# Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism

Qian Wang*, Nicolò Ripamonti, Jan S. Hesthaven

*Chair of Computational Mathematics and Simulation Science, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland*

**Abstract**

Closure modeling based on the Mori-Zwanzig formalism has proven effective to improve the stability and accuracy of projection-based model order reduction. However, closure models are often expensive and infeasible for complex nonlinear systems. Towards efficient model reduction of general problems, this paper presents a recurrent neural network (RNN) closure of parametric POD-Galerkin reduced-order model. Based on the short time history of the reduced-order solutions, the RNN predicts the memory integral which represents the impact of the unresolved scales on the resolved scales. A conditioned long short term memory (LSTM) network is utilized as the regression model of the memory integral, in which the POD coefficients at a number of time steps are fed into the LSTM units, and the physical/geometrical parameters are fed into the initial hidden state of the LSTM. The reduced-order model is integrated in time using an implicit-explicit (IMEX) Runge-Kutta scheme, in which the memory term is integrated explicitly and the remaining right-hand-side term is integrated implicitly to improve the computational efficiency. Numerical results demonstrate that the RNN closure can significantly improve the accuracy and efficiency of the POD-Galerkin reduced-order model of nonlinear problems. The POD-Galerkin reduced-order model with the RNN closure is also shown to be capable of making accurate predictions, well beyond the time interval of the training data.

*Keywords:* memory closure, POD-Galerkin, model reduction, conditioned long-short term memory, implicit-explicit Runge-Kutta

## 1. Introduction

High-fidelity numerical simulation has become an important tool for investigation of complex problems of scientific interest or of industrial value. However, high-fidelity simulations are often expensive since a large number of degrees of freedoms (DOFs) are needed to achieve high space/time accuracy. Therefore, high-fidelity simulation is usually not available for applications like design, control, optimization and uncertainty quantification, all of which require repeated model evaluations over a potentially large range of parameter values [6]. This has led to the development of *reduced order modeling* (ROM) [41] which aims at building low-dimensional models that are fast to evaluate while being able to accurately approximate the underlying high-fidelity solutions.

*Projection-based model reduction* is one of the well-known and widely-used ROM techniques, generally implemented in an offline-online paradigm [44]. During the *offline* stage, a set of reduced basis are extracted from a collection of high-fidelity solutions (snapshots). The reduced space,

*Corresponding author.

*Email addresses:* `qian.wang@epfl.ch` (Qian Wang), `nicolo.ripamonti@epfl.ch` (Nicolò Ripamonti), `Jan.Hesthaven@epfl.ch` (Jan S. Hesthaven) spanned by a set of basis functions, represents the main dynamics of the full-order model. Proper orthogonal decomposition (POD) is one of the most popular methods to construct the reduced basis. During the *online* stage, the reduced basis solutions are recovered by evaluating the reduced-order model, obtained as the projection of the full-order model onto the reduced space. Galerkin projection, in which the reduced basis functions are used as the test functions, is the simplest and most popular choice. The model reduction method, using the POD to generate the basis functions and the Galerkin projection to generate the reduced-order model, is typically referred to as the POD-Galerkin method.

The POD-Galerkin method has been successfully applied to a variety of problems. However, the POD-Galerkin lacks *a priori* guarantees of stability, accuracy and convergence [31, 33, 58] for general time-dependent nonlinear problems. Extensive efforts have been made to improve the stability and accuracy of the POD-Galerkin method, such as the structure-preservation [43], supremizer enrichment [4, 59],basis adaption [9, 53], $L^1$-norm minimization [2], and least-squares Petrov-Galerkin (LSPG) technique [10].

*Closure modeling* is an alternative approach to improve the stability and accuracy of the reduced-order model. The closure term is added to the reduced-order model to account for the effect of the unresolved dynamics on the resolved dynamics, and plays the same role as the subgrid-scale stress in large-eddy simulation (LES). The Mori-Zwanzig formalism, originally developed by Mori [46] and Zwanzig [67] and reformulated by Chorin and co-workers [16, 17, 19], offers a framework for closure modeling in model order reduction. For the POD-Galerkin, the basis functions represent the resolved scales, and the truncated higher-order terms represent the unresolved scales. In the Mori-Zwanzig formalism, the impact of the unresolved scales on the resolved scales is expressed as a *memory term* in the exact dynamics of the resolved scales. The memory term is an integral of a function of the resolved scales, which is called *memory kernel*, over the trajectory of the resolved scales. While the memory term is helpful for accuracy and stability of the reduced-order model, it is very expensive to compute, and thus does not offer a tool for practical problems. Towards cheaper memory effect modeling, *short-memory assumptions*, such as the $t$-model [16] and the $\tau$-model [50, 60], have been developed and applied to model reduction for several nonlinear problems [52, 64]. However, these models are not suitable for model reduction of general nonlinear problems, for a number of reasons. First, the assumptions that are made to derive these first-order approximations of the memory, are often not satisfied for general problems. Second, the model parameters determined by different kinds of strategies using the training data, do not generalize to unseen test data with a guarantee of accuracy.

Data-driven memory models, based on *artificial neural networks* (ANNs) have recently been proposed for closure modeling of model reduction to overcome the shortcomings of the theoretical memory models. In these models, the input of the network is a sequence of reduced coefficients, and the output is the predicted memory integral. Two kinds of neural networks have been used in memory effect modeling, including the feedforward neural network (FNN) [48] and recurrent neural network (RNN) [42, 45, 63]. Pan and Duraisamy [48] utilized a FNN for memory effect modeling for model reduction of ordinary differential equations (ODEs) in physical space and viscous Burgers equation in Fourier space. Wan et al. [63] utilized the long-short term memory (LSTM) network [29] to predict the memory term of the reduced-order models of incompressible flow in Fourier space. Ma et al. [42] utilized an LSTM as a subgrid-scale stress model in large-eddy simulations (LES). More recently, Maulik et al. [45] applied an LSTM memory approximation technique in [42] for non-intrusive predictive modeling for accuracy improvement. It should be noted that, the aforementioned neural network memory models can only be used in the model reduction without parametric dependence. However, for *parametric model reduction* [7], physical/geometrical parameters need to be considered in the memory model. Furthermore, as a network for processing sequential data [24], RNN is a more suitable choice than FNN for memory effect modeling.

This paper proposes an framework for efficient parametric POD-Galerkin model order reduction with a RNN memory closure. In this framework, the memory term of the POD-Galerkin reduced-order model is predicted by a *conditioned LSTM network*. In the conditioned LSTM network, the input is a sequence of reduced coefficients that is fed into the LSTM layer, the auxiliary input is the physical/geometrical parameter vector that is transformed into the initial hidden state of the LSTM through a fully-connected layer. The output is the predicted memory term obtained at the last time step by passing the final hidden state through another fully-connected layer. The optimal length of the input sequence is determined by selecting the network with the highest generalization accuracy, among the trained networks of which the corresponding memory lengths lie in the estimated range. A significant advantage of the proposed RNN memory model over existing FNN/RNN memory models is the involvement of the physical/geometrical parameters, which is necessary in parametric model reduction. The physical/geometrical parameter information can be maintained across the LSTM cells and delivered to the final output, since the LSTM has the capability to keep long-term memory due to its gated structure [24].

A bottleneck for closure modeling of model reduction is the efficiency problem, which arises from the expensive evaluation of the memory model. Typically, the computational cost of the reduced-order model with the memory closure is much higher than the original model. An efficient implementation of the RNN memory model in the POD-Galerkin framework is proposed in this paper to address the efficiency issue. An *implicit-explicit Runge-Kutta* (IMEX-RK) scheme is used to integrate the reduced-order model with RNN memory closure in time. More specifically, the memory term is integrated explicitly and the remaining term of the right-hand-side is integrated implicitly. In each Runge-Kutta stage, an implicit algebraic equation system is obtained after the computation of the memory term. For linear problems, the equation system is solved directly with a computational cost much smaller than the evaluation of the RNN memory model. For nonlinear problems, the equation system is solved iteratively with a computational cost much larger than the evaluation of the RNN memory model. Hence, for model reduction of nonlinear problems, the RNN closure can increase the accuracy of the reduced-order model drastically with an only slightly increased computational cost, thus significantly improve the efficiency of the reduced-order model.

A nonlinear right-hand-side (RHS) term treatment strategy is used to obtain a reduced-order model that is independent on the size of the full-order model. In this paper, we are interested in the model reduction of fluid flows, for which the dynamics are restricted to *quadratic polynomial nonlinearity*. The matrices in the RHS of the reduced-order model can be precomputed by expressing the nonlinear terms in the form of polynomial expansions. By taking advantage of the polynomial structure of the RHS, a reduced-order model independent of the full-order size is obtained without impacting the accuracy.

The proposed method is demonstrated on 3D Stokes flow, 1D Kuramoto–Sivashinsky equation and 2D Rayleigh-Bénard convection. Numerical results demonstrate that the conditioned LSTM network can predict the memory term accurately. Numerical resutls demonstrate that the conditioned LSTM memory model can significantly improve the accuracy and efficiency of the POD-Galerkin reduced-order model of nonlinear problems. The LSTM memory closure can significantly improve the stability and accuracy of the POD-Galerkin reduced-order model of the linear 3D Stokes problem. Furthermore, the POD-Galerkin with the LSTM memory closure is capable of making accurate predictions, in the sense that it can provide fast and accurate solutions beyond the time range of the training data.

The remainder of this paper is organized as follows. Section 2 presents the background of model reduction and Section 3 presents the memory modeling via the Mori-Zwanzig formalism. In Section 4, we introduce the structure and training of the conditioned LSTM network. Section 5 presents the efficient implementation of the POD-Galerkin with conditioned LSTM memory closure using the IMEX Runge-Kutta time integration. Section 6 presents the numerical results and Section 7 gathers the relevant conclusions.

## 2. Model Order Reduction

### *2.1. Reduced Basis*

Consider a parameterized, finite dimensional dynamical system, described by the following set of first order ordinary differential equations (ODE)

$$\begin{cases} \dfrac{d}{dt}u(t;\omega) = f(t,u;\omega), \\ u(0;\omega) = u_0(\omega). \end{cases} \tag{1}$$

Here $u \in \mathbb{R}^N$ is the state vector, $\omega \in \Gamma$ is the vector containing all the parameters belonging to a compact set $\Gamma \subset \mathbb{R}^d$ and $f : \mathbb{R} \times \mathbb{R}^N \times \Gamma \mapsto \mathbb{R}^N$ is the right-hand-side of the system. Problems described by (1) often arise from the semidiscrete formulation of systems of partial differential equations (PDEs), using a suitable high-fidelity (HF) method (finite element [54], discontinuous Galerkin [28], ...) for the treatment of the spatial derivatives. To achieve satisfactory results in terms of accuracy of the numerical approximations, these HF methods rely on high-order approximations or the use of a fine mesh, resulting in $N$ being large and, hence, increasing the computational cost of the simulation.

We introduce the set of all solutions to (1) for different values of the parameter $\omega \in \Gamma$ as

$$M = \{u(t;\omega) | t \in \mathbb{R}, \omega \in \Gamma\} \subset \mathbb{R}^N. \tag{2}$$

In the following, we assume that it is possible to integrate (1) with arbitrary accuracy $\forall\ \omega \in \Gamma$, so that the elements of $M$ can be identified with the corresponding discrete solutions. A parametrized PDE is said to be reducible if $M$ can be well approximated by some $m$-dimensional subspace $\mathbb{V}$. The reducibility of the problem can be quantified by the Kolmogorov $m$-width [62], which represents the smallest error that can be obtained by approximating $M$ with a $m$-dimensional space. Define the basis vectors $\{v_i\}_{i=1}^m$ of the subspace $\mathbb{V}$, known as the *reduced basis*, and consider the ansatz

$$u \approx V\alpha + \bar{u}, \tag{3}$$

where $V \in \mathbb{R}^{N\times m}$ is a matrix with the basis vectors as columns, $\alpha \in \mathbb{R}^m$ are the coefficients of $u$ with respect to the reduced basis and $\bar{u} \in \mathbb{R}^N$ is a reference value. The role of the reference value is to reduce the study of the original field $u$ to the study of fields fluctuating around a significant value (average, initial condition, $\cdots$). This procedure prevents the first reduced coefficient from containing the majority of the energy of the system and helps stabilize the reduced system. We show in Section 3.2 that an appropriate choice of $\bar{u}$ simplifies the form of the contribution of the unresolved part of the simulation. Inserting (3) into (1), we obtain the overdetermined system

$$\begin{cases} V\dfrac{d}{dt}\alpha(t;\omega) = f(t, V\alpha + \bar{u};\omega) + \mathfrak{r}(t;\omega), \\ \alpha(0;\omega) = V^T\left(u_0(\omega) - \bar{u}\right). \end{cases}$$

The quantity $\mathfrak{r}$ represents the residual due to (3). In the framework of a Petrov-Galerkin projection, if we consider a basis $W$ that is orthogonal to the residual and $W^TV$ is invertible, we recover the reduced system of $m$ equations

$$\begin{cases} \dfrac{d}{dt}\alpha(t;\omega) = (W^TV)^{-1}W^Tf(t, V\alpha + \bar{u};\omega), \\ \alpha(0;\omega) = V^T\left(u_0(\omega) - \bar{u}\right). \end{cases} \tag{4}$$

In this work we restrict ourselves to the Galerkin framework, reducing (4) to

$$\begin{cases} \dfrac{d}{dt}\alpha(t;\omega) = V^Tf(t, V\alpha + \bar{u};\omega), \\ \alpha(0;\omega) = V^T\left(u_0(\omega) - \bar{u}\right). \end{cases} \tag{5}$$

Several strategies have been developed to compute the reduced basis $\{v_i\}_{i=1}^m$, each being optimal with respect to a different error norm or suitable for specific problems. Among the most recognized, we mention the Proper Generalized Decomposition [13], the Matrix Interpolation [49], the Piecewise Tangential Interpolation [21], the Loewner framework and several greedy-based approaches for the identification of sub-optimal subspaces [8, 38]. In the following we describe the most widely used reduced basis technique, the Proper Orthogonal Decomposition (POD) [27, 56], which is used in what remains.

### *2.1.1. Proper Orthogonal Decomposition*

The Proper Orthogonal Decomposition (POD) method is a technique to compress data and retain fundamental information in the form of an orthogonal basis matrix, optimal in the least-squares sense. Consider $\{u(t_i;\omega_j)-\bar{u}\}_{i=1,\dots,p,j=1,\dots,q}$ as a set of solution frames at different times and for different values of the parameter $\omega$ minus the reference value $\bar{u}$. We define the snapshot matrix $S\in\mathbb{R}^{N\times pq}$ as the matrix having the collected snapshots as columns. Therefore, the POD basis of size $m$ is the solution to the minimization problem

$$\begin{aligned}\min_{V\in\mathbb{R}^{N\times m}} &\|S-VV^TS\|_F,\\ \text{subject to } &V^TV=\mathbb{I},\end{aligned}\tag{6}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\mathbb{I}\in\mathbb{R}^{pq\times pq}$ is the identity matrix.

The Eckart-Young theorem guarantees that (6) has a solution, given by the first $m$ left singular vectors of the matrix $S$. In particular, if

$$S=U_S\Sigma_SV_S^*,\qquad \Sigma_S=\mathrm{diag}(\sigma_i)$$

is the singular value decomposition (SVD) of S, we have

$$\min_{V\in\mathbb{R}^{N\times m}}\|S-VV^TS\|_F^2=\sum_{i=m+1}^{\min\{N,pq\}}\sigma_i^2.\tag{7}$$

Formula (7) states that the error in the POD basis is equal to the sum of the squares of the neglected singular values. If the snapshot matrix $S$ is computed using a sampling, rich enough to correctly capture the dynamic of the parametrized problem, the decay of its singular values can be used as a surrogate for the Kolmogorov $m$-width. Many problems exhibit an exponential decay of the singular values and hence, using (7), there exists a low-dimensional linear reduced space that can be used to approximate the HF solution with arbitrary accuracy.

For the time-dependent problems with many time steps, such as the Rayleigh-Bénard convection in Section 6.3, the snapshot matrix is large, leading to an expensive SVD. To overcome this difficulty, we use the randomized SVD algorithm [26] which only needs to perform SVD of small matrices, to efficiently generate a reduced basis of the problem with large snapshot matrix.

## *2.2. Treatment of nonlinear term*

The computational complexity of (5) still depends on $N$, even though we are solving a system of ODEs of dimension $m\ll N$. Suppose that the RHS of (3) is of the form $f(t,u;\omega)=Lu+g(t,u;\omega)$, where $L\in\mathbb{R}^{N\times N}$ represents the linear and affine part and $g$ is nonaffine and parameter dependent. The POD-Galerkin approximation of the original problem is

$$\frac{d}{dt}\alpha(t;\omega)=\underbrace{V^TLV}_{\tilde{L}}\alpha+\underbrace{V^TL\bar{u}}_{\tilde{u}}+\underbrace{V^Tg(t,V\alpha+\bar{u};\omega)}_{Q(V\alpha+\bar{u})},\tag{8}$$

where $\tilde{L} \in \mathbb{R}^{m\times m}$ and $\tilde{u} \in \mathbb{R}^m$ can be computed once. The computational bottleneck of the reduced simulation is the assembly and evaluation of the nonaffine term $Q(V\alpha + \bar{u})$, the cost of which scales with the dimension of the high-fidelity method. During the last decade, several hyper-reduction approaches have been introduced to enable significant speedups for nonaffine problems. Most of these techniques require the identification of a subset of components of the nonaffine function, which is evaluated during the online stage and used to compute an approximation of the nonaffine contribution to the dynamic of the system. The Empirical Interpolation method (EIM)[5], its discrete variant (DEIM)[12], Gappy-POD [20] and Missing Point Estimation (MPE)[3] belong to this class of hyper-reduction algorithms. In this work, we restrict ourselves to the case of quadratic nonlinearities, for which the hyper-reduction is not required because it is possible to avoid the aforementioned computational bottleneck [11]. Consider the following form for the nonaffine term

$$Q(V\alpha + \bar{u}) = V^T g(V\alpha + \bar{u}) = \left(V^T r(V\alpha + \bar{u})\right) \circ (V\alpha + \bar{u}),$$

where $\circ$ is the element-wise product operator and $r : \mathbb{R}^N \mapsto \mathbb{R}^N$ is an affine function. We have omitted the dependence on $t$ and $\omega$ for all the terms involved for the sake of simplicity. Hence, the nonaffine term can be rewritten as

$$Q(V\alpha + \bar{u}) = \sum_{i=1}^{m}\sum_{j=1}^{m} A_{ij}\alpha_i\alpha_j + \sum_{i=1}^{m} B_i\alpha_i + D, \tag{9}$$

with

$$\begin{aligned} A_{ij} &= \sum_{i=1}^{m}\sum_{j=1}^{m} V^T \mathrm{diag}\left(r(v_i)\right) v_j, \\ B_i &= \sum_{i=1}^{m} V^T\left(\mathrm{diag}\left(r(v_i)\right)\bar{u} + \mathrm{diag}\left(r(\bar{u})\right) v_i\right), \\ D &= V^T \mathrm{diag}\left(r(\bar{u})\right)\bar{u}. \end{aligned}$$

Once $A \in \mathbb{R}^{m\times m}, B \in \mathbb{R}^{m\times m}$ and $D \in \mathbb{R}^m$ have been assembled during the offline stage of the reduction, the online evaluation of (9) requires a number of operations proportional to $m^2$. A similar strategy can be devised for higher order polynomial nonlinearities. However, the beneficial effects of this direct approach in terms of computational cost decreases as the polynomial degree increases. Even though not all the discretized PDE equations show a dynamic that depends on polynomials of the state of the system, in [36], a lifting strategy via the introduction of auxiliary variables has been proposed to reduce general problems to the considered setting.

## 3. Mori-Zwanzig formalism

### *3.1. Introductory example*

We start by considering an illustrative example as an introduction to the Mori-Zwanzig formalism. Consider the linear system of equations

$$\begin{cases} \dfrac{d}{dt}y^1 = A_{11}y^1 + A_{12}y^2, \\ \dfrac{d}{dt}y^2 = A_{21}y^1 + A_{22}y^2, \end{cases} \tag{10}$$

with $A_{11} \in \mathbb{R}^{m\times m}$, $A_{12} \in \mathbb{R}^{m\times(N-m)}$, $A_{21} \in \mathbb{R}^{(N-m)\times m}$ and $A_{22} \in \mathbb{R}^{(N-m)\times(N-m)}$. Suppose we are interested in the dynamic described by the variable $y^1 \in \mathbb{R}^m$ in time, representing the first $m$

POD coefficients. Solving the entire system (10) can be computationally expensive and if $m$ is assumed to be much smaller than $N$, subselecting entries from the full solution of (10) is not worth the cost of the procedure.

Our goal is to write a reduced system that describes the evolution of $y^1$ such that its RHS depends only on $y^1$, that is

$$\frac{d}{dt}y^1 = A_{11}y^1 + \mathcal{M}(t, y^1). \tag{11}$$

To compute $\mathcal{M}$, assume that $y^1$ is known and integrate the second equation in (10) to obtain

$$y^2(t) = e^{A_{22}t}y^2(0) + \int_0^t e^{A_{22}(t-s)}A_{21}y^1(s)\,ds. \tag{12}$$

If we insert (12) into the first equation, we recover

$$\frac{d}{dt}y^1 = A_{11}y^1 + A_{12}\int_0^t e^{A_{22}(t-s)}A_{21}y^1(s)\,ds + A_{21}e^{A_{22}t}y^2(0). \tag{13}$$

Equation (13) is a generalized Langevin equation (GLE). This implies that the dynamic of the problem is governed by a Markovian contribution, a memory integral term depending only on the resolved $y^1$, and a term describing the influence of the unresolved initial condition $y^2(0)$. Its main property is that it is a closed equation in $y^1$, because we neglected the dependence on $y^2$. No approximations have been introduced and (13) is exact.

Except for the linear case, it is not straightforward to derive the term $\mathcal{M}$ from (10), that represents the impact that the unresolved $y^2$ has on $y^1$. In the nonlinear case, following the introduction of projection operators, the Mori-Zwanzig (MZ) formalism allows to introduce a GLE describing the evolution of the resolved part. Here we do not consider the original Zwanzig projector operator [66], which could lead to a formulation of the problem in terms of a kinetic equation, but the projection based Mori procedure [46], resulting in a GLE that is linear in the system variables. In particular we use the generalization, introduced by Chorin [18], to non-Hamiltonian problems.

The major contribution of the Mori-Zwanzig theory is the potential to reduce a large set of Markovian equations to a lower dimensional and non-Markovian set of equations for a subset of variables of the original problem, maintaining the effect of the unresolved set. This is useful in practice, since an exact solution of all the degrees of freedom is often unnecessary to understand physical phenomena. As expected, this separation of contributions to the resolved dynamic comes at a price.

First, the memory term depends on the so called projected dynamics, which is not the one given by the solution of the reduced system and rarely can be written explicitly. Even in the linear case, it requires the computation of the matrix-exponential of dimension $N - m$, i.e. $e^{A_{22}(t-s)}$. Second, it is difficult to use the resulting GLE equation as a direct computational tool given its integral-differential nature, which in principle requires a number of operations comparable to that of the solution of the full model (10).

More importantly, if our goal is to represent or study the additional term in the dynamical system (13), certain properties are required of the basis used to describe the physical problem. This question has a more profound physical implication related to the possibility of representing a physical system using a macroscopic description obtained from a microscopic representation, which introduces the concepts of fast and slow scales of the representation. Not all high-fidelity methods provide a dynamical system of the required form, in which the degrees of freedom of the system are hierarchically ordered and it is possible to identify slow/fast or large/small scales. On the other hand, globally hierarchical methods such as Fourier and POD methods are formulated to extract such a hierarchical set of structures. If the basis is chosen in an appropriate way, the dominating contribution to the dynamic of the resolved part of the simulation is given by the resolved part itself, while the additional terms in (13) can be neglected, simplified or modelled as noise, as discussed in the following Section.

## *3.2. Mathematical foundations*

Consider the generic system

$$\begin{cases} \dfrac{d}{dt}y = f(t, y(t)), \quad \forall t \in [0, T] \\ y(0) = y_0, \end{cases} \tag{14}$$

evolving on a smooth manifold $S$. We assume that $f : \mathbb{R}^N \mapsto \mathbb{R}^N$ is at least uniformly continuous, so that system (14) is well posed and has a unique solution for a general initial condition. To provide a context, consider the POD reduced system (4) . The general use of the formalism we describe here is to study vector-valued observales $g : S \mapsto \mathbb{R}^l$. The only property required to $g$ is to form a Banach space, to give a proper meaning to the subspaces and projectors involved, in accordance with [65], which we follow in the derivation of the Mori-Zwanzig system.

The potentially complex dynamic described by the observable $g$ can be formally represented by using a semigroup $\mathcal{K}(t, s)$ of operators acting on the Banach space of observables. In particular, we have

$$g(y(t)) = [\mathcal{K}(t, s)g]\, y(s),$$

where

$$\mathcal{K}(t, s) = e^{(t-s)\mathcal{L}}, \qquad \mathcal{L}g(y) = f(y) \cdot \nabla g(y). \tag{15}$$

In [18], it is shown that the nonlinear ordinary differential equation (14) is equivalent to the linear partial differential equation

$$\begin{cases} \dfrac{\partial}{\partial t}y = \mathcal{L}y, \\ y(0) = g(y_0), \end{cases} \tag{16}$$

known as the Liouville equation, with $\mathcal{L}$ being the Liouville operator, if $g$ is the identity operator. The equivalence holds in the sense that (14) is the set of infinite characteristics equations for (16). In particular, if $g(y(t)) = y_k(t)$, the solution to (16) is the $k$-th component of the solution to (14).

The Liouville operator $\mathcal{L}$ enjoys the following interesting property

$$e^{t\mathcal{L}}g(y(t)) = g(e^{t\mathcal{L}}y(t)), \tag{17}$$

which has been used in [25] for the *a priori* approximation of the memory integral.

Consider the solution to (14), $y(t) \in \mathbb{V}'$, and two spaces $\mathbb{V}$ and $\tilde{\mathbb{V}}$ such that $\mathbb{V}' = \mathbb{V} \oplus \tilde{\mathbb{V}}$. In particular we consider the case in which the elements of $\mathbb{V}$ are vectors with only the first $m$ components different from 0. Similarly, $\tilde{\mathbb{V}}$ contains elements with only the last $N - m$ components different from 0. This represents a common scenario in which a solution is represented in terms of hierarchical coefficients, as for POD and Fourier approximations. The first $m$ components denote the resolved part of the simulation while the remaining degrees of freedom denote the unresolved part.

The subspace $\mathbb{V}$ is described through an operator $\mathcal{P}$ having the subspace as its image, and acting on the set of observables. We define the natural complementary projector as $\mathcal{Q} = I - \mathcal{P}$.

The first step to derive an exact equation for the dynamic of the observables is to apply Dyson's identity to the Koopman operator

$$e^{tL} = e^{t\mathcal{QL}} + \int_0^t e^{s\mathcal{L}}\mathcal{PL}e^{(t-s)\mathcal{QL}}ds. \tag{18}$$

Starting from (18), we obtain the operator equation

$$\frac{\partial}{\partial t}e^{t\mathcal{L}} = e^{t\mathcal{L}}\mathcal{PL} + e^{t\mathcal{QL}}\mathcal{QL} + \int_0^t e^{s\mathcal{L}}\mathcal{PL}e^{(t-s)\mathcal{QL}}\mathcal{QL}ds. \tag{19}$$

The application of (19) to the initial state of the system gives the following Mori-Zwanzig identity in phase space

$$\begin{aligned}\frac{\partial}{\partial t} e^{t\mathcal{L}} y_0 &= e^{t\mathcal{L}} \mathcal{P}\mathcal{L} y_0 + e^{t\mathcal{Q}\mathcal{L}} \mathcal{Q}\mathcal{L} y_0 + \int_0^t e^{s\mathcal{L}} \mathcal{P}\mathcal{L} e^{(t-s)\mathcal{Q}\mathcal{L}} \mathcal{Q}\mathcal{L} y_0 ds \\ &= e^{t\mathcal{L}} \mathcal{P}\mathcal{L} y_0 + F(t, y_0) + \int_0^t K(t-s, e^{s\mathcal{L}} y_0) ds,\end{aligned} \tag{20}$$

where

$$F(t,x) = e^{t\mathcal{Q}\mathcal{L}} \mathcal{Q}\mathcal{L} x, \quad K(t,x) = \mathcal{P}\mathcal{L}F(t,x).$$

Define $y = (\hat{y}, \tilde{y})$ such that $\hat{y} \in \mathbb{V}$ and $\tilde{y} \in \tilde{\mathbb{V}}$, if we take as operator $\mathcal{P}$ the truncation $\mathcal{P}g(\hat{y}, \tilde{y}) = g(\hat{y}, 0)$, (20) can be rewritten in terms of $y(t) = (\hat{y}(t), \tilde{y}(t))$ as

$$\frac{d}{dt}\hat{y}(t) = f(t, \hat{y}(t)) + F(t, y_0) + \int_0^t K(t-s, \hat{y}(s)) ds. \tag{21}$$

We stress that (21) is not an approximation of (14), but an exact representation of the evolution equation for the first $m$ components, in which contributions from different sources are clearly separated and recast in the context of a linear equation.

It is worth to discuss the role of each contribution in (21). The first term, depending only on the resolved dynamic, represents the self-interaction of the resolved variables and it is the Markovian contribution to the time derivative of the resolved coefficients. The last term, commonly known as the memory, depends on the resolved part of the simulation at all time $s$ between 0 and $t$, with the integrand commonly known as the memory kernel. For the second term we have that $F(t, y_0)$ satisfies the linear PDE

$$\begin{cases} \dfrac{\partial}{\partial t} F(t, y_0) = \mathcal{Q}\mathcal{L}F(t, y_0), \\ F(0, y_0) = \mathcal{Q}\mathcal{L} y_0 = \mathcal{L} y_0 - \mathcal{P}\mathcal{L} y_0 = f(0, y_0) - f(0, \hat{y}_0), \end{cases} \tag{22}$$

known as the orthogonal dynamics equation. Projecting (22) gives

$$\begin{cases} \mathcal{P}\dfrac{\partial}{\partial t} F(t, y_0) = \mathcal{P}\mathcal{Q}\mathcal{L}F(t, y_0) = 0, \\ \mathcal{P}F(0, y_0) = \mathcal{P}f(0, y_0) - \mathcal{P}f(0, \hat{y}_0) = 0, \end{cases}$$

implying that $F$ is orthogonal to the image of the projector $\mathcal{P}$.

Hence, if the initial condition $y_0$ belongs to the space of observables, this contribution, known as the noise contribution, is null. A solution of (22) has been proved to exist, for Hamiltonian systems, in a classic sense for finite rank projectors and in a weak sense for projectors in the form of conditional expectations [22].

### *3.3. Application to reduced basis model reduction*

The formalism described in the previous Section represents a valuable tool for the closure of unresolved reduced-order models. To reproduce exactly the original HF problem (1) using the reduced POD-Galerkin equation (5), according to the projection error formula (7), a POD basis of size $N \gg m$ should be used to obtain results with machine precision accuracy. Numerical approximations of systems of that size are often not computationally practical. The long term accuracy is sacrified for efficiency by considering only the first $m \ll N$ elements of the POD basis, representing the resolved part of the simulation, and the effect on the dynamic of the resolved part of the remaining $N - m$ POD coefficients, the unresolved part, is truncated. However, even though unresolved POD coefficients are not relevant from a data compression point of view, they might be vital for the dynamic of the resolved part of the simulation, in particular at late times. Since POD is only a dimensionality-reduction technique, a possible correction is given by the Dynamic Mode Decomposition (DMD), which introduces accurate decompositions of complex dynamics into spatio-temporal coherent structures [37]. In our work we follow an approach based on the Mori-Zwanzig formalism (21), introduced in the previous Section, which provides the exact form of the truncated contribution. Previous work based on the application of Mori-Zwanzig formalism to the spectral approximations of the solution of PDE systems makes the assumption that the noise term $F(t, \alpha_0)$, due to the unresolved part of the initial condition, is small enough to be neglected. In case of a reduced basis approach, if the initial condition $u_0$ is not affected by the parameter $\omega$, choosing the reference value $\bar{u} = u_0$ guarantees that $F(t, \alpha_0) = 0$, $\forall\, t$. Hence, the dynamical system describing the exact evolution of the first $m$ POD coefficients is given by

$$\begin{cases} \dfrac{d}{dt}\alpha(t;\omega) = V^T f(t, V\alpha + \bar{u};\omega) + \displaystyle\int_0^t K(t-s, \alpha(s;\omega);\omega)\, ds, \\ \alpha(0;\omega) = 0. \end{cases} \tag{23}$$

As previously stated, solving (23) directly might be as expensive as solving the original HF model and a surrogate model is required. Gouasmi et al. [25] proposed an algorithm to estimate the memory kernel $K(t-s, \alpha(s);\omega)$, under the assumption that the composition property (17) holds for the orthogonal dynamic operator $e^{t\mathcal{QL}}$. Even though this algorithm is not practical for the online stage of model order reduction, it provides a picture of the interaction between the unresolved and the resolved part of the simulation. In Figure 1 (a) the normalized memory kernel $K(t-s, \alpha(s);\omega)$ is shown in case of model reduction of viscous Burgers' equation for different values of $m$ with a fixed $N = 68$. The initial condition is a sinusoidal function. We notice that $K(t-s, \alpha(s);\omega)$ decays exponentially with the time distance $t-s$, suggesting that the unresolved POD coefficients have a very fast decaying temporal correlations. Moreover, increasing $m$ produces an increase in the rate of decay of the kernel. We stress the fact that the short length of the unresolved correlation, and the resulting short support of $K(t-s, \alpha(s);\omega)$, depends on the choice of the modal representation framework and on the definition of a resolved set of coefficients [57]. Numerical experiments suggest that this property is satisfied by the hierarchical representation provided by POD and Fourier basis [39, 52].

Defining $\tau \in \mathbb{R}^+$ as the length of the memory kernel support, we have

$$\mathcal{M}(t,\alpha;\omega) = \int_0^t K(t-s,\alpha(s;\omega);\omega)\, ds \overset{(a)}{\approx} \int_{t-\tau}^t K(t-s,\alpha(s;\omega);\omega)\, ds \overset{(b)}{\approx} \frac{1}{2}\tau K(0,\alpha(t;\omega);\omega) \tag{24}$$

where we have used the short memory assumption $(a)$ and the trapezoidal rule $(b)$ to approximate the memory integral. Model (24) is known as the $\tau$-model and represents an extension of the $t$-model to short-memory systems of ODEs. This model can also be obtained, in relation to the observation of the exponentially decaying memory kernel, by forcing the memory kernel to have an exponential behaviour governed by the positive definite matrix $A \in \mathbb{R}^{m\times m}$, i.e.,

$$\int_0^t K(t-s,\alpha(s;\omega);\omega)\, ds \approx \int_0^t e^{-A(t-s)} K(0,\alpha(t;\omega);\omega)\, ds.$$

The $\tau$-model is obtained by setting $A = -c(\tau)\mathbb{I}$, where $c(\tau)$ is a function of $\tau$ and $\mathbb{I} \in \mathbb{R}^{m\times m}$ is the identity matrix. From a modeling point of view, this implies that memory kernels, related to different modes, decay independently from others and at the same rate. Even though this might be an oversimplification of the physics behind the problem, this hypothesis is often satisfied. Consider the previous example regarding the Burgers' equation: in Figure 1 (b) we notice that the time-average of the memory length is not strongly influenced by the mode considered once the size $m$ of the resolved part of the simulation is fixed.

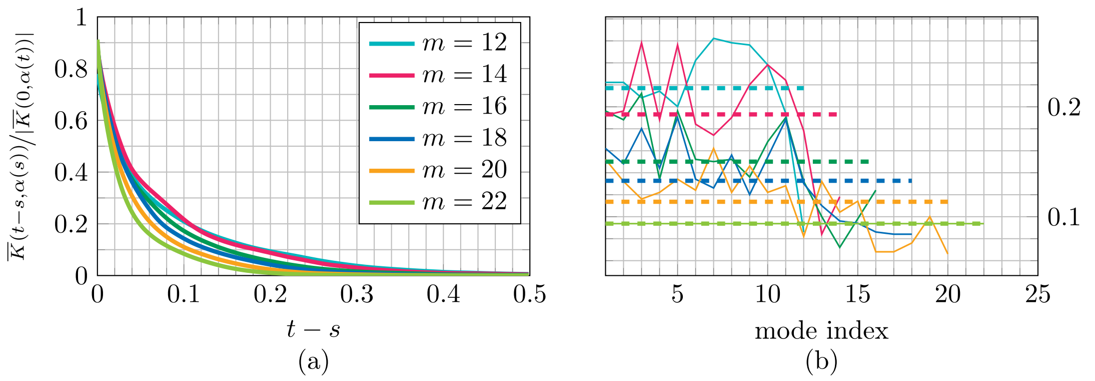


Figure 1: (a) Normalized average of the memory kernel for the POD-Galerkin model for the Burgers equation over $t$ and over the entire set of resolved POD coefficients for different values of $m$. (b) Dashed lines - Average over $t$ and over the entire set of resolved POD coefficients of the memory length $\tau$. Continuous lines - Average over $t$ of the memory length $\tau$.

Estimates of the memory length $\tau$ are used in this work for the hyper-parameters selection process. For problems that do not exhibit scale separation [51] , $\tau$ has been successfully modeled as a linear function of the Jacobian of the RHS in (5). A different technique to dynamically update the estimate of the memory length is based on the introduction of a sharp cutoff filter and on the Germano identity [48], with the additional task of modeling the relation of $\tau$ at different scales $m$. In our work we consider the least-squares solution, over the entire set of resolved POD coefficients and time, to the over-determined system (24). We define

$$\mathcal{M}_{\text{col}} = \begin{bmatrix} \mathcal{M}(0,\alpha;\omega) \\ \mathcal{M}(\Delta t,\alpha;\omega) \\ \dots \\ \mathcal{M}(T,\alpha;\omega) \end{bmatrix}, \; K_{\text{col}} = \begin{bmatrix} K(0,\alpha(0;\omega);\omega) \\ K(0,\alpha(\Delta t;\omega);\omega) \\ \dots \\ K(0,\alpha(T;\omega);\omega) \end{bmatrix},$$

where $\Delta t$ is the step used for the time integration, and hence, an approximation $\bar{\tau}$ of the memory length $\tau$ is given by

$$\bar{\tau} \approx \frac{2K_{\text{col}}^T \mathcal{M}_{\text{col}}}{K_{\text{col}}^T K_{\text{col}}}. \tag{25}$$

The scalar equation (25) is then solved for each value of the parameter $\omega$ in the training set.

## 4. Recurrent neural network memory model

This section presents the modeling of the memory effect using recurrent neural networks (RNNs). A conditioned long short term memory (LSTM) network is used to predict the memory integral, given a short history of the reduced basis solution. The structure and training of the network will be presented in the remainder of this section.

### *4.1. Regression of the memory term*

By the short memory assumption in Section 3.3, the memory integral $\mathcal{M}(t, \alpha; \omega)$ can be approximated using a short history of the reduced basis solution as

$$\begin{aligned}\mathcal{M}(t_n, \alpha; \omega) &= \int_0^{t_n} K(t_n - s, \alpha(s; \omega); \omega)\, ds \\ &\approx \int_{t_n-\tau}^{t_n} K(t_n - s, \alpha(s; \omega); \omega)\, ds \\ &\approx \mathcal{M}^{num}(\alpha_{n-n_{ts}+1}, \cdots, \alpha_{n-1}, \alpha_n; \omega),\end{aligned}$$

where $n_{ts}$ is the number of time steps, $\tau = n_{ts}\Delta t$ is the memory length and $\mathcal{M}^{num}$ is a numerical memory model that serves as the approximation of the map

$$(\alpha_{n-n_{ts}+1}, \cdots, \alpha_{n-1}, \alpha_n; \omega) \mapsto \mathcal{M}(t_n, \alpha; \omega) \tag{26}$$

The construction of the map in (26) is a regression task. In this section, we use an artificial neural network (ANN) as the regression model of the memory integral. The ANN seeks to predict the memory integral, given a sequence of the reduced coefficients.

### *4.2. Conditioned long short-term memory network*

Recurrent neural networks (RNNs) are a class of neural networks suitable for *sequential modeling* [24]. Hence, RNN is a natural choice for the memory integral regression which is a "many to one" sequential modeling task. In our work, the *long short-term memory* (LSTM) [29], one of the most popular gated RNNs that are developed to address the exploding/vanishing gradient issue that can be encountered when training traditional RNNs [24], is selected as the basic RNN structure for memory modeling.

A challenge in the design of the LSTM network for memory modeling is how to incorporate the physical/geometrical parameters. During the online stage of the model reduction of a parametrized system, we need to predict the memory term at arbitrary parameter location. Therefore, the LSTM network needs to be conditioned by the non-temporal physical/geometrical parameters. There are several existing works on feeding non-temporal data into the RNN. In the encoder-decoder network for machine translation [14, 61], the final state of the encoder LSTM network is set as the initial state of the decoder LSTM network. In the image caption network [34], the output of the convolutional neural network (CNN) is feed into the hidden state of the first gated recurrent unit (GRU). Inspired by these works, we condition the LSTM network by feeding the physical/geometrical parameters into the initial hidden state. The architecture of the conditoned LSTM network is shown in Figure 2.

As shown in Figure 2, the conditioned LSTM network consists of one dense layer, one LSTM layer and one another dense layer. If we denote the number of hidden units of the LSTM as $n_{hu}$, the first dense layer maps the parameter vector $\omega \in \mathbb{R}^d$ to the initial hidden state $h_0 \in \mathbb{R}^{n_{hu}}$, the LSTM layer maps the input sequence $\{x_1, x_2, \ldots, x_{n_{ts}}\} \in \mathbb{R}^{n_{ts} \times m}$ to the hidden states $\{h_1, h_2, \ldots, h_{n_{ts}}\} \in \mathbb{R}^{n_{ts} \times n_{hu}}$ and the second dense layer maps the final hidden state $h_{n_{ts}} \in \mathbb{R}^{n_{hu}}$ to the output $y \in \mathbb{R}^m$.

The initial hidden state $h_0$ is the output of the first dense layer, which has no bias and uses a linear (identity) activation function, i.e.,

$$h_0 = W_{h_0}\omega. \tag{27}$$

The initial cell state $c_0$ is set as zero. The forward propagation of the conditioned LSTM network is achieved by iterating the following recurrent relation for $t = 1$ to $n_{ts}$:

$$\begin{aligned} f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \\ i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \\ o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \\ \tilde{c}_t &= \sigma\left(W_c x_t + U_c h_{t-1} + b_c\right) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\ h_t &= o_t \circ \tanh\left(c_t\right) \end{aligned}$$

where $\sigma$ is the hard sigmoid activation function

$$\sigma(z) = \begin{cases} 0, & z < -2.5, \\ 0.2z + 0.5, & -2.5 \leq z \leq 2.5, \\ 1, & z > 2.5. \end{cases}$$

The final output of the network is obtained through the second dense layer with a linear (identity) activation function, i.e.,

$$y = W_y h_{n_{ts}} + b_y. \tag{28}$$

The matrices

$$W_{h_0} \in \mathbb{R}^{n_{hu} \times d}, W = \begin{pmatrix} W_f \\ W_i \\ W_o \\ W_c \end{pmatrix} \in \mathbb{R}^{4n_{hu} \times m}, U = \begin{pmatrix} U_f \\ U_i \\ U_o \\ U_c \end{pmatrix} \in \mathbb{R}^{4n_{hu} \times n_{hu}}, W_y \in \mathbb{R}^{m \times n_{hu}},$$

and vectors

$$b = \begin{pmatrix} b_f \\ b_i \\ b_o \\ b_c \end{pmatrix} \in \mathbb{R}^{4n_{hu}}, b_y \in \mathbb{R}^m,$$

are the trainable weights and biases that can be adjusted in the training process to achieve an optimal network configuration.

For the conditioned LSTM network for memory modeling, the input is the sequence of the reduced coefficients $x = \{\alpha_{n-n_{ts}+1}, \cdots, \alpha_n\}$, the auxiliary input is the physical/geometrical parameter vector $\omega$, and the output is the predicted memory integral $\mathcal{M}^{LSTM}\left(\alpha_{n-n_{ts}+1}, \cdots, \alpha_n; \omega\right)$. The conditioned LSTM network is an approximation of the map in (26).

### *4.3. Training of the network*

The training of the conditioned LSTM network is a *supervised learning* [24] task. In supervised learning, labeled data is used to train the network. The goal of the training is to minimize the difference between the predicted output $\mathcal{M}^{LSTM}$ and the desired output $\mathcal{M}$.

A training data set $\mathcal{D}_{tr} = \{\left(\left(\omega, x\right), \mathcal{M}\right)_i\}_{1 \leq i \leq N_{tr}}$ and a validation data set $\mathcal{D}_{va} = \{\left(\left(\omega, x\right), \mathcal{M}\right)_i\}_{1 \leq i \leq N_{va}}$ are used in the training. Here $(\omega, x)$ is the *input object*, $\mathcal{M}$ is the *desired output*. The training data is collected from high-fidelity simulations with uniformly sampled parameter values, and the validation data is collected from high-fidelity simulations with randomly sampled parameter values. The data generation process is described in Algorithm 1.

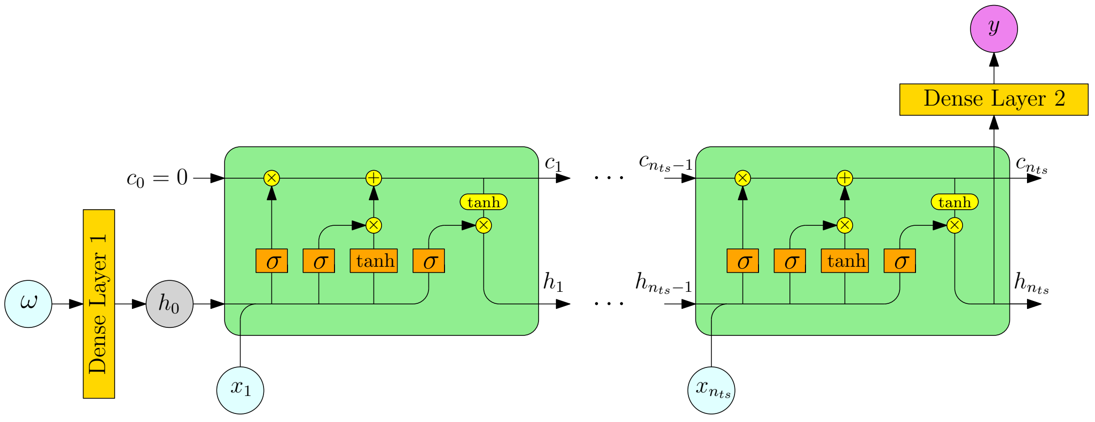


Figure 2: Conditioned LSTM network.

A component of the input pattern is called a *feature*. The *feature scaling* technique in which all the features are scaled to the same range, can be applied to the data sets to accelerate the training process [32]. In this paper, a feature $\chi$ is scaled by the mean normalization

$$\tilde{\chi} = \frac{\chi - \bar{\chi}}{\sigma_\chi},$$

where $\bar{\chi}$ and $\sigma_\chi$ are the mean and standard deviation of $\chi$, respectively.

The training of the network is implemented in Keras [15], with TensorFlow [1] as the backend. The optimal weights and biases of the network are obtained using the *Adam stochastic optimizer* [35], which uses *mini-batches* of size $N_b < N_{tr}$ of the training data to take a single optimization step by minimizing the loss function. More precisely, the full training data set with $N_{tr}$ data-points is shuffled, and $N_{tr}/N_b$ mini-batches are extracted to take $N_{tr}/N_b$ optimization steps. Once the entire training data set is exhausted, the training is said to complete one training epoch. The training is performed for a sufficient number of epochs to obtain a converged network. The convergence speed of the training is controlled by the *learning rate* $\eta$.

For the training in this paper, the loss function is the mean absolute error (MAE). To avoid possible overfitting, a *weight regularization* term that is the sum of the $L_2$-norm penalties of $W_{h_0}$, $W$, $U$ and $W_y$, is added to the loss function. The weight regularization effect is controlled by a hyper-parameter $\lambda$.

At the beginning of the training, the weights and biases of the network are randomly initialized using normal distributions [23]. Therefore, the training needs to be performed several times, following a *multiple restarts* approach [30], to prevent the training results from depending on the initialization of the weights. In this paper, ten restarts are performed for the training of each network, and the trained model with the best validation accuracy is selected as the final model. The validation accuracy metric is the mean squared error (MSE).

### *4.4. Model selection*

For a certain problem, the size of the conditioned LSTM mainly depends on the number of hidden units $n_{hu}$ and number of time steps $n_{ts}$. Therefore, we need a strategy to select a trained network with a proper combination of $n_{hu}$ and $n_{ts}$ as the regression model for the memory integral.

Given enough training data, more hidden units requires a larger network, resulting in a higher generalization accuracy. However, this is not the case for the number of time steps. As described in Section 3, every problem has a certain range of memory lengths. The network with too small number of time steps, corresponding to a too short memory length, does not have enough information to

**Algorithm 1** Labeled data generation for the training of the conditioned LSTM RNN.

```
1: function $\mathcal{D}$=MEMORY_DATA_GENERATION($V, \Gamma, f, n_{ts}$)
2:     $u(i,j)_{i=1,\cdots,p,j=1,\cdots,q} \leftarrow$ HIGH_FIDELITY($f, \Gamma$)  ▷ Generate snapshots with parameter set $\Gamma$
3:     $\mathcal{D} = \emptyset$
4:     for $j \leftarrow 1, q$ do
5:         $w \leftarrow \Gamma(j)$
6:         for $i \leftarrow 1, p$ do
7:             $t \leftarrow i\Delta t$
8:             $u_h \leftarrow u(i,j)$
9:             $\alpha(i) \leftarrow V^T (u_h - \bar{u})$  ▷ Project snapshot onto reduced space
10:            $\mathcal{M} \leftarrow V^T f(t, u_h; \omega) - V^T f(t, V\alpha(i) + \bar{u}; \omega)$  ▷ Compute exact memory term
11:            if $i \geq n_{ts}$ then
12:                $\mathcal{D} \leftarrow \mathcal{D} \cup \{\omega, \alpha(i - n_{ts} + 1), \cdots, \alpha(i), \mathcal{M}\}$  ▷ Collect training data
13:            end if
14:        end for
15:    end for
16: end function
```

accurately predict the memory integral; while the network with too many time steps, corresponding to a too long memory length, also can not accurately predict the memory integral since the hidden map between the input and output is too complicated and requires a larger network. Therefore, a suitable pair of $(n_{hu}, n_{ts})$ should be found to have a balance between the accuracy and cost.

In this paper, we train networks with different a number of time steps and hidden units. We select the the most accurate model from the models of which the memory lengths lie between the minimum and maximum values estimated by the method in (25).

## 5. Parametric POD-Galerkin with the RNN memory model

This section presents the implementation of the RNN memory model in the framework of parametric POD-Galerkin model order reduction.

### *5.1. POD-Galerkin with memory*

The POD-Galerkin reduced-order model is

$$\frac{d}{dt}\alpha(t;\omega) = V^T f(t, V\alpha + \bar{u}; \omega) = \tilde{f}(t, \alpha; \omega), \tag{29}$$

where $\alpha$ is the reduced coefficient vector and $\tilde{f}$ is the RHS term. A *memory closure* term $\mathcal{M}$ is added to the RHS, which results in a reduced-order model

$$\frac{d}{dt}\alpha(t,\omega) = \tilde{f}(t, \alpha; \omega) + \mathcal{M}(t, \alpha; \omega). \tag{30}$$

The motivation for the introduction of the memory term into the reduced-order model is to account for the effect of the unresolved POD modes on the resolved POD modes, which can improve the accuracy and stability of the reduced-order model.

The mechanism of the memory effect for the POD-Galerkin reduced-order model is sketched in Figure 3. By multiplying the reduced-order model of (30) by $2\alpha(t;\omega)^T$, we obtain the energy evolution equation

$$\frac{d}{dt}\alpha^T\alpha(t,\omega) = 2\alpha^T \tilde{f}(t, \alpha; \omega) + 2\alpha^T \mathcal{M}(t, \alpha; \omega), \tag{31}$$

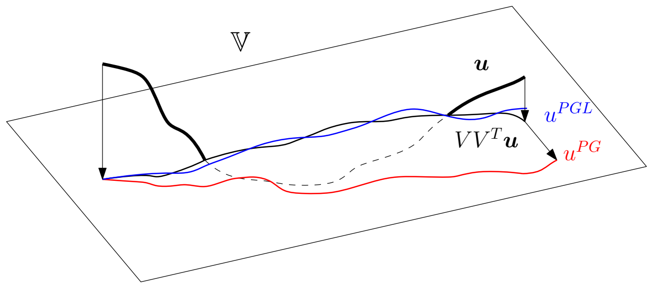


Figure 3: ▬ Solution $\boldsymbol{u}$ to the HF model; — Projection on $\mathbb{V}$ of $\boldsymbol{u}$; — Solution $u^{PG}$ to the POD-Galerkin reduced model; — Solution $u^{PGL}$ to the POD-Galerkin reduced model with memory closure based on LSTM network.

in which the second term in RHS describes the *energy exchange* between the resolved scales and the unresolved scales, which plays the same role as the subgrid-scale-stress (SGS) in a large-eddy simulation (LES). The introduction of the memory closure seeks to reduce the difference between the reduced basis solution and the projection of the high-fidelity solution onto the reduced space, by providing the missing dynamics, caused by omitting the unresolved POD modes, in the energy exchange term. With the memory closure, the trajectory of the reduced basis solution can follow the trajectory of the projection of the high-fidelity solution. The projection of the high-fidelity solution is the upper limit of the reduced basis solution in terms of accuracy.

## *5.2. Implicit-explicit Runge-Kutta time integration*

The POD-Galerkin model with memory closure suffers from a high computational cost, since the evaluation of the memory model is expensive, which is true for the conditioned LSTM as well as other existing memory models. In explicit or implicit time stepping, such as the Runge-Kutta (RK) method, the memory model needs to be evaluated in each stage or inner iteration step, leading to substantial additional computational cost. Therefore, the reduced-order model with the RNN memory closure is usually more expensive than the original reduced-order model.

An efficient implementation of the conditioned LSTM memory model in the POD-Galerkin framework is proposed in this paper to address this efficiency issue. The *implicit-explicit Runge-Kutta* (IMEX-RK) scheme is used to advance the reduced-order model in time. More specifically, the RHS $\tilde{f}$ is integrated using the diagonally implicit Runge-Kutta (DIRK) scheme and the memory term $m$ is integrated using the explicit Runge-Kutta (ERK) scheme.

The $s$-stage IMEX RK scheme is

$$\alpha_n^{(i)} = \alpha_n + \Delta t \sum_{j=1}^{i} a_{ij} \tilde{f}\left(t_n + c_j \Delta t, \alpha_n^{(j)}; \omega\right) + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} \mathcal{M}\left(t_n + \hat{c}_j \Delta t, \alpha^{(j)}; \omega\right), i = 1, \cdots, s, \tag{32}$$

$$\alpha_{n+1} = \alpha_n + \Delta t \sum_{i=1}^{s} b_i \tilde{f}\left(t_n + c_i \Delta t, \alpha_n^{(i)}; \omega\right) + \Delta t \sum_{i=1}^{s} \hat{b}_i \mathcal{M}\left(t_n + \hat{c}_i \Delta t, \alpha^{(i)}; \omega\right), \tag{33}$$

where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are the coefficients for the DIRK and $\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}$ are the coefficients for the ERK, defined by the following Butcher tableaux

$$\begin{array}{c|ccccc}
0 & 0 & 0 & \cdots & 0 & 0 \\
c_2 & a_{21} & a_{22} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
c_{s-1} & a_{s-1,1} & a_{s-1,2} & \cdots & a_{s-1,s-1} & 0 \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & a_{ss} \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}, \quad
\begin{array}{c|ccccc}
0 & 0 & 0 & \cdots & 0 & 0 \\
c_2 & \hat{a}_{21} & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
c_{s-1} & \hat{a}_{s-1,1} & \hat{a}_{s-1,2} & \cdots & 0 & 0 \\
c_s & \hat{a}_{s1} & \hat{a}_{s2} & \cdots & \hat{a}_{s,s-1} & 0 \\
\hline
 & \hat{b}_1 & \hat{b}_2 & \cdots & \hat{b}_{s-1} & \hat{b}_s
\end{array},$$

with the constraint

$$c_i = \sum_{j=1}^{i} a_{ij} = \sum_{j=1}^{i-1} \hat{a}_{ij}, \quad i = 1, \cdots, s.$$

The memory term $\mathcal{M}\left(t_n + \hat{c}_j \Delta t, \alpha^{(j)}\right)$ is computed as

$$\mathcal{M}\left(t_n + \hat{c}_j \Delta t, \alpha^{(j)}; \omega\right) = \mathcal{M}^{LSTM}\left(\alpha^{(j)}_{n-n_{ts}+1}, \cdots, \alpha^{(j)}_{n}; \omega\right). \tag{34}$$

For linear systems, (32) is solved directly; while for nonlinear systems, (32) is solved iteratively, e.g., through a Newton iteration. The detailed implementation of the POD-Galerkin with memory closure is described in Algorithm 2.

For nonlinear problems, in each stage of the IMEX-RK, the memory term needs to be computed only once, while the inner iteration needs to be performed until the residual reaches a certain threshold. Therefore, most computational time is consumed by this inner iteration. Numerical results for nonlinear problems demonstrate that the POD-Galerkin with memory is much more efficient than the original POD-Galerkin, since the introduction of the memory closure into the IMEX-RK leads to significant accuracy improvement and only small extra computational cost.

**Algorithm 2** Implicit-Explicit Runge-Kutta time integration of POD-Galerkin with memory.

```
1: function $\alpha(n+1)$=IMEX($\alpha(n), \tilde{f}, \mathcal{M}^{LSTM}, n_{ts}, \omega$)
2:     $t \leftarrow n\Delta t$
3:     $\alpha_{rk}(1,n) \leftarrow \alpha(n)$
4:     for $i \leftarrow 2, s$ do
5:         $mem(i-1) \leftarrow \mathcal{M}^{LSTM}(\alpha_{rk}(i-1, n-n_{ts}+1), \cdots, \alpha_{rk}(i-1,n); \omega)$    ▷ Evaluate the conditioned LSTM memory model
6:         $rhs \leftarrow \alpha(n) + \Delta t \sum_{j=1}^{i-1} a(i,j)\tilde{f}\,(t + c(j)\Delta t, \alpha_h(j); \omega) + \Delta t \sum_{j=1}^{i-1} \hat{a}(i,j) mem(j)$
7:         $\alpha_{rk}(i,n) \leftarrow$ SOLVE($\alpha_{rk}(i,n) - a(i,i)\Delta t \tilde{f}(t + c(i)\Delta t, \alpha_{rk}(i,n); \omega) - rhs = 0$)    ▷ Inner iteration
8:     end for
9: end function
```

The IMEX-RK schemes used in the numerical experiments in Section 6 are:

(1) IMEX-Euler

$$\alpha_{n+1} = \alpha_n + \Delta t f(t_{n+1}, \alpha_{n+1}; \omega) + \mathcal{M}(t_n, \alpha; \omega) \tag{35}$$

The corresponding Butcher tableaux are

$$\begin{array}{c|cc}
0 & 0 & 0 \\
1 & 0 & 1 \\
\hline
 & 0 & 1
\end{array}, \quad
\begin{array}{c|cc}
0 & 0 & 0 \\
1 & 1 & 0 \\
\hline
 & 1 & 0
\end{array}.$$

The IMEX-Euler scheme is first-order accurate.

(2) IMEX-Trapezoidal

$$\tilde{\alpha}_{n+1} = \alpha_n + \frac{\Delta t}{2}\left(f(t_n, \alpha_n; \omega) + f(t_{n+1}, \tilde{\alpha}_{n+1}; \omega)\right) + \Delta t \mathcal{M}(t_n, \alpha; \omega), \tag{36}$$

$$\alpha_{n+1} = \alpha_n + \frac{\Delta t}{2}\left(f(t_n, \alpha_n; \omega) + f(t_{n+1}, \tilde{\alpha}_{n+1}; \omega)\right) + \frac{\Delta t}{2}\left(\mathcal{M}(t_n, \alpha; \omega) + \mathcal{M}(t_{n+1}, \tilde{\alpha}; \omega)\right) \tag{37}$$

The corresponding Butcher tableaux are

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1/2 & 1/2 \\ \hline & 1/2 & 1/2 \end{array}, \quad \begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array}.$$

The IMEX-Trapezoidal scheme is second-order accurate.

## 6. Numerical results

This section presents the numerical results of the POD-Galerkin with the RNN memory closure for model reduction of a 3D Stokes flow, the 1D Kuramoto–Sivashinsky (KS) equation, and the 2D Rayleigh-Bénard convection. The linear 3D Stokes flow problem is used to validate the method. The 1D Kuramoto–Sivashinsky (KS) equation and 2D Rayleigh-Bénard convection problems are used to test the accuracy and efficiency of the POD-Galerkin with the RNN memory closure for nonlinear problems.

The following two metrics are used to measure the generalization accuracy of the trained network:

(1) the average relative error of the memory terms on the test data set:

$$\varepsilon_1 = \frac{1}{N_{te}} \sum_{i=1}^{N_{te}} \frac{||\mathcal{M}_i^{LSTM} - \mathcal{M}_i||_2}{||\mathcal{M}_i||_2},$$

(2) the relative error of the entire memory data set:

$$\varepsilon_2 = \sqrt{\frac{\sum_{i=1}^{N_{te}} ||\mathcal{M}_i^{LSTM} - \mathcal{M}_i||_2^2}{\sum_{i=1}^{N_{te}} ||\mathcal{M}_i||_2^2}},$$

where $N_{te}$ is the size of the test data set.

The accuracy of the reduced basis solution is measured by the following $L_2$ norm error [54]:

$$||\tilde{u} - u||_{L_2(0,T;L_2)} = \sqrt{\int_0^T ||\tilde{u} - u||_2^2 \quad dt}$$

where $\tilde{u}$ is the reduced basis solution and $u$ is the high-fidelity solution.

The following notations are used in some plots to distinguish results for different methods:

(1) *Full-Order*: The high-fidelity (full-order) solution;

(2) *Projection*: Projection of the high-fidelity solution onto the reduced space;

(3) *POD-Galerkin*: Reduced basis solution of the POD-Galerkin method;

(4) *LSTM*: Reduced basis solution of the POD-Galerkin with the conditioned LSTM memory closure.

## *6.1. 3D Stokes*

The POD-Galerkin with the RNN memory closure is applied to the model reduction of the blood flow in the human cardiovascular system [55]. The 3D Stokes equations

$$\begin{cases} \dfrac{\partial}{\partial t}\mathbf{u} = \nu \Delta \mathbf{u} - \nabla p, \quad t \in [0,4] \\ \nabla \cdot \mathbf{u} = 0, \\ \mathbf{u}(\partial\Omega_{\text{inlet}}, t) = f(t), \\ \mathbf{u}(\partial\Omega_{\text{wall}}, t) = 0, \\ \mathbf{u}\left(\Omega, 0\right) = 0, \end{cases} \tag{38}$$

are used to describe the blood flow. The computational domain $\Omega$ is shown in Figure 4. The velocity is $\mathbf{u} = \left[u_x, u_y, u_z\right]^\top$, where $u_x, u_y$ and $u_z$ are velocity components and $p$ is the pressure. The surface of the domain $\Omega$ consists of one velocity inlet, two free outlets and the no-slip wall. The boundary condition at the inlet is $\mathbf{f}\left(t\right) = [0, 0, f_z(t)]$, where $f_z$ is the time-dependent boundary value of $u_z$. The function $f_z(t)$ is shown in Figure 5.

Following the Mori-Zwanzig formalism, each variable of interest needs to be evaluated using a dynamical equation. Therefore, the continuity equation in (38)

$$\nabla \cdot \mathbf{u} = 0,$$

is replaced by

$$\beta \frac{\partial}{\partial t} p + \nabla \cdot \mathbf{u} = 0,$$

following the artificial compressibility method [47]. $\beta$ is the compressibility parameter, and is set as $\beta = 10^6$ in this test case.

The full-order model is discretized using the finite element method. The Taylor-Hood $P2 - P1$ finite elements are used to satisfy the inf-sup condition, with 20,914 nodes for each velocity component. To obtain a problem in the general dynamical system form in (1), we condense the mass matrix into a diagonal matrix using mass lumping and multiply the resulting system by the inverse of the mass matrix. The semi-discrete form of (38) is

$$\begin{cases} \frac{d}{dt}\mathbf{u}_h = A\left(\mu\right)\mathbf{u}_h + \mathbf{b}\left(t\right), \quad t \in [0,4] \\ \mathbf{u}_h\left(0\right) = \mathbf{u}_{h,0} \end{cases} \tag{39}$$

The implicit Euler method is used to integrate (39) in time, using $N_t = 1000$ time steps.

The coefficient $\nu$ is the only physical/geometrical parameter of this problem. The range of $\nu$ is $[2, 6]$. The snapshots for POD basis generation are collected from the high-fidelity simulations with 9 values of $\nu$ that are uniformly sampled. The leading 100 singular values of the snapshot matrix are plotted in Figure 6. The first 16 left singular vectors of the snapshot matrix are selected as the reduced basis functions.

The training, validation, and test data sets are generated from the high-fidelity simulation results, with 50 uniformly, 25 randomly, and 25 randomly sampled parameter values, respectively.

To get an accurate memory model, we train the conditioned LSTMs with 32, 64 and 128 hidden units and $2, 3, \cdots, 8$ time steps. Ten restarts are performed in each training, with 500 epochs in each restart. In each epoch, the training data is shuffled and divided into mini-batches of size of 1000. The learning rate is 0.005. The coefficient of the $L_2$ regularization is $10^{-9}$.

For model selection, we show the test errors of the trained networks with different number of time steps and hidden units, and the estimated memory length in Figure 7. The accuracy comparison in Figure 7 shows that the models with more hidden units are more accurate but also more costly. The models with 4 time steps are the most accurate among the models of which the memory lengths

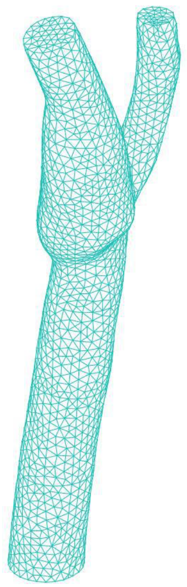


Figure 4: 3D carotid bifurcation geometry model and mesh.

are inside the estimated range. Therefore, the network with 128 hidden units and 4 time steps is selected as the memory model for further test.

The POD-Galerkin with the memory model is tested with the 25 random values of $\nu$ of the test data set. The simulations are performed using the IMEX-Euler time integration until $t = 8$, which is beyond the range of the training data $[0, 4]$, to test the prediction capability of the reduced-order model.

The energy contribution of the closure to the reduced-order system is shown in Figure 8 to provide a intuition of the accuracy of the memory model. It is observed in Figure 8 that the conditioned LSTM network accurately models the memory closure.

The error between the reduced solutions of the original POD-Galerkin and the POD-Galerkin with memory for 4 different parameter values are shown Figure 9. The results show that the error of the POD-Galerkin with memory is 2 to 3 orders of magnitude smaller than the original POD-Galerkin, which demonstrates that the LSTM memory model can significantly improve the accuracy of the POD-Galerkin method. We highlight the fact that in the prediction interval, compared to the original POD-Galerkin, the POD-Galerkin with memory closure also provides much more accurate solutions.

It is also observed in the numerical test that, for some small values of $\nu$, the POD-Galerkin is unstable. The solutions of $\nu = 3$ are shown in Figure 10 to give an example of instability of POD-Galerkin. The results in Figure 10 show that the solution of the POD-Galerkin with memory closure is quite close to the high-fidelity solution, while the solution of the original POD-Galerkin diverges. Hence, the memory closure improves not only the accuracy, but also the stability of the reduced-order model.

The error-cost plot of the reduced-order models is shown in Figure 11 for efficiency comparison. We can observe that the original POD-Galerkin model can reach a certain accuracy level using less computational time and is thus more efficient. As discussed in Section 5.2, we do not expect efficiency improvement for linear problems, because of the cost of the memory model.

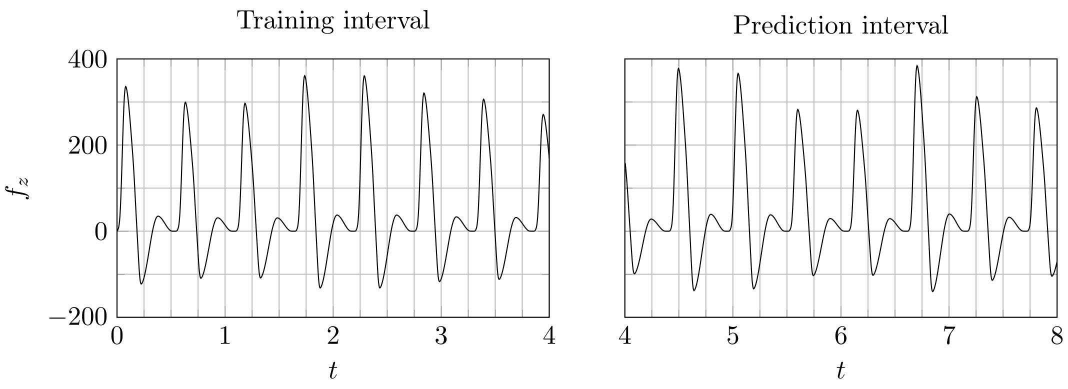


Figure 5: Time-dependent velocity in $z$-direction at the inlet boundary.

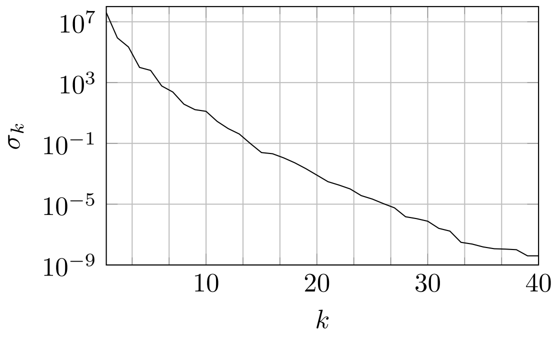


Figure 6: Singular value decay of the snapshot matrix $S$ in logarithmic scale. The spectrum shows a very fast decay, which suggests that a reduced basis with less than 20 elements is enough to represent the high-fidelity solution with a reasonable accuracy.

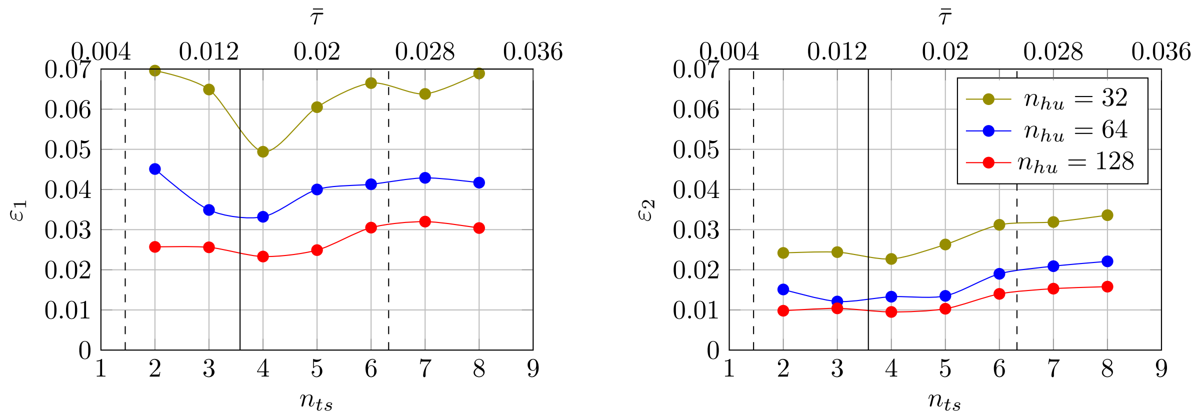


Figure 7: Test error of the trained networks for the Stokes problem. The estimated mean (solid vertical line), minimum and maximum (dashed vertical lines) of the memory length are also shown in the plot.

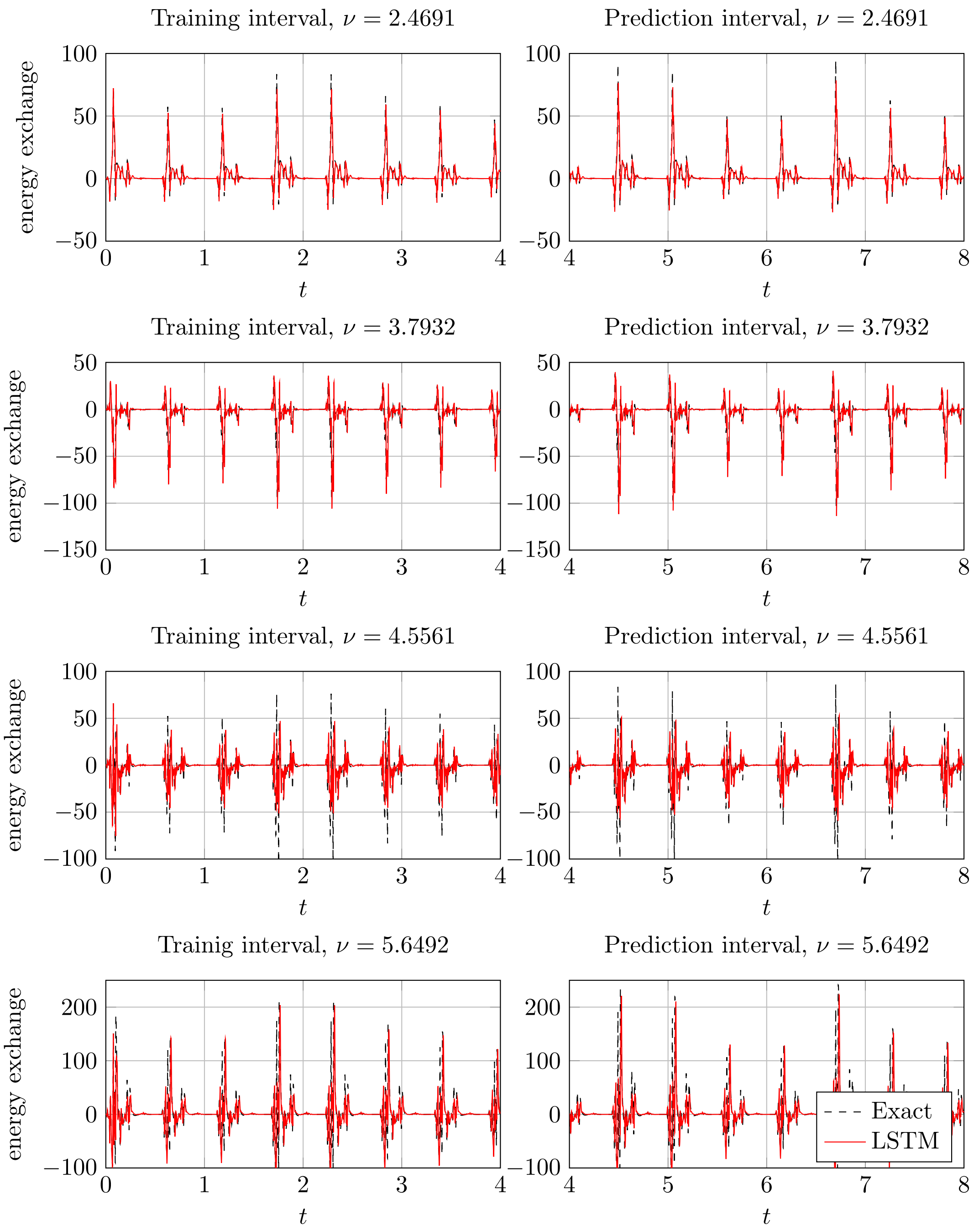


Figure 8: Evolution of the energy exchange term $2\alpha^\top\mathcal{M}$ of the reduced-order models of the Stokes problem.

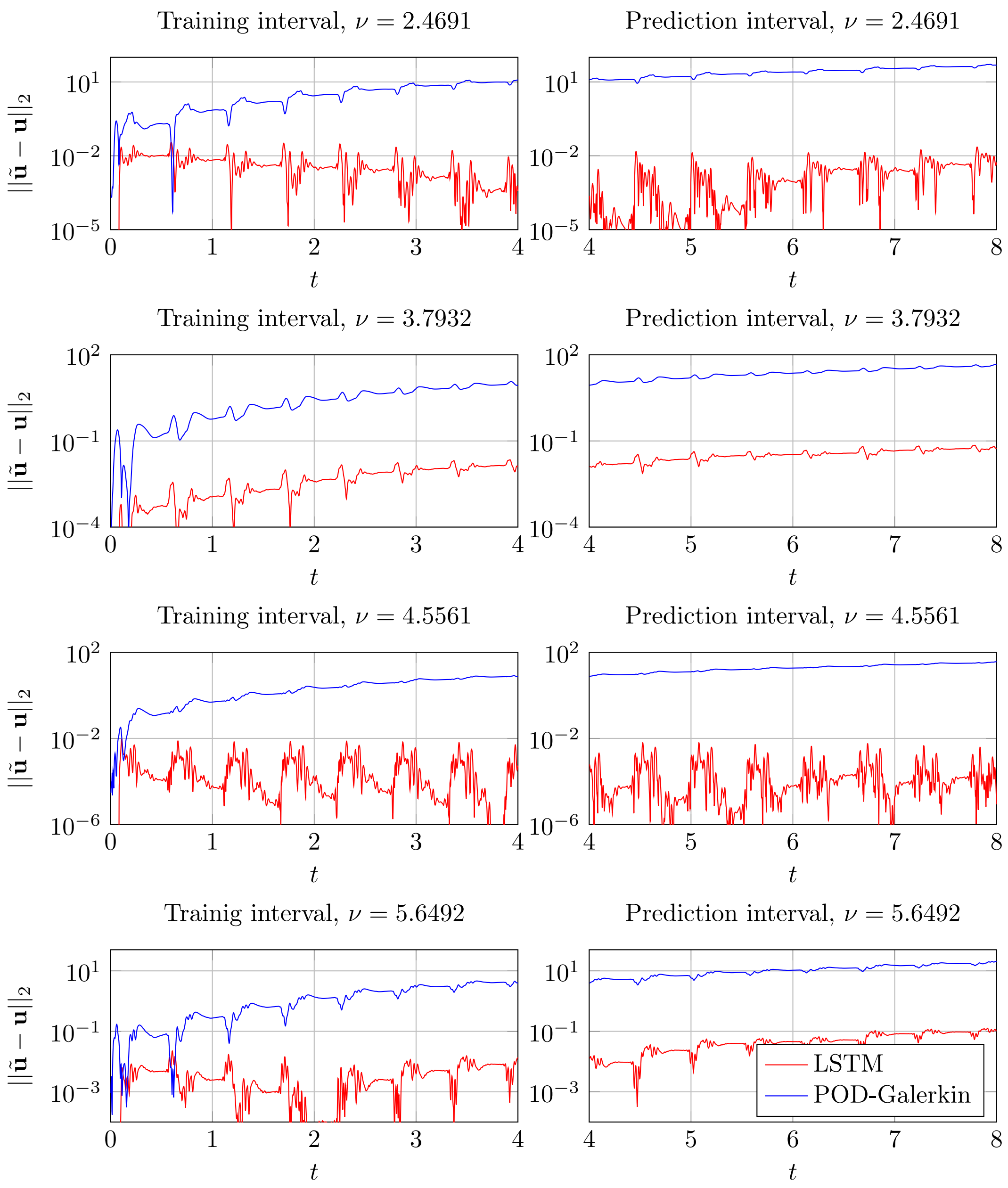
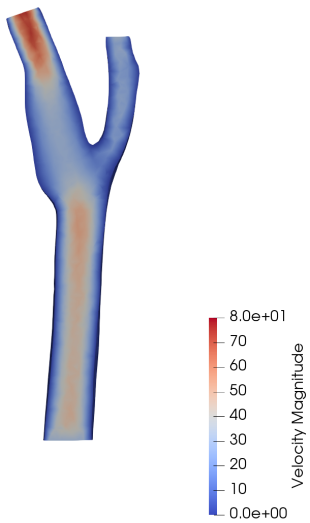
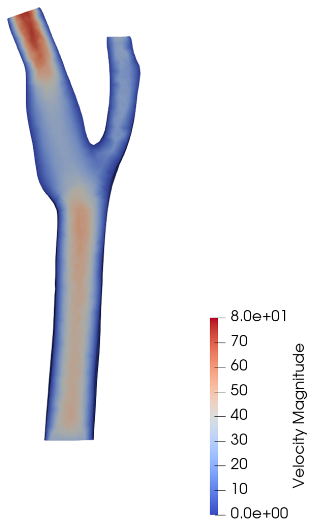


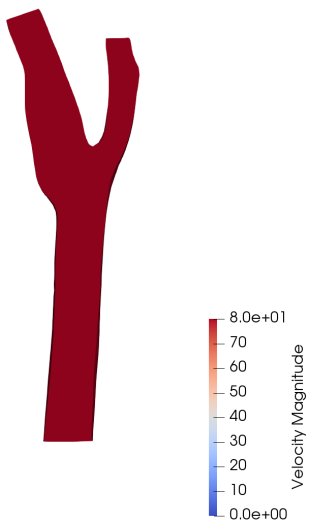
Figure 9: Evolution of the errors of the reduced-order models of the Stokes problem.



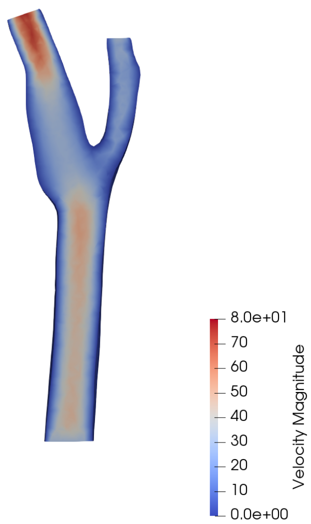
(a) Full-Order



(b) Projection



(c) POD-Galerkin



(d) LSTM

Figure 10: A sectional view of the velocity magnitude contours of the reduced-order solutions at $t = 2.4$ for $\nu = 3$.

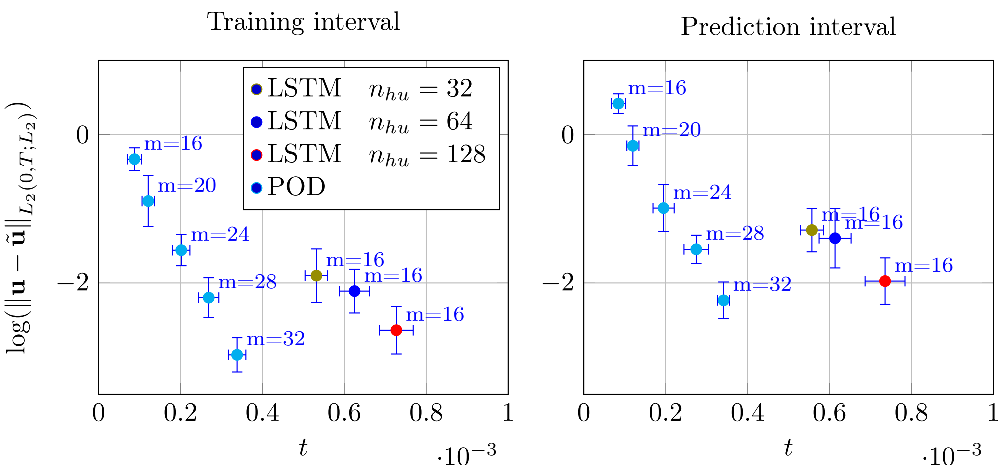


Figure 11: Error-cost plot of the reduced-order models of the Stokes problem.

## *6.2. Kuramoto–Sivashinsky equation*

The fourth-order one-dimensional Kuramoto–Sivashinsky (KS) equation is used to test the memory modeling capability of the conditioned LSTM network for highly nonlinear problems. The parametrized KS equation is

$$\begin{cases} \frac{\partial}{\partial t}u = -\frac{1}{2}\nabla \cdot u^2 - \Delta u - \nu\Delta^2 u, \\ u(x+L,t) = u(x,t), \\ u(x,0) = g(x), \end{cases} \tag{40}$$

where $L$ is the spatial period, $g(x)$ is the initial datum, and $\nu$ is the parameter. In our test, we take $L = 22$ and $g(x)$ is obtained by the inverse Fourier transform from a series of Fourier modes of which the first 4 mode coefficients are 0.06. The computational domain is partitioned into $N_h = 1024$ elements. The full-order solver utilizes a finite-element discretization and a second-order implicit trapezoidal time integration. The solution is updated until $t = 50$ using a time step size $\Delta t = 0.025$.

The coefficient $\nu$ of the fourth-order viscosity term is the only parameter in this problem. Following Lu et al. [40], the dissipative term $\Delta^2 u$ provides damping at small scales. Therefore, the smaller the $\nu$, the less dissipative the system. In our test, we see that very small $\nu$ yields a chaotic or quasi-chaotic system, making the model reduction difficult.

We set the range of the parameter $\nu$ as $[0.3, 1.5]$. The snapshots for the POD basis generation are collected from the high-fidelity simulations with 25 values of $\nu$ that are uniformly distributed in the log space, which means that more data points are sampled for small parameter values. The basis is extracted from the snapshots via POD. We chose 25 left singular vectors of the snapshot matrix as the reduced basis functions.

The training, validation, and test data sets are generated from high-fidelity simulation results, with 124 uniformly, 62 randomly, and 61 randomly sampled parameter values in the log space, respectively.

To get an accurate memory model, we train the conditioned LSTMs with 32, 64 and 128 hidden units and $2, 3, 4, 5, 6, 10$ time steps. Ten restarts are performed in each training, with 500 epochs in each restart. In each epoch, the training data is shuffled and divided into mini-batches of size of 1000. The learning rate is 0.005. The coefficient of the $L_2$ regularization is $10^{-9}$.

The relative errors of the trained networks on the test data set, and the estimated memory length are shown in Figure 12. We observe that the networks with 128 hidden units are the most accurate. The models with 3, 4 and 5 time steps are the most accurate among the models in which

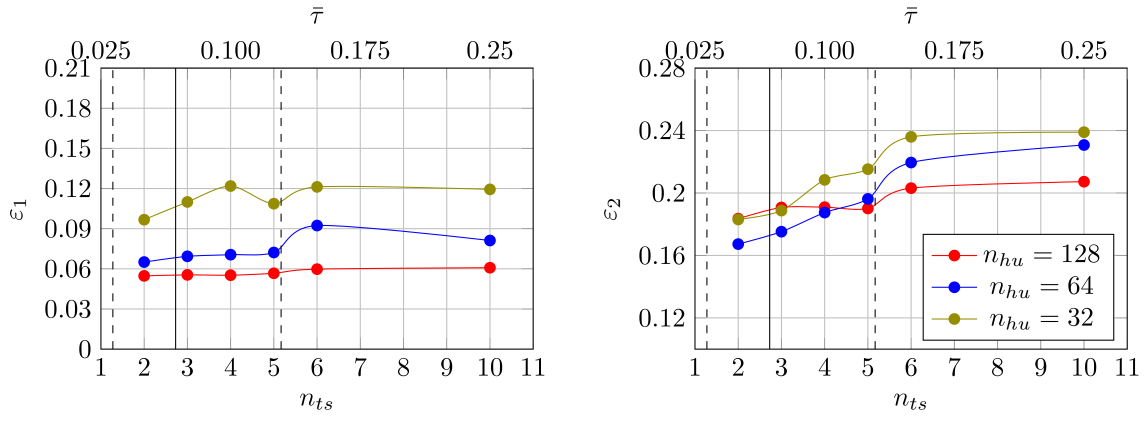


Figure 12: Test error of the trained networks for the KS equation. The estimated mean (solid vertical line), minimum and maximum (dashed vertical lines) of the memory length are also shown in the plot.

the memory length are inside the estimated range. However, we see in Figure 12 that the errors of the networks are quite large. To understand the cause of this large error, we show the test errors of the networks with 128 hidden units with respect to different parameter values in Figure 13. The results in Figure 13 show that the networks are very accurate for $\nu \in [0.6, 1.5]$, while less accurate for $\nu \in [0.3, 0.6]$, which is reasonable since the small values of $\nu$ correspond to quasi-chaotic dynamics that are very difficult to accurately capture. Furthermore, for the well modeled parameter range, the networks with different number of time steps have very similar accuracy. We chose the network with 128 hidden units and 4 time steps as the memory model.

The POD-Galerkin with the selected memory model is tested using the physical parameter sampling of the test data set which includes 61 random values of $\nu$. The simulations are performed using the IMEX-Trapezoidal time integration in (36) until $t = 100$, which is beyond the time range of the training data $[0, 50]$, to test the prediction capability of the reduced-order models.

The energy contribution of the closure to the reduced-order system is shown in Figure 14 to provide a intuition of the accuracy of the memory model. It is observed in Figure 14 that the conditioned LSTM network accurately models the memory closure.

The reduced solutions of the original POD-Galerkin and the POD-Galerkin with memory with 4 different parameter values are shown Figure 15-18. For the small parameter value ($\nu = 0.34756$) case, both the reduced-order models with/without memory models fails to follow the fast dynamics of the high-fidelity model. For the cases with large parameter values, the results show that the reduced basis solutions computed by the POD-Galerkin with memory are much closer to the high-fidelity solutions than the original POD-Galerkin. We note that, for certain values of the parameter, the POD-Galerkin reduced-order model is able to follow the high-fidelity solution only for a short time interval, as shown in Figure 18. While the reduced-order model with memory closure remains accurate for much longer time and make accurate predictions beyond the training time interval.

For a global view of the dynamics, the contours of solutions for $\nu = 0.64424$ on the $x - t$ plane are shown in Figure 19. The results in Figure 19 show that the POD-Galerkin with memory closure provides accurate solutions in both the training and prediction intervals, while the original POD-Galerkin model loses the dynamics after a certain time.

We show the error-cost plot of the reduced-order models in Figure 20 for efficiency comparison. We observe that, the POD-Galerkin model with memory has 10 to 100 times smaller error than the original POD-Galerkin model, with only slightly more computational time, and is thus much more efficient.

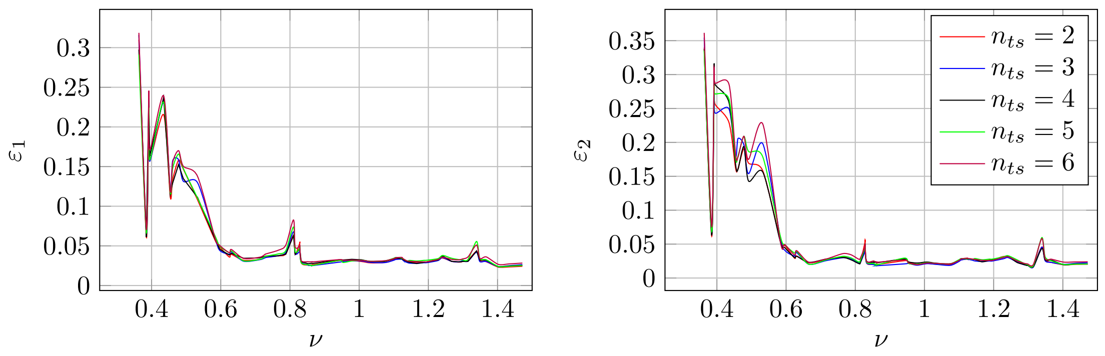


Figure 13: Test error plots of the trained networks with 128 hidden units for different parameter values.

## *6.3. Rayleigh-Bénard convection*

The two-dimensional Rayleigh-Bénard convection problem is used as the last case to test the capability of memory modeling of the conditioned LSTM network for multi-dimensional nonlinear problems. The non-dimensionalized governing equations are

$$\begin{cases} \nabla \cdot \mathbf{u} = 0, \\ \frac{\partial}{\partial t}\mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \sqrt{\frac{Pr}{Ra}}\Delta \mathbf{u} + T\mathbf{e}_y, \\ \frac{\partial}{\partial t}T + \mathbf{u} \cdot \nabla T = \frac{1}{\sqrt{PrRa}}\Delta T, \end{cases} \tag{41}$$

where $\mathbf{u}$, $T$ and $p$ are the dimensionless velocity, temperature and pressure, respectively. The Rayleigh number ($Ra$) and the Prandtl number ($Pr$) are the dimensionless quantities that control the flow. The simulation setup, including the computational domain and the boundary conditions, is shown in Figure 21. The high-fidelity simulations are performed until $t = 50$ on a triangular mesh with 152,888 nodes, using a finite-elment space discretization and the implicit trapezoidal time integration with a time step size $\Delta t = 0.01$. To make the problem reducible, the solution of a low Rayleigh number case $Ra = 33019.2725, Pr = 0.85$ at $t = 50$, starting from a stationary flow with linear temperature distribution between the hot and cold plates, is used as the initial condition for the high-fidelity simulations.

We select $Ra$ and $Pr$ as the two physical parameters for model reduction of this problem. The parameter domain of the problem is $(Ra, Pr) \in [5 \times 10^6, 1.5 \times 10^7] \times [0.85, 0.95]$. The snapshots for POD basis generation are collected from the high-fidelity simulations with 36 parameter values generated via the Latin hypercube sampling. The reduced basis functions are extracted from the snapshots using the randomized SVD [26]. The singular value decay is shown in Figure 22 and it is observed that the singular values decay slowly, implying that a large number of reduced bases is necessary to recover the main dynamics, making the model reduction of this problem being difficult. We select 24 left singular vectors of the snapshot matrix as the reduced basis functions.

The training data is generated using the same high-fidelity simulation results that are used for reduced basis generation. Both the validation and test data sets are obtained from high-fidelity simulations with 18 randomly sampled parameter values.

To obtain an accurate memory model, we train the conditioned LSTMs with 128 hidden units and $2, 3, \cdots, 8$ time steps. Each network is optimized by 10 restarts, with 1200 epochs in each restart. In each epoch, the training data is shuffled and divided into mini-batches of size 1000. The learning rate is 0.005. The coefficient $\lambda$ of the $L_2$ regularization is $10^{-9}$.

The relative errors of the trained networks on the test data set, and the estimated memory length are shown in Figure 23. It is shown in Figure 23 that the network with 3 time steps is the

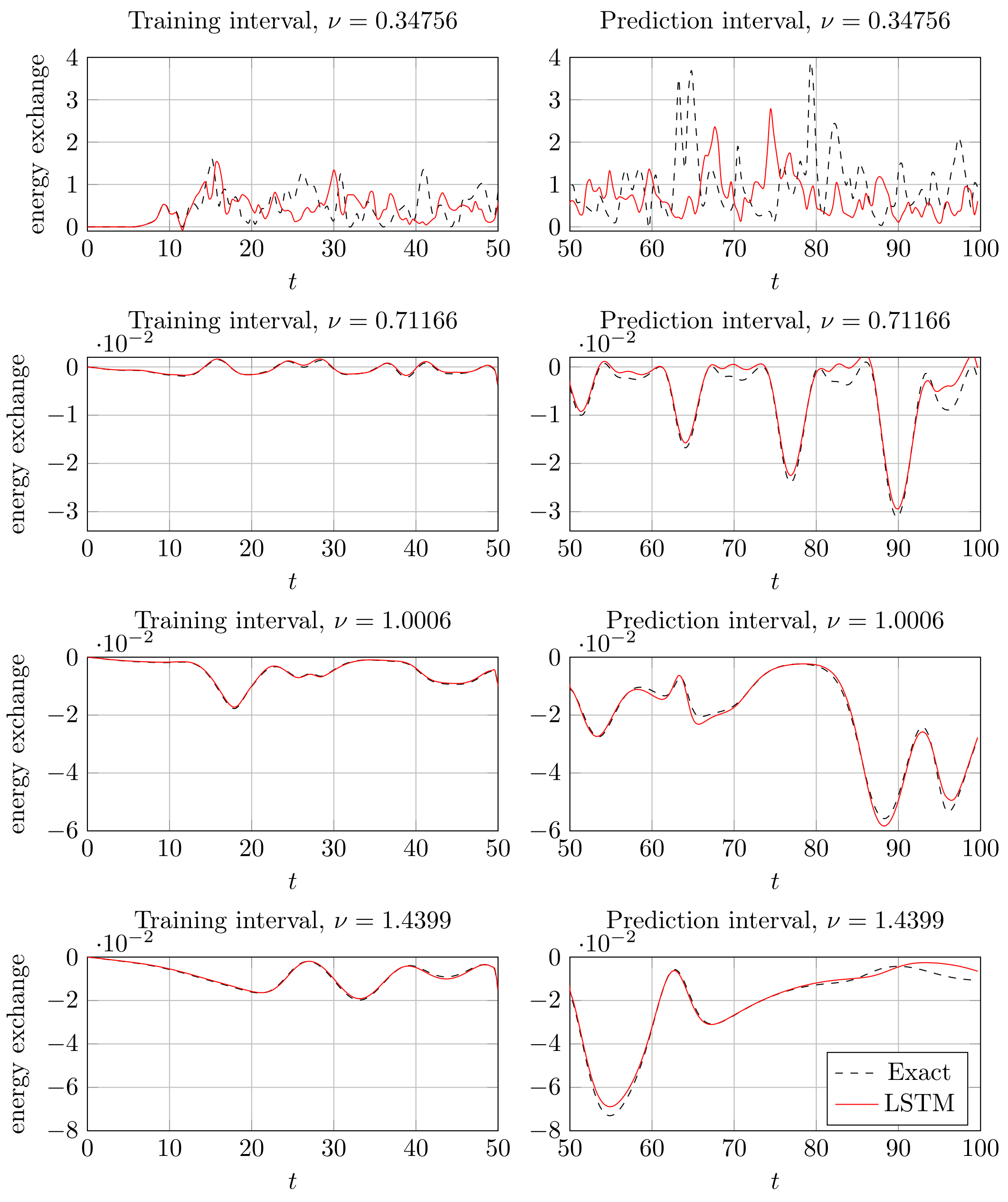


Figure 14: Evolution of the energy exchange term $2\alpha^{\top}\mathcal{M}$ of the reduced-order models of the KS equation.

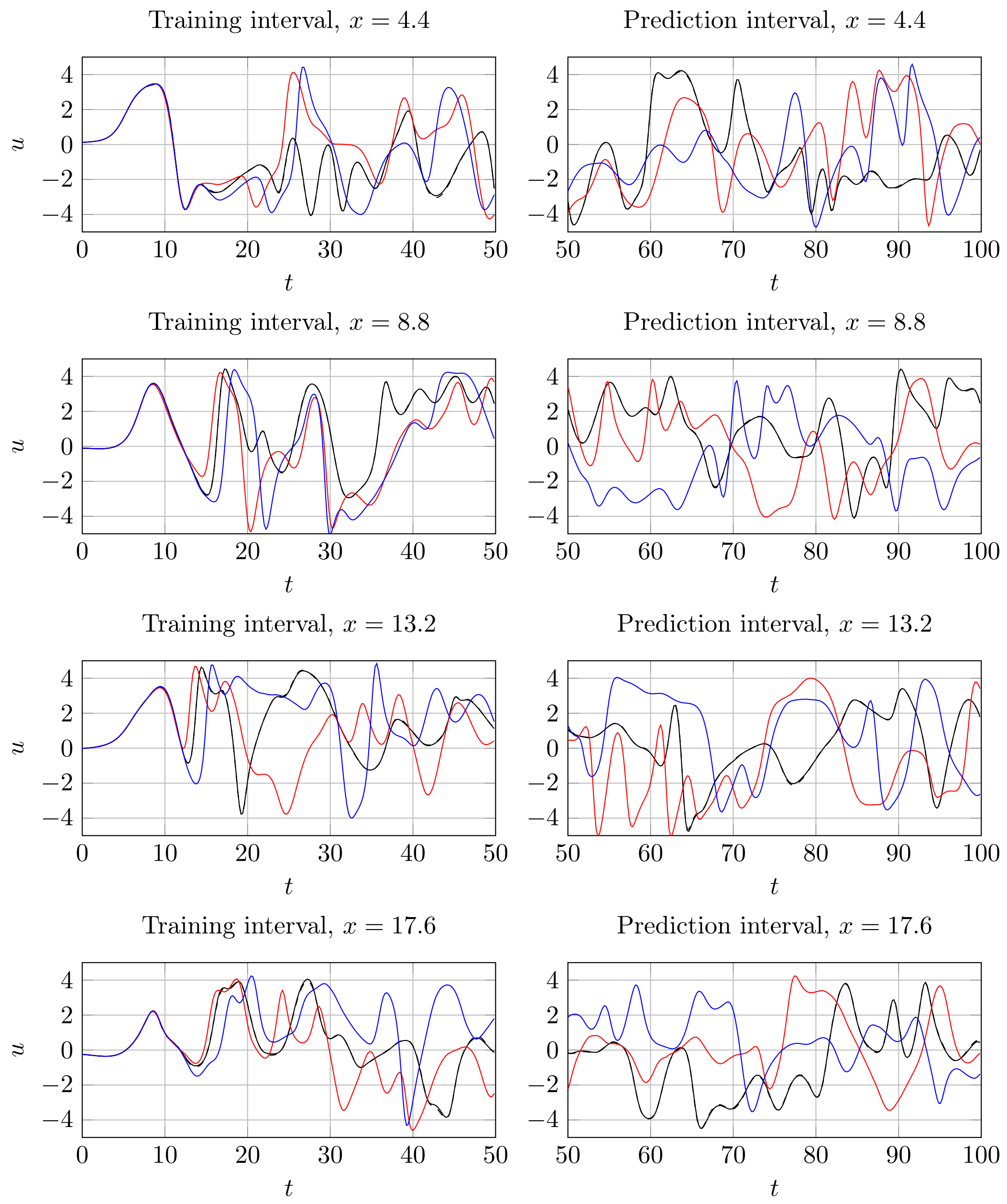


Figure 15: Numerical solutions for parameter $\nu = 0.34756$. ---- High-fidelity; —— Projection of high-fidelity; —— POD-Galerkin; —— POD-Galerkin with memory.

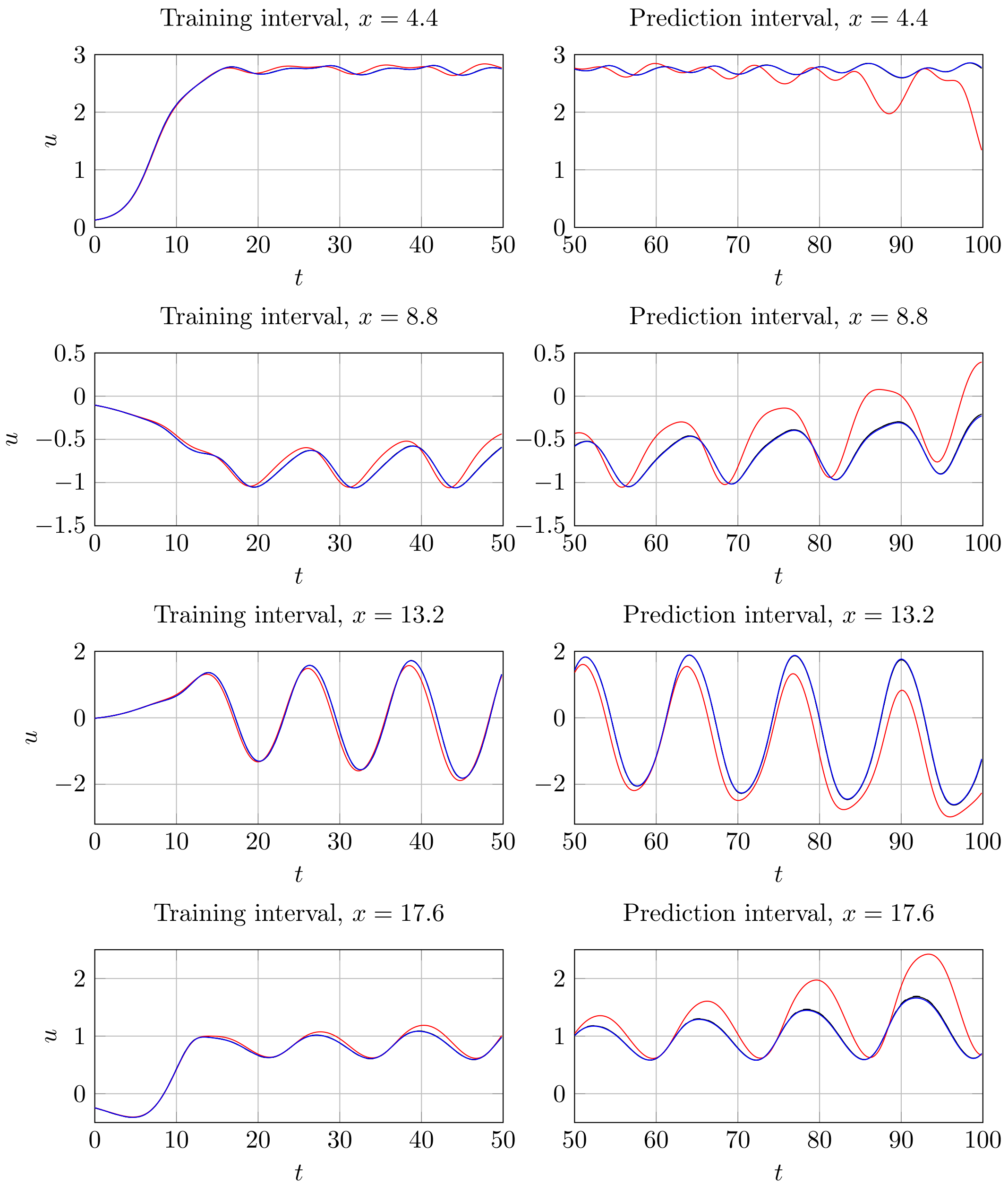


Figure 16: Numerical solutions for parameter $\nu = 0.71166$. - - - - High-fidelity; —— Projection of high-fidelity; —— POD-Galerkin; —— POD-Galerkin with memory.

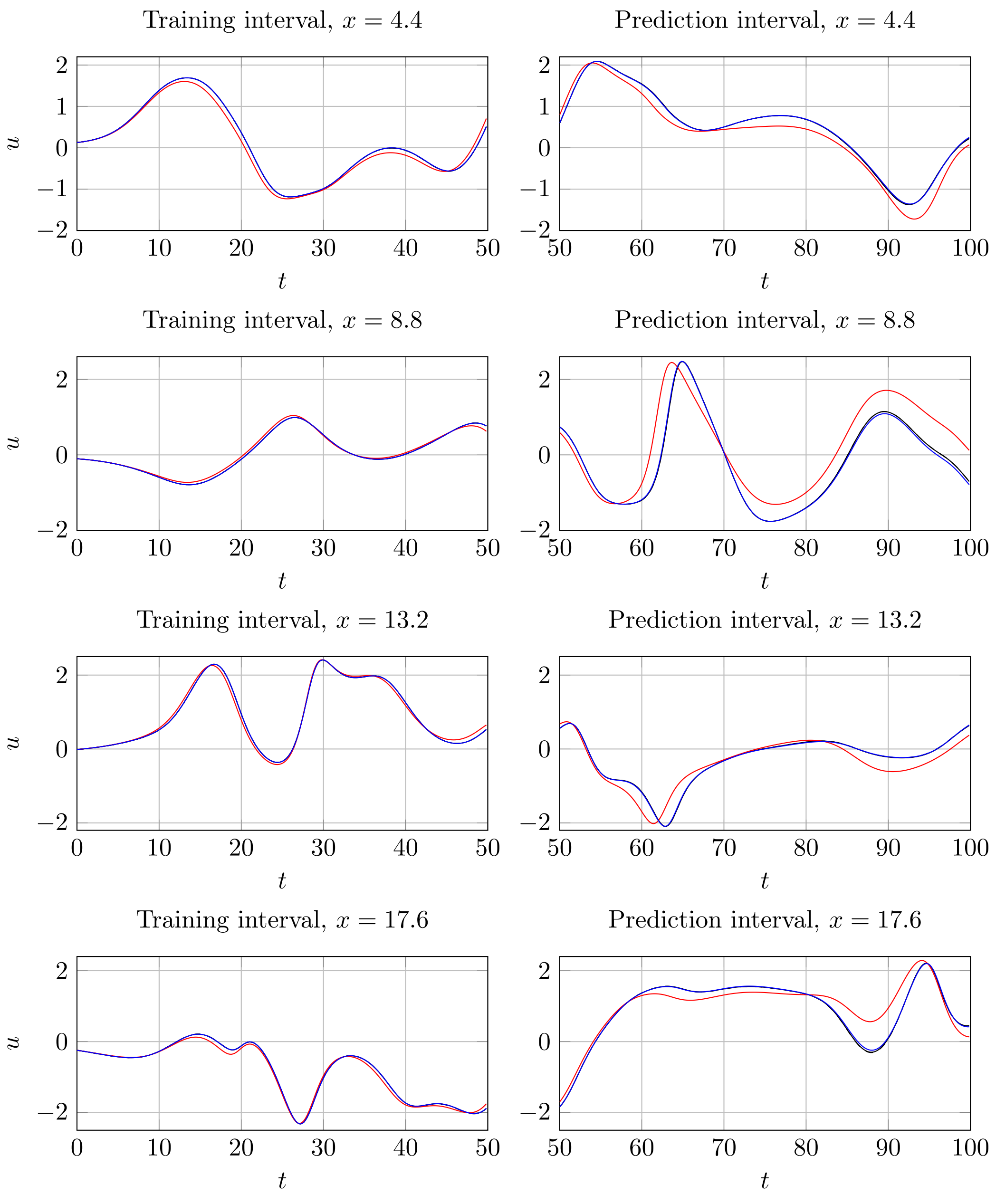


Figure 17: Numerical solutions for parameter $\nu = 1.0006$. ---- High-fidelity; —— Projection of high-fidelity; —— POD-Galerkin; —— POD-Galerkin with memory.

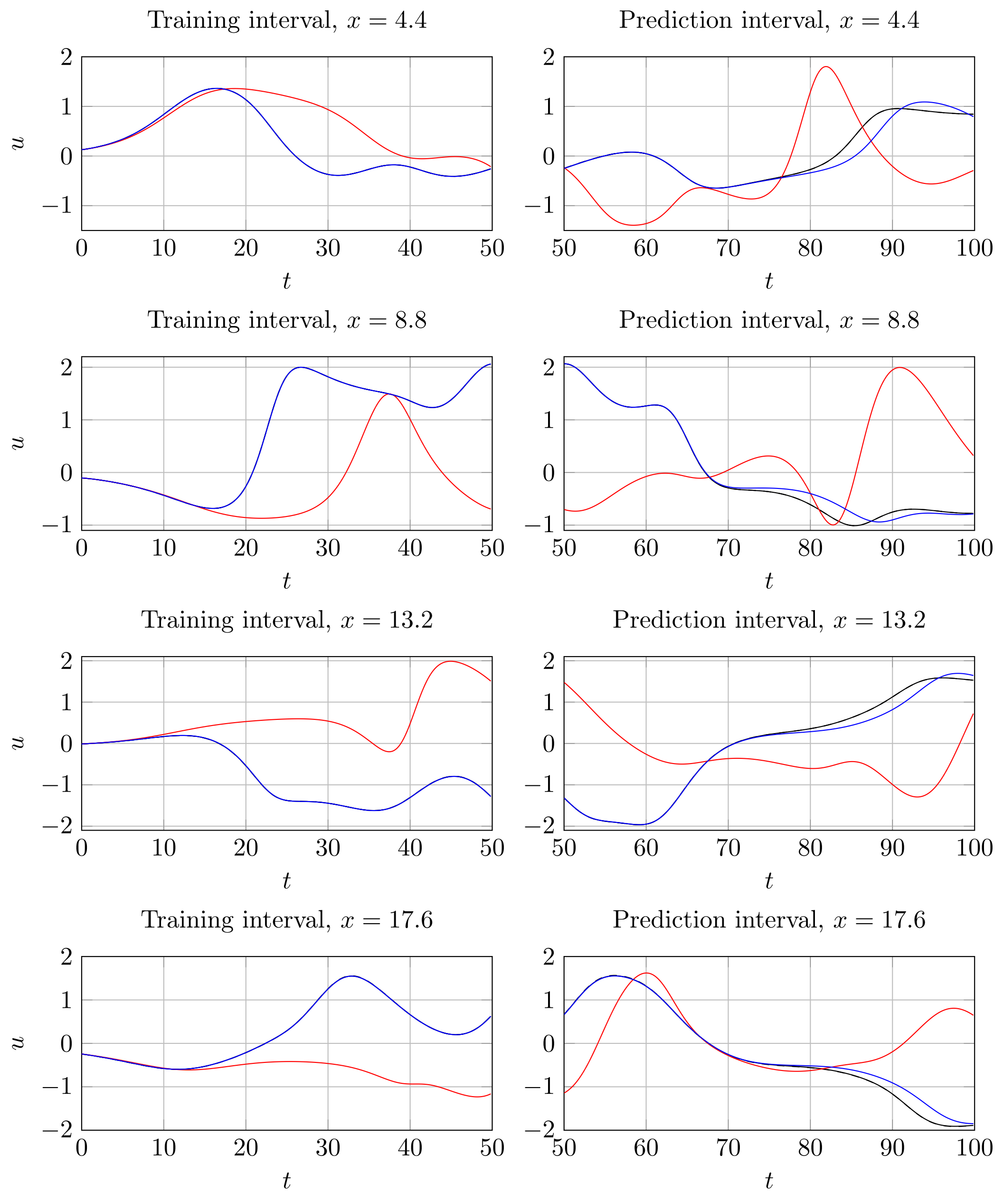


Figure 18: Numerical solutions for parameter $\nu = 1.4399$. - - - - High-fidelity; —— Projection of high-fidelity; —— POD-Galerkin; —— POD-Galerkin with memory.

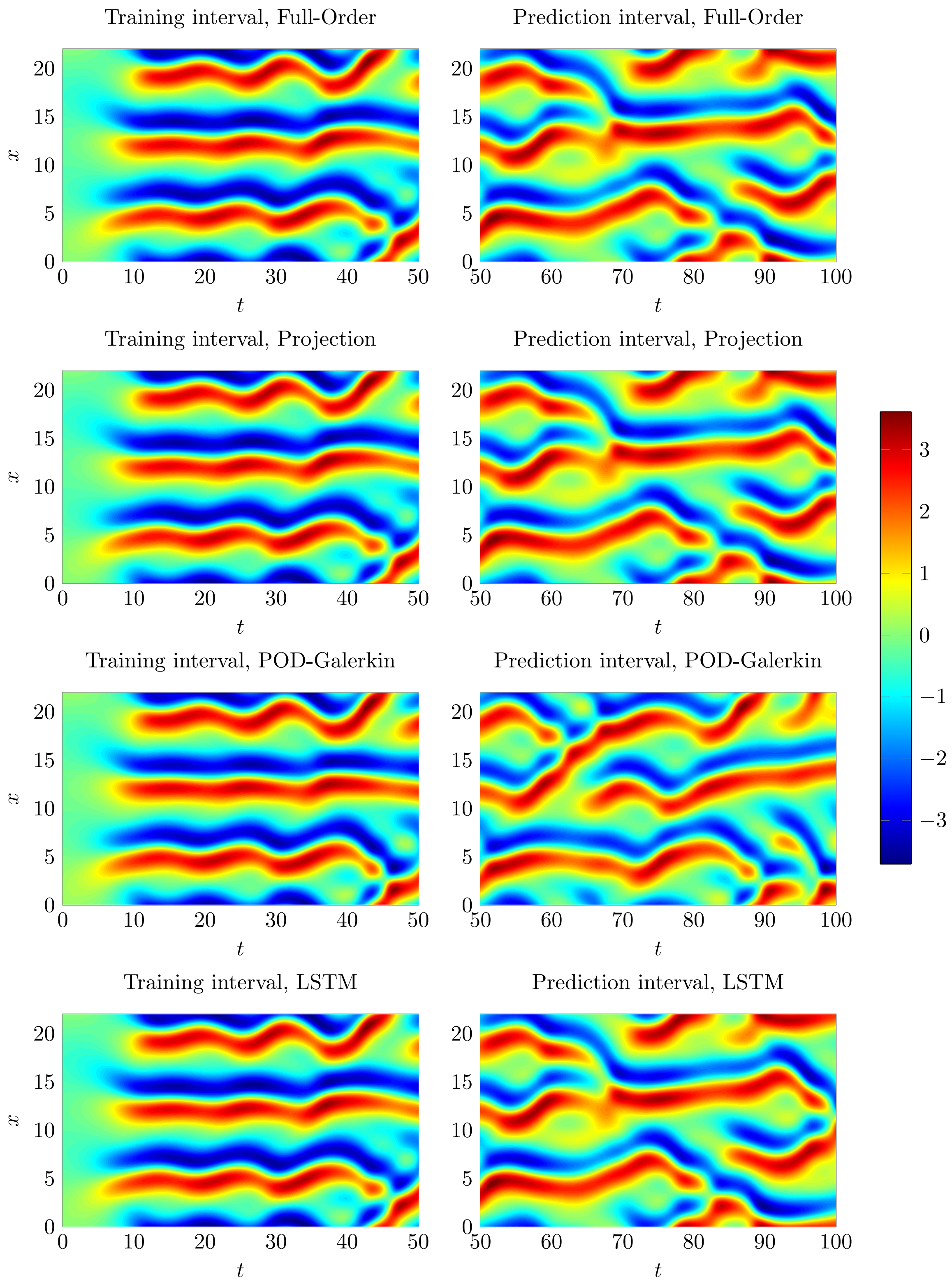


Figure 19: Evolution of the numerical solutions of the KS equation with $\nu = 0.64424$.

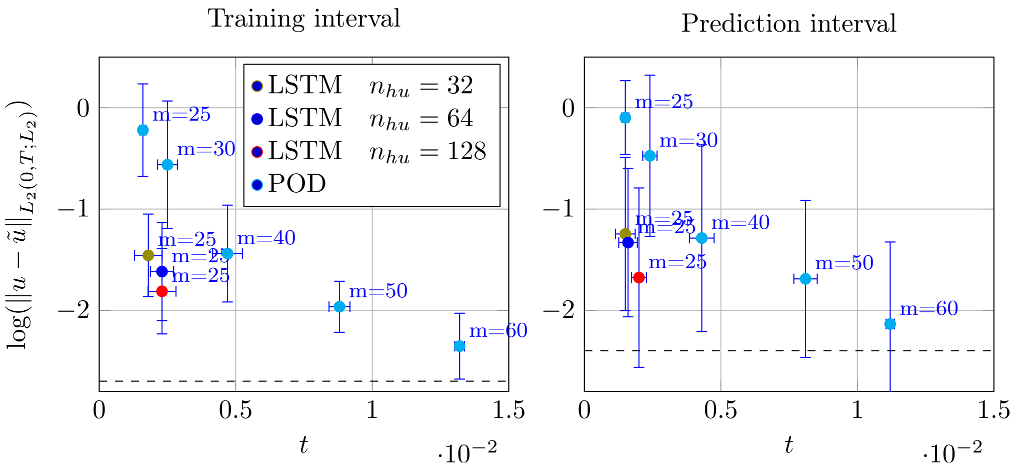


Figure 20: Error-cost plot of the reduced-order models of the KS equation. The dashed horizontal line represents the projection error.

most accurate among the models of which the memory length are inside the estimated range, and is thus selected as the memory model.

The POD-Galerkin with the selected memory model is tested using the physical parameter sampling of the test data set that includes 61 random values of $\nu$. The simulations are performed using the IMEX-Trapezoidal time integration until $t = 100$, which is beyond the time range of the training data $[0, 50]$, to test the prediction capability of the reduced-order model.

The energy contribution of the closure to the reduced-order system is shown in Figure 24 to provide a intuition of the accuracy of the memory model. It is observed in Figure 24 that the conditioned LSTM network accurately captures the memory closure.

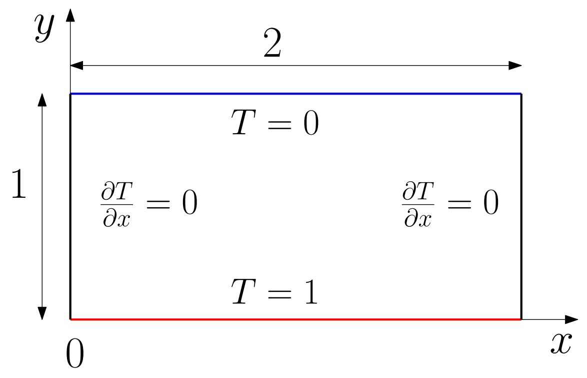
The reduced solutions of the original POD-Galerkin and the POD-Galerkin with memory for $(Ra, Pr) = (14024512.0002, 0.86976)$ are shown Figure 25. The results show that the reduced solutions computed by the POD-Galerkin with memory are much closer to the high-fidelity solutions. Furthermore, the results also show that the POD-Galerkin model with memory closure can make accurate predictions, even in the case that the reduced basis can not accurately represent the dynamics of the high-fidelity solution.

For a global view of the dynamics, the contours of solutions for $(Ra, Pr) = (14024512.0002, 0.86976)$ are shown in Figure 26-29. The results show that the POD-Galerkin with memory closure can provide much more accurate solutions in both the training and prediction intervals than the original POD-Galerkin model.

We show the error-cost plot of the reduced-order models in Figure 30 for efficiency comparison. We can see from Figure 30 that, the POD-Galerkin model with memory has 5 to 10 times smaller error than the original POD-Galerkin model, at only slightly more computational cost, and is thus much more efficient. We highlight the fact that the error of the POD-Galerkin model with memory is very close to the projection error, which means that the conditioned LSTM memory model is very accurate and the reduced basis solution evolves with almost exact dynamics.

## 7. Conclusions

This paper proposes a RNN closure for parametric POD-Galerkin reduced-order model. A conditioned LSTM network is used to predict the memory integral that accounts for the impact of the unresolved scales on the resolved scales, given the physical/geometrical parameter values and the short time history of the resolved scales. The RNN closure is embedded into the POD-Galerkin



$y$
2
1
$T = 0$
$\frac{\partial T}{\partial x} = 0$
$\frac{\partial T}{\partial x} = 0$
$T = 1$
0
$x$

Figure 21: Simulation setup of the Rayleigh-Bénard convection.

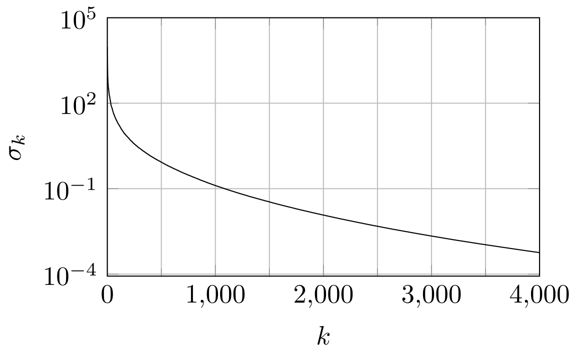


Figure 22: Singular values of the Rayleigh-Bénard convection problem.

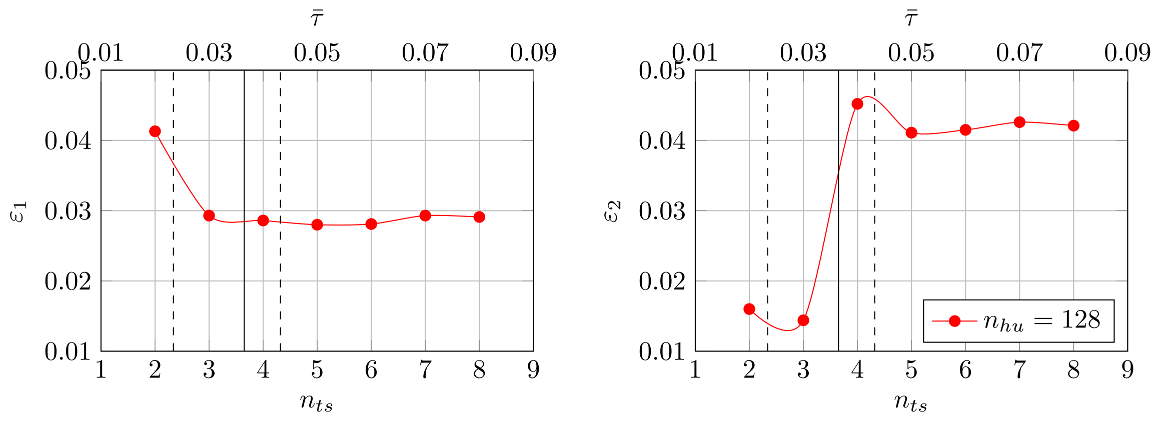


Figure 23: Test error of the trained networks for the Rayleigh-Bénard convection. The estimated mean (solid vertical line), minimum and maximum (dashed vertical lines) of the memory length are also shown in the plot.

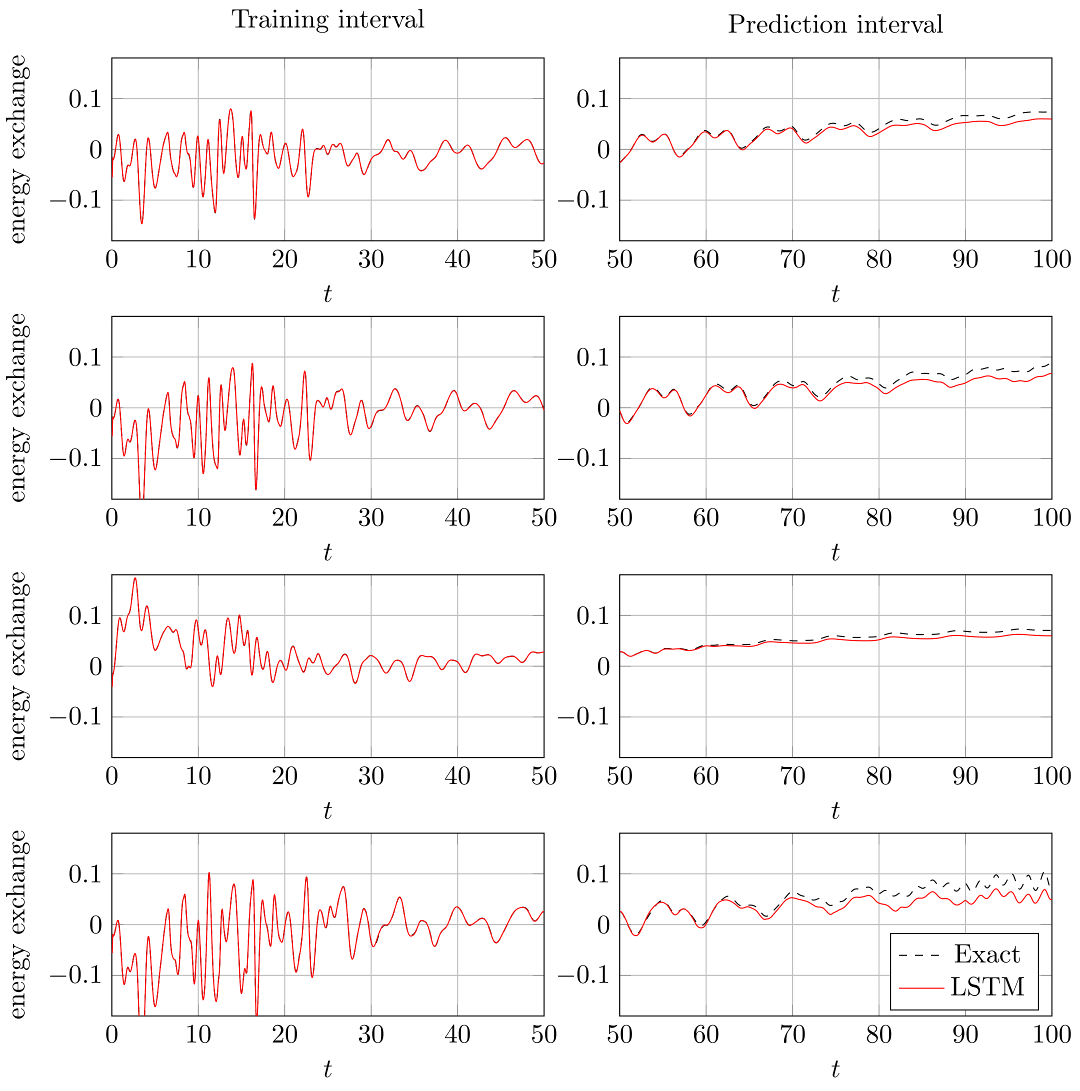


Figure 24: Evolution of the energy exchange term $2\alpha^\top \mathcal{M}$ of the reduced-order models of the Rayleigh-Bénard convection. From top to bottom: $(Ra, Pr) = (10315866.3651, 0.85392)$, $(12317729.8502, 0.86391)$, $(5793019.4994, 0.92232)$, $(14024512.0002, 0.86976)$.

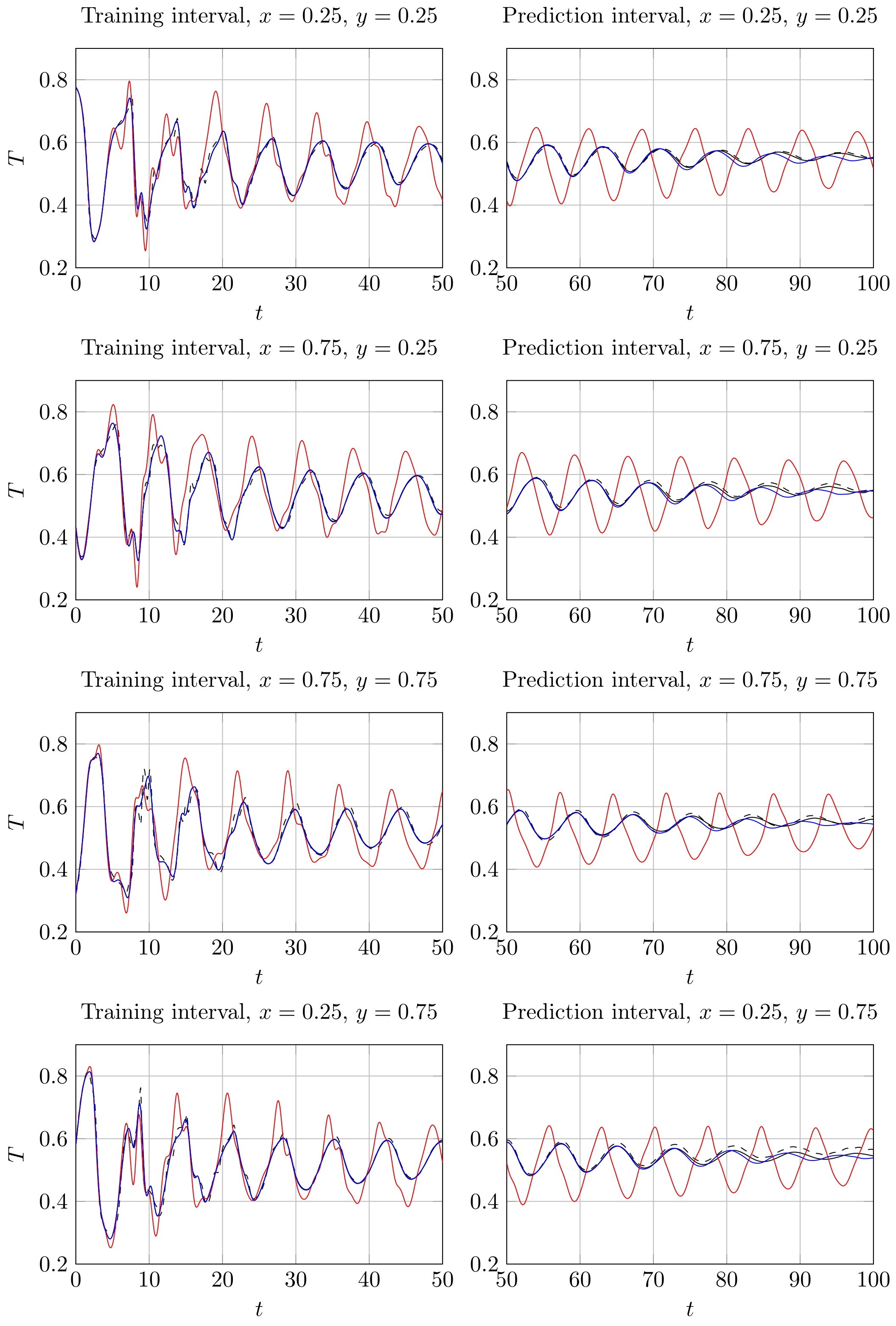


Figure 25: Temperature evolution of Rayleigh-Bénard convection problem at four points, for $Ra = 14024512.0002$, $Pr = 0.86976$. - - - - High-fidelity; —— Projection of high-fidelity; —— POD-Galerkin; —— POD-Galerkin with memory.

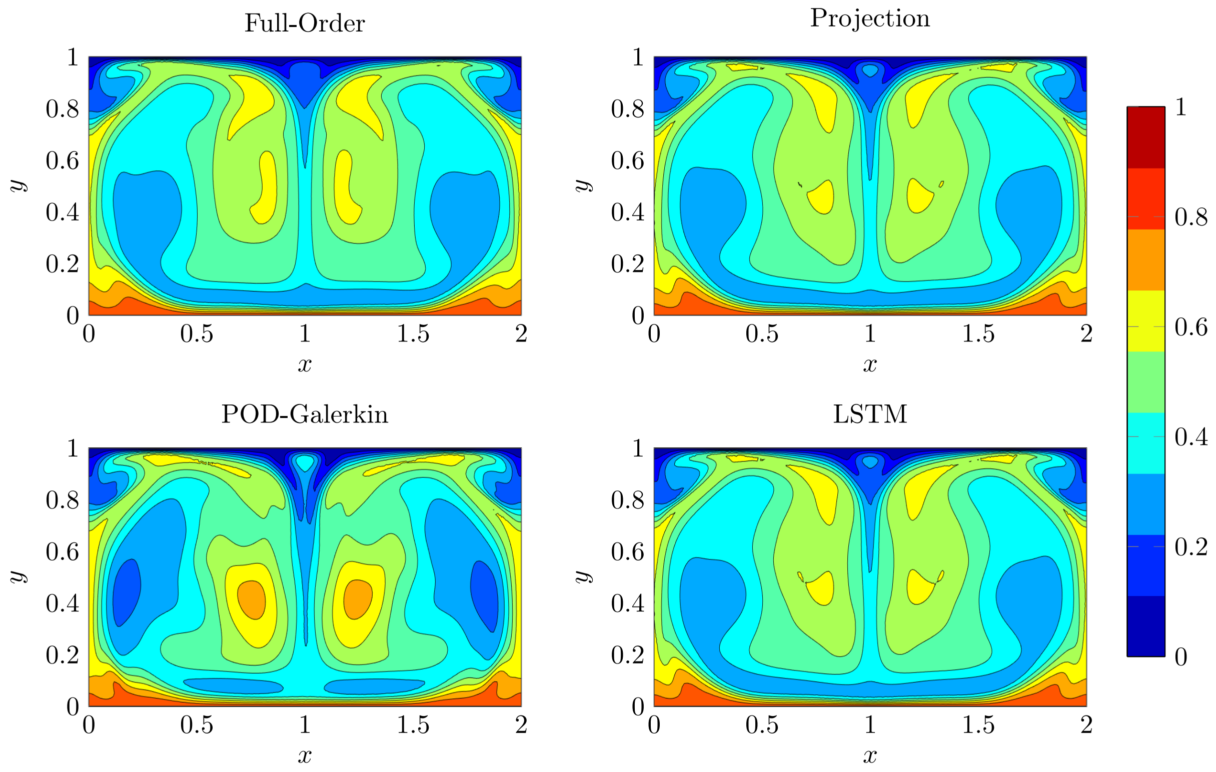


Figure 26: Contour of Rayleigh-Bénard convection problem at $t = 10$, for $Ra = 14024512.0002$, $Pr = 0.86976$.

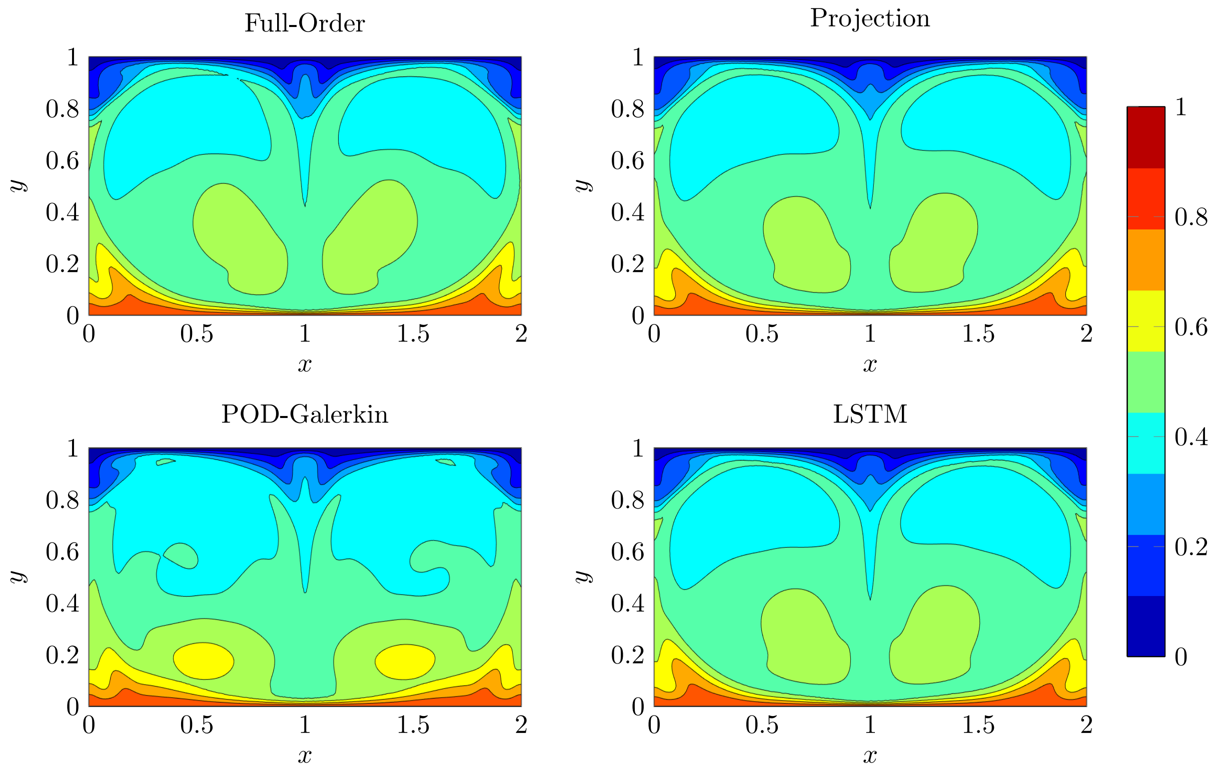


Figure 27: Contour of Rayleigh-Bénard convection problem at $t = 25$, for $Ra = 14024512.0002$, $Pr = 0.86976$.

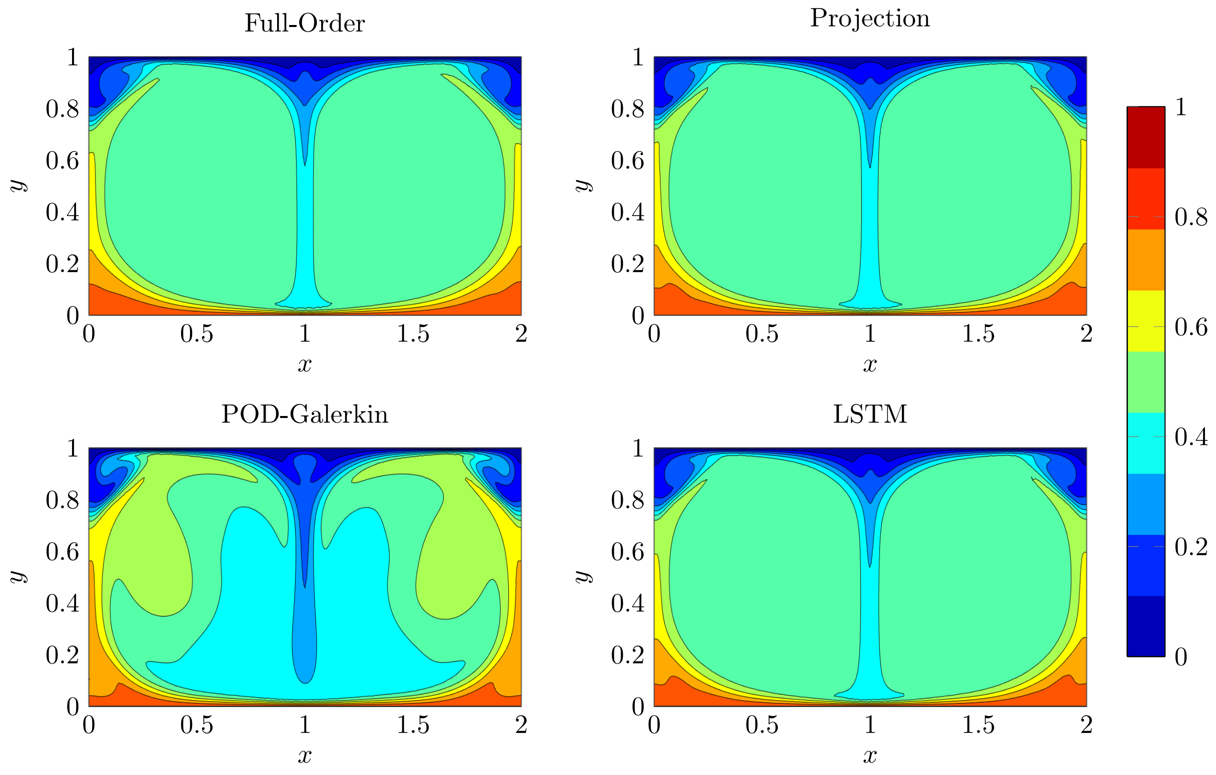


Figure 28: Contour of Rayleigh-Bénard convection problem at $t = 70$, for $Ra = 14024512.0002$, $Pr = 0.86976$.

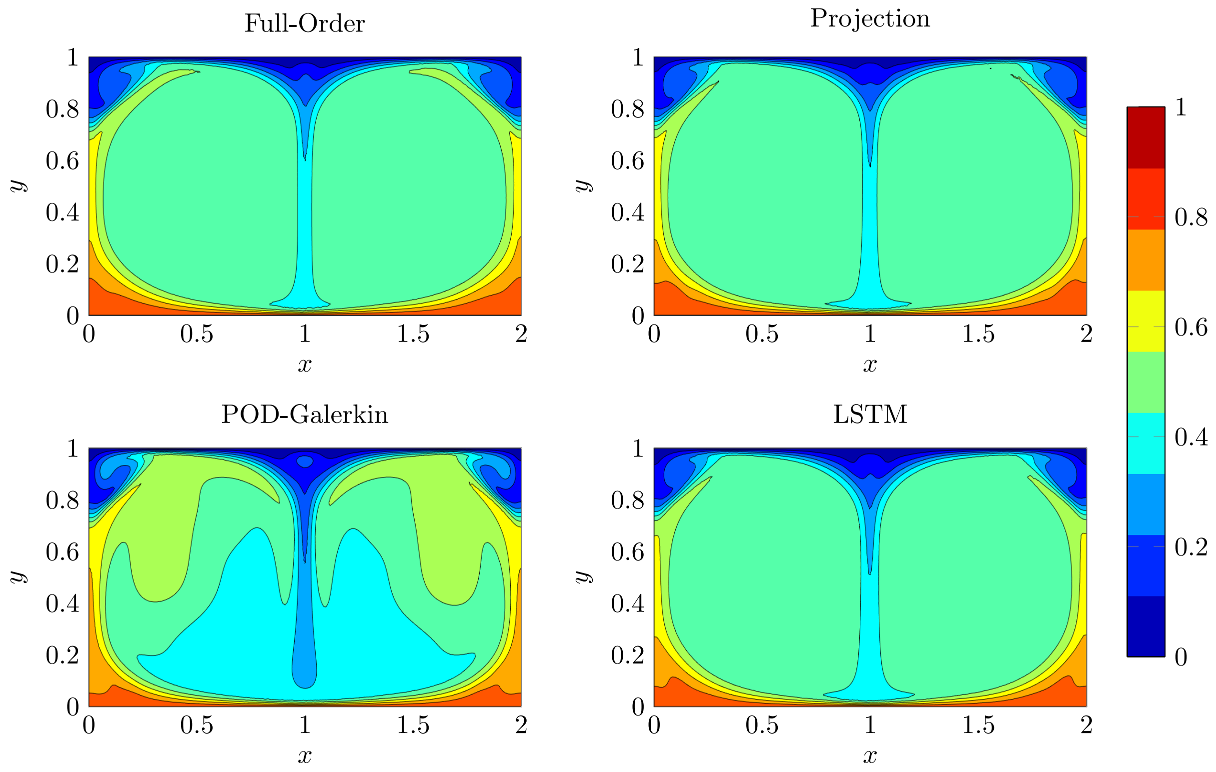


Figure 29: Contour of Rayleigh-Bénard convection problem at $t = 85$, for $Ra = 14024512.0002$, $Pr = 0.86976$.

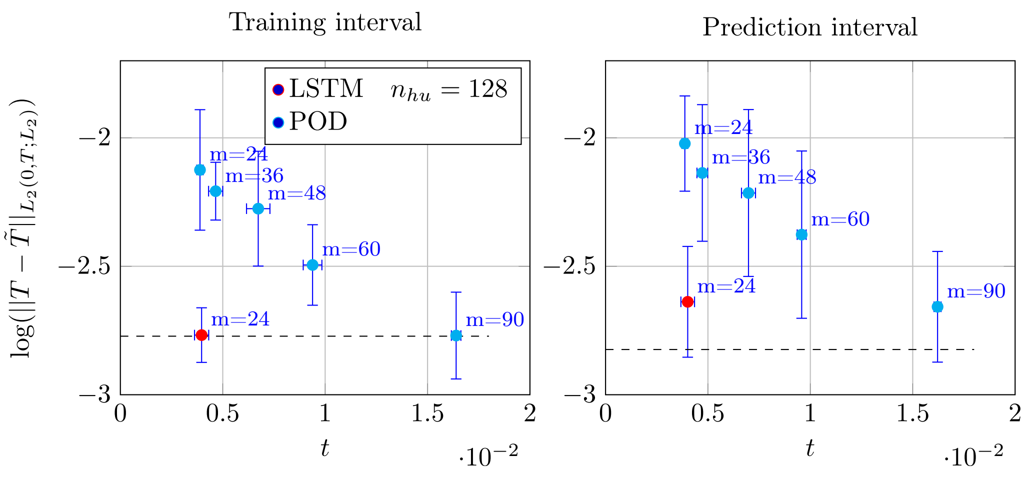


Figure 30: Error-cost plot of the reduced-order models of the Rayleigh-Bénard convection. The horizontal line represents the projection error.

model in the framework of implicit-explicit (IMEX) Runge-Kutta time integration, in which the RNN memory term is computed only once in each time step or inner iteration step, resulting in an efficient reduced-order model. Numerical results demonstrate that the POD-Galerkin reduced-order model with the RNN closure is much more efficient than its original scheme for nonlinear problems. The POD-Galerkin reduced-order model with the RNN closure is demonstrated to have strong prediction capability.

**Acknowledgements**

This work was partially supported by AFOSR under grant FA9550-17-1-9241.

**References**

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, and G. Research. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Technical report, 2015.

[2] R. Abgrall and R. Crisovan. Model reduction using L 1-norm minimization as an application to nonlinear hyperbolic problems. *International Journal for Numerical Methods in Fluids*, 87(12):628–651, 2018.

[3] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 2008.

[4] F. Ballarin, A. Manzoni, A. Quarteroni, and G. Rozza. Supremizer stabilization of POD–Galerkin approximation of parametrized steady incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Engineering*, 102(5):1136–1161, 2015.

[5] 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. *Comptes Rendus Mathematique*, 2004.

[6] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[7] P. Benner, M. Ohlberger, A. Patera, G. Rozza, and K. Urban. *Model Reduction of Parametrized Systems*. Springer, 2017.

[8] M. Billaud-Friess and A. Nouy. Dynamical model reduction method for solving parameter-dependent dynamical systems. *SIAM Journal on Scientific Computing*, 39(4):A1766–A1792, 2017.

[9] K. Carlberg. Adaptive h-refinement for reduced-order models. *International Journal for Numerical Methods in Engineering*, 102(5):1192–1210, 2015.

[10] 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*, 86(2):155–181, 2011.

[11] W. Cazemier, R. W. Verstappen, and A. E. Veldman. Proper orthogonal decomposition and low-dimensional models for driven cavity flows. *Physics of Fluids*, 1998.

[12] S. Chaturantabut and D. C. Sorensen. Nonlinear Model Reduction via Discrete Empirical Interpolation. *SIAM Journal on Scientific Computing*, 2010.

[13] F. Chinesta, P. Ladeveze, and E. Cueto. A Short Review on Model Order Reduction Based on Proper Generalized Decomposition. *Archives of Computational Methods in Engineering*, 18(4):395–404, 2011.

[14] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[15] F. Chollet. Keras, 2015.

[16] A. Chorin and P. Stinis. Problem reduction, renormalization, and memory. *Communications in Applied Mathematics and Computational Science*, 1(1):1–27, 2007.

[17] A. J. Chorin and O. H. Hald. *Stochastic tools in mathematics and science*, volume 3. Springer, 2009.

[18] 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*, 97(7):2968–2973, 2000.

[19] A. J. Chorin, O. H. Hald, and R. Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3-4):239–257, 2002.

[20] R. Everson and L. Sirovich. Karhunen–Loève procedure for gappy data. *Journal of the Optical Society of America A*, 1995.

[21] K. Gallivan, A. Vandendorpe, and P. Van Dooren. Model reduction via tangential interpolation. Technical report, 2002.

[22] D. Givon, R. Kupferman, and O. H. Hald. Existence proof for orthogonal dynamics and the Mori-Zwanzig formalism. *Israel Journal of Mathematics*, 145(1):221–241, 2005.

[23] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[24] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[25] 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*, 473(2205):20170385, 2017.

[26] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53:217–288, 2011.

[27] J. S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2015.

[28] J. S. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods*, volume 54. Springer, 2008.

[29] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[30] K. Hsu, H. V. Gupta, and S. Sorooshian. Artificial neural network modeling of the rainfall-runoff process. *Water resources research*, 31(10):2517–2530, 1995.

[31] C. Huang, K. Duraisamy, and C. Merkle. Challenges in Reduced Order Modeling of Reacting Flows. In *2018 Joint Propulsion Conference*, page 4675, 2018.

[32] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[33] 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*, 13(6):377–396, 2000.

[34] A. Karpathy and L. Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[35] D. P. Kingma and J. Ba Adam. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[36] B. Kramer and K. E. Willcox. Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition. *AIAA Journal*, 57(6):2297–2307, 2019.

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

[38] E. Lappano, F. Naets, W. Desmet, D. Mundo, and E. Nijman. A greedy sampling approach for the projection basis construction in parametric model order reduction for structural dynamics models. Technical report, KU Leuven, 2018.

[39] J. Li and P. Stinis. Mori-Zwanzig reduced models for uncertainty quantification. *Journal of Computational Dynamics*, 0(0):1–30, 2018.

[40] F. Lu, K. K. Lin, and A. J. Chorin. Data-based stochastic model reduction for the Kuramoto–Sivashinsky equation. *Physica D: Nonlinear Phenomena*, 340:46–57, 2017.

[41] D. J. Lucia, P. S. Beran, and W. A. Silva. Reduced-order modeling: new approaches for computational physics. *Progress in aerospace sciences*, 40(1-2):51–117, 2004.

[42] C. Ma, J. Wang, and W. E. Model Reduction with Memory and the Machine Learning of Dynamical Systems. *Commun. Comput. Phys*, 25(4):947–962, 2019.

[43] B. Maboudi Afkham, N. Ripamonti, Q. Wang, and J. S. Hesthaven. Conservative Model Order Reduction for Fluid Flow. Technical report, Ecole Polytechnique Fédérale de Lausanne, 2018.

[44] Y. Maday. Reduced basis method for the rapid and reliable solution of partial differential equations. In *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006*, pages 1255–1270. European Mathematical Society, 2009.

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

[46] H. Mori. Transport, collective motion, and Brownian motion. *Progress of theoretical physics*, 33(3):423–455, 1965.

[47] T. Ohwada and P. Asinari. Artificial compressibility method revisited: asymptotic numerical method for incompressible Navier–Stokes equations. *Journal of Computational Physics*, 229(5):1698–1723, 2010.

[48] S. Pan and K. Duraisamy. Data-Driven Discovery of Closure Models. *SIAM Journal on Applied Dynamical Systems*, 17(4):2381–2413, 2018.

[49] H. Panzer and J. Mohring. Parametric Model Order Reduction by Matrix Interpolation. *Automatisierungstechnik*, 58(8):475–484, 2010.

[50] E. J. Parish and K. Duraisamy. A dynamic subgrid scale model for large eddy simulations based on the Mori–Zwanzig formalism. *Journal of Computational Physics*, 349:154–175, 2017.

[51] E. J. Parish and K. Duraisamy. Non-Markovian closure models for large eddy simulations using the Mori-Zwanzig formalism. *Physical Review Fluids*, 2(1):14604, 2017.

[52] E. J. Parish, C. Wentland, and K. Duraisamy. A Residual-Based Petrov-Galerkin Reduced-Order Model with Memory Effects. *arXiv preprint arXiv:1810.03455*, 2018.

[53] B. Peherstorfer and K. Willcox. Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM Journal on Scientific Computing*, 37(4):A2123–A2150, 2015.

[54] A. Quarteroni. *Numerical Models for Differential Problems.* Springer Milan, Milano, 2009.

[55] A. Quarteroni and L. Formaggia. Mathematical modelling and numerical simulation of the cardiovascular system. *Handbook of numerical analysis*, 12:3–127, 2004.

[56] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced basis methods for partial differential equations: An introduction.* Springer, 2015.

[57] C. Rohilla Shalizi and C. Moore. What Is a Macrostate? Subjective Observations and Objective Dynamics. Technical report, 2000.

[58] C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for compressible flows using POD and Galerkin projection. *Physica D: Nonlinear Phenomena*, 189(1-2):115–129, 2004.

[59] G. Rozza and K. Veroy. On the stability of the reduced basis method for Stokes equations in parametrized domains. *Computer methods in applied mechanics and engineering*, 196(7):1244–1260, 2007.

[60] P. Stinis. Renormalized Mori–Zwanzig-reduced models for systems without scale separation. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2176):20140446, 2015.

[61] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[62] B. Unger and S. Gugercin. Kolmogorov n-widths for linear dynamical systems. *Advances in Computational Mathematics*, pages 1–14, may 2019.

[63] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, and T. Sapsis. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PloS one*, 13(5):e0197704, 2018.

[64] 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*, page 3531, 2019.

[65] Y. Zhu, J. M. Dominy, and D. Venturi. On the estimation of the Mori-Zwanzig memory integral. *arXiv:1708.02235v3 [math.NA]*, 2018.

[66] R. Zwanzig. Memory effects in irreversible thermodynamics. *Physical Review*, 124(4):983–992, 1961.

[67] R. Zwanzig. Nonlinear generalized Langevin equations. *Journal of Statistical Physics*, 9(3):215–220, 1973.