Physics-informed machine learning for reduced-order modeling of nonlinear problems

Wenqian Chen[a], Qian Wang[b,*], Jan S. Hesthaven[b], Chuhua Zhang[a]

[a]*Department of Fluid Machinery and Engineering, School of Energy and Power Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi, People's Republic of China*
[b]*Chair of Computational Mathematics and Simulation Science, École polytechnique fédérale de Lausanne, 1015 Lausanne, Switzerland*

Abstract

A physics-informed machine learning framework is developed for the reduced-order modeling of parametrized steady-state partial differential equations (PDEs). During the offline stage, a reduced basis is extracted from a collection of high-fidelity solutions and a reduced-order model is then constructed by a Galerkin projection of the full-order model onto the reduced space. A feedforward neural network is used to approximate the mapping from the physical/geometrical parameters to the reduced coefficients. The network can be trained by minimizing the mean squared residual error of the reduced-order equation on a set of points in parameter space. Such a network is referred to as physics-informed neural network (PINN). As the number of residual points is unlimited, a large data set can be generated to train a PINN to approximate the reduced-order model. However, the accuracy of such a network is often limited. This is improved by using the high-fidelity solutions that are generated to extract the reduced basis. The network is then trained by minimizing the sum of the mean squared residual error of the reduced-order equation and the mean squared error between the network output and the projection coefficients of the high-fidelity solutions. For complex nonlinear problems, the projection of high-fidelity solution onto the reduced space is more accurate than the solution of the reduced-order equation. Therefore, higher accuracy than the PINN for this network - referred to as physics-reinforced neural network (PRNN) - can be expected for complex nonlinear problems. Numerical results demonstrate that the PRNN is more accurate than the PINN and both are more accurate than a purely data-driven neural network for complex problems. During the reduced basis refinement, before reaching its accuracy limit, the PRNN obtains higher accuracy than the direct reduced-order model based on a Galerkin projection.

Keywords: Physics-informed machine learning, Feedforward neural network, Reduced-order modeling, Nonlinear PDE

1. Introduction

High-fidelity numerical simulation has been widely used in applications modeled by parametrized partial differential equations (PDEs) [1, 2], in which the parameters characterize geometric features, boundary conditions, source terms and physical properties, etc. However, for applications like design, control, optimization and uncertainty quantification, all of which require repeated model evaluations over a potentially large range of parameter values [3], high-fidelity simulation remains prohibitively expensive. The need of cost reduction in such applications has led to the development

*Corresponding author.
Email addresses: wenqianchen2016@gmail.com (Wenqian Chen), qian.wang@epfl.ch (Qian Wang), Jan.Hesthaven@epfl.ch (Jan S. Hesthaven), chzhang@mail.xjtu.edu.cn (Chuhua Zhang)

of reduced-order modeling (ROM) [4] that seeks to build low-dimensional models that are fast to evaluate while providing accurate predictions.

Reduced basis (RB) methods are a class of well-known and widely-used ROM techniques. RB methods are generally implemented in an offline-online paradigm [5]. During the offline stage, a reduced basis, which represents the main dynamics of the underlying problem, is extracted from a collection of high-fidelity solutions (snapshots). The reduced basis functions can be obtained by a variety of algorithms, such as proper orthogonal decomposition (POD) [1, 2, 6], the proper generalized decomposition [7], the piece-wise tangential interpolation [8], the matrix interpolation [9] and greedy algorithms[10–12]. Once the reduced basis functions are computed, a reduced-order model can be constructed in either an intrusive or a non-intrusive manner. During the online stage, for a new parameter location, the corresponding reduced-basis solution is recovered as a linear combination of the RB functions, with the expansion coefficients (also called the reduced coefficients) computed by solving the reduced-order model.

Intrusive reduced-order model is generally obtained by the projection of the full-order model onto the reduced space spanned by the RB functions. A Galerkin projection [13–15], in which the RB basis is used as the test functions, is the simplest and most popular choice. The popular method that uses POD to generate reduced basis and a Galerkin procedure to build reduced-order model is referred to as POD-G in this paper. During the online stage, the reduced coefficients are determined by solving the reduced-order equations. Projection-based methods have been successfully applied to a variety of problems. However, projection-based reduced-order methods lack *a priori* guarantees of stability, accuracy and convergence for complex nonlinear problems [13, 16, 17].

Non-intrusive reduced-order model generally predicts reduced coefficients for a new parameter value by interpolation or regression, based on a data base of reduced-order information [18]. Standard interpolation techniques may fail if only a small number of samples are available [19, 20]. Regression-based non-intrusive RB methods [21, 22] have recently been developed. Regression models, such as the artificial neural network (ANN) and Gaussian process (GP), are approximate maps between the parameter value and the projection coefficients of the high-fidelity solution onto the reduced space, and are trained by high-fidelity data in a supervised learning paradigm [23] during the offline stage. During the online stage, the reduced coefficients are determined by rapid evaluation of the regression map.

The advantages of the non-intrusive over the intrusive methods are *robustness* and *efficiency*. The robustness is due to the decoupling of the online stage and the high-fidelity scheme in the non-intrusive method. The efficiency is due to the fact that in the online stage, the reduced coefficients of the non-intrusive method are determined by evaluating the regression model, while reduced coefficients of the intrusive method are determined by solving a nonlinear equation system [21]. The major disadvantage of the non-intrusive over intrusive methods is the *offline cost*. Since a big data set, obtained by projecting a large amount of expensive high-fidelity snapshots onto the reduced space, is necessary to train an accurate regression model.

A reduced-order modeling framework based on physics-informed machine learning is proposed in this work to combine the advantages of intrusive and non-intrusive methods. In this framework, the reduced coefficients are predicted by a physics-informed neural network (PINN) [24–26]. The input of the PINN is the parameter value, the output is the set of the reduced coefficients. The PINN is optimized by minimizing the residual of the POD-G equation on sampled training points in parameter space. By taking the mean squared residual error of the reduced-order equation as the loss function, no labeled data is needed for the training. As the number of residual points is practically unlimited, a large data set can be generated to train an accurate PINN to approximate the reduced-order model at limited cost.

The accuracy of the network can be improved by using the high-fidelity solutions that are generated to extract the reduced basis. The loss function is defined as the sum of the mean squared residual error of the reduced-order equation and the mean squared error between the network output and the projection coefficients of the high-fidelity solutions onto the reduced space. The trained network is referred to as physics-reinforced neural network (PRNN). For complex nonlinear problems, the projection of the high-fidelity solution onto the reduced space is more accurate than the solution of the reduced-order equation, since the reduced-order equation does not take into account the impact of the unresolved scales (truncated modes) on the resolved scales (reduced basis modes) [14]. Therefore, the PRNN is expected to be more accurate than the PINN for complex nonlinear problems. By using the physics-informed machine learning technique, we can train neural networks to accurately and efficiently predict the reduced-order solution, without requiring extra high-fidelity simulation.

The one-dimensional Burgers' equation, the two-dimensional lid-driven flow and the two-dimensional natural convection problem are used to test the proposed reduced-order methods. Numerical results demonstrate that the PINN and PRNN can predict reliable reduced-order solutions and are much more accurate than a purely data-driven neural network. During the reduced basis refinement, before reaching its accuracy limit, the PRNN can obtain higher accuracy than the direct reduced-order model based on the Galerkin projection.

The remainder of the paper is organized as follows. Section 2 presents the model order reduction framework of parametrized PDEs. Section 3 discusses the physics-informed machine learning of the reduced-order model, while Section 4 presents the numerical results. Section 5 gathers the relevant conclusions.

2. Projection-based reduced basis method

Consider a nonlinear dynamical system governed by the parametrized partial differential equations (PDEs):

$$\begin{aligned} \mathcal{N}\left(\phi\left(\mathbf{x}\right);\boldsymbol{\mu}\right) &= 0, \qquad \mathbf{x} \in \Omega\left(\boldsymbol{\mu}\right), \\ \mathcal{B}\left(\phi\left(\mathbf{x}\right);\boldsymbol{\mu}\right) &= 0, \qquad \mathbf{x} \in \partial\Omega\left(\boldsymbol{\mu}\right), \end{aligned} \tag{1}$$

where $\mathcal{N}$ is a general nonlinear differential operator, $\phi\left(\mathbf{x}\right)$ are field variables to be determined on Cartesian coordinate $\mathbf{x}$. $\mathcal{B}$ is a boundary operator defined on the boundary $\partial\Omega$ of the physical domain Ω. $\boldsymbol{\mu} \in \mathcal{P}$ are the physical/geometrical parameters and $\mathcal{P} \subset \mathcal{R}^{n_p}$ is the parameter space. n_p is the number of parameters, i.e., the dimension of $\mathcal{P}$.

Snapshots used for generation of the reduced basis functions are obtained from high-fidelity (HF) simulations using the same amount of degree of freedoms (DOFs) on a set of parameter values. Physical coordinates of DOFs are variable when the shape or size of the physical domain Ω is controlled by geometrical parameters. Therefore, to ensure compatibility of the high-fidelity snapshots, we perform the simulations on a computational domain $\widetilde{\Omega}$. The invertible problem-dependent mapping from physical to computational coordinates is $\mathcal{X} : \mathbf{x} \in \Omega\left(\boldsymbol{\mu}\right) \rightarrow \boldsymbol{\xi} \in \widetilde{\Omega}$, as

$$\boldsymbol{\xi} = \mathcal{X}\left(\mathbf{x};\boldsymbol{\mu}\right), \tag{2}$$

and (1) is recast as

$$\begin{aligned} \mathcal{N}\left(\phi\left(\mathcal{X}^{-1}\left(\boldsymbol{\xi};\boldsymbol{\mu}\right)\right);\boldsymbol{\mu}\right) &:= \widetilde{\mathcal{N}}\left(\phi\left(\boldsymbol{\xi}\right);\boldsymbol{\mu}\right) = 0, \qquad \boldsymbol{\xi} \in \widetilde{\Omega}, \\ \mathcal{B}\left(\phi\left(\mathcal{X}^{-1}\left(\boldsymbol{\xi};\boldsymbol{\mu}\right)\right);\boldsymbol{\mu}\right) &:= \widetilde{\mathcal{B}}\left(\phi\left(\boldsymbol{\xi}\right);\boldsymbol{\mu}\right) = 0, \qquad \boldsymbol{\xi} \in \partial\widetilde{\Omega}, \end{aligned} \tag{3}$$

where $\widetilde{\mathcal{N}}$ and $\widetilde{\mathcal{B}}$ are the corresponding operators in the computational space.

2.1. Full-order model

A full-order model is obtained by the discretization of (3) using a high-fidelity scheme on a fine mesh. The number of interior DOFs is denoted as N, the number of boundary DOFs is denoted as

N_B. We denote the discrete solutions of (3) as $\boldsymbol{\phi}_h \in \mathbf{R}^N$ and $\boldsymbol{\phi}_h^B \in \mathbf{R}^{N_B}$. The full-order model is expressed as

$$\mathbf{L}_{\widetilde{\mathcal{N}}} \boldsymbol{\phi}_h + g_{\widetilde{\mathcal{N}}}(\boldsymbol{\phi}_h, \boldsymbol{\phi}_h^B) + \mathbf{C}_{\widetilde{\mathcal{N}}} = 0, \tag{4}$$

$$\mathbf{L}_{\widetilde{\mathcal{B}}} \boldsymbol{\phi}_h + \mathbf{L}_{\widetilde{\mathcal{B}}}^B \boldsymbol{\phi}_h^B + \mathbf{C}_{\widetilde{\mathcal{B}}} = 0, \tag{5}$$

where $\mathbf{L}_{\widetilde{\mathcal{N}}} \in \mathbf{R}^{N \times N}$, $\mathbf{L}_{\widetilde{\mathcal{B}}} \in \mathbf{R}^{N_B \times N}$ and $\mathbf{L}_{\widetilde{\mathcal{B}}}^B \in \mathbf{R}^{N_B \times N_B}$ represent matrices derived from the linear parts of the operators $\widetilde{\mathcal{N}}$ and $\widetilde{\mathcal{B}}$. $\mathbf{C}_{\widetilde{\mathcal{N}}} \in \mathbf{R}^N$ and $\mathbf{C}_{\widetilde{\mathcal{B}}} \in \mathbf{R}^{N_B}$ are constant vectors. $g_{\widetilde{\mathcal{N}}} : \mathbf{R}^N \times \mathbf{R}^{N_B} \mapsto \mathbf{R}^N$ is a nonlinear function derived from the nonlinear part of operator $\widetilde{\mathcal{N}}$. Discretization of the boundary conditions result in a linear equation system (5), as commonly used boundary conditions, e.g., Dirictlet, Neumann and Robin, are linear and thus $\mathcal{B}$ is a linear operator.

In this work, the high-order accurate Chebyshev pseudospectral method [27, 28], is used as the high-fidelity scheme.

2.1.1. Chebyshev pseudospectral method

In the Chebyshev pseudospectral method, a d-dimensional physical domain is mapped to the computational domain $\widetilde{\Omega} = [-1, 1]^d$, with each dimension discretized by $N_p + 1$ Chebyshev-Gauss-Lobatto points,

$$\xi_i = \cos\left(\frac{\pi i}{N_p}\right), \quad 0 \le i \le N_p. \tag{6}$$

The solution is approximated by $\phi(\xi) = \sum_{i=0}^{N_p} \phi_i \ell_i(\xi)$, where $\ell_i(\xi)$ is the Lagrange polynomial. Spatial derivatives are discretized as

$$\left.\frac{\partial^s \phi}{\partial \xi^s}\right|_{\xi_i} = \mathbf{D}^s \boldsymbol{\phi}, \tag{7}$$

where $\boldsymbol{\phi} = \{\phi_i\}_{i=0}^{N_p}$, s is the order of the derivative and $\mathbf{D}^s$ is the s-th-order difference matrix of size $(N_p + 1) \times (N_p + 1)$ with elements defined by

$$D_{i,j}^0 = \delta_{ij}, \qquad 0 \le i, j \le N_p, \tag{8}$$

$$\begin{cases} D_{i,j}^1 = \dfrac{B_i}{B_j} \dfrac{(-1)^{i+n}}{2\sin\left(\frac{(i+j)\pi}{2N_x}\right)\sin\left(\frac{(j-i)\pi}{2N_p}\right)} & 0 \le i, j \le N_p, i \ne j \\ D_{i,i}^1 = -\displaystyle\sum_{j=0, j\ne i}^{N_p} D_{i,j}^1 \quad 1 \le i \le N_p - 1 & \\ D_{0,0}^1 = -D_{N_p,N_p}^1 = -\dfrac{2{N_p}^2 + 1}{6} & \\ B_i = \begin{cases} 2 & i = 0, N_p \\ 1 & 1 \le i \le N_p - 1 \end{cases} & \end{cases}, \tag{9}$$

$$D_{i,j}^s = D_{i,k}^1 D_{k,j}^{s-1} \qquad 0 \le i, j, k \le N_p. \tag{10}$$

In the flow simulations in this paper, the $IP_N - IP_{N-2}$ method [29] is adopted to prevent spurious pressure mode [30]. In the $IP_N - IP_{N-2}$ method, the pressure derivative is approximated without pressure values on boundary points, thus pressure is approximated with a polynomial of two order lower than other flow variables. The first-order difference matrix for the pressure is $\hat{\mathbf{D}}^1$

defined as

$$\begin{cases} \hat{D}^1_{i,j} = 0 & i = 0, N_p \text{ or } j = 0, N_p \\ \hat{D}^1_{i,i} = \dfrac{3\xi_i}{2(1-\xi_i^2)} & 1 \le i \le N_p - 1 \\ \hat{D}^1_{i,j} = \dfrac{(-1)^{i+j}(1-\xi_j^2)}{2(1-\xi_i^2)(\xi_i - \xi_j)} & 1 \le i \ne j \le N_p - 1 \end{cases}. \tag{11}$$

For more details of the Chebyshev pseudospectral method, we refer the reader to the reference [27, 28, 30].

2.2. Reduced-order model

The solution of the full-order model (3) is often expensive due to the large amount of DOFs. Therefore, we are interested in replacing the full-order model with a low-dimensional reduced-order model to efficiently resolve the main dynamics of the underlying problem. The full-order model is said to be reducible if the high-fidelity solution $\boldsymbol{\phi}_h$ can be well approximated in an m-dimensional subspace $\mathcal{V}$. The subspace is spanned by a suitable set of basis vectors $\{\mathbf{v}_i \in \mathbf{R}^N\}_{i=1}^m$. The size of the problem is significantly reduced if $m \ll N$. The full-order solution of the interior DOFs $\boldsymbol{\phi}_h$ can be approximated by its projection onto the subspace $\mathcal{V}$ as

$$\boldsymbol{\phi}_h = \mathbf{V}\boldsymbol{\alpha} + \boldsymbol{\epsilon} \quad \approx \mathbf{V}\boldsymbol{\alpha}, \tag{12}$$

where $\mathbf{V} \in \mathbf{R}^{N\times m}$ is a matrix with the basis vectors as columns, $\boldsymbol{\alpha}$ are the projection coefficients and $\boldsymbol{\epsilon}$ is the projection error. As shown in (5), the boundary DOFs are linearly related to the interior DOFs. Therefore, the boundary vector $\boldsymbol{\phi}_h^B$ can be treated as a function of the interior DOFs vector $\boldsymbol{\phi}_h$ as

$$\boldsymbol{\phi}_h^B = -(\mathbf{L}^B_{\widetilde{\mathcal{B}}})^{-1}\left(\mathbf{L}_{\widetilde{\mathcal{B}}}\boldsymbol{\phi}_h + \mathbf{C}_{\widetilde{\mathcal{B}}}\right). \tag{13}$$

Substituting (12) and (13) into (4), we obtain the following over-determined system for the reduced coefficients $\boldsymbol{\alpha}$

$$\mathbf{L}_{\widetilde{\mathcal{N}}}\mathbf{V}\boldsymbol{\alpha} + \mathbf{g}_{\widetilde{\mathcal{N}}}\left(\mathbf{V}\boldsymbol{\alpha} + \boldsymbol{\epsilon}, -\left(\mathbf{L}^B_{\widetilde{\mathcal{B}}}\right)^{-1}\left(\mathbf{L}_{\widetilde{\mathcal{B}}}\mathbf{V}\boldsymbol{\alpha} + \mathbf{C}_{\widetilde{\mathcal{B}}} + \mathbf{L}_{\widetilde{\mathcal{B}}}\boldsymbol{\epsilon}\right)\right) + \mathbf{L}_{\widetilde{\mathcal{N}}}\boldsymbol{\epsilon} + \mathbf{C}_{\widetilde{\mathcal{N}}} = 0. \tag{14}$$

(14) can be expressed in a compact form

$$\mathbf{L}\mathbf{V}\boldsymbol{\alpha} + g\left(\mathbf{V}\boldsymbol{\alpha}\right) + \mathbf{C} = \tilde{\boldsymbol{\epsilon}}, \tag{15}$$

where $\mathbf{L} \in \mathbf{R}^{N\times N}$, $\mathbf{C} \in \mathbf{R}^N$ and $g : \mathbf{R}^N \mapsto \mathbf{R}^N$ are the matrix, vector and nonlinear function derived from (14), respectively. The residual error $\tilde{\boldsymbol{\epsilon}}$ results from the projection error $\boldsymbol{\epsilon}$.

By performing the procedure of (12) -(15), we obtain a reduced-order model for the interior DOFs with enforcement of the boundary conditions.

In the framework of Petrov-Galerkin, by projecting the over-determined system (15) onto a subspace $\mathbf{W} \in \mathbf{R}^{N\times m}$ and requiring that the residual is orthogonal to the test space, we recover a system of m equations

$$\mathbf{W}^T\left(\mathbf{L}\mathbf{V}\boldsymbol{\alpha} + \mathbf{g}\left(\mathbf{V}\boldsymbol{\alpha}\right) + \mathbf{C}\right) = 0. \tag{16}$$

In this work, we restrict ourselves to the Galerkin framework, namely $\mathbf{W}$=$\mathbf{V}$, reducing (16) to

$$\mathbf{V}^T\left(\mathbf{L}\mathbf{V}\boldsymbol{\alpha} + g\left(\mathbf{V}\boldsymbol{\alpha}\right) + \mathbf{C}\right) = 0. \tag{17}$$

Once the reduced coefficients $\boldsymbol{\alpha}$ are obtained by solving the reduced-order equation system (17), the interior solution $\boldsymbol{\phi}_h$ and boundary solution $\boldsymbol{\phi}_h^B$ can be recovered by (12) and (13), respectively.

A key point of the reduced-order model is to get a suitable set of reduced basis functions. There exists several methods for reduced basis generation, such as the proper orthogonal decomposition [1, 2, 6], the proper generalized decomposition [7], the piece-wise tangential interpolation [8], the matrix interpolation [9] and greedy algorithms[10–12]. The most widely-used reduced basis generation method, the proper orthogonal decomposition, is described and used in the rest of this paper.

2.2.1. Proper orthogonal decomposition

The proper orthogonal decomposition (POD) is one of the most widely-used techniques to compress data and extract an optimal set of orthonormal basis in the least-squares sense. Let $\mathcal{P}_{N_s} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_{N_s}\} \subset \mathcal{P}$ be a discrete and finite set of N_s parameter values generated with a suitable sampling, and $\mathbf{\Phi}_{\mathcal{P}_{N_s}} = \{\boldsymbol{\phi}_h(\boldsymbol{\mu}_1), \boldsymbol{\phi}_h(\boldsymbol{\mu}_2), ..., \boldsymbol{\phi}_h(\boldsymbol{\mu}_{N_s})\} \in \mathbf{R}^{N \times N_s}$ be the corresponding snapshot matrix. The POD basis of size m is the solution to the minimization problem

$$\begin{aligned} &\min_{\mathbf{V} \in \mathbf{R}^{N \times m}} \left\| \mathbf{\Phi}_{\mathcal{P}_{N_s}} - \mathbf{V}\mathbf{V}^T \mathbf{\Phi}_{\mathcal{P}_{N_s}} \right\|_F , \\ &s.t. \qquad \mathbf{V}^T \mathbf{V} = \mathbf{I}, \end{aligned} \tag{18}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\mathbf{I} \in \mathbf{R}^{m \times m}$ is the identity matrix. According to the Eckart-Young theorem [31], the solution of (18) is exactly the first m left singular vectors of the matrix $\mathbf{\Phi}_{\mathcal{P}_{N_s}}$, which can be obtained from the singular value decomposition (SVD)

$$\mathbf{\Phi}_{\mathcal{P}_{N_s}} = \mathbf{U}_S \mathbf{\Sigma}_S \mathbf{V}_S, \qquad \mathbf{\Sigma}_S = \operatorname{diag}(\sigma_i), \tag{19}$$

where the singular values σ_i, $1 \le i \le \min(N, N_s)$ are sorted in descending order. Choosing the first m columns of $\mathbf{U}_S$, we have an error estimation for (18) as

$$\min_{\mathbf{V} \in \mathbf{R}^{N \times m}} \left\| \mathbf{\Phi}_{\mathcal{P}_{N_s}} - \mathbf{V}\mathbf{V}^T \mathbf{\Phi}_{\mathcal{P}_{N_s}} \right\|_F^2 = \sum_{i=m+1}^{\min(N,N_s)} \sigma_i^2. \tag{20}$$

For (20), it is clear that the error in the POD basis is equal to the sum of the squares of the neglected singular values, i.e., by controlling the size m, we can approximate the snapshot matrix $\mathbf{\Phi}_{\mathcal{P}_{N_s}}$ with arbitrary accuracy. Many problems exhibit an exponentially decay of the singular values [2], allowing the use of a low-dimensional reduced space to approximate the high-fidelity solution with satisfactory accuracy.

2.2.2. Treatment of nonlinear term

The cost of the reduced-order model (17) depends on the computational complexities of the three terms. The cost of the two linear terms scales with m, since they can be simplified as

$$\begin{aligned} \mathbf{V}^T \mathbf{L} \mathbf{V} \boldsymbol{\alpha} &= \tilde{\mathbf{L}} \boldsymbol{\alpha}, \\ \mathbf{V}^T \mathbf{C} &= \tilde{\mathbf{C}}, \end{aligned} \tag{21}$$

where the matrix $\tilde{\mathbf{L}} \in \mathbf{R}^{m \times m}$ and the vector $\tilde{\mathbf{C}} \in \mathbf{R}^m$ need to be computed only once.

The computational bottleneck is the computation of the nonlinear term $\mathbf{V}^T g(\mathbf{V}\boldsymbol{\alpha})$ that scales with the full-order size N, even though the dimension of the equation system has been reduced to $m \ll N$. A further cost reduction of the nonlinear term is necessary to obtain an efficient reduced-order model. Several such hyper-reduction methods have been developed to enable significant speedups for the nonlinear term computation, e.g., the empirical interpolation method (EIM) [32], its discrete variant (DEIM) [33], gappy-POD [34] and missing point estimation (MPE) [35]. These methods seek a trade-off between accuracy and efficiency. In the present work, we restrict ourselves to the case of quadratic nonlinearity, for which hyper-reduction is not needed since the nonlinear term can be transformed exactly into a quadratic form [36]. Consider a quadratic nonlinearity

$$g(\mathbf{V}\boldsymbol{\alpha}) = (\mathbf{V}\boldsymbol{\alpha}) \otimes (\mathbf{V}\boldsymbol{\alpha}) \tag{22}$$

where $\otimes$ denotes the element-wise multiplication operator. Substituting (22) into (17), the nonlinear term becomes

$$\mathbf{V}^T g(\mathbf{V}\boldsymbol{\alpha}) = \mathbf{V}^T ((\mathbf{V}\boldsymbol{\alpha}) \otimes (\mathbf{V}\boldsymbol{\alpha})) \quad = \sum_{k=0}^{m} \left(\boldsymbol{\alpha}^T \mathbf{A}^k \boldsymbol{\alpha} \right) \mathbf{I}_k, \qquad 0 \le k \le m, \tag{23}$$

where $\mathbf{I}_k \in \mathbf{R}^m$ is k-th column of the identity matrix $\mathbf{I}$. $\mathbf{A}^k \in \mathbf{R}^{m\times m}$ is defined as

$$\mathbf{A}^k_{i,j} = \mathbf{v}_k^T \left(\mathbf{v}_i \otimes \mathbf{v}_j\right), \qquad 0 \le i, j, k \le m, \tag{24}$$

where $\mathbf{v}_l$ is the l-th column of $\mathbf{V}$.

Once the matrices $\mathbf{A}^k$ are computed and saved during the offline stage, the online evaluation of the nonlinear term has a computational complexity of $\mathcal{O}(m^3)$.

3. Physics-informed machine learning of reduced-order model

This section presents the physics-informed machine learning for reduced-order modeling. A POD-Galerkin reduced-order model can be built following the procedures in Section 2. However, the intrusive POD-G model suffers from stability, accuracy and convergence issues for complex nonlinear problems [13, 16, 17]. An alternative approach is to perform the non-intrusive reduced-order modeling based on regression [21]. The regression model approximates the mapping from the parameters to the projection coefficients of the underlying high-fidelity solution onto the reduced space. The neural network used as the regression model is referred to as projection driven neural network (PDNN) in this paper. The drawback of the non-intrusive reduced-order modeling is the high cost of the offline stage, since a large number of high-fidelity simulations are needed to generate a sufficiently large data set to train an accurate regression model.

A reduced-order modeling framework based on physics-informed machine learning is proposed in this section to combine the advantages of both intrusive and non-intrusive methods. In this framework, the reduced coefficients are predicted by a physics-informed neural network (PINN) [24–26]. The PINN seeks to minimize the mean squared residual error of the POD-G equation on sampled training points in parameter space. By taking the residual of the reduced-order equation as the loss function, no labeled data is needed for the training. However, the accuracy of PINN for complex nonlinear problem is often limited. To address this, the labeled data obtained by the projection of the high-fidelity snapshots that are used for basis generation, can be used to improve the accuracy. The usage of this data set does not require extra high-fidelity simulations. By using the labeled data, the loss function of the network becomes the sum of the mean squared residual error of the POD-G equation and the mean squared error between the network output and the label. The network trained by this strategy is referred to as physics-reinforced neural network (PRNN). The framework of the physics-informed machine learning for reduced-order modeling is sketched in Fig. 1.

In the remainder of this section, the physics-informed machine learning framework, including the network construction, data preparation and training procedure will be presented in detail.

3.1. Feedforward neural network

One of the most important issues for the design of an artificial neural network (ANN) is the architecture of the network. Several network architectures have been proposed, among which the feedforward neural network (FNN) is often preferred for function regression tasks.

A FNN maps the input $\mathbf{x}$ to the output $\mathbf{y}$. Generally, a FNN consists of an input layer, L hidden layers and an output layer. The l-th layer has n_l neurons, where $l = 0, 1, ..L, L+1$ denotes the input layer, first hidden layer,..., L-th hidden layer and the output layer, respectively. The forward propagation, i.e., the function $\mathbf{y} = f_{nn}(\mathbf{x})$ is

$$\begin{cases} \mathbf{y}^{(0)} = \mathbf{x} \\ \mathbf{y}^{(l)} = f_{act}\left(\mathbf{W}^{(l)}\mathbf{y}^{(l-1)} + \mathbf{b}^{(l)}\right), \quad l = 1, \cdots, L \\ \mathbf{y} = \mathbf{y}^{(L+1)} = \mathbf{W}^{(L+1)}\mathbf{y}^{(L)} + \mathbf{b}^{(L+1)} \end{cases}, \tag{25}$$

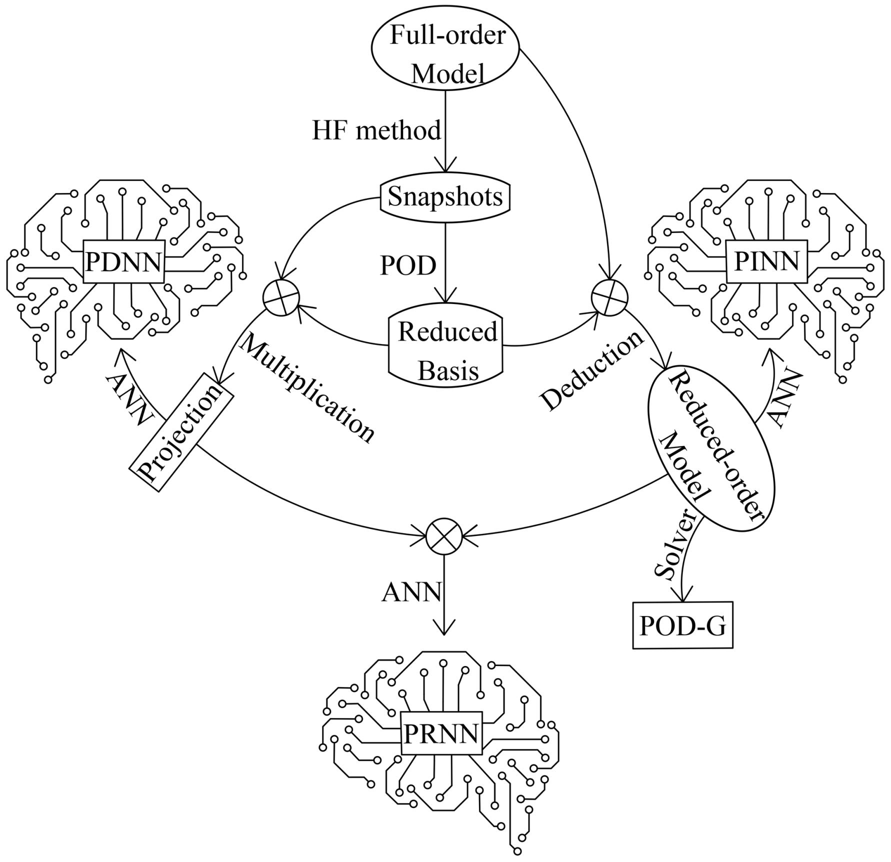


Figure 1: *Framework of physics-informed machine learning for reduced-order modeling.*

where $\mathbf{y}^{(l)} \in \mathbf{R}^{n_l}$ is the output of l-th layer, n_0 is the input dimension and n_{L+1} is the output dimension, $\mathbf{W}^{(l)} \in \mathbf{R}^{n_l \times n_{l-1}}$ are the weight matrices, $\mathbf{b}^{(l)} \in \mathbf{R}^{n_l}$ are the biases and f_{act} is the nonlinear element-wise activation function.

In this work, the network is used to predict the reduced coefficients $\boldsymbol{\alpha}$, given the parameter value $\boldsymbol{\mu}$. Therefore, the input dimension is $n_0 = n_p$ and output dimension is $n_{L+1} = m$. We put the same number of neurons in each hidden layer, namely $n_1 = n_2 = \cdots = n_L = n_H$. The nonlinear activation function is the Swish function $f_{act}(x) = x \cdot \text{sigmoid}(x)$ [37] .

The network is trained, i.e., the weights and biases are optimized by minimizing a loss function. In the next two subsections, we will define several machine learning strategies by using different loss functions.

The training of the FNN is implemented in PyTorch [38]. The optimal weights and biases of the network are obtained using the Adam stochastic optimizer [39], which uses mini-batches of the training data within a single 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 η.

3.2. Data preparation

In this subsection, we will present the procedure to generate the projection data set for the training process. The projection data is collected from the high-fidelity snapshots used to generate the reduced basis. As mentioned in Section 2, the snapshots $\Phi_{\mathcal{P}_{N_s}} = \{\boldsymbol{\phi}_h(\boldsymbol{\mu}_1), \boldsymbol{\phi}_h(\boldsymbol{\mu}_2), ..., \boldsymbol{\phi}_h(\boldsymbol{\mu}_{N_s})\} \in \mathbf{R}^{N\times N_s}$ are obtained on the sampled parameter set $\mathcal{P}_{N_s} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_{N_s}\} \subset \mathcal{P}$.

The projection coefficients of a snapshot $\boldsymbol{\phi}_h(\boldsymbol{\mu})$ onto the reduced space $\mathcal{V}$ are

$$\boldsymbol{\alpha}_{\mathcal{V}}(\boldsymbol{\mu}) = \mathbf{V}^T \boldsymbol{\phi}_h(\boldsymbol{\mu}). \tag{26}$$

The projection is the best approximation of the high-fidelity solution in the reduced space. Therefore, for a parameter value $\boldsymbol{\mu}$, the "best" output of the network is $\boldsymbol{\alpha}_{\mathcal{V}}(\boldsymbol{\mu})$. A data set $\mathcal{D}_{Pr} = \{(\boldsymbol{\mu}_i, \boldsymbol{\alpha}_{\mathcal{V}}(\boldsymbol{\mu}_i))\}_{i=1}^{N_s}$ can be collected from the high-fidelity solutions.

The training speed of neural networks is affected by the range of the input/output [40], thus feature scaling needs to be performed before feeding the training data into the network. For the input $\boldsymbol{\mu} \in \mathcal{P}$, as the boundary of the parameter space is predefined, the minimum and maximum of the parameters can be utilized to scale the input to $[-1, 1]^{n_p}$ as follows

$$\tilde{\boldsymbol{\mu}} = \frac{\boldsymbol{\mu} - (\boldsymbol{\mu}_{\max} + \boldsymbol{\mu}_{\min})/2}{(\boldsymbol{\mu}_{\max} - \boldsymbol{\mu}_{\min})/2}. \tag{27}$$

For the output $\boldsymbol{\alpha}$, there is not a predefined range. We turn to a statistical method to do the normalization with an assumption that the outputs satisfy a Gaussian distribution. To this end, the mean $\overline{\boldsymbol{\alpha}}$ and standard derivation $\boldsymbol{\sigma}_{\boldsymbol{\alpha}}$, computed from the data set $\mathcal{D}_{Pr}$, are used to normalize the output as

$$\tilde{\boldsymbol{\alpha}} = \frac{\boldsymbol{\alpha} - \overline{\boldsymbol{\alpha}}}{\boldsymbol{\sigma}_{\boldsymbol{\alpha}}}. \tag{28}$$

Feature scaling of the input/output can be viewed as a pre-processing, thus a corresponding post-processing is necessary to compute the final prediction

$$\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu}) = f_{nn}(\tilde{\boldsymbol{\mu}}) \otimes \boldsymbol{\sigma}_{\boldsymbol{\alpha}} + \overline{\boldsymbol{\alpha}}. \tag{29}$$

3.3. Physics-informed machine learning

Once the data preparation is finished, we can train a network to predict reduced coefficients using the data set $\mathcal{D}_{Pr}$, following the method proposed in [21]. As the network is trained on the data set collected by the projection of high-fidelity snapshots onto the reduced space, we refer to the network as projection driven neural network (PDNN). In the training procedure, $\mathcal{D}_{Pr}$ is shuffled and split into a training data set $\mathcal{D}_{Pr}^{tr}$ and validation data set $\mathcal{D}_{Pr}^{va}$. The ratio between the size of $\mathcal{D}_{\mathrm{Pr}}^{tr}$ and $\mathcal{D}_{Pr}$ is set as 0.7. The loss function is defined as the mean square error (MSE)

$$loss_{\mathrm{PDNN}} = \frac{1}{N_{\mathcal{D}}} \sum_{\boldsymbol{\mu}, \boldsymbol{\alpha} \in \mathcal{D}} \| f_{nn}(\tilde{\boldsymbol{\mu}}) - \tilde{\boldsymbol{\alpha}}_{\mathcal{V}}(\boldsymbol{\mu}) \|_2^2, \tag{30}$$

where $\mathcal{D}$ is the chosen data set of size $N_{\mathcal{D}}$, which can be the train set $\mathcal{D}_{Pr}^{tr}$ or the validation set $\mathcal{D}_{Pr}^{va}$.

In real applications, we do not expect to generate too many snapshots, since the high-fidelity simulation is expensive. Thus the data set $\mathcal{D}_{Pr}$ is typically small. To prevent over-fitting, L_2 weight regularization is employed as

$$\widetilde{loss}_{\mathrm{PDNN}} = loss_{\mathrm{PDNN}} + \lambda \|\mathbf{W}\|_{\mathrm{F}}^2, \tag{31}$$

where λ is the parameter used to control the amount of regularization. Furthermore, we adopt the early stopping technique: training will be immediately stopped if the validation loss keeps increasing over K_{early} epoches. In this work, we set $\lambda = 10^{-4}$ and $K_{\text{early}} = 6$.

3.3.1. Physics-informed neural network (PINN)

Due to the limited amount of snapshots, it is difficult to train an accurate PDNN. A physics-informed machine learning for reduced-order modeling is considered for more accurate network prediction of the reduced-order solution without performing extra high-fidelity simulations.

Recently, physics-informed neural network (PINN) has been developed and applied to PDEs [24–26]. The idea is to use the residual of the PDE as part of the loss function, to make the network solution fulfill the physics. As there is no labeled data involved in the residual loss, unlimited training data points can be sampled in the space-time domain, which significantly reduces the amount of labeled data points needed for the training. It is also possible to train a network to predict the solution of a PDE without any labeled data [41]. Inspired by these observations, we propose to train a PINN to predict $\boldsymbol{\alpha}$, which is the solution of the reduced-order equation (17), without any high-fidelity snapshot data.

In the training of the PINN for the reduced-order model, the output of the network is not compared with the projection of high-fidelity snapshots. Instead, the network output satisfies the reduced-order equation. We can sample a large number of training points in parameter space. Using Latin hypercube sampling in parameter space, we generate the data set $\mathcal{D}_{Resi} = \{\boldsymbol{\mu}_i\}_{i=1}^{N_{Resi}}$, containing N_{Resi} residual points in parameter space. The loss function is defined as

$$loss_{\mathrm{PINN}} = \frac{1}{N_{\mathcal{D}}} \sum_{\boldsymbol{\mu}\in\mathcal{D}} \left\| \mathbf{V}^T \left(\mathbf{L}\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu}) + g\left(\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu})\right) + \mathbf{C}\right)\right\|_2^2, \tag{32}$$

which is the mean squared residual error of (17). As we have an unlimited amount of training data, there is no problem of over-fitting and thus no need of validation data set. The training procedure of the PINN is summarized in Algorithm 1.

As the PINN is trained to approximate the reduced-order model, the accuracy limit of the PINN is the POD-G.

Algorithm 1 Training process of the PINN.

```
 1: function $[\mathbf{W}_{tr}, \mathbf{b}_{tr}]$=PINN_TRAINING($\mathcal{D}_{Resi}, N_{epo}, N_{batch}, \eta_0$)
 2:     $\mathbf{W}, \mathbf{b} \leftarrow$ INIT($\mathbf{W}, \mathbf{b}$)                        ▷ initialize weights and biases
 3:     $N_b \leftarrow N_{Resi}/N_{batch}$                                   ▷ Mini-batch size
 4:     for $epoch \leftarrow 1, N_{epo}$ do                                ▷ training epoch loop
 5:         $\eta \leftarrow$ LEARNING_RATE_DECAY($\eta_0, epoch$)               ▷ learning rate decay
 6:         $\mathcal{D}_{Resi} \leftarrow$ SHUFFLE($\mathcal{D}_{Resi}$)                ▷ shuffle the training data set
 7:         for $s \leftarrow 1, N_{batch}$ do                               ▷ mini-batch loop
 8:             $\mathcal{D}_{Resi,s} \leftarrow \mathcal{D}_{Resi}[(s-1)*N_b+1 : s*N_b]$        ▷ the $s$-th mini-batch
 9:             $loss \leftarrow \frac{1}{N_b}\sum_{\boldsymbol{\mu}\in\mathcal{D}_{Resi,s}} \left\|\mathbf{V}^T\left(\mathbf{L}\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu}) + g\left(\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu})\right)+\mathbf{C}\right)\right\|_2^2$   ▷ compute loss
10:             $\Delta\mathbf{W} \leftarrow -\eta \mathrm{G}_{\mathrm{ADAM}}(\frac{\partial loss}{\partial \mathbf{W}}), \Delta\mathbf{b} \leftarrow -\eta \mathrm{G}_{\mathrm{ADAM}}(\frac{\partial loss}{\partial \mathbf{b}})$   ▷ Adam optimizer step
11:             $\mathbf{W} \leftarrow \mathbf{W} + \Delta\mathbf{W}, \mathbf{b} \leftarrow \mathbf{b} + \Delta\mathbf{b}$         ▷ update weights and biases
12:         end for
13:     end for
14:     $\mathbf{W}_{tr} \leftarrow \mathbf{W}, \mathbf{b}_{tr} \leftarrow \mathbf{b}$                ▷ save optimized weights and biases
15: end function
```

3.3.2. Physics-reinforced neural network (PRNN)

For complex nonlinear problems, the projection of the high-fidelity solution onto the reduced space is more accurate than the solution of the reduced-order equation, since the reduced-order equation does not take into account of the impact of the unresolved scales (truncated modes) on the resolved scales (reduced basis modes). Therefore, the projection coefficients can be used to provide more physical information during the network training and thus improve the accuracy. For this reason, we refer to the trained network as physics-reinforced neural network (PRNN). The loss function is defined as the sum of the mean squared residual error of the reduced-order equation and the mean squared error between the network output and the projection coefficients of the high-fidelity solutions. The detailed form of the loss function is

$$loss_{\text{PRNN}} = loss_{\text{PDNN}}|_{\mathcal{D}=\mathcal{D}_{Pr}} + loss_{\text{PINN}}|_{\mathcal{D}=\mathcal{D}_{Resi}} \, . \tag{33}$$

The training procedure of the PRNN is summarized in Algorithm 2. We note that since the projection data set is small, we just use one mini-batch for the projection data set.

Algorithm 2 Training process of the PRNN.

```
1: function [$\mathbf{W}_{tr}, \mathbf{b}_{tr}$]=PRNN_TRAINING($\mathcal{D}_{Resi}, \mathcal{D}_{Pr}, N_{epo}, N_{batch}, \eta_0$)
2:     $\mathbf{W}, \mathbf{b} \leftarrow$ INIT($\mathbf{W}, \mathbf{b}$)                        ▷ initialize weights and biases
3:     $N_b \leftarrow N_{Resi}/N_{batch}$                                  ▷ Mini-batch size for $\mathcal{D}_{Resi}$
4:     for $epoch \leftarrow 1, N_{epo}$ do                                  ▷ training epoch loop
5:         $\eta \leftarrow$ LEARNING_RATE_DECAY($\eta_0, epoch$)                ▷ learning rate decay
6:         $\mathcal{D}_{Resi} \leftarrow$ SHUFFLE($\mathcal{D}_{Resi}$)                       ▷ shuffle the data set $\mathcal{D}_{Resi}$
7:         for $s \leftarrow 1, N_{batch}$ do                                  ▷ mini-batch loop
8:             $\mathcal{D}_{Resi,s} \leftarrow \mathcal{D}_{Resi}[(s-1)*N_b+1 : s*N_b]$        ▷ the $s$-th mini-batch
9:             $loss \leftarrow \frac{1}{N_b}\sum_{\boldsymbol{\mu}\in\mathcal{D}_{Resi,s}} \left\| \mathbf{V}^T\left(\mathbf{L}\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu}) + g\left(\mathbf{V}\boldsymbol{\alpha}_{nn}(\boldsymbol{\mu})\right) + \mathbf{C}\right)\right\|_2^2 +$
               $\frac{1}{N_s}\sum_{\boldsymbol{\mu},\boldsymbol{\alpha}\in\mathcal{D}_{Pr}} \left\| f_{nn}\left(\tilde{\boldsymbol{\mu}}\right) - \tilde{\boldsymbol{\alpha}}_{\mathcal{V}}(\boldsymbol{\mu})\right\|_2^2$     ▷ compute loss
10:            $\Delta\mathbf{W} \leftarrow -\eta \mathrm{G}_{\mathrm{ADAM}}(\frac{\partial loss}{\partial \mathbf{W}}), \Delta\mathbf{b} \leftarrow -\eta \mathrm{G}_{\mathrm{ADAM}}(\frac{\partial loss}{\partial \mathbf{b}})$    ▷ Adam optimizer step
11:            $\mathbf{W} \leftarrow \mathbf{W} + \Delta\mathbf{W}, \mathbf{b} \leftarrow \mathbf{b} + \Delta\mathbf{b}$              ▷ update weights and biases
12:        end for
13:    end for
14:    $\mathbf{W}_{tr} \leftarrow \mathbf{W}, \mathbf{b}_{tr} \leftarrow \mathbf{b}$                   ▷ save optimized weights and biases
15: end function
```

The PRNN is trained using the information of physics and projection data, and an ideal PRNN is supposed to achieve accuracy that is higher than the POD-G and lower than the projection. However, in the numerical results in Section 4, we observe that for the PDNN, PINN and PRNN, the accuracy reaches a certain limit during the reduced basis refinement. This is caused by the prediction error of the networks, and is impossible to avoid since the generalization accuracy of an artificial neural network is limited.

4. Numerical results

This section presents the numerical results of the reduced basis methods based on PDNN, PINN and PRNN for the one-dimensional Burgers' equation, the two-dimensional lid-driven cavity flow and the two-dimensional natural convection. The results of the following two methods are also presented for comparison:

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

(2) **POD-G**: Intrusive POD-Galerkin reduced-order model.

The following two metrics are used to measure the accuracy of the methods:

(1) mean relative projection error

$$\varepsilon_{\text{Proj}} = \frac{1}{N_{\mathcal{D}_{te}}} \sum_{\boldsymbol{\mu},\boldsymbol{\phi}_h \in \mathcal{D}_{te}} \frac{\left\| \boldsymbol{\phi}_h - \mathbf{V}\mathbf{V}^T \boldsymbol{\phi}_h \right\|_2}{\left\| \boldsymbol{\phi}_h \right\|_2}; \tag{34}$$

(2) mean relative error of the reduced-order solution

$$\varepsilon_{\bullet} = \frac{1}{N_{\mathcal{D}_{te}}} \sum_{\boldsymbol{\mu},\boldsymbol{\phi}_h \in \mathcal{D}_{te}} \frac{\left\| \boldsymbol{\phi}_h - \mathbf{V}\boldsymbol{\alpha}_{\bullet}(\boldsymbol{\mu}) \right\|_2}{\left\| \boldsymbol{\phi}_h \right\|_2}. \tag{35}$$

where $\mathcal{D}_{te} = \{\boldsymbol{\mu}_i, \boldsymbol{\phi}_h(\boldsymbol{\mu}_i)\}_{i=1}^{N_{\mathcal{D}_{te}}}$ is the test data set of size $N_{\mathcal{D}_{te}}$, the subscript "$\bullet$" is used to take the place of "POD-G","PDNN","PINN" and "PRNN", respectively.

The sampling methods in the parameter space in the numerical experiments are:

(1) high-fidelity snapshots: Sobol sequence;

(2) residual points: Latin hypercube sampling;

(3) test data points: Uniform sampling;

We refer the reader to the repository [1] for details of the code and data.

4.1. Burgers' equation

To validate the proposed methods, we first consider the one-dimensional parameterized Burgers' equation

$$\begin{cases} \phi(x;\boldsymbol{\mu})\dfrac{\partial \phi(x;\boldsymbol{\mu})}{\partial x} - \dfrac{\partial \phi^2(x;\boldsymbol{\mu})}{\partial x^2} = s(x;\boldsymbol{\mu}) \qquad -1 \le x \le 1 \\ \phi\left(x = \pm 1;\boldsymbol{\mu}\right) = 0 \end{cases}, \tag{36}$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2) \in [1, 10] \times [1, 10]$ are the parameters in the manufactured solution

$$\phi(x;\boldsymbol{\mu}) = (1 + \mu_1 x)\sin(-\mu_2 x/3)(x^2 - 1). \tag{37}$$

The source term $s(x;\boldsymbol{\mu})$ can be derived analytically by substituting the solution (37) into (36). High-fidelity snapshots are obtained using the Chebyshev pseudospectral method with 129 Chebyshev Gauss-Lobatto points.

A baseline reduced-order model is constructed to validate the proposed methods. The reduced basis functions are extracted using $N_s = 80$ high-fidelity snapshots. The singular value decay is shown in Fig. 2. The singular values decay quickly, implying that a small number of reduced basis functions is enough to recover the main dynamics of the solution. Meanwhile, the number of high-fidelity snapshots can be also reduced. The PDNN, PINN and PRNN have the same network architecture with $L = 3$ hidden layers and $n_H = 20$ neurons in each hidden layer. The number of residual points for the PINN and PRNN is $N_{Resi} = 5000$. The hyper-parameters for the training of the networks are listed in Table 1. The accuracy of the physics-informed machine learning based reduced-order model depends on the sizes of the projection data set, the residual data set and the feedforward network. The impacts of these three factors on the accuracy will be investigated separately, e.g., we train networks of different sizes on the same projection and residual data sets, to study the effect of the network size. A data set with $101 \times 101 = 10201$ uniformly sampled points in parameter space is used to test the accuracy of the models.

[1] available at `https://github.com/cwq2016/POD-PINN`

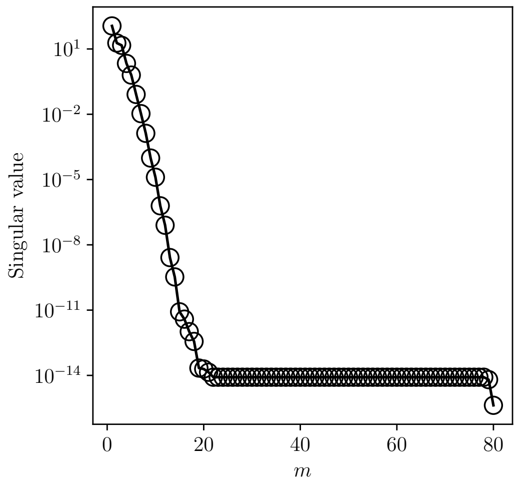


Figure 2: *Singular values of the Burgers' equation with 80 snapshots.*

Table 1: *Network training hyper-parameters for Burgers' equation.*

	Number of mini-batches					
Network	$\mathcal{D}_{Resi}$	$\mathcal{D}_{Pr}$	Epochs	Learning rate for epoch i	λ	K_{early}
PDNN		1	20,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$	10^{-4}	6
PINN	10		20,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$		
PRNN	10	1	20,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$		

Reduced-order models built with $N_s = 10$, 80 and 320 snapshots are tested to investigate the impact of the size of the projection data on the accuracy of the methods. The prediction error is shown in Fig. 3. The POD-G is accurate as its error is only slightly larger than the projection, which indicating that the impact of the unresolved scales on the resolved scales can be neglected. The PINN is as accurate as PRNN, and both follow the POD-G closely until they reach the accuracy limit of $\mathcal{O}(10^{-4})$ at $m = 6$. The results show that adding the projection data to the physics-informed learning is not beneficial in this case, since the POD-G model is already very accurate so that an accurate PINN can be obtained by learning the POD-G model on the residual data set. The PINN and PRNN are much more accurate than the PDNN. With $N_s = 10$ and $m \geq 6$, the error of the PINN/PRNN is up to three orders of magnitude smaller than that of the PDNN. As the dynamics of the problem can be recovered accurately on a small set of reduced basis, significant accuracy improvement due to the refinement of the high-fidelity snapshots is not observed in Fig. 3, except for the purely data-driven PDNN.

A important observation in Fig. 3 is that for the network-based reduced-order methods, the accuracy reaches a limit during the reduced basis refinement. This is caused by the reduced coefficients prediction error, and is impossible to avoid since the generalization accuracy of an artificial neural network is limited. As lower-order modes play more important roles than higher-order modes, prediction error of the reduced coefficients becomes dominant when the set of reduced basis is sufficiently large and the accuracy of the reduced-order solution saturates.

Reduced-order models built with $N_{Resi} = 1250$, 2500 and 10000 are tested to investigate the impact of the size of the residual data set on the accuracy of PINN and PRNN. In Fig. 4, it is

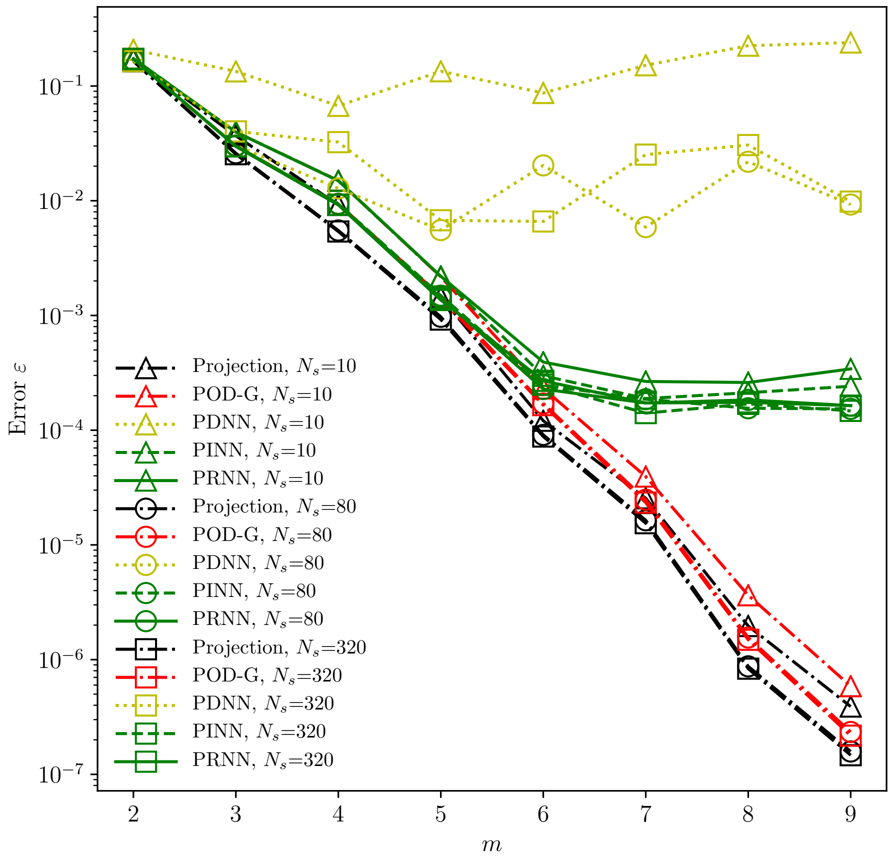


Figure 3: *Prediction error for the Burgers' equation with different number of snapshots.*

shown that the accuracy improves with the residual data set refinement, which indicates that the PINN and PRNN can learn the reduced-order equation more accurately with more residual points.

Reduced-order models based on networks with different hidden layer widths are tested to investigate the impact of the network size on the accuracy. The number of hidden layers is fixed as $L = 3$. The hidden layer widths are set as $n_H = 10$, 20 and 30. The errors of the PDNN, PINN and PRNN are shown in Fig. 5. The results show that the larger PINN and PRNN are more accurate since larger networks have stronger ability to learn the physics of the reduced-order model. It is also observed that larger PDNN results in larger error, which is reasonable since a larger data-driven network on a small data suffers from the over-fitting and can not achieve higher accuracy.

4.2. Lid-driven cavity flow

In this subsection, the lid-driven cavity flow problem is used to test the proposed reduced-order methods. The viscous flow in a parallelogram-shaped cavity is described by the following non-dimensionalized incompressible Navier-Stokes equations

$$\begin{cases} \nabla_{\mathbf{x}} \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla_{\mathbf{x}} \mathbf{u} = -\nabla_{\mathbf{x}} p + \dfrac{1}{Re} \nabla_{\mathbf{x}}^2 \mathbf{u} \end{cases}, \tag{38}$$

where $\mathbf{u} = (u, v)$ is the velocity in the Cartesian coordinate system $\mathbf{x} = (x, y)$, p is the pressure and Re is the Reynolds number. The geometry of the problem is depicted in Fig. 6 (a), and is

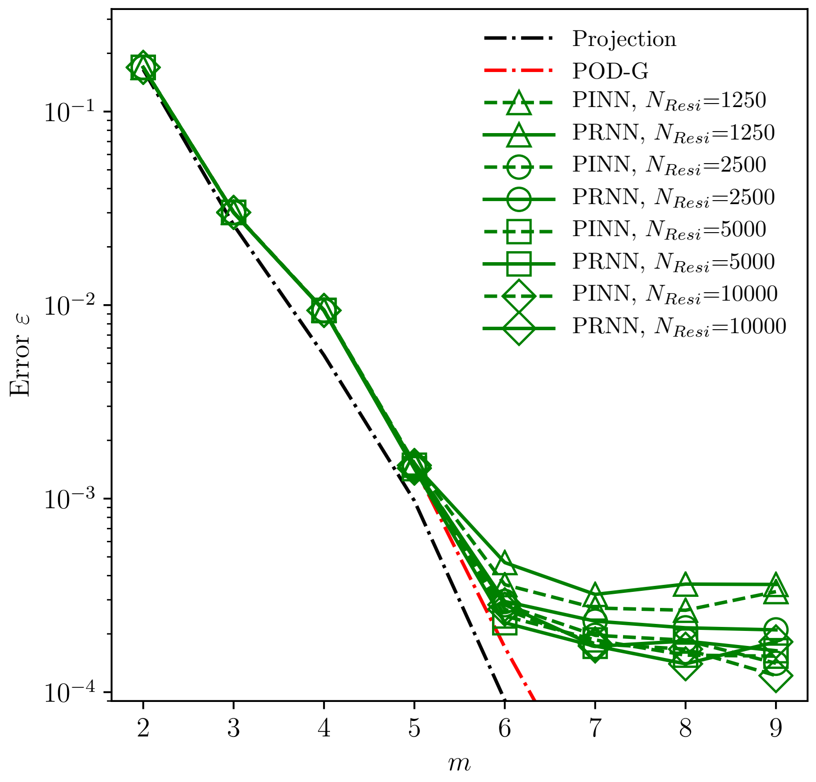


Figure 4: *Prediction error for the Burgers' equation with different number of residual points.*

affected by the angle θ between the oblique sides and the positive x-semiaxis. $\boldsymbol{\mu} = (Re, \theta)$ are the parameters for this problem.

The high-fidelity simulations are performed in the computational domain $[-1,1]^2$. The mapping from the physical coordinate $\mathbf{x} = (x, y)$ to the computational coordinate $\boldsymbol{\xi} = (\xi_1, \xi_2) \in [-1,1]^2$ is defined as

$$\mathcal{X} : \begin{cases} x = 1/2\xi_1 + 1/2\xi_2 \cos(\theta) \\ y = 1/2\xi_2 \sin(\theta) \end{cases} \quad \rightleftharpoons \quad \mathcal{X}^{-1} : \begin{cases} \xi_1 = 2(x - y\cot(\theta)) \\ \xi_2 = 2y/\sin(\theta) \end{cases}. \tag{39}$$

The Jacobian matrix of the mapping is

$$\mathbf{J} = \frac{\partial \mathbf{x}}{\partial \boldsymbol{\xi}} = \begin{bmatrix} 2 & 0 \\ -2\cot(\theta) & 2/\sin(\theta) \end{bmatrix}. \tag{40}$$

The governing equations in computational space are

$$\begin{cases} \left(\mathbf{J}^{-1}\nabla_{\boldsymbol{\xi}}\right) \cdot \mathbf{u} = 0 \\ \left(\mathbf{u} \cdot \left(\mathbf{J}^{-1}\nabla_{\boldsymbol{\xi}}\right)\right)\mathbf{u} = -\left(\mathbf{J}^{-1}\nabla_{\boldsymbol{\xi}}\right)p + \frac{1}{Re}\left(\mathbf{J}^{-1}\nabla_{\boldsymbol{\xi}}\right)^2\mathbf{u} \end{cases}. \tag{41}$$

The flow is driven by the top wall moving with a velocity

$$u_w = (1+\xi_1)^2(1-\xi_1)^2. \tag{42}$$

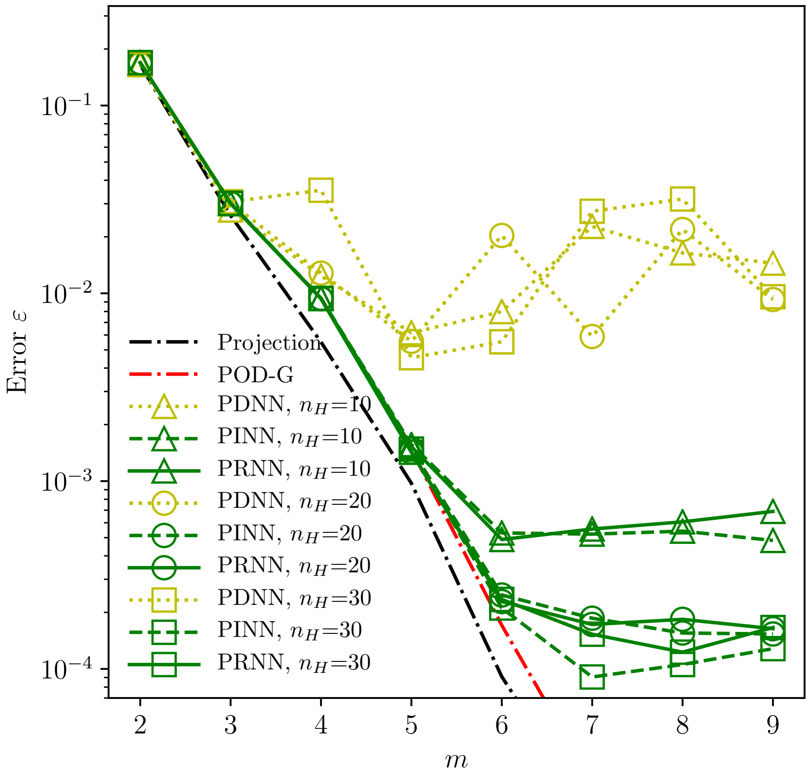
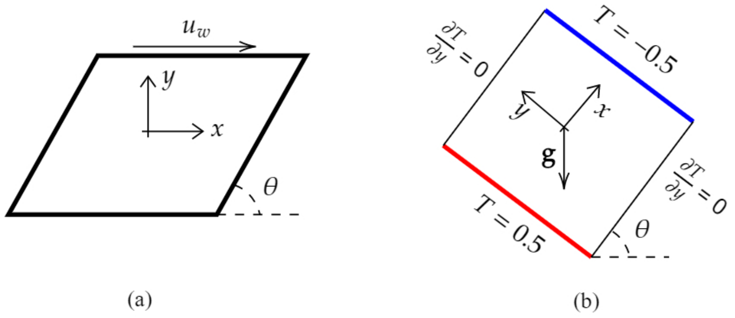


Figure 5: *Prediction error for the Burgers' equation with different network sizes.*



u_w
y
x
θ
(a)
$T=-0.5$
$\frac{\partial T}{\partial y}=0$
y
x
$\mathbf{g}$
$\frac{\partial T}{\partial y}=0$
$T=0.5$
θ
(b)

Figure 6: *The geometry and boundary conditions for (a) Lid-driven cavity flow and (b) Natural convection.*

The other three boundaries are static non-slip walls. The governing equations (41) are discretized by the Chebyshev pseudospectral method on a tensor-product mesh of 49×49 points. The $IP_N - IP_{N-2}$ method [29, 30] is adopted to remove the spurious modes of pressure. The artificial compressibility method [42] is used to address the velocity-pressure coupling issue by adding pseudo time derivatives to the steady-state equation (41). The discretized equations are advanced in pseudo time using the explicit four-stage fourth-order Runge-Kutta scheme.

The parameter space of interest is $\boldsymbol{\mu} = (Re, \theta) \in [100, 500] \times [\pi/3, 2\pi/3]$. The singular values for $N_s = 100$ snapshots are shown in Fig. 7. It shows that the singular values decay slowly, implying that a large number of basis functions are required to recover the main dynamics of the problem accurately.

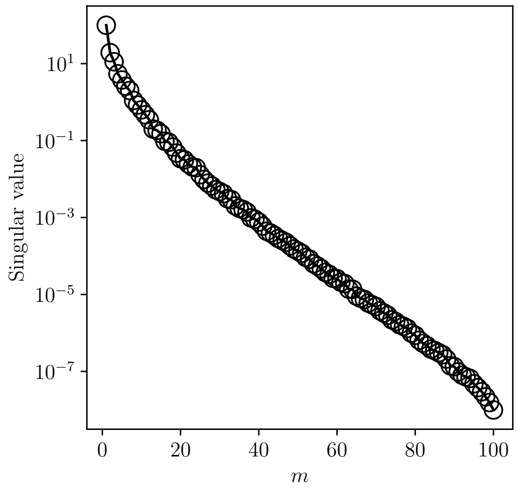


Figure 7: *Singular values of the lid -driven cavity flow problem with 100 snapshots.*

The POD-G solver diverges for many parameter values if a proper initial reduced-order solution can not be provided. To avoid the divergence issue, for a parameter $\boldsymbol{\mu}$, we use the projection of the corresponding high-fidelity solution onto the reduced space as the initial solution for the POD-G solver. This of course comes at high cost.

The PDNN, PINN and PRNN have the same network architecture with $L = 5$ hidden layers and $n_H = 30$ neurons in each hidden layer. The number of residual points for the PINN and PRNN is $N_{Resi} = 20,000$. The hyper-parameters for the training of the networks are listed in Table 2.

Table 2: *Network training hyper-parameters for the lid-driven cavity flow.*

	Number of mini-batches					
Network	$\mathcal{D}_{Resi}$	$\mathcal{D}_{Pr}$	Epochs	Learning rate for epoch i	λ	K_{early}
PDNN		1	40,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$	10^{-4}	6
PINN	20		40,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$		
PRNN	20	1	40,000	$0.01 \times 0.96^{\text{int}(\frac{i-1}{200})}$		

Reduced-order models are built with $N_s = 30$, 60 and 100 snapshots. A data set with $11 \times 11 = 121$ uniformly sampled points in parameter space is used to test the accuracy of the reduced-order models. The prediction errors of the reduced-order models are shown in Fig. 8. It shows that the POD-G error is larger than the projection error, due to the difficulty of recovering the main dynamics of the problem accurately using a small set of basis functions. As the projection of the high-fidelity solution onto the reduced space is more accurate than the POD-G solution, the use of the projection data in the training makes the PRNN more accurate than the PINN. It is shown in Fig. 8 that the error curve of the PRNN lies in between that of the POD-G and projection when $m \leq 20$, which demonstrates that the PRNN can reach higher accuracy than the POD-G in a low-dimensional reduced space. Both PINN and PRNN are more accurate than the pure data-driven PDNN, which suffers from the low-accuracy and over-fitting issues on small data sets. The error of the PRNN is around one order of magnitude smaller than that of the PINN for $m \geq 20$.

Predictions by the PRNN for several parameter values are compared with the corresponding high-fidelity solutions in Fig. 9 to intuitively show the accuracy of the proposed reduced-order method. The PRNN reduced-order solutions agree well with the high-fidelity solutions.

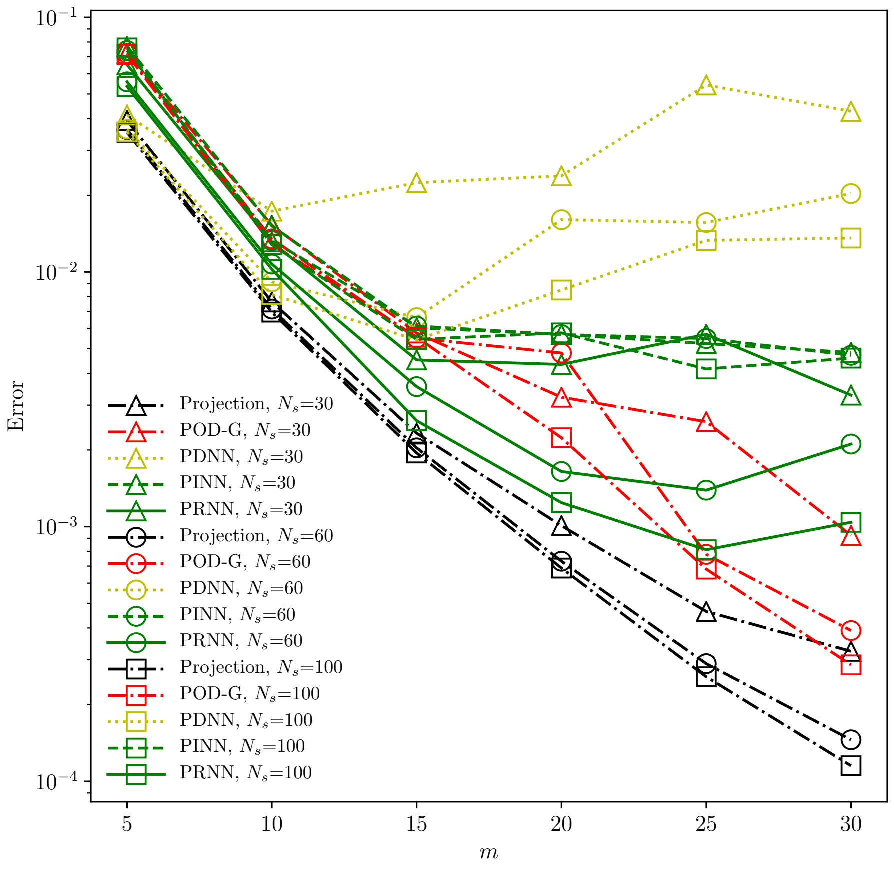


Figure 8: *Prediction error for the lid-driven cavity flow problem with different number of snapshots.*

4.3. Natural convection

The natural convection problem is used to further test the proposed reduced-order methods. The unit square cavity is filled with an incompressible Newtonian fluid. The cavity is heated and cooled on the left and right walls, while the other two boundaries are adiabatic walls. The geometry and boundary conditions are shown in Fig. 6(b). The flow is driven by the vertical buoyancy force, which is modeled by the Boussinesq approximation. The orientation of the cavity is controlled by an angle θ. The physical coordinate system $x-y$ is attached to the cavity, by setting the origin as the center of the cavity and the y-axis parallel to the heated/cooled walls. The natural convection can be modeled by the non-dimensionalized incompressible Navier-Stokes equations plus an energy equation

$$\begin{cases} \nabla_{\mathbf{x}} \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla_{\mathbf{x}} \mathbf{u} = -\nabla_{\mathbf{x}} p + \sqrt{\left(\dfrac{Pr}{Ra}\right)} \nabla_{\mathbf{x}}^2 \mathbf{u} - T\mathbf{n}_g \\ \mathbf{u} \cdot \nabla_{\mathbf{x}} T = \dfrac{1}{\sqrt{(Pr \times Ra)}} \nabla_{\mathbf{x}}^2 T \end{cases}, \tag{43}$$

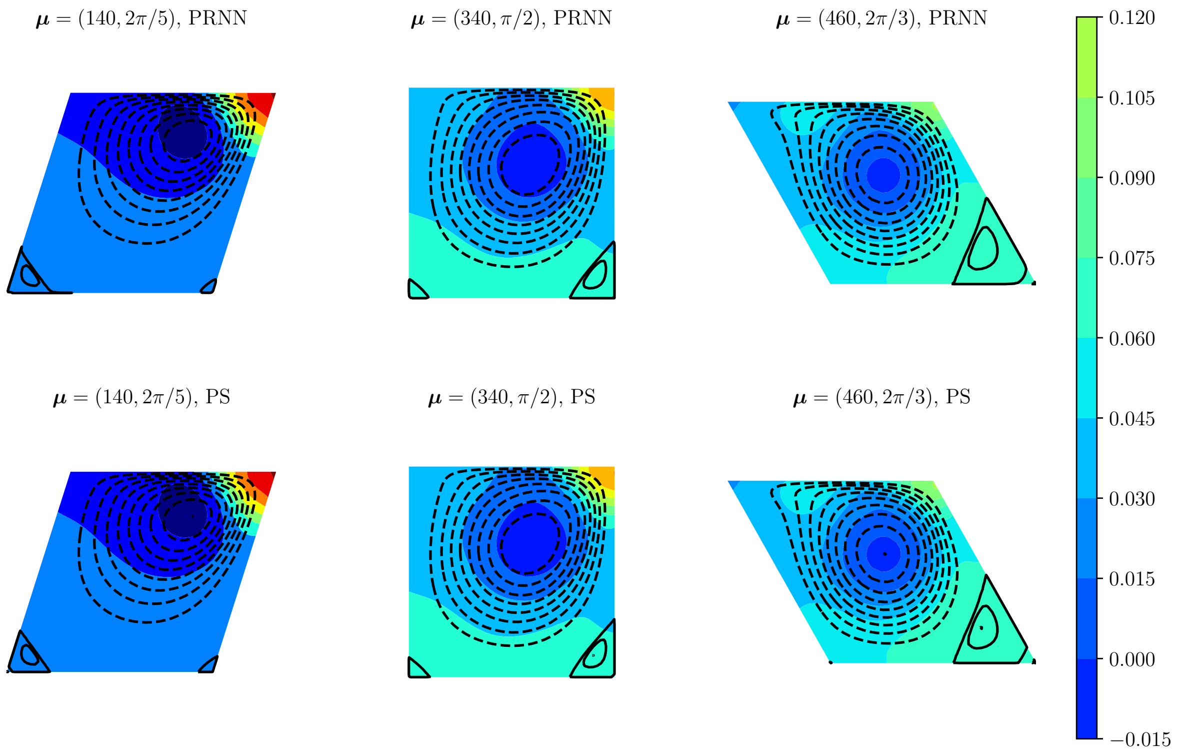


Figure 9: *Pressure contours and streamlines of the lid-driven cavity flow computed by the high-fidelity solver and the PRNN for three parameter values. The reduced basis is of size $m = 25$ and extracted from 100 snapshots. The solid streamline indicates an anticlockwise vortex and the dashed streamline indicates an clockwise vortex.*

where $\mathbf{u} = (u, v)$ is the velocity in the Cartesian coordinate system $\mathbf{x} = (x, y)$, p is the pressure, Ra is the Rayleigh number, Pr is the Prandtl number and $\mathbf{n}_g = (-\sin\theta, -\cos\theta)$ is the direction of gravity. The temperatures on the heated and cooled walls are $T = 0.5$ and $T = -0.5$, respectively. Similar to the lid-driven cavity problem in Section 4.2, the high-fidelity simulations are obtained using the Chebyshev pseudospectral method on a mesh with 49×49 points. As the physical domain is a unit square, the physical coordinate can be linearly scaled to the computational coordinate. The governing equations in computational space can be easily obtained from (43) and is thus omitted.

The parameter space of interest is $\boldsymbol{\mu} = (Ra, Pr, \theta) \in [10^4, 10^5] \times [0.6, 0.8] \times [\pi/4, \pi/2]$. The singular values for $N_s = 200$ snapshots are shown in Fig. 10. It shows that the singular values decay slowly, suggesting that a large number of basis functions are needed to accrately recover the main dynamics of the problem.

Similar to the lid-driven cavity problem, to avoid the divergence issue, we use the projection of the corresponding high-fidelity solution onto the reduced space as the initial solution for the POD-G solver.

The PDNN, PINN and PRNN have the same network architecture with $L = 5$ hidden layers and $n_H = 30$ neurons in each hidden layer. The number of residual points for the PINN and PRNN is $N_{Resi} = 20,000$. The hyper-parameters for the training of the networks are listed in Table 2.

Reduced-order models are built with $N_s = 30$, 100 and 200 snapshots. A data set with $6 \times 6 \times 6 = 216$ uniformly sampled points in parameter space is used to test the accuracy of the reduced-order models. The prediction errors of the reduced-order models are shown in Fig. 11. It is

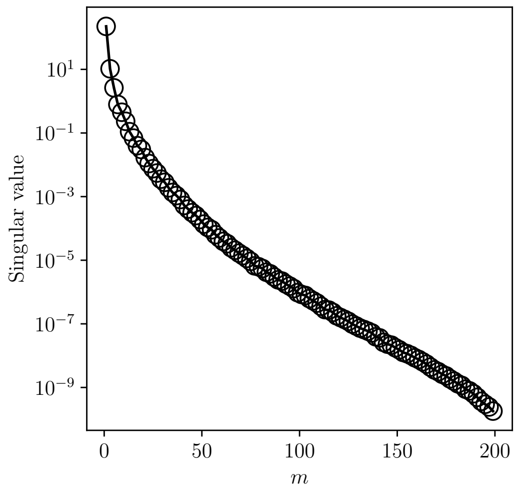


Figure 10: *Singular value decay of the natural convection problem with 200 snapshots.*

shown that the POD-G error is larger than the projection error, due to the difficulty of recovering the main dynamics of the problem accurately using a small set of basis functions. The PRNN is more accurate than the PINN because of the use of the projection data in the training. It is shown in Fig. 11 that the error curve of the PRNN lies between that of the POD-G and projection when $m \leq 20$, which demonstrates that the PRNN reaches higher accuracy than the POD-G in low-dimensional reduced space. Both PINN and PRNN are more accurate than the purely data-driven PDNN.

Predictions by the PRNN for several parameter values are compared with the corresponding high-fidelity solutions in Fig. 12 to show the accuracy of the proposed reduced-order method. The PRNN reduced-order solutions agree well with the high-fidelity solutions.

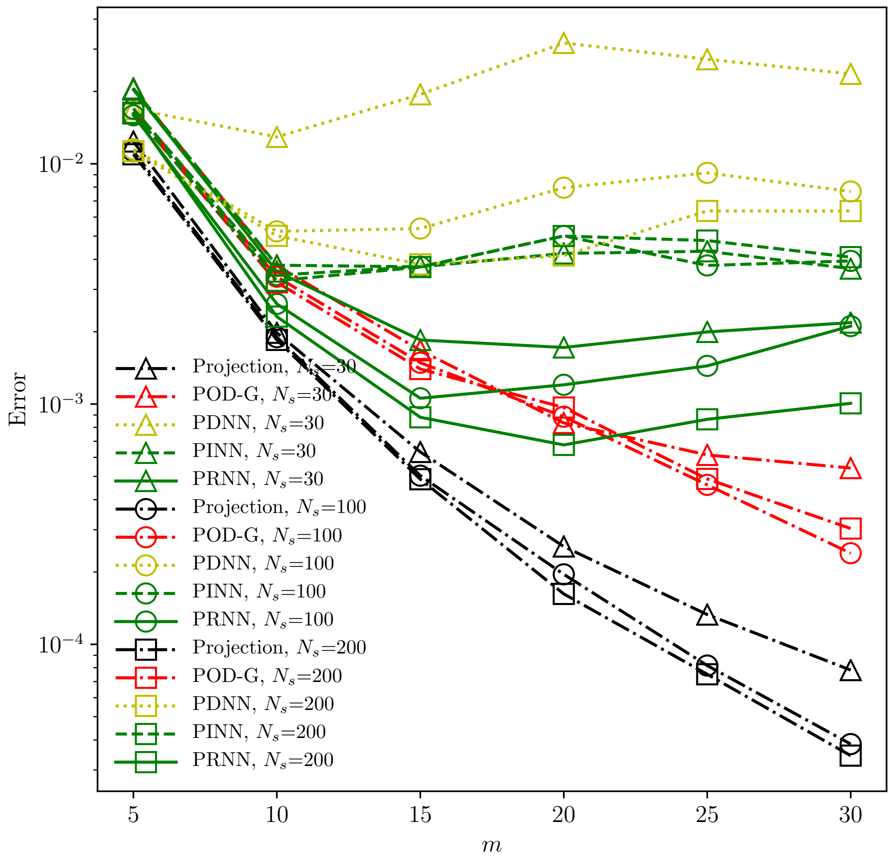


Figure 11: *Prediction error for the natural convection problem with different number of snapshots.*

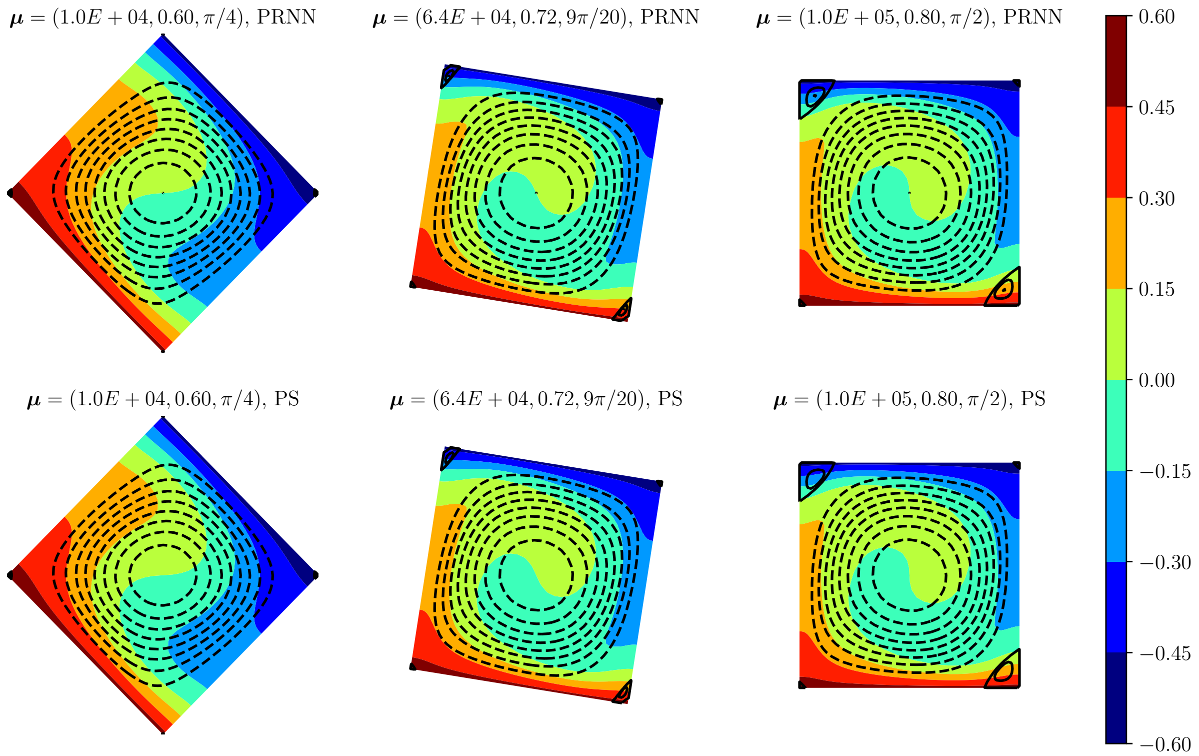


Figure 12: *Temperature contours and streamlines of the natural convection computed by the high-fidelity solver and the PRNN for three parameter values. The reduced basis is of size* $m = 20$ *and extracted from 200 snapshots. The solid streamline indicates an anticlockwise vortex and the dashed streamline indicates an clockwise vortex.*

5. Conclusions

This paper proposes a physics-informed machine learning for reduced-order modeling. The reduced coefficients are predicted by a feedforward neural network. A physics-informed neural network (PINN) can be used to learn the POD-G reduced-order equation system, by setting the mean squared residual error of the reduced-order equation as the loss function in the training procedure. The PINN is an approximation of the reduced-order model that is obtained without any labeled data. For a more accurate network, the projection data, collected from the high-fidelity snapshots that are used in the reduced basis generation, are introduced into the training to provide more physical information. The physics-reinforced neural network (PRNN) is trained by minimizing the residual error of the reduced-order model and the matching error on the projection data set. The physics-informed machine learning technique makes it possible to train an accurate network to predict reduced-order solution, without the requirement of extra high-fidelity simulations.

Numerical results demonstrate that the PRNN achieves higher accuracy than the POD-G when the reduced basis set is small. The online stage of the physics-informed learning based reduced-order method is efficient as it just requires the evaluation of the network. Therefore, the PRNN is an accurate and efficient reduced-order modeling tool for general nonlinear problems.

Acknowledgments

The first author is financially supported by Xi'an Jiaotong University Graduate Short-term Academic Visiting Program, National Key Research and Development Project of China [Grant number 2016YFB0200901], National Science and Technology Major Project of China [Grant number 2017-II-0006-0020].

References

[1] J. S. Hesthaven, G. Rozza, B. Stamm, et al., Certified reduced basis methods for parametrized partial differential equations, Vol. 590, Springer, 2016.

[2] A. Quarteroni, A. Manzoni, F. Negri, Reduced basis methods for partial differential equations: an introduction, Vol. 92, Springer, 2015.

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

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

[5] 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, 2007, pp. 1255–1270.

[6] Y. Liang, H. Lee, S. Lim, W. Lin, K. Lee, C. Wu, Proper orthogonal decomposition and its applications-part i: Theory, Journal of Sound and vibration 252 (3) (2002) 527–544.

[7] F. Chinesta, P. Ladeveze, E. Cueto, A short review on model order reduction based on proper generalized decomposition, Archives of Computational Methods in Engineering 18 (4) (2011) 395.

[8] K. Gallivan, A. Vandendorpe, P. Van Dooren, Model reduction via tangential interpolation, in: MTNS 2002 (15th Symp. on the Mathematical Theory of Networks and Systems), 2002, p. 6.

[9] H. Panzer, J. Mohring, R. Eid, B. Lohmann, Parametric model order reduction by matrix interpolation, at-Automatisierungstechnik 58 (8) (2010) 475–484.

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

[11] E. Lappano, F. Naets, W. Desmet, D. Mundo, E. Nijman, A greedy sampling approach for the projection basis construction in parametric model order reduction for structural dynamics models, in: Proceedings of ISMA, 2016, pp. 19–21.

[12] J. S. Hesthaven, B. Stamm, S. Zhang, Efficient greedy algorithms for high-dimensional parameter spaces with applications to empirical interpolation and reduced basis methods, ESAIM: Mathematical Modelling and Numerical Analysis 48 (1) (2014) 259–283.

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

[14] Q. Wang, N. Ripamonti, J. S. Hesthaven, Recurrent neural network closure of parametric pod-galerkin reduced-order models based on the mori-zwanzig formalism, Journal of Computational Physics (2020) 109402.

[15] A. Deane, I. Kevrekidis, G. E. Karniadakis, S. Orszag, Low-dimensional models for complex geometry flows: application to grooved channels and circular cylinders, Physics of Fluids A: Fluid Dynamics 3 (10) (1991) 2337–2354.

[16] C. Huang, K. Duraisamy, C. Merkle, Challenges in reduced order modeling of reacting flows, in: 2018 Joint Propulsion Conference, 2018, p. 4675.

[17] A. Iollo, S. Lanteri, J.-A. Désidéri, Stability properties of pod–galerkin approximations for the compressible navier–stokes equations, Theoretical and Computational Fluid Dynamics 13 (6) (2000) 377–396.

[18] F. Casenave, A. Ern, T. Lelièvre, A nonintrusive reduced basis method applied to aeroacoustic simulations, Advances in Computational Mathematics 41 (5) (2015) 961–986.

[19] D. Amsallem, Interpolation on manifolds of cfd-based fluid and finite element-based structural reduced-order models for on-line aeroelastic predictions, Ph.D. thesis, Stanford University (2010).

[20] V. Barthelmann, E. Novak, K. Ritter, High dimensional polynomial interpolation on sparse grids, Advances in Computational Mathematics 12 (4) (2000) 273–288.

[21] J. S. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, Journal of Computational Physics 363 (2018) 55–78.

[22] M. Guo, J. S. Hesthaven, Reduced order modeling for nonlinear structural analysis using gaussian process regression, Computer methods in applied mechanics and engineering 341 (2018) 807–826.

[23] M. Riedmiller, Advanced supervised learning in multi-layer perceptrons—from backpropagation to adaptive learning algorithms, Computer Standards & Interfaces 16 (3) (1994) 265–278.

[24] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378 (2019) 686–707.

[25] M. Raissi, A. Yazdani, G. E. Karniadakis, Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science 367 (6481) (2020) 1026–1030.

[26] Z. Mao, A. D. Jagtap, G. E. Karniadakis, Physics-informed neural networks for high-speed flows, Computer Methods in Applied Mechanics and Engineering 360 (2020) 112789.

[27] W. Chen, Y. Ju, C. Zhang, A multidomain multigrid pseudospectral method for incompressible flows, Numerical Heat Transfer, Part B: Fundamentals 74 (1) (2018) 415–431.

[28] W. Chen, Y. Ju, C. Zhang, A parallel inverted dual time stepping method for unsteady incompressible fluid flow and heat transfer problems, Computer Physics Communications (2020) 107325.

[29] W. Zhang, C. Zhang, G. Xi, An explicit chebyshev pseudospectral multigrid method for incompressible navier–stokes equations, Computers & Fluids 39 (1) (2010) 178–188.

[30] R. Peyret, Spectral methods for incompressible viscous flow, Vol. 148, Springer Science & Business Media, 2013.

[31] C. Eckart, G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1 (3) (1936) 211–218.

[32] M. Barrault, Y. Maday, N. C. Nguyen, A. T. Patera, An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations, Comptes Rendus Mathematique 339 (9) (2004) 667–672.

[33] S. Chaturantabut, D. C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM Journal on Scientific Computing 32 (5) (2010) 2737–2764.

[34] R. Everson, L. Sirovich, Karhunen–loeve procedure for gappy data, JOSA A 12 (8) (1995) 1657–1664.

[35] P. Astrid, S. Weiland, K. Willcox, T. Backx, Missing point estimation in models described by proper orthogonal decomposition, IEEE Transactions on Automatic Control 53 (10) (2008) 2237–2251.

[36] W. Cazemier, R. Verstappen, A. Veldman, Proper orthogonal decomposition and low-dimensional models for driven cavity flows, Physics of fluids 10 (7) (1998) 1685–1699.

[37] P. Ramachandran, B. Zoph, Q. V. Le, Searching for activation functions, arXiv preprint arXiv:1710.05941.

[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[39] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

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

[41] L. Sun, H. Gao, S. Pan, J.-X. Wang, Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data, Computer Methods in Applied Mechanics and Engineering 361 (2020) 112732.

[42] J. R. Clausen, Entropically damped form of artificial compressibility for explicit simulation of incompressible flow, Physical Review E 87 (1) (2013) 013309.