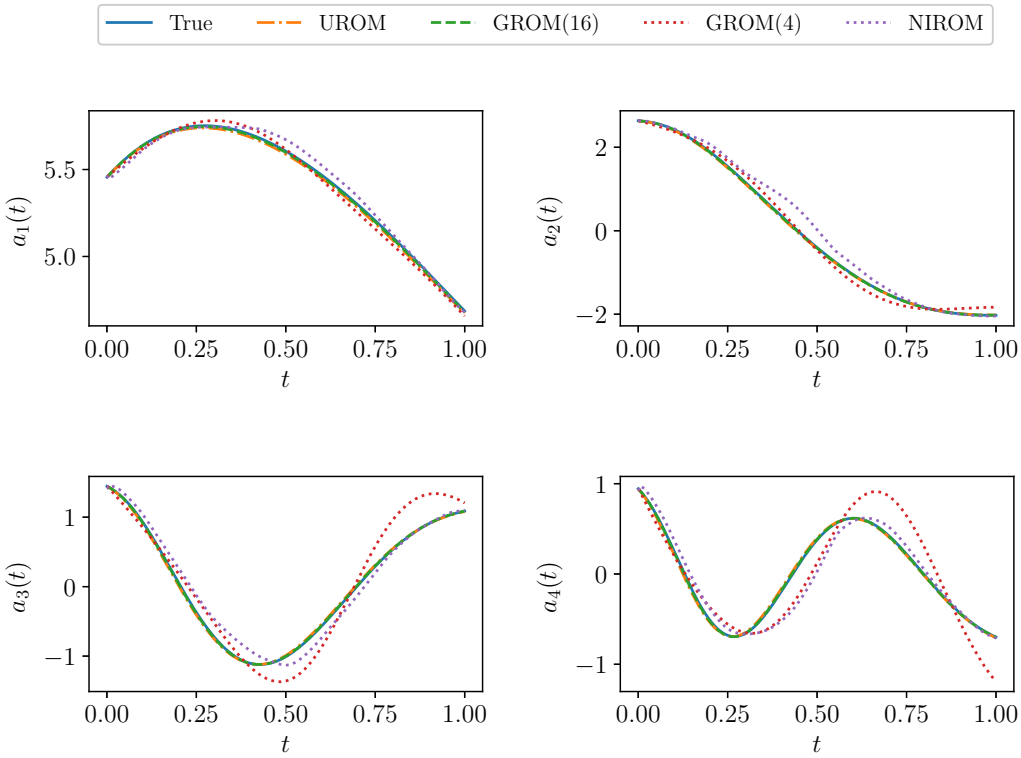


Figure 2.4: Temporal evolution of the first 4 POD modal coefficients for Burgers problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at $\mathrm{Re} = 500$. Note that GROM(4) and NIROM yield inaccurate results.

For field reconstruction, we present the temporal field evolution in Fig. 2.5 for FOM snapshots, true projection, UROM, GROM, and NIROM. It can be seen that UROM gives more accurate for field reconstruction than those predicted by GROM(4) and NIROM. For better visualizations, we show the velocity field at $t = 1$ in Fig. 2.6 with a close-up view on the region characterizing the wave structure.

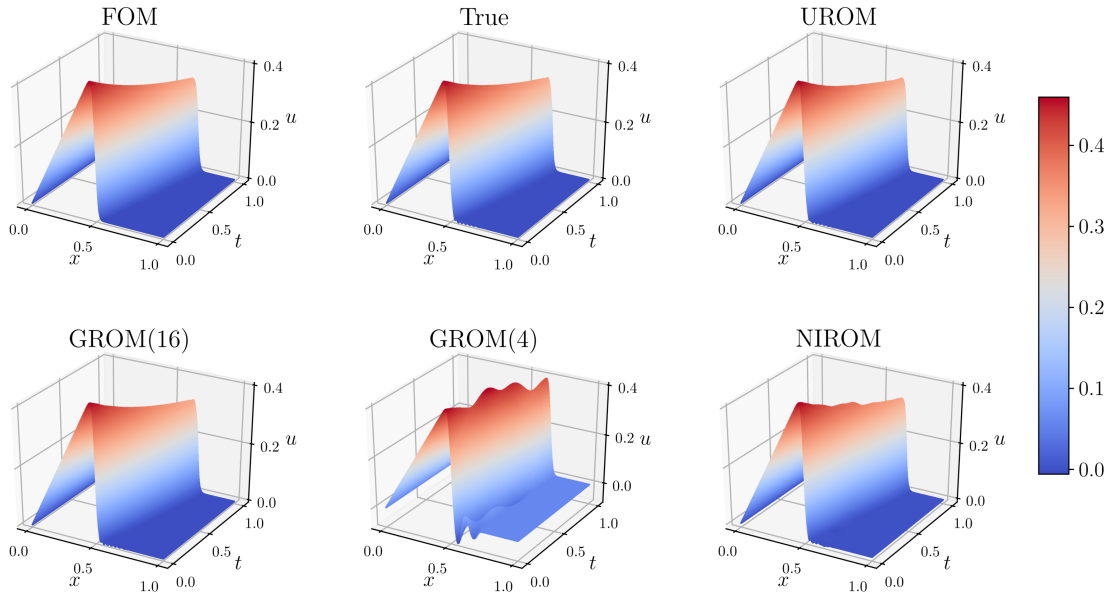


Figure 2.5: Temporal evolution of velocity fields for Burgers problem at Re = 500 with $R = 4$ and $Q = 16$.

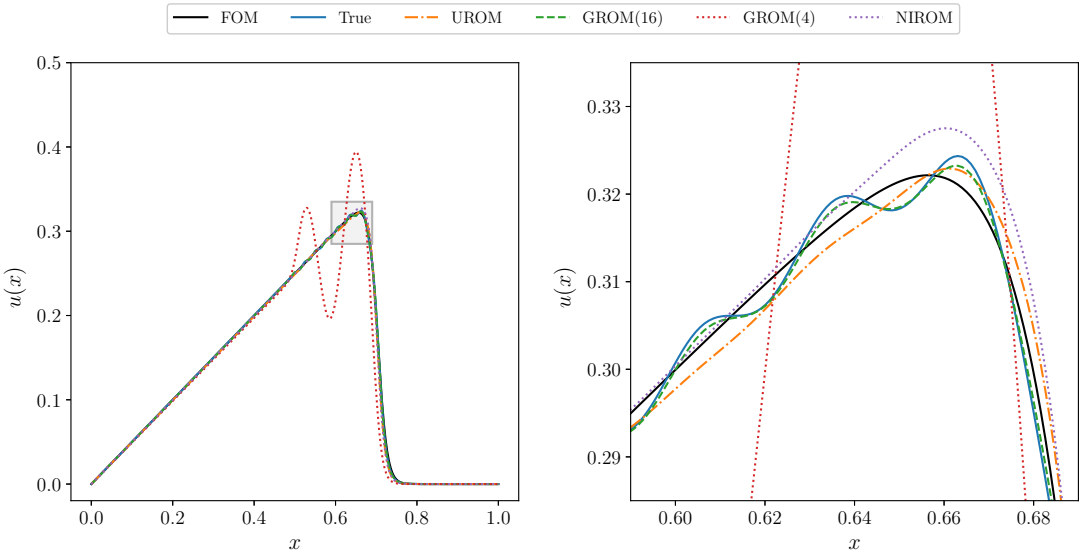


Figure 2.6: Final velocity fields (at $t = 1$) for Burgers problem at Re = 500 with a zoom-in view on the right using $R = 4$ and $Q = 16$. Note that UROM is giving smooth predictions while GROM(4) is showing significant oscillations.

2.7.1.2 Re = 1000: demonstrating extrapolatory capability

In order to investigate the exptrapolatory performance of UROM, we test the approach at Re = 1000, with the basis at Re = 800 as reference point for interpolation.

The POD modal coefficients are shown in Fig. 2.7, where we can see that both UROM and GROM(16) are still capable of capturing the true projected trajectory. Interestingly, the NIROM predictions are less satisfactory, giving non-physical behavior at some time instants. This suggests that the physical core of UROM promotes its generality, compared to the totally data-driven NIROM approach. However, we note that the deficient behavior of NIROM can be partly due to the sub-optimal architecture of our network as we only use the same hyperparameters (except for the size of input and output layers) for all simulations (as given in Table 2.1). More sophisticated architectures and further tuning of hyperparameters would probably improve NIROM predictions.

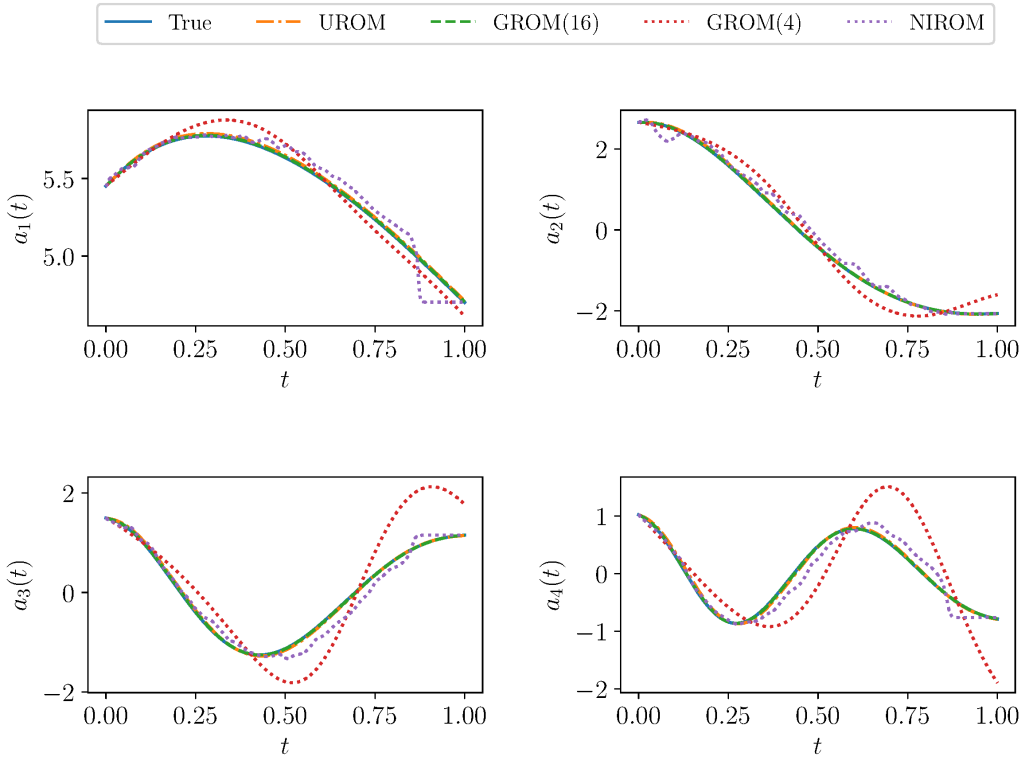


Figure 2.7: Temporal evolution of the first 4 POD modal coefficients for Burgers problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re = 1000. GROM(4) deviates from true trajectory while NIROM gives non-physical predictions.

The temporal field evolution of flow field is shown in Fig. 2.8, which illustrates the non-physical and unstable behavior of both GROM(4) and NIROM approaches. The final field is plotted in Fig. 2.9 with a close-up view on the right. It can be seen that even the true projected fields do not match the FOM and show some fluctuations at

the shock region. For this type of behavior, a larger subspace is required to capture most of the dynamics of the flow at Re = 1000.

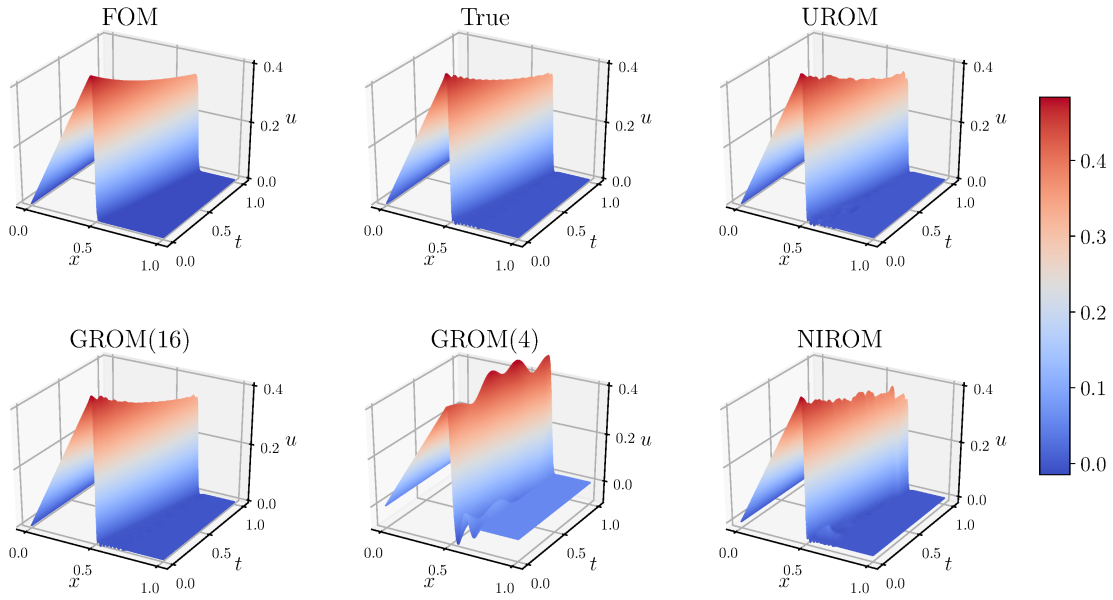


Figure 2.8: Temporal evolution of velocity fields for Burgers problem at Re = 1000 with $R = 4$ and $Q = 16$.

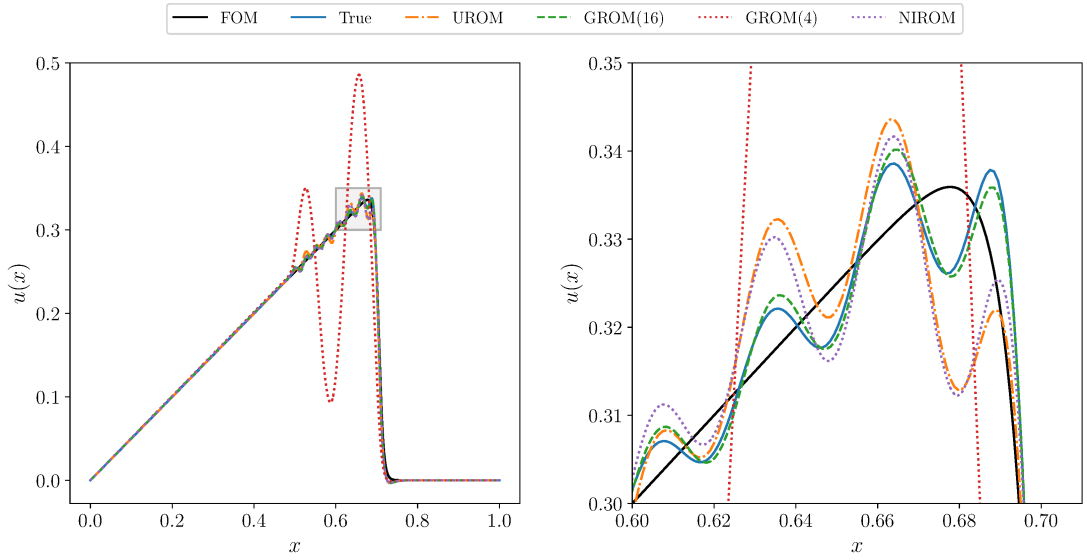


Figure 2.9: Final velocity fields (at $t = 1$) for Burgers problem at Re = 1000 with a zoom-in view on the right using $R = 4$, and $Q = 16$. Oscillations in UROM and GROM(16) occur mainly because a subspace spanning the first 16 modes is insufficient to capture the dynamics at this Reynolds number.

As indicated in the previous discussion, a more sophisticated architecture for LSTM and elegant tuning of the hyperparameters would be needed to get acceptable

performance for NIROM. It is quite common that NIROM suffers in long time predictions. In other words, during training and validation, the LSTM learns a map from the *true* modal coefficients to their values after one time step. Therefore, the network is always supposed to be fed with the true values. However, during actual deployment in the testing phase, the network is fed with true values *only* at the initial time (t_0). Then, the output of the network is returned back as input in a recursive manner to provide long time predictions. Thus, when the network encounters any error in the output (which is to be expected for testing at different parameters/region), this error is amplified in the subsequent time steps. Being fully non-intrusive, the network has no way to account for errors. As a result, after a few time steps, the output of the LSTM might blow-up giving non-physical results, unless the network is stabilized.

Figure 2.10 shows the predictions for NIROM at Re = 1000 with simple variations of the neural network architectures (different number of layers and neurons). We note that all of these combinations give a converging performance during training/validation, where the training and validation losses fall below 1×10^{-5}. On the other hand, they are not doing very well during actual testing, in the presence of numerical errors and instabilities. In that sense, a hybridization between physics-based and data-driven models helps to stabilize the predictions during the online deployment phase. In the rest of the chapter, we use the same architecture for NIROM and UROM, but the reader should be aware of these issues, which implies the need for the development of more involved architectures and/or more elegant training and validation.

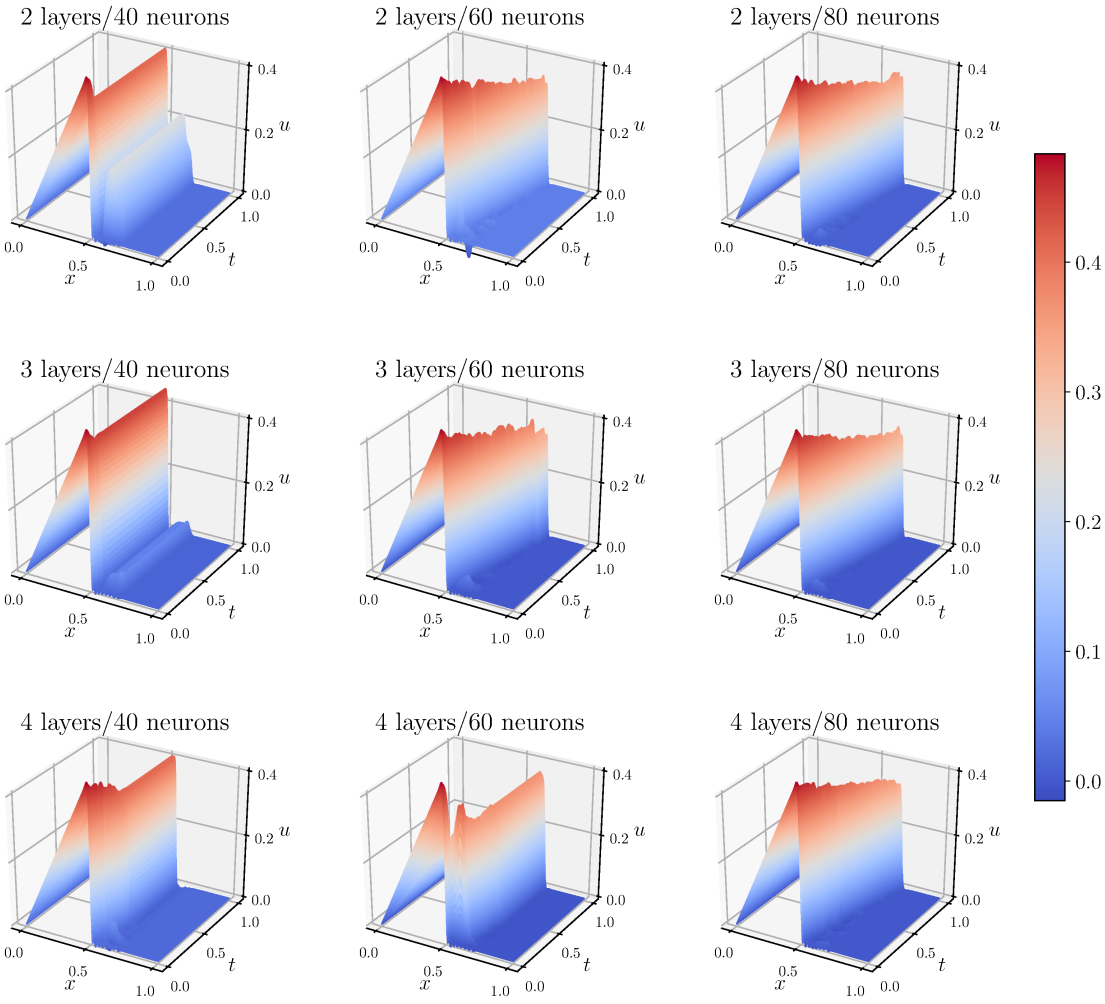


Figure 2.10: NIROM predictions for Burgers problem at Re = 1000 with $Q = 16$, and different numbers of layers and neurons.

2.7.2 2D vortex merger problem

As an application for 2D Navier-Stokes equations, we examine the vortex merger problem (i.e., the merging of co-rotating vortex pair) [117]. The merging process occurs when two vortices of the same sign with parallel axes are within a certain critical distance from each other, ending as a single, nearly axisymmetric, final vortex [118]. We consider an initial vorticity field of two Gaussian-distributed vortices with a unit circulation as follows,

$$\omega(x, y, 0) = \exp\left(-\rho\left[(x - x_1)^2 + (y - y_1)^2\right]\right) + \exp\left(-\rho\left[(x - x_2)^2 + (y - y_2)^2\right]\right), \tag{2.34}$$

where ρ is an interacting constant set as $\rho = \pi$ and the vortices centers are initially located at $(x_1, y_1) = \left(\frac{3\pi}{4}, \pi\right)$ and $(x_2, y_2) = \left(\frac{5\pi}{4}, \pi\right)$. We use a Cartesian domain $(x, y) \in [0, 2\pi] \times [0, 2\pi]$ over a spatial grid of 256×256, with periodic boundary conditions. For this 2D problem, we collect 200 snapshots for $t \in [0, 20]$, while varying Reynolds number as $\mathrm{Re} \in \{200, 400, 600, 800\}$. For solving the full order model equations, we use a third-order Arakawa scheme [119] for spatial derivatives, and a third-order total variation diminishing Runge–Kutta scheme (TVD-RK3) [120] for temporal integration. Similar to the 1D Burgers problem, we test our framework at $\mathrm{Re} = 500$ and $\mathrm{Re} = 1000$. Also, for basis interpolations, we use reference points at $\mathrm{Re} = 600$ and $\mathrm{Re} = 800$, respectively.

2.7.2.1 Re = 500: demonstrating interpolatory capability

Figure 2.11 shows the temporal evolution of the first 4 POD coefficients for the vorticity field. Recall that the temporal coefficients for vorticity and streamfunction fields are the same, as discussed in Section 2.5.2. We can see that both UROM and GROM(16) accurately predict the true modal dynamics. For better visualizations, the final vorticity field at $t = 20$ is given in Fig. 2.12, where GROM(4) is showing instabilities manifested in the reconstructed field. For this interpolatory case, NIROM is providing *acceptable* results.

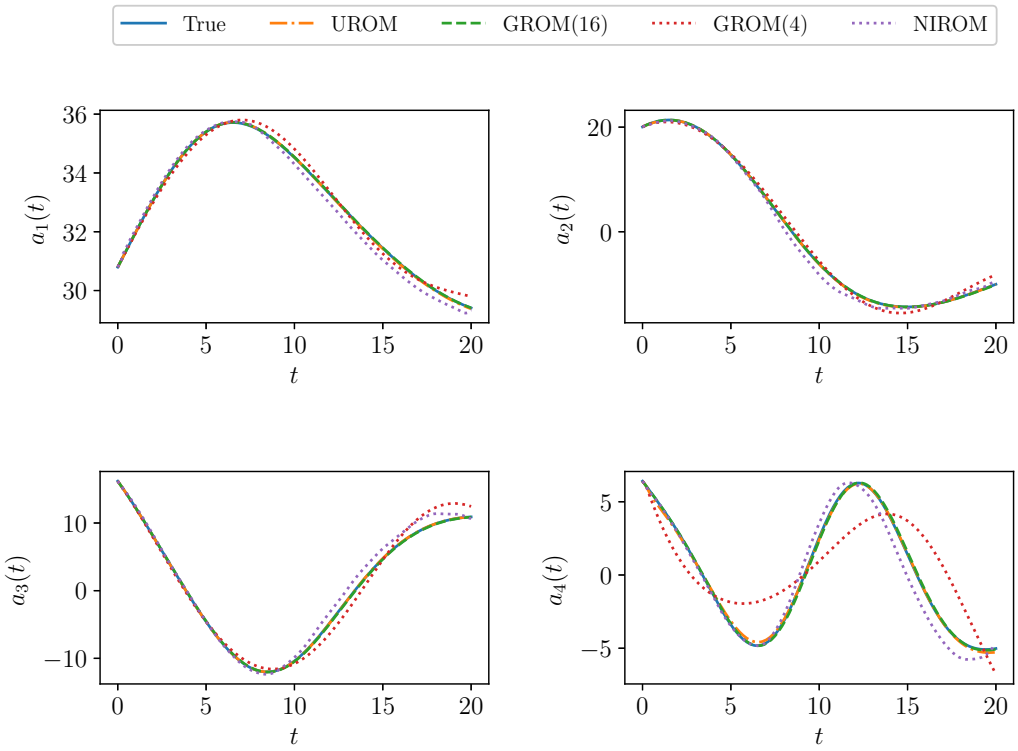


Figure 2.11: Temporal evolution of the first 4 POD modal coefficients of vorticity field for 2D vortex merger problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at $\mathrm{Re} = 500$.

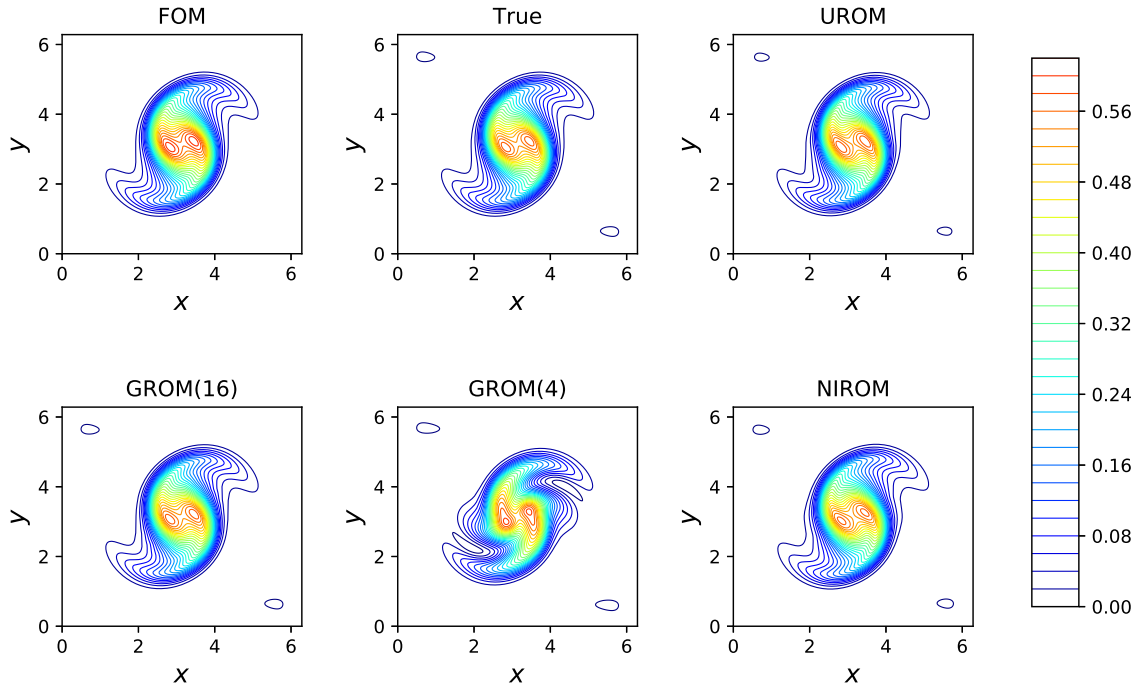


Figure 2.12: Final vorticity fields (at $t = 20$) for 2D vortex merger problem at $\mathrm{Re} = 500$ with $R = 4$, and $Q = 16$.

2.7.2.2 Re = 1000: demonstrating extrapolatory capability

The investigated approaches, namely GROM, UROM and NIROM, are tested at Re = 1000 as a case that is out-of-range compared to the training set. The POD modal coefficients predicted by these approaches are given in Fig. 2.13. It can be easily seen that as time increases, the predictions of GROM(4) and NIROM become poor. Using a neural network as time-stepping integrator in NIROM increases its sensitivity to computational noise and this recursive deployment accumulates the error until predictions totally depart from the true trajectory. This is even clearer in the reconstructed field shown in Fig. 2.14, where the orientation of the merging vortices is not matching the true orientation. GROM(4) prediction is also suffering from severe deformation of the true flow topology. On the other hand, the field reconstruction via UROM is accurate compared to the true projection and GROM(16).

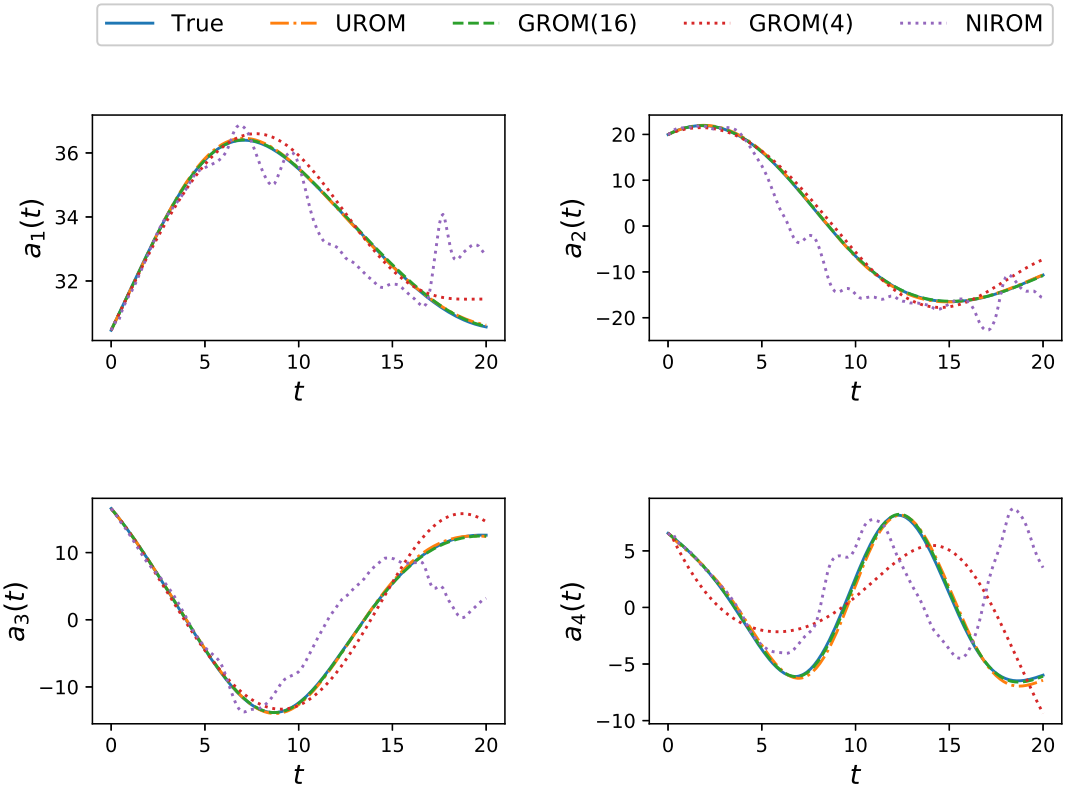


Figure 2.13: Temporal evolution of the first 4 POD modal coefficients of vorticity field for 2D vortex merger problem as predicted by UROM, GROM(4), GROM(16), and NIROM compared with the true values obtained by projection of FOM field on the interpolated modes at Re = 1000.

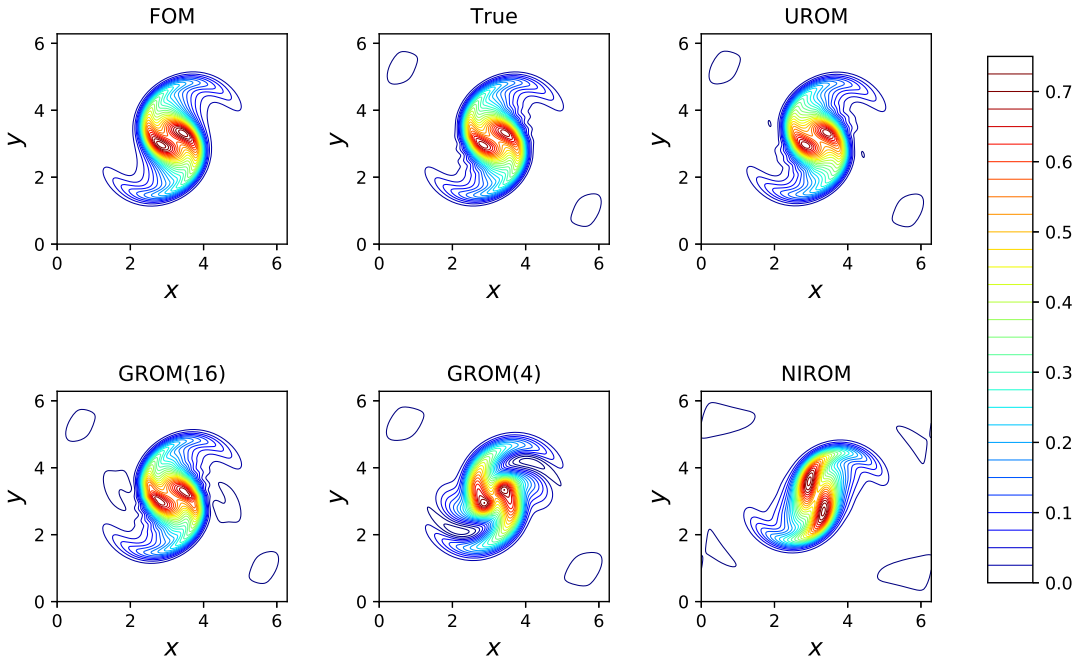


Figure 2.14: Final vorticity fields (at $t = 20$) for 2D vortex merger problem at $\mathrm{Re} = 1000$ with $R = 4$, and $Q = 16$.

2.7.3 Accuracy-computational efficiency trade-off

Finally, we report the "online" computing time for the investigated approaches. In particular, we show the computational time as well as the root-mean squared error (RMSE) of reconstructed fields at final time for the two test cases at $\mathrm{Re} = 1000$ in Table 2.2. The reported RMSE is computed as

$$\mathrm{RMSE}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{u}^{FOM}(\mathbf{x},t) - \mathbf{u}^{ROM}(\mathbf{x},t)\right)^2}, \tag{2.35}$$

where N represents the spatial resolution (i.e., $N = N_x \times N_y$). In this table, we also report the NIROM results using 4 modes in the input and output layers. Although GROM(4) is the fastest, its predictions are very poor and further corrections and stabilization might be required. Also, a subspace spanned by only the first four POD modes might be insufficient in complex applications. On the other hand, GROM(16) is the slowest. We can also observe that computation time of UROM is close to that of NIROM and much lower than GROM(16). In Fig. 2.15, we present a bar chart for both the computing time and RMSE of reconstructed fields at final time to illustrate the time-accuracy trade-off.

Table 2.2: Computing time (in seconds) and RMSE of UROM, GROM(4), GROM(16), NIROM(4), and NIROM(16) for Re = 1000. We note that the computing time assessments documented in this table are based on Python executions.

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

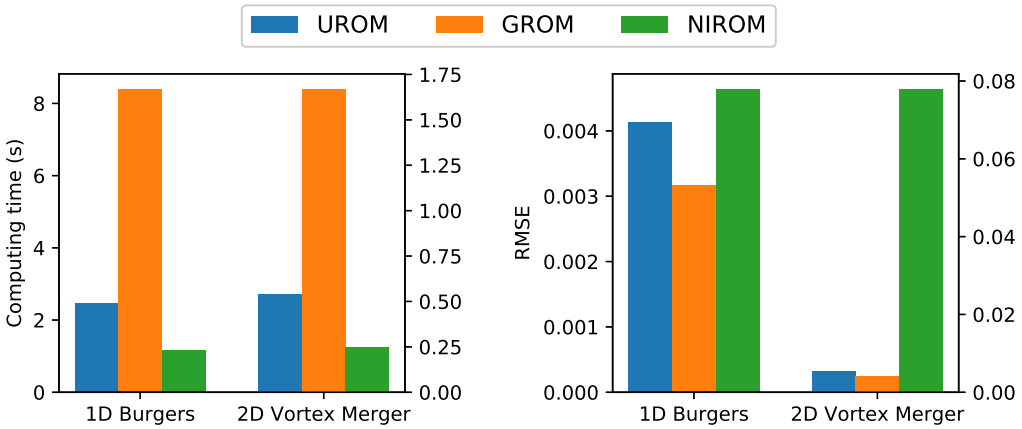


Figure 2.15: Computing time of testing (online) stage at Re = 1000 and RMSE of reconstructed fields at final time for UROM $(4 + 12)$ (i.e., $R = 4$ and $Q = 16$), GROM(16), and NIROM(16).

We note that Table 2.2 and Fig. 2.15 document the performance of our implementation rather than that of the approaches. We should emphasize that, in this chapter, we are not aiming at benchmarking the computational performance of these approaches. Instead, our main objective is to demonstrate the feasibility of hybrid approaches fusing physics-based and machine learning models. Nonetheless, the runtimes in Table 2.2 indicate that GROM approximately (not exactly due to various other loading/writing abstractions in our Python implementations) scales with R^3. Therefore, combining NIROM and GROM, UROM yields better computational performance. We also note that, if written more optimally, we would also expect that the execution time of UROM (with 16 modes) can be reduced to the sum of the execution times of GROM (with 4 modes) and NIROM (with 16 modes). Indeed, we remark

that the second LSTM in UROM (representing the map g) need not be used at all times and can be deployed only at the instant of interest. In that case, the UROM computing times become 1.31 s for Burgers case and 0.30 s for vortex merger (which are very close to NIROM computing time).

In a nutshell, our investigation, for the considered test cases, shows that GROM with 4 modes provides inaccurate results, while GROM with 16 modes gives good predictions. However, the computational cost of the latter is significantly higher than the computational cost of the former. Adopting the UROM approach, we are able to get an accuracy similar to the GROM(16) accuracy, but with a minimal computational cost. Moreover, UROM is more stable than NIROM for moderate LSTM architectures, since the UROM is always constrained in its core by the GROM update step. Conversely, NIROM is totally non-intrusive and the output is fed back to the LSTM in recursively, resulting in amplification of error for long time predictions. This framework can also be generalized to get higher accuracy or address more complex problems by increasing R and/or Q.

2.8 Conclusions

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

Two test cases, representing convection-dominated flows in 1D and 2D settings, have been used to evaluate the UROM. For testing, two out-of-sample control parameters have been investigated to study the interpolatory and extrapolatory performances. In all cases, UROM has showed very good results, compared to GROM and NIROM. In particular, UROM$(4 + 12)$ has been demonstrated to provide more

accurate results than both GROM(4) and NIROM. In contrast to NIROM where the deployment is fully data-driven, the LSTMs in UROM take their inputs from a physics-based approach. This can be considered as one way of leveraging physical information and intuition into LSTM. On the other hand, UROM has provided significant speed-ups compared to GROM(16) with comparable accuracy. Although we have presented the results for $Q = 16$, more complex flows can require much larger Q, which makes GROM(Q) unfeasible. Finally, this UROM approach is thought to open new avenues to utilize data-driven tools to enhance existing physical models as well as use physics to inform data-driven approaches to maximize the pros of both approaches and mitigate their cons.

CHAPTER 3

Physics Guided Machine Learning for Variational Multiscale Reduced Order Modeling

3.1 Abstract

We propose a new physics guided machine learning (PGML) paradigm that leverages the variational multi-scale (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.

3.2 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 analytic 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 equa-

This chapter is adapted from: *Ahmed, S. E., San, O., Rasheed, A., Iliescu, T., & Veneziani, A. (2022). Physics guided machine learning for variational multiscale reduced order modeling. SIAM Journal on Scientific Computing (under review).*

tions 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 multiple forward evaluations with varying inputs (e.g., flow control [27, 33, 121], optimization [122–128], and digital twinning [13, 129, 22, 130–132]) or studies requiring several simulations like computational-aided clinical trials [133].

Reduced order models (ROMs) are defined as computationally light surrogates that can mimic the behavior of FOMs with sufficient accuracy [16, 28, 25, 134, 1]. 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 [97, 19]. Of particular interest, the combination of proper orthogonal decomposition (POD) and Galerkin projection has been a powerful driver for ROM progress. 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., [34, 135, 136, 46, 137–140, 1]), 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 [43, 141]. 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 two-fold. 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* [142, 143, 2]. 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* [59, 144, 49, 63, 145, 57, 146, 47, 147–149]. 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 [1].

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 having diffusive effects on the larger scales. Therefore, eddy viscosity-based frameworks have been often used in the ROM literature [24]. Nonetheless, it was found that introducing eddy viscosity to *all* resolved scales can actually unnecessarily contaminate the dynamics of the *largest* scales. To mitigate this problem, the *variational multi-scale (VMS)* method, which was proposed by Hughes' group [150–152] in the finite element setting (see, e.g., [153, 154] for a survey), was utilized to add eddy viscosity dissipation to only a portion of the ROM resolved scales in [155, 156, 59]. A data-driven version of VMS (DD-VMS) has been recently proposed in [157], where the effects of the truncated modes onto the GROM dynamics are not restricted to be diffusive.

In the present study, we transform the DD-VMS [157, 158] 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 [157]. 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 [159–163]. Moreover, we adopt the physics guided machine learning (PGML) framework introduced in [164–166] 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 [167] to reduce the projection error without affecting the computational efficiency, by learning the correlations among the small unresolved scales to provide much fewer latent space variables. We also perform a numerical investigation of the proposed strategies (ML-VMS-ROM, PGML-VMS-ROM, and NLPOD-VMS-ROM), with a particular focus on the locality of scale interactions, which is a cornerstone of the VMS framework.

The rest of the chapter is organized as follows. We briefly describe the reduced order modeling methodology by the nexus of POD and Galerkin projection in Section 3.3. The relevant background information and notations for the VMS approach are given in Section 3.4. The use of the PGML methodology to provide reliable predictions is explained in Section 3.5, while the NLPOD approach is discussed in Section 3.6. The proposed NLPOD-PGML-VMS framework is tested for the parametric unsteady vortex-merger problem, which exemplifies convection-dominated flow systems. Results and discussions are presented in Section 3.7, followed by the concluding remarks in Section 3.8.

3.3 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). In order to eliminate the pressure term, we consider the NSE in the vorticity-vector potential formulation. In particular, we consider the 2D case where the vector potential is reduced to the streamfunction as follows:

$$\begin{aligned} \partial_t \omega - \nu \Delta \omega + (\boldsymbol{u} \cdot \nabla)\omega = 0, \qquad & \text{in } \Omega \times [0, T], \\ \Delta \psi + \omega = 0, \qquad & \text{in } \Omega \times [0, T], \end{aligned} \tag{3.1}$$

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:

$$\boldsymbol{u} = \nabla^{\perp} \psi, \quad \nabla^{\perp} = [\partial_y, -\partial_x]^T. \tag{3.2}$$

By using Eq. (3.2), Eq. (3.1) can be further rewritten as follows:

$$\partial_t \omega - \nu \Delta \omega + J(\omega, \psi) = 0, \qquad \text{in } \Omega \times [0, T], \tag{3.3}$$

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

$$J(\omega, \psi) = \frac{\partial \omega}{\partial x}\frac{\partial \psi}{\partial y} - \frac{\partial \omega}{\partial y}\frac{\partial \psi}{\partial x}. \tag{3.4}$$

The vorticity transport equation (Eq. (3.3)) 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:

$$\begin{aligned} IC : \omega(\boldsymbol{x}, 0) &= \omega_0(\boldsymbol{x}), && \text{in } \Omega, \\ BC \ (\text{non} - \text{slip}) : \psi(\boldsymbol{x}, t) &= 0, \quad \frac{\partial \psi}{\partial \boldsymbol{n}} = 0, && \text{in } \Gamma \times [0, T]. \end{aligned} \tag{3.5}$$

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 (Section 3.3.1), followed by the Galerkin method, where the FOM operators in Eq. (3.1) are projected onto the POD subspace to define the sought GROM (Section 3.3.2).

3.3.1 Proper orthogonal decomposition

We consider a collection of system realizations defined by an ensemble of vorticity fields $\{\omega(\boldsymbol{x}, t_0), \omega(\boldsymbol{x}, t_1), \ldots, \omega(\boldsymbol{x}, t_{M-1})\}$. These are often called *snapshots* and come from either experimental measurements or numerical simulations of Eq. (3.1) or Eq. (3.3) 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 $m = 0, 1, \ldots, M-1$ and $\Delta t = \dfrac{T}{M-1}$. 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 $\{\phi_1, \phi_2, \ldots, \phi_R\}$ that optimally approx-

imates the space spanned by the snapshots in the following sense [24]:

$$\begin{aligned} &\min \ \left\langle \left\| \omega(\cdot,\cdot) - \sum_{k=1}^{R} \big(\omega(\cdot,\cdot), \phi_k(\cdot)\big)\phi_k(\cdot) \right\|^2 \right\rangle, \\ &\text{subject to} \qquad \|\phi\| = 1, \qquad \big(\phi_i(\cdot), \phi_j(\cdot)\big) = \delta_{ij}, \end{aligned} \tag{3.6}$$

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:

$$\langle\omega\rangle = \frac{1}{M}\sum_{m=0}^{M-1} \omega(\cdot, t_m). \tag{3.7}$$

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 Eq. (3.6) amounts to solving the following eigenvalue problem [168]:

$$\mathbf{D}\Phi = \Phi\Lambda, \tag{3.8}$$

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

$$\big[\mathbf{D}\big]_{ij} = \Big\langle \boldsymbol{\omega}(\boldsymbol{x}_i, \cdot)\boldsymbol{\omega}(\boldsymbol{x}_j, \cdot) \Big\rangle, \tag{3.9}$$

where $\boldsymbol{\omega}(\boldsymbol{x}_i, \cdot)$ is the i-th entry of $\boldsymbol{\omega}$. For fluid flow problems, the length of the vector $\boldsymbol{\omega}$ is often large, which makes the eigenvalue problem in Eq. (3.8) computationally challenging.

Sirovich [169–171] proposed a numerical procedure, known as the *method of snapshots*, to reduce the computational cost of solving Eq. (3.8). 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 Eq. (3.8) 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:

$$\big[\mathbf{K}\big]_{ij} = \frac{1}{M}\Big(\omega(\cdot, t_i), \omega(\cdot, t_j)\Big). \tag{3.10}$$

The following eigenvalue problem is thus considered:

$$\mathbf{K}\mathbf{v}_k = \lambda_k \mathbf{v}_k, \tag{3.11}$$

where $\mathbf{v}_k$ is the k^{th} eigenvector of $\mathbf{K}$ and λ_k is the associated eigenvalue. To obtain the hierarchy of the POD basis, the eigenpairs are sorted in a 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 [168]:

$$\phi_k(\cdot) = \frac{1}{\sqrt{\lambda_k}} \sum_{m=0}^{M-1} [\mathbf{v}_k]_m \omega(\cdot, t_m), \tag{3.12}$$

where $[\mathbf{v}_k]_m$ denotes the m^{th} component of $\mathbf{v}_k$. It can be verified that the basis functions in Eq. (3.12) 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. A metric that evaluates the quality of a given set of retained modes in representing the system is the relative information content (RIC) [1], defined as follows:

$$\text{RIC}(k) = \frac{\sum_{l=1}^{k} \lambda_l}{\sum_{l=1}^{M} \lambda_l}, \tag{3.13}$$

where k is the POD index at which modal truncation takes place. We emphasize that the same approach can be applied considering parameters other than time. In this case, the temporal correlation matrix is substituted by a generalized parameter correlation matrix.

3.3.2 Galerkin projection

The GROM starts by the Galerkin truncation step, making use of the optimality criterion in Eq. (3.6) as follows:

$$\omega(\boldsymbol{x}, t_m) \approx \omega_R(\boldsymbol{x}, t_m) = \sum_{k=1}^{R} a_k(t_m) \phi_k(\boldsymbol{x}), \tag{3.14}$$

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:

$$a_k(t_m) = \big(\omega(\cdot, t_m), \phi_k(\cdot)\big). \tag{3.15}$$

Next, the vorticity field ω in Eq. (3.3) is replaced by its approximation ω_R from Eq. (3.14). After this, the Galerkin projection step comes into play, by defining the POD test subspace $\boldsymbol{X}_R$ as follows:

$$\boldsymbol{X}_R := \text{span}\{\phi_1, \phi_2, \ldots, \phi_R\}. \tag{3.16}$$

Finally, Eq. (3.3) with ω replaced by ω_R 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:

$$(\partial_t \omega, \phi) - \nu(\Delta\omega, \phi) + \big(J(\omega, \psi), \phi\big) = 0, \qquad \forall \phi \in \boldsymbol{X}_R. \tag{3.17}$$

Equation (3.17) can be written in a tensorial form as follows:

$$\dot{\boldsymbol{a}} = A\boldsymbol{a} + \boldsymbol{a}^\top B\boldsymbol{a}, \tag{3.18}$$

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.

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 [1].

The projection error and closure error are illustrated in Fig. 3.1, for a toy system whose full-rank approximation can be represented with 3 modes as follows:

$$\omega(x,t) = a_1(t)\phi_1(x) + a_2(t)\phi_2(x) + a_3(t)\phi_3(x). \tag{3.19}$$

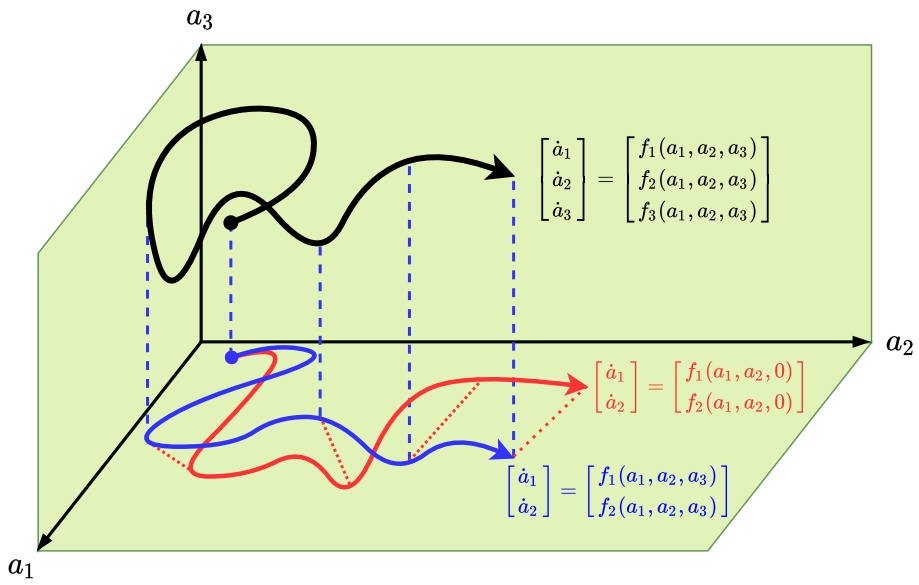


Figure 3.1: Representation of the repercussions of modal truncation onto the ROM solution. The solid black curve denotes the FOM trajectory, assuming that the full rank expansion is defined by a_1, a_2, and a_3. The solid blue curve defines the projection of the FOM trajectory onto a two-dimensional 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.

Assuming that the FOM is written in the following form:

$$\dot{\omega} = F(\omega), \tag{3.20}$$

then 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:

$$\begin{bmatrix}\dot{a}_1\\ \dot{a}_2\\ \dot{a}_3\end{bmatrix} = \begin{bmatrix}f_1(a_1,a_2,a_3)\\ f_2(a_1,a_2,a_3)\\ f_3(a_1,a_2,a_3)\end{bmatrix}. \tag{3.21}$$

In other words, evolving $\{a_k\}_{k=1}^3$ using Eq. (3.21) and reconstructing ω with Eq. (3.19) recovers the FOM field (equivalent to solving Eq. (3.20) 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 Eq. (3.21)

as follows:

$$\begin{bmatrix} \dot{a}_1 \\ \dot{a}_2 \end{bmatrix} = \begin{bmatrix} f_1(a_1, a_2, a_3) \\ f_2(a_1, a_2, a_3) \end{bmatrix}. \tag{3.22}$$

Approximating ω with just two modes results in losing the flow structures that are contained in the truncated mode (the vertical direction in Fig. 3.1), 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 onto the resolved scales are assumed to be negligible, as follows:

$$\begin{bmatrix} \dot{a}_1 \\ \dot{a}_2 \end{bmatrix} = \begin{bmatrix} f_1(a_1, a_2, 0) \\ f_2(a_1, a_2, 0) \end{bmatrix}. \tag{3.23}$$

We denote the reference trajectory described by Eq. (3.22) as the true projection, which is related to Eq. (3.15). 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 Eq. (3.23)) 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 3.4 to develop the PGML methodology in Section 3.5. Then, we utilize the NLPOD approach in Section 3.6 to reduce the projection error by learning a compressed latent space that encapsulates some of the truncated flow structures.

3.4 Variational Multiscale Method

The variational multi-scale (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 [150–152], has made a profound impact in many areas of computational mechanics (see, e.g., [153, 154] for a survey).

To illustrate the standard VMS methodology, we consider a general nonlinear partial differential equation

$$\dot{\omega} = \boldsymbol{F}(\omega), \tag{3.24}$$

whose weak (variational) form is

$$(\dot{\omega}, \phi) = (\boldsymbol{F}(\omega), \phi), \quad \forall \phi \in \boldsymbol{X}, \tag{3.25}$$

where $\boldsymbol{F}$ 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 Eq. (3.24) 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:

$$(\dot{\omega}, \phi) = (\boldsymbol{F}(\omega), \phi), \quad \forall \phi \in \boldsymbol{X}_1. \tag{3.26}$$

However, system Eq. (3.26) is *not* closed since its right-hand side involves ω components that do not belong to $\boldsymbol{X}_1$ (i.e., $\omega_2 \in \boldsymbol{X}_2$, $\omega_3 \in \boldsymbol{X}_3$, ...):

$$(\boldsymbol{F}(\omega), \phi) = (\boldsymbol{F}(\omega_1, \omega_2, \omega_3, \dots), \phi), \quad \forall \phi \in \boldsymbol{X}_1. \tag{3.27}$$

Thus, the VMS closure problem needs to be solved. That is, Eq. (3.27) needs to be replaced with an equation that involves only terms that belong to $\boldsymbol{X}_1$. In general, the VMS system in Eq. (3.26) 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 Section 3.3.1 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 multi-level VMS ROM. In particular, we detail the two-scale and the three-scale VMS ROMs, while further extensions become straightforward.

3.4.1 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:

$$\begin{aligned} \boldsymbol{X}_1 &:= \operatorname{span}\{\phi_1, \phi_2, \dots, \phi_R\}, \\ \boldsymbol{X}_2 &:= \operatorname{span}\{\phi_{R+1}, \phi_{R+2}, \dots, \phi_N\}, \end{aligned} \tag{3.28}$$

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:

$$\omega = \sum_{k=1}^{R} a_k \phi_k + \sum_{k=R+1}^{N} a_k \phi_k = \underbrace{\omega_R}_{\text{resolved}} + \underbrace{\omega'}_{\text{unresolved}}, \tag{3.29}$$

where $\omega_R \in \boldsymbol{X}_1$ is the resolved ROM component of ω, while $\omega' \in \boldsymbol{X}_2$ is the unresolved component. Using this decomposition, Eq. (3.26) can be rewritten as follows:

$$\big(\dot{\omega}_R, \phi_k\big) = \big(\boldsymbol{F}(\omega_R), \phi_k\big) + \underbrace{\Big[\big(\boldsymbol{F}(\omega), \phi_k\big) - \big(\boldsymbol{F}(\omega_R), \phi_k\big)\Big]}_{\text{VMS-2 closure term}}, \ \forall k \in \{1, \ldots, R\}. \tag{3.30}$$

The bracketed term in Eq. (3.30) 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 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 $\boldsymbol{G}(\omega_R)$ as follows:

$$\big(\boldsymbol{G}(\omega_R), \phi_k\big) \approx \big(\boldsymbol{F}(\omega), \phi_k\big) - \big(\boldsymbol{F}(\omega_R), \phi_k\big), \tag{3.31}$$

and the VMS-2 ROM can be written as

$$\big(\dot{\omega}_R, \phi_k\big) = \big(\boldsymbol{F}(\omega_R), \phi_k\big) + \big(\boldsymbol{G}(\omega_R), \phi_k\big). \tag{3.32}$$

The form and parameters of $\boldsymbol{G}$ will be defined in Section 3.5.

3.4.2 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:

$$\begin{aligned}\boldsymbol{X}_1 &:= \text{span}\{\phi_1, \phi_2, \ldots, \phi_r\},\\ \boldsymbol{X}_2 &:= \text{span}\{\phi_{r+1}, \phi_{r+2}, \ldots, \phi_R\},\\ \boldsymbol{X}_3 &:= \text{span}\{\phi_{R+1}, \phi_{R+2}, \ldots, \phi_N\}.\end{aligned} \tag{3.33}$$

Compared to the decomposition into resolved and unresolved scales in Section 3.4.1, $\boldsymbol{X}_1$ now represents the *large resolved* ROM scales, $\boldsymbol{X}_2$ represents the *small resolved* ROM scales, while $\boldsymbol{X}_3$ denotes the unresolved ROM scales. With these definitions, ω can be written as follows:

$$\begin{aligned}\omega &= \sum_{k=1}^{r} a_k\phi_k + \sum_{k=r+1}^{R} a_k\phi_k + \sum_{k=R+1}^{N} a_k\phi_k\\ &= \underbrace{\omega_L}_{\text{large resolved}} + \underbrace{\omega_S}_{\text{small resolved}} + \underbrace{\omega'}_{\text{unresolved}}.\end{aligned} \tag{3.34}$$

This is similar to Eq. (3.29) with $\omega_R = \omega_L + \omega_S$. To construct the VMS-3 ROM, we project system Eq. (3.24) onto each of the spaces $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, as follows:

$$\big(\dot{\omega}_L, \phi_k\big) = \big(\boldsymbol{F}(\omega_L + \omega_S), \phi_k\big) + \Big[\big(\boldsymbol{F}(\omega), \phi_k\big) - \big(\boldsymbol{F}(\omega_L + \omega_S), \phi_k\big)\Big],\ k = 1, \ldots, r, \tag{3.35}$$

$$\big(\dot{\omega}_S, \phi_k\big) = \big(\boldsymbol{F}(\omega_L + \omega_S), \phi_k\big) + \Big[\big(\boldsymbol{F}(\omega), \phi_k\big) - \big(\boldsymbol{F}(\omega_L + \omega_S), \phi_k\big)\Big],\ k = r+1, \ldots, R. \tag{3.36}$$

Although the two bracketed terms in Eq. (3.35) and Eq. (3.36) defining the VMS-3 closure terms look similar, they have different roles. The first term models basically the interaction between the large and the small resolved modes, because the interaction large-unresolved is assumed to be negligible (according to the VMS principle of locality of modal interactions). The second term models the interaction between the small resolved and the 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 3.7.

3.5 Physics Guided Machine Learning

In this section, the VMS-2 and VMS-3 closure terms defined in Section 3.4 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 [157]), we exploit the capabilities of 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 [172–174]. 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.

3.5.1 ML-VMS ROM

The VMS-2 ROM in Eq. (3.32) can be rewritten as follows:

$$\dot{\boldsymbol{a}} = \boldsymbol{f}(\boldsymbol{a}) + \boldsymbol{c}(\boldsymbol{a}), \tag{3.37}$$

where $\boldsymbol{a} = [a_1, a_2, \ldots, a_R]^T \in \mathbb{R}^R$ is the vector of coefficients for resolved POD modes, $\boldsymbol{f}(\boldsymbol{a}) = [\big(\boldsymbol{F}(\omega_R), \phi_1\big), \big(\boldsymbol{F}(\omega_R), \phi_2\big), \ldots, \big(\boldsymbol{F}(\omega_R), \phi_R\big)]$ represents the Galerkin projection of the FOM operators onto the POD subspace, and $\boldsymbol{c}(\boldsymbol{a}) = [c_1, c_2, \ldots, c_R]^R \in \mathbb{R}^R$ is the vector of the closure (correction) terms, i.e., $c_k = (\boldsymbol{G}(\omega_R), \phi_k)$. 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:

$$\begin{aligned} \mathbf{h}^{(n)} &= f_h^h(\boldsymbol{a}^{(n)}, \mathbf{h}^{(n-1)}), \\ \boldsymbol{c}^{(n)} &= f_h^o(\mathbf{h}^{(n)}), \end{aligned} \tag{3.38}$$

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 Eq. (3.38), $\mathbf{h} \in \mathbb{R}^H$ 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 [175–177, 146, 178] shows that non-Markovian terms are required to account for the effects of the unresolved scales onto 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 [157, 158], 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:

$$
\begin{aligned}
\mathbf{g}_f^{(n)} &= \sigma_f(\mathbf{W}_f[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_f), \\
\mathbf{g}_i^{(n)} &= \sigma_i(\mathbf{W}_i[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_i), \\
\tilde{\mathbf{s}}^{(n)} &= \tanh\left(\mathbf{W}_s[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_s\right), \\
\mathbf{s}^{(n)} &= \mathbf{g}_f^{(n)} \odot \mathbf{s}^{(n-1)} + \mathbf{g}_i^{(n)} \odot \tilde{\mathbf{s}}^{(n)}, \\
\mathbf{g}_o^{(n)} &= \sigma_o(\mathbf{W}_o[\mathbf{h}^{(n-1)}, \boldsymbol{a}^{(n)}] + \mathbf{b}_o), \\
\mathbf{h}^{(n)} &= \mathbf{g}_o^{(n)} \odot \tanh\left(\mathbf{s}^{(n)}\right),
\end{aligned}
\tag{3.39}
$$

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 element-wise 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

$$
\boldsymbol{c}^{(n)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \mathbf{h}_{l-1}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}) \tag{3.40}
$$

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 l^{th} layer passes only the hidden state at the final time $\mathbf{h}_l^{(n)}$ to the output layer.

To summarize, Eqs. (3.37) to (3.40) yield the ML-VMS-2 ROM. In order to make use of the locality of modal interactions, the VMS-3 ROM is written as

$$\begin{bmatrix} \dot{\boldsymbol{a}}_L \\ \dot{\boldsymbol{a}}_S \end{bmatrix} = \boldsymbol{f}(\boldsymbol{a}) + \begin{bmatrix} \boldsymbol{c}_L(\boldsymbol{a}) \\ \boldsymbol{c}_S(\boldsymbol{a}) \end{bmatrix}, \tag{3.41}$$

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:

$$\begin{aligned} \boldsymbol{c}_L^{(n)} &\approx \pi_{L,\theta}(\boldsymbol{a}^{(n):(n-\tau)}) \\ &\approx \mathcal{L}_{_L}(\cdot) \circ \mathbf{h}_{l_L}^{(n)}(\cdot) \circ \mathbf{h}_{l-1_L}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_{1_L}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}), \\ \boldsymbol{c}_S^{(n)} &\approx \pi_{S,\theta}(\boldsymbol{a}^{(n):(n-\tau)}) \\ &\approx \mathcal{L}_{_S}(\cdot) \circ \mathbf{h}_{l_S}^{(n)}(\cdot) \circ \mathbf{h}_{l-1_S}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_{1_S}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}). \end{aligned} \tag{3.42}$$

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.

3.5.2 PGML-VMS ROM

Critical aspects that should be considered during the adoption of ML based approach include their reliability, robustness, and trustworthiness. Previous studies [164–166] 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 augments 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 3.7.

In the PGML framework, the features extracted from the physics-based model

are embedded into the generic i^{th} 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

$$\begin{aligned}\boldsymbol{c}^{(n)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_i^{(n):(n-\tau)}(\cdot), \boldsymbol{f}^{(n):(n-\tau)}\right) \circ \mathbf{h}_{i-1}^{(n):(n-\tau)}(\cdot) \\ \circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}),\end{aligned} \tag{3.43}$$

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 Fig. 3.2. 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 hidden layers. One of the main advantages of the novel PGML framework in Fig. 3.2 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

$$\begin{aligned}\boldsymbol{c}_L^{(n)} \approx \mathcal{L}_L(\cdot) \circ \mathbf{h}_{l_L}^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_{i_L}^{(n):(n-\tau)}(\cdot), \boldsymbol{f}_L^{(n):(n-\tau)}\right) \circ \mathbf{h}_{i-1_L}^{(n):(n-\tau)}(\cdot) \\ \circ \cdots \circ \mathbf{h}_{1_L}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}), \\ \boldsymbol{c}_S^{(n)} \approx \mathcal{L}_S(\cdot) \circ \mathbf{h}_{l_S}^{(n)}(\cdot) \circ \cdots \circ \mathcal{C}\left(\mathbf{h}_{i_S}^{(n):(n-\tau)}(\cdot), \boldsymbol{f}_S^{(n):(n-\tau)}\right) \circ \mathbf{h}_{i-1_S}^{(n):(n-\tau)}(\cdot) \\ \circ \cdots \circ \mathbf{h}_{1_S}^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{a}^{(n):(n-\tau)}).\end{aligned} \tag{3.44}$$

Note that in Eq. (3.44), 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 Galerkin propagator of only a few relevant neighboring modes (i.e., $\boldsymbol{f}_L$ and $\boldsymbol{f}_S$ in Eq. (3.44)), rather than including all of them in the LSTM learning (i.e., $\boldsymbol{f}$ in Eq. (3.43)).

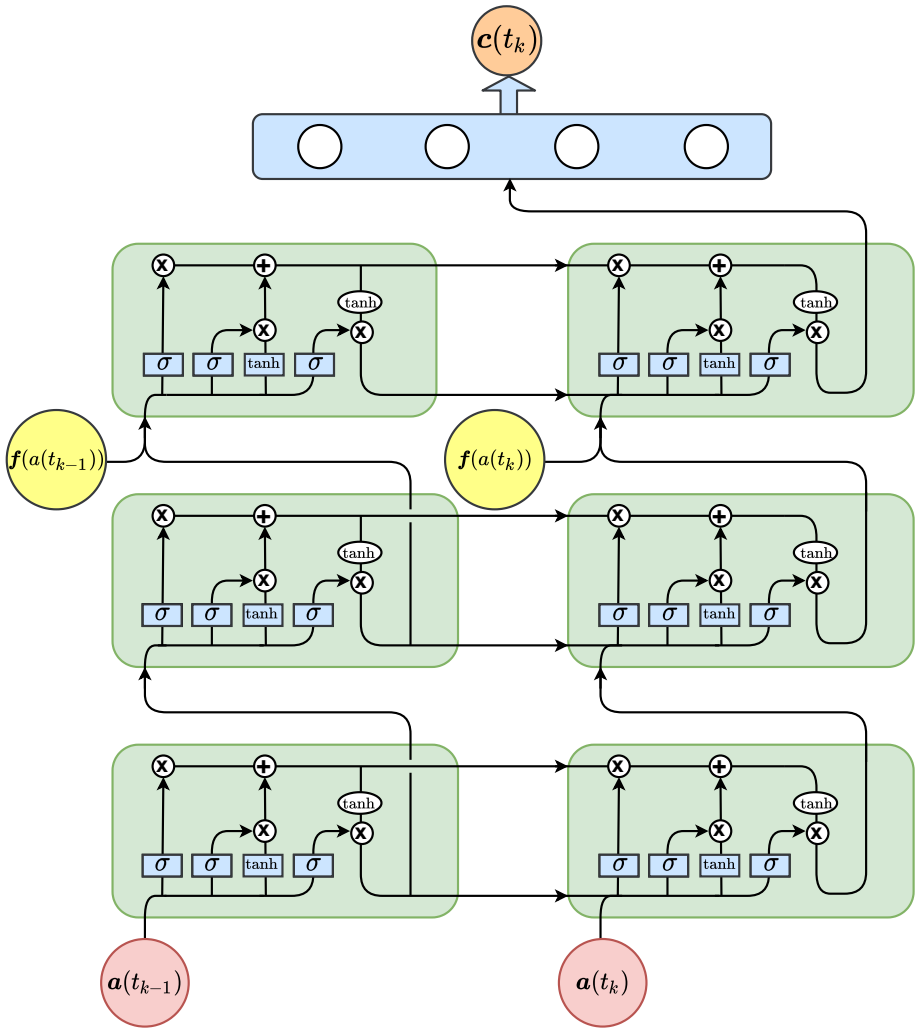


Figure 3.2: Illustration of the PGML methodology with concatenated LSTM layers. In this figure, a time history of 2 time steps is used while physics-based features (yellow circles in the figure) are injected into the LSTM latent space after the second hidden layer ($i = 2$).

3.6 Nonlinear POD

In Section 3.4 and Section 3.5, we addressed the closure problem. That is, we aimed at correcting the ROM equations for the dynamics of the resolved scales including the effects of the unresolved scales onto the dynamics of the resolved scales. However, the reconstructed flow fields were approximated within the span of the retained modes, as shown in Eq. (3.14). Nonetheless, for turbulent 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 [167], to model the unresolved part of the field. Fig. 3.3 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 leverages 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., $\text{RIC}(K) \geq 99.99\%$). The goal is to learn $\boldsymbol{z} = \{z_k\}_{k=1}^{q}$, where $q \ll K$ denotes the dimension of the AE latent space.

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:

$$\text{Encoder } \eta : \boldsymbol{b} \in \mathbb{R}^{K-R} \mapsto \boldsymbol{z} \in \mathbb{R}^{q}, \qquad \text{Decoder } \zeta \quad : \boldsymbol{z} \in \mathbb{R}^{q} \mapsto \boldsymbol{b} \in \mathbb{R}^{K-R}, \tag{3.45}$$

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

$$\mathcal{J} = \sum_{n=1}^{N_{train}} \|\boldsymbol{b}^{(n)} - (\eta \circ \zeta)(\boldsymbol{b}^{(n)})\|, \tag{3.46}$$

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

In order to temporally propagate $\boldsymbol{z}$, we can use any of the regression tools, in-

cluding 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 3.5 to learn the one time-step mapping from $\boldsymbol{z}^{(n)}$ to $\boldsymbol{z}^{(n+1)}$, as follows:

$$\boldsymbol{z}^{(n+1)} \approx \mathcal{L}(\cdot) \circ \mathbf{h}_l^{(n)}(\cdot) \circ \mathbf{h}_{l-1}^{(n):(n-\tau)}(\cdot) \circ \cdots \circ \mathbf{h}_1^{(n):(n-\tau)}(\cdot) \circ \mathcal{I}(\boldsymbol{z}^{(n):(n-\tau)}). \tag{3.47}$$

Note that the number of layers, l, and the length of time history, τ, are not necessarily equal to those in Section 3.5. 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.

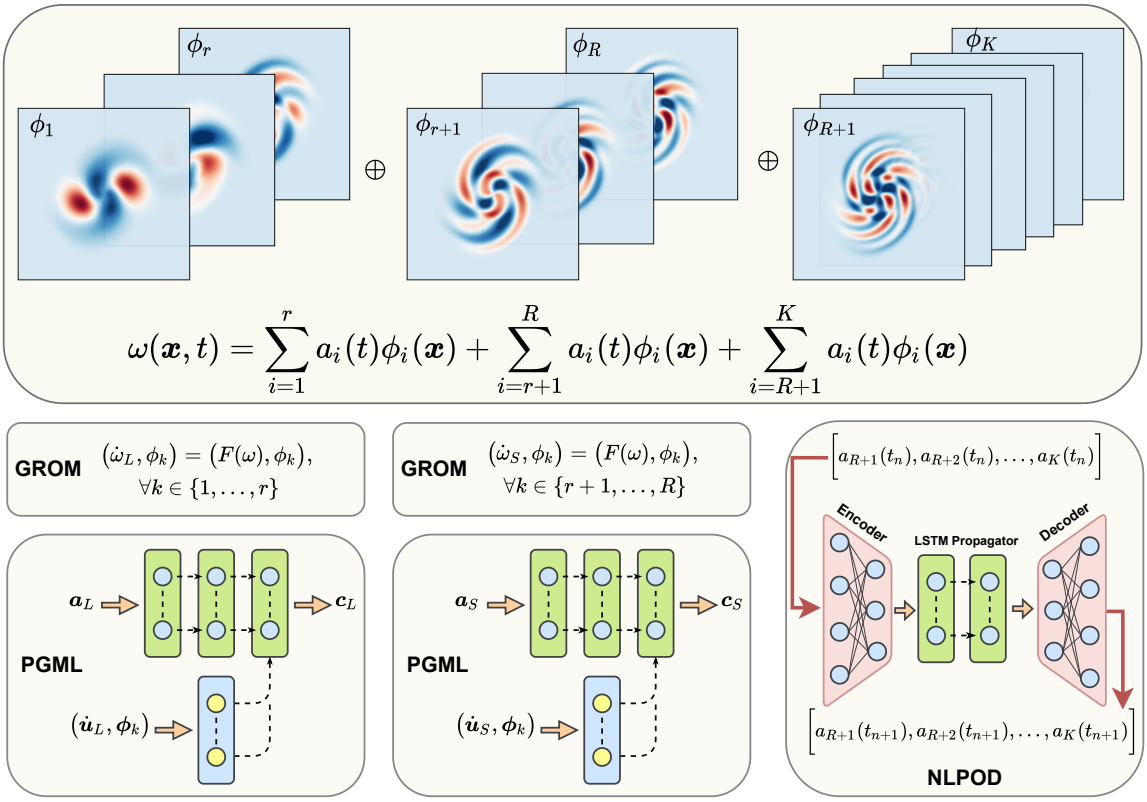


Figure 3.3: Schematic representation of the PGML-VMS-3 model for the large and small resolved scales, combined with NLPOD for enhanced field reconstruction. We note that PGML-VMS-3 is built upon a GROM for the first R modes and mitigates the closure error (i.e., the effect of the truncated scales onto 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 onto the flow field reconstruction).

3.7 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 two dimensional (2D) vortex merger problem [179], governed by the following vorticity transport equation:

$$\partial_t \omega + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \Delta \omega, \qquad \text{in } \Omega \times [0, T]. \tag{3.48}$$

We consider a spatial domain of dimensions $(2\pi \times 2\pi)$ with periodic boundary conditions. The flow is initialized with a pair of co-rotating 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:

$$\omega(x, y, 0) = \exp\left(-\rho\left[(x - x_1)^2 + (y - y_1)^2\right]\right) + \exp\left(-\rho\left[(x - x_2)^2 + (y - y_2)^2\right]\right), \tag{3.49}$$

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 of 0.001. Vorticity snapshots are collected every 100 time-steps for $t \in [0, 30]$, totalling 300 snapshots. The evolution of the vortex merger problem at selected values of the Reynolds number is depicted in Fig. 3.4, which illustrates the convective and dissipative mechanisms affecting the transport and development of the two vortices.

In terms of POD analysis, we use $R = 6$ 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 Fig. 3.5.

Following a systematic approach, in Section 3.7.1, 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 Section 3.7.2, we reveal the additional role of the NLPOD approach by illustrating the performance of the PGML-VMS-3+NLPOD approach.

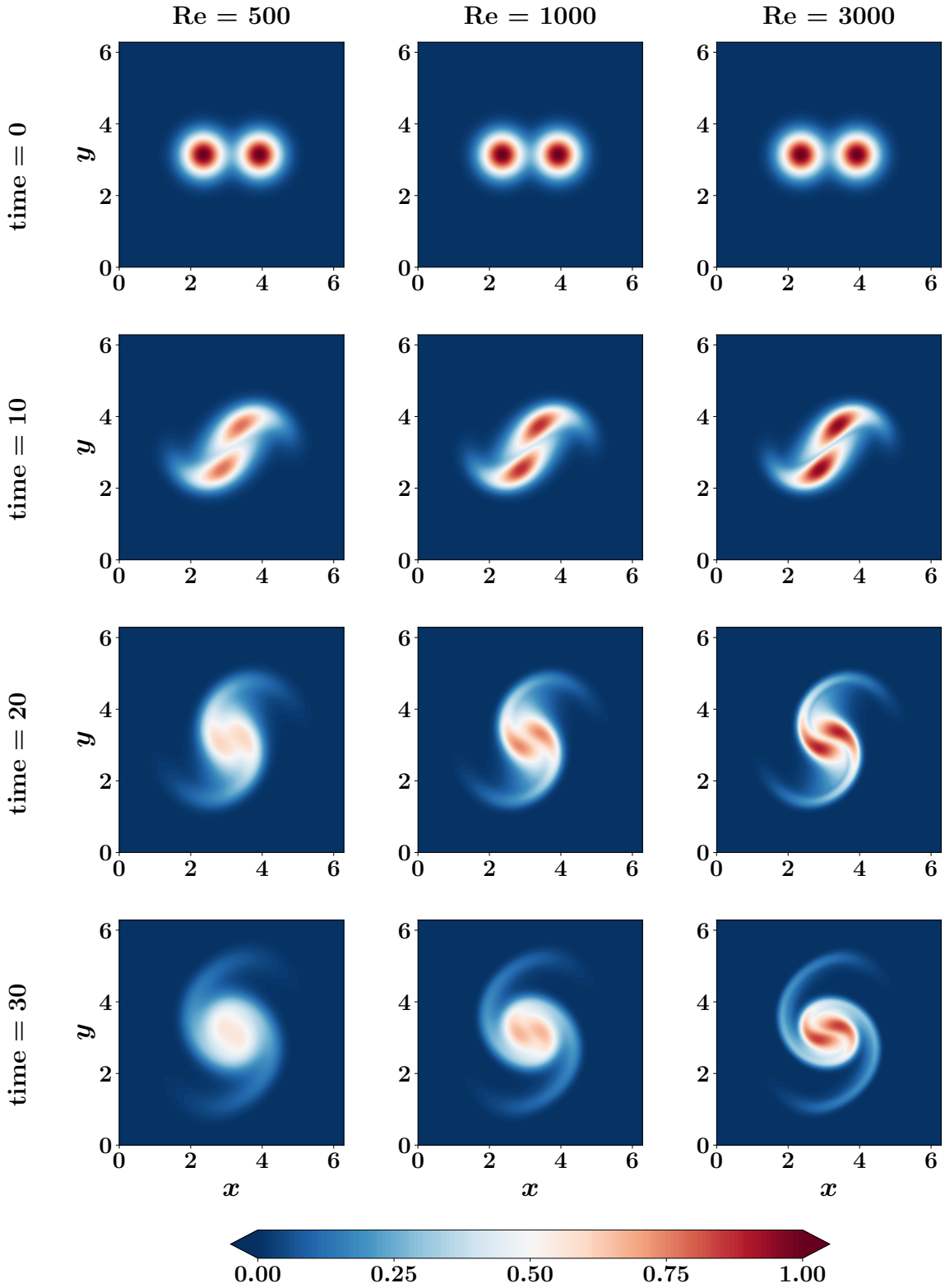


Figure 3.4: Samples of temporal snapshots of the vorticity field for the vortex merger problem at different values of Reynolds number.

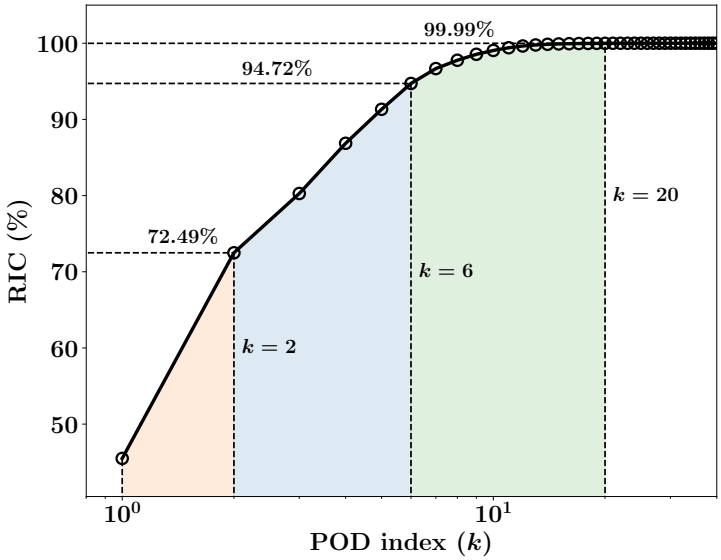


Figure 3.5: RIC values as a function of the modal truncation for the vortex merger problem at Re = 3000.

3.7.1 Multi-level VMS closure for resolved scales

We store data corresponding to Re $\in \{500, 750, 1000, \ldots, 3000\}$, but we use only the data collected at Re $\in \{500, 750, 1000\}$ for neural network training and reserve the remaining data for testing purposes. First, we explore the combination of multi-level variational multi-scale methods with machine learning. Figure 3.6 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 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 3.5. Figure 3.7 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 GROM trajectory from the true projections are larger for the latest 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.

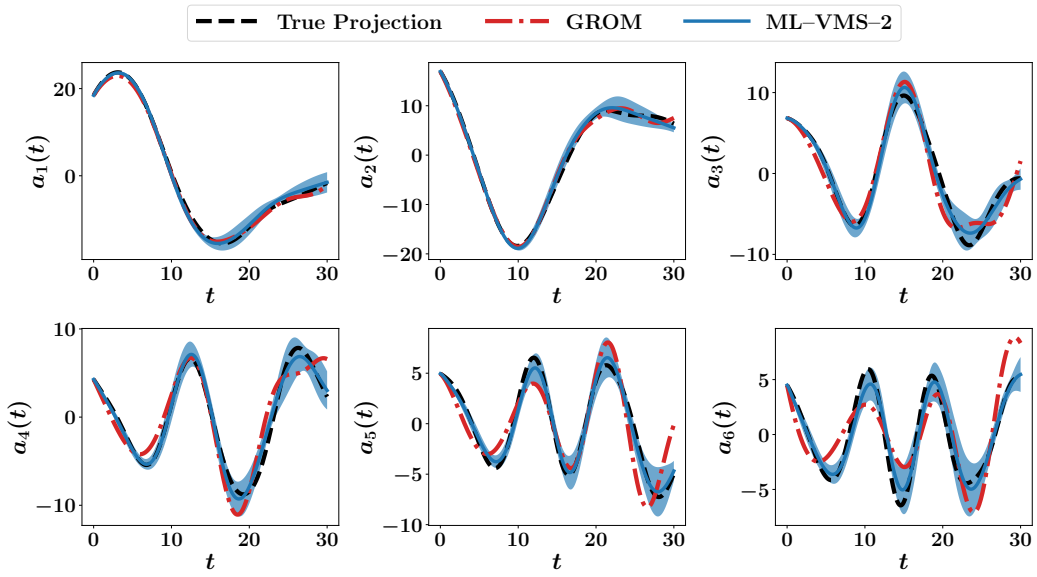


Figure 3.6: The time evolution for the first 6 modes of the vortex merger problem with the two-level VMS using ML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values (μ) from an ensemble of 10 different LSTM neural networks trained with different weight initalizations, while the shaded area defines the uncertainty bounds using standard deviation (σ) values. For better visualization, the shaded band is plotted with $\mu \pm 5\sigma$.

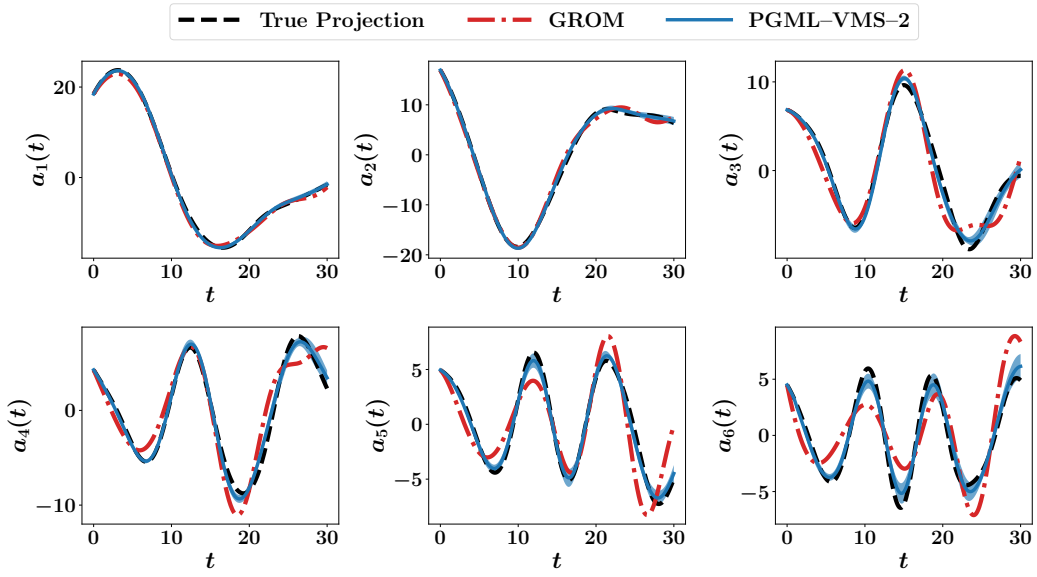


Figure 3.7: The time evolution for the first 6 modes of the vortex merger problem with the two-level VMS using PGML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values (μ) from an ensemble of 10 different LSTM neural networks trained with different weight initalizations, while the shaded area defines the uncertainty bounds using standard deviation (σ) values. For better visualization, the shaded band is plotted with $\mu \pm 5\sigma$.

In Fig. 3.8, 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 ones. 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 which 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.

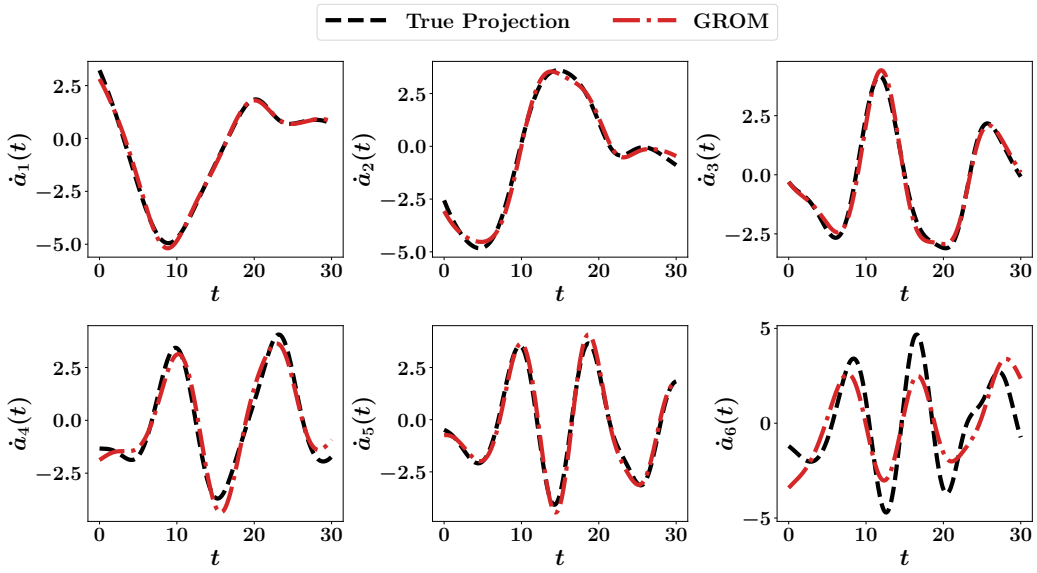


Figure 3.8: Comparison between the ROM propagator computed by Galerkin projection (with truncation, i.e., $\dot{a}_k = (-J(\omega_R, \psi_R) + \nabla^2\omega_R, \phi_k)$, against the true (FOM projection) propagator (i.e., $\dot{a}_k = (-J(\omega, \psi) + \nabla^2\omega, \phi_k)$ at Re $= 3000$ and $R = 6$. We notice that the Galerkin projection accurately captures the dynamics of the first modes, but a discrepancy appears at the latest modes, which motivates the use of multi-level VMS closure.

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. In particular, we split the resolved scales into two parts: the first 2 modes represent the largest 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 Fig. 3.9. Compared to Fig. 3.6, the ML-VMS-3 provides more accurate results than the ML-VMS-2, even in terms of uncertainty levels.

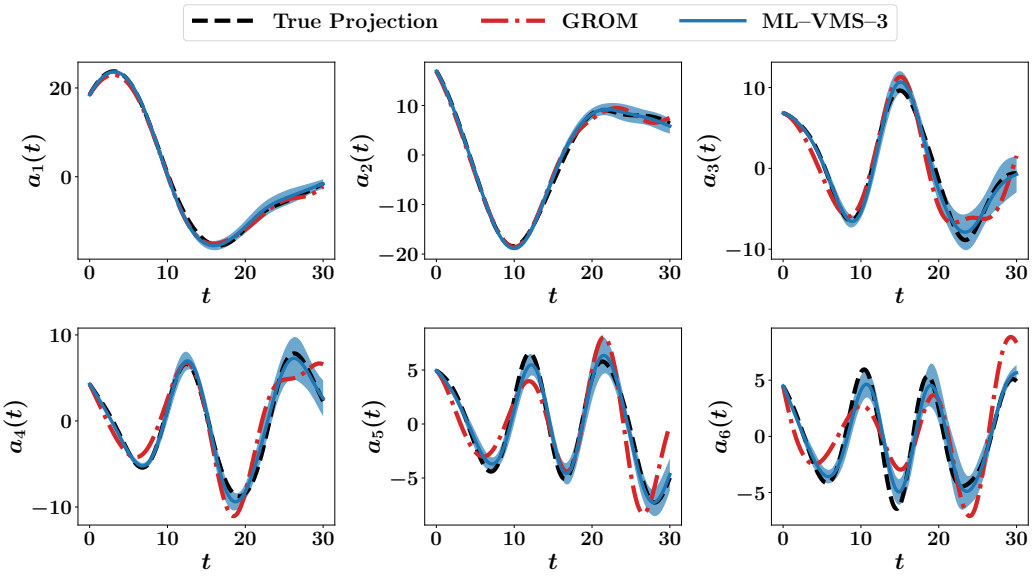


Figure 3.9: The time evolution for the first 6 modes of the vortex merger problem with the three-level VMS using ML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values (μ) from an ensemble of 10 different LSTM neural networks trained with different weight initalizations, while the shaded area defines the uncertainty bounds using standard deviation (σ) values. For better visualization, the shaded band is plotted with $\mu \pm 5\sigma$.

Finally, the PGML-VMS-3 results are shown in Fig. 3.10, where we can see improved results across all the resolved scales with very low levels of uncertainty. 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 Fig. 3.11. 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 number values 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 is providing more accurate ROMs than VMS-2, making use of the locality of information transfer to build more localized closure models.

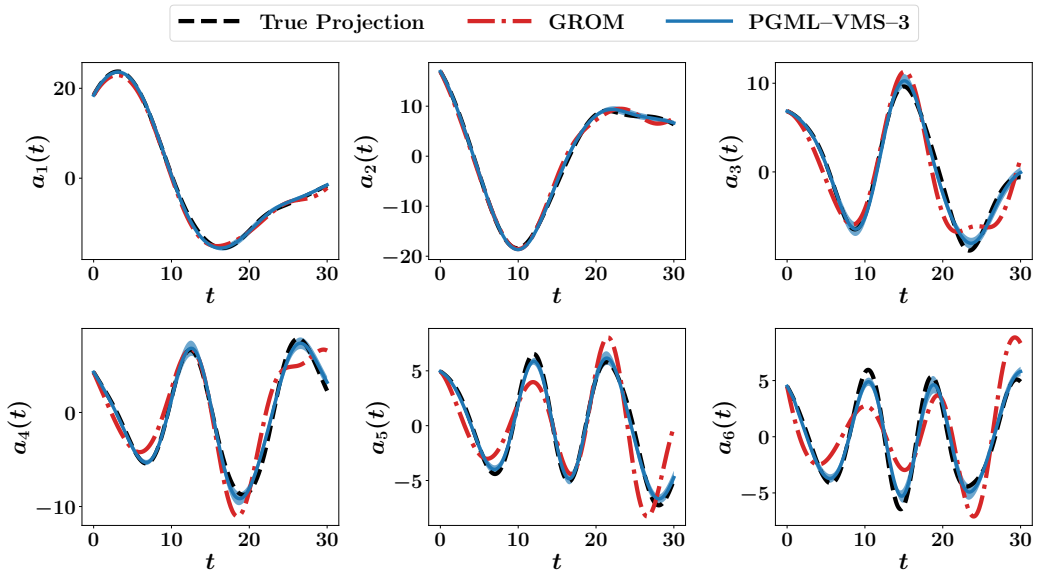


Figure 3.10: The time evolution for the first 6 modes of the vortex merger problem with the three-level VMS using PGML closure, compared to the true projection and GROM (without closure) predictions. The solid line represents the mean values (μ) from an ensemble of 10 different LSTM neural networks trained with different weight initalizations, while the shaded area defines the uncertainty bounds using standard deviation (σ) values. For better visualization, the shaded band is plotted with $\mu \pm 5\sigma$.

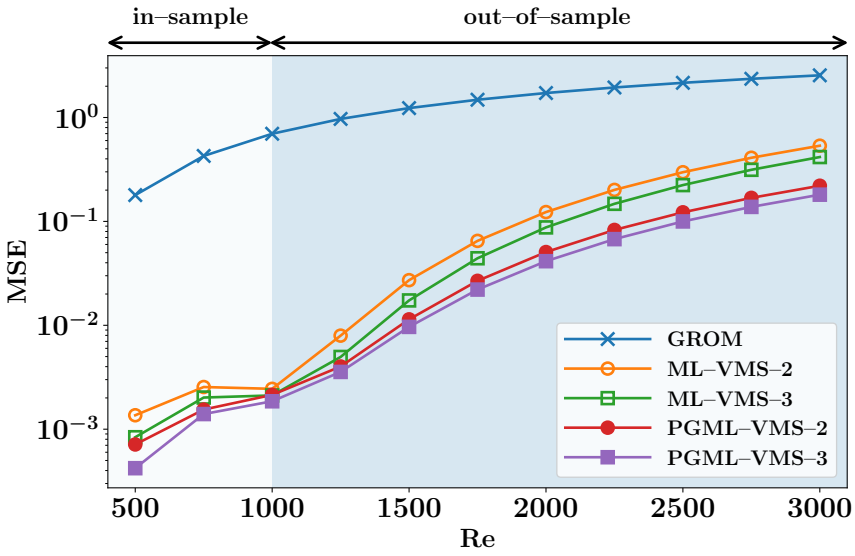


Figure 3.11: Mean squared error (MSE) between the true values of modal coefficients and the predictions of GROM, ML-VMS-2, ML-VMS-3, PGML-VMS-2, and PGML-VMS-3.

3.7.2 NLPOD for unresolved scales

The reconstructed vorticity fields from GROM, true projection, and PGML-VMS-3 at final time (i.e., $t = 30$) at $\mathrm{Re} = 3000$ are visualized in Fig. 3.12. We can see that the GROM field is significantly inaccurate. In contrast, the PGML-VMS-3 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 values of the Reynolds number. On the other hand, building a projection-based ROM with increased number of modes will result in an undesired higher computational burden.

In order to cure this limitation, we apply the NLPOD methodology from Section 3.6 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 two-dimensional compression of the resolved scales, i.e., $\boldsymbol{z} = \{z_k\}_{k=1}^{2} \in \mathbb{R}^2$. Fig. 3.13 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 3.1. For the ROMs, we can see that the majority of the time is spent to train 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 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 resulting in computationally light models that can be used efficiently in the online stage. Moreover, we notice that the costs of

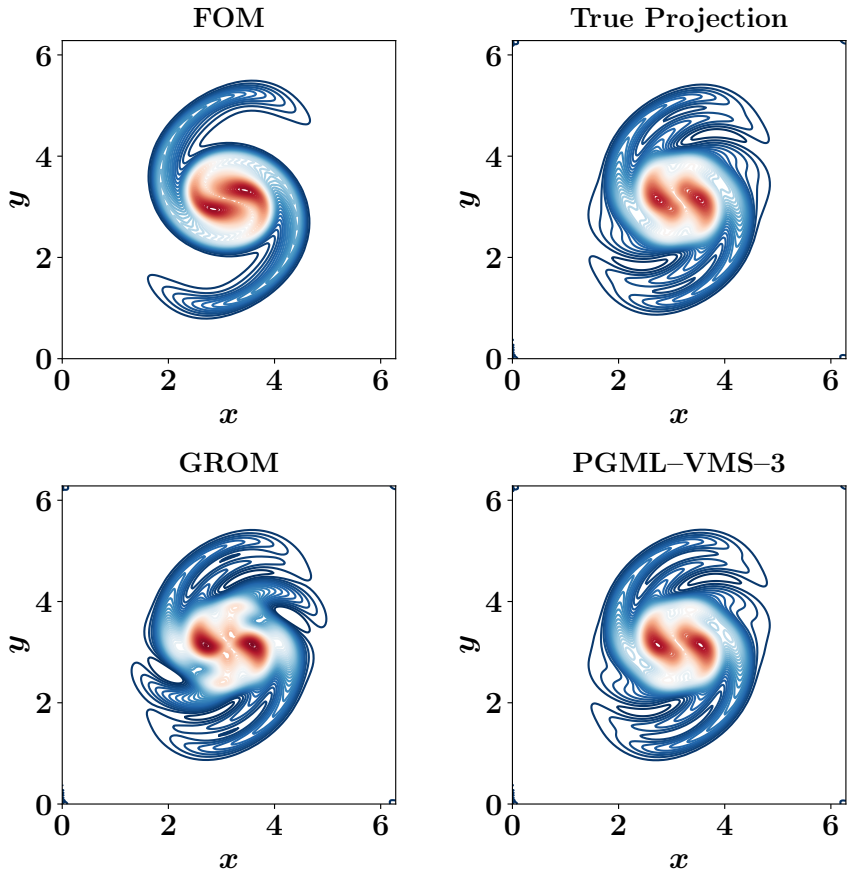


Figure 3.12: Comparison between the FOM vorticity field at the final time (i.e., $t = 30$) and the reconstruction from true projection (i.e., optimal reconstruction), GROM, and PGML-VMS-3. Note that the PGML-VMS-3 field is very similar to the true projection field, which implies that the closure error 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.

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.

3.8 Conclusions

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 multi-level variational multi-scale (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

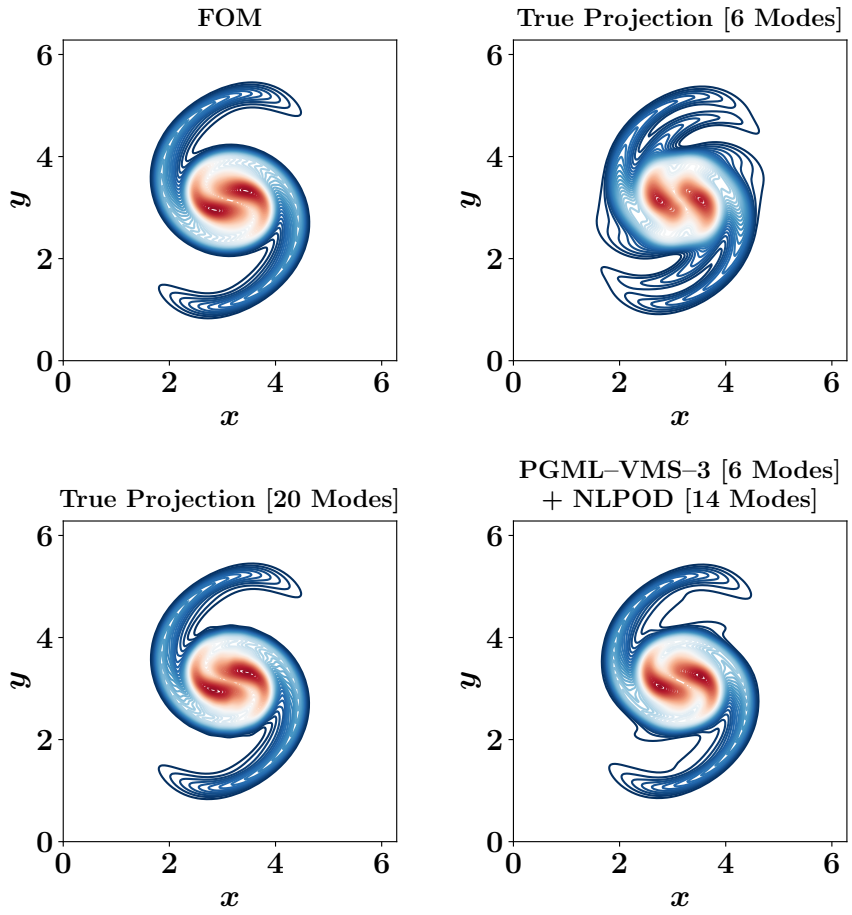


Figure 3.13: Comparison between the FOM vorticity field at final time (i.e., $t = 30$) and the reconstruction from true projection (i.e., optimal reconstruction) at two different values of modal truncation as well as the predictions of the PGML-VMS-3 for the dynamics of the first 6 modes, augmented with NLPOD for the following 14 modes (i.e., a total of $K = 20$ modes) to reduce the projection error.

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

Table 3.1: 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 similar 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

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 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 in 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.

Part II

Interface Learning

CHAPTER 4

Interface Learning of Multi-Physics and Multi-Scale Systems

4.1 Abstract

Complex natural or engineered systems comprise multiple characteristic scales, multiple spatiotemporal domains, and even multiple physical closure laws. To address such challenges, we introduce an interface learning (IL) paradigm and put forth a data-driven closure approach based on memory embedding to provide physically correct boundary conditions at the interface. To enable IL for hyperbolic systems by considering the domain of influence and wave structures into account, we put forth the concept of *upwind learning* towards a physics-informed domain decomposition. The promise of the proposed approach is shown for a set of canonical illustrative problems. We highlight that high-performance computing environments can benefit from this methodology to reduce communication costs among processing units in emerging machine learning ready heterogeneous platforms toward exascale era.

4.2 Introduction

Specification of boundary conditions is essential for the accurate solutions of mathematical models representing physical system [180]. Moreover, numerical simulations of multi-scale, multicomponent, multi-physics and multi-disciplinary systems require additional treatment of interface boundary conditions among solvers or heterogeneous computational entities. Otherwise, in a naive implementation, the stiffest part of the domain dictates the overall spatial mesh resolution and time stepping requirements, making such simulations computationally daunting. Most of such hierarchical problems that incorporate some sort of information exchange can be put into the following six categories, explained with examples as follows:

1. Reduced order model - full order model coupling: With the emergence of digital

This chapter is adapted from: *Ahmed, S. E., San, O., Kara, K., Younis, R., & Rasheed, A. (2020). Interface learning of multiphysics and multiscale systems. Physical Review E, 102(5), 053304.*

twin like technologies, there is a demand for lighter models that can run in real time [128, 181]. In the context of weather prediction, the full order model (FOM) has been in use for a long time; however, they are incapable of modeling phenomena associated with scales smaller than what the coarse mesh can handle (like buildings and small terrain variations). These fine scale flow structures can be modeled using a much refined mesh but then the simulations become computationally intractable. To tackle this problem, a large variety of reduced order models (ROMs) are being developed. In order to make these ROMs realistic, there is a need to couple them to the FOM model so that the interface conditions (both in space and time) are exchanged between the FOM and ROM.

2. Multi-physics and multi-scale coupling: Various flow dichotomies with a multi-physics coupling of interacting subsystems can be identified in many scientific and engineering applications [182]. For instance, in a gas turbine flow, the rotating parts and wall turbulence largely govern the flow within compressor and turbine sections. On the other hand, for the flow within the combustor, chemical reactions, heat release, acoustics, and the presence of fuel spray come into play. Thus, a simulation of the flow within the combustion chamber is significantly more expensive and demanding than other sections. It would require a finer and more sophisticated mesh, smaller time step, and less numerical simplifications. Therefore, using a unified global solver for the whole system would be either too expensive (matching the level of the fidelity required for the more complex part), or unacceptably inaccurate (following the level of fidelity required for the inexpensive part). Instead, multiple solvers are usually utilized to address different parts [183–185], and information is transferred between solvers. Another common example in this category might be the use of a particle based approach in part of the domain, while using a continuum approach in the rest of the domain [186, 187].

3. Geometric multi-scale: One example is the blood flow in the whole circulatory system which is mathematically described by means of heterogeneous problems featuring different degrees of detail and different geometric dimensions that interact together through appropriate interface coupling conditions. Proper exchange of interface conditions between models operating at different geometric approximations opens altogether new vistas for biofluids simulations [188–190]. Such multi-dimension modeling has been also promoted in porous media flows [191–193].

4. Model fusion: Turbulence modeling generally requires an apriori selection of the most suited model to handle a particular kind of flow. However, it is seldom that

one model is sufficient for different kind of zones in the computational domain. To alleviate this problem, hybrid and blending models have been extensively utilized to lift technical barriers in industrial applications, especially in settings where the Reynolds-averaged Navier-Stokes (RANS) approach is not sufficient and large eddy simulation (LES) is too expensive [194–196]. The approach can be extended to blend any number of turbulence models provided the exchange of information at the interface can be accurately modeled [197].

5. Nested solvers: To decrease the computational cost required for an accurate representation of the numerous interconnected physical systems, e.g., oceanic and atmospheric flows, several classes of nested models have been developed and form the basis of highly successful applications and research at numerous weather and climate centers. Enforcing consistent flow conditions between successive nesting levels is also considered one form of interface matching. For example, a spectral nudging approach has been successfully implemented to force the large-scale atmospheric states from global climate models onto a regional climate model [198–203].

6. Domain decomposition: Since various zones in multi-scale systems are connected through interfaces, data sharing, and communicating consistent interface boundary conditions among respective solvers are inevitable [204]. Likewise, multi-rate and locally adaptive stepping methods can yield a mismatch at the space-time interface, and simple interpolation or extrapolation might lead to solution divergence or instabilities [205]. An analogous situation usually occurs in parallel computing environments with domain decomposition and distribution over separate processors with message passing interface to communicate information between processors. The heterogeneity of different processing units creates an asynchronous computational environment, and the slowest processors will control the computational speed unless efficient load-balancing is performed [206, 207].

In short we can conclude that developing novel methodologies to model the information exchange at the interface will have far reaching impacts on a large variety of problems as shown in Fig. 4.1. To this end, the current chapter puts forth an approach based on memory embedding via machine learning to provide physically correct interfacial conditions. In particular, the proposed technique relies on the time history of local information to estimate consistent boundary conditions at the sub-domain boundaries without the need to resolve the neighboring regions (on the other side of the interface). It enables us to focus our computational resources on the region or scales of interest. We first present proof-of-concept computations on a bi-zonal

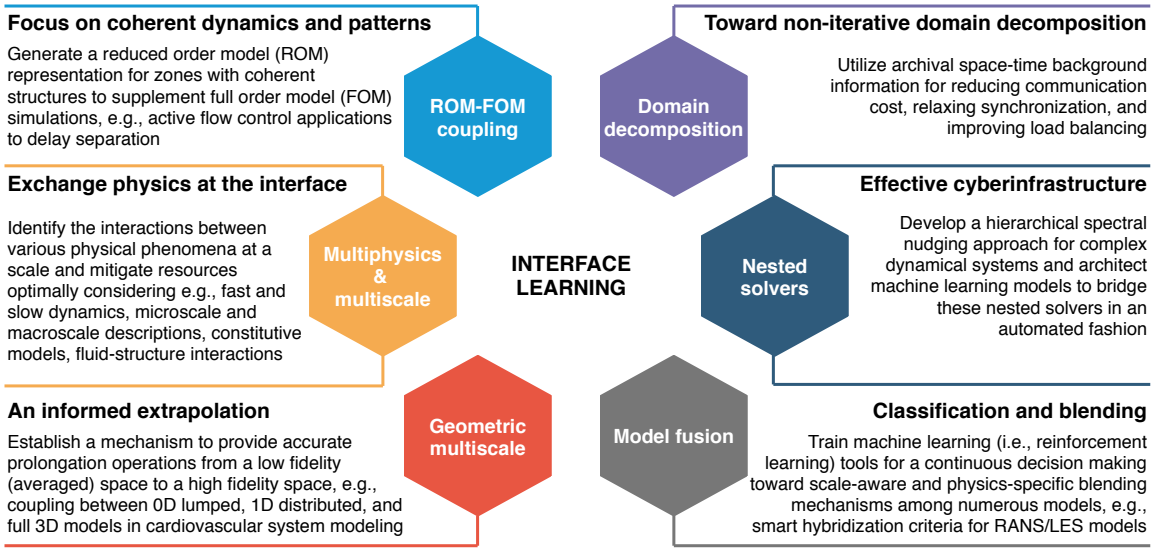


Figure 4.1: Overview of the interface learning paradigm considering numerous scientific and engineering interpretations.

one-dimensional Burgers' problem to showcase the proposed approach's promise for stiff multi-scale systems. Moreover, we demonstrate that upwinding ideas can be easily incorporated in the IL framework to make the proposed approach physically more consistent with the underlying characteristics and wave structures in hyperbolicity dominated systems. With this in mind, we then propose the *upwind learning* approach toward establishing an eclectic framework for physics-informed data-driven domain decomposition approaches. The efficacy of the upwind learning is revealed using a set of canonical problems, including the hyperbolic Euler equations and pulsed flow equations. We also highlight that high-performance computing environments can benefit from this methodology to reduce communication costs among processing units.

4.3 Two-Component System

As a first demonstration of IL, we consider an application to the one-dimensional (1D) viscous Burgers problem. It combines the effects of viscous diffusion, friction, and nonlinear advection, and thus serves as a prototypical test bed for several numerical simulations studies. In order to mimic multi-scale and multi-physics systems, we suppose the domain consists of two distinguishable zones corresponding to different

physical parameters as follows,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} - \gamma u, \tag{4.1}$$

$$(\nu, \gamma) = \begin{cases} (10^{-2}, 0), & \text{for } 0 \leq x \leq x_b, \\ (10^{-4}, 1), & \text{for } x_b < x \leq 1, \end{cases} \tag{4.2}$$

where x_b is the spatial location of the interface. In a naive implementation, a numerical solution of this problem would imply the use of a grid resolution and time step corresponding to the stiffest part (i.e., the left zone in this case) all over the domain unless we adopt an implicit scheme which is unconditionally stable but requires a nonlinear solver typically at each time step. Certainly, this puts an excessive and unnecessary computational burden. For instance, if we opt to using a spatial resolution of 4096 grid spacings with a simple forward in time central in space (FTCS) finite difference scheme, the maximum time step that can be used in the left zone is approximately 2.5×10^{-6} (i.e., $\delta t \leq \delta x^2/(2\nu)$ based on von Neumann stability analysis). On the other hand, the right zone gives the flexibility of using two orders of magnitude larger time step. However, resolving the whole domain simultaneously would dictate the smaller time step, even if we are only interested in the right zone. A similar scenario would take place in multi-component systems with varying spatial grid resolutions, where a unified resolution all over the domain becomes unpractical. Thus, we explore the introduction of a memory embedding architecture to enable resolving the zone of interest independently of the rest of the domain.

4.4 Memory Embedding of Interface Boundaries

For machine learning applicability, a pattern must exist and most fluid flows are dominated by coherent structures. Thus, our underlying hypothesis is the existence of a dynamical context or correlation between the time history of flow features at the interface in addition to the interactions with its one-sided neighbors (i.e., $u(x_b, t_n)$, $u(x_b+\delta x, t_n)$, $u(x_b+2\delta x, t_n)$, $\dots$), and the future state at the interface (i.e., $u(x_b, t_n+\delta t)$). This corresponds to the *Learn from Past* (LP) model in Fig. 4.2. Since we incorporate a fully explicit time stepping scheme in our simulation, the interface neighboring points might be evolved in time before the interface condition is updated (e.g., using locally-frozen boundary conditions). Thus, a variant of the LP model based on a combination between old and updated values, namely the *Learn from*

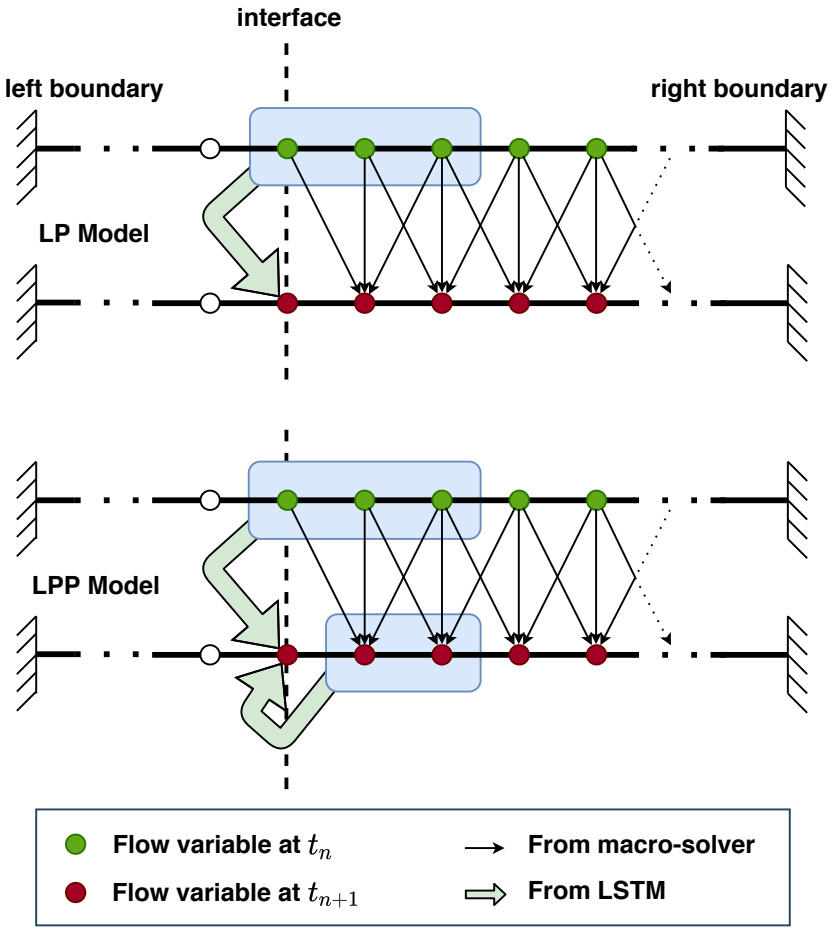


Figure 4.2: Different models to utilize LSTM mapping for learning boundary conditions at the interface

Past and Present (LPP) model, can be utilized as well. Furthermore, we extend this mapping to take into account the time history dependence in a non-Markovian manner through the adoption of recurrent neural networks. Those exploit an internal state feature that reserves information from past input to learn the *context* to improve and refine the output. For the neural network architecture, we use a simplistic long short-term memory (LSTM) of two layers, 20 neurons each. Although more sophisticated ML architectures and/or numerical schemes might be utilized (e.g., [13, 208–210]), the main objective of the present study is to emphasize the potential of neural networks to advance computational fluid dynamics (CFD) simulations for multi-scale and multi-component systems.

4.5 Proof-of-Concept Results

For the demonstration and assessment of the introduced methodology of IL, we first consider two examples of varying complexity for the 1D Burgers problem with quadratic nonlinearity and Laplacian dissipation defined in Eq. (4.1). We then introduce and discuss the need for *upwind learning* to provide physically-aware domain decomposition to address hyperbolicity-dominated systems. For IL, we consider two schemes/models for the training as illustrated in Fig. 4.2 to learn the dynamics at the internal boundary separating the two compartments.

4.5.1 Example 1: travelling square wave

. In this first example, we address the problem of a travelling square wave, where the initial condition is defined with an amplitude of 1 in the left zone (i.e., $0 \leq x \leq x_b$), and zero in the right zone as below,

$$u(x,0) = \begin{cases} 1, & \text{for } 0 \leq x \leq x_b, \\ 0, & \text{for } x_b < x \leq 1. \end{cases}$$

In other words, the interface is placed exactly at the discontinuity location of the initial propagating wave. So, the wave is guaranteed to instantaneously enter the right zone once the flow is triggered. We solve the presented viscous 1D Burgers problem for a time span of $[0, 1]$ using a time step of 2.5×10^{-6} to resolve the whole domain $x \in [0, 1]$ over a spatial grid resolution of 4096. For external boundary conditions, we assume zero Dirichlet boundary conditions (i.e., $u(0,t) = u(1,t) = 0$). Data snapshots are stored every 100 time steps (corresponding to the coarse time step of 2.5×10^{-4}). In particular, we generate data for $x_b \in \{1/8, 2/8, 3/8, 4/8, 5/8, 6/8, 7/8\}$, and we use field data at $x_b \in \{1/8, 3/8, 5/8, 7/8\}$ for training and reserve the remaining cases for the out-of-sample testing.

We demonstrate IL performance by we considering a truncated 1D domain (defined as $x_b \leq x \leq 1$) and resolve the flow dynamics in this portion using a coarse time step of 2.5×10^{-4}, denoted the macro-solver here. We adopt the LSTM learning to update the left boundary condition (i.e., at $x = x_b$). For the right boundary (i.e., at $x = 1$), we keep the standard zero Dirichlet conditions. To enhance the neural network performance, we augment the input vector with the spatial and temporal information as well. In other words, LP model can be interpreted as the mapping $u(x_b, t_n +$

$\delta t) = G_1\big(u(x_b, t_n), u(x_b + \delta x, t_n), u(x_b + 2\delta x, t_n), \ldots, x_b, x_b + \delta x, x_b + 2\delta x, \ldots, t_n\big)$, while LPP scheme learns the map $u(x_b, t_n + \delta t) = G_2\big(u(x_b, t_n), u(x_b + \delta x, t_n), u(x_b + 2\delta x, t_n), \ldots, u(x_b + \delta x, t_n + \delta t), u(x_b + 2\delta x, t_n + \delta t), \ldots, x_b, x_b + \delta x, x_b + 2\delta x, \ldots, t_n, t_n + \delta t\big)$.

We compare the predicted velocity field within the truncated domain using the proposed LSTM boundary condition (BC) closure approach with respect to the true solution obtained by solving the whole domain. We note here that the LSTM BC closure results are based on utilizing the macro-solver (i.e., using a time step of 2.5×10^{-4}), while the true solution is obtained by adopting the micro-solver (i.e., using a time step of 2.5×10^{-6}). The spatio-temporal evolution of the velocity field for $x_b \in \{1/4, 2/4, 3/4\}$ is shown in Fig. 4.3, where x_b is the location of the interface. We note that Fig. 4.3 corresponds to a three-point stencil for the LSTM mapping. In other words, the LP model uses values of $u(x_b, t_n)$, $u(x_b + \delta x, t_n)$, and $u(x_b + 2\delta x, t_n)$ for the prediction of $u(x_b, t_n + \delta t)$, while the LPP model uses $u(x_b, t_n)$, $u(x_b + \delta x, t_n)$, $u(x_b + 2\delta x, t_n)$, $u(x_b + \delta x, t_n + \delta t)$, and $u(x_b + 2\delta x, t_n + \delta t)$. Visual results advocate the capability of the presented approach of predicting accurate values for the interface boundary condition at different times.

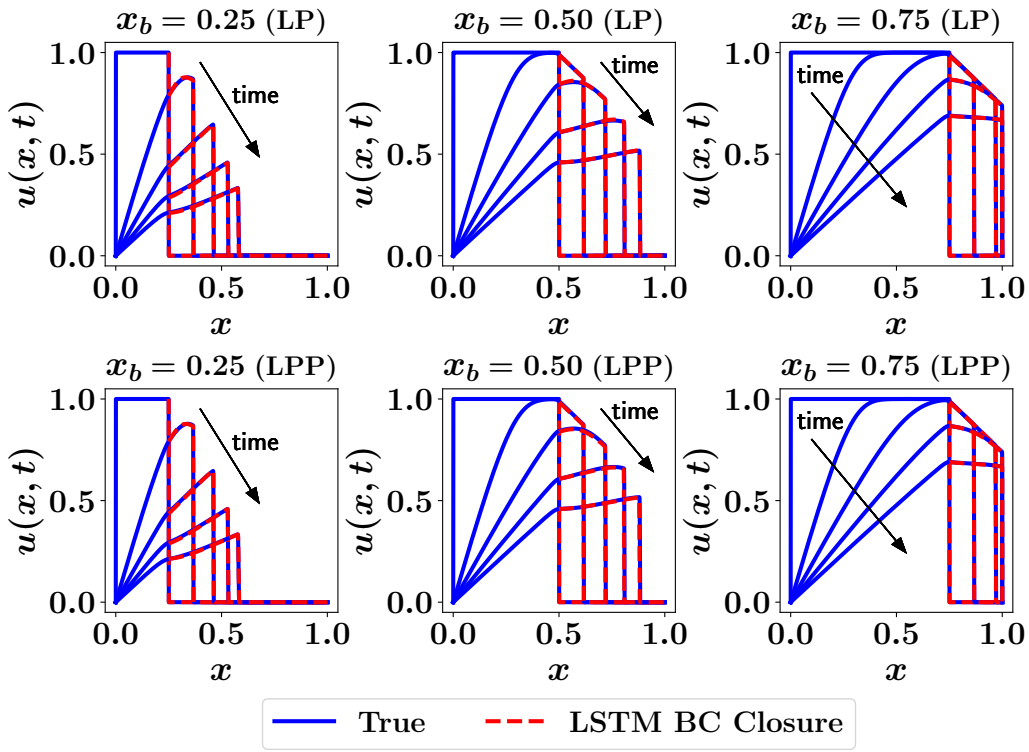


Figure 4.3: Results for LSTM boundary condition closure for different values of x_b. Predicted velocity fields are shown at $t \in \{0.0, 0.25, 0.50, 0.75, 1.0\}$.

For more quantitative assessment, we also compute the resulting root mean-squares error (RMSE) defined as

$$\text{RMSE} = \sqrt{\frac{1}{N_t N_x} \sum_{n=1}^{N_t} \sum_{i=1}^{N_x} \left(u^{\text{T}}(x_i, t_n) - u^{\text{P}}(x_i, t_n)\right)^2}, \tag{4.3}$$

where u^{T} is the true velocity field, and u^{P} represents the predictions by the LSTM BC closure approach. In the above formula, N_x stands for the number of grid points involved only in the truncated domain. In other words, it considers only the flow field values within $[x_b, 1]$.

The RMSE values of the LSTM BC closure predictions using a two- and three-point stencils are documented in Table 4.1 using the LP and LPP models. Quantitative results imply that the LP model is giving slightly better results than the LPP model. We believe that this behavior is because the LP model is more consistent with the adopted explicit numerical scheme, where the time evolution relies solely on the *old* values of the flow field. Moreover, this might be attributed to the sub-optimal architecture we use for the LSTM. Although we found that results are not very sensitive to the given *hyper-parameters*, further tuning might be required to provide *optimal* performance. We also see from Table 4.1 that an increase in the stencil size from 2 to 3 improves results. Nonetheless, a 2-point stencil mapping still provides acceptable predictions, confirming the validity and robustness of the LSTM memory embedding skills to yield physically consistent and accurate state estimates at the interface using local information, and may hold immense potential for designing ML-ready predictive engines in physical sciences.

Table 4.1: RMSE of LSTM boundary condition closure results using different models with two-point and three-point mapping.

	LP Model		LPP Model	
x_b	2 Points	3 Points	2 Points	3 Points
0.125	1.5×10^{-2}	2.9×10^{-3}	2.6×10^{-2}	1.4×10^{-2}
0.250	4.4×10^{-2}	3.5×10^{-3}	2.8×10^{-2}	3.4×10^{-3}
0.375	8.4×10^{-3}	3.6×10^{-3}	2.0×10^{-2}	1.6×10^{-2}
0.500	2.9×10^{-2}	2.6×10^{-3}	1.7×10^{-2}	2.3×10^{-2}
0.625	6.4×10^{-3}	3.7×10^{-3}	1.9×10^{-2}	1.3×10^{-2}
0.750	1.0×10^{-2}	5.4×10^{-3}	1.2×10^{-2}	1.7×10^{-3}
0.875	2.1×10^{-3}	1.8×10^{-3}	6.7×10^{-3}	4.4×10^{-3}

4.5.2 Example 2: pulse problem

. In a second example of increasing complexity, we study the evolution of a pulse wave completely contained in a portion of the left region. The initiation of flow dynamics in the truncated domain is controlled by the interplay between advection, diffusion, and friction in different regions. Specifically, we consider an initial condition of a pulse, completely contained in the left region and study its propagation and travel from the left to right compartments. In particular, the initial pulse can be represented as

$$u(x,0) = \begin{cases} 1, & \text{for } 0 \leq x \leq w_p, \\ 0, & \text{for } w_p < x \leq 1, \end{cases}$$

where w_p is the pulse width. For illustration, we store results corresponding to 7 varying pulse widths as $w_p \in \{0.20, 0.21, 0.22, 0.23, 0.24, 0.25, 0.26\}$. The same numerical schemes and resolutions of Example 1 are adopted here. Data corresponding to $w_p \in \{0.20, 0.22, 0.24, 0.26\}$ are used for training and validation, while we assign $w_p \in \{0.21, 0.23, 0.25\}$ for out-of-sample testing. For interface, we consider a fixed interface location at $x_b = 0.3$ (i.e., on the right of the largest pulse width). This is to let the interplay between the different interacting mechanisms (i.e., advection, diffusion, and friction) to come into effect *before* the pulse travels into the truncated zone. Thus the state at the interface is more dependent on the flow dynamics in *both* domain partitions. Since the pulse width is a key factor in this problem setting, we augment our input vector with w_p as well. For this particular example, we found that enforcing higher memory embedding is crucial in providing accurate results. Specifically, we adopt a sliding window of a three-time step (also called a lookback of 3) in our LSTM implementation [211]. Results for the LP and LPP schemes are shown in Fig. 4.4 using 3-point mapping. We find that both schemes can sufficiently learn the interface dynamics and accurately predict its condition at out-of-sample settings.

Although the wave in the previous examples moves from left to right, both LP and LPP are able to predict the interface conditions from the right-sided neighbors. In other words, the upstream conditions are inferred from the time history of the downstream flow. However, we highlight that this was greatly feasible due to the significant dissipative (viscous) effects that enable the rapid dissipation of information across the whole domain. Therefore, as soon as the solver is initialized within the right sub-domain, the incoming wave is already *felt* downstream. That is why a

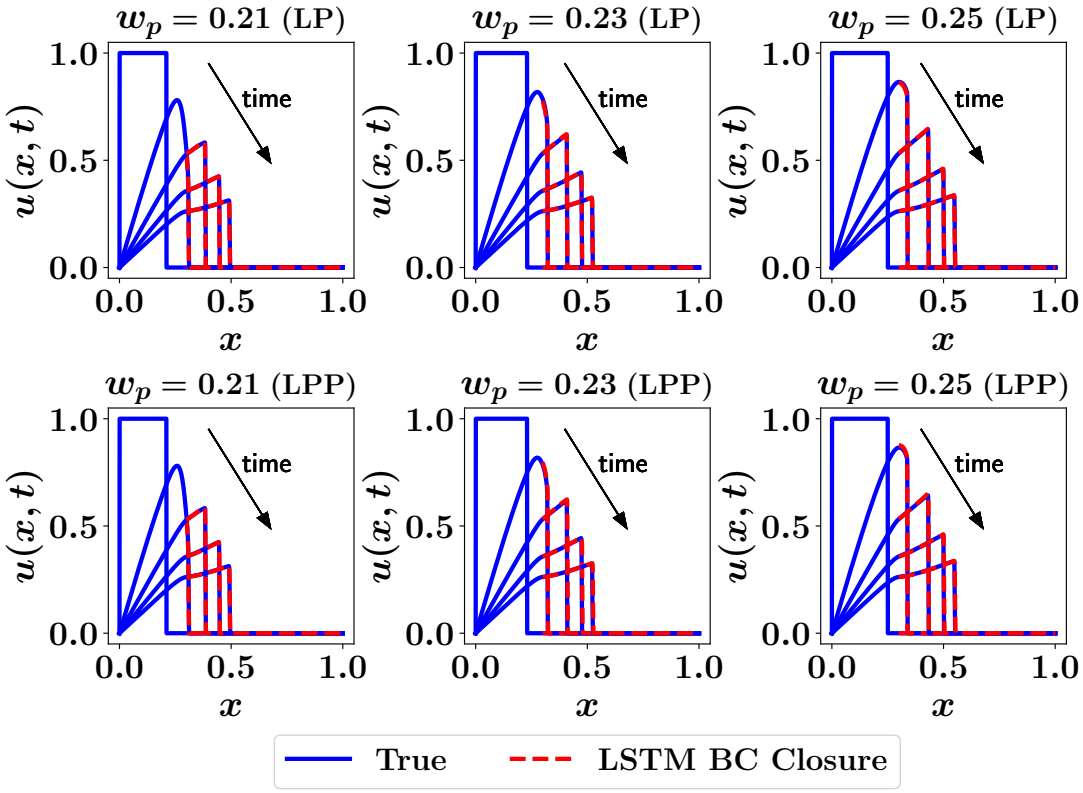


Figure 4.4: Results for LSTM boundary condition closure for the pulse problem using different values of w_p. Predicted velocity fields are shown at $t \in \{0.0, 0.25, 0.50, 0.75, 1.0\}$.

deeper sliding window was needed in Example 2 to allow the effect of the pulse to pass to the right zone. On the other hand, in the hyperbolic limit, this cannot be established. For instance, if we look at the linear advection problem given as $u_t + au_x = 0$ (where a is the constant wave speed), we know that the solution can be written as $u(x,t) = u(x - at, 0)$. This means that the specification of the interface condition $u(x_b, t)$ relies on the information from the direction where the wave is coming from (e.g., for positive a, we need information from the left sided neighbors).

This is one limitation of the presented interface learning framework. In order to mitigate and treat this issue, we put forth an *upwind learning* methodology to enforce physics into the learning process. For such, the domain decomposition procedure is performed to allow upwind learning from the upstream neighbors. In other words, instead of solving the whole domain, we replace the insignificant downstream subcomponents by an LSTM architecture and infer their effects from the time history of upstream flow dynamics. Thus, the main purpose of the upwind learning framework is to account for the effects of downstream domain with an ML model that provides

the interface condition in order to be able to solve for the upstream domain. We illustrate the upwind learning methodology using a gas dynamics flow problem governed by the hyperbolic Euler equations as well as a pulsatile flow through a network of branched elastic tubes.

4.5.3 Example 3: Euler equations of gas dynamics

. In this example, we apply the proposed upwind learning concept on the Sod's shock tube problem [212], considering a long one-dimensional (1D) tube, closed at its ends with a thin diaphragm dividing the tube into two regions. The governing one dimensional Euler equations can be written in a conservative form as below

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho e\end{bmatrix}+\frac{\partial}{\partial x}\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho u h\end{bmatrix}=0, \tag{4.4}$$

where ρ is the density, u is the horizontal component of the velocity, e is the internal energy, $p=\rho(\gamma-1)(e-u^2/2)$ is the pressure and $h=e+p/\rho$ is the static enthalpy. Here, the ratio of specific heats is set to $\gamma=7/5$. The two regions are initially filled with the same gas, but with different thermodynamic parameters, as follows,

$$(\rho,p,u)=\begin{cases}(1.0,1.0,0.0), & \text{for } 0\le x\le 0.5,\\ (0.125,0.1,0.0), & \text{for } 0.5<x\le 1,\end{cases}$$

with Dirichlet boundary conditions at $x=0$ and $x=1$. Data are collected for the time evolution from $t=0$ to $t=0.2$ using a time step of 10^{-4} and spatial step size of 2.5×10^{-3}. As a result of the diaphragm breakage, a contact discontinuity and a shock wave move from the left the right, while a rarefaction (expansion) wave moves from the right to the left.

First, we apply the interface learning approaches with the interface at $x_b=0.5$ and consider the left zone to demonstrate the uplift learning concept. Results at final time (i.e. $t=0.2$) with both the LP and LPP implementations are given in Fig. 4.5, which shows the success of the upwind learning framework to infer valid boundary conditions at the interface from the dynamics and flow pattern of its left-sided neighbors.

In order to understand the performance of the upwind learning, we recall that the directions of characteristics (i.e., the curves $dx/dt=\lambda$ along which the Riemann

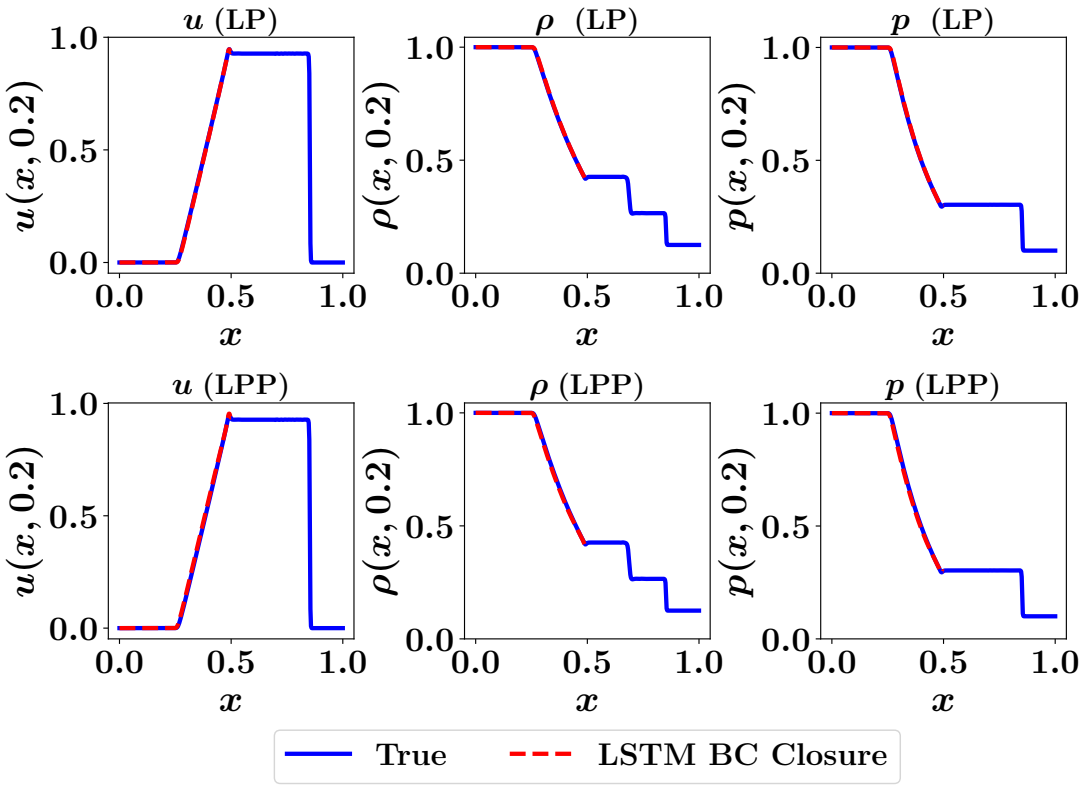


Figure 4.5: Results for LSTM boundary condition closure for the Sod's shock tube problem at time $t = 0.2$ with LP (top) and LPP (below) implementations.

invariants are constant) are defined as $\lambda_1 = u$, $\lambda_2 = u - a$, and $u + a$, where a is the local speed of sound given as $a = \sqrt{\gamma(p/\rho)}$. We plot the space-time contour plots of u, $u-a$, and $u+a$ in Fig. 4.6, along with the line plot of the characteristics directions at the interface location. We observe that for $x = 0.5$, u and $u + a$ are always non-negative, while $u-a$ is initially negative (since the gas is initially at rest), but quickly approaches zero. In other words, the majority of the information at $x = 0.5$ flows from left to right, and hence the success of the upwind learning scheme. Moreover, we remark that the interface conditions are almost constant with time, except for the initial transition period following the breakdown of the diaphragm, which minimizes the computational burden on the LSTM model.

It can be seen from Fig. 4.6 that moving the interface to the left will result in $u - a$ being significantly negative. In other words, some of the information should be inferred from the truncated region and the ML model has to account for this contribution. Indeed, we find that the same architecture with a single step lookback suffers in learning the interface conditions at $x_b = 0.4$. However, augmenting the upwind learning framework with an increased time history of the modeled quantities

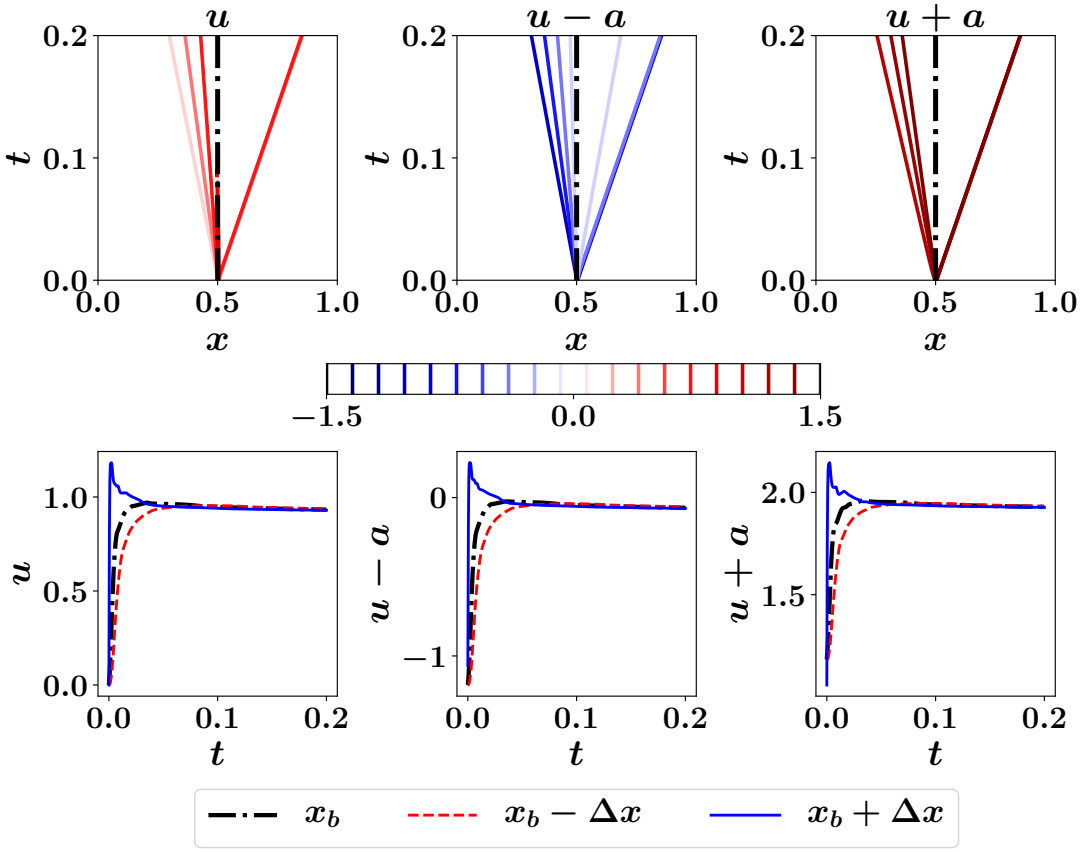


Figure 4.6: Contour plots of characteristics directions (top) as well as their time variation at $x_b = 0.5$ (bottom) for the Sod's shock tube problem.

(i.e., using a lookback of 3 steps) significantly improves the predictive capability as seen in Fig. 4.7. That is enforcing a deeper time dependence facilitates learning the missing information due to domain truncation. Figure 4.7 also depicts the spatial distribution of the velocity u, the density ρ and the pressure p at final time using the LP approach, augmented with a history of 3 time steps.

4.5.4 Example 4: Fluid structure interaction in network flows

. To demonstrate the feasibility of the upwind interface learning in network domains, we construct a bifurcating flow in elastic tubes. This system is ubiquitous in cardiovascular system modeling [213, 214] and open channel networks [215, 216], and it is often represented by the Saint-Venant equations. For such problems, a boundary closure issue appears at the bifurcation points and constitutive relations have to be imposed at these locations. Most often, this yields a system of nonlinear equations, which has to be solved with iterative schemes (e.g., Newton-Raphson method). This

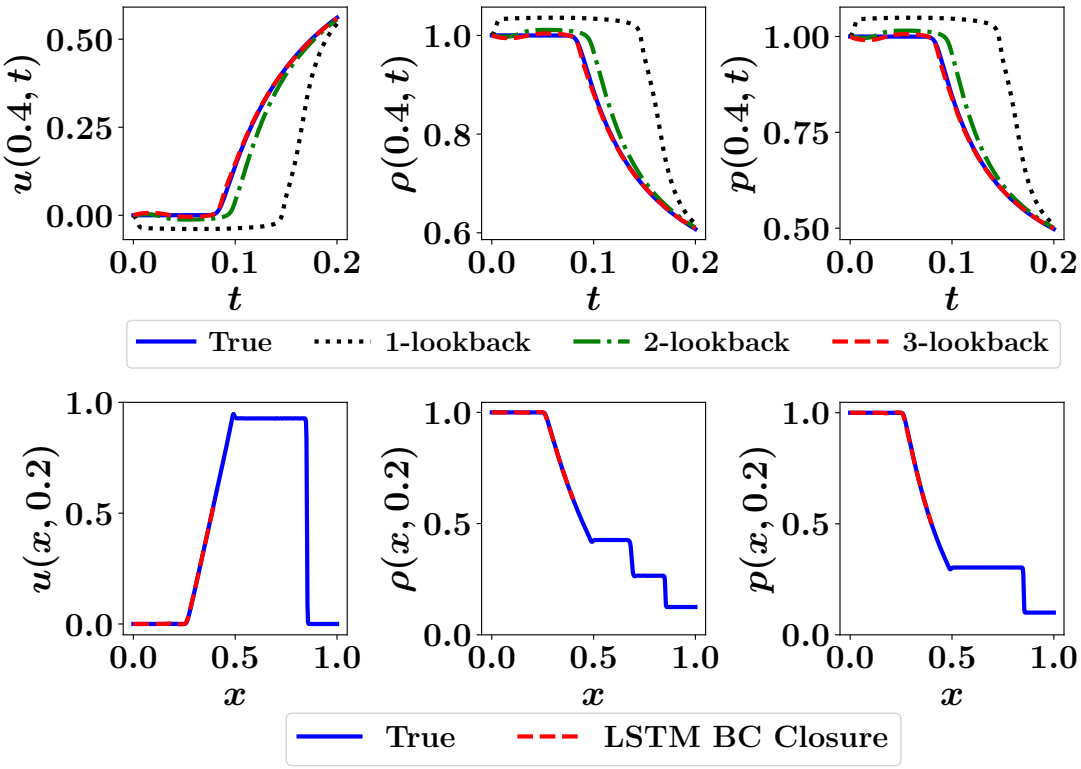


Figure 4.7: Time evolution of velocity, density, and pressure at $x_b = 0.4$ with different lookback lengths (top), and their spatial distribution at final time with the LP approach using a lookback of 3 steps (bottom).

incurs an additional computational cost to solve the system of nonlinear equations. It also requires careful selection of the numerical scheme to solve this system in order to guarantee its convergence after a few iterations. Therefore, adopting the interface learning technique for these systems has the potential to address the bifurcation points treatment. Moreover, when the network grows largely, a large number of segments have to be considered simultaneously. However, with interface learning, the insignificant downstream segments can be truncated and their effects are modeled by the adopted ML architecture.

In this example, we consider a 3-segment network with a single bifurcation point with one mother (upstream) segment and two daughter (downstream) segments, governed by the following pulsed flow equations

$$\frac{\partial A}{\partial t} + \frac{\partial (uA)}{\partial x} = 0, \tag{4.5}$$

$$\frac{\partial u}{\partial t} + \alpha u \frac{\partial u}{\partial x} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu \frac{\partial^2 u}{\partial x^2} - \beta \pi \nu \frac{u}{A}, \tag{4.6}$$

where ρ and ν are the density and kinematic viscosity of the fluid, and α and β are parameters depending on the assumed *radial* velocity profile [217]. Here, $\rho = 10^3$ kg/m^3, $\nu = 10^{-6}$ m^2/s, $\alpha = 1$, and $\beta = 8$. A linear theory of elasticity can be used to relate the pressure and cross sectional area via $p = p_0 + 2\rho c_0^2(1 - \sqrt{\eta})$, where c_0 is the wave propagation speed prescribed by the Moens-Korteweg equation, $c_0 = \sqrt{Eh/(2\rho R_0(1 - v^2))}$, and $\eta = A_0/A$ with $A_0 = \pi R_0^2$ being the nominal reference value of cross sectional area when the pressure is p_0. Here, E, v, and h refer to the elastic modulus, Poisson ratio and thickness of the tube. A schematic diagram of the problem setup we are solving is depicted in Fig. 4.8, where the length (L) and nominal radius (R_0) of each segment are given in Table 4.2. At bifurcation, six quantities become unknown (i.e., the ending cross sectional area and velocity of mother segment and the starting cross sectional area and velocity for each daughter branch. Continuity and total pressure equivalence constitute three equations, and the remaining three equations might come from characteristics (i.e., Riemann invariants). For pulsating flow equations, the Riemann invariants are approximated as follows [217],

$$q^{\pm} = u \pm 4c_0[1 - \eta^{1/4}], \tag{4.7}$$

where the plus-minus sign defines the direction of the characteristics. Within the mother segment, the information moves from the interior points to the bifurcation point through the right-travelling wave (i.e, with positive sign). Similarly, within each daughter branch, information flows from interior points (i.e, right-hand side points) to the bifurcation point via the backward-travelling wave (i.e., with negative sign). This forms a total of six nonlinear equations that can be solved using Newton-type methods as discussed above.

Initial conditions read as $A(x, 0) = A_0$ and $u(x, 0) = 0$ for all segments. A composite Gaussian and triangular input wave signal is given as a boundary condition to the mother segment as follows:

$$u(0, t) = 0.05 \max\left(e^{-k_1(t-k_2)^2}, 1 - \left|\frac{t - k_2 - 3.5k_3}{k_3}\right|, 0\right), \tag{4.8}$$

$$A(0, t) = A_0\left(1 - \frac{u(0, t)}{4c_0}\right)^{-4}, \tag{4.9}$$

where $k_1 = 10^4$ s^{-2}, $k_2 = 0.05$ s, and $k_3 = k_2/3$ and reflecting outflow boundary conditions are imposed at the end of daughter segments. The second order Lax-Wendroff

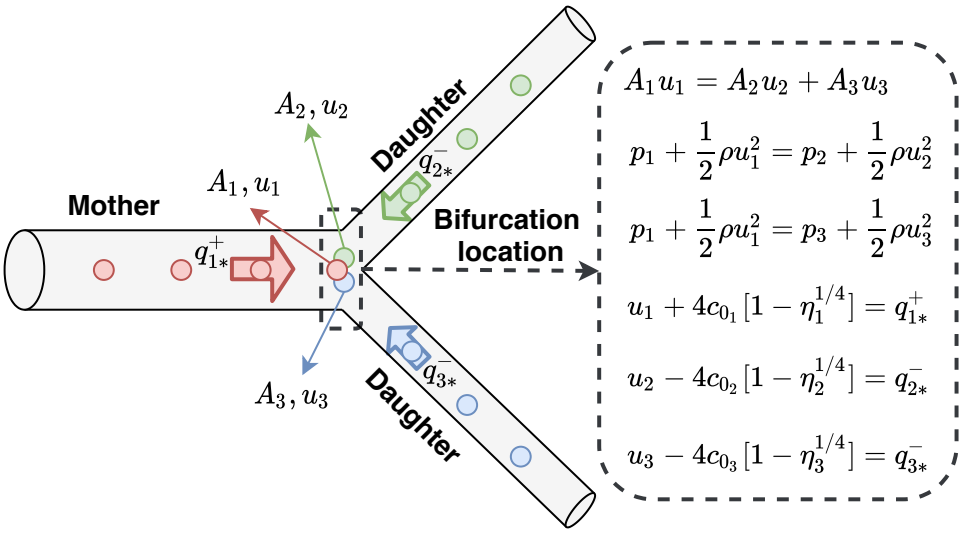


Figure 4.8: Problem setup for the bifurcating flow example. Subscripts 1, 2, and 3 refer to the mother, first daughter, and second daughter, respectively, while the subscript $*$ denotes the nearest point to the bifurcation location (i.e., the second grid point for the daughter branches and second-to-last grid point for the mother segment).

scheme is followed to collect equispaced $40,000$ time snapshots for a maximum time of 0.4 collected with a spatial step size of 10^{-4}. For the upwind learning, we only solve for the mother branch and use an LSTM at the bifurcation point to model the effects of the daughter branches. The obtained results for the cross sectional area and axial velocity are given in Fig. 4.9 for the LP implementation in order to demonstrate the feasibility of the proposed approach. Once the mother segment bifurcates to the daughters of its half radius, the wave with one-third of its speed will be reflected from the bifurcation point. As shown in Fig. 4.9, the reflected pulse has been modeled accurately by LSTM boundary conditions (notice the middle and right panels of the bottom row).

Table 4.2: Properties of the 3-segment branching network.

Segment	L (m)	R_0 (cm)	E (MPa)	h (cm)	v
Mother	1.0	1.0	0.4	0.10	0.5
Daughter	1.0	0.5	0.4	0.05	0.5
Daughter	1.0	0.5	0.4	0.05	0.5

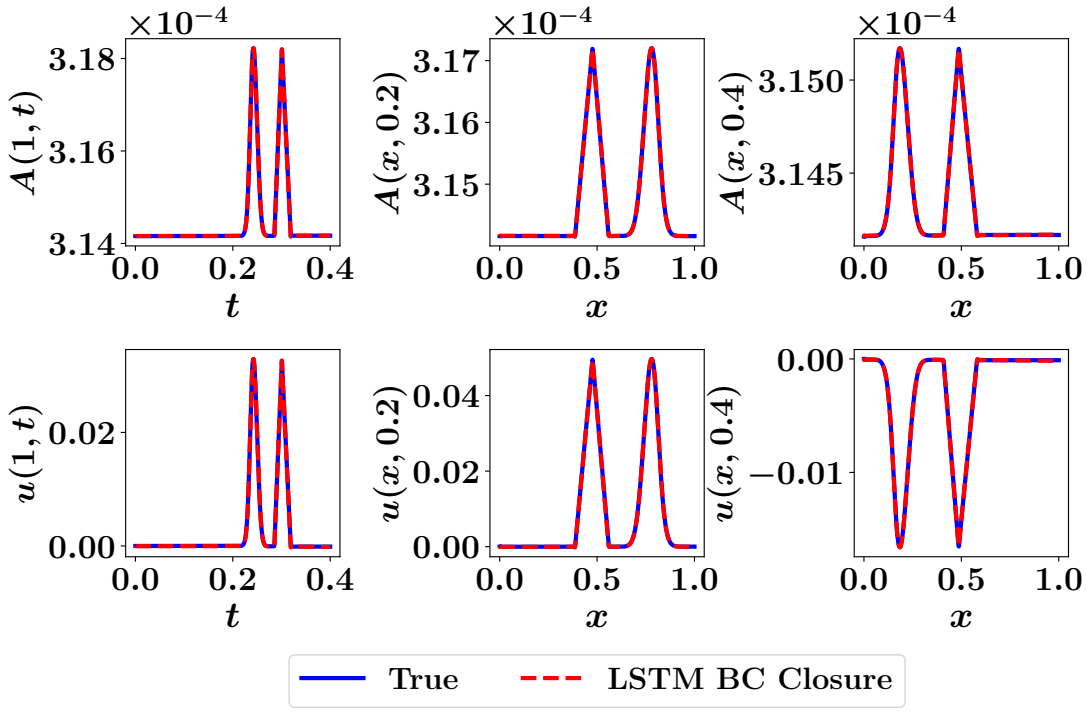


Figure 4.9: Results for cross sectional area (top) and axial velocity (bottom) in the mother branch for the flow through a network of branched elastic tubes using the upwind learning compared to reference values obtained by solving the whole network and the bifurcation point. Subfigures in the middle and right columns illustrate the wave structures at $t = 0.2$ s (before reaching to the bifurcation point) and $t = 0.4$ s (after reflected from the bifurcation point), respectively.

4.6 Conclusions

In this work, we demonstrate the potential of machine learning tools to advance and facilitate CFD simulations of multi-scale, multi-component systems. In particular, we show the capability of memory embedding to learn the dynamics at the interface between different zones. This is especially beneficial where the domain contains zones with strong dynamics and components with complex configuration that might dictate a very fine mesh resolutions and time stepping. The proposed approach enables us to focus our efforts onto the domain portion of interest, while satisfying physically consistent interface conditions. It can serve as a non-iterative domain decomposition method. Toward model fusion technologies, such an interface learning methodology might also hold significant promise for the development of blending criteria in hybrid RANS/LES models. A proof-of-concept is first demonstrated using the 1D viscous Burgers equation over a two-zone domain with different physical parameters. An LSTM is used to bypass the micro-solver corresponding the stiff region and provide

valid interface boundary conditions to enable the macro-solver to run independently. To consider hyperbolicity and wave structures, we furthermore propose the concept of *upwind learning* towards a physics-informed domain decomposition, with illustrations using a shock tube problem in gas dynamics and a fluid-structure interaction application in network flows. We illustrate the success and robustness of the proposed methodology using different learning configurations. Both LP and LLP models are the key concepts introduced in this chapter, especially for designing intelligent boundary closure schemes, which may bear huge potential in many scientific disciplines. Moreover, we demonstrate that the performance of interface learning architectures can be significantly improved by increasing the length of lookbacks (i.e., enforcing deeper time history). Finally, we emphasize that a similar interface closure technique can be adopted in high performance computing environments, to minimize the communication cost and delay between different asynchronous processors, a topic we would like to pursue further in the future.

CHAPTER 5

Multi-Fidelity Computing for Coupling Full and Reduced Order Models

5.1 Abstract

Hybrid physics-machine learning models are increasingly being used in simulations of transport processes. Many complex multi-physics systems relevant to scientific and engineering applications include multiple spatiotemporal scales and comprise a multi-fidelity problem sharing an interface between various formulations or heterogeneous computational entities. To this end, we present a robust hybrid analysis and modeling approach combining a physics-based full order model (FOM) and a data-driven reduced order model (ROM) to form the building blocks of an integrated approach among mixed fidelity descriptions toward predictive digital twin technologies. At the interface, we introduce a long short-term memory network to bridge these high and low-fidelity models in various forms of interfacial error correction or prolongation. The proposed interface learning approaches are tested as a new way to address FOM-ROM coupling problems solving nonlinear advection-diffusion flow situations with a bifidelity setup that captures the essence of a broad class of transport processes.

5.2 Introduction

Numerical simulations are the workhorse for the design, testing, and implementation of scientific infrastructure and engineering applications. While immense advances in computational mathematics and scientific computing have come to fruition, such simulations usually suffer a curse of dimensionality limiting turnaround. The field of multi-fidelity computing, therefore, aims to address this computational challenge by exploiting the relationship between high-fidelity and low-fidelity models. One such multi-fidelity approach becomes crucial, especially for multi-query applications, such as optimization, inference, and uncertainty quantification, that require multiple model

This chapter is adapted from: *Ahmed, S. E., San, O., Kara, K., Younis, R., & Rasheed, A. (2021). Multifidelity computing for coupling full and reduced order models. PLoS ONE, 16(2), e0246092.*

evaluations in an outer-workflow loop. To this end, sampling-based approaches have been often introduced to leverage information from many evaluations of inexpensive low-fidelity models fused by only a few carefully selected high-fidelity computations. An excellent review of the state-of-the-art multi-fidelity approaches for outer-loop contexts can be found in [128].

In this chapter, we focus on a different type of multi-fidelity formulation targeting domain decomposition type problems that consist of multiple zones with different characteristics as well as multi-physics systems where different levels of solvers are devoted to coupled physical phenomena. A key aspect of the zonal multi-fidelity approach is its ability to handle intrinsic heterogeneous physical properties, varying geometries, and underlying governing dynamics. This heterogeneity can be mild as in aerospace applications with spatially varying parameters. However, in media where there is a permittivity such as in electrostatics or porous media, this might be more pronounced. For example, fluid flow in rock often follows Darcy's law, whereas flow in a fracture is modeled as Poiseuille flow. Moreover, a related process in subsurface flows might include a high fidelity approach around wells (that drive the flow) and a low-fidelity model for subdomains in the interior [218, 219]. This discussion can also be extended to an active flow control problem to elucidate the concept of the *zonal multi-fidelity approach* that we tackle in this work. In general, boundary-layer control poses a grand challenge in many aerospace applications including lift enhancement, noise mitigation, transition delay, and drag reduction. Among many other actuator technologies, blooming jets [220–222] and sweeping jets [223–225] offer new prospective solutions in improving the aerodynamics efficiency and performance of the future air vehicle systems. The size of these actuators is usually orders of magnitude smaller than the length scales of the entire computational domain (e.g., an aircraft wing or tail). Including the full representations of each controller's internal flow dynamics in a comprehensive numerical analysis of the entire system is an extremely daunting approach [226]. Meanwhile, the effective flow physics of these actuators can often be accurately characterized by a latent reduced order space due to the existence of strong coherent structures such as quasi-periodic or time-periodic shedding, pulsation, or jet actuation. Therefore, in practice, those flow actuators can be modeled by considering a reduced order surrogate coupled and tied to the global simulation of the whole wing or tail [227, 228].

The above examples illustrate that different levels of models and descriptions can be devoted to different zones and components of the problem in order to allocate

computational resources more effectively and economically. This might be the case for many other coupled multi-physics systems, such as geometric multi-scale [188, 229, 189, 190, 192, 193] and heterogeneous multi-scale [230, 231] problems, fluid-structure interactions [232], and industrial scale applications [183, 184, 233]. Since various zones and/or physics in these systems are connected through interfaces, data sharing, and consistent interface treatment among respective models are inevitable. Likewise, multi-rate and locally adaptive stepping methods can yield a mismatch at the space-time interface, and simple interpolation or extrapolation might lead to solution divergence or instabilities [205]. Moreover, even if we are interested in simulating just one portion of the domain corresponding to some specific dynamics, we still need to specify the physically consistent interface conditions. Running a high fidelity solver everywhere only to provide the flow state at the interface seems to be unreasonable. Therefore, we consider formulating an interface modeling approach to facilitate the development of efficient and reliable multi-fidelity computing. This should serve and advance the applicability of the emerging digital twin technologies in many sectors [13]. However, just like any technology, it comes with its own needs and challenges [130, 22, 234, 235]. In practice, two modeling paradigms are in order.

- *Physics-based modeling:* This approach involves careful observation of a physical phenomenon of interest, development of its partial understanding, expression of the understanding in the form of mathematical equations, and ultimately, solution of these equations. Due to the partial understanding and numerous assumptions along the steps from observation to the solution of the equations, a large portion of the essential governing physics might be, intentionally or unintentionally, ignored. The applicability of high fidelity simulators with minimal assumptions has so far been limited to the offline design phase only. Despite this significant drawback, what makes these models attractive are sound foundations from first principles, interpretability, generalizability, and existence of robust theories for the analysis of stability and uncertainty. However, most of these models are generally computationally expensive, do not adapt to new scenarios automatically, and can be susceptible to numerical instabilities.

- *Data-driven modeling:* With the abundant supply of big data, open-source cutting edge and easy-to-use machine learning libraries, cheap computational infrastructure, and high quality, readily available training resources, data-driven modeling has become very popular. Compared to the physics-based modeling

approach, these models thrive on the assumption that data is a manifestation of both known and unknown physics and hence when trained with an ample amount of data, the data-driven models might learn the full physics on their own. This approach, involving in particular deep learning, has started achieving human-level performance in several tasks that were until recently considered impossible for computers. Notable among these are image classification [236], dimensionality reduction [237], medical treatment [238], smart agriculture [239], physical sciences [67, 240, 241] and beyond. Some of the advantages of these models are online learning capability, computational efficiency for inference, accuracy even for very challenging problems as far as the training, validation and test data are prepared properly. However, due to their data-hungry and black-box nature, poor generalizability, inherent bias and lack of robust theory for the analysis of model stability, their acceptability in high stake applications like digital twin and autonomous systems is fairly limited. In fact, the numerous vulnerabilities of deep neural networks have been exposed beyond doubt in several recent works [242–244].

In this work, we put forth a *hybrid analysis and modeling (HAM)* framework as a new paradigm in modeling and simulations by promoting the strengths and mitigating the weaknesses of physics-driven and data-driven modeling approaches. Our HAM approach combines the generalizability, interpretability, robust foundation and understanding of physics-based modeling with the accuracy, computational efficiency, and automatic pattern-identification capabilities of advanced data-driven modeling technologies. In the context of multi-fidelity computing, we advocate and explore the utilization of statistical inference to bridge low-fidelity and high-fidelity descriptions. In particular, we adopt the long short-term memory (LSTM) neural network to match the reduced order model (ROM) and full order model (FOM) solutions at their intersect. To form the building blocks of our HAM approach for coupling ROM and FOM descriptions, we introduce an array of interface modeling paradigms as depicted in Fig. 5.1 and described next.

The *Direct Prolongation Interface (DPI)* approach utilizes standard projection based ROMs, where the system's state at the interface is obtained by the reconstruction of a Galerkin projection ROM solution. However, traditional Galerkin ROM often yields an inaccurate solution in case of systems with strong nonlinearity. Therefore, we utilize machine learning to correct and augment ROM solution in a hybrid

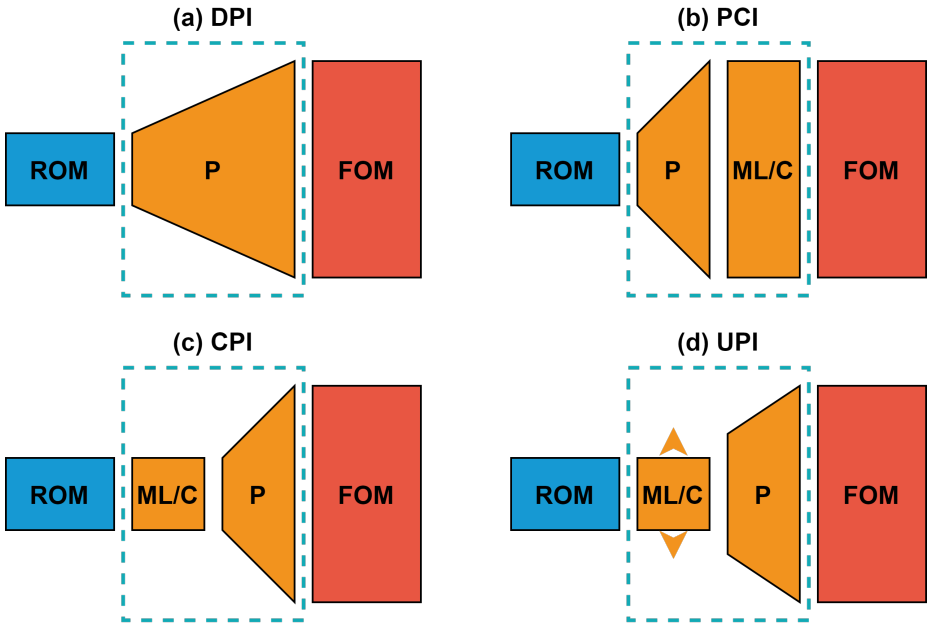


Figure 5.1: The proposed multi-fidelity concepts toward hybrid FOM-ROM coupling. Dashed blocks refer to the interface learning approaches introduced in the present work: (a) Direct Prolongation Interface (DPI), (b) Prolongation followed by a machine learning Correction Interface (PCI), (c) machine learning Correction followed by a Prolongation Interface (CPI), and (d) Uplifted Prolongation Interface (UPI) where the latent space is enhanced through machine learning before we apply the prolongation operator.

framework.

In the *Prolongation followed by machine learning Correction Interface (PCI)* methodology, an LSTM is used to rectify the field reconstruction from Galerkin ROM at the interface by learning the correction in the higher dimensional space. Although this seems to be a straightforward implementation, it might amount to learning a high dimensional correction vector, especially for two- and three-dimensional domains.

To mitigate the potential computational challenges dealing with excessively large input/output vectors, a *machine learning Correction followed by Prolongation Interface (CPI)* approach can be employed to provide a closure effect to remedy the instabilities and inaccuracies of Galerkin ROM due to modal truncation.

For CPI, the LSTM learns the correction terms in ROM space, defined by the number of modes in ROM approximation. As a result, the reconstruction quality will eventually be limited by the Galerkin ROM dimension. Therefore, the *Uplifted Prolongation Interface (UPI)* framework not only corrects the Galerkin ROM solution, but also expands the ROM subspace to enhance the reconstruction quality. Our pri-

mary motivation in this chapter is to describe and test these four interface modeling approaches to tackle FOM-ROM coupling problems and show how we can elucidate these multi-fidelity mechanisms within the HAM framework.

5.3 FOM-ROM Coupling Framework

In order to demonstrate the performance of the introduced HAM approaches for FOM-ROM coupling, we consider a coupled system as follows,

$$\frac{\partial u}{\partial t} = f_1(u;\mu_1) + g_1(u,v;\mu_1,\mu_2), \tag{5.1}$$

$$\frac{\partial v}{\partial t} = f_2(v;\mu_2) + g_2(u,v;\mu_1,\mu_2), \tag{5.2}$$

where u and v are the coupled variables and g_1 and g_2 define this coupling, while μ_1 and μ_2 denote the set of system's parameters. We highlight that the coupled variables might represent the state variables at different regions of the domain (e.g., multi-component systems), different physics (e.g., fluid-structure interactions) and/or different scales within the same domain (e.g., multi-scale systems). We suppose that the dynamics of u can be approximated by a reduced order model (ROM) while a full order model resolves v and both solvers need to communicate information to satisfy the coupling. We begin by describing the derivation of a reduced order model of u via Galerkin projection equipped with proper orthogonal decomposition (POD) for basis construction. Then, we formulate the coupling between ROM and FOM solvers.

5.3.1 Reduced order model

Introducing a spatial discretization to Eq. (5.1), it can be rewritten in a semi-discrete continuous-time as follows,

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathcal{F}(\mathbf{u},\mathbf{v};\boldsymbol{\mu}) = \mathcal{L}_1\mathbf{u} + \mathcal{L}_2\mathbf{v} + \mathcal{N}(\mathbf{u},\mathbf{v}), \tag{5.3}$$

where the boldfaced symbols represent the arrangement of discretized variables in 1D vector (e.g., $\mathbf{u} \in \mathbb{R}^{n_1}$ and $\mathbf{v} \in \mathbb{R}^{n_2}$, where n_i denotes the spatial resolution), $\boldsymbol{\mu} \in \mathbb{R}^p$ defines the system's parameters, and $\mathcal{F} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^p \to \mathbb{R}^n$ is a deterministic operator with linear and nonlinear components $\mathcal{L}$, and $\mathcal{N}$, respectively. These operators depends on the numerical scheme adopted for spatial discretization.

We exploit the advances and developments of ROM techniques to build surrogate

models to economically resolve portions of domain and/or physics. The ROM solution can thus be used to infer the flow conditions at the interface so that a FOM solver can be efficiently employed for the sub-domains of interest. The standard Galerkin ansatz is applied for the dynamics of $\mathbf{u}$ as

$$\mathbf{u}(t) \approx \Phi\boldsymbol{\alpha}(t), \tag{5.4}$$

where the columns of matrix $\Phi = [\phi_1, \phi_2, \ldots, \phi_r] \in \mathbb{R}^{n_1 \times r}$ form the orthonormal bases of a reduced subspace with an intrinsic dimension of r, and $\boldsymbol{\alpha}$ defines the projection coordinates associated with Φ. Usually, the basis functions ϕ are constructed to capture the dominant modes or underlying structures of the flow. Proper orthogonal decomposition (POD) is one popular technique to systematically construct Φ such that the solution manifold preserves as much variance as possible when projected onto the subspace spanned by Φ [83, 84, 24]. By substituting this approximation into Eq. (5.3) and performing the inner product with Φ, we get the following,

$$\frac{\mathrm{d}\boldsymbol{\alpha}}{\mathrm{d}t} = \Phi^T\mathcal{L}_1\Phi\boldsymbol{\alpha} + \Phi^T\mathcal{L}_2\mathbf{v} + \Phi^T\mathcal{N}(\Phi\boldsymbol{\alpha}, \mathbf{v}). \tag{5.5}$$

The first coefficient $(\Phi^T\mathcal{L}_1\Phi)$ can be precomputed, so the computational cost for evaluating the linear term depends on r. However, in general, the evaluation of the third term on the right hand side (nonlinear term) depends on the FOM dimension n. Fortunately, most fluid flow systems are characterized by quadratic nonlinear operator, which allows the reduction of Eq. (5.5) into

$$\frac{\mathrm{d}\boldsymbol{\alpha}}{\mathrm{d}t} = \mathfrak{L}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathfrak{N}\boldsymbol{\alpha} + \mathfrak{C}, \tag{5.6}$$

where $\mathfrak{L}$ is an $(r \times r)$ matrix and $\mathfrak{N}$ is an $(r \times r \times r)$ tensor representing the model coefficients while $\mathfrak{C}$ defines the contribution of $\mathbf{v}$ into the ROM of $\mathbf{u}$. We will see that the computation of $\mathfrak{C}$ may either be computed offline during ROM construction or as part of the online FOM solver of $\mathbf{v}$ with negligible computational overhead. Thus, the floating point operation (flop) count to evaluate the right hand side of the ROM (i.e., Eq. (5.6)) is often $O(r^3)$.

In the following, we formulate the four methodologies outlined in Fig. 5.1 to match the ROM solution for $\boldsymbol{\alpha}$ with the FOM solution at the interface. For all cases, a ROM representation is adopted for u, which can be economically solved to compute

an estimate of the interface flow condition to feed the FOM solver of v. For example, in multi-component systems like that depicted in Fig. 5.2, the ROM solution at the interface is regarded as a boundary condition for the FOM.

1. *DPI: Direct Prolongation Interface.* The objective of the DPI approach is to recover the flow variables at the interface from the ROM solution (i.e., the time integration of Eq. (5.6)). In other words, we seek to learn a mapping $\mathcal{G}_1 : \mathbb{R}^r \to \mathbb{R}^d$, that minimizes $\|u^{(i)} - \mathcal{G}_1(\boldsymbol{\alpha})\|$ where $u^{(i)}$ represent the portion of information at the interface that is shared from the ROM to the FOM solver, with d being the dimension of the interface. For multi-component systems, this interface can be a single point (e.g., for 1D systems), a line (e.g., for 2D systems), or a surface (e.g., for 3D systems). Indeed, this prolongation map naturally results from the Galerkin ansatz, and can be written as

$$\mathcal{G}_1(\boldsymbol{\alpha}) = \boldsymbol{\Theta}\boldsymbol{\alpha}, \tag{5.7}$$

   where $\boldsymbol{\Theta}$ represents the portion of the basis Φ that is computed at the interface location.

   Since the ROM approximation is built upon the assumption of representing the flow within a low order subspace, the approximation given by Eq. (5.4) basically introduces a projection error. This error can be significant for complex systems, where the flow dynamics are characterized by a wide spectrum while only few modes are considered to minimize the computational burden of solving the ROM. Moreover, the nonlinear interactions as well as the modal truncation coupled with the Galerkin projection methodology usually cause Eq. (5.6) to yield erroneous predictions of the coefficients $\boldsymbol{\alpha}(t)$. Therefore, the solution from the DPI approach is potentially inaccurate [2]. Consequently, the reconstruction $\mathcal{G}_1(\boldsymbol{\alpha})$ is no longer optimal and a correction needs to be introduced.

2. *PCI: Prolongation followed by Correction Interface.* The PCI framework aims to correct the mapping $\mathcal{G}_1$ to yield more accurate interface condition. To do so, we utilize a long short-term memory (LSTM) neural network to learn a mapping $\mathcal{G}_2 : \mathbb{R}^i \to \mathbb{R}^i$ such that

$$\mathcal{G}_2\left(\mathcal{G}_1(\boldsymbol{\alpha})\right) = u^{(i)} - \mathcal{G}_1(\boldsymbol{\alpha}). \tag{5.8}$$

In other words, LSTM is fed with a predictor of $u^{(i)}$ obtained by DPI and approximates the deviation of this value from the true state variables at the interface. Hence, this deviation estimate can be added as a correction term in a predictor-corrector fashion. In PCI, both inputs and outputs of the LSTM lie in the FOM space and thus the LSTM map can be considered as nudging scheme from the ROM prolongation $\mathcal{G}_1$ to the FOM solution [245].

We highlight that the PCI approach can be feasible for one-dimensional (1D) problems (where the interface can be just a single point). However, for higher dimensional systems, the sizes of input and output vectors grow rapidly (unless a too coarse mesh is adopted). For such cases, PCI becomes prohibitive, and the learning and correction should be performed in a reduced latent space instead.

3. *CPI: Correction followed by Prolongation Interface.* The CPI methodology works by introducing the correction in the latent subspace, rather than the FOM space. This is especially crucial for 2D and 3D configurations. In particular, the CPI aims at curing the deviation in modal coefficients predicted from solving the Galerkin ROM equations, known as closure error. Due to the modal cut-off in ROM approximation, Eq. (5.6) does not necessarily capture the true projected trajectory of $\boldsymbol{\alpha}(t)$. Therefore, we introduce an LSTM mapping $\mathcal{G}_3 : \mathbb{R}^r \to \mathbb{R}^r$ to provide a closure effect to adjust the Galerkin ROM trajectory. Specifically, the LSTM for CPI takes the values of modal coefficients acquired from integrating Eq. (5.6) in time and predicts the discrepancy between these values and their optimal values. Those are defined by the true projection (TP) of the FOM solution onto the basis functions as follows,

$$\boldsymbol{\alpha}^{\mathrm{TP}} = \Phi^T \mathbf{u}. \tag{5.9}$$

Therefore, the CPI contribution can be written as

$$\mathcal{G}_3(\boldsymbol{\alpha}) = \Phi^T \mathbf{u} - \boldsymbol{\alpha}. \tag{5.10}$$

We highlight here that the size of the input and output vectors is $O(r)$, independent of the FOM resolution, which offers a potential flexibility dealing with 2D and 3D problems. Once the modal coefficients are corrected, they are prolonged from the ROM space to the FOM space using the reconstruction map $\mathcal{G}_1$. For all results, we also show the results obtained from the true projection of FOM

solution onto the ROM subspace at the interface as,

$$\mathbf{u}^{\mathrm{TP}} = \mathcal{G}_1(\boldsymbol{\alpha}^{\mathrm{TP}}). \tag{5.11}$$

We highlight that u^{TP} represents the best approximation of the true flow field that can be achieved using a linear subspace with an intrinsic dimension of r.

4. *UPI: Uplifted Prolongation Interface.* Although the CPI methodology cures the closure error and provides a stabilized solution, it does not address projection error. Unless a large number of modes are resolved, the projection error can be significant, especially for problems with discontinuities and shocks. To deal with those situations, an uplifting ROM has been proposed [2], where both closure and projection errors are taken care of. For closure, similar to CPI, the Galerkin ROM predictions are tuned to match their true projection values. In addition, following Galerkin ROM solution, the ROM subspace is expanded to capture some of the smaller scales missing in the initial subspace as follows,

$$\mathbf{u} \approx \Phi\boldsymbol{\alpha} + \Psi\boldsymbol{\beta}, \tag{5.12}$$

where the columns of $\Psi = [\psi_1, \psi_2, \ldots, \psi_q]$ form orthonormal basis functions for a q-dimensional subspace complementing that spanned by Φ and $\boldsymbol{\beta}$ defines the corresponding projection coordinates. Similar to Φ, Ψ can be computed through the POD algorithm. Note that Φ and Ψ are orthogonal to each others (i.e., $\Phi^T\Psi = \Psi^T\Phi = \mathbf{0}$). Indeed, Ψ represents the next q basis functions generated by POD after the first r being dedicated to Φ. Those are also constructed a priori during an offline stage using the collected set of snapshot data. We highlight that the Galerkin ROM equations only solve for $\boldsymbol{\alpha}$ to keep the computational cost as low as possible.

Therefore, a complementary model for $\boldsymbol{\beta}$ has to be constructed so that the uplifting approach can be employed. To accomplish this, a mapping from the first r modal coefficients to the next q modes is assumed to exist. Nonlinear Galerkin projection has been pursued to express this mapping as $\boldsymbol{\beta} = \mathcal{H}(\boldsymbol{\alpha})$, but it has been found challenging for most systems [81]. Instead, we exploit the LSTM learning capabilities to infer this map from data. This uplifting approach enhances the quality of prolonged solution by providing a super-resolution effect. In particular, the UPI architecture is trained to read the Galerkin ROM predic-

tion for the first r modal coefficients as input, and return the true coefficients of the first $r+q$ modes. Thus, it provides a closure effect for the first r modes and a super-resolution effect for the next q modes, simultaneously in a single network as follows,

$$\mathcal{G}_4 : \mathbb{R}^r \rightarrow \mathbb{R}^{r+q} \tag{5.13}$$

$$\mathcal{G}_4(\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{\alpha}^{\mathrm{TP}} \\ \boldsymbol{\beta}^{\mathrm{TP}} \end{bmatrix}. \tag{5.14}$$

We note here that the first $r+q$ spatial modes have to be built and stored beforehand, which introduces slightly more storage costs. For the present study, we explore the specific case where $q = r$, but a generalization is straightforward.

The FOM-ROM coupling philosophy as well as the introduced interface learning approaches are summarized in the cartoon shown in Fig. 5.2. These methodologies are also applicable to a wide range of computational problems with multi-fidelity domain decomposition. The depicted system is assumed to be fully characterized by three mutually orthogonal sets of basis functions, namely Φ, Ψ, and ζ as below

$$\mathbf{u} = \Phi\boldsymbol{\alpha} + \Psi\boldsymbol{\beta} + \zeta\boldsymbol{\gamma}, \tag{5.15}$$

where $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ are the corresponding projection coordinates. We also suppose that Galerkin ROM resolves the Φ set of modes (i.e., truncating the contributions of Ψ, and ζ). We reiterate here that the Galerkin ROM yields inaccurate solution (sketched by the noisy (rough) curve of predicted $\boldsymbol{\alpha}$). Consequently, the quality of the direct prolongation mapping is compromised. The PCI aims at correcting the reconstructed solution at the interface. Even though the PCI technique acts only on a small portion of the domain (i.e., the interface), it might amount to excessively large input and output sizes.

On the other hand, CPI treats the Galerkin ROM deficiencies at the ROM level. In particular, it introduces a closure effect to better predict $\boldsymbol{\alpha}$. This closure simply compensates the effects of truncated modes (i.e., Ψ and ζ) onto the dynamics of Φ. This yields a better estimate of $\boldsymbol{\alpha}$, as illustrated by the smooth curve in Fig. 5.2. We note that in CPI, the effects of Ψ and ζ are only considered to improve the prediction of $\boldsymbol{\alpha}$. However, their contributions to the solution manifold reconstruction are not included, resulting in a substantial reconstruction error (projection error).

To deal with this caveat, UROM seeks to add a super-resolution enhancement by incorporating the Ψ set of basis into the reconstruction step by learning the dynamics of the corresponding $\boldsymbol{\beta}$ coordinates. This is represented by a higher resolution (denser) reconstruction in UPI case, compared to CPI, and DPI. Note that the PCI is still showing the highest resolution (the densest reconstruction) as it nudges the prediction at the interface to its FOM counterpart (i.e., including all Φ, Ψ, and ζ).

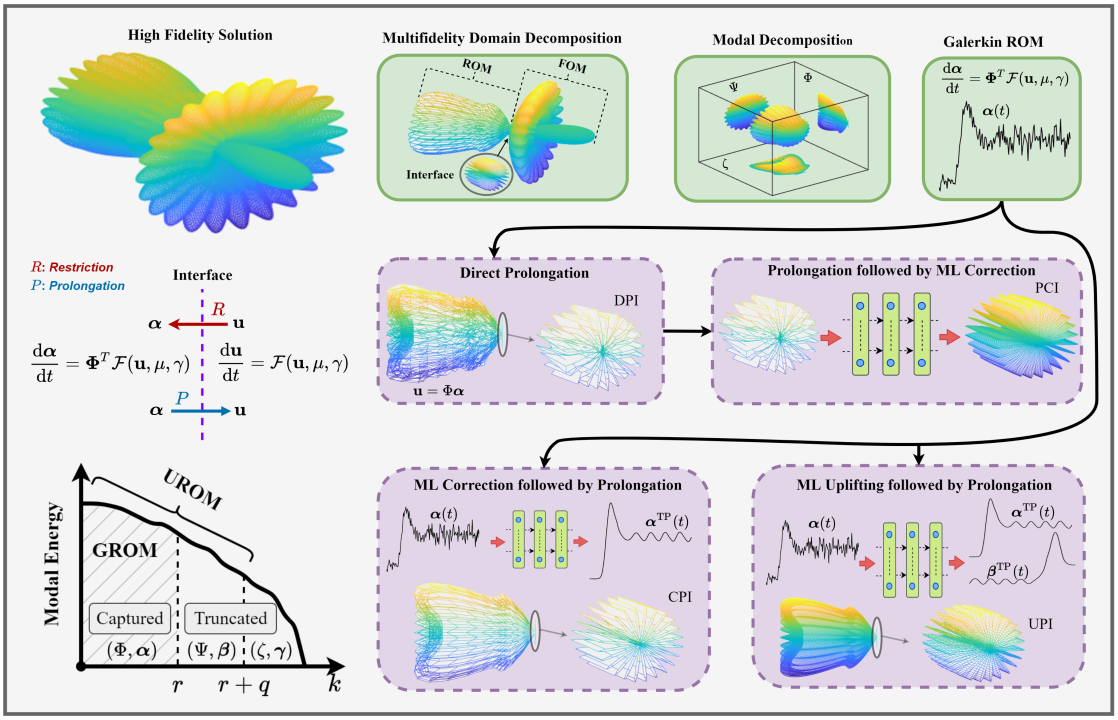


Figure 5.2: Schematic illustration of the methodologies introduced to utilize ROM to economically provide sound interface conditions in a multi-fidelity domain decomposition problems. Galerkin ROM yields inaccurate predictions (represented by rough curve), and direct prolongation of these results might be not efficient. PCI adds a correction effect to the prolonged solution in FOM space. Instead, CPI and UPI introduce the correction at ROM level before prolongation. UPI adds an extra super-resolution effect to augment solution quality.

5.4 Demonstrations

We demonstrate the FOM-ROM coupling methodologies using two examples of varying complexities. In the first one, we describe a fluid flow scenario over a bi-zonal domain with heterogeneous physical properties using the one-dimensional Burgers problem. For this case, we shall see that the interface between different sub-domains is defined by a single point (i.e., $d = 1$). Second, we consider the Marsigli flow problem represented by the two-dimensional Boussinesq equations to demonstrate the FOM-

ROM coupling for multi-physics systems. In particular, a ROM solver is devoted for the mass and momentum transport equations while a FOM is reserved for the energy transport.

5.4.1 The one-dimensional Burgers problem

In order to represent a zonal multi-fidelity simulation, we consider the following one dimensional (1D) viscous Burgers problem,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}\left(\nu\frac{\partial u}{\partial x}\right) - \gamma u, \tag{5.16}$$

$$(\nu, \gamma) = \begin{cases} (10^{-2}, 0), & \text{for } 0 \leq x \leq x_b, \\ (10^{-4}, 1), & \text{for } x_b < x \leq 1, \end{cases} \tag{5.17}$$

where x_b is the spatial location of the interface defining the physical heterogeneity. We highlight that Eq. (5.16) includes the $\frac{\partial}{\partial x}\left(\nu\frac{\partial u}{\partial x}\right)$ term instead of the commonly used $\nu\frac{\partial^2 u}{\partial x^2}$ term to permit the spatial variation of ν. For the given setup, the stiffness and physical properties in the left part dictates higher spatial as well as temporal resolutions than those required for the right partition. If we opt to a global unified (unique fidelity) solver over the whole domain, the quality of the solution will be dominated by stiffness of the left zone. In other words, a smaller time step would be required to satisfy numerical stability when using an explicit temporal integration scheme. Specifically, assuming that we utilize the forward in time and central in space finite difference scheme (FTCS) to solve Eq. (5.16) with a spatial grid resolution of 4096, a time step of approximately 2.5×10^{-6} will be required for the left part of domain, while a time step of 2.5×10^{-4} would be sufficient if we were able to only resolve the right part. Therefore, domain decomposition approaches might be adopted to segregate partitions with varying numerical requirements. Despite the effectiveness and efficiency of these approaches, idle delays can arise in order to accommodate information transfer from the left zone to the right zone through their common interface. Instead, low-fidelity proxy models can be utilized to avoid such lags by approximating the *effective* dynamics of stiff regions and providing sound interface boundary conditions to the rest of the computational domain.

The discretized domain is divided into a left zone with $\mathbf{u}^L \in \mathbb{R}^{n_1}$ for $x \in [0, x_b]$ and a right zone with $\mathbf{u}^R \in \mathbb{R}^{n_2}$ for $x \in [x_b, 1]$, where $n_1 + n_2 = n + 1$. We build a

reduced order model for the left sub-domain as follows,

$$\mathbf{u}^L(t) = \Phi\boldsymbol{\alpha}(t), \tag{5.18}$$

$$\frac{\mathrm{d}\boldsymbol{\alpha}}{\mathrm{d}t} = \mathfrak{L}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathfrak{N}\boldsymbol{\alpha}, \tag{5.19}$$

where $\mathfrak{L}$ and $\mathfrak{N}$ represent the tensorial ROM coefficients which can be precomputed during the offline stage as,

$$\mathfrak{L}_{i,k} = \langle \frac{\partial}{\partial x}\left(\nu\frac{\partial u}{\partial x}\right) - \gamma\phi_i; \phi_k\rangle, \qquad \mathfrak{N}_{i,j,k} = \langle -\phi_i\frac{\partial\phi_j}{\partial x}; \phi_k\rangle, \tag{5.20}$$

where the angle parentheses denote the inner product (i.e., $\langle \mathbf{a}; \mathbf{b}\rangle = \mathbf{a}^T\mathbf{b}$).

For data generation, we solve the full order model representing the 1D Burgers equation (i.e., Eq. (5.16)) over the entire domain using a spatial mesh resolution of 4096 and time step of 2.5×10^{-6}. For initial condition, we consider a square-like wave defined as,

$$u(x, 0) = 0.5 - 0.5\tanh\frac{x - x_b}{\epsilon}, \tag{5.21}$$

where a value of $x_b = 0.75$ is considered as the location of the interface and $\epsilon = 0.005$ is used to define the sharpness of the shock at x_b. Dirichlet boundary conditions are assumed at both boundaries (i.e., $u(0, t) = u(1, t) = 0$). We compute the evolution of the velocity field for $t \in [0, 2]$, and collect snapshots every 100 time steps.

For ROM construction, we consider the truncated solution snapshots for the left part of the domain (i.e., $0 \leq x \leq 0.75$) for $t \in [0, 1]$. For POD basis generation, we use only 200 snapshots distributed evenly from $t = 0$ to $t = 1$, to reduce the computational cost of solving the corresponding eigenvalue problem. Once ROM is constructed, it is integrated in time with a time step of 2.5×10^{-4} to match the time step in the right part of the domain (to be handled via a FOM solver). During the deployment phase of the coupled system, the ROM feeds the FOM solver with the boundary condition at x_b. On the other hand, the effects of the interface on the ROM dynamics are considered during the offline stage of data generation and basis construction.

We utilize labelled data at the time interval of $t \in [0, 1]$ for the training and validation of the LSTM neural networks. In particular, four fifths of the collected data set are randomly selected for training while the remaining one fifth is reserved for validation purposes (to avoid overfitting). We highlight that numerical experiments

reveal that the performance in our case is not significantly sensitive to the neural network hyperparameters. Thus, we manually tune the hyperparameters during the training and validation stages by repeating the previous procedure multiple times while varying the seeds for the random number generator and using averaged loss values for assessment. We adopt an LSTM architecture of 2 layers with 20 cells in each layer. In the meantime, we note that automated hyperparameter search tools can be utilized for optimized architectures. Testing of the proposed schemes is performed for $t \in [0, 2]$, corresponding to a temporal extrapolation behavior with respect to the training interval.

Since the coupling between ROM and FOM is represented by the physical interface at x_b, we notice that the $\boldsymbol{\Theta}$ in the DPI map (i.e., $\mathcal{G}_1(\boldsymbol{\alpha}) = \boldsymbol{\alpha}$) is simply the last row of the matrix $\boldsymbol{\Phi}$. In Fig. 5.3, we plot the velocity at the interface obtained from adopting ROM for the left part of the domain, and corrected with machine learning architectures as described before for $r = 2$ and $r = 4$. We note that FOM solution corresponds to the velocity at the interface obtained by applying a FOM solver all over the domain using a time step of 2.5×10^{-6}, while the true projection (TP) represents the projection of the truncated velocity field in the left zone onto the POD subspace. The shaded area in Fig. 5.3 stands for the time interval utilized for POD basis generation, ROM formulation, and LSTMs training.

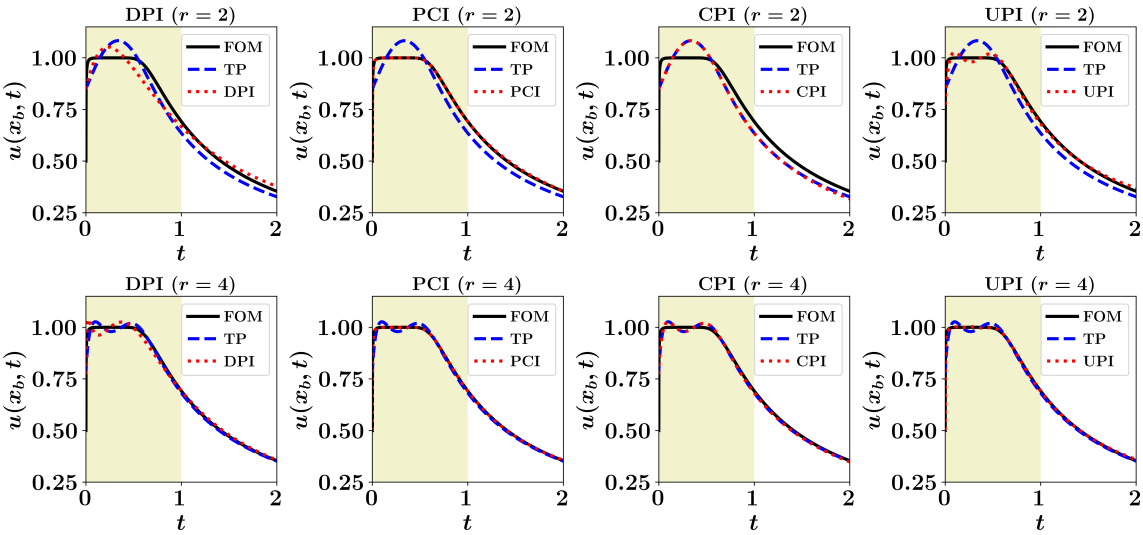


Figure 5.3: Velocity at the interface obtained by considering ROM in the left part of the domain, with $r = 2$ (top) and $r = 4$ (bottom). FOM solution corresponds to solving the governing equation over the entire domain, while TP denotes the projection of the FOM solution in the left zone onto the corresponding POD subspace.

It can be seen that DPI results, especially for $r = 2$, are not very accurate due to the mutual effects of modal truncation and model nonlinearity in Galerkin ROM. On the other hand, the PCI solution gives almost perfect match with FOM. As the PCI approach nudges the prolonged ROM solution to its FOM counterpart, it gives even higher accuracy than TP. That is TP is limited by the maximum quality that can be obtained using a rank-r approximation. For CPI, since the LSTM introduces a closure effect, it steers the ROM results to match the TP solution. Finally, the UPI recovers some of the smaller scales (truncated modes) so it yields better reconstruction than TP since it spans a larger subspace. For $r = 2$, UPI uplifts the solution to a rank-4 approximation, while for $r = 4$, it is uplifted to a rank-8 approximation.

For quantitative assessment, we document the ℓ_2 norm of the deviation of the predicted velocity at the interface compared to the FOM solution in Table 5.1. Results are reported for $r \in \{2, 4, 8\}$. We see that the error in the CPI case almost matches that of TP, while PCI gives the highest accuracy since it is trained to learn the correction with respect to the FOM solution. Also, CPI results are significantly close to TP, illustrating the closure effect introduced by CPI to account for the effect of modal truncation on ROM dynamics. Another interesting observation is that UPI quality at a given value of r is equivalent to TP with twice that value. This indicates that UPI is able to give a super-resolution effect up to $2r$ (since we select $q = r$). Also, we notice that at $r = 2$, DPI yields lower ℓ_2 norm than TP. This is because TP solution is obtained by the projection of the FOM solution onto the POD subspace generated using data at $t \in [0, 1]$. So, this subspace is optimal only for $t \in [0, 1]$, while testing is performed up to $t = 2$. Therefore, TP solution no longer represents the best rank-r approximation beyond $t = 1$.

Table 5.1: ℓ_2 norm for the deviation of the velocity at the interface with respect to its FOM value for $t \in [0, 2]$.

Setup	$r = 2$	$r = 4$	$r = 8$
TP	4.56	1.25	0.21
DPI	3.67	1.46	0.44
PCI	0.23	0.07	0.03
CPI	4.57	1.27	0.23
UPI	1.28	0.24	0.11

Finally, we investigate the coupling efficiency by solving the right part (i.e., $0.75 \leq x \leq 1$) using a high fidelity FOM solver applied only onto this subdomain. This is fed with a boundary condition $u(0.75, t)$ from the low-fidelity interface learning

approaches described before. Figure 5.4 shows the spatiotemporal velocity profile with $r = 2$, compared to the FOM predictions. Again, we observe that the solution with PCI boundary is similar to this FOM solution. Also, CPI matches the TP results, but they both smooth-out the surface peak because of the low rank limitations. Although it seems that the accuracy of UPI cannot exceed that of CPI with $r + q$ assuming optimal performance for both CPI and UPI, there is a computational side in this comparison. The CPI with $r + q$ modes implies the solution of a Galerkin ROM with a dimension of $r + q$, while the UPI requires the solution of a Galerkin ROM with r. We have seen that for fluid flows with quadratic nonlinearity, the computational cost of solving a Galerkin ROM scales cubically with the number of modes. Thus, the implementation of CPI with $r + q$ with $q = r$ is about 8 times more costly than UPI with $r + q$. At this point, we highlight that the selection of r and q is highly dependent on the problem in hand, the corresponding decay of POD eigenvalues, and the level of accuracy sought. A compromise between computational cost of solving a Galerkin ROM with r and the corresponding stability, the amount of information captured by r and $r + q$ modes, and reliability of UPI with $r + q$ is always in place.

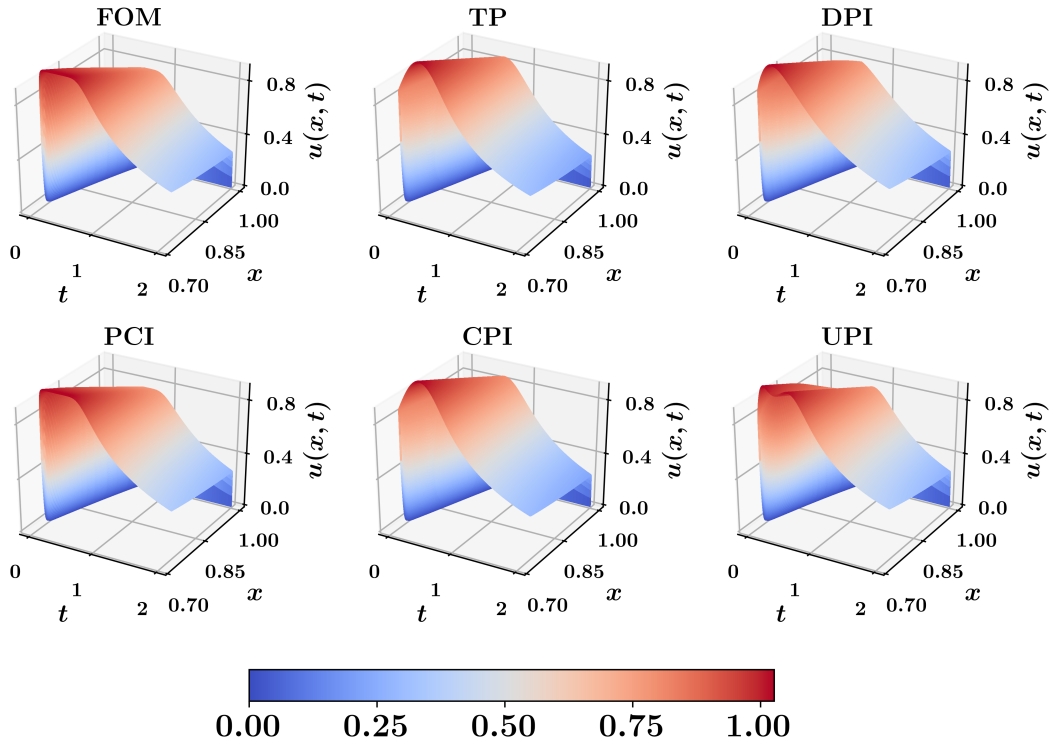


Figure 5.4: Spatio-temporal velocity profile obtained from applying high fidelity (FOM) solver onto the right subdomain ($0.75 \leq x \leq 1$), fed with interface boundary from a low-fidelity (ROM) solution with $r = 2$.

5.4.2 The two-dimensional Boussinesq problem

Boussinesq equations represent a simple approach for modeling geophysical waves such as oceanic and atmospheric circulations induced by temperature differences [246] as well as other situations, like isothermal flows with density stratification. In the Boussinesq approximation, variations of all fluid properties other than the density are neglected completely. Moreover, the density dependence is ignored in all terms except for gravitational force (giving rise to buoyancy effects). As a result, the continuity equation can be adopted in its constant density form, and the momentum equation can be simplified significantly. The dimensionless form of the 2D incompressible Boussinesq equations in vorticity-streamfunction formulation can be represented by the following two coupled transport equations [247, 248],

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \nabla^2 \omega + \mathrm{Ri} \frac{\partial \theta}{\partial x}, \tag{5.22}$$

$$\frac{\partial \theta}{\partial t} + J(\theta, \psi) = \frac{1}{\mathrm{RePr}} \nabla^2 \theta, \tag{5.23}$$

where ω, ψ and θ denote the vorticity, streamfunction and temperature fields, respectively. In Boussinesq flow systems, there are three leading physical mechanisms, namely viscosity, conductivity, and buoyancy, giving rise to three dimensionless numbers; Reynolds number Re, Richardson number Ri, and Prandtl number Pr.

We utilize the 2D Boussinesq equation to illustrate the FOM-ROM coupling for multi-physics situations. In particular, we suppose that we are more interested in the temperature field predictions. Thus, we dedicate a FOM solver for Eq. (5.23). However, we see that the solution of this equation requires evaluating the streamfunction field at each time step. The kinematic relationship between vorticity and streamfunction is given by the following Poisson equation,

$$\nabla^2 \psi = -\omega, \tag{5.24}$$

which implies that the streamfunction is not a prognostic variable, and can be computed from the vorticity field at each timestep. In typical simulations, the solution of Eq. (5.24) consumes significant amount of time and computational resources and is considered the bottleneck for most incompressible flow solvers. Therefore, we can

consider a ROM solver for the voriticity dynamics as follows,

$$\omega(x,y,t) = \bar{\omega}(x,y) + \sum_{k=1}^{r} \alpha_k(t)\phi_k^{\omega}(x,y), \tag{5.25}$$

where $\bar{\omega}$ denotes the time-averaged vorticity field and the POD is performed on the fluctuating part of ω. We note that Eq. (5.24) allows us to assume the same modal coefficients $\alpha_k(t)$ for both ω and ψ as follows,

$$\psi(x,y,t) = \bar{\psi}(x,y) + \sum_{k=1}^{r} \alpha_k(t)\phi_k^{\psi}(x,y), \tag{5.26}$$

where the time-averaged streamfunction field and the corresponding basis functions can be computed from the following relations,

$$\nabla^2\bar{\psi}(x,y) = -\bar{\omega}(x,y), \tag{5.27}$$

$$\nabla^2\phi_k^{\psi}(x,y) = -\phi_k^{\omega}(x,y). \tag{5.28}$$

Thus, the Galerkin ROM of Eq. (5.22) can be written as

$$\frac{\mathrm{d}\alpha_k}{\mathrm{d}t} = \mathfrak{B}_k + \sum_{i=1}^{R} \mathfrak{L}_{i,k}^{(\omega,\psi)}\alpha_i + \sum_{i=1}^{R} \mathfrak{L}_{i,k}^{(\omega,\theta)}\beta_i + \sum_{i=1}^{R}\sum_{j=1}^{R} \mathfrak{N}_{i,j,k}^{(\omega,\psi)}\alpha_i\alpha_j, \tag{5.29}$$

where β represent the projection of the temperature fields onto the reduced subspace defined as

$$\beta_k(t) = \big\langle \theta(x,y,t_n) - \bar{\theta}(x,y); \phi_k^{\theta}(x,y)\big\rangle, \tag{5.30}$$

where the $\bar{\theta}$ denotes the time-averaged field of θ and $\mathbf{\Phi}^{\theta}$ are the corresponding orthonormal POD modes. The predetermined coefficients in Eq. (5.29) are calculated as follows,

$$\mathfrak{B}_k = \big\langle -J(\bar{\omega},\bar{\psi}) + \frac{1}{\mathrm{Re}}\nabla^2\bar{\omega} + \mathrm{Ri}\frac{\partial\bar{\theta}}{\partial x}; \phi_k^{\omega}\big\rangle, \tag{5.31}$$

$$\mathfrak{L}_{i,k}^{(\omega,\psi)} = \big\langle \frac{1}{\mathrm{Re}}\nabla^2\phi_i^{\omega} - J(\phi_i^{\omega},\bar{\psi}) - J(\bar{\omega},\phi_i^{\psi}); \phi_k^{\omega}\big\rangle, \tag{5.32}$$

$$\mathfrak{L}_{i,k}^{(\omega,\theta)} = \big\langle \mathrm{Ri}\frac{\phi_i^{\theta}}{\partial x}; \phi_k^{\omega}\big\rangle, \tag{5.33}$$

$$\mathfrak{N}_{i,j,k}^{(\omega,\psi)} = \big\langle -J(\phi_i^{\omega},\phi_j^{\psi}); \phi_k^{\omega}\big\rangle. \tag{5.34}$$

We notice that the ROM defined by Eq. (5.29) equipped with Eq. (5.26) can be adopted to approximate the instantaneous streamfunction field, which is required to solve Eq. (5.23) for the temperature in FOM space. On the other hand, the solution of the FOM (i.e., Eq. (5.23)) along with Eq. (5.30) feeds the ROM solver with β values. This constitutes a *two-way* FOM-ROM coupling problem, in contrast to the *one-way* coupling in the aforementioned 1D Burgers example. We also highlight that the computational cost of the projection step (i.e., Eq. (5.30)) is minimal compared to the solution of Eq. (5.23).

For demonstration, we consider a strong-shear flow exhibiting the Kelvin-Helmholtz instability, known as Marsigli flow or lock-exchange problem. The physical process in this flow problem explains how differences in temperature/density can cause currents to form in the ocean, seas and natural straits. For example, Marsigli discovered that the Bosporus currents are a consequence of the different water densities in the Black and Mediterranean seas[249]. Basically, when fluids of two different densities meet, the higher density fluid slides below the lower density one. This is one of the primary mechanisms by which ocean currents are formed [250].

The problem is defined by two fluids of different temperatures, in a rectangular domain $(x, y) \in [0, 8] \times [0, 1]$ with a a vertical barrier dividing the domain at $x = 4$, keeping the temperature, θ, of the left half at 1.5 and temperature of the right half at 1. Initially, the flow is at rest (i.e., $\omega(x, y, 0) = \psi(x, y, 0) = 0$), with uniform temperatures at the right and left regions (i.e., $\theta(x, y, 0) = 1.5 \ \forall \ x \in [0, 4]$ and $\theta(x, y, 0) = 1 \ \forall \ x \in (4, 8]$). Free slip boundary conditions are assumed for flow field, and adiabatic boundary conditions are prescribed for temperature field. Reynolds number of $\mathrm{Re} = 10^4$, Richardson number of $\mathrm{Ri} = 4$, and Prandtl number of $\mathrm{Pr} = 1$ are set in Eqs. (5.22) and (5.23)). A Cartesian grid of 4096×512, and a timestep of $\Delta t = 5 \times 10^{-4}$ are used for the FOM simulations. Standard second-order central finite difference schemes are adopted for the discretization of linear terms while the second order Arakawa scheme [251] is used to compute the Jacobian term. The evolution of the temperature field is shown in Fig. 5.5 at $t = 0, 2, 4, 8$. At time zero, the barrier is removed instantaneously triggering the lock-exchange problem, where the higher density fluid (on the right) slides below the lower density fluid (on the left) causing an undercurrent flow moving from right to left. Conversely, an upper current flow moves from left to right, causing a strong shear layer between the counter-current flows. As a result, vortex sheets are produced, exhibiting the Kelvin-Helmholtz instability.

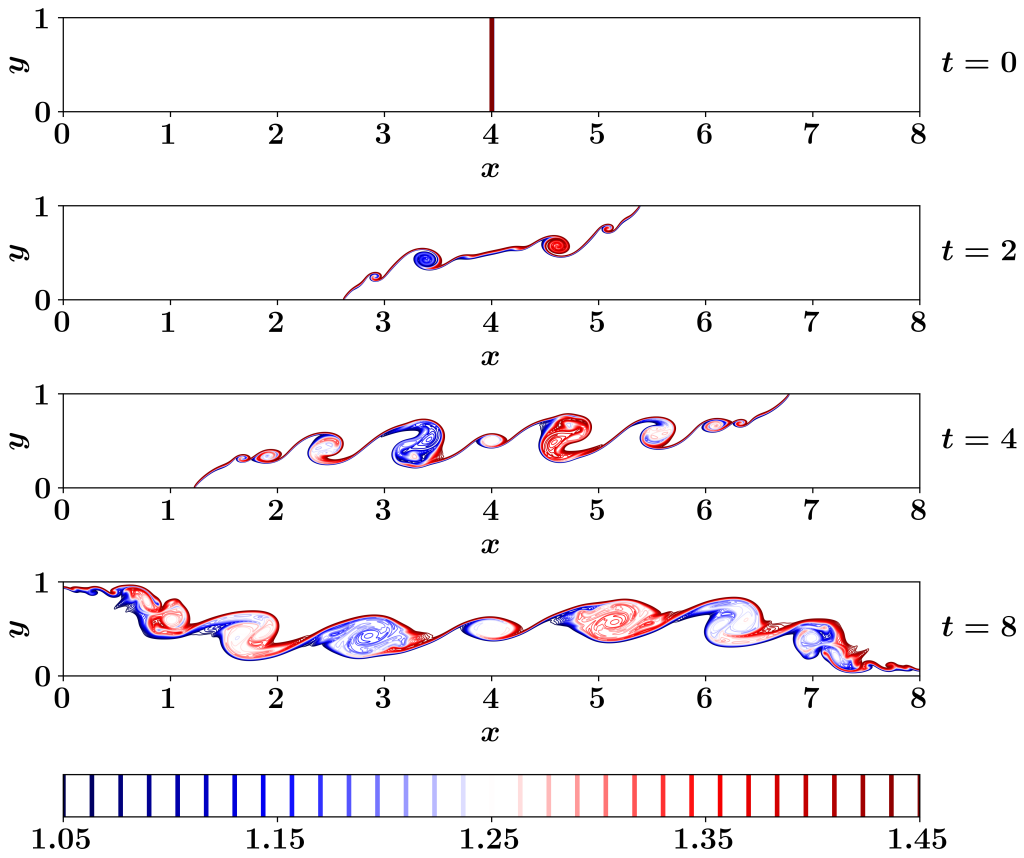


Figure 5.5: Temperature field at different time instances for the 2D Boussinesq problem using 4096×512 grid and $\Delta t = 0.0005$.

Considering the dimensionality of this problem, we emphasize that the PCI approach becomes highly unfeasible. With the current mesh resolution (i.e., 4096×512), the dimension of the state space of the prolonged interface is $\approx 2 \times 10^6$. Building and training of LSTMs with an input and output vector sizes of two millions become prohibitive. Even the training of convolutional neural network with such high resolutions (which are typical in fluid flow simulations) requires excessive computational resources. Therefore, we present results for FOM-ROM coupling using approaches that operate in ROM space (i.e., DPI, CPI and UPI). We note that 800 time snapshots are stored for POD basis construction. For the Galerkin ROM solver, $r = 8$ modes are retained. We also utilize the dataset of the stored 800 snapshots for LSTM training and validation (80% randomly selected for training and the rest for validation, similar to the previous example). A two-layer LSTM with 20 cell in each layer constitutes our LSTM architecture. During the testing phase, the trained neural networks are deployed at every timestep.

Figure 5.6 shows the predictions of the temperature field at final time (i.e., $t = 8$) computed from DPI, CPI, and UPI approaches compared to the FOM field. We emphasize that the FOM-ROM coupling results correspond to the solution of the vorticity equation with a ROM solver, which feeds the FOM solver with streamfunction to solve the temperature equation only as opposed to the FOM results which comes from the solution of both the 2D Boussinesq equations using a FOM simulation. Although the CPI results are better than those of DPI, we can observe that the fine details of the flow field are not accurately captured. That is 8 modes are not sufficient to sufficiently represent the flow field. This is a common problem for convection-dominated flows which exhibit slow decay in the POD eigenvalues and the generated global basis functions suffer from modal deformation [104]. On the other hand, the implementation of the UPI approach with $r = 8$ and $q = 8$ recovers an increased amount of the fine flow structures that are not well-represented by the first $r = 8$ modes.

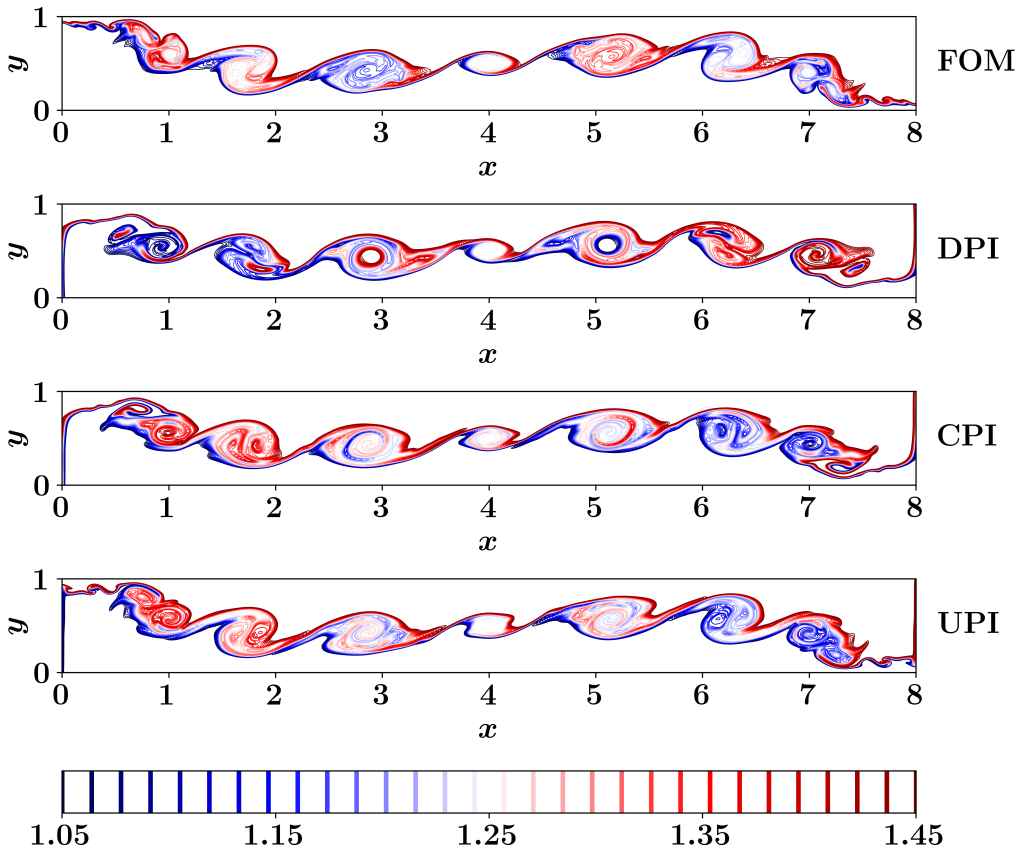


Figure 5.6: Final temperature fields as obtained from different FOM-ROM coupling approaches, compared to the FOM solution. We note that the PCI becomes infeasible for higher dimensional systems.

Although the ROM is built for the vorticity equation only, the basis functions of the temperature fields should be generated as well to carry-out the coupling from FOM to ROM. Moreover, in order to illustrate the temporal variation of the coupling quality, we project the resolved temperature fields at different times onto their POD basis. This is depicted in Fig. 5.7, showing the effect of different approaches on the resulting predictions of temperature fields. The FOM trajectory corresponds to the solution of both the 2D Boussinesq equations in FOM space, then projecting the obtained fields on the basis functions of θ (see Eq. (5.30)). For the rest, the streamfunction fields are obtained from ROM predictions and fed into FOM solver to compute the temperature fields.

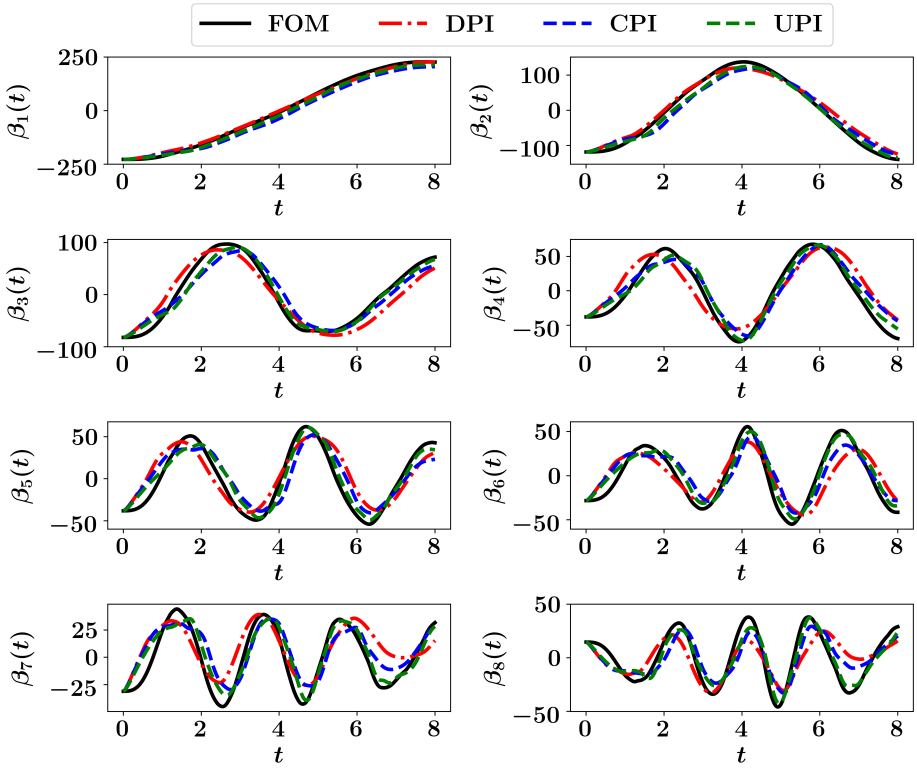


Figure 5.7: Projection of the predicted temperature fields at different times from FOM, DPI, CPI and UPI onto the POD basis function.

5.5 Conclusions

We provide an interface learning approach via FOM-ROM coupling for multi-fidelity simulation environments. This learning paradigm is built with a hybrid analysis and modeling (HAM) framework to enhance the ROM approximation of interface conditions. A demonstration with a bi-zonal 1D Burgers problem is considered to assess

the performance of the introduced learning schemes for multi-component systems. For 1D problems, we find that a prolongation followed by a machine learning correction interface (PCI) yields very good predictions. However, this might be unfeasible for 2D and 3D cases, where a correction in ROM subspace is essential. For such, a machine learning correction in ROM space followed by a prolongation interface (CPI) can produce sufficient accuracy. For complex systems where the projection error is significantly large, an uplifted prolongation interface (UPI) methodology can be adopted to recover some of the truncated scales. This is further illustrated using the lock-exchange problem governed by the 2D Boussinesq problem, where the ROM and FOM solvers address the vorticity and temperature equations, respectively. The coupling from ROM to FOM is represented by the 2D streamfunction fields reconstructed from the ROM solver, saving the run time for Possion equation, which is the most demanding part of an incompressible flow solver. Owing to the relative simplicity, robustness and ease of these interface learning methods, we expect a growing number of applications in a large variety of interfacial problems in science and engineering. Of particular interest, this FOM-ROM coupling could be a viable method for developing next generation digital twin technologies.

Part III

Data Assimilation and Machine Learning

CHAPTER 6

Forward Sensitivity Analysis for Adaptive Closures in Nonlinear Reduced Order Modeling

6.1 Abstract

In this chapter, we propose a variational approach to estimate eddy viscosity using forward sensitivity method (FSM) for closure modeling in nonlinear reduced order models. FSM is a data assimilation technique that blends model's predictions with noisy observations to correct initial state and/or model parameters. We apply this approach on a projection based reduced order model (ROM) of the one-dimensional viscous Burgers equation with a square wave defining a moving shock, and the two-dimensional vorticity transport equation formulating a decay of Kraichnan turbulence. We investigate the capability of the approach to approximate an optimal value for eddy viscosity with different measurement configurations. Specifically, we show that our approach can sufficiently assimilate information either through full field or sparse noisy measurements to estimate eddy viscosity closure to cure standard Galerkin reduced order model (GROM) predictions. Therefore, our approach provides a modular framework to correct forecasting error from a sparse observational network on a latent space. We highlight that the proposed GROM-FSM framework is promising for emerging digital twin applications, where real-time sensor measurements can be used to update and optimize surrogate model's parameters.

6.2 Introduction

Data assimilation (DA) is a family of algorithms and techniques that aim at blending mathematical models with (noisy) observations to provide better predictions by correcting initial condition and/or model's parameters [252–254]. DA plays a key role in geophysical and meteorological sciences to make more reliable numerical weather

This chapter is adapted from: *Ahmed, S. E., Bhar, K., San, O., & Rasheed, A. (2020). Forward sensitivity approach for estimating eddy viscosity closures in nonlinear model reduction. Physical Review E, 102(4), 043302.*

predictions. Standard popular algorithms that are often adopted in weather prediction centers include variational methods (e.g., 3D-VAR [255, 256] and 4D-VAR [257–262] methods), sequential methods (e.g., reduced rank (ensemble) Kalman filters [263–270]), and hybrid methods [271–277]. Another method that mitigates the computational cost in solving the inherent optimization problem in variational methods is called the forward sensitivity method (FSM) developed by Lakshmivarahan and Lewis [278, 279]. FSM builds on the assumption that model error stems from incorrect specification of the control elements, which include initial conditions, boundary conditions, and physical/empirical parameters. The FSM approach corrects the control elements using information from the time evolution of sensitivity functions, defined as the derivatives of model output with respect to the elements of control.

Other than meteorology [280], DA tools are gaining popularity in different disciplines like reservoir engineering [281], and neuroscience [282]. Recent works have also drawn techniques and ideas from DA to enrich reduced order modeling of fluid flows and vice versa [51, 283–288, 14, 289]. In conventional projection-based model reduction approaches, a set of system's realizations are used to build a reduced order model (ROM) that sufficiently represent the system's dynamics with significantly lower computational cost [15–22, 24–31]. This process includes the extraction of a handful of basis functions representing the underlying flow patterns or coherent structures that dominate the majority of the bulk mass, momentum and energy transfers. In the fluid dynamics research community, proper orthogonal decomposition (POD) is, generally speaking, the most popular and effective technique that produces hierarchically ordered solution-adapted basis functions (or modes) that provide the optimal basis to represent a given collection of field data or snapshots [83, 84, 24, 85, 86]. To emulate the system's dynamics, a surrogate model is often built by performing a Galerkin projection of the full order model (FOM) operators onto a reduced subspace spanned by the formerly constructed POD modes [33–40].

However, the off-design performance of ROMs is usually questionable since the reduced basis and operators are formed offline for a given set of operating conditions, while the ROM has to be solved online for different conditions. Therefore, a dynamic update of model operators and parameters is often sought to enhance the applicability of ROMs in realistic contexts. That being said, adoption of DA tools to absorb real observations to correct and update ROMs should present a viable cure for this caveat. The present chapter aims at pushing towards utilizing DA techniques to improve the performance of nonlinear ROMs. A common problem that emerges in such ROMs

is the inaccuracy of solution trajectory, especially for long time integration of quasi-stationary problems. This solution inaccuracy has been commonly attributed to the modal truncation and intrinsic interactions between truncated modes and retained modes. A correction term compensating the effects of truncated modes has been often introduced to achieve more accurate ROM results [59, 49, 53, 290–292]. Furthermore, recent studies have shown that model's performance can be improved by the choice of the projection method [293, 98, 108] and the definition of the adopted inner product [56].

In order to enhance the solution accuracy, closure and stabilization techniques have been introduced to account for the effects of discarded modes on the dynamics of the ROM. In particular, eddy viscosity closures, inspired from large eddy simulations (LES), have shown a significant success in ROM closure modeling [294, 295, 60, 52, 51, 63]. The estimation of an optimal value of the eddy viscosity parameter has been the topic for many research works though. For example, empirical relations can be adopted [296, 49, 297], or ideas can be borrowed from LES frameworks to dynamically compute a better approximation of the eddy viscosity parameter [59, 47, 298, 147]. Moreover, a 4D-VAR approach has been suggested to provide an optimal nonlinear eddy viscosity estimate in Galerkin projection based ROMs[51]. An adaptive nudging technique has also been recently introduced to force ROMs towards the reference solution corresponding to the observed data [283].

Instead, in the present chapter, we propose a novel framework to estimate eddy viscosity closure using noisy observations from a sparse observation network. In particular, we adopt the forward sensitivity method to evaluate the sensitivity of ROM predictions to the eddy viscosity parameter. Observations, whenever available, can therefore be used to approximate a more representative value of eddy viscosity to better reflect the true system's dynamics. We highlight that the proposed approach is very suitable for emerging digital twin applications [130, 299, 300, 13, 234, 301], where real-time measurements are abundant (and noisy). Thus, efficiently assimilating these measurements to improve ROMs can be a key enabler for such applications which require many-query and near real-time simulations. We test our approach using two test cases of varying complexity, namely the one-dimensional viscous Burgers equation with a square wave representing a moving shock and the two-dimensional vorticity transport equation applied to Kraichnan turbulence. We apply the proposed GROM-FSM to assimilate information from either full field or sparse field measurements. Therefore, our approach provides a modular framework to optimally estimate closure

parameters for submodal scale physics, which can be effectively used in emerging sensor-centric applications in transport processes.

The rest of the chapter is outlined here. In Section 6.3, we review the forward sensitivity method and its mathematical foundation as an established data assimilation algorithm. We then construct the standard Galerkin ROM and the corresponding reduced operators for the 1D Burgers problem and the 2D vorticity transport equation in Section 6.4. Then, we describe the proposed approach for closure estimation via FSM, namely GROM-FSM, in Section 6.5. Results and relevant discussions are provided in Section 6.6. In particular, we consider the assimilation of full field and sparse field measurements. For the latter, we explore two approaches for assimilating information from sparse observations. We also extend the eddy viscosity estimation framework to permit mode-dependent closures. Concluding remarks and insights are drawn in Section 6.7.

6.3 Forward Sensitivity Method

In this section, we briefly describe the forward sensitivity method (FSM) proposed by Lakshmivarahan and Lewis [278]. The idea behind this technique is to find optimal control parameters by iteratively correcting the control for the least squares fit of the model to the observational data. The control parameters in question here can be any unknown such as initial conditions, boundary conditions, and physical model parameters. The correction to each control parameter is dictated by its corresponding sensitivity function. In essence, the sensitivity function is the quantitative measure of influence of each control parameter on the model states. The nature of combining physical models with actual data to solve an inverse problem is what makes FSM a modular DA approach.

Let the dynamical system of interest be defined by a set of ordinary differential equations (ODEs) as below,

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \boldsymbol{\alpha}), \tag{6.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the system state-vector with the initial condition $\mathbf{x}^0$ and $\boldsymbol{\alpha} \in \mathbb{R}^p$ denotes the physical parameters. The vector of control parameters is represented as $\mathbf{c} = [\mathbf{x}^0, \boldsymbol{\alpha}]^T \in \mathbb{R}^{n+p}$. Here, it is assumed that the solution $\mathbf{x}(t)$ exists and is unique and has a smooth dependence with the control vector $\mathbf{c}$.

Discretizing Eq. (6.1) by using some numerical method like Runge-Kutta schemes,

we get a model equation which gives the evolution of model states in discrete time as,

$$\mathbf{x}^{k+1} = \mathbf{M}(\mathbf{x}^k, \boldsymbol{\alpha}), \tag{6.2}$$

where $\mathbf{x}^k = [x_1^k, x_2^k, \ldots, x_n^k]^T$ denotes the time-discretized model states at discrete time t_k and $\mathbf{M} = [M_1(\mathbf{x}^k, \boldsymbol{\alpha}), M_2(\mathbf{x}^k, \boldsymbol{\alpha}), \ldots, M_n(\mathbf{x}^k, \boldsymbol{\alpha})]^T$ refer to the state transition maps from time t_k to t_{k+1}. Differentiating Eq. (6.2) with respect to $\mathbf{x}^0$, we get

$$\frac{\partial x_i^{k+1}}{\partial x_j^0} = \sum_{q=1}^{n} \left(\frac{\partial M_i}{\partial x_q^k}\right) \left(\frac{\partial x_q^k}{\partial x_j^0}\right), \tag{6.3}$$

where $1 \leqslant i, j \leqslant n$. Similarly, differentiating Eq. (6.2) with respect to $\boldsymbol{\alpha}$, we obtain

$$\frac{\partial x_i^{k+1}}{\partial \alpha_j} = \sum_{q=1}^{n} \left(\frac{\partial M_i}{\partial x_q^k}\right) \left(\frac{\partial x_q^k}{\partial \alpha_j}\right) + \frac{\partial M_i}{\partial \alpha_j} \tag{6.4}$$

where $1 \leqslant i \leqslant n$ and $1 \leqslant j \leqslant p$. In Eqs. (6.3) and (6.4), the superscript refers to the discrete time index while the subscript refers to the specific component. Now, we can define $\mathbf{U}^k$ as the sensitivity matrix of $\mathbf{x}^k$ with respect to initial state, where $[\mathbf{U}^k]_{ij} = \partial x_i^k / \partial x_j^0$ for $1 \leqslant i, j \leqslant n$. Also, we define $\mathbf{V}^k$ as the sensitivity matrix of $\mathbf{x}^k$ with respect to the parameter-vector $\boldsymbol{\alpha}$, where $[\mathbf{V}^k]_{ij} = \partial x_i^k / \partial \alpha_j$ for $1 \leqslant i \leqslant n$ and $1 \leqslant j \leqslant p$. Then, we can rewrite Eqs. (6.3) and (6.4) in matrix as below

$$\mathbf{U}^{k+1} = \mathbf{D}_{\mathbf{x}}^k(\mathbf{M})\mathbf{U}^k, \tag{6.5}$$

$$\mathbf{V}^{k+1} = \mathbf{D}_{\mathbf{x}}^k(\mathbf{M})\mathbf{V}^k + \mathbf{D}_{\boldsymbol{\alpha}}(\mathbf{M}), \tag{6.6}$$

initialized as $\mathbf{U}^0 = \mathbf{I}$ and $\mathbf{V}^0 = \mathbf{0}$.

Here, $\mathbf{D}_{\mathbf{x}}^k(\mathbf{M})$ and $\mathbf{D}_{\boldsymbol{\alpha}}(\mathbf{M})^k$ are the Jacobian matrices of $\mathbf{M}(\cdot)$ with respect to $\mathbf{x}$ and $\boldsymbol{\alpha}$ at discrete time t_k, respectively. Moreover, $\mathbf{U}^k \in \mathbb{R}^{n \times n}$ and $\mathbf{V}^k \in \mathbb{R}^{n \times p}$ are called the forward sensitivity matrices with respect to initial conditions and parameters, respectively. In effect, the system dynamics in Eq. (6.2) gets reduced to a set of linear matrix equations (i.e., Eqs. (6.5) and (6.6)) which give the evolution of the sensitivity matrices in discrete time. By first order approximation, we have

$$\Delta\mathbf{x}^k \approx \delta\mathbf{x}^k = \mathbf{U}^k\delta\mathbf{x}^0 + \mathbf{V}^k\delta\boldsymbol{\alpha}, \tag{6.7}$$

where $\delta\mathbf{x} \in \mathbb{R}^n$.

So far, no observational data have been used. Let $\mathbf{z}(t) \in \mathbb{R}^m$ be the observation vector available for N time snapshots; and $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^m$ maps the model space $\mathbb{R}^n$ to the observation space $\mathbb{R}^m$. Hence, the observation vector can be defined mathematically as follows,

$$\mathbf{z}(t) = \mathbf{h}(\tilde{\mathbf{x}}(t)) + \mathbf{v}(t), \tag{6.8}$$

where $\tilde{\mathbf{x}}(t) \in \mathbb{R}^n$ is the true state of the system and $\mathbf{v}(t) \in \mathbb{R}^m$ represents the measurement noise, which is assumed to be white Guassian noise with zero mean and covariance matrix $\mathbf{R}(t) \in \mathbb{R}^{m \times m}$. Writing Eq. (6.8) in the discrete-time form we get,

$$\mathbf{z}^k = \mathbf{h}(\tilde{\mathbf{x}}^k) + \mathbf{v}^k, \tag{6.9}$$

where $\mathbf{v}^k$ is white Gaussian noise with the covariance matrix $\mathbf{R}^k$. In most cases, $\mathbf{R}^k$ is a diagonal matrix. For simplicity, we assume that $\mathbf{R}^k = \sigma_{Obs}^2 \mathbf{I}_m$, where $\mathbf{I}_m$ is the $m \times m$ identity matrix.

Assuming that the model is a perfect representation of the actual physical phenomenon and given a starting guess value of the control $\mathbf{c}$, we can run the model forward to predict $\mathbf{x}^k \; \forall \; 1 \leqslant k \leqslant N$, then the forecast error $\mathbf{e}_F^k \in \mathbb{R}^m$ defined as,

$$\mathbf{e}_F^k = \mathbf{z}^k - \mathbf{h}(\mathbf{x}^k). \tag{6.10}$$

The forecast error $\mathbf{e}_F^k$ is composed of the sum of a deterministic part defined as $\mathbf{h}(\tilde{\mathbf{x}}^k) - \mathbf{h}(\mathbf{x}^k)$ and a random part $\mathbf{v}^k$. The random error stems from the inherent error in the mapping $\mathbf{h} : \mathbf{x}^k \to \mathbf{z}^k$ and we have no control on it, however it is the goal of FSM to minimize the deterministic part in a least squares sense at all the N time snaps by choosing an optimal value for $\mathbf{c}$.

Now, the goal of FSM is to find a perturbation to the control $\delta\mathbf{c}$ from the given starting guess $\mathbf{c}$. This, in turn, would cause a $\delta\mathbf{x}^k$ change in $\mathbf{x}^k$ such that the actual observation matches with the forecast observation from the model as follows,

$$\mathbf{z}^k = \mathbf{h}(\mathbf{x}^k + \delta\mathbf{x}^k) \approx \mathbf{h}(\mathbf{x}^k) + \mathbf{D}_{\mathbf{x}}^k(\mathbf{h})\delta\mathbf{x}^k. \tag{6.11}$$

Thus, the forecast error $\mathbf{e}_F^k$ can be written as,

$$\mathbf{e}_F^k = \mathbf{D}_{\mathbf{x}}^k(\mathbf{h})\delta\mathbf{x}^k. \tag{6.12}$$

Combining Eq. (6.7) with Eq. (6.12), and setting $\mathbf{H}_1^k = \mathbf{D}_{\mathbf{x}}^k(\mathbf{h})\mathbf{U}^k \in \mathbb{R}^{m\times n}$, $\mathbf{H}_2^k = \mathbf{D}_{\mathbf{x}}^k(\mathbf{h})\mathbf{V}^k \in \mathbb{R}^{m\times p}$, we get,

$$\mathbf{e}_F^k = \mathbf{H}_1^k \delta\mathbf{x}^0 + \mathbf{H}_2^k \delta\boldsymbol{\alpha}. \tag{6.13}$$

Equation (6.13) can be further simplified and written in terms of the perturbation to the control $\delta\mathbf{c}$ as

$$\mathbf{H}^k \delta\mathbf{c} = \mathbf{e}_F^k, \tag{6.14}$$

where $\mathbf{H}^k = [\mathbf{H}_1^k, \mathbf{H}_2^k] \in \mathbb{R}^{m\times(n+p)}$ and $\delta\mathbf{c} = [\delta\mathbf{x}^0, \delta\boldsymbol{\alpha}]^T \in \mathbb{R}^{n+p}$.

Equation (6.14) can be formulated for all the N time snapshots for which observations are available and the following linear equation is obtained,

$$\mathbf{H}\delta\mathbf{c} = \mathbf{e}_F, \tag{6.15}$$

where the matrix $\mathbf{H} \in \mathbb{R}^{Nm\times(n+p)}$ and the vector $\mathbf{e}_F \in \mathbb{R}^{Nm}$ are defined as follows,

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}^1 \\ \mathbf{H}^2 \\ \vdots \\ \mathbf{H}^N \end{bmatrix}, \qquad \mathbf{e}_F = \begin{bmatrix} \mathbf{e}_F^1 \\ \mathbf{e}_F^2 \\ \vdots \\ \mathbf{e}_F^N \end{bmatrix}. \tag{6.16}$$

Depending on the value of Nm relative to $(n+p)$, Eq. (6.15) can give rise to either an over-determined or an under-determined linear inverse problem. In either case, the inverse problem can be solved in a weighted least squares sense to find an optimal value of $\delta\mathbf{c}$, with $\mathbf{R}^{-1}$ as a weighting matrix, where $\mathbf{R}$ is a block-diagonal matrix constructed as follows,

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}^1 & & & \\ & \mathbf{R}^2 & & \\ & & \ddots & \\ & & & \mathbf{R}^N \end{bmatrix}. \tag{6.17}$$

For simplicity, we assume that $\mathbf{R}$ is a diagonal matrix defined as $\mathbf{R} = \sigma_{Obs}^2 \mathbf{I}_{Nm}$, where $\mathbf{I}_{Nm}$ is the $Nm \times Nm$ identity matrix. Then, the solution of Eq. (6.15) can be written as

$$\delta\mathbf{c} = \begin{cases} \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{e}_F, & \text{over-determined,} \\ \mathbf{R}^{-1}\mathbf{H}^T\left(\mathbf{H}\mathbf{R}^{-1}\mathbf{H}^T\right)^{-1}\mathbf{e}_F, & \text{under-determined.} \end{cases} \tag{6.18}$$

It has been seen that the first order approximation progressively yield better results by repeating the entire process for multiple iterations until convergence with certain tolerance [278].

6.4 Reduced Order Modeling

In this section, we briefly derive a reduced order model (ROM) for a dynamical system governed by the following autonomous partial differential equation (PDE)

$$\frac{\partial q}{\partial t} = \mathcal{F}(q), \tag{6.19}$$

where q is the state of system (flow field variables) and $\mathcal{F}(q)$ governs the dynamics of q. We follow the standard Galerkin projection to construct the sought ROM which includes two main steps. First, the flow field variable $q(\mathbf{x}, t)$ (where $q(\mathbf{x}, t)$ represents the vectorized form of q at time t) is approximated as a linear superposition of the contributions of a few modes, which can be mathematically expressed as

$$q(\mathbf{x}, t) = \bar{q}(\mathbf{x}) + \sum_{k=1}^{R} a_k(t)\phi_k(\mathbf{x}), \tag{6.20}$$

where $\bar{q}(\mathbf{x})$ represents the mean-field, $\phi_k(\mathbf{x})$ are the spatial modes (or basis functions), $a_k(t)$ are the time-dependent modal coefficients (i.e., weighting functions), and R is the number of retained modes in ROM approximation (i.e., ROM dimension). The second step is to project the governing equation (i.e., Eq. (6.19)) onto the subspace spanned by $\{\phi_k\}_{k=1}^{R}$. Thus, the two main ingredients for building a Galerkin ROM (GROM) are the basis functions $\{\phi_k\}_{k=1}^{R}$ and a Galerkin projection of the governing equation. To compute the basis functions $\{\phi_k\}_{k=1}^{R}$, we follow the popular proper orthogonal decomposition (POD) approach described in Section 6.4.1, followed by derivation of GROM equations in Section 6.4.2. Overall, this GROM approach utilizes a linear decomposition technique that is able to properly treat the nonlinearity of $\mathcal{F}(q)$, since it accounts for nonlinear coupling of terms acting within the linear space defined by the POD basis functions [16].

6.4.1 Proper orthogonal decomposition

Proper orthogonal decomposition (POD) is a data-driven modal decomposition technique that gained remarkable popularity in the fluid mechanics community due to its

simplicity as well as robustness. Given a set of solution trajectories or realizations (known as snapshots), POD lays out a systematic approach to compute a solution-adapted basis functions that provide the optimal basis to represent a given set of simulation data or snapshots. Specifically, POD produces hierarchically organized basis functions, based on their contribution to the total system's energy, which makes the modal selection a trivial process. In particular, given a collection of system realizations, we build a snapshot matrix $\mathbf{A} \in \mathbb{R}^{n \times N}$ as follows,

$$\mathbf{A} = \begin{bmatrix} q(x_1, t_1) & q(x_1, t_2) & \dots & q(x_1, t_N) \\ q(x_2, t_1) & q(x_2, t_2) & \dots & q(x_2, t_N) \\ \vdots & \vdots & \ddots & \vdots \\ q(x_n, t_1) & q(x_n, t_2) & \dots & q(x_n, t_N) \end{bmatrix}, \tag{6.21}$$

where n is the number of spatial locations and N is the number of snapshots. A mean-subtracted snapshot matrix $\tilde{\mathbf{A}}$ is defined as $\tilde{\mathbf{A}} = \mathbf{A} - \frac{1}{N}\mathbf{A}\mathbf{1}_{N \times N}$, where $\mathbf{1}_{N \times N}$ is an $N \times N$ matrix of ones. Then, a thin singular value decomposition (SVD) is performed on $\tilde{\mathbf{A}}$ as follows ,

$$\tilde{\mathbf{A}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{6.22}$$

where $\mathbf{U} \in \mathbb{R}^{n \times N}$ is a matrix with orthonormal columns are the left singular vectors of $\tilde{\mathbf{A}}$, which represent the spatial basis as,

$$\mathbf{U} = \begin{bmatrix} U_1(x_1) & U_2(x_1) & \dots & U_N(x_1) \\ U_1(x_2) & U_2(x_2) & \dots & U_N(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ U_1(x_n) & U_2(x_n) & \dots & U_N(x_n) \end{bmatrix}, \tag{6.23}$$

while the columns of $\mathbf{V} \in \mathbb{R}^{N \times N}$ are the right singular vectors of $\tilde{\mathbf{A}}$, representing the temporal basis as

$$\mathbf{V} = \begin{bmatrix} V_1(t_1) & V_2(t_1) & \dots & V_N(t_1) \\ V_1(t_2) & V_2(t_2) & \dots & V_N(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ V_1(t_N) & V_2(t_N) & \dots & V_N(t_N) \end{bmatrix}. \tag{6.24}$$

The singular values of $\tilde{\mathbf{A}}$ are stored in descending order as the entries of the diagonal matrix $\mathbf{\Sigma} \in \mathbb{R}^{N \times N}$,

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_N \end{bmatrix}, \tag{6.25}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_N \geq 0$. For dimensionality reduction purposes, only the first R columns of $\mathbf{U}$, corresponding to the largest R singular values, are stored. Those represent the most effective R POD modes, denoted as $\{\phi_k\}_{k=1}^{R}$ in the rest of the manuscript. The computed basis functions are orthonormal by construction as

$$\langle \phi_i; \phi_j \rangle = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{otherwise,} \end{cases} \tag{6.26}$$

where the angle parentheses $\langle \cdot; \cdot \rangle$ stands for the standard inner product in Euclidean space (i.e., dot product). We note that the presented direct algorithm might be unfeasible for larger data sets, as stacking snapshots into a single huge matrix is usually prohibitive. Instead, the method of snapshots [83] can be followed to efficiently approximate the POD bases.

6.4.2 Galerkin ROM

Having a set of POD basis functions in hand, an orthogonal projection can be performed to obtain the Galerkin ROM (GROM). To do so, the ROM approximation (i.e., Eq. (6.20)) is substituted into the governing equation and an inner product with the POD basis functions is carried out. In deriving the GROM equations, we highlight that the POD bases are only spatial functions (i.e., independent of time) and the modal coefficients are independent of space. We also utilize the orthonormality property of the basis functions to get the following set of ordinary differential equations (ODEs) representing the tensorial GROM,

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \mathfrak{B}_k + \sum_{i=1}^{R} \mathfrak{L}_{i,k} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,k} a_i a_j, \tag{6.27}$$

where $\mathfrak{B}$, $\mathfrak{L}$ and $\mathfrak{N}$ are the vector, matrix and tensor of predetermined model coefficients corresponding to constant, linear and nonlinear terms, respectively. We note

here that the last term results from the quadratic nonlinearity encountered in most of the fluid flow systems. In particular, we consider here two cases of particular interest. First, we consider the one-dimensional Burgers equation as a prototypical test bed for transport systems with quadratic nonlinearity and Laplacian dissipation. For this case, we solve the problem of a moving shock, which can be considered as a challenging case for ROM applications [297]. In the second case, we consider the vorticity transport equation, with an application to the two-dimensional decaying turbulence.

6.4.2.1 1D Burgers problem

The one-dimensional (1D) Burgers equation is defined with the following partial differential equation (PDE)

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + \frac{1}{\mathrm{Re}}\frac{\partial^2 u}{\partial x^2}, \tag{6.28}$$

where Re is Reynolds number relating the inertial and viscous effects. Using the following definition

$$u(x,t) = \bar{u}(x) + \sum_{k=1}^{R} a_k(t)\phi_k(x), \tag{6.29}$$

the GROM model coefficients can be precomputed during an offline stage as

$$\begin{aligned}
\mathfrak{B}_k &= \big\langle -\bar{u}\frac{\partial \bar{u}}{\partial x} + \frac{1}{\mathrm{Re}}\frac{\partial^2 \bar{u}}{\partial x^2}; \phi_k \big\rangle, \\
\mathfrak{L}_{i,k} &= \big\langle -\bar{u}\frac{\partial \phi_i}{\partial x} - \phi_i\frac{\partial \bar{u}}{\partial x} + \frac{1}{\mathrm{Re}}\frac{\partial^2 \phi_i}{\partial x^2}; \phi_k \big\rangle, \\
\mathfrak{N}_{i,j,k} &= \big\langle -\phi_i\frac{\partial \phi_j}{\partial x}; \phi_k \big\rangle.
\end{aligned}$$

6.4.2.2 2D Kraichnan turbulence

The two-dimensional (2D) Kraichnan turbulence problem models how randomly generated vortices evolve [302]. Despite the apparent simplicity, the decaying 2D Kraichnan turbulence is very rich in its dynamics and follows the 2D Navier-Stokes equations, which can be written in vorticity-streamfunction formulation (vorticity-transport equation) as follows

$$\frac{\partial \omega}{\partial t} = -J(\omega,\psi) + \frac{1}{\mathrm{Re}}\nabla^2\omega, \tag{6.30}$$

where ω is the vorticity and ψ is the streamfunction. $J(\omega, \psi)$ and $\nabla^2\omega$ are the Jacobian and Laplacian operators, respectively, which can be defined as

$$J(\omega, \psi) = \frac{\partial \omega}{\partial x}\frac{\partial \psi}{\partial y} - \frac{\partial \omega}{\partial y}\frac{\partial \psi}{\partial x}, \tag{6.31}$$

$$\nabla^2\omega = \frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}. \tag{6.32}$$

The vorticity-streamfunction formulation enforces the incompressibility condition, where the vorticity and streamfunction fields are related by the following Poisson equation

$$\nabla^2\psi = -\omega. \tag{6.33}$$

With the POD algorithm implemented, a set of POD basis functions $\{\phi_k(x, y)\}_{k=1}^{R}$ are obtained from the snapshots of vorticity fields. The prognostic variable vorticity in Eq. (6.30) is defined as

$$\omega(x, y, t) = \bar{\omega}(x, y) + \sum_{k=1}^{R} a_k(t)\phi_k(x, y). \tag{6.34}$$

Since the vorticity and streamfunction are related by the kinematic relationship given by Eq. (6.33), the basis functions ($\theta_k(x, y)$) and mean field ($\bar{\psi}(x, y)$) corresponding to the streamfunction can be obtained from those of the vorticity as follows,

$$\nabla^2\bar{\psi}(x, y) = -\bar{\omega}(x, y), \tag{6.35}$$

$$\nabla^2\theta_k(x, y) = -\phi_k(x, y), \quad k = 1, 2, \ldots, R, \tag{6.36}$$

which might result in a set of basis functions for the streamfunction that are not necessarily orthonormal. Moreover, the reduced order approximation of streamfunction shares the same temporal coefficients $a_k(t)$,

$$\psi(x, y, t) = \bar{\psi}(x, y) + \sum_{k=1}^{R} a_k(t)\theta_k(x, y). \tag{6.37}$$

The GROM coefficients for this case can be defined as follows [43],

$$\begin{aligned}
\mathfrak{B}_k &= \big\langle -J(\bar{\omega},\bar{\psi}) + \frac{1}{\mathrm{Re}}\nabla^2\bar{\omega};\phi_k\big\rangle,\\
\mathfrak{L}_{i,k} &= \big\langle -J(\bar{\omega},\theta_i) - J(\phi_i,\bar{\psi}) + \frac{1}{\mathrm{Re}}\nabla^2\phi_i;\phi_k\big\rangle,\\
\mathfrak{N}_{i,j,k} &= \big\langle -J(\phi_i,\theta_j);\phi_k\big\rangle.
\end{aligned} \tag{6.38}$$

Due to the quadratic nonlinearity in the aforementioned systems, the computational cost of solving Eq. (6.27) is $O(R^3)$. Therefore, the number of retained modes has to be reduced as much as possible to keep the computational cost affordable. However, this truncation ignores the dyadic interactions between the first R modes and the remaining ones. As a result, an erroneous behaviour might arise in the ROM solution [44–46, 64], and closure/stabilization techniques have been introduced to improve ROM accuracy [61, 59, 62, 55, 52]. As highlighted earlier in Section 6.2, closure approaches based on physical significance have been usually relied on the analogy between LES and ROMs, e.g., the addition of an artificial viscosity term [294]. Recently, data-driven closure methods have been also pursued, e.g., using variational multiscale techniques [303, 304, 298], machine learning algorithms [305, 145, 306, 307], and polynomial approximations [57, 58].

6.5 Closure Estimation via FSM

In order to stabilize the GROM, a closure model is usually necessary for complex flows. In the present chapter, we consider adding a linear eddy-viscosity term to the governing equation as follows,

$$\text{1D Burgers:}\quad \frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} + (\nu+\nu_e)\frac{\partial^2 u}{\partial x^2}, \tag{6.39}$$

$$\text{2D Turbulence:}\quad \frac{\partial \omega}{\partial t} = -J(\omega,\psi) + (\nu+\nu_e)\nabla^2\omega, \tag{6.40}$$

where ν is the physical (kinematic) viscosity and ν_e is an (artificial) eddy viscosity to add an extra dissipation to stabilize the system. If we follow the same procedure in Section 6.4.2, we get the following GROM with closure,

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \mathfrak{B}_k + \nu_e\widehat{\mathfrak{B}}_k + \sum_{i=1}^{R}\mathfrak{L}_{i,k}a_i + \nu_e\sum_{i=1}^{R}\widehat{\mathfrak{L}}_{i,k}a_i + \sum_{i=1}^{R}\sum_{j=1}^{R}\mathfrak{N}_{i,j,k}a_ia_j, \tag{6.41}$$

where $\widehat{\mathfrak{B}}$ and $\widehat{\mathfrak{L}}$ are the constant and linear coefficients resulting from the introduction of the eddy viscosity term and defined as follows

$$\begin{aligned}
\text{1D Burgers:} \quad & \widehat{\mathfrak{B}}_k = \Big\langle \frac{\partial^2 \bar{u}}{\partial x^2}; \phi_k \Big\rangle, \quad \widehat{\mathfrak{L}}_{i,k} = \Big\langle \frac{\partial^2 \phi_i}{\partial x^2}; \phi_k \Big\rangle, \\
\text{2D Turbulence:} \quad & \widehat{\mathfrak{B}}_k = \big\langle \nabla^2 \bar{\omega}; \phi_k \big\rangle, \quad \widehat{\mathfrak{L}}_{i,k} = \big\langle \nabla^2 \phi_i; \phi_k \big\rangle.
\end{aligned}$$

It remains to compute or assume a good estimate for ν_e. Using an a priori estimate for ν_e can produce a stable ROM solution. However, as the flow evolves, this prior value might become less effective. Therefore, there should be a strategy to dynamically update this estimate based on the flow conditions/regimes.

In this regard, we borrow ideas from meteorological data assimilation to correct and update our parameter estimate using live and realistic (possibly noisy) measurements. In particular, we use the forward sensitivity method (FSM), described in Section 6.3, to compute an optimal value for eddy viscosity given a few field observations. This also allows us to update our estimate whenever a new observation becomes available. We start with a prior estimate of eddy viscosity (e.g., zero if no priors are available), and solve the ROM equation for a given period of time, T_w. As we solve GROM, we also collect some field measurements during this period T_w. A penalty term is thus computed as the difference between the GROM prediction and observations, which is used to update our prior estimate for ν_e. This updated value is therefore used to evolve the GROM until new observations become available to match with model's predictions, and so on. The period over which measurements are collected T_w is called the data assimilation window. Note that model's states (e.g., $a_k(t)$) can be different from the measured quantities (e.g., $u(x,t)$), and a mapping between model space and observation space has to be defined. In the following, we formalize our framework for FSM-based eddy viscosity estimation for GROM, called GROM-FSM in the present study. Defining our dynamic model as

$$\frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t} = \mathbf{f}(\mathbf{a}, \nu_e), \tag{6.42}$$

where $\mathbf{a}$ is the vector of modal coefficients defined as $\mathbf{a} = [a_1, a_2, \ldots, a_R]^T$ (the superscript T denotes transpose). The (time-continuous) model map $\mathbf{f} = [f_1, f_2, \ldots, f_R]^T$ is defined as $f_k = \mathfrak{B}_k + \nu_e \widehat{\mathfrak{B}}_k + \sum_{i=1}^{R} \mathfrak{L}_{i,k} a_i + \nu_e \sum_{i=1}^{R} \widehat{\mathfrak{L}}_{i,k} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,k} a_i a_j$.

A time-discretization scheme can be utilized to convert this model from continuous-

time map $\mathbf{f}$ to a discrete-time map $\mathbf{M}$ as

$$\mathbf{a}^{k+1} = \mathbf{M}(\mathbf{a}^k, \nu_e), \tag{6.43}$$

where the superscript k denotes the time index. In our implementation, we adopt the fourth-order Runge-Kutta scheme (RK4) for temporal discretization.

Suppose we collect measurements $\mathbf{z}^k$ at a single time instant t_k, where $t_k \in [0, T_w]$. The forecast error is defined at t_k as

$$\mathbf{e}_F^k = \mathbf{z}^k - \mathbf{h}(\mathbf{a}^k), \tag{6.44}$$

where $\mathbf{h}(\cdot)$ defines the mapping from model space to observation space. In our results, we consider two mapping cases. In the first case, we preprocess field observations to compute the "observed" coefficients (i.e., $\mathbf{z}^k = \mathbf{a}_{Obs}^k$), where the mapping is simply identity matrix (i.e., $\mathbf{h}(\mathbf{a}^k) = \mathbf{a}^k$). In the second case, we keep observations as velocity field measurement ($\mathbf{z}^k = \mathbf{u}_{Obs}^k$), where the mapping becomes a reconstruction map (i.e., $\mathbf{h}(\mathbf{a}^k) = \mathbf{u}^k$). Specific details are to be given in Section 6.6.

Although the FSM can be used to treat uncertainties in initial conditions as well as model parameters, we only consider the estimation of the eddy viscosity parameter ν_e. Thus,

$$\mathbf{H}_2^k \delta\nu_e = \mathbf{e}_F^k, \tag{6.45}$$

where $\mathbf{H}_2^k = \mathbf{D}_{\mathbf{a}}^k(\mathbf{h})\mathbf{V}^k$ as defined in Section 6.3. Details of defining model Jacobian are given in **Appendix A**. For more than a single observation time, we stack Eq. (6.45) at different observation times to get the following equation,

$$\mathbf{H}_2 \delta\nu_e = \mathbf{e}_F. \tag{6.46}$$

Also, a block-diagonal matrix $\mathbf{R}$ is constituted with the measurement covariance matrices $\mathbf{R}^k$ at subsequent observation times. Equation (6.46) defines an over-determined system of linear equations in $\delta\nu_e$. A weighted least-squares solution can be computed, with a weighting matrix of $\mathbf{R}^{-1}$ as follows,

$$\delta\nu_e = \left(\mathbf{H}_2^T \mathbf{R}^{-1} \mathbf{H}_2\right)^{-1} \mathbf{H}_2^T \mathbf{R}^{-1} \mathbf{e}_F, \tag{6.47}$$

where $\delta\nu_e$ is added to our prior estimate of ν_e (also called background) to obtain a

better approximation and the process is repeated until convergence. The procedure for using FSM to compute the eddy viscosity is summarized in Algorithm 1. A tolerance limit has to be set to define convergence (e.g., 1×10^{-6}). We also note that an initial guess for eddy viscosity parameter is required for proper implementation of the algorithm. If no prior knowledge of ν_e is available, a zero initial guess usually works fine. Meanwhile, since collected FOM snapshots are already available during an offline stage, they can be treated as field measurement data with negligible noise (corresponding to the underlying solution's assumptions and numerical approximations). Thus, Algorithm 1 can be applied offline along with the construction of GROM model to provide a prior estimate of the suitable closure parameter.

Algorithm 1: Forward sensitivity method for estimating eddy viscosity in GROM closure

Input : Dynamic model $\mathbf{M}(\cdot)$, observation operator $\mathbf{h}(\cdot)$, initial condition $\mathbf{a}^1$, a set of observations $\mathbf{z}^1, \mathbf{z}^2, \dots \mathbf{z}^N$, an initial guess for eddy viscosity parameter ν_e, and a tolerance *tol* value

Output: An estimate of the eddy viscosity ν_e

initialization

for $i \leftarrow 1$ **to** *max iter* **do**

- $\mathbf{V}^1 = \mathbf{0}$
- $\mathbf{e}_F = \mathbf{z}^1 - \mathbf{h}(\mathbf{a}^1)$
- $\mathbf{H}_2 = \mathbf{D}^1_{\mathbf{a}}(\mathbf{h})\mathbf{V}^1$
- $\mathbf{R} = \mathbf{R}^1$
- **for** $n \leftarrow 1$ **to** $N-1$ **do**
  - $\mathbf{a}^{n+1} = \mathbf{M}(\mathbf{a}^n, \nu_e)$
  - $\mathbf{V}^{n+1} = \mathbf{D}^n_{\mathbf{a}}(\mathbf{M})\mathbf{V}^n + \mathbf{D}^{\mathbf{n}}_{\boldsymbol{\nu}_\mathbf{e}}(\mathbf{M})$
  - **if** *(observation* $\mathbf{z}^{n+1}$ *is available)* **then**
    - $\mathbf{e}_F^{n+1} = \mathbf{z}^{n+1} - \mathbf{h}(\mathbf{a}^{n+1})$
    - $\mathbf{H}_2^{n+1} = \mathbf{D}^{n+1}_{\mathbf{a}}(\mathbf{h})\mathbf{V}^{n+1}$
    - $\mathbf{e}_F = \begin{bmatrix} \mathbf{e}_F \\ \mathbf{e}_F^{n+1} \end{bmatrix}, \qquad \mathbf{H}_2 = \begin{bmatrix} \mathbf{H}_2 \\ \mathbf{H}_2^{n+1} \end{bmatrix}, \qquad \mathbf{R} = \begin{bmatrix} \mathbf{R} & \\ & \mathbf{R}^{n+1} \end{bmatrix}$
  - **end**
- **end**
- $\delta\nu_e = \left(\mathbf{H}_2^T\mathbf{R}^{-1}\mathbf{H}_2\right)^{-1}\mathbf{H}_2^T\mathbf{R}^{-1}\mathbf{e}_F$
- **if** *(*$\delta\nu_e \le tol$*)* **then**
  - *break*
- **else**
  - $\nu_e = \nu_e + \delta\nu_e$
- **end**

end

6.6 Results

In this section, we present our results for the utilization the proposed methodology to compute and update the eddy viscosity parameter via FSM, applied to the introduced two test problems (i.e., 1D Burgers problem and 2D Kraichnan turbulence).

6.6.1 1D Burgers problem

For the 1D Burgers problem, we assume an initial condition of a square wave defined as

$$u(x,0) = \begin{cases} 1, & \text{if } 0 < x \leq L/2, \\ 0, & \text{if } L/2 < x \leq L, \end{cases} \tag{6.48}$$

with zero Dirichlet boundary conditions, $u(0,t) = u(L,t) = 0$. We consider a spatial domain of $L = 1$, and solve at $\text{Re} = 10^4$ for $t \in [0,1]$. For numerical computations, we use a family of fourth order compact schemes for spatial derivatives [308], and skew-symmetric formulation for the nonlinear term. Also, we use the fourth order Runge-Kutta (RK4) scheme for temporal integration with a time step of 10^{-4} over a spatial grid of 4096. For POD basis generation, we collect 100 snapshots (i.e., every 100 time steps). The temporal evolution of the 1D Burgers problem using the described setup is shown in Fig. 6.1, where we can see the advection of the shock wave.

The described Burgers problem with square wave is challenging for ROM applications. In our GROM implementation, we consider $R = 8$ modes and $\Delta t = 0.01$ for time integration. In the following, we discuss the estimation of eddy viscosity via FSM using full and sparse field measurements.

6.6.1.1 Full field measurement

In our first case, we investigate the assimilation of noisy full field measurement as

$$u_{Obs}(x,t) = u(x,t) + v(x,t), \tag{6.49}$$

where $v(x,t)$ is a white Gaussian noise with zero mean and covariance matrix $\mathbf{R}(t)$. In particular, we define $\mathbf{R}(t) = \sigma_{Obs}^2 \mathbf{I}$, with $\sigma_{Obs} = 0.1$. We assume a data assimilation window of 0.5 s and collect measurements at $t = 0.25$ and $t = 0.5$, as demonstrated in Fig. 6.2.

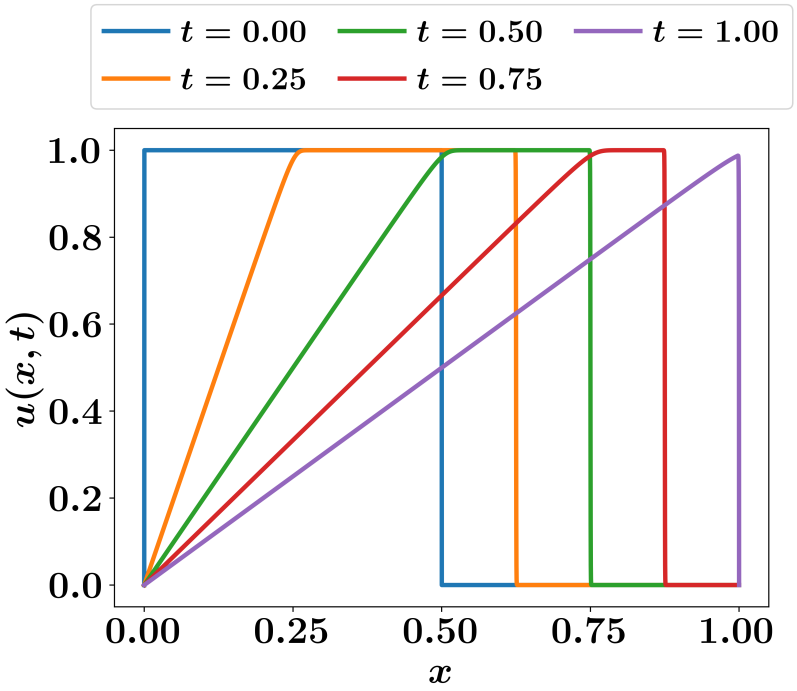


Figure 6.1: Evolution of the FOM velocity field, characterized by a moving shock with square wave.

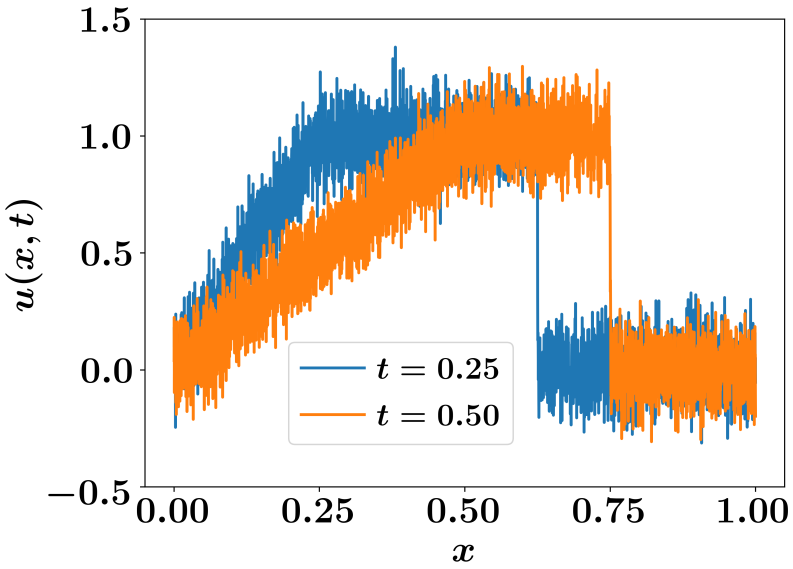


Figure 6.2: Noisy measurement of velocity fields at $t = 0.25$ s and $t = 0.50$ s, assuming sensors are located at all grid points.

Instead of defining a map between model space and observation space, we pre-process our measurement by projecting them onto the POD basis to compute the

"observed" modal coefficients as

$$a^k_{i,Obs} = \langle \mathbf{u}^k_{Obs} - \bar{\mathbf{u}}; \phi_i \rangle. \tag{6.50}$$

Thus, $\mathbf{z}^k = \mathbf{a}^k_{Obs}$ and the observation operator is defined $\mathbf{h}(\mathbf{a}) = \mathbf{a}$, with a Jacobian equal to the identity matrix (i.e., $\mathbf{D}_\mathbf{a}(\mathbf{h}) = \mathbf{I}_R$, where $\mathbf{I}_R$ is the $R \times R$ identity matrix). Also, the observational covariance matrix is set as $\mathbf{R}^k = \sigma^2_{Obs}\mathbf{I}_R$. If we implement the procedure described in Section 6.3 to obtain an estimate for ν_e and solve GROM with and without closure, we obtain the results in Fig. 6.3 for the temporal evolution the modal coefficients. For comparison, we also plot the true projection values of $\mathbf{a}$, defined as

$$a^k_i = \langle \mathbf{u}^k_{FOM} - \bar{\mathbf{u}}; \phi_i \rangle. \tag{6.51}$$

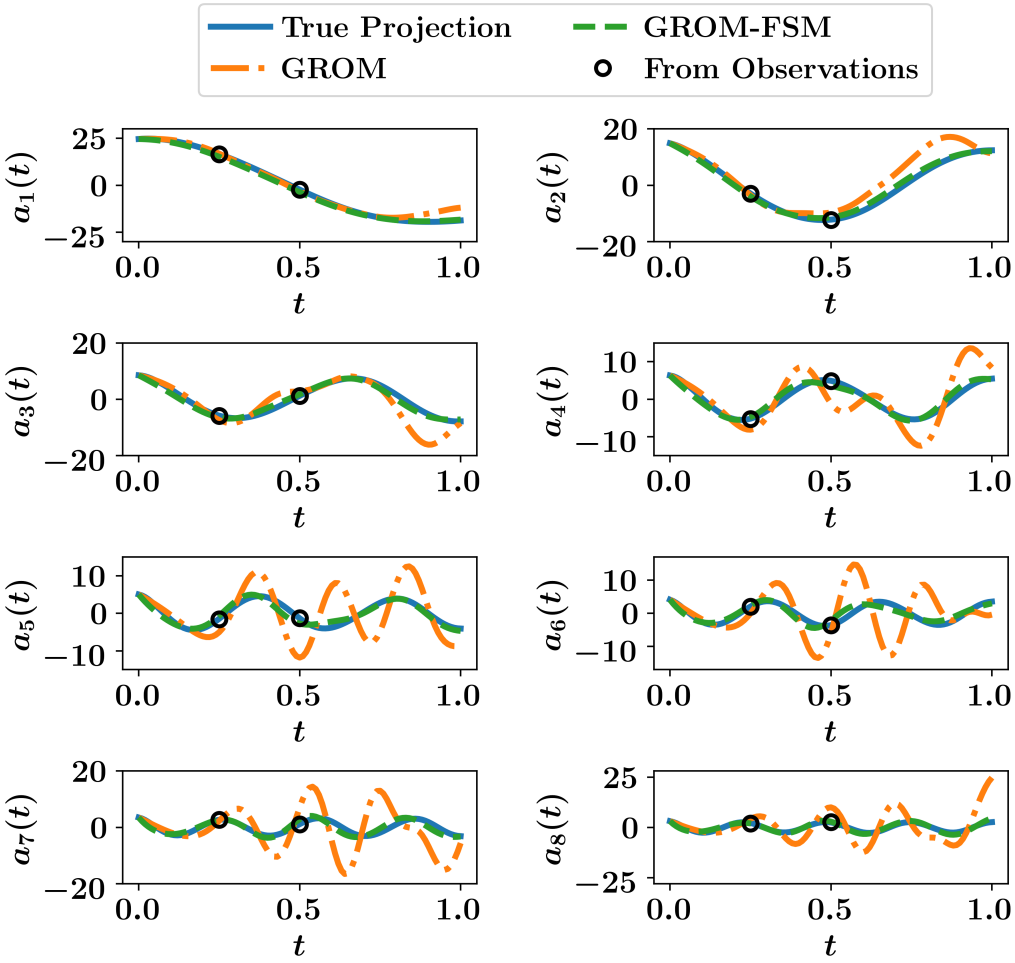


Figure 6.3: Temporal evolution of POD coefficients, assuming full field measurements are available.

Also, we sketch reconstructed velocity field at final time $t = 1$ in Fig. 6.4. It is clear that GROM without closure is unable to capture the true dynamical behavior of the described Burgers problem. On the other hand, GROM-FSM is shown to almost match the true projection. It is assumed that true projected values represent the best values that projection-based ROM can provide. For quantitative assessment, we also plot the root mean squares error (RMSE) of ROM predictions defined as

$$RMSE(t) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\Big(u_{FOM}(x_i,t) - u_{ROM}(x_i,t)\Big)^2} \tag{6.52}$$

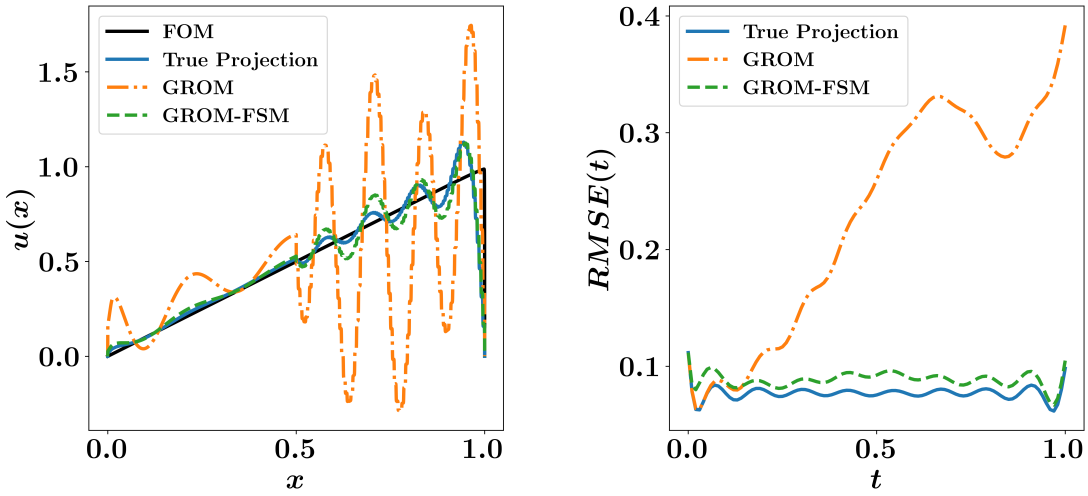


Figure 6.4: Velocity field reconstruction in case of full field measurements. Left: reconstruction of final velocity field using GROM and GROM with FSM eddy viscosity compared to the FOM and true projection fields. Right: RMSE of reconstructed fields at different time instants.

6.6.1.2 Sparse field measurement

Since full field measurements are usually inaccessible, we extend our study to consider sparse field measurements. In particular, we locate sensors at 8 points, equally spaced at $1/8, 2/8, 3/8, 4/8, 5/8, 6/8, 7/8, 8/8$ as shown in Fig. 6.5. To assimilate those measurements, we consider two cases. The first one is similar to the full field measurement case, where we preprocess those measurements to compute a least-squares approximation of the corresponding observed modal coefficients. In the second case, we keep our observation as field measurements and define an operator to map model state (i.e., modal coefficients) to observations (i.e., velocity).

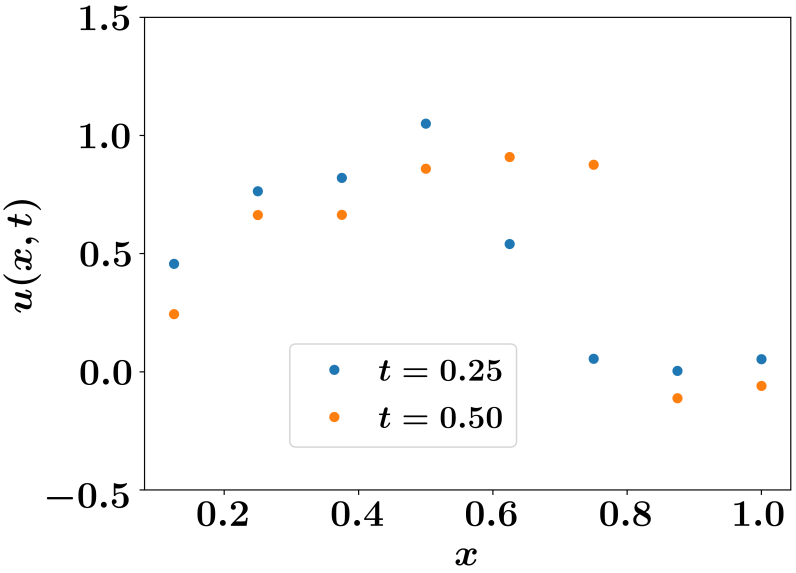


Figure 6.5: Noisy measurement of velocity fields at $t = 0.25$ s and $t = 0.50$ s, assuming sensors are located at 8 grid points.

From measurements to POD coefficients In order to preprocess the sparse measurements to approximate the observed modal coefficients, we sample Eq. (6.20) at the sensors locations as follows,

$$\begin{bmatrix} \phi_1(x_{O1}) & \phi_2(x_{O1}) & \dots & \phi_R(x_{O1}) \\ \phi_1(x_{O2}) & \phi_2(x_{O2}) & \dots & \phi_R(x_{O2}) \\ \vdots & & & \vdots \\ \phi_1(x_{O8}) & \phi_2(x_{O8}) & \dots & \phi_R(x_{O8}) \end{bmatrix} \begin{bmatrix} a^k_{1,Obs} \\ a^k_{2,Obs} \\ \vdots \\ a^k_{R,Obs} \end{bmatrix} = \begin{bmatrix} u^k_{Obs}(x_{O1}) - \bar{u}(x_{O1}) \\ u^k_{Obs}(x_{O2}) - \bar{u}(x_{O2}) \\ \vdots \\ u^k_{Obs}(x_{O8}) - \bar{u}(x_{O8}) \end{bmatrix}, \tag{6.53}$$

which can be generally solved using the pseudo-inverse. Then, the same observation operator and its Jacobian as defined in Section 6.6.1.1 are used. The temporal evolution of the modal coefficients are given in Fig. 6.6. Although the GROM-FSM results are better than GROM, they are significantly worse than those in Fig. 6.3. Of course, this is to be expected since we are using measurements at only 8 points, rather than 4096 locations. However, we also find that the observed modal coefficients calculations using Eq. (6.53) is greatly sensitive to the level of noise. Indeed, we find that least-squares computations sometimes do not converge (a remedy will be provided in Section 6.6.1.2). Moreover, we can see that the POD modal coefficients from observations are significantly different than the true ones.

The reconstructed field at final time as well as the $RMSE$ at different times are

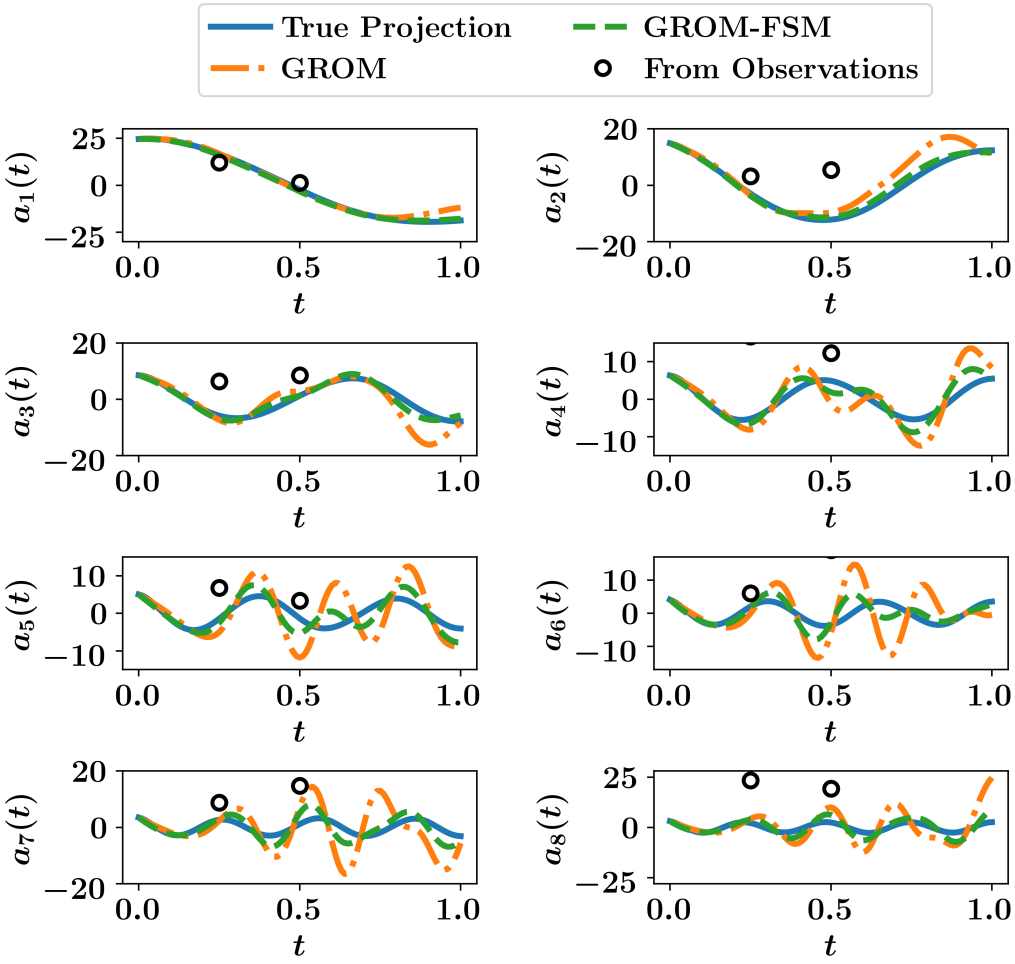


Figure 6.6: Temporal evolution of POD coefficients, where sparse field measurements are preprocessed to estimate the observed POD coefficients.

demonstrated in Fig. 6.7. We see that a small improvement is obtained in GROM-FSM, compared to GROM. We also note that for different noise levels, we get different performances for the GROM-FSM. This implies that this way of assimilating sparse observations is less reliable, and a more robust approach should be utilized. In Section 6.6.1.2, we discuss another way of using sparse observations to perform data assimilation for ROM closure.

From POD coefficients to measurements Now, we discuss defining an observational operator to construct a robust map between model space and measurement space. Similar to Section 6.6.1.2, we sample Eq. (6.20) at sensor location, but we introduce a map to reconstruct the velocity field at these locations using the model predicted coefficients. In other words, in Section 6.6.1.2, we use the sensors mea-

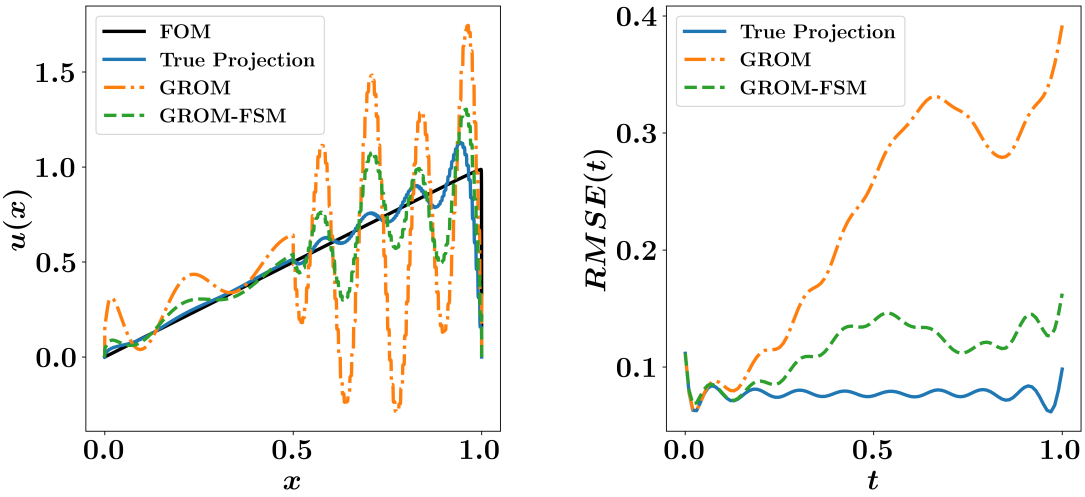


Figure 6.7: Velocity field reconstruction in case of preprocessing sparse field measurements to compute the observed POD coefficients. Left: reconstruction of final velocity field using GROM and GROM with FSM eddy viscosity compared to the FOM and true projection fields. Right: RMSE of reconstructed fields at different time instants.

surements to approximate a value for $\mathbf{a}^k_{Obs}$. But in this section, we use model predicted coefficients $\mathbf{a}^k$ to approximate the velocity field values at sensor locations (i.e., $u^k(x_{O1}), u^k(x_{O2}), \ldots, u^k(x_{O8})$) as follows,

$$\begin{bmatrix} \phi_1(x_{O1}) & \phi_2(x_{O1}) & \ldots & \phi_R(x_{O1}) \\ \phi_1(x_{O2}) & \phi_2(x_{O2}) & \ldots & \phi_R(x_{O2}) \\ \vdots & & & \vdots \\ \phi_1(x_{O8}) & \phi_2(x_{O8}) & \ldots & \phi_R(x_{O8}) \end{bmatrix} \begin{bmatrix} a^k_1 \\ a^k_2 \\ \vdots \\ a^k_R \end{bmatrix} = \begin{bmatrix} u^k(x_{O1}) - \bar{u}(x_{O1}) \\ u^k(x_{O2}) - \bar{u}(x_{O2}) \\ \vdots \\ u^k(x_{O8}) - \bar{u}(x_{O8}) \end{bmatrix}. \tag{6.54}$$

Thus, we define $\mathbf{z}^k = \mathbf{u}^k_{Obs}$, and the observation operator $\mathbf{h}(\mathbf{a}) = \mathbf{C}\mathbf{a}$, where $\mathbf{C}$ is the matrix of basis functions sampled at sensors locations as follows,

$$\mathbf{C} = \begin{bmatrix} \phi_1(x_{O1}) & \phi_2(x_{O1}) & \ldots & \phi_R(x_{O1}) \\ \phi_1(x_{O2}) & \phi_2(x_{O2}) & \ldots & \phi_R(x_{O2}) \\ \vdots & & & \vdots \\ \phi_1(x_{O8}) & \phi_2(x_{O8}) & \ldots & \phi_R(x_{O8}) \end{bmatrix}. \tag{6.55}$$

Thus, the Jacobian of $\mathbf{h}(\cdot)$ is defined as $\mathbf{D}_{\mathbf{a}}(\mathbf{h}) = \mathbf{C}$. We repeat the same GROM-FSM implementation with those redefined operators. Results are shown in Figs. 6.8 and 6.9, where we can see that this approach of assimilating measurements is more

robust than the one discussed in Section 6.6.1.2 with higher accuracy. We also note that similar performance is achieved using higher level of noise in measurements, while the approach in Section 6.6.1.2 requires very low level of observational noise.

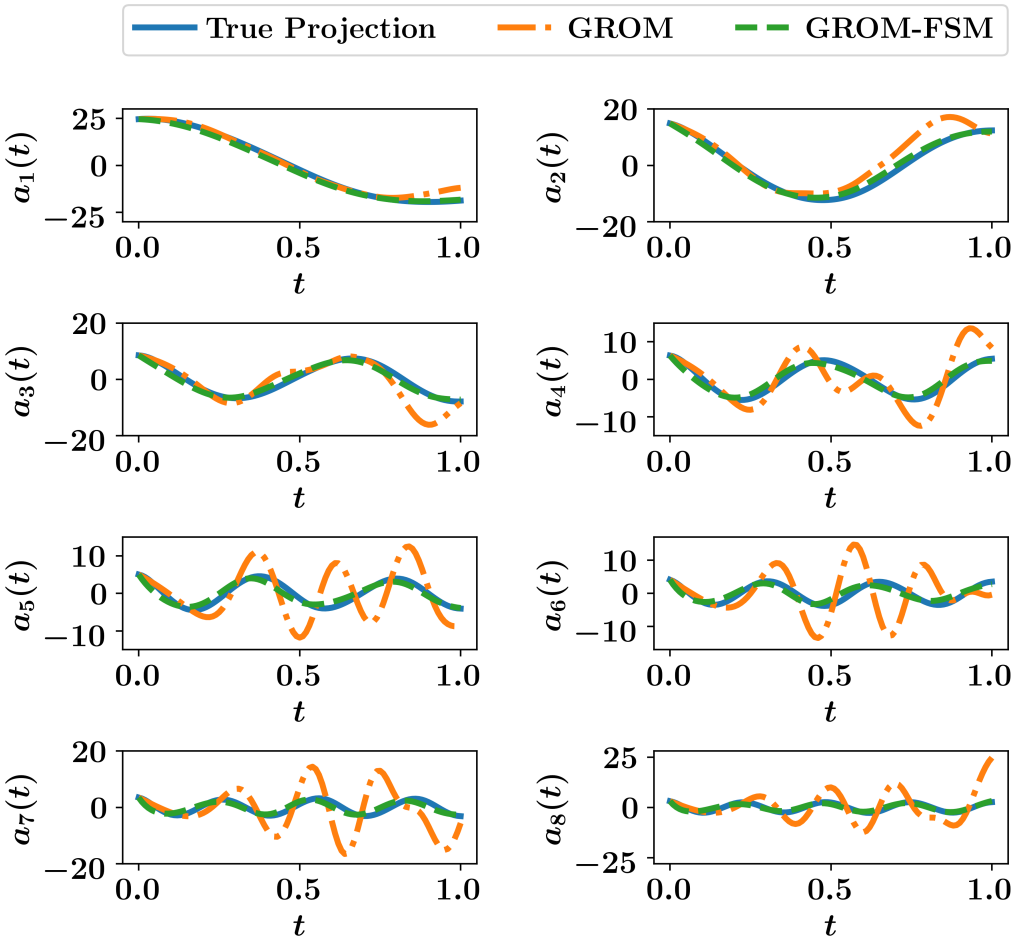


Figure 6.8: Temporal evolution of POD coefficients, where sparse field measurements are compared against POD field reconstruction using the observer operator $\mathbf{C}$.

Finally, for a big picture comparison, we plot the spatio-temporal evolution of reconstructed velocity fields for all discussed measurement setups compared to FOM and true projection fields in Fig. 6.10. From this figure, we notice that solution of GROM without closure is unstable, and brings non-physical predictions. On the other hand, predictions of GROM-FSM with full field measurements almost match the true projected fields. Also, assimilating sparse observations via the reconstruction map $\mathbf{C}$ is significantly superior to approximating observed coefficients using the pseudo-inverse approach. The latter shows some non-physical predictions, similar to GROM.

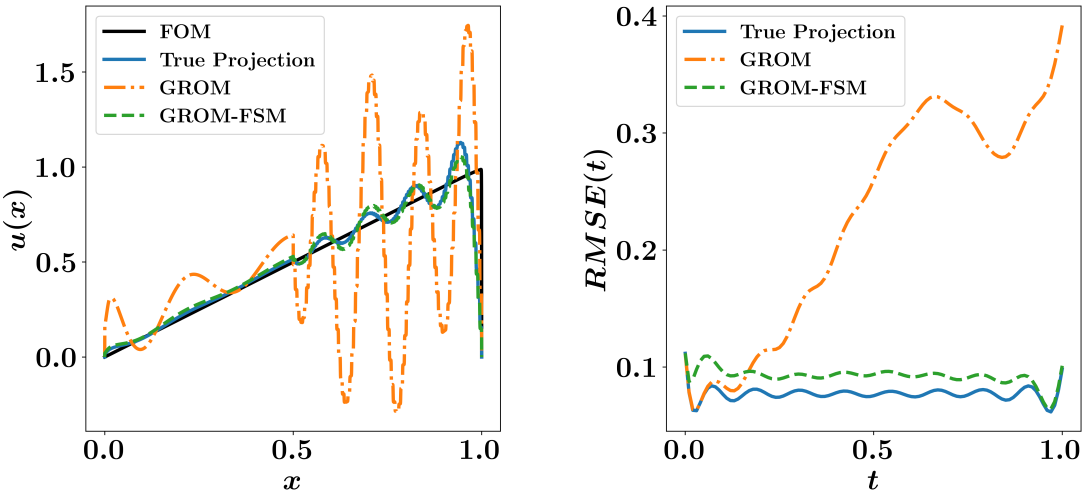


Figure 6.9: Velocity field reconstruction where sparse field measurements are compared against POD field reconstruction using the observer operator $\mathbf{C}$. Left: reconstruction of final velocity field using GROM and GROM with FSM eddy viscosity compared to the FOM and true projection fields. Right: RMSE of reconstructed fields at different time instants.

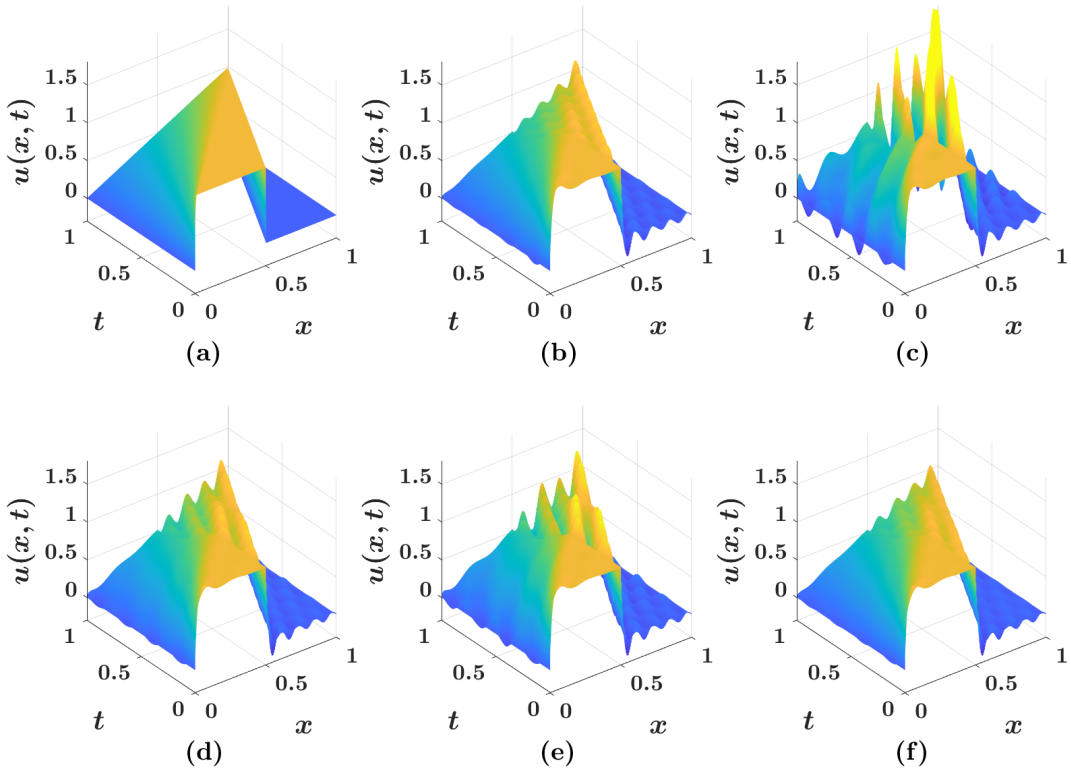


Figure 6.10: Surface plots for the temporal evolution of velocity fields from (a) FOM, (b) true projection, (c) GROM and FSM with (d) full and (e–f) sparse measurements configurations. Note: (e) shows the results using the method presented in Section 6.6.1.2, while (f) refers to the method presented in Section 6.6.1.2.

6.6.2 2D Kraichnan turbulence

For 2D turbulence, the inertial range in the energy spectrum is proportional to k^{-3} in the inviscid limit according to the Kraichnan–Batchelor–Leith (KBL) theory [309–311]. In our numerical experiments, the initial energy spectrum in Fourier space is given by

$$E(k) = \frac{4k^4}{3\sqrt{\pi}k_p^5} \exp\left[-\left(\frac{k}{k_p}\right)^2\right], \tag{6.56}$$

where $k = \sqrt{k_x^2 + k_y^2}$ and k_p is the wavenumber at which the maximum value of initial energy spectrum occurs. During the time evolution process, due to the nonlinear interactions, this spectrum quickly approaches toward k^{-3} spectrum. The magnitude of the vorticity Fourier coefficients is related to the energy spectrum as

$$|\tilde{\omega}(k)| = \sqrt{\frac{k}{\pi}E(k)}. \tag{6.57}$$

Thus, the initial vorticity distribution (in Fourier space) is obtained by introducing a random phase. For more details regarding derivation of the initial vorticity distribution from an assumed energy spectrum can be found in [312]. In the present study, we use a spatial computational domain of $(x, y) \in [0, 2\pi] \times [0, 2\pi]$ and a time domain of $t \in [0, 4]$. Periodic boundary conditions are applied in both x and y directions. A spatial grid of 512^2 and a timestep of $\Delta t = 0.001$ are used for FOM solution, and 800 snapshots of vorticity fields are stored for POD basis generation. A fourth-order accurate Arakawa scheme [119] is adopted for spatial discretization. Contours of voriticy field at different time instants are shown in Fig. 6.11 initiated by introducing a random phase shift in the Fourier space, with $k_p = 10$.

For ROM implementation, we consider $R = 16$ corresponding to a RIC value of around 80%. Similar to the 1D Burgers problem, we test the FSM capabilities to estimate an optimal value of eddy viscosity considering full and sparse field measurements with $\sigma_{Obs} = 0.1$. We assume a data assimilation window of 2, and measurement data are collected at $t = 1$ and $t = 2$, while testing is performed up to $t = 4$. We highlight here that all the 2D fields are rearranged into 1D column vectors to follow the same notations provided in Section 6.4 (e.g., the Euclidean inner product). However, for contour plots, they are reshaped back into 2D fields.

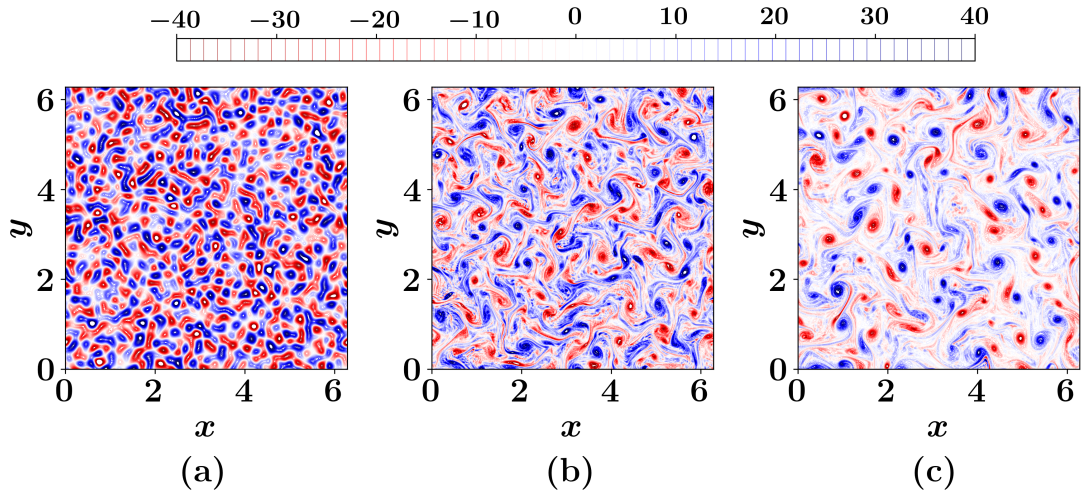


Figure 6.11: FOM results showing the time evolution of vorticity fields for the 2D turbulence problem: (a) $t = 0.0$, (b) $t = 2.0$, and (c) $t = 4.0$.

6.6.2.1 Full field measurement

Since full field measurements are available (though noisy), we can project these field data onto the basis functions ϕ to obtain the corresponding *observed* modal coefficients as

$$a^k_{i,Obs} = \langle \boldsymbol{\omega}^k_{Obs} - \bar{\boldsymbol{\omega}}; \phi_i \rangle. \tag{6.58}$$

Following the same procedure as in Section 6.6.1.1, an optimal value of eddy viscosity parameter is estimated with the FSM methodology. In Fig. 6.12, we show the GROM solution equipped by an eddy viscosity closure, computed by the proposed approach compared to the background solution of standard GROM without closure. Also, we plot the true projection (TP) results, where the true POD modal coefficients are obtained as

$$a^k_i = \langle \boldsymbol{\omega}^k_{FOM} - \bar{\boldsymbol{\omega}}; \phi_i \rangle. \tag{6.59}$$

From Fig. 6.12, we can observe an improvement in the prediction of modal coefficients incorporating the estimated eddy viscosity. Reconstructed vorticity fields are also provided in Fig. 6.13 along with the variation of root mean squares error with time. Although predictions are improved with the GROM-FSM implementation compared to the GROM solution, it is observed that this improvement is only significant at later times. Moreover, some modes (especially the first few) show better results than the others, see Fig. 6.12. We believe this is caused by the assumption of global (mode-independent) eddy viscosity contribution for all modes. Therefore,

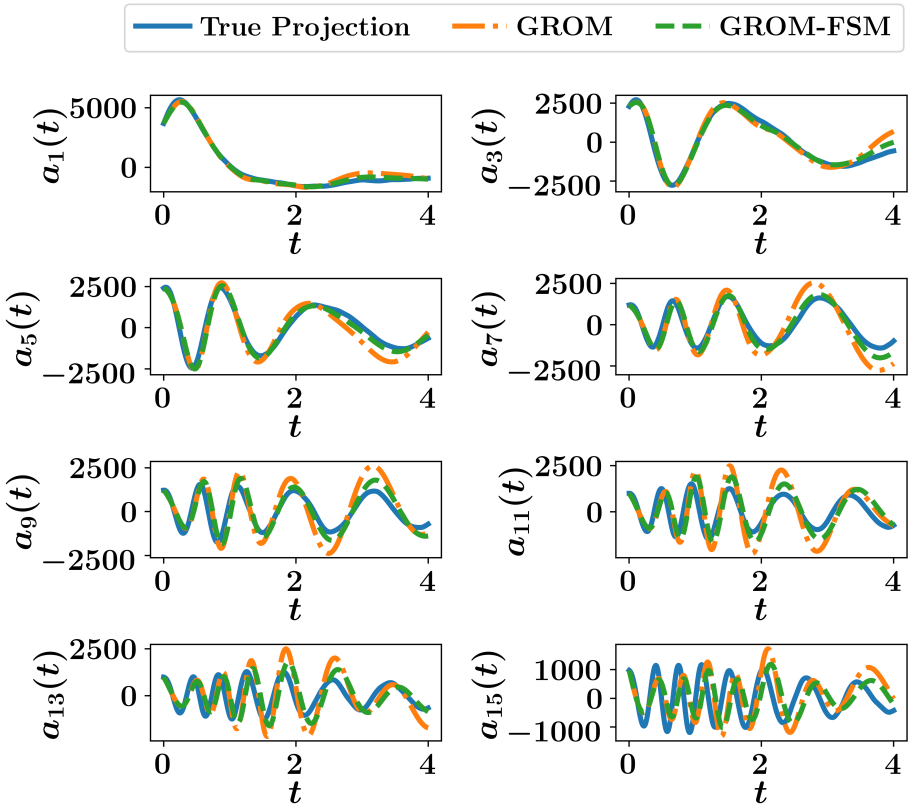


Figure 6.12: Time evolution of the modal coefficients for the 2D turbulence case, assuming full field measurements are available at $t = 1$ and $t = 2$.

in the optimization process involved in FSM, higher importance is given to those first few modes (since they possess the largest contribution). Consequently, a value of eddy viscosity that yields the best correction for those first modes is computed and applied for *all* modes. To mitigate this issue, we extend our closure estimation framework to allow mode-dependent variations of the eddy viscosity parameter.

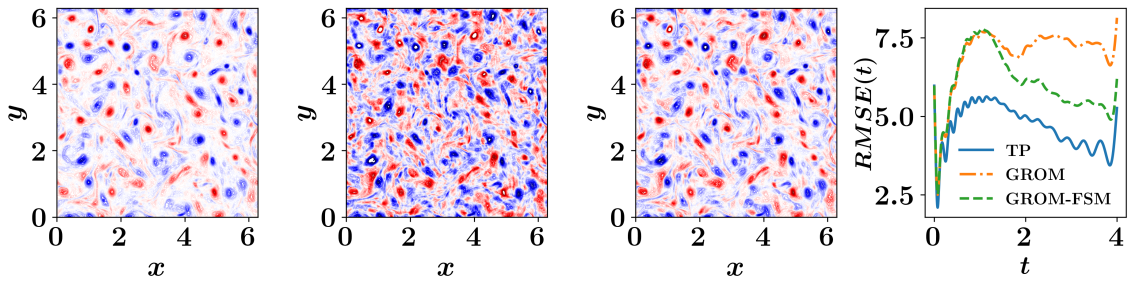


Figure 6.13: Reconstructed vorticity fields for the 2D turbulence problem at $t = 4$ for (a) true projection, (b) GROM, and (c) GROM-FSM along with the $RMSE$ variation with time (d), assuming full field measurements are available at $t = 1$ and $t = 2$.

Mode-dependent eddy viscosity. Instead of assuming a global eddy viscosity parameter as is the case in Eq. (6.41), we permit the variation of this parameter with modes as follows,

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \mathfrak{B}_k + \nu_{e,k}\widehat{\mathfrak{B}}_k + \sum_{i=1}^{R} \mathfrak{L}_{i,k} a_i + \nu_{e,k} \sum_{i=1}^{R} \widehat{\mathfrak{L}}_{i,k} a_i + \sum_{i=1}^{R}\sum_{j=1}^{R} \mathfrak{N}_{i,j,k} a_i a_j. \tag{6.60}$$

Thus, our goal now is to compute the values of $\nu_{e,k}$, where $k = 1, 2, \ldots, R$. In other words, we need to estimate R local values of eddy viscosity parameters, rather than a single global value. Indeed, this approach is also common in large eddy simulations, where a spatially varying eddy viscosity is considered. Figure 6.14 displays the time evolution of modal coefficients with a mode-dependent eddy viscosity computations. We can observe that almost equivalent improvements are obtained for all the modes, which highlights the superiority of this approach over the assumption of global eddy viscosity.

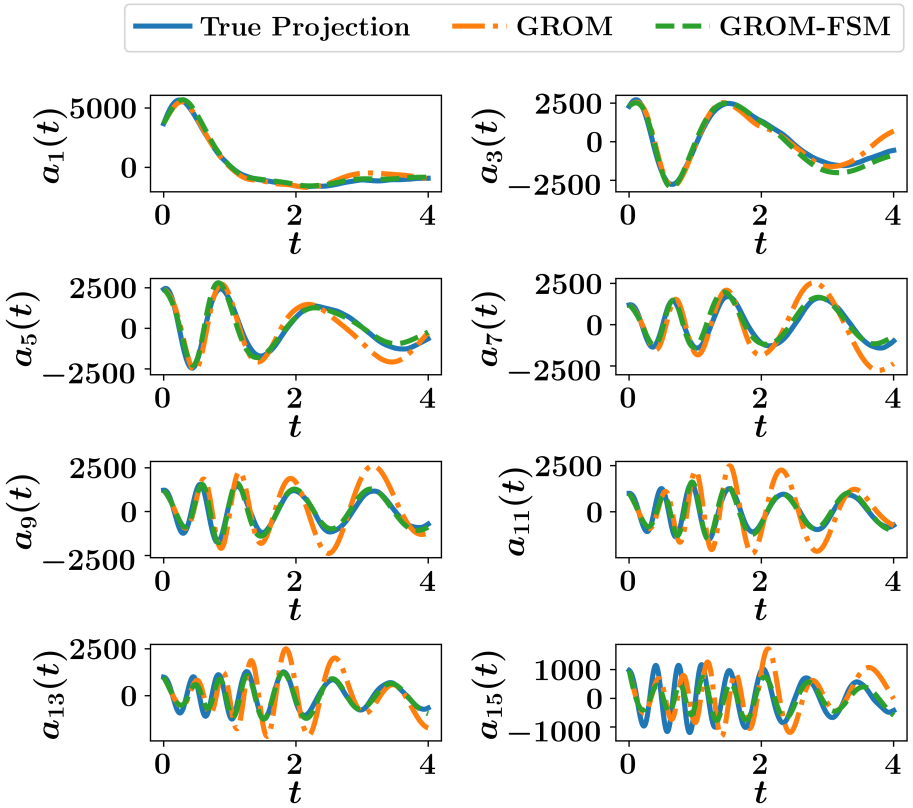


Figure 6.14: Time evolution of the modal coefficients for the 2D turbulence case, with full field measurements and mode-dependent eddy viscosity closure. Note the equal improvements in predictions for all the modes.

The quality of vorticity field reconstruction is also manifested in Fig. 6.15 with the contour plots at final time and variation of $RMSE$ with time. Interestingly, it can be noted that reductions of $RMSE$ are obtained at earlier times than those in Fig. 6.13. To understand this behavior, we investigate the GROM predictions without closure. we can see that the deviation of GROM predictions for the dynamics of the first few modes do not exhibit significant deviations during the assimilation window of $t = 2$. On the other hand, the latest modes show larger deviations during the same period. However, for a global eddy viscosity, the contribution of the first few modes (corresponding to the large convective scales) is predominant. As a result, a small value of eddy viscosity is computed to match the level of correction required for those large scales. Considering global eddy viscosity implementation, the latest modes (corresponding to small dissipating scales) receive minor corrections. On the other hand, a mode-dependent eddy viscosity implementation allows for detection of larger corrections required for disspative scales as seen in the modal coefficients predictions in Fig. 6.14 and $RMSE$ trend in Fig. 6.15.

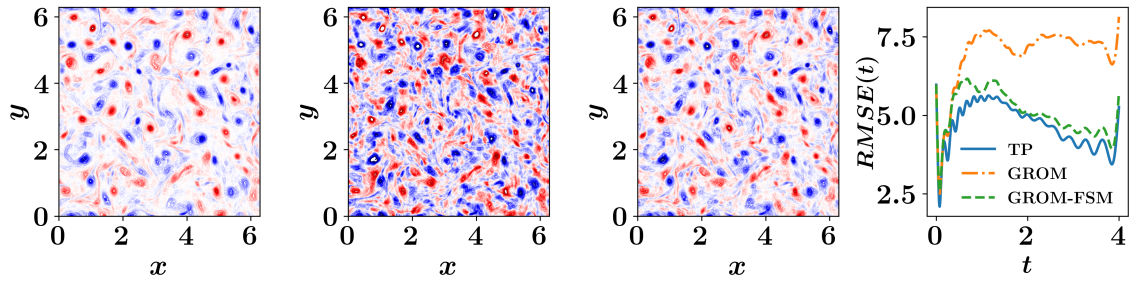


Figure 6.15: Reconstructed vorticity fields for the 2D turbulence problem at $t = 4$ for (a) true projection, (b) GROM, and (c) GROM-FSM along with the $RMSE$ variation with time (d), with full field measurements and mode-dependent eddy viscosity closure. Note the significant decrease in $RMSE$ at earlier times than those in Fig. 6.13.

6.6.2.2 Sparse field measurement

For measurement sparsity investigation, we assume a sensor located each 32 grid points. This corresponds to placing sensors at around 0.1% of the total spatial locations. Since it has already been shown that defining a mapping from the POD coefficients to the measured field variables provides a robust data assimilation framework, we follow the same procedure here. We also consider global and local eddy viscosity implementations.

Global eddy viscosity. With the mapping defined in Section 6.6.1.2, we apply the FSM eddy viscosity estimation framework with sparse data and global eddy viscosity. The time dynamics of the resolved modes is demonstrated in Fig. 6.16 for a few selected modal coefficients. We obtain similar results as those obtained with full field measurements, and we can observe that the predictions for the first few modes are much better than the remaining modes. Also, the reconstructed vorticity fields and computed $RMSE$ are shown in Fig. 6.17.

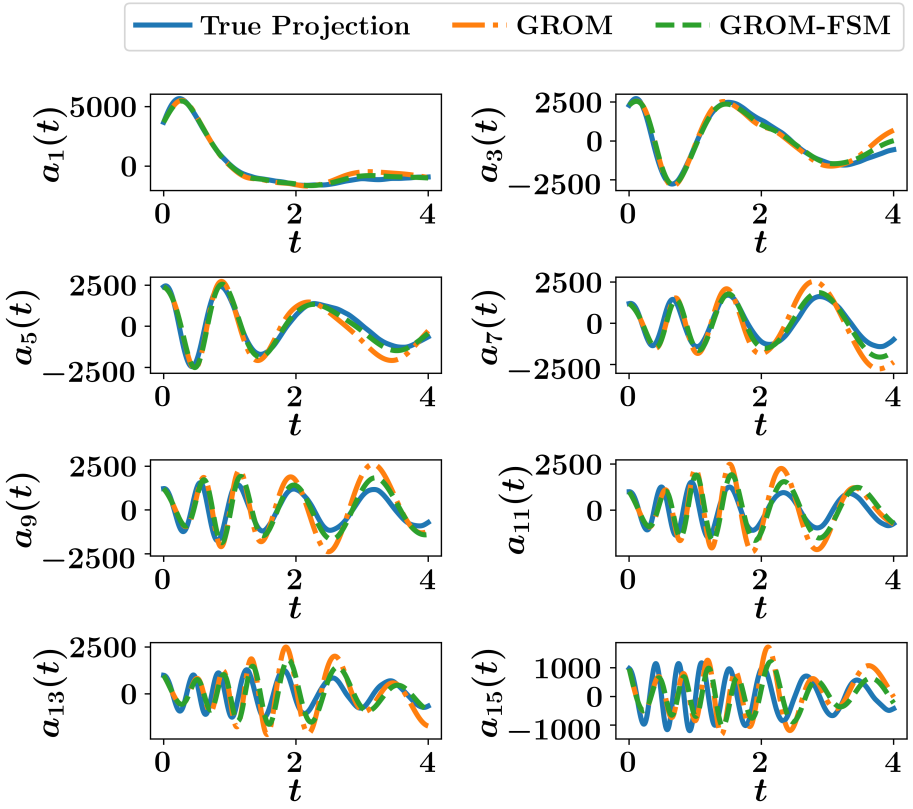


Figure 6.16: Time evolution of the modal coefficients for the 2D turbulence case, with sparse field measurements and global eddy viscosity closure. Note the better improvements in the first few modes compared to the latest ones.

Mode-dependent eddy viscosity. Allowing the variation of eddy viscosity yields notable enhancement of the prediction accuracy for all the modes as depicted in Fig. 6.18, compared to Fig. 6.16. Reconstructed vorticity fields at $t = 4$ with GROM, GROM-FSM and true projection results are plotted in Fig. 6.19. We also see the reduction of $RMSE$ even with the considered 0.1% sparsity in measurements data.

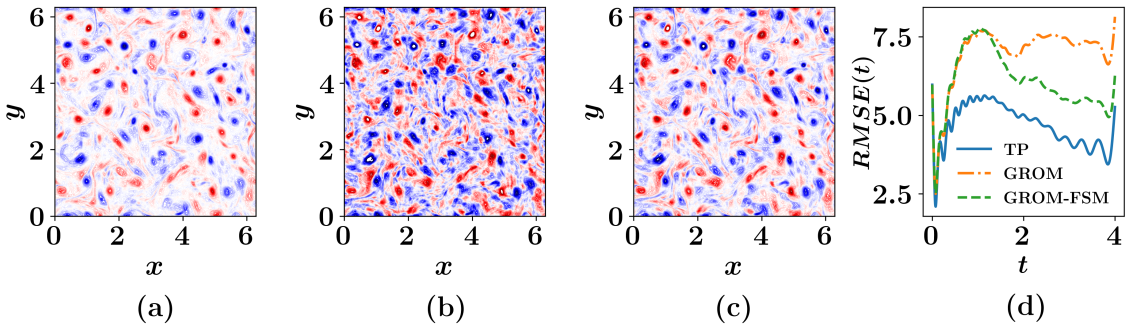


Figure 6.17: Reconstructed vorticity fields for the 2D turbulence problem at $t = 4$ for (a) true projection, (b) GROM, and (c) GROM-FSM along with the $RMSE$ variation with time (d), with sparse field measurements and global eddy viscosity closure. Reduction of $RMSE$ compared to GROM without closure starts to become remarkable around $t = 2$.

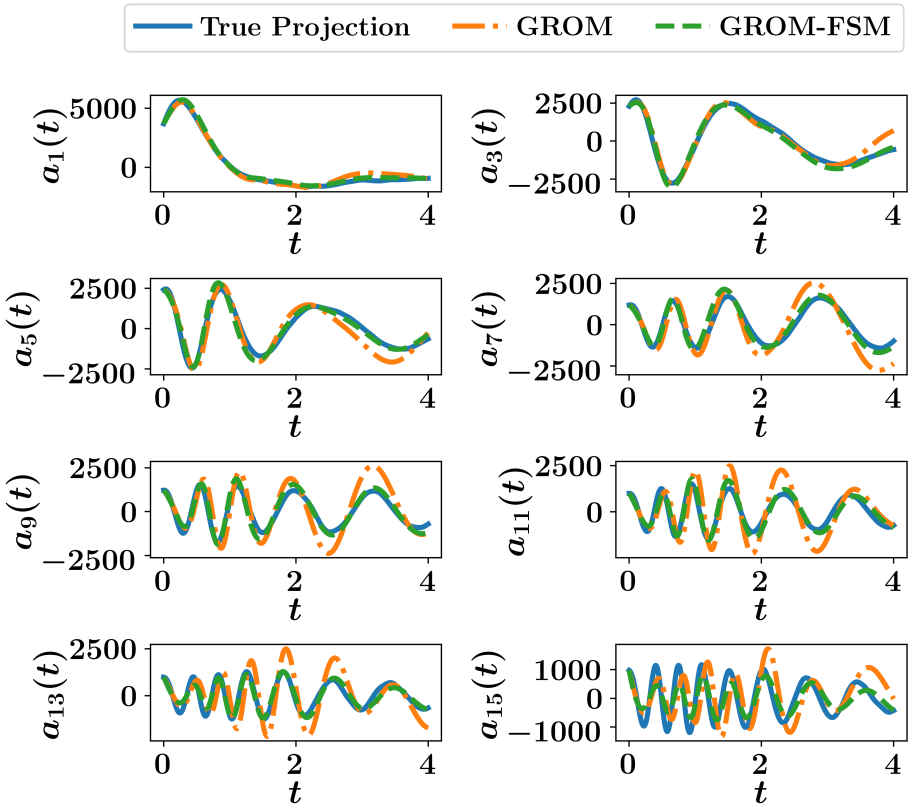


Figure 6.18: Time evolution of the modal coefficients for the 2D turbulence case, with sparse field measurements and mode-dependent eddy viscosity closure.

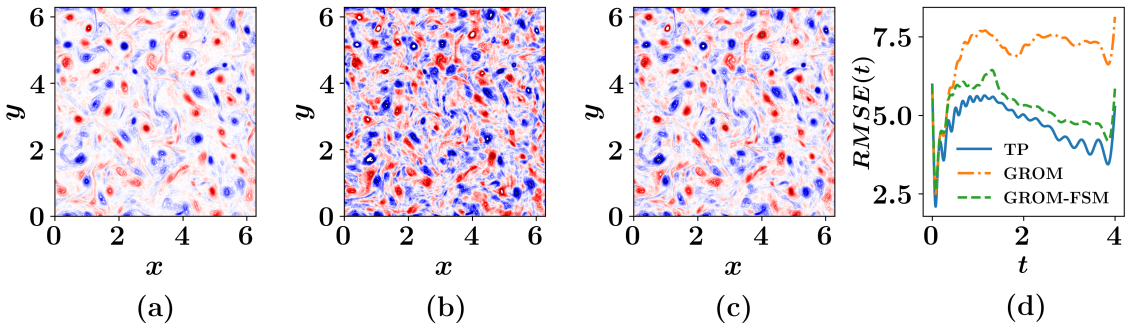


Figure 6.19: Reconstructed vorticity fields for the 2D turbulence problem at $t = 4$ for (a) true projection, (b) GROM, and (c) GROM-FSM along with the $RMSE$ variation with time (d), with sparse field measurements and mode-dependent eddy viscosity closure.

6.6.3 Computational cost

The forward sensitivity method avoids the solution of the adjoint problem usually encountered in variational approaches for assimilating observational data to improve model's predictions. Nonetheless, the main computational burden results from the recursive matrix-matrix multiplication as described in Eq. (6.6). However, since all computations are implemented in ROM space, the size of Jacobian matrices are $O(R)$, which reduces the memory and computational time requirements. Also, the deployed forward model in each iteration is the GROM model, which is computationally efficient when a few modes are retained in the ROM approximation. Moreover, our numerical experiments show that convergence occurs after a couple of iterations. We document the computational time for each iteration and the number of iterations for the explored test cases in Tables 6.1 and 6.2 using Python implementation. We highlight here that each iteration takes around twice the time of solving the GROM equations. For instance, for the 1D Burgers problem, the solution of the GROM equations takes about $0.176s$, while a single iteration of the FSM algorithm consumes less than $0.35s$. Similarly, for the 2D turbulence case, the CPU time to solve the GROM equations is almost $4.445s$, while a single iteration takes an order of $8s$. This is comparable to the computational time of variational approaches, where the forward and adjoint problems have to be solved in each iteration. Further reductions in FSM computing time might be achieved by efficient matrix-matrix multiplication algorithms.

From Table 6.2, we can also see that the CPU time per iteration for full field measurement case is slightly smaller than that in case of sparse data. We believe that

this is attributed to the difference in measurement space sizes in each case. For the full measurement case, we first project the data onto the basis functions to estimate the *observed* modal coefficients, and assume our measurements live in ROM space. So, the size of observational vector and forward sensitivty matrices are all $O(R)$. However, for the sparse case where we define a map from modal coefficient to field reconstruction, the observations are assimilated in FOM space. Therefore, the size of resulting vectors and matrices, corresponding to measurements, depends on the number of sensor data. For the 2D case, this number is relatively larger than R, and thus the resulting matrix computations become slightly more expensive. On the other hand, for the 1D Burgers problem in Table 6.1, the CPU for either the full or sparse measurements is similar since we are using 8 sensors for the sparse case. This is the same as the number of modes in the ROM approximation. We also notice in Table 6.2 that the CPU time for mode-dependent (i.e., local) eddy viscosity estimation is larger than that in case of fixed scalar eddy viscosity approximation. This is caused by the larger sizes of the model Jacobians with respect to its parameters (see Eq. (6.4)) as well as the solution of a bigger weighted least-squares problem (e.g., Eq. (6.47)).

Table 6.1: The CPU time (in seconds) per iteration and number of iterations required for eddy viscosity estimation using the proposed FSM-based methodology for the 1D Burgers problem. Sparse 1 refers to the implementation of mapping from measurement to POD coefficients (Section 6.6.1.2), while Sparse 2 refers to the mapping from POD coefficients to measurement (Section 6.6.1.2).

Measurement	CPU time (s)	No. of iterations
Full	0.342	7
Sparse 1	0.331	9
Sparse 2	0.344	8

Table 6.2: The CPU time (in seconds) per iteration and number of iterations required for eddy viscosity estimation using the proposed FSM-based methodology for the 2D turbulence problem. Here, [global] refers to the estimation of a global eddy viscosity parameter for all modes, while [local] refers to the estimation of mode-dependent eddy viscosities.

Measurement	CPU time (s)	No. of iterations
Full [global]	7.766	5
Full [local]	9.482	4
Sparse [global]	7.898	4
Sparse [local]	9.532	4

6.7 Conclusions

In the present study, we propose a data assimilation-based approach to provide accurate ROMs for digital twin applications. In particular, we use the forward sensitivity method (FSM) to estimate as well as update an optimal value of eddy viscosity for ROM closure. We exploit ongoing streams of observational data to improve the stability and accuracy of ROM predictions. We test the framework with the prototypical one-dimensional viscous Burgers equation characterized by strong nonlinearity and the two-dimensional vorticity transport equation for the 2D Kraichnan turbulence problem. We investigate the assimilation of full field and sparse field measurements. For full field measurements, we illustrate that projecting those noisy measurements produces good estimate of *observed* modal coefficients, which can therefore used to estimate an optimal value for eddy viscosity. However, we find that a similar approach of using sparse field measurements to approximate the observed states is significantly sensitive to measurements noise.

On the other hand, we demonstrate that defining an observational operator via a ROM reconstruction map can be successful in utilizing sparse and noisy data. Using real-time observations can steer ROM parameters and predictions to reflect actual flow conditions. We also remark that the collected snapshots of full order model solutions can be assimilated by treating them as full field measurements, with negligible noise (corresponding to discretization and numerical approximation errors). This should provide a prior estimate for the eddy viscosity parameterization during an offline stage. We emphasize that fusing ideas between physics-based closures (e.g., the ansatz for eddy viscosity) and model reduction with variational data assimilation techniques can provide valuable tools to construct reliable ROMs for long-time as well off-design predictions. This should leverage ROM implementation for real-life application.

Appendix A: Computing Model Jacobians in Discrete-Time Formulation

We describe the computation of the model Jacobian (with respect to the sate $\mathbf{a}$ and the parameter ν_e) in discrete-time formulations. while we show the derivations for the case with a fixed global eddy viscosity parameter, extension to mode-dependent closure estimation is straightforward. For temporal discretization of the GROM equations, we use fourth-order Runge-Kutta (RK4) method as follows,

$$\mathbf{a}^{k+1} = \mathbf{a}^k + \frac{\Delta t}{6}(\mathbf{g}_1 + 2\mathbf{g}_2 + 2\mathbf{g}_3 + \mathbf{g}_4), \tag{6.61}$$

where:

$$\begin{aligned} \mathbf{g}_1 &= \mathbf{f}(\mathbf{a}^k, \nu_e), \\ \mathbf{g}_2 &= \mathbf{f}(\mathbf{a}^k + \frac{\Delta t}{2} \cdot \mathbf{g}_1, \nu_e), \\ \mathbf{g}_3 &= \mathbf{f}(\mathbf{a}^k + \frac{\Delta t}{2} \cdot \mathbf{g}_2, \nu_e), \\ \mathbf{g}_4 &= \mathbf{f}(\mathbf{a}^k + \Delta t \cdot \mathbf{g}_3, \nu_e). \end{aligned} \tag{6.62}$$

Thus the discrete-time map defining the transition from time t_k to time t_{k+1} is written as

$$\mathbf{M}(\mathbf{a}^k, \nu_e) = \mathbf{a}^n + \frac{\Delta t}{6}(\mathbf{g}_1 + 2\mathbf{g}_2 + 2\mathbf{g}_3 + \mathbf{g}_4). \tag{6.63}$$

Then, the 'total' Jacobian of $\mathbf{M}$ is an $R \times (R+1)$ matrix, computed as

$$\mathbf{D}^k(\mathbf{M}) = [\mathbf{D}^k_{\mathbf{a}}(\mathbf{M}),\ \mathbf{D}^k_{\nu_e}(\mathbf{M})] \tag{6.64}$$

$$= \mathbf{P} + \frac{\Delta t}{6}\Big(\mathbf{D}\mathbf{g}_1 + 2\mathbf{D}\mathbf{g}_2 + 2\mathbf{D}\mathbf{g}_3 + \mathbf{D}\mathbf{g}_4\Big), \tag{6.65}$$

where $\mathbf{P} = \Big[\mathbf{I}_R,\ \ \mathbf{0}_{R\times 1}\Big] \in \mathbb{R}^{R\times(R+1)}$. The Jacobian of the model $\mathbf{M}$ with respect to the model state $\mathbf{a}^k$ is the first R columns of $\mathbf{D}(\mathbf{M})$, while the Jacobian of $\mathbf{M}$ with respect to the the eddy viscosity parameter ν_e is the last column of $\mathbf{D}(\mathbf{M})$.

Here, $\mathbf{Dg}_1$, $\mathbf{Dg}_2$, $\mathbf{Dg}_3$, and $\mathbf{Dg}_4$ are evaluated using the chain rule as follows,

$$\begin{aligned}
\mathbf{Dg}_1 &= \mathbf{Df}(\mathbf{a}^k, \nu_e), \\
\mathbf{Dg}_2 &= \left(\mathbf{Df}(\mathbf{a}^k + \frac{\Delta t}{2} \cdot \mathbf{g}_1, \nu_e)\right)\left(\mathbf{I}_{(R+1)} + \frac{\Delta t}{2}\begin{bmatrix}\mathbf{Dg}_1 \\ \mathbf{Q}\end{bmatrix}\right), \\
\mathbf{Dg}_3 &= \left(\mathbf{Df}(\mathbf{a}^k + \frac{\Delta t}{2} \cdot \mathbf{g}_2, \nu_e)\right)\left(\mathbf{I}_{(R+1)} + \frac{\Delta t}{2}\begin{bmatrix}\mathbf{Dg}_2 \\ \mathbf{Q}\end{bmatrix}\right), \\
\mathbf{Dg}_4 &= \left(\mathbf{Df}(\mathbf{a}^k + \Delta t \cdot \mathbf{g}_3, \nu_e)\right)\left(\mathbf{I}_{(R+1)} + \Delta t\begin{bmatrix}\mathbf{Dg}_3 \\ \mathbf{Q}\end{bmatrix}\right),
\end{aligned} \tag{6.66}$$

where $\mathbf{Q} = \mathbf{0}_{1\times(R+1)}$. Finally, the Jacobian of $\mathbf{Df}(\mathbf{a}^k, \nu_e)$ is defined as $\mathbf{Df}(\mathbf{a}^k, \nu_e) = [\mathbf{D_a f}(\mathbf{a}^k, \nu_e),\ \mathbf{D}_{\nu_e}\mathbf{f}(\mathbf{a}^k, \nu_e)]$, where

$$\frac{\partial f_k}{\partial a_j} = (\nu + \nu_e)\mathfrak{L}_{j,k} + \sum_{i=1}^{R}\mathfrak{N}_{i,j,k}a_i + \sum_{i=1}^{R}\mathfrak{N}_{j,i,k}a_i, \tag{6.67}$$

$$\frac{\partial f_k}{\partial \nu_e} = \sum_{i=1}^{R}\mathfrak{L}_{i,k}a_i. \tag{6.68}$$

Appendix B: Extension to Generalized Latent Control Formulation

The closure problem can be approached by considering as a control input in the latent space (i.e., latent control or latent action) that counteracts the induced instabilities and inaccuracies from the GROM truncation. We employ a continuous time control signal to correct and stabilize the GROM trajectory by deriving low-rank closure models with linear damping and dissipation terms using principles from the Kolmogorov energy cascade of turbulence and energy conservation. We utilize the forecast error, measured as the discrepancy between GROM predictions and collected sensor data, as the feedback and develop a variational approach to update the control input. Finally, we extend the FSM to derive first-order estimates of the relationships between the feedback and the desired control parameters.

The GROM for the 1D Burgers equation can be written in a compact form as follows:

$$\dot{\mathbf{a}} = \mathbf{b} + \mathbf{L}\mathbf{a} + \mathbf{a}^T\mathbf{N}\mathbf{a}, \tag{6.69}$$

where $\mathbf{b} \in \mathbb{R}^R$, $\mathbf{L} \in \mathbb{R}^{R\times R}$, and $\mathbf{N} \in \mathbb{R}^{R\times R\times R}$ respectively represent the constant, linear, and nonlinear terms that result from the inner product between the FOM operators and the POD basis functions. Then, we modify Eq. (6.69) by adding a control input $\mathbf{c}(t) = [c_1, c_2, \ldots, c_R]^T$ as follows:

$$\dot{\mathbf{a}} = \mathbf{b} + \mathbf{L}\mathbf{a} + \mathbf{a}^T\mathbf{N}\mathbf{a} + \mathbf{c}(t). \tag{6.70}$$

The goal of the control $\mathbf{c}$ is to steer the GROM predictions toward the target trajectory defined from the projection of FOM solution on POD subspace. The control input that would result in values of $\mathbf{a}$ that are exactly equal to their optimal values can be defined as follows:

$$c_k(t) = \langle \mathcal{F}(u;\mu); \phi_k \rangle - \langle \mathcal{F}(\bar{u} + \sum_{i=1}^{R} a_i\phi_i; \mu); \phi_k \rangle. \tag{6.71}$$

We highlight that Eq. (6.71) is not useful in practice because it requires solving the FOM to compute $\boldsymbol{u}$ at each time step. Therefore, alternative approximate models

This discussion is adapted from: *Ahmed, S. E., & San, O. (2022). Forward sensitivity analysis and mode dependent control for closure modeling of Galerkin systems. International Journal of Adaptive Control and Signal Processing (under review).*

are sought to estimate $\mathbf{c}$ as a function of the available information in the ROM subspace (i.e., $\{a_k, \phi_k\}_{k=1}^R$). The present study draws concepts from the Kolmogorov energy cascade and energy conservation principles to define the effect of the modal truncation on ROM dynamics. In order to derive the form of the closure model, we add a combination of linear friction and diffusion terms to Eq. (6.28) as follows:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} + \gamma u + \beta\frac{\partial^2 u}{\partial x^2}, \tag{6.72}$$

where γ and β are the friction and diffusion parameters, respectively. Projecting Eq. (6.72) onto the POD subspace leads to a model for $\mathbf{c}$ as follows:

$$\mathbf{c} = \gamma\mathbf{e} + \gamma\mathbf{a} + \beta\mathbf{q} + \beta\boldsymbol{D}\mathbf{a} \tag{6.73}$$

$$\text{where:} \qquad [\mathbf{e}]_k = \langle\bar{u}; \phi_k\rangle, \qquad [\mathbf{q}]_k = \langle\frac{\partial^2\bar{u}}{\partial x^2}; \phi_k\rangle, \qquad [\mathbf{D}]_{k,i} = \langle\frac{\partial^2\phi_i}{\partial x^2}; \phi_k\rangle. \tag{6.74}$$

Thus, correcting the GROM trajectory amounts for estimating the parameters γ and β and we refer to them as the *control parameters* or simply the *control*. We note that when $R = N$, all the modes are retained and the FOM solution is obtained. In this case, $\boldsymbol{c} = 0$ and $\gamma = \beta = 0$, and thus Eq. (6.28) is recovered. Finally, we set $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_R]^T \in \mathbb{R}^n$ and $\beta = [\beta_1, \beta_2, \ldots, \beta_R]^T \in \mathbb{R}^n$ to allow variability of the closure model with different modes. The use of mode-dependent correction has been shown to provide better closure models, e.g., by matching energy levels between FOM and ROM [313], incorporating spectral kernels [48, 49], or utilizing the variational multi-scale framework [59, 156, 314].

Results We use the same setup for the FOM as in Section 6.6 and retain $n = 6$ modes for in the ROM computations. An eigenvalue analysis reveals that 6 modes capture about 92% of the total system turbulent kinetic energy.

- Full field observations: First, we assess the feasibility of adopting the FSM methodology to control the ROM trajectory by considering a direct measurement of the full spatial flow field. We assume that the sensor signal is contaminated by a white Gaussian noise with zero mean and standard deviation of 0.1, which represents 10% of the peak velocity. We collect measurement after every 10 time integrations of the GROM (i.e., $\Delta t_{Obs} = 0.1$). We refer to the solution with the target trajectory (given by Eq. (6.51)) as "True Closure" while the solution of the *uncontrolled* GROM (i.e., Eq. (6.69)) is denoted as the "No Clo-

sure" solution. Finally, the solution of the *controlled* GROM (i.e., Eq. (6.70)) with FSM used to parameterize the presumed closure model in Eq. (6.73) is labeled as "FSM Closure".

Figure 6.20 depicts the predicted dynamics in the latent ROM space using the considered different approaches. We observe that GROM leads to inaccuracies and significantly amplifies the magnitude of predicted coefficients, especially for the last mode. This behavior is likely to cause long term instabilities in the solution even if the actual system is stable. On the other hand, the FSM effectively controls the GROM trajectory and keeps it closer to the target trajectory. We emphasize that we implement a mode-dependent control to respect the distinct characteristics of the resolved modes defining recurrent flow structures.

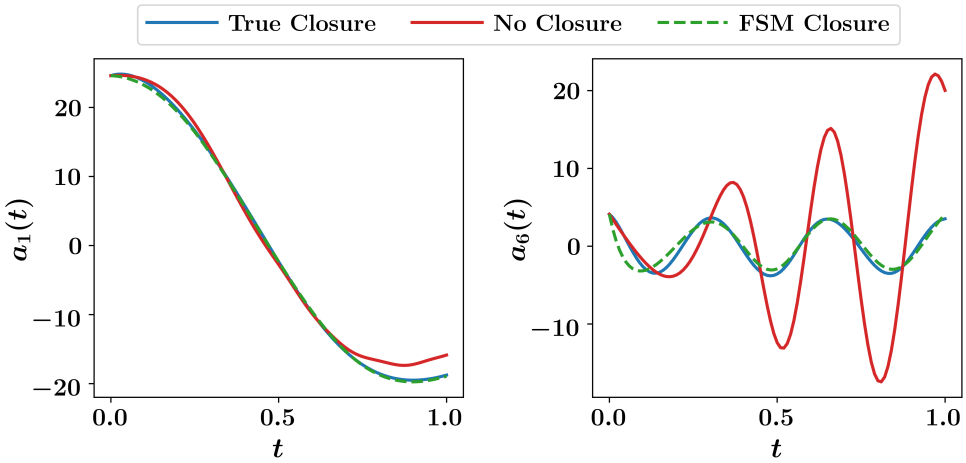


Figure 6.20: Dynamics of the first and last modal coefficients with full field measurement for the FSM Closure.

In Fig. 6.21, we evaluate the performance in the physical space by computing the reconstructed flow field using Eq. (6.29) compared to the FOM fields. We see that results from FSM Closure are very similar to the True Closure which represents the minimum reconstruction error that could be obtained using 6 modes. On the other hand, vanilla-type GROM without closure yields inaccurate and even non-physical solution in the spatio-temporal space.

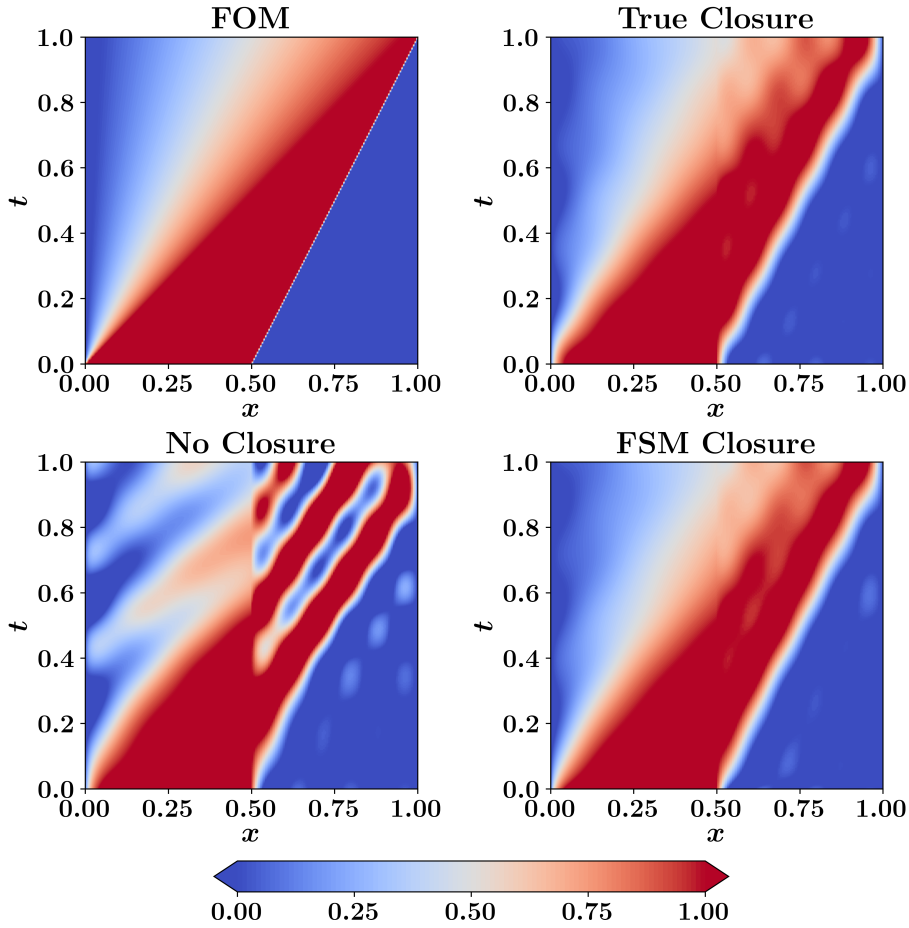


Figure 6.21: Spatio-temporal field predictions of Burgers problem using FOM and GROM approaches. Full field measurements are considered for the FSM Closure.

- Sparse field observations: We extend our numerical experiments to explore incomplete field measurement scenarios. In particular, we consider a sparse signal $\mathbf{s} \in \mathbb{R}^m$ of the flow field $\mathbf{u}$ as follows:

$$\mathbf{s} = \boldsymbol{\Theta}\mathbf{u}, \tag{6.75}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{m \times n}$ is a sampling matrix, constructed by taking m rows of the $n \times n$ identity matrix (i.e., $[\boldsymbol{\Theta}]_{ij} = 1$ if the i^{th} sensor is located at the j^{th} location and $[\boldsymbol{\Theta}]_{ij} = 0$, otherwise). Sensors can be placed at equally-spaced locations, random locations, or carefully selected places. Optimal observation placement is an active field of research, also known as optimal experimental design (OED). We utilize a greedy compressed sensing algorithm based on QR decomposition with column pivoting to set-up a near-optimal sensor placement strategy as

follows:

$$\mathbf{\Psi}^T\mathbf{P} := \mathbf{QR}, \tag{6.76}$$

where $\mathbf{\Psi} = [\psi_1, \psi_2, \ldots, \psi_m] \in \mathbb{R}^{n\times m}$ includes the first m POD basis functions for $\mathbf{u}$, and $\mathbf{P} \in \mathbb{R}^{n\times n}$ is the permutation matrix. Manohar et al. [315] showed that by using the first m rows of $\mathbf{P}$ to define the sampling matrix $\mathbf{\Theta}$, a near optimal sensor placement is obtained with similarities to the A- and D-optimality criteria in OED studies.

Figure 6.22 displays the time evolution of the first and sixth modal coefficients with the adopted FSM closure methodology in the case of sparse measurements. In particular, we selected 25 locations (about 0.5% of the total number of grid points) using the described QR-based algorithm to define the sensors data. We see that FSM closure yields very accurate results that are close to the the target trajectory even with the sparse measurement data. The reconstruction accuracy is also demonstrated using Fig. 6.23, showing significant improvements compared the GROM predictions without control.

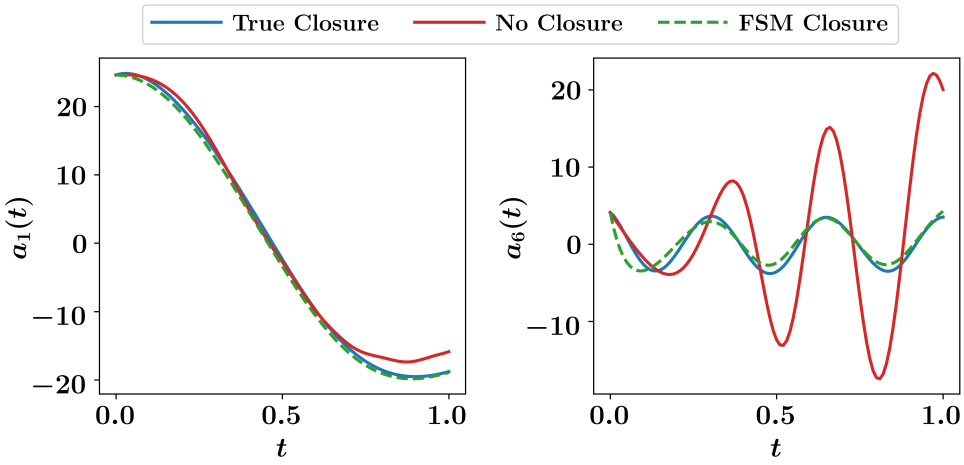


Figure 6.22: Dynamics of the first and last modal coefficients with sparse field measurement for the FSM Closure.

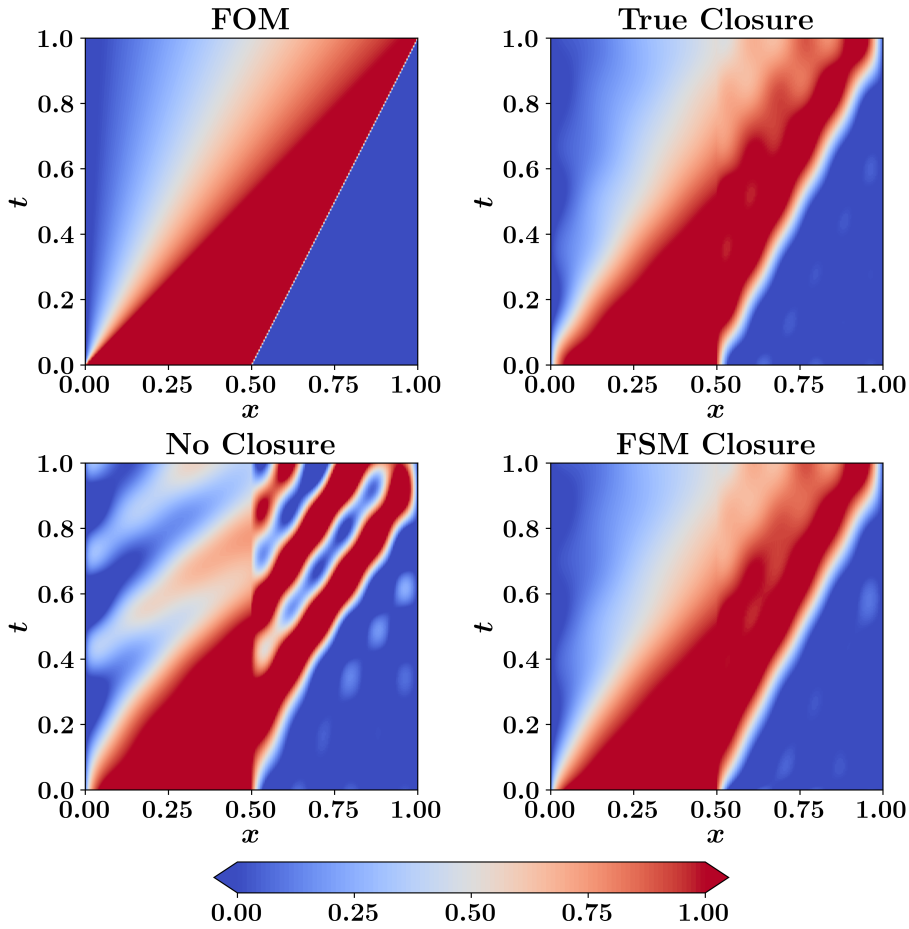


Figure 6.23: Spatio-temporal field predictions of Burgers problem using FOM and GROM approaches. Sparse field measurements are considered for the FSM Closure.

CHAPTER 7

A Nudged Hybrid Analysis and Modeling Approach for Realtime Wake-Vortex Transport and Decay Prediction

7.1 Abstract

We put forth a long short-term memory (LSTM) nudging framework for the enhancement of reduced order models (ROMs) of fluid flows utilizing noisy measurements for air traffic improvements. Toward emerging applications of digital twins in aviation, the proposed approach allows for constructing a realtime predictive tool for wake-vortex transport and decay systems. We build on the fact that in realistic application, there are uncertainties in initial and boundary conditions, model parameters, as well as measurements. Moreover, conventional nonlinear ROMs based on Galerkin projection (GROMs) suffer from imperfection and solution instabilities, especially for advection-dominated flows with slow decay in the Kolmogorov n-width. In the presented LSTM nudging (LSTM-N) approach, we fuse forecasts from a combination of imperfect GROM and uncertain state estimates, with sparse Eulerian sensor measurements to provide more reliable predictions in a dynamical data assimilation framework. We illustrate our concept by solving the two-dimensional vorticity transport equation. We investigate the effects of measurements noise and state estimate uncertainty on the performance of the LSTM-N behavior. We also demonstrate that it can sufficiently handle different levels of temporal and spatial measurement sparsity, and offer a huge potential in developing next-generation digital twin technologies for aerospace applications.

7.2 Introduction

Aircraft wings are optimized to produce maximum lift and minimum drag. Their design ensures that there is a high-pressure zone below the wing and a low-pressure

This chapter is adapted from: *Ahmed, S. E., Pawar, S., San, O., Rasheed, A., & Tabib, M. (2021). A nudged hybrid analysis and modeling approach for realtime wake-vortex transport and decay prediction. Computers & Fluids, 221, 104895.*

zone above. Owing to this pressure gradient, air from below the wing is drawn around the wingtip into the region above the wing causing a vortex to trail from each wing tip. These wake vortices (WVs) are stable under calm atmospheric conditions and remain present in the free atmosphere for a very long time, retaining its shape and energy [316–318]. Furthermore, at very low altitudes, they might rebound from the ground and linger on in the flight path corridor, posing significant risks to other aircraft that might encounter them. This is, in particular, crucial for large jetliners as WVs can cause violent rolling motions and even flip a small aircraft upside down when a pilot trailing a large aircraft flies into such vortices [319].

It is due to the hazards posed by WVs left behind by a taking off or landing aircraft that serious precautions are to be taken. The operational minimum aircraft separation for different weight class configurations, used by the Air Traffic Control (ATC), varies from 2.5 to 6 nautical miles. However, when deciding the separation distance following those guidelines, the weather conditions and associated transport and decay of WVs are not often taken into account. This was not a serious issue a couple of decades ago, but with the significant increase of the air traffic and a push for remote towers for cost effective and safe operation, major airports around the world are feeling the pressure. In this regard, Digital Twins of airports appear like a potential solution. In the current context, there is a need to develop a more efficient wake turbulence separation consisting of time-based minima between different aircraft types which takes into account the dynamic meteorological factors along with the variation in the wake generation mechanism associated with different classes of aircraft. Such information will enable air traffic controllers to deliver consistent and safe spacing between aircraft leading to increased airport capacity, enhanced safety, reduced fuel consumption, improved predictability and increased resilience.

While the current solutions range from actively modifying/dissipating the wake-vortices using physical devices [320, 321] to accurately estimating the strengths of the vortices using LIDARS and RADARS [322, 317], one shortfall of the two approaches is that none of them predicts the evolution of the vortices in the future. This gap is being filled by advanced computational fluid dynamics modeling which involves solving the highly non-linear Navier-Stokes equations at varying levels of approximations. However, their utility owing to their computationally demanding nature has been limited to offline simulations geared towards developing a better understanding of the WV dynamics. At the moment, most of the fast WV models that are state-of-the art in WV predictive systems use physics-based empirical parameterizations to mimic

vortex transport and decay. Unfortunately, the computational efficiency of the fast WV models comes at the expense of accuracy. A good overview of the models can be found in [323]. To alleviate the problems associated with the existing WV models, data-driven machine learning methods might appear attractive at a first glance, but their limited interpretability owing to their black-box nature make them a misfit for the kind of safety-critical application under consideration.

To this end, building upon our recent works on the hybrid analysis and modeling (HAM) framework [82, 305, 2], we present a data assimilation-empowered approach to utilize a machine learning methodology to fuse computationally-light physics-based models with the available real-time measurement data to provide more accurate and reliable predictions of wake-vortex transport. In particular, we build a surrogate reduced order model (ROM), by combining proper orthogonal decomposition (POD) for basis construction [84, 85, 324, 86, 325, 326] and Galerkin projection to model the dynamical evolution on the corresponding low-order subspace [35, 327–329, 27, 330, 331, 40, 332]. Although ROMs based on Galerkin projection (denoted as GROMs in the present study) have been traditionally considered the standard approach for reduced order modeling, they often become inaccurate and unstable for long-term predictions of convection-dominated flows with strong nonlinearity [56, 108, 44–46]. Ideas like closure modeling [48–53, 55, 57, 58, 60–64, 295, 156, 333, 334, 47, 335, 336, 147] and Petrov-Galerkin projection [96, 97, 337, 98, 338–340, 99, 341] have been investigated to address this deficiency. Alternatively, we exploit the nudging method [342] as a data assimilation (DA) framework, which works by relaxing the model state toward observations by adding correction (or nudging) terms, proportional to the difference between observations and model state, known as innovation in DA context. In classical DA nudging, this proportionality is assumed to be linear, and the proportionality constants (or weights) are empirically tuned. Instead, we introduce the hybridization at this stage, using a simplistic long short-term memory (LSTM) architecture to generalize this relation to consider nonlinear mappings among the innovation and nudging terms.

In other words, we utilize LSTM to combine the possibly defective model prediction with noisy measurements to "nudge" the model's solution towards the true states [245, 148]. We apply the proposed LSTM nudging framework (denoted LSTM-N) for the reduced order modeling of the two-dimensional wake vortex problem in order to accurately predict the transport of wake vortices. Moreover, we suppose that both inputs (i.e., the physics-based model and data) are imperfect, thus avoiding biases in

predictions. GROMs are inherently imperfect due to the modal truncation and intrinsic nonlinearity. Recurrent neural networks, and in particular LSTMs, have shown success modeling the effect of truncated scales on the retained ones (i.e., model closure) with roots from the Mori-Zwanzig formalism. For example, Wang et al. [178] utilized a conditioned LSTM for the memory term in the GROM equations, representing the closure model, for parametric systems. In addition to model imperfection, we also perturb the initial conditions to further mimic erroneous state estimates in practice. Meanwhile, we realize that, more often than not, sensor signals are noisy. So, we intentionally inject some noise to the synthesized observation data (using a twin-experiment approach). We test the performance of LSTM-N with various levels of measurement noises, initial field perturbations, and sensors signals sparsity.

7.3 Wake-Vortices Transport and Decay Prediction System

Every aircraft generates a wake of turbulent air as it flies. This disturbance is caused by a pair of tornado-like counter-rotating vortices (called wake vortex) that trail from the tips of the wings [343]. Relatively turbulent weather conditions and rough terrain can help dissipate these vortices. A faster wake-decay was seen with increase in terrain roughness in [344] for wind-farms, where it was observed that a secondary vortex (SV) gets established more rapidly around the periphery of primary wake-vortex (WV), and the subsequent interactions between SV and WV creates a higher turbulence state. The phenomena of WV rebound and generation of omega-shaped hair-pin vortices take place during this SV-WV interaction. This complex wake decay phenomena is also applicable for wake-vortex emanating from aircraft. Such facts have also been observed and exploited to artificially destroy wakes close to the ground using plates [345]. Therefore, understanding the complex dynamics of these wake vortices (WV) from its generation to decay is important in order to ensure flight safety, to increase airport capacity and to test new methods for destroying WVs and mitigating their effect.

Air traffic control can potentially benefit from the emerging concept of a digital twin (DT), defined as the virtual replica of a physical system, where both of them are able to actively communicate with each other [13, 234, 130]. Given the WV and associated airport traffic case, a DT would receive streams of data, concerning operating conditions, airport traffic status, aircraft characteristics (e.g., weight, size) and flight mode (e.g., take-off or landing). Then, the DT should process these data

and assess a bunch of possible scenarios corresponding to potential choices to provide an informed decision with regard to the separation distance and flight scheduling, for instance. Considering the WV problem, numerical modeling based on the Navier-Stokes equation, if accurate, can be a cost effective and easily employable pursuit for wake analysis. In Fig. 7.1, an aircraft is admitted to land safely, based on the decay of wake-vortices from a leading aircraft. These wake-vortices are generated using the aircraft information and an analytical function on a set of two-dimensional (2D) planes (also called gates) perpendicular to the flight path [346]. Once the flight corridor is clear and free of any influence of the wake-vortices left behind by the leading aircraft, the following aircraft can land or take-off safely.

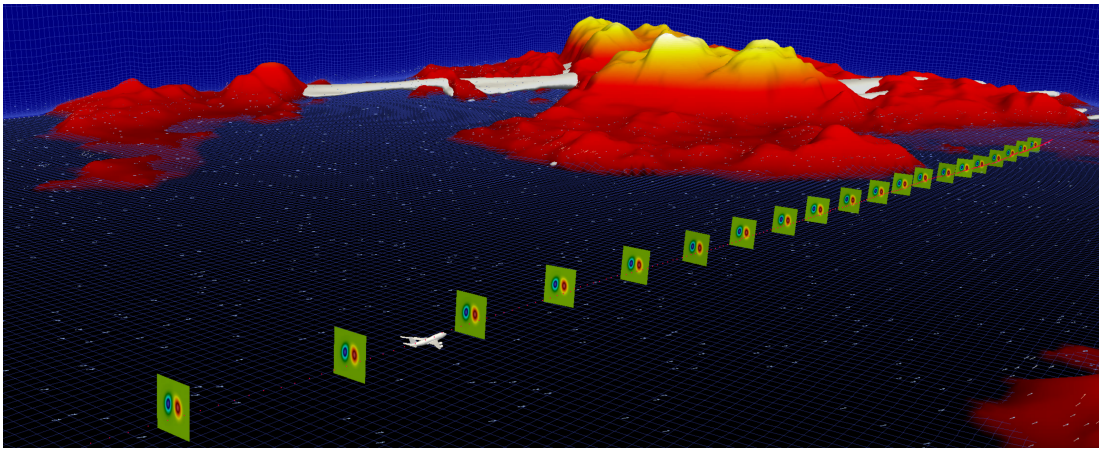


Figure 7.1: Transported and diffused wakes on a set of 2D planes (a.k.a. gates) to make sure that the flight corridor is clear for the following aircraft.

Nonetheless, direct full order numerical simulations require large discretized systems for adequate approximation and are not practical for real-time wake prediction, which is an essential ingredient for feasible DT technologies. Therefore, reduced order modeling (ROM) rises as a natural choice for the successful implementation of DT applications. ROM represents a family of protocols that aim at emulating the relevant system's dynamics with minimal computational burden. Typical ROM approaches consist of two major steps; (1) tailor a low-order subspace, where the flow trajectory can be sufficiently approximated to live (see Section 7.4.2), (2) build a surrogate model to cheaply propagate this trajectory in time (see Section 7.4.3). Traditionally, building surrogate models to evolve on a reduced manifolds has relied on the projection of the full order model (FOM) operators onto a reduced subspace (e.g., using Galerkin-type techniques) to structure a reduced order model (ROM). Those FOM operators are usually the outcome of the numerical discretization of the well-

established governing equation, derived from first principles and conservation laws. Such ROMs are attractive due to their reasonable interpretability and generalizability, as well as the existence of robust techniques for stability and uncertainty analysis. However, Galerkin ROM (GROM) can be expensive to solve for turbulent and advection-dominated flows. GROM also might suffer from inaccuracies and instabilities for long-time predictions. Meanwhile, in the digital twin context, the availability of rich stream of data and measurements opens new avenues for further ROM development. One way to utilize this abundance of data is the purely data-driven nonintrusive ROM (NIROM) approach. NIROMs have largely benefited from the widespread of open-source cutting edge and easy-to-use machine learning (ML) libraries, and cheap computational infrastructure to solely rely on data for building stable and accurate models, compared to their GROM counterparts [68, 67, 347–351, 211, 352–354]. However, purely data-driven tools often lack human interpretability and generalizability, and sometimes become prohibitively "data-hungry".

Alternatively, hybrid approaches can be pursued, where data-driven tools only assist the physics-based models whenever data are available, rather than replacing them entirely. Data assimilation (DA) is a framework which can efficiently achieve this objective. DA generally refers to the discipline of intelligently fusing theory and observations to yield an optimal estimate of the system's evolution [252–254, 355, 356]. Measurements are usually sparse (both in time and space) and noisy, while dynamical models are often imperfect due to the underlying assumptions and approximations introduced during either model derivation (e.g., neglecting source terms) or numerical solution of the resulting model (e.g., truncation error). DA algorithms have rich history in numerical weather predictions and are utilized on a daily basis to provide reliable forecasts. In this chapter, we suppose that our dynamical model is the truncated GROM and we aim at utilizing live streams of measurements to correct the GROM trajectory. Specifically, we exploit the nudging method as our data assimilation framework, which works by relaxing the model state toward observations by adding correction (or nudging) terms, to mitigate the discrepancy between observations and model state [357]. We employ LSTM mappings to account for this nudging term based on a combination between GROM predictions and available measurement data (see Section 7.4.4). We highlight that a similar approach was proposed in [245] for the state correction of the Lorenz 96 system. However, in that study, no model order reduction was employed and the model was assumed to be perfect. Moreover, a static correction was adopted using distinct background and assimilated trajecto-

ries, where the background trajectory is not updated each assimilation window. In contrast, we follow a dynamic correction approach where the corrected states after each assimilation window act as the background initial condition for the following window. This overall methodology was briefly introduced in [148] for the ROM of the 1D Burgers problem, and in the present study, we extend this approach to the 2D case of wake vortex transport problem and exploring higher sparsity ratios. We also aim at emphasizing and demonstrating the potential of the LSTM-N method for digital twin frameworks for real-time monitoring and control.

7.4 Methodology

In this section, we first give an overview of the full order model used to simulate the wake-vortex transport problem. Then, we present the reduced order formulations adopted in this study. In particular, we utilize proper orthogonal decomposition (POD) as a data-driven tool to extract the flow's coherent structures and build a reduced order subspace that best approximate the flow fields of interest. After that, we utilize a Galerkin approach to project the full order model operators onto that reduced space to build a *structure-preserving*, *physics-constrained* reduced order model.

7.4.1 Vorticity transport equation

We consider the two-dimensional (2D) vorticity transport equation as our full order model (FOM) that resolves the wake-vortex transport, defined by the 2D Navier-Stokes equations in vorticity-streamfunction formulation as follows,

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \nabla^2 \omega, \tag{7.1}$$

where ω and ψ denote the vorticity and streamfunction fields, respectively. Re is the dimensionless Reynolds number, relating the inertial and viscous effects. Equation (7.1) is complemented by the kinematic relationship between the vorticity and streamfunction as below,

$$\nabla^2 \psi = -\omega. \tag{7.2}$$

Equations (7.1) and (7.2) involve two operators, the Jacobian ($J(\cdot,\cdot)$) and the Laplacian ($\nabla^2(\cdot)$) defined as

$$J(\omega,\psi) = \frac{\partial\omega}{\partial x}\frac{\partial\psi}{\partial y} - \frac{\partial\omega}{\partial y}\frac{\partial\psi}{\partial x}, \tag{7.3}$$

$$\nabla^2\omega = \frac{\partial^2\omega}{\partial x^2} + \frac{\partial^2\omega}{\partial y^2}. \tag{7.4}$$

In order to mimic the wake-vortex problem, several models have been investigated [322, 358, 359, 316], including Gaussian vortex [360], Rankine vortex [361, 362], Lamb-Oseen vortex [363, 364], and Proctor vortex [365, 366] among others. In the present study, we initialize the flow with a pair of counter-rotating Gaussian vortices with equal strengths centered at (x_1, y_1) and (x_2, y_2) as follows,

$$\omega(x,y,0) = \exp\left(-\rho\left[(x-x_1)^2 + (y-y_1)^2\right]\right) - \exp\left(-\rho\left[(x-x_2)^2 + (y-y_2)^2\right]\right), \tag{7.5}$$

where ρ is an interacting parameter that controls the mutual interactions between the two vortical motions. We also consider periodic boundary conditions for the demonstration provided in the current study.

7.4.1.1 Numerical methods

For the spatial discretization of Eq. (7.1), we use the standard second-order central finite difference scheme in linear term as follows,

$$\nabla^2\omega_{i,j} = \frac{\omega_{i+1,j} - 2\omega_{i,j} + \omega_{i-1,j}}{\Delta x^2} + \frac{\omega_{i,j+1} - 2\omega_{i,j} + \omega_{i,j-1}}{\Delta y^2}, \tag{7.6}$$

where Δx and Δy are the mesh sizes in the x- and y-directions, respectively. For the nonlinear term, Arakawa[251] suggested that the conservation of energy, enstrophy, and skew-symmetry is sufficient to avoid computational instabilities stemming from nonlinear interactions. Therefore, we adopt the following second order Arakawa scheme for the Jacobian term,

$$J(\omega_{i,j},\psi_{i,j}) = \frac{1}{3}(J_1 + J_2 + J_3), \tag{7.7}$$

where the discrete Jacobians have the following forms,

$$J_1 = \frac{1}{4\Delta x \Delta y}\Big[(\omega_{i+1,j} - \omega_{i-1,j})(\psi_{i,j+1} - \psi_{i,j-1}) - (\omega_{i,j+1} - \omega_{i,j-1})(\psi_{i+1,j} - \psi_{i-1,j})\Big],$$

$$J_2 = \frac{1}{4\Delta x \Delta y}\Big[\omega_{i+1,j}(\psi_{i+1,j+1} - \psi_{i+1,j-1}) - \omega_{i-1,j}(\psi_{i-1,j+1} - \psi_{i-1,j-1})$$
$$- \omega_{i,j+1}(\psi_{i+1,j+1} - \psi_{i-1,j+1}) + \omega_{i,j-1}(\psi_{i+1,j-1} - \psi_{i-1,j-1})\Big],$$

$$J_3 = \frac{1}{4\Delta x \Delta y}\Big[\omega_{i+1,j+1}(\psi_{i,j+1} - \psi_{i+1,j}) - \omega_{i-1,j-1}(\psi_{i-1,j} - \psi_{i,j-1})$$
$$- \omega_{i-1,j+1}(\psi_{i,j+1} - \psi_{i-1,j}) + \omega_{i+1,j-1}(\psi_{i+1,j} - \psi_{i,j-1})\Big].$$

Nonetheless, solving Eq. (7.2) is often the most computationally-demanding part for typical incompressible flow solvers. In our study, we make use of the periodicity of boundary conditions and implement a fast Poisson solver employing the fast Fourier transform (FFT). We first discretize Eq. (7.2) using the standard second-order central finite difference scheme (similar to Eq. (7.6)) as follows,

$$\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} = -\omega_{i,j}, \tag{7.8}$$

Then, we apply the forward FFT to find the Fourier coefficients $\hat{\omega}_{i,j}$ from the grid values of $\omega_{i,j}$. Thus, the Fourier coefficients $\hat{\psi}_{i,j}$ for the streamfunction can be evaluated as follows,

$$\hat{\psi}_{i,j} = -\frac{\hat{\omega}_{i,j}}{c_1 \cos(2\pi i/N_x) + c_2 \cos(2\pi j/N_y) - c_3}, \tag{7.9}$$

where $c_1 = 2/\Delta x^2$, $c_2 = 2/\Delta y^2$, and $c_3 = c_1 + c_2$. Finally, we apply an inverse FFT to find the grid values $\psi_{i,j}$ from their Fourier coefficients $\hat{\psi}_{i,j}$.

7.4.2 Proper orthogonal decomposition

The first step for building a projection-based reduced order model is to tailor a low-order subspace that is capable of capturing the essential features of the system of interest. In the fluid mechanics community, proper orthogonal decomposition (POD) is one of the most popular techniques in this regard [25, 134, 28]. Starting from a collection of system's realizations (snapshots), POD provides a systematic algorithm to construct a set of orthonormal basis functions (called POD modes) that best describes that collection of snapshot data (in the ℓ_2 sense). More importantly, those

bases are sorted based on their contributions to the system's total energy, making the modal selection a straightforward process. This is a significant advantage compared to other modal decomposition techniques like dynamic mode decomposition, where further sorting and selection criterion has to be carefully defined. Usually, the method of snapshots [83] is followed to perform POD efficiently and economically, especially for high dimensional systems. However, we demonstrate the singular value decomposition (SVD) based approach here for the sake of simplicity and brevity of presentation.

We collect N system realizations, denoted as $\omega_k = \{\omega(x_i, y_j, t_k)\}_{i=1,j=1,k=1}^{i=N_x,j=N_y,k=N}$, thus we build a snapshot matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$ as $\mathbf{A} = [\mathbf{\Omega}_1, \mathbf{\Omega}_2, \ldots, \mathbf{\Omega}_N]$, where $\mathbf{\Omega}_k \in \mathbb{R}^{M\times 1}$ is the k^{th} snapshot reshaped into a column vector, M is the number of spatial locations (i.e., $M = N_x N_y$), and N is the number of snapshots. Then, a thin (reduced) SVD is performed on $\mathbf{A}$,

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{7.10}$$

where $\mathbf{U} \in \mathbb{R}^{M\times N}$ is a matrix with orthonormal columns defining the left singular vectors of $\mathbf{A}$ while the columns of $\mathbf{V} \in \mathbb{R}^{N\times N}$ denote the right singular vectors of $\mathbf{A}$. We note that the columns of $\mathbf{U}$ represent the spatial basis, and the columns of $\mathbf{V}$ carry the respective temporal information. The singular values of $\mathbf{A}$ are stored in descending order as the entries of the diagonal matrix $\mathbf{\Sigma} \in \mathbb{R}^{N\times N}$. For model order reduction purposes, only the first R columns of $\mathbf{U}$, the first R columns of $\mathbf{V}$, and the upper-left $R \times R$ sub-matrix of $\mathbf{\Sigma}$ are considered, corresponding to the largest R singular values. Specifically, the first R columns of $\mathbf{U}$ represent the most energetic R POD modes, denoted as $\{\phi_k\}_{k=1}^{R}$ for now on.

The vorticity field $\omega(x, y, t)$ is thus approximated as a linear superposition of the contributions of the first R modes, which can be mathematically expressed using the Galerkin ansatz as

$$\omega(x, y, t) = \sum_{k=1}^{R} a_k(t)\phi_k(x, y), \tag{7.11}$$

where $\phi_k(x, y)$ stand for the spatial modes, $a_k(t)$ designate the time-dependent modal coefficients/amplitudes (also known as generalized coordinates), and R is the number of the retained modes in the ROM approximation (i.e., ROM dimension). We note

that the POD basis functions ϕ are orthonormal by construction as follows,

$$\langle \phi_i; \phi_j \rangle = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{otherwise,} \end{cases} \tag{7.12}$$

where the angle parentheses $\langle \cdot; \cdot \rangle$ stands for the standard inner product in Euclidean space (i.e., dot product). We highlight that the SVD-based computation of the POD basis imply the use of the Euclidean inner product. However, this might be problematic, especially when combined with non-uniform or unstructured grids. For such cases, the of other inner products (e.g., L^2 or H^1) would be recommended.

Since the vorticity and streamfunction fields are related by Eq. (7.2), they share the same modal amplitudes, $a_k(t)$. Moreover, the basis functions for the streamfunction (denoted as $\theta_k(x, y)$) are derived from those of the vorticity as follows,

$$\nabla^2 \theta_k = -\phi_k, \qquad k = 1, 2, \ldots, R, \tag{7.13}$$

and the ROM approximation of the streamfunction field can be written as

$$\psi(x, y, t) = \sum_{k=1}^{R} a_k(t) \theta_k(x, y), \tag{7.14}$$

7.4.3 Galerkin projection

After constructing a set of POD basis functions, an orthogonal Galerkin projection can be performed to obtain the Galerkin ROM (GROM). To do so, the ROM approximation (Eqs. (7.11) and (7.14)) is substituted into the governing equation (Eq. (7.1)). Noting that the POD bases are only spatial functions (i.e., independent of time) and the modal amplitudes are independent of space, we get the the following set of ordinary differential equations (ODEs) representing the tensorial GROM

$$\frac{\mathrm{d}a_k}{\mathrm{d}t} = \sum_{i=1}^{R} \mathfrak{L}_{i,k} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,k} a_i a_j, \tag{7.15}$$

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

once during an offline stage as

$$\mathfrak{L}_{i,k} = \Big\langle \frac{1}{\mathrm{Re}} \nabla^2 \phi_i ; \phi_k \Big\rangle, \qquad \mathfrak{N}_{i,j,k} = \Big\langle -J(\phi_i, \theta_j); \phi_k \Big\rangle.$$

Equation (7.15) can be rewritten in compact form as

$$\dot{\mathbf{a}} = \mathbf{f}(\mathbf{a}), \tag{7.16}$$

where the (continuous-time) model map $\mathbf{f}$ is defined as follows,

$$\mathbf{f} = \begin{bmatrix} \sum_{i=1}^{R} \mathfrak{L}_{i,1} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,1} a_i a_j \\ \sum_{i=1}^{R} \mathfrak{L}_{i,2} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,2} a_i a_j \\ \vdots \\ \sum_{i=1}^{R} \mathfrak{L}_{i,R} a_i + \sum_{i=1}^{R} \sum_{j=1}^{R} \mathfrak{N}_{i,j,R} a_i a_j \end{bmatrix}.$$

Alternatively, Eq. (7.16) can be given in discrete-time form as

$$\mathbf{a}^{n+1} = \mathbf{M}(\mathbf{a}^n), \tag{7.17}$$

where $\mathbf{M}(\cdot)$ is a one time-step forward mapping obtained by any suitable temporal integration technique. Here, we use the fourth-order Runge-Kutta (RK4) method as follows,

$$\mathbf{a}^{n+1} = \mathbf{a}^n + \frac{\Delta t}{6}(\mathbf{g}_1 + 2\mathbf{g}_2 + 2\mathbf{g}_3 + \mathbf{g}_4), \tag{7.18}$$

where

$$\mathbf{g}_1 = \mathbf{f}(\mathbf{a}^n), \quad \mathbf{g}_2 = \mathbf{f}(\mathbf{a}^n + \frac{\Delta t}{2}\mathbf{g}_1), \quad \mathbf{g}_3 = \mathbf{f}(\mathbf{a}^n + \frac{\Delta t}{2}\mathbf{g}_2), \quad \mathbf{g}_4 = \mathbf{f}(\mathbf{a}^n + \Delta t \mathbf{g}_3).$$

Thus the discrete-time map defining the transition from time t_n to time t_{n+1} is written as

$$\mathbf{M}(\mathbf{a}^n) = \mathbf{a}^n + \frac{\Delta t}{6}(\mathbf{g}_1 + 2\mathbf{g}_2 + 2\mathbf{g}_3 + \mathbf{g}_4). \tag{7.19}$$

7.4.4 Long short-term memory nudging

Due to the quadratic nonlinearity in the governing equation (and consequently the GROM), the *online* computational cost of solving Eq. (7.15) is $O(R^3)$ (i.e., it scales

cubically with the number of retained modes). Therefore, this has to be kept as low as possible for feasible implementation of ROM in digital twin applications that require near real-time responses. However, this is often not an easy task for systems with slow decay of the Kolmogorov n-width. Examples include advection-dominated flows with strong nonlinear interactions among a wide span of modes. As a result, the resulting GROM is intrinsically imperfect model. This imperfection implies that the GROM might give inaccurate or false predictions even when fed with the true initial conditions and in the absence of numerical discretization errors.

Moreover, in most realistic cases, proper specification of the initial state, boundary conditions, and/or model parameters is rarely attainable. This uncertainty in problem definition, in conjunction with model imperfection, poses challenges for accurate predictions. In this study, we put forth a nudging-based methodology that fuses prior model forecast (using imperfect initial condition specification and imperfect model) with the available Eulerian sensor measurements to provide more accurate posterior prediction. Relating our setting to realistic applications, we build our framework on the assumption that measurements are noisy and sparse both in space and time. Nudging has a prestigious history in data assimilation, being a simple and unbiased approach. The idea behind nudging is to penalize the dynamical model evolution with the discrepancy between model's predictions and observations [367–369]. In other words, the forward model given in Eq. (7.17) is supplied with a nudging (or correction) term rewritten in the following form,

$$\mathbf{a}^{n+1} = \mathbf{M}(\mathbf{a}^n) + \mathbf{G}(\mathbf{z}^{n+1} - h(\mathbf{a}^{n+1})), \tag{7.20}$$

where $\mathbf{G}$ is called the nudging (gain) matrix and $\mathbf{z}$ is the set of measurements (observations), while $h(\cdot)$ is a mapping from model space to observation space. For example, $h(\cdot)$ can be a reconstruction map, from ROM space to FOM space. In other words, $h(\mathbf{a})$ represents the "model forecast for the measured quantity", while $\mathbf{z}$ is the "actual" observations. Despite the simplicity of Eq. (7.20), the specification/definition of the gain matrix $\mathbf{G}$ is not as simple [370–372, 342].

In the proposed framework, we utilize a recurrent neural network, namely the long short-term memory (LSTM) variant, to define the nudging map. We denote our approach as LSTM-Nudging (LSTM-N). In particular, Eq. (7.20) implies that each component of $\mathbf{a}^{n+1}$ (i.e., $a_1, a_2 \ldots, a_R$) is corrected using a *linear* superposition of the components of $\mathbf{z}^{n+1} - h(\mathbf{a}^{n+1})$, weighted by the gain matrix. Here, we relax this

linearity assumption and generalize it to a possibly nonlinear mapping $\mathbf{C}(\mathbf{a}, \mathbf{z})$ as,

$$\mathbf{a}^{n+1} = \mathbf{M}(\mathbf{a}^n) + \mathbf{C}(\mathbf{a}_b^{n+1}, \mathbf{z}^{n+1}), \tag{7.21}$$

where the map $\mathbf{C}(\mathbf{a}, \mathbf{z})$ is learnt (or inferred) using an LSTM neural network, and $\mathbf{a}_b^{n+1}$ is the prior model prediction computed using imperfect model and/or imperfect initial conditions (called background in data assimilation terminology), defined as $\mathbf{a}_b^{n+1} = \mathbf{M}(\mathbf{a}^n)$. Thus, Eq. (7.21) can be rewritten as follows,

$$\mathbf{a}^{n+1} = \mathbf{a}_b^{n+1} + \mathbf{C}(\mathbf{a}_b^{n+1}, \mathbf{z}^{n+1}). \tag{7.22}$$

In order to learn the map $\mathbf{C}(\mathbf{a}_b, \mathbf{z})$, we consider the case with imperfect model, defective initial conditions, and noisy observations. Moreover, we suppose sensors are sparse in space and measurement signals are sparse in time, too. Specifically, we use sensors located at a few equally-spaced grid points, but a generalization to off-grid or adaptive sensor placement is possible. Also, we assume sensors send measurement signals every τ time units. In order to mimic sensor measurements and noisy initial conditions, we run a twin experiment as follows,

1. Solve the FOM equation (i.e., Eq. (7.1)) and sample *true* field data ($\omega_{true}(x, y, t_n)$) each τ time units. In other words, store $\omega_{true}(x, y, t_n)$ at $t_n \in \{0, \tau, 2\tau, \dots T\}$ where T is the total (maximum) time and τ is the time window over which measurements are collected.

2. Define erroneous initial field estimate as $\omega_{err}(x, y, t_n) = \omega_{true}(x, y, t_n) + \epsilon_b$, where $t_n \in \{0, \tau, 2\tau, \dots T - \tau\}$. Here, ϵ_b stands for noise in initial state estimate, assumed as white Gaussian noise with zero mean and covariance matrix B (i.e., $\epsilon_b \sim \mathcal{N}(0, B)$).

3. Define sparse and noisy measurements as $\mathbf{z} = \omega_{true}(x_{Obs}, y_{Obs}, t_n) + \epsilon_m$, for $t_n \in \{\tau, 2\tau, \dots T\}$. Similarly, ϵ_m stands for the measurements noise, assumed to be white Gaussian noise with zero mean and covariance matrix Q (i.e., $\epsilon_m \sim \mathcal{N}(0, Q)$).

For LSTM training data, we project the erroneous field estimates (from Step 2) onto the POD basis functions to get the erroneous POD modal coefficients (i.e., $\mathbf{a}_{err}(t_n)$, for $t_n \in \{0, \tau, 2\tau, \dots T - \tau\}$. Then, we integrate those erroneous coefficients for τ time units to get the background prediction $\mathbf{a}_b(t_n)$, for $t_n \in \{\tau, 2\tau, \dots T\}$. We note that in

the actual deployment, the ROM solver is *re-initialized* with the nudged states each τ time units when measurements become available. Therefore, we perform our training on a bunch of samples initiated at time t_n (with the Gaussian noise representing the uncertainty) and integrated up to $t_n + \tau$ to mimic the re-initialization each τ time units. That said, in Step 2, we use the same level of uncertainty at $t = 0$ (defined by the background covariance matrix B) as an upper limit for the uncertainty at the beginning of the following periods (i.e., $t = \tau, 2\tau, \dots$) since the nudging algorithm does not provide an update for the background covariance matrix (unlike Kalman filtering approaches which evolve this information along with the state).

We train the LSTM using $\mathbf{a}_b(t_n)$ and $\mathbf{z}(t_n)$ as inputs, and set the target as the correction $(\mathbf{a}_{true}(t_n) - \mathbf{a}_b(t_n))$, for $t_n \in \{\tau, 2\tau, \dots T\}$. The true modal coefficients ($\mathbf{a}_{true}$) are obtained by projecting the *true* field data (from Step 1) onto the POD bases, where the projection is defined via the inner product as $a_k(t) = \langle \omega(x, y, t); \phi_k(x, y) \rangle$. We remark that the LSTM-N framework is fed with the whole $\mathbf{a}_b(t_n)$ vector (not individual components) after propagating the GROM in time. Therefore, it is independent of the numerical method of time integration and it is generalizable to either explicit or implicit schemes. In order to enrich the training data set, Step 2 and Step 3 are repeated several times giving an ensemble of erroneous state estimates and noisy measurements at every time instant of interest. Each member of those ensembles represents one training sample. This also assists the LSTM network to handle wider range of noise.

We should highlight here that extra care should be taken when considering data-driven correction approaches as data-driven closure might yield non-physical results, and putting some extra constraints on the characteristics of the predicted closure is essential in several cases. Wu et al. [373] demonstrated that the Reynolds-averaged Navier–Stokes (RANS) with explicit treatment of data-driven closure can be ill-conditioned and yield large errors. In another study, Wu et al. [374] also proposed a physics-informed machine learning approach for improved data-driven RANS closure by training the ML models for the linear and nonlinear parts of the Reynolds stress separately. Nonetheless, we emphasize that the proposed LSTM-N approach not only cures model imperfection (i.e., provides model closure as well as accounts for any missing physical processes), but also treats uncertainties in initial state estimates. This might be caused by the selection of inaccurate wake vortex model, or the idealizations embedded in this model compared to reality. Moreover, the field measurements (i.e., the nudging data) are assumed to be sparse and noisy to mimic

real-life situations.

7.5 Results

In order to test and verify the proposed ideas, we consider a square 2D domain with a side length of 2π. Flow is initiated using a pair of Gaussian vortices as given in Eq. (7.5) centered at $(x_1, y_1) = \left(\frac{5\pi}{4}, \pi\right)$ and $(x_2, y_2) = \left(\frac{3\pi}{4}, \pi\right)$ with an interaction parameter of $\rho = \pi$. Results in this section are shown at $Re = 10^4$. For FOM simulations, a regular Cartesian grid resolution of 512×512 is considered (i.e., $\Delta x = \Delta y = 2\pi/512$), with a time-step of 0.001. Snapshots of vorticity fields are collected every 100 time-steps for $t \in [0, 30]$, totalling 300 snapshots. The evolution of the wake vortex problem is depicted in Fig. 7.2, demonstrating the convective and interactive mechanisms affecting the transport and development of the two vortices.

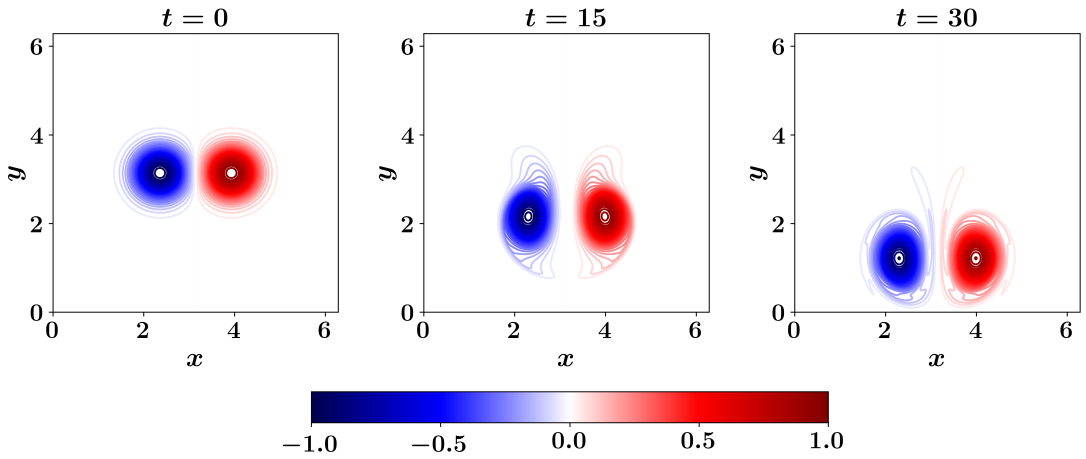


Figure 7.2: Evolution of the FOM vorticity field for the wake vortex transport problem with a Reynolds number of 10^4. Flow is initiated at time $t = 0$ with a pair of Gaussian distributed vortices.

For ROM computations, 6 modes are retained in the reduced order approximation (i.e., $R = 6$), capturing more than 99% of the snapshot data variance. A time step of 0.1 is adopted for the temporal integration of GROM equations. In order to implement the LSTM-N approach, we begin at erroneous initial condition defined as $\omega_{err}(x, y, 0) = \omega_{true}(x, y, 0) + \epsilon_b$, where $\omega_{true}(x, y, 0)$ is defined with Eq. (7.5), and ϵ_b is a white Gaussian noise with zero mean and covariance matrix B. For simplicity, we assume $B = \sigma_b^2 \mathbf{I}$, where σ_b is the standard deviation in the "background" estimate of the initial condition and $\mathbf{I}$ is the identity matrix. We note that this formulation implies that the errors in our estimates of the initial vorticity field at different spatial

locations are uncorrelated. As nudging field data, we locate sensors to measure the vorticity field $\omega(x, y, t)$ every 64 grid points (i.e., 9 sensors in each direction, with $s_{freq} = 64$, where s_{freq} is the number of spatial steps between sensors locations), and collect measurements every 10 time steps (i.e., each 1 time unit with $t_{freq} = 10$, where t_{freq} is the number of time steps between measurement signals). To account for noisy observations, a white Gaussian noise of zero mean and covariance matrix of Q is added to the true vorticity field obtained from the FOM simulation at sensors locations. Similar to B, we set $Q = \sigma_m^2 \mathbf{I}$, where σ_m is the standard deviation of measurement noise. This assumes that the noise in sensors measurements are not correlated to each other, and all sensors have similar quality (i.e., add similar amounts of noise to the measurements). As a base case, we set $\sigma_b = 1.0$, and $\sigma_m = 1.0$. These levels of initial condition perturbation and measurement noise are guided by the values vorticity fields varying between -1.0 and 1.0 as indicated by the colorbar in Fig. 7.2. Thus, setting $\sigma_b = 1.0$, and $\sigma_m = 1.0$ guarantees extensive levels of uncertainty in the background information and collected observations.

The procedure presented in Section 7.4.4 is applied using the numerical setup described above, and compared against the reference case of GROM with the erroneous initial condition and inherent model imperfections due to modal truncation (denoted as background forecast). Since we synthesize the initial condition perturbation and measurement noise using a pseudo-random number generator, we utilize an ensemble of 30 realizations with different seed numbers and the sample mean is computed from these 30 experiments. Whenever applicable, we also sketch uncertainty regions bounded by the sample mean +/- the standard deviation. In Fig. 7.3, the temporal evolution of the POD modal coefficients is shown for the true projection, background, and LSTM-N results. The true projection results are obtained by the projection of the true FOM field at different time instants onto the corresponding basis functions (i.e., via inner product, $a_{k,true}(t) = \langle \omega(x, y, t); \phi_k(x, y) \rangle$). The background trajectory is the reference solution obtained by standard GROM using the erroneous initial condition, without any closure or corrections. It can be seen that the background trajectory gets off the true trajectory by time as a manifestation of model. Also, note that the background solution does not begin from the same point as true projection due to the noise in the initial condition. On the other hand, the LSTM-N yields very good predictions, comparable to the true projection solution, implying that the approach is capable of blending noisy observations with a prior estimate to gain more accurate predictions.

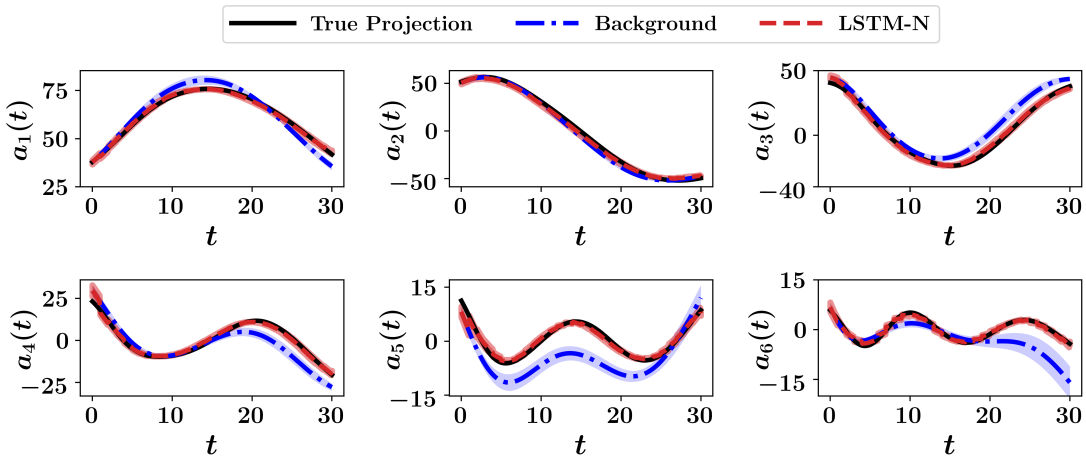


Figure 7.3: Temporal evolution of the POD modal coefficients for the 2D wake vortex transport problem. The dark lines denote the mean value from a sample of 30 realizations (using different seeds for the random number generator to simulate initial condition perturbation and measurement noise). The shaded area defines the mean values +/- the standard deviation of the sample. [Base case with $\sigma_b = 1.0$ and $\sigma_m = 1.0$]

In order to better visualize the predictive capabilities of the LSTM-N methodology, we compute the reconstructed vorticity field using Eq. (7.11). The ensemble average of final field reconstruction (at $t = 30$) is shown in Fig. 7.4, comparing the true projection, background, LSTM-N results. Note that the field obtained from true projection at any time instant can be computed as $\omega_{true}(x, y, t) = \sum_{k=1}^{R} a_{k,true}(t)\phi_k(x, y)$, and represents the optimal reduced-rank approximation that can be obtained using a linear subspace spanned by R bases. Comparing true projection results from Fig. 7.4 against FOM at final time from Fig. 7.2 reveals that, for this particular case, 6 modes are qualitatively capable to capture most of the relevant features of the flow field. The LSTM-N outputs significantly match the projection of the FOM field, while the background forecasts show some visible deviations.

We also compare the LSTM-N results against a simple forward nudging implementation (denoted as linear nudging) [375], where Eq. (7.20) is rewritten as follows,

$$\mathbf{a}^{n+1} = \mathbf{M}(\mathbf{a}^n) + \zeta \mathbf{D}_h^T(\mathbf{z}^{n+1} - h(\mathbf{a}^{n+1})), \tag{7.23}$$

where ζ is a constant and $\mathbf{D}_h$ is the Jacobian matrix of the observation operator $h(\cdot)$. In the present study, we find that a value of $\zeta = \Delta t$ provides acceptable results. We also highlight that different data assimilation techniques might be adopted.

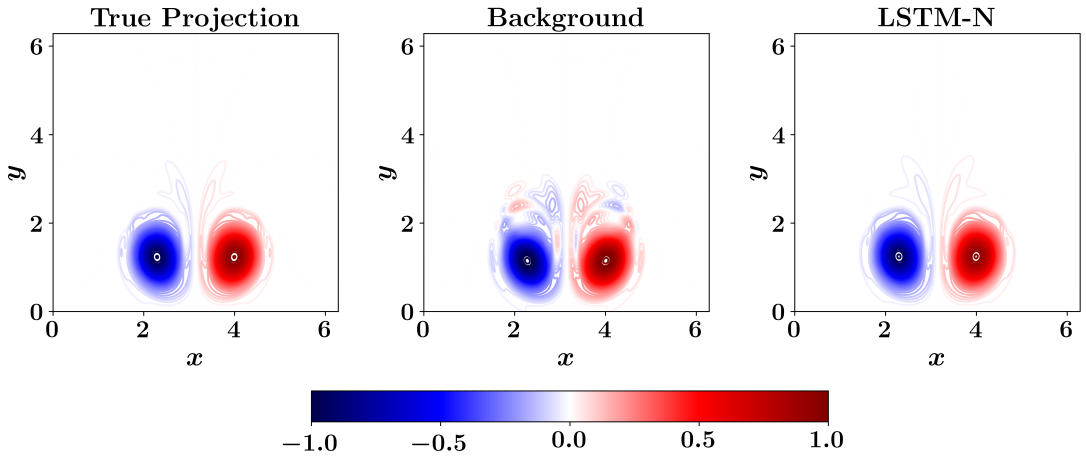


Figure 7.4: Final vorticity field (at $t = 30$) for the wake-vortex transport problem, with $\sigma_b = 1.0$, and $\sigma_m = 1.0$. Results show the mean of an ensemble of 30 realizations with different seeds for the random number generator.

For example, in the three dimensional variational (3DVAR) framework [254], solving Eq. (7.20) and defining the gain matrix are reformulated as an optimization problem. In particular, the output of the data assimilation (called analysis or analyzed state) is defined as the minimizer of the following cost functional,

$$J(\mathbf{a}^{n+1}) = \|\mathbf{a}^{n+1} - \mathbf{M}(\mathbf{a}^n)\|^2_{W_1} + \|\mathbf{z}^{n+1} - h(\mathbf{a}^{n+1})\|^2_{W_2}, \tag{7.24}$$

where W_1 and W_2 are some suitable symmetric positive definite (SPD) matrix. Most often, W_1 is set as the background covariance matrix, while W_2 is defined using the measurement noise covariance matrix. The 3DVAR, as a variational approach, addresses the inference problem by formulating an optimization problem, taking into account measurement noise and background perturbation. For linear observation operator $h(\cdot)$, this can be directly solved with proper definitions of the weighting matrices. However, for nonlinear operators, some approximations and/or linearization become necessary. Indeed, the LSTM-N can be also thought of as minimizing that cost functional iteratively using the training samples and the back-propagation algorithm.

For quantitative assessment, the root mean-squares error (RMSE) of the *whole* reconstructed field with respect to the FOM solution is calculated as a function of time as follows,

$$\epsilon_f(t) = \sqrt{\frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \Big(\omega_{FOM}(x_i, y_j, t) - \omega_{ROM}(x_i, y_j, t) \Big)^2}, \tag{7.25}$$

where ω_{FOM} is the true vorticity field obtained from solving the FOM equation, while ω_{ROM} is the reduced order approximation computed through true projection, background (reference) solution, or LSTM-N method. The RMSE at different times is plotted in Fig. 7.5, comparing the LSTM-N, Linear Nudging, and 3DVAR results. demonstrating the capability of LSTM-N framework to efficiently recover the optimal reconstruction given a few sparse measurements. We also remark that we find that the performance of the Linear Nudging and the 3DVAR approaches is highly dependent on the level of measurement noise. Indeed, for larger values of σ_m, the solution does not converge. This behavior might be attributed to the definition of the observation map using the ROM reconstruction process. In particular, we find that the basis functions, when sampled at sparse locations, yield ill-conditioned matrices, and the resulting mapping becomes very sensitive to the measurement noise. One approach to mitigate this issue is to exploit compressed sensing and adaptive sampling techniques to improve this reconstruction mapping and minimize the effect of noise.

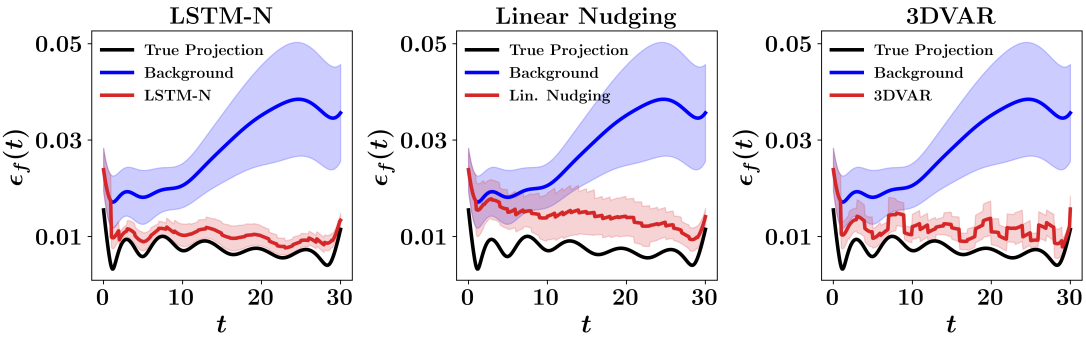


Figure 7.5: Root mean-squares error for the wake-vortex transport problem, with $\sigma_b = 1.0$, and $\sigma_m = 1.0$, using the LSTM-N, Linear Nudging, and 3DVAR approaches.

Another metric to reveal the inference quality of the framework is defined using the error between the true measurements and the predicted solution at the sensors locations only as follows,

$$\epsilon_i(t) = \sqrt{\frac{1}{N_x^{meas} N_y^{meas}} \sum_{i=1}^{N_x^{meas}} \sum_{j=1}^{N_y^{meas}} \Big(\omega_{FOM}(x_i, y_j, t) - \omega_{ROM}(x_i, y_j, t) \Big)^2}, \qquad (7.26)$$

where N_x^{meas} and N_y^{meas} represent the number of sensors in the x- and y-directions, respectively. Results for LSTM-N, Linear Nudging, and 3DVAR are presented in Fig. 7.6 (top row). We observe that the 3DVAR gives better accuracy at observation locations than the nudging frameworks (LSTM-N and Linear Nudging). In order to

understand this observation, we plot the absolute error between FOM solution and the ensemble average at final time from LSTM-N, Linear Nudging, and 3DVAR (i.e., $|\omega_{FOM}(x_i, y_j, t) - \omega_{ROM}(x_i, y_j, t)|$) in Fig. 7.6 (bottom row). We can notice that the 3DVAR error at the observation locations (denoted as black circles) is small (note the bluish color). However, away from these measurement points, the error becomes larger (as denoted by the reddish color). On the other hand, for the LSTM-N, the error is small almost everywhere except for a small portion near the front of the vortices.

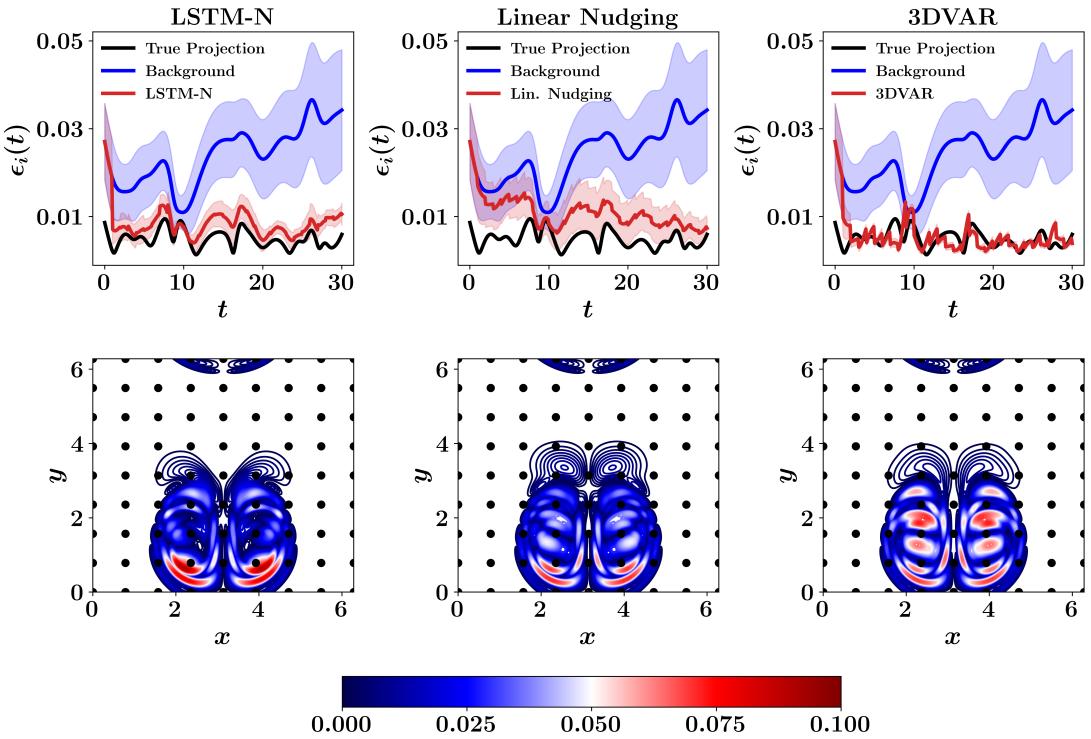


Figure 7.6: Comparison of LSTM-N, Linear Nudging, and 3DVAR predictions, with $\sigma_b = 1.0$, and $\sigma_m = 1.0$. Top rows depicts the inference quality metric define by Eq. (7.26) and bottom row shows the absolute error between FOM solution and the ensemble mean for the respective approaches. Observation locations are denoted using black circles.

7.5.1 Effect of noise

Next, We explore the effect of noise on the LSTM-N results. In other words, we investigate how much noise the framework can tolerate. We note that we keep the same LSTM, trained with the base level of noise (i.e., $\sigma_b = 1.0$ and $\sigma_m = 1.0$) while we test it using different levels of noise. First, we gradually increase the standard

deviation of measurement noise from 1.0 to 2.0 (2 times larger), 3.0 (3 times larger), and 4.0 (4 times larger). In Fig. 7.7, we plot the temporal evolution of ϵ_f metrics for the explored levels of measurement noise. We find that performance deteriorates a bit with an increase in measurement noise, especially in terms of uncertainty levels. Nonetheless, the predicted results are still significantly better than the background forecast (starting from the same initial conditions).

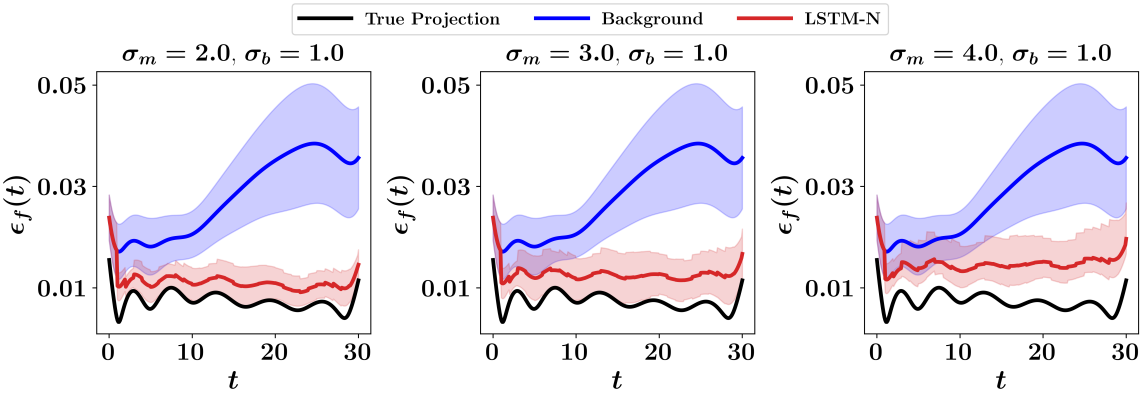
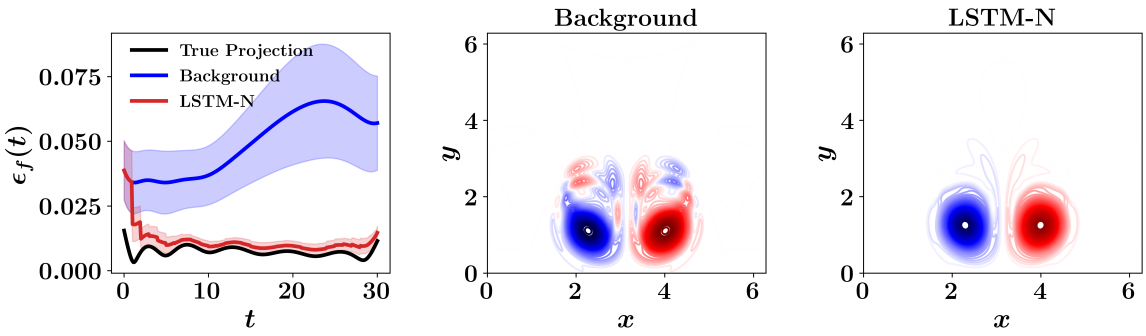


Figure 7.7: Root mean-squares error of LSTM-N predictions for different levels of measurement noise.

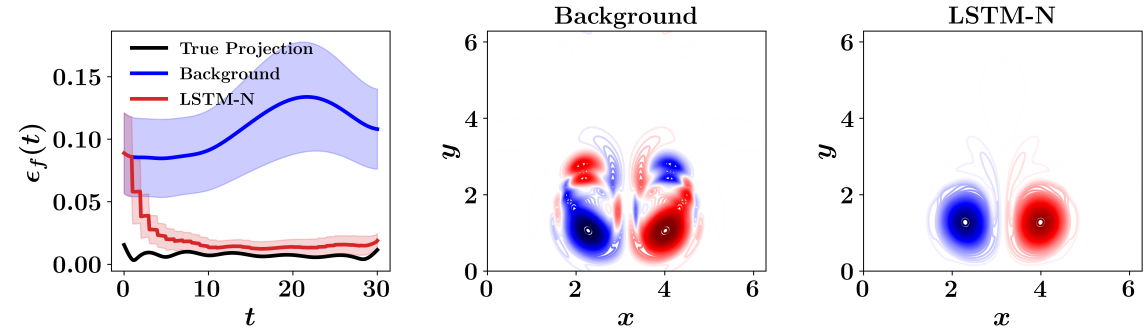
For testing the effect of initial state perturbation, we increase σ_b from 1 to 2, and 5. Figure 7.8 displays the effect of those levels of initial field perturbations on background forecasts. Despite that, LSTM-N is performing very well even at those high levels of initial perturbations. This is even clearer from the $RMSE$ plots, beginning from relatively large values and quickly decaying to the level of true projection once measurements are available. From Figs. 7.7 and 7.8, we can deduce that the influence of the level of measurement noise on LSTM-N performance is more prominent that of the initial field perturbation. We reiterate that in both cases, the LSTM is trained with $\sigma_b = 1.0$ and $\sigma_m = 1.0$ and tested for different noise and perturbation levels.

7.5.2 Effect of measurements sparsity

Finally, we consider the effect of measurement sparsity on the accuracy of the presented approach. This is crucial for the trade-off between quality and quantity of sensors, since it has been shown in Section 7.5.1 that measurement noise significantly affects the LSTM-N output. For the base case, sensors are placed at every 64 grid points. Now, we place sensors every 32 grid points, representing a denser case, as well as 128 and 256 grid points, representing scarcer sensors. We find that the framework



(a) $\sigma_m = 1.0$, $\sigma_b = 2.0$

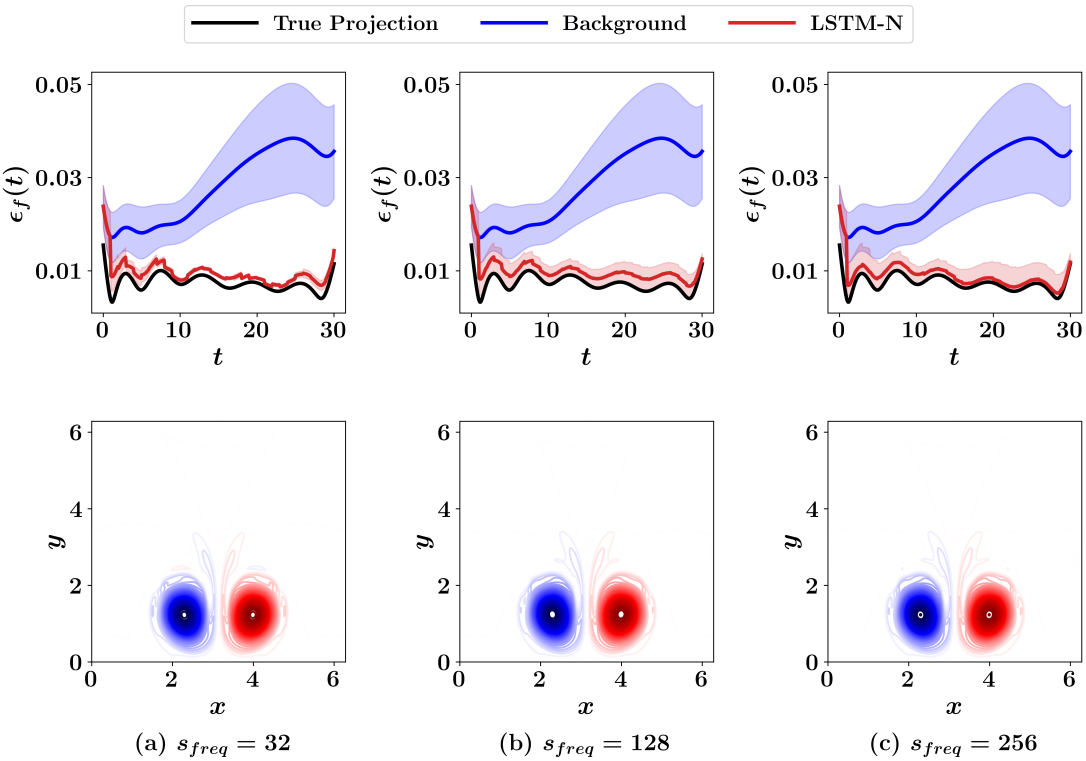


(b) $\sigma_m = 1.0$, $\sigma_b = 5.0$

Figure 7.8: Reconstructed vorticity fields at final time, along with $RMSE$ for different levels of background noise.

is quite robust, providing very good results as illustrated in Fig. 7.9. We note here, however, that the same original LSTM cannot be utilized for testing with varying sparsity. This is because sensors sparsity controls the size of the input vector. Therefore, a new LSTM has to be re-trained for each case with the corresponding number of measurements. Moreover, we notice that the LSTM training suffers for the dense case with $s_{freq} = 32$ (s_{freq} denote the number of grid points between every two consecutive sensors). This is due to the very large input vector, requiring excessive amounts of meta-data for proper training. We also observe that some of the features in the input vector become either redundant or useless (e.g., away from the vortices). In order to reduce the size of input vector in this case (i.e., $s_{freq} = 32$), we benefit from the similarities between the left and right vortices and consider measurements from only one half of the domain. We also emphasize that compressed sensing techniques should be adopted for optimized sensors placement, rather than the simple collocated equidistant arrangement followed in the present study.

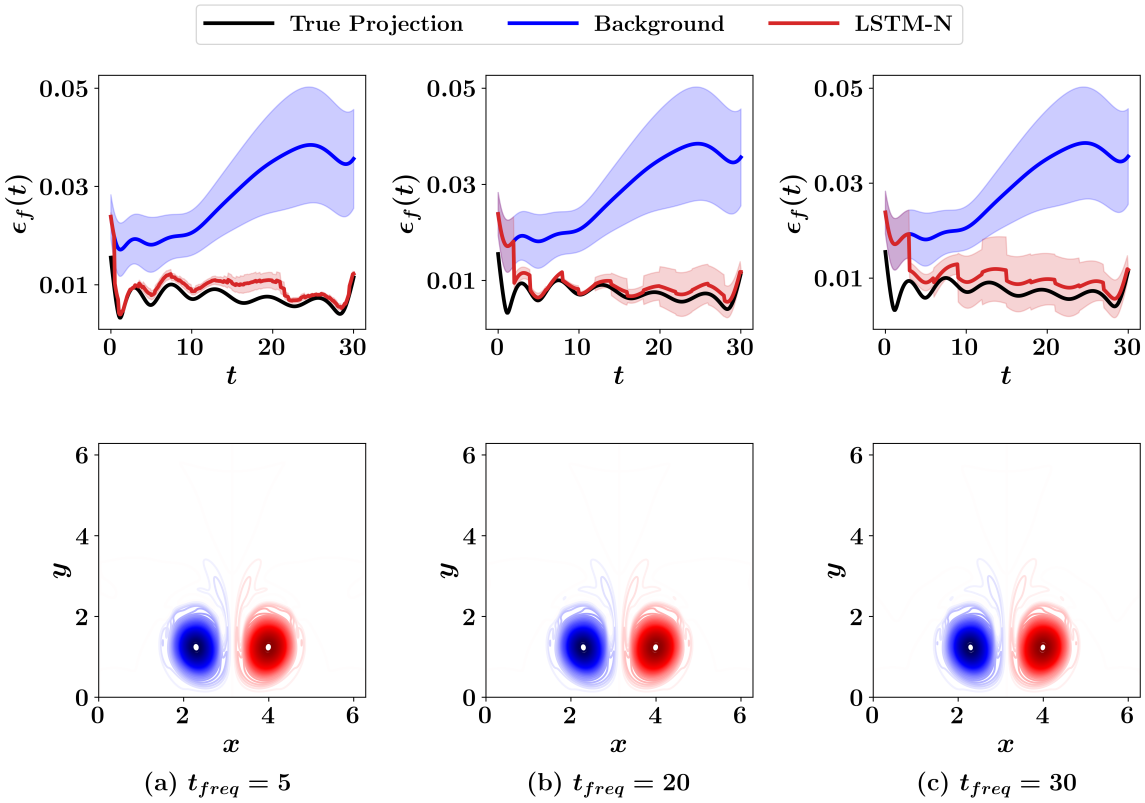
Regarding temporal sparsity, we collect measurement each 5, 20, and 30 time-steps, compared to the reference case where measurement are collected every 10



(a) $s_{freq} = 32$ (b) $s_{freq} = 128$ (c) $s_{freq} = 256$

Figure 7.9: Comparison of resulting vorticity fields at final time as well as the line plots of root mean-squares error at different times, with different number of sensors located sparsely at grid points. Here, s_{freq} denotes the number of grid points between every two consecutive sensors.

time-steps. We can see from Fig. 7.10 that all cases yield very good predictions. Nevertheless, we notice that increasing t_{freq} results in larger uncertainty bounds as the LSTM-N interferes at fewer time instants. Furthermore, ϵ_f plots provide valuable insights about the capability of LSTM-N to effectively fuse measurement with background forecast to produce more accurate state estimates. For example, when measurement signals are collected every 30 time-steps, this corresponds to 3 time-units, meaning that the LSTM-N directly adopts the GROM prediction without correction for this amount of time, before correction is added. This is evident from Fig. 7.10c, where the red curve starts and continues with the blue curve, then a sharp reduction of the ϵ_f (and the corresponding uncertainty) is observed. This behavior is repeated as the red curve departs from the black one (corresponding to true projection) before correction is added every $\tau = 3$ time-units (i.e., 30 time-steps). On the other hand, when more frequent measurement signals are available (e.g., every 5 time-steps), deviation from the true projection results is less observed, as shown in Fig. 7.10a.



(a) $t_{freq} = 5$ (b) $t_{freq} = 20$ (c) $t_{freq} = 30$

Figure 7.10: Comparison of resulting vorticity fields at final time as well as the line plots of root mean-squares error at different times using different measurement signal frequencies. Here, t_{freq} denotes the number of time steps between the measurement collection instants.

7.6 Conclusions

We demonstrate a machine learning based nudging approach for an idealized vortex transport problem. Such a hybrid analysis and modeling (HAM) approach is envisioned to be a promising enabler for digital twin application of an airport. Specifically, we investigate the problem of wake-vortex transport and decay as a key factor for the determination of separation distance between consecutive aircraft. Reduced order modeling based on Galerkin projection and proper orthogonal decomposition is adopted to provide computationally light models. We develop a methodology to exploit machine learning to cure model deficiency through online measurement data adopting ideas from dynamic data assimilation. Specifically, an LSTM architecture is trained to nudge prior predictions toward optimal state values using a combination of background information along with sparse and noisy observations. The proposed framework is distinguished from previous studies in the sense that it is built on the

assumption that all the computing ingredients are intrinsically imperfect, including a truncated GROM model, erroneous initial conditions, and defective sensors.

We study the effects of measurement noise and initial condition perturbation on LSTM-N behavior. The framework works sufficiently well for a wide range of noise and perturbation. Nonetheless, numerical experiments indicate relatively more dependence of performance on measurement quality (noise). Meanwhile, we find that sensors sparsity has minimal effects on results. We emphasize that the proposed framework represents a way of merging human knowledge, physics-based models, measurement information, and data-driven tools to maximize their benefits rather than discarding any of them. This becomes a key concept for building novel HAM approaches. The presented framework paves the way for viable digital twin applications to enhance airports capacities by regulating air traffic without compromising consecutive aircraft safety. Nevertheless, the scalability of the approach has yet to be tested using different vortex models and taking into account other effective factors (e.g., wind). For example, the wake vortex initial conditions can be related to specific aircraft types, using aircraft mass, span and speed. Also, as outlined in [8], 3D simulations are required to resolve instability mechanisms and turbulence in order to mimic wake vortex decay in a realistic way. Finally, the important effects of thermal stratification on WV descent and decay need to be considered in the transport equations.

CHAPTER 8

Concluding Remarks and Future Work

In this chapter we summarize the major facets of this study and outline some potential research that may emerge from our developments.

8.1 Summary of Study

In this dissertation, we have proposed hybrid analysis and modeling frameworks to enable next-generation of digital twins (DTs). The primary focus of the developed methods is large scale dynamical systems like those in the convection-dominated and turbulent fluid flow systems. In particular, we address three major challenges that are associated with digital twinning of such systems. These include (1) the large dimensionality of the system hindering the applicability of classical full order model (FOM) simulations, (2) the complexity of the system that requires an efficient synergy between heterogeneous solvers for different components, geometries, scales, and physics, and (3) the dynamical nature of the physical system that requires its digital replica to be self-adaptive for new flow regimes and operating conditions. These three challenges correspond to the three thrusts of the dissertation.

8.1.1 Reduced order modeling

Reduced order models (ROMs) have been recently leveraged to enable DT applications. In particular, projection-based ROMs have gained large popularity, wherein the governing equations or the FOM operators are projected onto a set of predefined basis functions. This generates a system of much fewer dynamical equations that define the evolution of the reduced variables. The standard approach involves the proper orthogonal decomposition (POD) for basis construction and the Galerkin method for predicting the ROM dynamics. Nonetheless, this approach usually yields inaccurate and unstable results for convective flows when a few basis functions are retained for computational efficiency. We put forth a set of hybrid physics-based and data-driven approaches to improve the quality of ROMs. Some of our developments

through the course of this research may be outlined in the following:

1. We put forth an uplifted ROM (UROM) methodology, characterized by three modeling layers for the accurate predictions of flow fields. In the first layer, we retain a physics-based ROM in the core of the framework to enhance the generalizability of the method for off-design conditions. A data-driven correction for the ROM dynamics is embedded in the second layer to represent the effects of the truncated modes onto the physics-based ROM dynamics. A key novelty of the UROM approach is defined in the third layer, where we exploit the capabilities of artificial neural networks to learn the nonlinear correlations between the resolved scales and the truncated ones. This third layer acts as a super-resolver that recovers some of the finer details of the flow field at the time of interest.

2. We propose a hybrid variational multi-scale (VMS) framework that benefits from the locality of modal interactions and information transfer to build more accurate ROMs. We attach the VMS framework with a physics-guided machine learning (PGML) methodology to learn the closure terms for different levels of resolved and unresolved scales. We encode the governing equations into the long short-term memory (LSTM) latent space by injecting the projection of the FOM operators on the respective modes into the LSTM hidden layers. We find that the developed three-level VMS ROM is superior to the standard two-level ROM. Moreover, the PGML yields more robust and reliable on-the-fly corrections than vanilla-type ML models as demonstrated with reduced uncertainty levels.

3. We devise a nonlinear POD (NLPOD) methodology that combines the physical soundness of the POD with the compression capabilities of neural network-based auto-encoders. The NLPOD provides a latent space compression for the near full-rank approximation of the flow field. The resulting NLPOD compression intrinsically respects the constitutive laws of underlying system. Moreover, it is equally applicable to uniform and non-uniform, structured and unstructured, and Cartesian, spherical, and cylindrical grids. This is in stark contrast to the existing convolutional auto-encoders which are specifically tailored for uniform and structured grids.

8.1.2 Interface learning

Complex systems are often characterized by multiple scales, physics, dominating dynamics, geometries, and mathematical abstractions. Utilizing a unique global solver for all of these is usually not possible as it would unnecessarily exhaust the available computational resources. Instead, a multitude of solvers are adopted to consider different aspects and portions of the system. Due to near real-time response requirements, an efficient synergy should be maintained to reduce the communication costs between different computing units and reduce the idle times. We launch an interface learning (IL) paradigm that take advantage of of available data sets, machine learning (ML) tools, and numerical analysis for the seamless integration of heterogeneous solvers. The following frameworks have been proposed to implement the IL methodology in multi-scale, multi-physics, and multi-component systems:

1. We propose an upwind learning algorithm that takes into account the direction of characteristics and wave structure in hyperbolic systems to enable physics-informed and data-enabled non-iterative domain decomposition. This approach significantly benefits high performance computing environments by reducing the communication costs among processing units in emerging ML ready heterogeneous platforms toward exascale era.

2. We develop a series of correction, uplifting, and prolongation mappings to enable the efficient coupling between FOM and ROM solvers. We apply the FOM-ROM coupling for a multi-component system dominated by convection and diffusion mechanisms with non-homogeneous physical properties. Moreover, we enable the multi-physics coupling for the Marsigli flow problem by dedicating ROM solver for the mass and momentum transfer while the energy transport is addressed by FOM solver.

8.1.3 Data assimilation

In ML and data-driven paradigms, there is always the notion of offline training and online deployment. In other words, the ML model is trained and parameterized by considering a loss or a utility function that is defined by a given set of data sets. After that, the model is put in place to make new predictions during an online phase. The applicability of the ML models is usually limited by the training algorithm and the training data sets. It can make predictions for conditions that are similar or close

to those that appear in the training phase. However, complex convection-dominated and turbulent flows are continuously evolving. The operating conditions and the flow regimes during the deployment stages might be significantly different from those in the training. Therefore, the DT should self-adapt to the new condition as we go. In order to add this capability, we ruggedize the hybrid framework using data assimilation (DA) algorithms to take advantage of in-situ measurements from the physical system. The following developments have been introduced in our study:

1. We utilize variational DA algorithms, including the forward sensitivity method (FSM), for the dynamic parameterization of closure models. In particular, we combine the Galerkin ROM dynamics with the sparse and noisy measurements of the flow field variables to estimate and update scale-aware control parameters to correct the ROM predictions.

2. We extend the classical nudging method by proposing an LSTM-Nudging framework that benefits from the neural network capabilities to learn nonlinear nudging operators. We illustrate the LSTM-Nudging for the efficient prediction of the wake-vortex transport and decay towards building DTs of airports.

8.2 Future Work

The studies described in this document give rise to some interesting questions and may be built on for the following future research:

1. An immediate extension to the proposed frameworks is to perform scalability tests and explore their performance and robustness in industrial settings with increased system complexity and actual field data sets.

2. While the hybrid VMS framework yields improved results, the separation between large and small resolved scales is performed arbitrarily and is believed to be sub-optimal. Instead, formal definitions of ROM length scales can be utilized for the determination of the number of modes in each level of the VMS framework.

3. Nonlinear proper orthogonal decomposition can be applied for the reduced order modeling of compressible flows where classical Galerkin-based methods are computationally expensive and unstable.

4. Automated hyper-parameter selection tools can be utilized to optimize the points of injecting the physics-based features into the latent space of the PGML algorithm.

5. Interface learning methods can be explored for the coupling between solvers with mixed geometries. For example, the blood flow in the whole circulatory system is mathematically described by means of heterogeneous problems describing organs and arteries with different degrees of detail and different geometric dimensions. Interface learning would enable the efficient interaction between these parts by learning appropriate interface coupling conditions.

6. Interface learning can be applied to enforce consistent flow conditions between successive nested solvers. Examples include numerical weather prediction models featuring local models for specific region with increased levels of details and global models covering larger portions of the globe.

7. Optimal design of experiments studies are required to optimize the experimental configuration and sensor placements to decrease the costs of data collection and improve the quality of the inference algorithm.

8. The portability of the proposed approaches onto edge computing devices can be explored to enable the digital twin capabilities to be integrated in-situ with the physical system.

References

[1] 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," *Physics of Fluids*, vol. 33, p. 091301, 2021.

[2] S. E. Ahmed, O. San, A. Rasheed, and T. Iliescu, "A long short-term memory embedding for hybrid uplifted reduced order models," *Physica D: Nonlinear Phenomena*, vol. 409, p. 132471, 2020.

[3] J. Yu, C. Yan, and M. Guo, "Non-intrusive reduced-order modeling for fluid problems: A brief review," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 233, no. 16, pp. 5896–5912, 2019.

[4] S. Fresca, L. Dede, and A. Manzoni, "A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized pdes," *Journal of Scientific Computing*, vol. 87, no. 2, pp. 1–36, 2021.

[5] B. C. Csáji, "Approximation with artificial neural networks," *Faculty of Sciences, Eötvös Loránd University, Hungary*, vol. 24, no. 48, p. 7, 2001.

[6] J. S. Hesthaven and S. Ubbiali, "Non-intrusive reduced order modeling of nonlinear problems using neural networks," *Journal of Computational Physics*, vol. 363, pp. 55–78, 2018.

[7] Q. Wang, J. S. Hesthaven, and D. Ray, "Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem," *Journal of Computational Physics*, vol. 384, pp. 289–307, 2019.

[8] P. Wu, J. Sun, X. Chang, W. Zhang, R. Arcucci, Y. Guo, and C. C. Pain, "Data-driven reduced order model with temporal convolutional neural network," *Computer Methods in Applied Mechanics and Engineering*, vol. 360, p. 112766, 2020.

[9] S. E. Ahmed, O. San, D. A. Bistrian, and I. M. Navon, "Sampling and resolution characteristics in reduced order models of shallow water equations: Intrusive vs nonintrusive," *International Journal for Numerical Methods in Fluids*, vol. 92, no. 8, pp. 992–1036, 2020.

[10] S. E. Ahmed, P. H. Dabaghian, O. San, D. A. Bistrian, and I. M. Navon, "Dynamic mode decomposition with core sketch," *Physics of Fluids*, vol. 34, no. 6, p. 066603, 2022.

[11] S. E. Ahmed, S. Pawar, and O. San, "PyDA: A hands-on introduction to dynamical data assimilation with Python," *Fluids*, vol. 5, no. 4, p. 225, 2020.

[12] V. Singh and K. E. Willcox, "Engineering design with digital thread," *AIAA Journal*, vol. 56, no. 11, pp. 4515–4528, 2018.

[13] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21 980–22 012, 2020.

[14] R. Arcucci, L. Mottet, C. Pain, and Y.-K. Guo, "Optimal reduced space for variational data assimilation," *Journal of Computational Physics*, vol. 379, pp. 51–69, 2019.

[15] Z. Bai, "Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems," *Applied Numerical Mathematics*, vol. 43, no. 1-2, pp. 9–44, 2002.

[16] D. J. Lucia, P. S. Beran, and W. A. Silva, "Reduced-order modeling: New approaches for computational physics," *Progress in Aerospace Sciences*, vol. 40, no. 1-2, pp. 51–117, 2004.

[17] M. Hess, A. Alla, A. Quaini, G. Rozza, and M. Gunzburger, "A localized reduced-order modeling approach for PDEs with bifurcating solutions," *Computer Methods in Applied Mechanics and Engineering*, vol. 351, pp. 379–403, 2019.

[18] B. Kramer and K. E. Willcox, "Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition," *AIAA Journal*, vol. 57, no. 6, pp. 2297–2307, 2019.

[19] R. Swischuk, L. Mainini, B. Peherstorfer, and K. Willcox, "Projection-based model reduction: Formulations for physics-based machine learning," *Computers & Fluids*, vol. 179, pp. 704–717, 2019.

[20] M. Korda, M. Putinar, and I. Mezić, "Data-driven spectral analysis of the Koopman operator," *Applied and Computational Harmonic Analysis*, vol. 48, no. 2, pp. 599–629, 2020.

[21] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[22] 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, Berlin, 2018, pp. 167–179.

[23] S. Peitz, S. Ober-Blöbaum, and M. Dellnitz, "Multiobjective optimal control methods for the Navier-Stokes equations using reduced order modeling," *Acta Applicandae Mathematicae*, vol. 161, no. 1, pp. 171–199, 2019.

[24] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley, *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge University Press, New York, 2012.

[25] 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 Journal*, vol. 55, pp. 4013–4041, 2017.

[26] 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 Journal*, pp. 1–25, 2019.

[27] B. R. Noack, M. Morzynski, and G. Tadmor, *Reduced-Order Modelling for Flow Control*. Springer-Verlag, Berlin, 2011, vol. 528.

[28] C. W. Rowley and S. T. M. Dawson, "Model reduction for flow analysis and control," *Annual Review of Fluid Mechanics*, vol. 49, pp. 387–417, 2017.

[29] N. J. Nair and M. Balajewicz, "Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent

shocks," *International Journal for Numerical Methods in Engineering*, vol. 117, no. 12, pp. 1234–1262, 2019.

[30] E. Kaiser, B. R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Östh, S. Krajnović, and R. K. Niven, "Cluster-based reduced-order modelling of a mixing layer," *Journal of Fluid Mechanics*, vol. 754, pp. 365–414, 2014.

[31] B. Haasdonk, M. Dihlmann, and M. Ohlberger, "A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 17, no. 4, pp. 423–442, 2011.

[32] M. A. Dihlmann and B. Haasdonk, "Certified PDE-constrained parameter optimization using reduced basis surrogate models for evolution problems," *Computational Optimization and Applications*, vol. 60, no. 3, pp. 753–787, 2015.

[33] K. Ito and S. Ravindran, "A reduced-order method for simulation and control of fluid flows," *Journal of Computational Physics*, vol. 143, no. 2, pp. 403–425, 1998.

[34] A. Iollo, S. Lanteri, and J.-A. Désidéri, "Stability properties of POD–Galerkin approximations for the compressible Navier–Stokes equations," *Theoretical and Computational Fluid Dynamics*, vol. 13, no. 6, pp. 377–396, 2000.

[35] C. W. Rowley, T. Colonius, and R. M. Murray, "Model reduction for compressible flows using POD and Galerkin projection," *Physica D: Nonlinear Phenomena*, vol. 189, no. 1-2, pp. 115–129, 2004.

[36] R. Milk, S. Rave, and F. Schindler, "pyMOR–generic algorithms and interfaces for model order reduction," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. S194–S216, 2016.

[37] V. Puzyrev, M. Ghommem, and S. Meka, "pyROM: A computational framework for reduced order modeling," *Journal of Computational Science*, vol. 30, pp. 157–173, 2019.

[38] M. Bergmann, C.-H. Bruneau, and A. Iollo, "Enablers for robust POD models," *Journal of Computational Physics*, vol. 228, no. 2, pp. 516–538, 2009.

[39] M. Couplet, C. Basdevant, and P. Sagaut, "Calibrated reduced-order POD-Galerkin system for fluid flow modelling," *Journal of Computational Physics*, vol. 207, no. 1, pp. 192–220, 2005.

[40] K. Kunisch and S. Volkwein, "Galerkin proper orthogonal decomposition methods for parabolic problems," *Numerische Mathematik*, vol. 90, no. 1, pp. 117–148, 2001.

[41] A. Kolmogoroff, "Über die beste Annäherung von Funktionen einer gegebenen Funktionenklasse," *Annals of Mathematics*, vol. 37, no. 1, pp. 107–110, 1936.

[42] A. Pinkus, *N-widths in approximation theory*. Springer-Verlag, Berlin, 1985, vol. 7.

[43] S. E. Ahmed and O. San, "Breaking the Kolmogorov barrier in model reduction of fluid flows," *Fluids*, vol. 5, no. 1, p. 26, 2020.

[44] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza, "Model order reduction in fluid dynamics: Challenges and perspectives," in *Reduced Order Methods for Modeling and Computational Reduction*. Springer, Berlin, 2014, pp. 235–273.

[45] D. Rempfer, "On low-dimensional Galerkin models for fluid flow," *Theoretical and Computational Fluid Dynamics*, vol. 14, no. 2, pp. 75–88, 2000.

[46] B. R. Noack, K. Afanasiev, M. Moryński, G. Tadmor, and F. Thiele, "A hierarchy of low-dimensional models for the transient and post-transient cylinder wake," *Journal of Fluid Mechanics*, vol. 497, pp. 335–363, 2003.

[47] S. M. Rahman, S. E. Ahmed, and O. San, "A dynamic closure modeling framework for model order reduction of geophysical flows," *Physics of Fluids*, vol. 31, no. 4, p. 046602, 2019.

[48] S. Sirisup and G. E. Karniadakis, "A spectral viscosity method for correcting the long-term behavior of POD models," *Journal of Computational Physics*, vol. 194, no. 1, pp. 92–116, 2004.

[49] O. San and T. Iliescu, "Proper orthogonal decomposition closure models for fluid flows: Burgers equation," *International Journal of Numerical Analysis and Modeling, Series B*, vol. 5, no. 3, pp. 285–305, 2014.

[50] O. San and J. Borggaard, "Basis selection and closure for POD models of convection dominated Boussinesq flows," in *21st International Symposium on Mathematical Theory of Networks and Systems*, vol. 5, no. 3, 2014.

[51] B. Protas, B. R. Noack, and J. Östh, "Optimal nonlinear eddy viscosity in Galerkin models of turbulent flows," *Journal of Fluid Mechanics*, vol. 766, pp. 337–367, 2015.

[52] L. Cordier, B. R. Noack, G. Tissot, G. Lehnasch, J. Delville, M. Balajewicz, G. Daviller, and R. K. Niven, "Identification strategies for model-based control," *Experiments in Fluids*, vol. 54, no. 8, p. 1580, 2013.

[53] J. Östh, B. R. Noack, S. Krajnović, D. Barros, and J. Borée, "On the need for a nonlinear subscale turbulence term in POD models as exemplified for a high-Reynolds-number flow over an Ahmed body," *Journal of Fluid Mechanics*, vol. 747, pp. 518–544, 2014.

[54] M. Couplet, P. Sagaut, and C. Basdevant, "Intermodal energy transfers in a proper orthogonal decomposition-Galerkin representation of a turbulent separated flow," *Journal of Fluid Mechanics*, vol. 491, p. 275, 2003.

[55] V. L. Kalb and A. E. Deane, "An intrinsic stabilization scheme for proper orthogonal decomposition based low-dimensional models," *Physics of Fluids*, vol. 19, no. 5, p. 054106, 2007.

[56] I. Kalashnikova and M. F. Barone, "On the stability and convergence of a Galerkin reduced order model (ROM) of compressible flow with solid wall and far-field boundary treatment," *International Journal for Numerical Methods in Engineering*, vol. 83, no. 10, pp. 1345–1375, 2010.

[57] X. Xie, M. Mohebujjaman, L. G. Rebholz, and T. Iliescu, "Data-driven filtered reduced order modeling of fluid flows," *SIAM Journal on Scientific Computing*, vol. 40, no. 3, pp. B834–B857, 2018.

[58] M. Mohebujjaman, L. G. Rebholz, and T. Iliescu, "Physically-constrained data-driven correction for reduced order modeling of fluid flows," *International Journal for Numerical Methods in Fluids*, vol. 89, no. 3, pp. 103–122, 2019.

[59] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu, "Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison," *Comput. Meth. Appl. Mech. Eng.*, vol. 237-240, pp. 10–26, 2012.

[60] I. Akhtar, Z. Wang, J. Borggaard, and T. Iliescu, "A new closure strategy for proper orthogonal decomposition reduced-order models," *Journal of Computational and Nonlinear Dynamics*, vol. 7, no. 3, 2012.

[61] M. Balajewicz and E. H. Dowell, "Stabilization of projection-based reduced order models of the Navier–Stokes," *Nonlinear Dynamics*, vol. 70, no. 2, pp. 1619–1632, 2012.

[62] D. Amsallem and C. Farhat, "Stabilization of projection-based reduced-order models," *International Journal for Numerical Methods in Engineering*, vol. 91, no. 4, pp. 358–377, 2012.

[63] O. San and T. Iliescu, "A stabilized proper orthogonal decomposition reduced-order model for large scale quasigeostrophic ocean circulation," *Advances in Computational Mathematics*, vol. 41, pp. 1289–1319, 2015.

[64] M. Gunzburger, T. Iliescu, M. Mohebujjaman, and M. Schneier, "An evolve-filter-relax stabilized reduced order stochastic collocation method for the time-dependent Navier–Stokes equations," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 7, no. 4, pp. 1162–1184, 2019.

[65] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[66] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Computation*, vol. 12, pp. 2451–2471, 1999.

[67] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine learning for fluid mechanics," *Annual Review of Fluid Mechanics*, vol. 52, pp. 477–508, 2020.

[68] J. N. Kutz, "Deep learning in fluid dynamics," *Journal of Fluid Mechanics*, vol. 814, pp. 1–4, 2017.

[69] P. A. Durbin, "Some recent developments in turbulence closure modeling," *Annual Review of Fluid Mechanics*, vol. 50, pp. 77–103, 2018.

[70] K. Duraisamy, G. Iaccarino, and H. Xiao, "Turbulence modeling in the age of data," *Annual Review of Fluid Mechanics*, no. 51, pp. 357–377, 2019.

[71] J. C. B. Gamboa, "Deep learning for time-series analysis," *arXiv preprint arXiv:1701.01887*, 2017.

[72] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature Communications*, vol. 9, no. 1, p. 4950, 2018.

[73] S. E. Otto and C. W. Rowley, "Linearly recurrent autoencoder networks for learning dynamics," *SIAM Journal on Applied Dynamical Systems*, vol. 18, no. 1, pp. 558–593, 2019.

[74] K. Lee and K. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," *Journal of Computational Physics*, vol. 404, p. 108973, 2020.

[75] J. Pathak, A. Wikner, R. Fussell, S. Chandra, B. R. Hunt, M. Girvan, and E. Ott, "Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 4, p. 041101, 2018.

[76] K. Yeo and I. Melnyk, "Deep learning algorithm for data-driven simulation of noisy dynamical system," *Journal of Computational Physics*, vol. 376, pp. 1212–1231, 2019.

[77] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[78] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[79] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, "Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2213, p. 20170844, 2018.

[80] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth annual conference of the international speech communication association*, 2014.

[81] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, and T. Sapsis, "Data-assisted reduced-order modeling of extreme events in complex dynamical systems," *PLoS ONE*, vol. 13, no. 5, p. e0197704, 2018.

[82] S. Pawar, S. E. Ahmed, O. San, and A. Rasheed, "Data-driven recovery of hidden physics in reduced order modeling of fluid flows," *Physics of Fluids*, vol. 32, no. 3, p. 036602, 2020.

[83] L. Sirovich, "Turbulence and the dynamics of coherent structures. I-Coherent structures. II-Symmetries and transformations. III-Dynamics and scaling," *Quarterly of Applied Mathematics*, vol. 45, pp. 561–571, 1987.

[84] G. Berkooz, P. Holmes, and J. L. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows," *Annual Review of Fluid Mechanics*, vol. 25, no. 1, pp. 539–575, 1993.

[85] A. Chatterjee, "An introduction to the proper orthogonal decomposition," *Current Science*, vol. 78, pp. 808–817, 2000.

[86] M. Rathinam and L. R. Petzold, "A new look at proper orthogonal decomposition," *SIAM Journal on Numerical Analysis*, vol. 41, no. 5, pp. 1893–1925, 2003.

[87] L. N. Trefethen and D. Bau III, *Numerical linear algebra*. SIAM, Philadelphia, 1997, vol. 50.

[88] D. Amsallem and C. Farhat, "Interpolation method for adapting reduced-order models and application to aeroelasticity," *AIAA Journal*, vol. 46, no. 7, pp. 1803–1813, 2008.

[89] D. Amsallem, J. Cortial, K. Carlberg, and C. Farhat, "A method for interpolating on manifolds structural dynamics reduced-order models," *International Journal for Numerical Methods in Engineering*, vol. 80, no. 9, pp. 1241–1258, 2009.

[90] R. Zimmermann, B. Peherstorfer, and K. Willcox, "Geometric subspace updates with applications to online adaptive nonlinear model reduction," *SIAM Journal on Matrix Analysis and Applications*, vol. 39, no. 1, pp. 234–261, 2018.

[91] R. Zimmermann, "Manifold interpolation and model reduction," *arXiv preprint arXiv:1902.06502*, 2019.

[92] M. Oulghelou and C. Allery, "A fast and robust sub-optimal control approach using reduced order model adaptation techniques," *Applied Mathematics and Computation*, vol. 333, pp. 416–434, 2018.

[93] ——, "Non intrusive method for parametric model order reduction using a bi-calibrated interpolation on the grassmann manifold," *Journal of Computational Physics*, vol. 426, p. 109924, 2021.

[94] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.

[95] P.-A. Absil, R. Mahony, and R. Sepulchre, "Riemannian geometry of grassmann manifolds with a view on algorithmic computation," *Acta Applicandae Mathematica*, vol. 80, no. 2, pp. 199–220, 2004.

[96] K. Carlberg, C. Bou-Mosleh, and C. Farhat, "Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 2, pp. 155–181, 2011.

[97] K. Carlberg, M. Barone, and H. Antil, "Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction," *Journal of Computational Physics*, vol. 330, pp. 693–734, 2017.

[98] Y. Choi and K. Carlberg, "Space–time least-squares Petrov–Galerkin projection for nonlinear model reduction," *SIAM Journal on Scientific Computing*, vol. 41, no. 1, pp. A26–A58, 2019.

[99] D. Xiao, F. Fang, J. Du, C. C. Pain, I. M. Navon, A. G. Buchan, A. H. Elsheikh, and G. Hu, "Non-linear Petrov–Galerkin methods for reduced order modelling of the Navier–Stokes equations using a mixed finite element pair," *Computer Methods In Applied Mechanics and Engineering*, vol. 255, pp. 147–157, 2013.

[100] D. Amsallem, "Interpolation on manifolds of CFD-based fluid and finite element-based structural reduced-order models for on-line aeroelastic predictions," Ph.D. dissertation, Stanford University, 2010.

[101] R. Ştefănescu, A. Sandu, and I. M. Navon, "Comparison of POD reduced order strategies for the nonlinear 2D shallow water equations," *International Journal for Numerical Methods in Fluids*, vol. 76, no. 8, pp. 497–521, 2014.

[102] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations," *C. R. Acad. Sci. Paris, Ser. I*, vol. 339, pp. 667–672, 2004.

[103] S. Chaturantabut and D. C. Sorensen, "Nonlinear model reduction via discrete empirical interpolation," *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.

[104] S. E. Ahmed, S. M. Rahman, O. San, A. Rasheed, and I. M. Navon, "Memory embedded non-intrusive reduced order modeling of non-ergodic flows," *Physics of Fluids*, vol. 31, no. 12, p. 126602, 2019.

[105] M. Dihlmann, M. Drohmann, and B. Haasdonk, "Model reduction of parametrized evolution problems using the reduced basis method with adaptive time-partitioning," *Proc. of ADMOS*, vol. 2011, p. 64, 2011.

[106] R. Maulik, B. Lusch, and P. Balaprakash, "Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders," *Physics of Fluids*, vol. 33, no. 3, p. 037106, 2021.

[107] K. Lee and K. T. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," *Journal of Computational Physics*, vol. 404, p. 108973, 2020.

[108] S. Grimberg, C. Farhat, and N. Youkilis, "On the stability of projection-based model order reduction for convection-dominated laminar and turbulent flows," *Journal of Computational Physics*, vol. 419, p. 109681, 2020.

[109] J. N. Kani and A. H. Elsheikh, "Reduced-order modeling of subsurface multiphase flow models using deep residual recurrent neural networks," *Transport in Porous Media*, vol. 126, no. 3, pp. 713–741, 2019.

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

[111] Z. Wang, D. Xiao, F. Fang, R. Govindan, C. C. Pain, and Y. Guo, "Model identification of reduced order fluid dynamics systems using deep learning," *International Journal for Numerical Methods in Fluids*, vol. 86, no. 4, pp. 255–268, 2018.

[112] S. M. Rahman, S. Pawar, O. San, A. Rasheed, and T. Iliescu, "Nonintrusive reduced order modeling framework for quasigeostrophic turbulence," *Physical Review E*, vol. 100, p. 053306, 2019.

[113] S. Wiewel, M. Becher, and N. Thuerey, "Latent space physics: Towards learning the temporal evolution of fluid flow," in *Computer Graphics Forum*, vol. 38, no. 2. Wiley Online Library, 2019, pp. 71–82.

[114] D. Xiao, F. Fang, J. Zheng, C. Pain, and I. Navon, "Machine learning-based rapid response tools for regional air pollution modelling," *Atmospheric Environment*, vol. 199, pp. 463–473, 2019.

[115] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[116] M. Maleewong and S. Sirisup, "On-line and off-line POD assisted projective integral for non-linear problems: A case study with Burgers' equation," *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering*, vol. 5, no. 7, pp. 984–992, 2011.

[117] J. D. Buntine and D. Pullin, "Merger and cancellation of strained vortices," *Journal of Fluid Mechanics*, vol. 205, pp. 263–295, 1989.

[118] J. Von Hardenberg, J. McWilliams, A. Provenzale, A. Shchepetkin, and J. Weiss, "Vortex merging in quasi-geostrophic flows," *Journal of Fluid Mechanics*, vol. 412, pp. 331–353, 2000.

[119] A. Arakawa, "Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I," *Journal of Computational Physics*, vol. 1, no. 1, pp. 119–143, 1966.

[120] S. Gottlieb and C.-W. Shu, "Total variation diminishing Runge-Kutta schemes," *Mathematics of Computation*, vol. 67, no. 221, pp. 73–85, 1998.

[121] S. S. Ravindran, "A reduced-order approach for optimal control of fluids using proper orthogonal decomposition," *International Journal for Numerical Methods in Fluids*, vol. 34, no. 5, pp. 425–448, 2000.

[122] 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*, vol. 23, no. 3, pp. 1695–1706, 2009.

[123] M. J. Zahr and C. Farhat, "Progressive construction of a parametric reduced-order model for PDE-constrained optimization," *International Journal for Numerical Methods in Engineering*, vol. 102, no. 5, pp. 1111–1135, 2015.

[124] J. Degroote, J. Vierendeels, and K. Willcox, "Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis," *International Journal for Numerical Methods in Fluids*, vol. 63, no. 2, pp. 207–230, 2010.

[125] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat, "Design optimization using hyper-reduced-order models," *Structural and Multidisciplinary Optimization*, vol. 51, no. 4, pp. 919–940, 2015.

[126] P. Benner, E. Sachs, and S. Volkwein, "Model order reduction for PDE constrained optimization," *Trends in PDE constrained optimization*, pp. 303–326, 2014.

[127] T. Bui-Thanh, K. Willcox, O. Ghattas, and B. van Bloemen Waanders, "Goal-oriented, model-constrained optimization for reduction of large-scale systems," *Journal of Computational Physics*, vol. 224, no. 2, pp. 880–896, 2007.

[128] B. Peherstorfer, K. Willcox, and M. Gunzburger, "Survey of multifidelity methods in uncertainty propagation, inference, and optimization," *SIAM Review*, vol. 60, no. 3, pp. 550–591, 2018.

[129] M. G. Kapteyn, J. V. Pretorius, and K. E. Willcox, "A probabilistic graphical model foundation for enabling predictive digital twins at scale," *Nature Computational Science*, vol. 1, no. 5, pp. 337–347, 2021.

[130] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2018.

[131] S. Haag and R. Anderl, "Digital twin–proof of concept," *Manufacturing Letters*, vol. 15, pp. 64–66, 2018.

[132] S. Boschert and R. Rosen, "Digital twin—the simulation aspect," in *Mechatronic Futures*. Springer, 2016, pp. 59–74.

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

[134] 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 Journal*, vol. 58, no. 3, pp. 998–1022, 2020.

[135] I. Akhtar, A. H. Nayfeh, and C. J. Ribbens, "On the stability and extension of reduced-order Galerkin models in incompressible flows," *Theoretical and Computational Fluid Dynamics*, vol. 23, no. 3, pp. 213–237, 2009.

[136] E. W. Sachs and S. Volkwein, "POD-Galerkin approximations in PDE-constrained optimization," *GAMM-Mitteilungen*, vol. 33, no. 2, pp. 194–208, 2010.

[137] 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*, vol. 30, no. 5, p. 055006, 2014.

[138] H. Yang and A. Veneziani, "Efficient estimation of cardiac conductivities via POD-DEIM model order reduction," *Applied Numerical Mathematics*, vol. 115, pp. 180–199, 2017.

[139] S. Hijazi, G. Stabile, A. Mola, and G. Rozza, "Data-driven POD-Galerkin reduced order model for turbulent flows," *Journal of Computational Physics*, vol. 416, p. 109513, 2020.

[140] M. Girfoglio, A. Quaini, and G. Rozza, "A POD-Galerkin reduced order model for a LES filtering approach," *Journal of Computational Physics*, vol. 436, p. 110260, 2021.

[141] B. Peherstorfer, "Breaking the Kolmogorov barrier with nonlinear model reduction," *Notices of the American Mathematical Society*, vol. 69, no. 5, 2022.

[142] J. R. Singler, "New POD error expressions, error bounds, and asymptotic results for reduced order models of parabolic PDEs," *SIAM Journal on Numerical Analysis*, vol. 52, no. 2, pp. 852–876, 2014.

[143] D. Amsallem and B. Haasdonk, "PEBL-ROM: Projection-error based local reduced-order models," *Advanced Modeling and Simulation in Engineering Sciences*, vol. 3, no. 1, pp. 1–25, 2016.

[144] T. P. Sapsis and A. J. Majda, "Blending modified gaussian closure and non-gaussian reduced subspace methods for turbulent dynamical systems," *Journal of Nonlinear Science*, vol. 23, no. 6, pp. 1039–1071, 2013.

[145] O. San and R. Maulik, "Extreme learning machine for reduced order modeling of turbulent geophysical flows," *Physical Review E*, vol. 97, no. 4, p. 042322, 2018.

[146] S. Pan and K. Duraisamy, "Data-driven discovery of closure models," *SIAM Journal on Applied Dynamical Systems*, vol. 17, no. 4, pp. 2381–2413, 2018.

[147] H. Imtiaz and I. Akhtar, "Nonlinear closure modeling in reduced order models for turbulent flows: A dynamical system approach," *Nonlinear Dynamics*, vol. 99, no. 1, pp. 479–494, 2020.

[148] 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*, 2020, p. 2946.

[149] A. Gupta and P. F. Lermusiaux, "Neural closure models for dynamical systems," *Proceedings of the Royal Society A*, vol. 477, no. 2252, p. 20201004, 2021.

[150] T. J. R. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy, "The variational multiscale method – a paradigm for computational mechanics," *Com-*

puter Methods in Applied Mechanics and Engineering, vol. 166, no. 1, pp. 3–24, 1998.

[151] T. J. Hughes, L. Mazzei, and K. E. Jansen, "Large eddy simulation and the variational multiscale method," *Computing and Visualization in Science*, vol. 3, no. 1-2, pp. 47–59, 2000.

[152] T. J. Hughes, A. A. Oberai, and L. Mazzei, "Large eddy simulation of turbulent channel flows by the variational multiscale method," *Physics of Fluids*, vol. 13, no. 6, pp. 1784–1799, 2001.

[153] R. Codina, S. Badia, J. Baiges, and J. Principe, "Variational multiscale methods in computational fluid dynamics," *Encyclopedia of Computational Mechanics Second Edition*, pp. 1–28, 2018.

[154] V. John, *Finite element methods for incompressible flow problems*. Springer, Berlin, 2016.

[155] T. Iliescu and Z. Wang, "Variational multiscale proper orthogonal decomposition: Convection-dominated convection-diffusion-reaction equations," *Mathematics of Computation*, vol. 82, no. 283, pp. 1357–1378, 2013.

[156] ——, "Variational multiscale proper orthogonal decomposition: Navier-stokes equations," *Numerical Methods for Partial Differential Equations*, vol. 30, no. 2, pp. 641–663, 2014.

[157] C. Mou, B. Koc, O. San, L. G. Rebholz, and T. Iliescu, "Data-driven variational multiscale reduced order models," *Computer Methods in Applied Mechanics and Engineering*, vol. 373, p. 113470, 2021.

[158] B. Koc, C. Mou, H. Liu, Z. Wang, G. Rozza, and T. Iliescu, "Verifiability of the data-driven variational multiscale reduced order model," *arXiv preprint arXiv:2108.04982*, 2021.

[159] H. Mori, "Transport, collective motion, and brownian motion," *Progress of Theoretical Physics*, vol. 33, no. 3, pp. 423–455, 1965.

[160] R. Zwanzig, "Problems in nonlinear transport theory," in *Systems far from equilibrium*. Springer, Berlin, 1980, pp. 198–225.

[161] A. J. Chorin, O. H. Hald, and R. Kupferman, "Optimal prediction and the Mori–Zwanzig representation of irreversible processes," *Proceedings of the National Academy of Sciences*, vol. 97, no. 7, pp. 2968–2973, 2000.

[162] ——, "Optimal prediction with memory," *Physica D: Nonlinear Phenomena*, vol. 166, no. 3-4, pp. 239–257, 2002.

[163] A. J. Chorin and O. H. Hald, *Stochastic tools in mathematics and science*. Springer-Verlag, New York, 2009, vol. 1.

[164] S. Pawar, O. San, B. Aksoylu, A. Rasheed, and T. Kvamsdal, "Physics guided machine learning using simplified theories," *Physics of Fluids*, vol. 33, no. 1, p. 011701, 2021.

[165] S. Pawar, O. San, A. Nair, A. Rasheed, and T. Kvamsdal, "Model fusion with physics-guided machine learning: Projection-based reduced-order modeling," *Physics of Fluids*, vol. 33, no. 6, p. 067123, 2021.

[166] S. Pawar, O. San, P. Vedula, A. Rasheed, and T. Kvamsdal, "Multi-fidelity information fusion with concatenated neural networks," *Scientific Reports*, vol. 12, no. 1, pp. 1–13, 2022.

[167] S. E. Ahmed, O. San, A. Rasheed, and T. Iliescu, "Nonlinear proper orthogonal decomposition for convection-dominated flows," *Physics of Fluids*, vol. 33, no. 12, p. 121702, 2021.

[168] S. Volkwein, "Proper orthogonal decomposition: Theory and reduced-order modelling," *Lecture Notes, University of Konstanz*, 2013, http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/POD-Book.pdf.

[169] L. Sirovich, "Turbulence and the dynamics of coherent structures. i. coherent structures," *Quarterly of applied mathematics*, vol. 45, no. 3, pp. 561–571, 1987.

[170] ——, "Turbulence and the dynamics of coherent structures. ii. symmetries and transformations," *Quarterly of Applied mathematics*, vol. 45, no. 3, pp. 573–582, 1987.

[171] ——, "Turbulence and the dynamics of coherent structures. iii. dynamics and scaling," *Quarterly of Applied mathematics*, vol. 45, no. 3, pp. 583–590, 1987.

[172] 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, 2002, pp. 193–200.

[173] 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.

[174] 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 Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[175] P. Stinis, "Renormalized Mori–Zwanzig-reduced models for systems without scale separation," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 471, no. 2176, p. 20140446, 2015.

[176] R. Zwanzig, *Nonequilibrium statistical mechanics*. Oxford University Press, Oxford, 2001.

[177] 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," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2205, p. 20170385, 2017.

[178] 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," *Journal of Computational Physics*, vol. 410, p. 109402, 2020.

[179] O. San and A. E. Staples, "A coarse-grid projection method for accelerating incompressible flow computations," *Journal of Computational Physics*, vol. 233, pp. 480–508, 2013.

[180] A. J. Roberts, "Boundary conditions for approximate differential equations," *The ANZIAM Journal*, vol. 34, no. 1, pp. 54–80, 1992.

[181] F. Naets, D. De Gregoriis, and W. Desmet, "Multi-expansion modal reduction: A pragmatic semi–a priori model order reduction approach for nonlinear structural dynamics," *International Journal for Numerical Methods in Engineering*, vol. 118, no. 13, pp. 765–782, 2019.

[182] D. E. Keyes, L. C. McInnes, C. Woodward, W. Gropp, E. Myra, M. Pernice, J. Bell, J. Brown, A. Clo, and J. Connors, "Multiphysics simulations: Challenges and opportunities," *The International Journal of High Performance Computing Applications*, vol. 27, no. 1, pp. 4–83, 2013.

[183] S. Shankaran, J. Alonso, M.-F. Liou, N.-S. Liu, and R. Davis, "A multi-code-coupling interface for combustor/turbomachinery simulations," in *39th Aerospace Sciences Meeting and Exhibit*, 2001, p. 974.

[184] J. Xu, C. Huang, and K. Duraisamy, "Reduced-order modeling framework for combustor instabilities using truncated domain training," *AIAA Journal*, vol. 58, no. 2, pp. 618–632, 2020.

[185] C. Chen, A. J. Roberts, and J. E. Bunder, "Boundary conditions for macroscale waves in an elastic system with microscale heterogeneity," *IMA Journal of Applied Mathematics*, vol. 83, no. 3, pp. 347–379, 2018.

[186] S. T. O'connell and P. A. Thompson, "Molecular dynamics–continuum hybrid computations: A tool for studying complex fluid flows," *Physical Review E*, vol. 52, no. 6, p. R5792, 1995.

[187] S. Pawar, S. E. Ahmed, and O. San, "Interface learning in fluid dynamics: Statistical inference of closures within micro–macro-coupling models," *Physics of Fluids*, vol. 32, no. 9, p. 091704, 2020.

[188] A. Quarteroni and A. Veneziani, "Analysis of a geometrical multiscale model based on the coupling of ODE and PDE for blood flow simulations," *Multiscale Modeling & Simulation*, vol. 1, no. 2, pp. 173–195, 2003.

[189] T. Passerini, M. De Luca, L. Formaggia, A. Quarteroni, and A. Veneziani, "A 3D/1D geometrical multiscale model of cerebral vasculature," *Journal of Engineering Mathematics*, vol. 64, no. 4, p. 319, 2009.

[190] A. Quarteroni, A. Veneziani, and C. Vergara, "Geometric multiscale modeling of the cardiovascular system, between theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 302, pp. 193–252, 2016.

[191] J. Nordbotten and W. Boon, "Modeling, structure and discretization of hierarchical mixed-dimensional partial differential equations," in *International Conference on Domain Decomposition Methods*. Springer, 2017, pp. 87–101.

[192] W. M. Boon, J. M. Nordbotten, and I. Yotov, "Robust discretization of flow in fractured porous media," *SIAM Journal on Numerical Analysis*, vol. 56, no. 4, pp. 2203–2233, 2018.

[193] J. M. Nordbotten, W. M. Boon, A. Fumagalli, and E. Keilegavlen, "Unified approach to discretization of flow in fractured porous media," *Computational Geosciences*, vol. 23, no. 2, pp. 225–237, 2019.

[194] P. Sagaut, *Multiscale and multiresolution approaches in turbulence: LES, DES and hybrid RANS/LES methods: Applications and guidelines*. Imperial College Press, Danvers, MA, USA, 2013.

[195] A. Fadai-Ghotbi, C. Friess, R. Manceau, and J. Borée, "A seamless hybrid RANS-LES model based on transport equations for the subgrid stresses and elliptic blending," *Physics of Fluids*, vol. 22, no. 5, p. 055104, 2010.

[196] M. L. Shur, P. R. Spalart, M. K. Strelets, and A. K. Travin, "A hybrid RANS-LES approach with delayed-DES and wall-modelled LES capabilities," *International Journal of Heat and Fluid Flow*, vol. 29, no. 6, pp. 1638–1649, 2008.

[197] R. Maulik, O. San, J. D. Jacob, and C. Crick, "Sub-grid scale model classification and blending through deep learning," *Journal of Fluid Mechanics*, vol. 870, pp. 784–812, 2019.

[198] K. M. Waldron, J. Paegle, and J. D. Horel, "Sensitivity of a spectrally filtered and nudged limited-area model to outer model options," *Monthly Weather Review*, vol. 124, no. 3, pp. 529–547, 1996.

[199] H. von Storch, H. Langenberg, and F. Feser, "A spectral nudging technique for dynamical downscaling purposes," *Monthly Weather Review*, vol. 128, no. 10, pp. 3664–3673, 2000.

[200] R. Radu, M. Déqué, and S. Somot, "Spectral nudging in a spectral regional climate model," *Tellus A: Dynamic Meteorology and Oceanography*, vol. 60, no. 5, pp. 898–910, 2008.

[201] G. Miguez-Macho, G. L. Stenchikov, and A. Robock, "Spectral nudging to eliminate the effects of domain position and geometry in regional climate model simulations," *Journal of Geophysical Research: Atmospheres*, vol. 109, no. D13, 2004.

[202] B. Rockel, C. L. Castro, R. A. Pielke Sr, H. von Storch, and G. Leoncini, "Dynamical downscaling: Assessment of model system dependent retained and added variability for two different regional climate models," *Journal of Geophysical Research: Atmospheres*, vol. 113, no. D21, 2008.

[203] M. Schubert-Frisius, F. Feser, H. von Storch, and S. Rast, "Optimal spectral nudging for global dynamic downscaling," *Monthly Weather Review*, vol. 145, no. 3, pp. 909–927, 2017.

[204] H. Tang, R. Haynes, and G. Houzeaux, "A review of domain decomposition methods for simulation of fluid flows: Concepts, algorithms, and applications," *Archives of Computational Methods in Engineering*, pp. 1–33, 2020.

[205] M. J. Gander and L. Halpern, "Techniques for locally adaptive time stepping developed over the last two decades," in *Domain Decomposition Methods in Science and Engineering XX*. Springer, Berlin, 2013, pp. 377–385.

[206] D. A. Donzis and K. Aditya, "Asynchronous finite-difference schemes for partial differential equations," *Journal of Computational Physics*, vol. 274, pp. 370–392, 2014.

[207] A. Mittal and S. Girimaji, "Proxy-equation paradigm: A strategy for massively parallel asynchronous computations," *Physical Review E*, vol. 96, no. 3, p. 033304, 2017.

[208] A. Bowers, L. Rebholz, A. Takhirov, and C. Trenchea, "Improved accuracy in regularization models of incompressible flow via adaptive nonlinear filtering," *International Journal for Numerical Methods in Fluids*, vol. 70, no. 7, pp. 805–828, 2012.

[209] C. C. Manica, M. Neda, M. Olshanskii, and L. G. Rebholz, "Enabling numerical accuracy of navier-stokes-α through deconvolution and enhanced stability," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 45, no. 2, pp. 277–307, 2011.

[210] J. Borggaard and T. Iliescu, "Approximate deconvolution boundary conditions for large eddy simulation," *Applied Mathematics Letters*, vol. 19, no. 8, pp. 735–740, 2006.

[211] S. M. Rahman, S. Pawar, O. San, A. Rasheed, and T. Iliescu, "A nonintrusive reduced order modeling framework for quasigeostrophic turbulence," *Physical Review E*, vol. 100, p. 053306, 2019.

[212] G. A. Sod, "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws," *Journal of Computational Physics*, vol. 27, no. 1, pp. 1–31, 1978.

[213] N. Stergiopulos, D. Young, and T. Rogge, "Computer simulation of arterial flow with applications to arterial and aortic stenoses," *Journal of Biomechanics*, vol. 25, no. 12, pp. 1477–1488, 1992.

[214] G. Kissas, Y. Yang, E. Hwuang, W. R. Witschey, J. A. Detre, and P. Perdikaris, "Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 358, p. 112623, 2020.

[215] P. Garcia-Navarro and J. M. Saviron, "McCormack's method for the numerical simulation of one-dimensional discontinuous unsteady open channel flow," *Journal of Hydraulic Research*, vol. 30, no. 1, pp. 95–105, 1992.

[216] S. Abhyankar, G. Betrie, D. A. Maldonado, L. C. Mcinnes, B. Smith, and H. Zhang, "Petsc dmnetwork: A library for scalable network pde-based multiphysics simulations," *ACM Transactions on Mathematical Software (TOMS)*, vol. 46, no. 1, pp. 1–24, 2020.

[217] O. San and A. E. Staples, "An improved model for reduced-order physiological fluid flows," *Journal of Mechanics in Medicine and Biology*, vol. 12, no. 03, p. 1250052, 2012.

[218] K. S. Bjørkevoll, "Use of high fidelity models for real time status detection with field examples from automated MPD operations in the North Sea," in *The 2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*, 2015.

[219] J. Macpherson, "Technology focus: Drilling management and automation," *Journal of Petroleum Technology*, vol. 67, no. 09, pp. 136–136, 2015.

[220] W. C. Reynolds, D. E. Parekh, P. J. D. Juvet, and M. J. D. Lee, "Bifurcating and blooming jets," *Annual Review of Fluid Mechanics*, vol. 35, no. 1, pp. 295–315, 2003.

[221] A. Tyliszczak, "Multi-armed jets: A subset of the blooming jets," *Physics of Fluids*, vol. 27, no. 4, p. 041703, 2015.

[222] T. B. Gohil, A. K. Saha, and K. Muralidhar, "Simulation of the blooming phenomenon in forced circular jets," *Journal of Fluid Mechanics*, vol. 783, pp. 567–604, 2015.

[223] E. Phillips and I. Wygnanski, "Use of sweeping jets during transient deployment of a control surface," *AIAA Journal*, vol. 51, no. 4, pp. 819–828, 2013.

[224] M. Koklu, "Effect of a Coanda extension on the performance of a sweeping-jet actuator," *AIAA Journal*, vol. 54, no. 3, pp. 1131–1134, 2016.

[225] ——, "Effects of sweeping jet actuator parameters on flow separation control," *AIAA Journal*, vol. 56, no. 1, pp. 100–110, 2018.

[226] K. Kara, D. Kim, and P. J. Morris, "Flow-separation control using sweeping jet actuator," *AIAA Journal*, vol. 56, no. 11, pp. 4604–4613, 2018.

[227] R. E. Childs, P. M. Stremel, L. K. Kushner, J. T. Heineck, and B. L. Storms, "Simulation of sweep-jet flow control, single jet and full vertical tail," in *54th AIAA Aerospace Sciences Meeting*, 2016, p. 0569.

[228] S. Aram and H. Shan, "Synchronization effect of an array of sweeping jets on a separated flow over a wall-mounted hump," in *AIAA Aviation 2019 Forum*, 2019, p. 3396.

[229] C. D'Angelo and A. Quarteroni, "On the coupling of 1D and 3D diffusion-reaction equations: Application to tissue perfusion problems," *Mathematical Models and Methods in Applied Sciences*, vol. 18, no. 08, pp. 1481–1504, 2008.

[230] I. G. Kevrekidis and G. Samaey, "Equation-free multiscale computation: Algorithms and applications," *Annual Review of Physical Chemistry*, vol. 60, pp. 321–344, 2009.

[231] W. E, B. Engquist, and Z. Huang, "Heterogeneous multiscale method: A general methodology for multiscale modeling," *Physical Review B*, vol. 67, no. 9, p. 092101, 2003.

[232] E. Van Brummelen, "Partitioned iterative solution methods for fluid–structure interaction," *International Journal for Numerical Methods in Fluids*, vol. 65, no. 1-3, pp. 3–27, 2011.

[233] M. S. Siddiqui, A. Rasheed, M. Tabib, and T. Kvamsdal, "Numerical investigation of modeling frameworks and geometric approximations on NREL 5 MW wind turbine," *Renewable Energy*, vol. 132, pp. 1058–1075, 2019.

[234] R. Ganguli and S. Adhikari, "The digital twin of discrete dynamic systems: Initial approaches and future challenges," *Applied Mathematical Modelling*, vol. 77, pp. 1110–1128, 2020.

[235] M. G. Kapteyn, D. J. Knezevic, D. Huynh, M. Tran, and K. E. Willcox, "Data-driven physics-based digital twins via a library of component-based reduced-order models," *International Journal for Numerical Methods in Engineering*, 2020.

[236] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *The Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA, February 4—9*, 2017.

[237] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[238] Z. Liu, C. Yao, H. Yu, and T. Wu, "Deep reinforcement learning with its application for lung cancer detection in medical internet of things," *Future Generation Computer Systems*, vol. 97, pp. 1 – 9, 2019.

[239] F. Bu and X. Wang, "A smart agriculture IoT system based on deep reinforcement learning," *Future Generation Computer Systems*, vol. 99, pp. 500 – 507, 2019.

[240] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais *et al.*, "Deep learning and process understanding for data-driven Earth system science," *Nature*, vol. 566, no. 7743, pp. 195–204, 2019.

[241] J. Schmidt, M. R. Marques, S. Botti, and M. A. Marques, "Recent advances and applications of machine learning in solid-state materials science," *Computational Materials*, vol. 5, no. 1, pp. 1–36, 2019.

[242] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14 410–14 430, 2018.

[243] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[244] H. X. Y. M. Hao-Chen, L. D. Deb, H. L. J.-L. T. Anil, and K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151–178, 2020.

[245] S. Pawar, S. E. Ahmed, O. San, A. Rasheed, and I. M. Navon, "Long short-term memory embedded nudging schemes for nonlinear data assimilation of geophysical flows," *Physics of Fluids*, vol. 32, p. 076606, 2020.

[246] A. Majda, *Introduction to PDEs and Waves for the Atmosphere and Ocean*. American Mathematical Society, Providence, 2003, vol. 9.

[247] J.-G. Liu, C. Wang, and H. Johnston, "A fourth order scheme for incompressible Boussinesq equations," *Journal of Scientific Computing*, vol. 18, no. 2, pp. 253–285, 2003.

[248] A. Nicolás and B. Bermúdez, "2D thermal/isothermal incompressible viscous flows," *International Journal for Numerical Methods in Fluids*, vol. 48, no. 4, pp. 349–366, 2005.

[249] B. Soffientino and M. E. Q. Pilson, "The Bosporus Strait: A special place in the history of oceanography," *Oceanography*, vol. 18, no. 2, pp. 16–23, 2005.

[250] A. E. Gill, *Atmosphere-ocean dynamics*. Academic Press, 1982.

[251] A. Arakawa, "Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow." *Journal of Computational Physics*, vol. 135, no. 2, pp. 103–114, 1997.

[252] M. Ghil and P. Malanotte-Rizzoli, "Data assimilation in meteorology and oceanography," in *Advances in Geophysics*. Elsevier, 1991, vol. 33, pp. 141–266.

[253] E. Kalnay, *Atmospheric modeling, data assimilation and predictability*. Cambridge University Press, Cambridge, 2003.

[254] J. M. Lewis, S. Lakshmivarahan, and S. Dhall, *Dynamic data assimilation: A least squares approach*. Cambridge University Press, Cambridge, 2006, vol. 104.

[255] A. C. Lorenc, "Analysis methods for numerical weather prediction," *Quarterly Journal of the Royal Meteorological Society*, vol. 112, no. 474, pp. 1177–1194, 1986.

[256] D. F. Parrish and J. C. Derber, "The national meteorological center's spectral statistical-interpolation analysis system," *Monthly Weather Review*, vol. 120, no. 8, pp. 1747–1763, 1992.

[257] P. Courtier, "Dual formulation of four-dimensional variational assimilation," *Quarterly Journal of the Royal Meteorological Society*, vol. 123, no. 544, pp. 2449–2461, 1997.

[258] F. Rabier, H. Järvinen, E. Klinker, J.-F. Mahfouf, and A. Simmons, "The ECMWF operational implementation of four-dimensional variational assimilation. I: Experimental results with simplified physics," *Quarterly Journal of the Royal Meteorological Society*, vol. 126, no. 564, pp. 1143–1170, 2000.

[259] H. Elbern, H. Schmidt, O. Talagrand, and A. Ebel, "4D-variational data assimilation with an adjoint air quality model for emission analysis," *Environmental Modelling & Software*, vol. 15, no. 6-7, pp. 539–548, 2000.

[260] P. Courtier, J.-N. Thépaut, and A. Hollingsworth, "A strategy for operational implementation of 4D-Var, using an incremental approach," *Quarterly Journal of the Royal Meteorological Society*, vol. 120, no. 519, pp. 1367–1387, 1994.

[261] A. C. Lorenc and F. Rawlins, "Why does 4D-Var beat 3D-Var?" *Quarterly Journal of the Royal Meteorological Society*, vol. 131, no. 613, pp. 3247–3257, 2005.

[262] P. Gauthier, M. Tanguay, S. Laroche, S. Pellerin, and J. Morneau, "Extension of 3DVAR to 4DVAR: Implementation of 4dvar at the meteorological service of Canada," *Monthly Weather Review*, vol. 135, no. 6, pp. 2339–2354, 2007.

[263] P. L. Houtekamer and H. L. Mitchell, "Data assimilation using an ensemble Kalman filter technique," *Monthly Weather Review*, vol. 126, no. 3, pp. 796–811, 1998.

[264] G. Burgers, P. Jan van Leeuwen, and G. Evensen, "Analysis scheme in the ensemble Kalman filter," *Monthly Weather Review*, vol. 126, no. 6, pp. 1719–1724, 1998.

[265] G. Evensen, "The ensemble Kalman filter: Theoretical formulation and practical implementation," *Ocean Dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

[266] P. L. Houtekamer and H. L. Mitchell, "A sequential ensemble Kalman filter for atmospheric data assimilation," *Monthly Weather Review*, vol. 129, no. 1, pp. 123–137, 2001.

[267] ——, "Ensemble Kalman filtering," *Quarterly Journal of the Royal Meteorological Society*, vol. 131, no. 613, pp. 3269–3289, 2005.

[268] D. Treebushny and H. Madsen, "A new reduced rank square root Kalman filter for data assimilation in mathematical models," in *International Conference on Computational Science*. Springer, Berlin, 2003, pp. 482–491.

[269] M. Buehner and P. Malanotte-Rizzoli, "Reduced-rank Kalman filters applied to an idealized model of the wind-driven ocean circulation," *Journal of Geophysical Research: Oceans*, vol. 108, no. C6, 2003.

[270] S. Lakshmivarahan and D. J. Stensrud, "Ensemble Kalman filter," *IEEE Control Systems Magazine*, vol. 29, no. 3, pp. 34–46, 2009.

[271] M. Zupanski, "Maximum likelihood ensemble filter: Theoretical aspects," *Monthly Weather Review*, vol. 133, no. 6, pp. 1710–1726, 2005.

[272] G. Desroziers, J.-T. Camino, and L. Berre, "4DEnVar: Link with 4D state formulation of variational assimilation and different possible implementations," *Quarterly Journal of the Royal Meteorological Society*, vol. 140, no. 684, pp. 2097–2110, 2014.

[273] A. C. Lorenc, N. E. Bowler, A. M. Clayton, S. R. Pring, and D. Fairbairn, "Comparison of hybrid-4DEnVar and hybrid-4DVar data assimilation methods for global NWP," *Monthly Weather Review*, vol. 143, no. 1, pp. 212–229, 2015.

[274] X. Wang, D. M. Barker, C. Snyder, and T. M. Hamill, "A hybrid ETKF–3DVAR data assimilation scheme for the WRF model. Part I: Observing system simulation experiment," *Monthly Weather Review*, vol. 136, no. 12, pp. 5116–5131, 2008.

[275] M. Buehner, J. Morneau, and C. Charette, "Four-dimensional ensemble-variational data assimilation for global deterministic weather prediction." *Nonlinear Processes in Geophysics*, vol. 20, no. 5, 2013.

[276] D. T. Kleist and K. Ide, "An OSSE-based evaluation of hybrid variational–ensemble data assimilation for the NCEP GFS. Part I: System description and 3D-hybrid results," *Monthly Weather Review*, vol. 143, no. 2, pp. 433–451, 2015.

[277] ——, "An OSSE-based evaluation of hybrid variational–ensemble data assimilation for the NCEP GFS. Part II: 4DEnVar and hybrid variants," *Monthly Weather Review*, vol. 143, no. 2, pp. 452–470, 2015.

[278] S. Lakshmivarahan and J. M. Lewis, "Forward sensitivity approach to dynamic data assimilation," *Advances in Meteorology*, vol. 2010, p. 1–12, 2010.

[279] S. Lakshmivarahan, J. M. Lewis, and R. Jabrzemski, *Forecast error correction using dynamic data assimilation*. Springer, Switzerland, 2017.

[280] B. Wang, X. Zou, and J. Zhu, "Data assimilation and its applications," *Proceedings of the National Academy of Sciences*, vol. 97, no. 21, pp. 11 143–11 144, 2000.

[281] S. I. Aanonsen, G. Nævdal, D. S. Oliver, A. C. Reynolds, B. Vallès *et al.*, "The ensemble Kalman filter in reservoir engineering–a review," *SPE Journal*, vol. 14, no. 03, pp. 393–412, 2009.

[282] A. Hutt and R. Potthast, "Forecast of spectral features by ensemble data assimilation," *Frontiers in Applied Mathematics and Statistics*, vol. 4, p. 52, 2018.

[283] C. Zerfas, L. G. Rebholz, M. Schneier, and T. Iliescu, "Continuous data assimilation reduced order models of fluid flow," *Computer Methods in Applied Mechanics and Engineering*, vol. 357, p. 112596, 2019.

[284] D. Xiao, J. Du, F. Fang, C. Pain, and J. Li, "Parameterised non-intrusive reduced order methods for ensemble Kalman filter data assimilation," *Computers & Fluids*, vol. 177, pp. 69–77, 2018.

[285] D. N. Daescu and I. Navon, "Efficiency of a POD-based reduced second-order adjoint model in 4D-Var data assimilation," *International Journal for Numerical Methods in Fluids*, vol. 53, no. 6, pp. 985–1004, 2007.

[286] R. Ştefănescu, A. Sandu, and I. M. Navon, "POD/DEIM reduced-order strategies for efficient four dimensional variational data assimilation," *Journal of Computational Physics*, vol. 295, pp. 569–595, 2015.

[287] Y. Cao, J. Zhu, I. M. Navon, and Z. Luo, "A reduced-order approach to four-dimensional variational data assimilation using proper orthogonal decomposition," *International Journal for Numerical Methods in Fluids*, vol. 53, no. 10, pp. 1571–1583, 2007.

[288] C. Robert, S. Durbiano, E. Blayo, J. Verron, J. Blum, and F.-X. Le Dimet, "A reduced-order strategy for 4d-var data assimilation," *Journal of Marine Systems*, vol. 57, no. 1-2, pp. 70–82, 2005.

[289] A. A. Popov, C. Mou, A. Sandu, and T. Iliescu, "A multifidelity ensemble Kalman filter with reduced order control variates," *SIAM Journal on Scientific Computing*, vol. 43, no. 2, pp. A1134–A1162, 2021.

[290] J. Baiges, R. Codina, and S. Idelsohn, "Reduced-order subscales for POD models," *Computer Methods in Applied Mechanics and Engineering*, vol. 291, pp. 173–196, 2015.

[291] D. Kondrashov, M. D. Chekroun, and M. Ghil, "Data-driven non-Markovian closure models," *Physica D: Nonlinear Phenomena*, vol. 297, pp. 33–55, 2015.

[292] L. Fick, Y. Maday, A. T. Patera, and T. Taddei, "A stabilized POD model for turbulent flows over a range of Reynolds numbers: Optimal parameter sampling and constrained projection," *Journal of Computational Physics*, vol. 371, pp. 214–243, 2018.

[293] A. A. Oberai and J. Jagalur-Mohan, "Approximate optimal projection for reduced-order models," *International Journal for Numerical Methods in Engineering*, vol. 105, no. 1, pp. 63–80, 2016.

[294] J. Borggaard, T. Iliescu, and Z. Wang, "Artificial viscosity proper orthogonal decomposition," *Mathematical and Computer Modelling*, vol. 53, no. 1-2, pp. 269–279, 2011.

[295] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu, "Two-level discretizations of nonlinear closure models for proper orthogonal decomposition," *Journal of Computational Physics*, vol. 230, no. 1, pp. 126–146, 2011.

[296] D. Rempfer, "Kohärente strukturen und chaos beim laminar-turbulenten grenzschichtumschlag," Ph.D. dissertation, University of Stuttgart, 1997.

[297] M. Ahmed and O. San, "Stabilized principal interval decomposition method for model reduction of nonlinear convective systems with moving shocks," *Computational and Applied Mathematics*, vol. 37, no. 5, pp. 6870–6902, 2018.

[298] C. Mou, H. Liu, D. R. Wells, and T. Iliescu, "Data-driven correction reduced order models for the quasi-geostrophic equations: A numerical investigation," *International Journal of Computational Fluid Dynamics*, vol. 34, no. 2, pp. 147–159, 2020.

[299] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2018.

[300] A. M. Madni, C. C. Madni, and S. D. Lucero, "Leveraging digital twin technology in model-based systems engineering," *Systems*, vol. 7, no. 1, p. 7, 2019.

[301] S. Chakraborty, S. Adhikari, and R. Ganguli, "The role of surrogate models in the development of digital twins of dynamic systems," *Applied Mathematical Modelling*, vol. 90, pp. 662–681, 2021.

[302] P. Tabeling, "Two-dimensional turbulence: A physicist approach," *Physics Reports*, vol. 362, no. 1, pp. 1–62, 2002.

[303] G. Stabile, F. Ballarin, G. Zuccarino, and G. Rozza, "A reduced order variational multiscale approach for turbulent flows," *Advances in Computational Mathematics*, pp. 1–20, 2019.

[304] R. Reyes and R. Codina, "Projection-based reduced order models for flow problems: A variational multiscale approach," *Computer Methods in Applied Mechanics and Engineering*, vol. 363, p. 112844, 2020.

[305] S. Pawar, S. E. Ahmed, O. San, and A. Rasheed, "An evolve-then-correct reduced order model for hidden fluid dynamics," *Mathematics*, vol. 8, no. 4, p. 570, 2020.

[306] O. San and R. Maulik, "Neural network closures for nonlinear model order reduction," *Advances in Computational Mathematics*, vol. 44, no. 6, pp. 1717–1750, 2018.

[307] S. A. McQuarrie, C. Huang, and K. E. Willcox, "Data-driven reduced-order models via regularised operator inference for a single-injector combustion process," *Journal of the Royal Society of New Zealand*, vol. 51, no. 2, pp. 194–211, 2021.

[308] S. K. Lele, "Compact finite difference schemes with spectral-like resolution," *Journal of Computational Physics*, vol. 103, no. 1, pp. 16–42, 1992.

[309] R. H. Kraichnan, "Inertial ranges in two-dimensional turbulence," *Physics of Fluids*, vol. 10, pp. 1417–1423, 1967.

[310] G. Batchelor, "Computation of the energy spectrum in homogeneous two-dimensional turbulence," *Physics of Fluids*, vol. 12, pp. 233–239, 1969.

[311] C. E. Leith, "Atmospheric predictability and two-dimensional turbulence," *Journal of Atmospheric Science*, vol. 28, pp. 145–161, 1971.

[312] O. San and A. E. Staples, "High-order methods for decaying two-dimensional homogeneous isotropic turbulence," *Computers & Fluids*, vol. 63, pp. 105–127, 2012.

[313] D. Rempfer and H. Fasel, "The dynamics of coherent structures in a flat-plate boundary layer," in *Advances in Turbulence IV*. Springer, Berlin, 1993, pp. 73–77.

[314] F. G. Eroglu, S. Kaya, and L. G. Rebholz, "A modular regularized variational multiscale proper orthogonal decomposition for incompressible flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 325, pp. 350–368, 2017.

[315] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 63–86, 2018.

[316] F. Holzäpfel, T. Hofbauer, D. Darracq, H. Moet, F. Garnier, and C. F. Gago, "Analysis of wake vortex decay mechanisms in the atmosphere," *Aerospace Science and Technology*, vol. 7, no. 4, pp. 263–275, 2003.

[317] F. Holzäpfel, T. Gerz, F. Köpp, E. Stumpf, M. Harris, R. I. Young, and A. Dolfi-Bouteyre, "Strategies for circulation evaluation of aircraft wake vortices measured by LIDAR," *Journal of Atmospheric and Oceanic Technology*, vol. 20, no. 8, pp. 1183–1195, 2003.

[318] C. Breitsamter, "Wake vortex characteristics of transport aircraft," *Progress in Aerospace Sciences*, vol. 47, no. 2, pp. 89–134, 2011.

[319] R. Luckner, G. Höhne, and M. Fuhrmann, "Hazard criteria for wake vortex encounters during approach," *Aerospace Science and Technology*, vol. 8, no. 8, pp. 673–687, 2004.

[320] A. Stephan, J. Schrall, and F. Holzäpfel, "Numerical optimization of plate-line design for enhanced wake-vortex decay," *Journal of Aircraft*, vol. 54, no. 3, pp. 995–1010, 2017.

[321] F. Holzäpfel, A. Stephan, and G. Rotshteyn, "Plate lines reduce lifetime of wake vortices during final approach to Vienna airport," in *AIAA Scitech 2020 Forum*, 2020, p. 0050.

[322] T. Gerz, F. Holzäpfel, and D. Darracq, "Commercial aircraft wake vortices," *Progress in Aerospace Sciences*, vol. 38, no. 3, pp. 181–208, 2002.

[323] J. N. Hallock and F. Holzäpfel, "A review of recent wake vortex research for increasing airport capacity," *Progress in Aerospace Sciences*, vol. 98, pp. 27 – 36, 2018.

[324] Y. Liang, H. Lee, S. Lim, W. Lin, K. Lee, and C. Wu, "Proper orthogonal decomposition and its applications – Part I: Theory," *Journal of Sound and Vibration*, vol. 252, no. 3, pp. 527–544, 2002.

[325] G. Kerschen, J.-c. Golinval, A. F. Vakakis, and L. A. Bergman, "The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview," *Nonlinear Dynamics*, vol. 41, no. 1-3, pp. 147–169, 2005.

[326] S. Volkwein, "Model reduction using proper orthogonal decomposition," *Lecture Notes, Faculty of Mathematics and Statistics, University of Konstanz*, 2011, http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/POD-Vorlesung.pdf.

[327] M.-L. Rapún and J. M. Vega, "Reduced order models based on local POD plus Galerkin projection," *Journal of Computational Physics*, vol. 229, no. 8, pp. 3046–3063, 2010.

[328] S. Lorenzi, A. Cammi, L. Luzzi, and G. Rozza, "POD-Galerkin method for finite volume approximation of Navier–Stokes and RANS equations," *Computer Methods in Applied Mechanics and Engineering*, vol. 311, pp. 151–179, 2016.

[329] K. Kunisch and S. Volkwein, "Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics," *SIAM Journal on Numerical Analysis*, vol. 40, no. 2, pp. 492–515, 2002.

[330] C. Huang, W. E. Anderson, and C. Merkle, "Exploration of POD-Galerkin techniques for developing reduced order models of the Euler equations," in *AIAA Modeling and Simulation Technologies Conference*, 2016, p. 1917.

[331] E. Rezaian and M. Wei, "Impact of symmetrization on the robustness of pod-galerkin roms for compressible flows," in *AIAA Scitech 2020 Forum*, 2020, p. 1318.

[332] B. Xu, H. Gao, M. Wei, and J. Hrynuk, "POD-Galerkin projection ROM for the flow passing a rotating elliptical airfoil," in *AIAA Aviation 2020 Forum*, 2020, p. 3082.

[333] X. Xie, D. Wells, Z. Wang, and T. Iliescu, "Approximate deconvolution reduced order modeling," *Comput. Methods Appl. Mech. Engrg.*, vol. 313, pp. 512–534, 2017.

[334] ——, "Numerical analysis of the Leray reduced order model," *Journal of Computational and Applied Mathematics*, vol. 328, pp. 12–29, 2018.

[335] E. Rezaian and M. Wei, "A hybrid stabilization approach for reduced-order models of compressible flows with shock-vortex interaction," *International Journal for Numerical Methods in Engineering*, vol. 121, no. 8, pp. 1629–1646, 2020.

[336] ——, "Multi-stage stabilization of roms in strongly nonlinear systems," in *AIAA Aviation 2020 Forum*, 2020, p. 3083.

[337] C. R. Wentland, C. Huang, and K. Duraisamy, "Closure of reacting flow reduced-order models via the adjoint Petrov-Galerkin method," in *AIAA Aviation 2019 Forum*, 2019, p. 3531.

[338] A. Lozovskiy, M. Farthing, and C. Kees, "Evaluation of Galerkin and Petrov–Galerkin model reduction for finite element approximations of the shallow water equations," *Computer Methods in Applied Mechanics and Engineering*, vol. 318, pp. 537–571, 2017.

[339] G. Collins, K. Fidkowski, and C. E. Cesnik, "Petrov-Galerkin projection-based model reduction with an optimized test space," in *AIAA Scitech 2020 Forum*, 2020, p. 1562.

[340] E. J. Parish, C. R. Wentland, and K. Duraisamy, "The Adjoint Petrov–Galerkin method for non-linear model reduction," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, p. 112991, 2020.

[341] F. Fang, C. C. Pain, I. M. Navon, A. H. Elsheikh, J. Du, and D. Xiao, "Non-linear Petrov–Galerkin methods for reduced order hyperbolic equations and discontinuous finite element methods," *Journal of Computational Physics*, vol. 234, pp. 540–559, 2013.

[342] S. Lakshmivarahan and J. M. Lewis, "Nudging methods: A critical overview," in *Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications (Vol. II)*. Springer, New York, 2013, pp. 27–57.

[343] P. R. Spalart, "Airplane trailing vortices," *Annual Review of Fluid Mechanics*, vol. 30, no. 1, pp. 107–138, 1998.

[344] M. Tabib, A. Rasheed, and F. Fuchs, "Analyzing complex wake-terrain interactions and its implications on wind-farm performance." *Journal of Physics: Conference Series*, vol. 753, no. 3, p. 032063, 2016.

[345] A. Stephan, J. Schrall, and F. Holzäpfel, "Numerical optimization of plate-line design for enhanced wake-vortex decay," *Journal of Aircraft*, vol. 54, no. 3, pp. 995–1010, 2017.

[346] F. G. Fuchs, A. Rasheed, M. Tabib, and E. Fonn, "Wake modeling in complex terrain using a hybrid Eulerian-Lagrangian split solver," in *Journal of Physics: Conference Series*, vol. 753, 2016, p. 082031.

[347] M. P. Brenner, J. D. Eldredge, and J. B. Freund, "Perspective on machine learning for advancing fluid mechanics," *Physical Review Fluids*, vol. 4, no. 10, p. 100501, 2019.

[348] X. Xie, G. Zhang, and C. G. Webster, "Non-intrusive inference reduced order model for fluids using deep multistep neural network," *Mathematics*, vol. 7, no. 8, p. 757, 2019.

[349] J. Yu and J. S. Hesthaven, "Flowfield reconstruction method using artificial neural network," *AIAA Journal*, vol. 57, no. 2, pp. 482–498, 2019.

[350] S. Pawar, S. Rahman, H. Vaddireddy, O. San, A. Rasheed, and P. Vedula, "A deep learning enabler for nonintrusive reduced order modeling of fluid flows," *Physics of Fluids*, vol. 31, no. 8, p. 085101, 2019.

[351] O. San, R. Maulik, and M. Ahmed, "An artificial neural network framework for reduced order modeling of transient flows," *Communications in Nonlinear Science and Numerical Simulation*, vol. 77, pp. 271–287, 2019.

[352] R. Maulik, A. Mohan, B. Lusch, S. Madireddy, P. Balaprakash, and D. Livescu, "Time-series learning of latent-space dynamics for reduced-order model closure," *Physica D: Nonlinear Phenomena*, vol. 405, p. 132368, 2020.

[353] S. A. Renganathan, R. Maulik, and V. Rao, "Machine learning for nonintrusive model order reduction of the parametric inviscid transonic flow past an airfoil," *Physics of Fluids*, vol. 32, no. 4, p. 047110, 2020.

[354] R. Maulik, K. Fukami, N. Ramachandra, K. Fukagata, and K. Taira, "Probabilistic neural networks for fluid flow surrogate modeling and data recovery," *Physical Review Fluids*, vol. 5, no. 10, p. 104401, 2020.

[355] A. Lorenc, R. Bell, and B. Macpherson, "The meteorological office analysis correction data assimilation scheme," *Quarterly Journal of the Royal Meteorological Society*, vol. 117, no. 497, pp. 59–89, 1991.

[356] J. Derber and A. Rosati, "A global oceanic data assimilation system," *Journal of Physical Oceanography*, vol. 19, no. 9, pp. 1333–1347, 1989.

[357] P. C. Di Leoni, A. Mazzino, and L. Biferale, "Synchronization to big data: Nudging the Navier-Stokes equations for data assimilation of turbulent flows," *Physical Review X*, vol. 10, no. 1, p. 011023, 2020.

[358] J. N. Hallock and F. Holzäpfel, "A review of recent wake vortex research for increasing airport capacity," *Progress in Aerospace Sciences*, vol. 98, pp. 27–36, 2018.

[359] N. N. Ahmad and F. Proctor, "Review of idealized aircraft wake vortex models," in *52nd Aerospace Sciences Meeting*, 2014, p. 0927.

[360] S. Lugan, L. Bricteux, B. Macq, P. Sobieski, G. Winckelmans, and D. Douxchamps, "Simulation of LIDAR-based aircraft wake vortex detection using a bi-Gaussian spectral model," in *2007 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2007, pp. 4806–4809.

[361] V. J. Rossow, "Convective merging of vortex cores in liftgenerated wakes," *Journal of Aircraft*, vol. 14, no. 3, pp. 283–290, 1977.

[362] Y. Aboelkassem, G. H. Vatistas, and N. Esmail, "Viscous dissipation of Rankine vortex profile in zero meridional flow," *Acta Mechanica Sinica*, vol. 21, no. 6, pp. 550–556, 2005.

[363] H. Lamb, *Hydrodynamics*. University Press, 1924.

[364] F. Holzapfel, T. Gerz, M. Frech, and A. Dornbrack, "Wake vortices in convective boundary layer and their influence on following aircraft," *Journal of Aircraft*, vol. 37, no. 6, pp. 1001–1007, 2000.

[365] F. Proctor, "The NASA-Langley wake vortex modelling effort in support of an operational aircraft spacing system," in *36th AIAA Aerospace Sciences Meeting and Exhibit*, 1998, p. 589.

[366] F. Proctor, D. Hamilton, and J. Han, "Wake vortex transport and decay in ground effect-vortex linking with the ground," in *38th aerospace sciences meeting and exhibit*, 2000, p. 757.

[367] R. A. Anthes, "Data assimilation and initialization of hurricane prediction models," *Journal of the Atmospheric Sciences*, vol. 31, no. 3, pp. 702–719, 1974.

[368] L. Lei and J. P. Hacker, "Nudging, ensemble, and nudging ensembles for data assimilation in the presence of model error," *Monthly Weather Review*, vol. 143, no. 7, pp. 2600–2610, 2015.

[369] D. R. Stauffer and J.-W. Bao, "Optimal determination of nudging coefficients using the adjoint equations," *Tellus A*, vol. 45, no. 5, pp. 358–369, 1993.

[370] X. Zou, I. M. Navon, and F. Le Dimet, "An optimal nudging data assimilation scheme using parameter estimation," *Quarterly Journal of the Royal Meteorological Society*, vol. 118, no. 508, pp. 1163–1186, 1992.

[371] P. Vidard, F. Le Dimet, and A. Piacentini, "Determination of optimal nudging coefficients," *Tellus A: Dynamic Meteorology and Oceanography*, vol. 55, no. 1, pp. 1–15, 2003.

[372] D. Auroux and J. Blum, "Back and forth nudging algorithm for data assimilation problems," *Comptes Rendus Mathematique*, vol. 340, no. 12, pp. 873–878, 2005.

[373] J. Wu, H. Xiao, R. Sun, and Q. Wang, "Reynolds-averaged Navier–Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned," *Journal of Fluid Mechanics*, vol. 869, pp. 553–586, 2019.

[374] J. Wu, H. Xiao, and E. Paterson, "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework," *Physical Review Fluids*, vol. 3, no. 7, p. 074602, 2018.

[375] M. Asch, M. Bocquet, and M. Nodet, *Data assimilation: Methods, algorithms, and applications*. SIAM, 2016.

VITA

Shady E. Ahmed

Candidate for the Degree of

Doctor of Philosophy

Dissertation: Model-Data Fusion in Digital Twins of Large Scale Dynamical Systems

Major Field: Mechanical & Aerospace Engineering

Biographical:

Education:

Completed the requirements for the degree of Doctor of Philosophy with a major in Mechanical & Aerospace Engineering at Oklahoma State University in July 2022.
Received a Master of Science in Mechanical Power Engineering at Mansoura University, Egypt in March 2017.
Received a Bachelors of Science in Mechanical Power Engineering at Mansoura University, Egypt in July 2013.

Experience:

Graduate Research Assistant, Computational Fluid Dynamics Laboratory, Oklahoma State University.
Graduate Teaching Assistant, Mechanical & Aerospace Engineering, Oklahoma State University.
Assistant Lecturer, Mechanical Power Engineering, Mansoura University.

Professional Affiliations:

American Physical Society (APS) Member.
Society for Industrial and Applied Mathematics (SIAM) Member.
American Institute of Aeronautics and Astronautics (AIAA) Member.