Physics-data combined machine learning for parametric reduced-order modelling of nonlinear dynamical systems in small-data regimes

Jinlong Fu[a], Dunhui Xiao[b,*], Rui Fu[a], Chenfeng Li[a,c], Chuanhua Zhu[a], Rossella Arcucci[d], Ionel M. Navon[e]

[a]*Zienkiewicz Centre for Computational Engineering, Faculty of Science and Engineering, Swansea University, Swansea SA1 8EN, UK*
[b]*School of Mathematical Sciences, Tongji University, Shanghai, 200092, P.R. China*
[c]*Energy Safety Research Institute, Faculty of Science and Engineering, Swansea University, Swansea SA1 8EN, UK*
[d]*Department of Earth Science and Engineering, Imperial College London, London SW7 2BP, UK*
[e]*Department of Scientific Computing, Florida State University, Tallahassee, FL, 32306-4120, USA*

Abstract

Repeatedly solving nonlinear partial differential equations with varying parameters is often an essential requirement to characterise the parametric dependences of dynamical systems. Reduced-order modelling (ROM) provides an economical way to construct low-dimensional parametric surrogates for rapid predictions of high-dimensional physical fields. This paper presents a physics-data combined machine learning (PDCML) method for non-intrusive parametric ROM in small-data regimes. Proper orthogonal decomposition (POD) is adopted for dimension reduction by deriving basis functions from a limited number of high-fidelity snapshots, and parametric ROM is thus transformed into establishing reliable mappings between the system parameters and the POD coefficients. To overcome labelled data scarcity, a physics-data combined ROM framework is developed to jointly integrate the physical principle and the small labelled data into feedforward neural networks (FNN) via a step-by-step training scheme. Specifically, a preliminary FNN model is firstly fitted via data-driven training, and then the governing physical rules are embedded into the loss function to improve the model interpolation and extrapolation performances through physics-guided training constrained by the labelled data. During the constrained optimization procedure, dynamic weighting factors are used to adjust the physics-data proportion of the loss functions, aiming at continuously highlighting the physics loss as the primary optimization objective and keeping the data loss as the constraint. This new PDCML method is tested on a series of nonlinear problems with different numbers of physical variables, and it is also compared with the data-driven ROM, the physics-guided ROM and the traditional projection-based ROM methods. The results demonstrate that the proposed method provides a cost-effective way for non-intrusive parametric ROM via machine learning, and it possesses good characteristics of high prediction accuracy, strong generalization capability and small data requirement.

Keywords: Physics-data combination; Model order reduction; Feedforward neural network; Nonlinear dynamics; Non-intrusive; Small data

1. Introduction

In modern engineering and scientific applications, it is a remarkably common situation to repeatedly solve parametrized partial differential equations (PDEs) over a broad range of parameter values through high-fidelity numerical simulations [1, 2, 3, 4]. The parameters might describe material properties, geometric characteristics, initial conditions, boundary settings and system configurations in various disciplines, such as uncertainty quantification, sensitive analysis, data assimilation, structure design, weather forecasting and pore-scale modelling [5, 6]. However, complete reliance on high-fidelity numerical simulation to explore relevant problems usually brings a prohibitive computation burden, and even super-computing platforms can be overstretched to deal with large-scale and complicated problems. Besides, fast and nearly real-time simulations are also urgent demands in many industrial applications [7], such as online dynamic system control, structural health monitoring, online advanced manufacturing feedback and automated driving control. Therefore, it is of great practical significance to construct parametric surrogate models for efficient and accurate prediction of the high-dimensional quantities of interest, such as velocity, pressure, temperature, strain, stress and displacement fields.

[*] Corresponding author
Email address: xiaodunhui@tongji.edu.cn (Dunhui Xiao)

Parametric reduced-order modelling (ROM) [1, 8, 9] derives low-dimensional representations from full-order numerical systems by preserving the main dynamic features, aiming at achieving rapid and reliable solutions. Among various ROM techniques, projection-based methods [2, 9, 10, 11] are the most commonly used category, where reduced basis functions are extracted from the high-fidelity solutions (snapshots) via a wide range of algorithms, including proper orthogonal decomposition [4, 8, 12, 13], dynamic model decomposition [14], proper generalized decomposition [15], discrete empirical interpolation [16] and autoencoder [17, 18, 19]. Once the reduced basis functions are obtained, reduced-order models can be constructed in an intrusive or non-intrusive manner. The reduced-order solutions of the physical quantities of interest can then be recovered through a combination of the reduced basis functions weighted by the predicted expansion coefficients.

Without having access to the source codes of full-order numerical systems, non-intrusive ROM provides a flexible and effective approach to characterising the parametric dependence of nonlinear systems [1, 2, 20]. In essence, non-intrusive parametric ROM using projection-based methods [10, 11] is to construct a surrogate model that approximates the mapping relation between the configuration parameters and the expansion coefficients. It is easy to see the natural connection between non-intrusive ROM and artificial intelligence (AI). Currently, various machine/deep learning algorithms have been incorporated into ROM to establish the nonlinear mapping between the input parameters and the output expansion coefficients [4, 8, 20, 21], due to the powerful capacities of AI in complex-data analysis and hidden-rule exploration. From the perspective of the 'fuel' to drive the 'engine' of machine/deep learning, the AI-based ROM can be roughly divided into three categories: data-driven modelling, physics-informed modelling (physics-guided modelling) and physics-data combined modelling (the proposed approach), as illustrated in Figure 1.

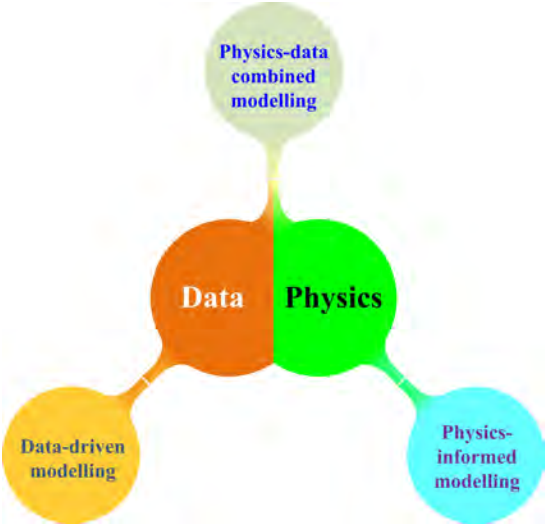


Figure 1: Schematic illustration of three categories of machine/deep learning-based modelling: data-driven modelling, physics-informed modelling and physics-data combined modelling.

Data-driven machine learning [22, 23, 24] could be the most effective and easiest approach to representing the parametric dependences of nonlinear problems when sufficient labelled training data is available. And data-driven learning usually converges rapidly, because the loss functions are often simple. Although data-driven modelling is capable of fitting the observed data very well, it usually possesses a poor generalization performance for unseen data, leading to remarkable prediction error or extrapolation bias [2, 25]. Besides, data-driven models work in a black-box manner, because interpretable information or knowledge can rarely be inferred from the data deluge. It is still expensive to acquire large volumes of computational or experimental data in many application fields [3, 8]. For instance, each data point in the parameter space of super-large-scale turbulence flow requires an extremely costly numerical simulation or experimental measurement [26, 27]. The very limited amount of training data can be too sparse to construct an accurate data-driven model, and thus the predictive capacity can be weak in aspects of both interpolation and extrapolation.

Physics-informed machine learning [4, 8, 21, 28] integrates governing physical rules and domain knowledge into machine/deep learning to provide strong theoretical constraints and inductive biases. The embedding of prior scientific principles can be leveraged to enhance the generalization performance of predictive models. This novel learning philosophy significantly reduces the data dependency of traditional data-driven learning, even label-free in many situations [3, 29, 30]. However, physics-informed modelling usually requires a deep neural network architecture to embed physical laws, and the loss function often consists of multiple terms, forming a complicated non-convex optimization problem that requires intensive computation to solve [25]. Current physics-informed learning struggles to preserve the high-frequency features in the target solution, because it is hard to accurately penalize the PDE residuals for the sharp regions with steep gradients [25, 31, 32]. Moreover, high-order derivatives in PDEs are commonly involved in the loss function of physics-informed learning, and repeatedly computation of them also makes the model training phase very time-consuming [29, 30]. Purely physics-informed leaning [4, 28] doesn't guarantee convergence to the global optimum, because high-order derivatives extracted from the unsmooth/sharp areas of the reduced basis functions by using classical numerical schemes (such as finite difference and Chebyshev pseudospectral methods) are often less accurate. The above limitations may explain why purely physical-informed learning is rarely applied to large-scale and complicated systems.

According to the above introductions, it is clear that data-driven learning and physics-informed learning have complementary strengths. Therefore, how to take full advantage of these two learning paradigms to achieve a more cost-effective and reliable ROM method for nonlinear problems is a question worth exploring. In fact, the vast majority of scientific and engineering problems have both insufficient data and (partially) known physical principles [7, 25, 33]. This is even more true with parametric model order reduction using projection-based methods [9, 10, 11]. A necessary number of snapshots resulting from high-fidelity numerical simulations are the prerequisite to generating the reduced basis functions, and the underlying governing equations are undoubtedly already known, both of which play the roles of the data component and the physics component respectively. However, it is still an open challenge to construct credible reduced-order models via machine learning when the labelled training data is very limited and sparse [3, 8, 26, 27].

Motivated by the above-unsolved problem, this study is devoted to developing a physics-data combined machine learning (PDCML) method for non-intrusive parametric ROM of nonlinear dynamical systems in small-data regimes. The proposed methodology seamlessly integrates the governing physical laws and the limited labelled data into feedforward neural networks (FNN) through a step-by-step training scheme. To be specific, the starting step is a purely data-driven training process to generate a preliminary FNN model, and this model is further enhanced in terms of interpolation and extrapolation performances through constrained optimisations in the following steps, where governing physical rules are embedded in the loss functions with the constraint of labelled data. A dynamic weight factor is properly designed to adjust the physics-data proportion of the loss function, in order to continuously highlight the physics loss term as the primary optimization objective and keep the data loss term as the constraint. To verify the effectiveness of the proposed method, it is tested on a series of nonlinear problems and also compared to the data-driven ROM, the physics-guided ROM and the traditional projection-based ROM methods.

The remainder of this paper is organised as follows: Section 2 briefly introduces the concept and basic theory of parametric model order reduction; The methodology of non-intrusive parametric ROM using FNN is provided in Section 3, where the data-driven ROM, the physics-guided ROM and the physics-data combined ROM methods are detailedly explained; In Section 4, different types of ROM methods are applied to a series of nonlinear problems and compared in terms of prediction performances; Section 5 discusses the strengths and weaknesses of the proposed method; Finally, the main contributions of this study are summarised in Section 6.

2. Parametric model order reduction

2.1. Parameterized full-order numerical model

This study considers a nonlinear dynamical system that maps the input parameter(s) onto one or more physical field variable(s). The physical field variable $\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu})$ varies with respect to space $\mathbf{x} \in \mathcal{X}$ and time $t \in \mathcal{T}$, and also relies on the input parameter(s) $\boldsymbol{\mu} \in \mathcal{P}$, which can be denoted by the following function:

$$\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu}) : \mathcal{X} \times \mathcal{T} \times \mathcal{P} \to \mathbb{R} \tag{1}$$

where $\mathcal{X}$, $\mathcal{T}$ and $\mathcal{P}$ denote the space, time and parameter domains, respectively. Here, the physical field variable $\mathbf{u}$ can be velocity components, pressure, temperature, etc.

The evolution of this nonlinear dynamical system is governed by its underlying physical laws, which are often represented in the form of parametrized partial differential equations (PDEs). It is extremely difficult, even impossible to analytically solve the nonlinear PDEs in most situations. Resorting to numerical methods (such as finite difference, finite element and finite volume methods) is the routine way to approximately solve the problems [34, 35, 36], where the governing PDEs are discretized and reformulated into a solvable system of algebraic equations that approximately characterise the nonlinear dynamical system's behaviours. The general form of the parameterized full-order numerical model can be expressed as follows:

$$\mathcal{F}\Big(\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu}),\ \mathbf{x},\ t,\ \boldsymbol{\mu}\Big) = s(\mathbf{x},\ t;\ \boldsymbol{\mu}) \tag{2}$$

where $\mathcal{F}(\,\cdot\,)$ is the function that models the dynamic processes and conversation laws, and $s(\mathbf{x},\ t;\ \boldsymbol{\mu})$ denotes the source term.

To achieve accurate numerical approximations, physical fields that are continuous in the space-time domain have to be meshed into thousands or millions of tiny elements, and each element has a specific number of degrees of freedom. Therefore, the full-order numerical model that embeds the governing PDEs is often in typically high dimension, no matter which numerical discretization method is chosen. To evaluate this full-order numerical model, a large-scale system of equations with a huge number of unknowns has to be solved. Motivated by the urgent requirements of fast and accurate computations in many modern engineering and scientific applications, model order reduction (MOR) techniques are extensively developed to seek the latent low-dimensional representations of the full-order numerical models, in order to efficiently resolve the primary dynamics of nonlinear systems.

2.2. *Proper orthogonal decomposition*

Here, proper orthogonal decomposition (POD) [9, 10, 11] is adopted to derive the low-dimensional parametrization from the numerical discretization model, where an optimal set of orthogonal basis functions are empirically extracted from high-fidelity numerical solutions (also called "snapshots"). To compute the POD basis, a set of $\mathcal{N}_p$ snapshots of the physical field(s) $\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}) \in \mathbb{R}^{\mathcal{N}_x \times \mathcal{N}_t}$ corresponding to the varying parameter(s) $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1,\ \boldsymbol{\mu}_2,\ \ldots\ \boldsymbol{\mu}_i,\ \ldots\ ,\boldsymbol{\mu}_{\mathcal{N}_p}\} \in \mathcal{P}$ are considered. Each snapshot can be further rearranged into a column vector $\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_i) \in \mathbb{R}^{\mathcal{N}}$, and merging them yields a 2D matrix $\mathbf{U}$, given by:

$$\mathbf{U} = \Big[\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_1),\ \mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_2),\ \ldots\ ,\ \mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_i),\ \ldots\ ,\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_{\mathcal{N}_p})\Big] \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}_p} \qquad (1 \leq i \leq \mathcal{N}_p) \tag{3}$$

where $\mathcal{N}_x$ and $\mathcal{N}_t$ are the dimensions of the numerical discretization of the spatial domain and time domain respectively, $\mathcal{N}_p$ is the number of parameters that are generated from an appropriate data sampling, and $\mathcal{N} = \mathcal{N}_x \times \mathcal{N}_t$.

Before performing POD, these snapshots are standardised by removing the fixed component, called "particular solution" or "static correction" $\overline{\mathbf{u}} \in \mathbb{R}^{\mathcal{N}}$, and a standardised snapshot matrix $\widetilde{\mathbf{U}}$ is thus obtained:

$$\begin{aligned}\widetilde{\mathbf{U}} &= \Big[\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_1) - \overline{\mathbf{u}},\ \mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_2) - \overline{\mathbf{u}},\ \ldots\ ,\ \mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_i) - \overline{\mathbf{u}},\ \ldots\ ,\mathbf{u}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_{\mathcal{N}_p}) - \overline{\mathbf{u}}\Big] \\ &= \Big[\widetilde{\mathbf{u}}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_1),\ \widetilde{\mathbf{u}}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_2),\ \ldots\ ,\ \widetilde{\mathbf{u}}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_i),\ \ldots\ ,\widetilde{\mathbf{u}}(\cdot\ ,\cdot\ ;\boldsymbol{\mu}_{\mathcal{N}_p})\Big] \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}_p}\end{aligned} \tag{4}$$

where $\overline{\mathbf{u}} \in \mathbb{R}^{\mathcal{N}}$ is computed as the mean of all snapshots [37], and it is considered irrelevant to the parametric dependence of the studied system.

Applying singular value decomposition (SVD) to the matrix $\widetilde{\mathbf{U}}$ results in the following equation:

$$\widetilde{\mathbf{U}} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{W}^{\mathrm{T}} = \sum_{j=1}^{\mathcal{N}_p} \sigma_j v_j w_j^{\mathrm{T}} \tag{5}$$

where $\mathbf{V} \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}_p}$ is a matrix whose columns are orthogonal and represent the left-singular vectors, $\mathbf{W} \in \mathbb{R}^{\mathcal{N}_p \times \mathcal{N}_p}$ is a matrix with orthogonal columns representing the right-singular vectors, and $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1,\ \sigma_2,\ \ldots\ ,\ \sigma_{\mathcal{N}_p}) \in \mathbb{R}^{\mathcal{N}_p \times \mathcal{N}_p}$ is a diagonal matrix with positive entries that are called singular values and listed in descending order.

To obtain an efficient low-dimensional representation of the snapshot data, only the first $\mathcal{N}_{\mathrm{r}}$ columns of the matrix V (left-singular vectors) that corresponds to the $\mathcal{N}_r$ largest singular values are chosen as the POD basis functions. This orthogonal POD basis is denoted as:

$$\boldsymbol{\Phi} = \Big[\phi_1,\ \phi_2,\ \ldots\ \phi_k,\ \ldots\ ,\ \phi_{\mathcal{N}_{\mathrm{r}}}\Big] \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}_{\mathrm{r}}} \tag{6}$$

where each POD mode $\phi_k \in \mathbb{R}^{\mathcal{N}}$ is the k-th left-singular vector in the matrix $\mathbf{V}$. It should be noted that $\boldsymbol{\Phi}$ is the optimal orthogonal basis of size $\mathcal{N}_{\mathrm{r}}$ to minimise the least square error of snapshot reconstruction.

Due to the truncation in selecting the POD basis functions, the information of snapshot data is inevitably lost. Such information loss $\mathcal{L}_\mathrm{r}$ can be quantitatively measured by the sum of the squares of the neglected singular values, given by:

$$\mathcal{L}_\mathrm{r} = \sum_{j=\mathcal{N}_r+1}^{\mathcal{N}_p} \sigma_j^2 \tag{7}$$

Therefore, the size $\mathcal{N}_\mathrm{r}$ of the POD basis can be determined from the singular values. To provide quantitative guidance, the ratio β of the covered information to the complete information is compared to a user-specified tolerance $\eta \lesssim 1$, given by:

$$\beta = \frac{\sum_{j=1}^{\mathcal{N}_r} \sigma_j^2}{\sum_{j=1}^{\mathcal{N}_p} \sigma_j^2} \geq \eta \tag{8}$$

As $\mathcal{N}_\mathrm{r}$ is increased to $\mathcal{N}_p$, the value of β will tend to 1, which would imply zero information loss.

2.3. Parameterized reduced-order numerical model

Learned from the snapshot data of the nonlinear dynamical system, the POD basis $\mathbf{\Phi}$ can provide a physics-based parametrization of the physical field variable(s). The complexity of this nonlinear problem would be significantly simplified if $\mathcal{N}_\mathrm{r} \ll \mathcal{N}_p$. And the full-order solution $\mathbf{u}(\mathbf{x}, t;\ \boldsymbol{\mu}_i)$ throughout the entire space-time computation domain can be approximated by its projection onto the subspace:

$$\mathbf{u}(\cdot\,,\cdot\,;\boldsymbol{\mu}_i) \approx \mathbf{u}_\mathrm{r}(\cdot\,,\cdot\,;\boldsymbol{\mu}_i) = \overline{\mathbf{u}} + \mathbf{\Phi}\alpha(\boldsymbol{\mu}_i) = \overline{\mathbf{u}} + \sum_{k=1}^{\mathcal{N}_\mathrm{r}} \phi_k \alpha_k(\boldsymbol{\mu}_i) \in \mathbb{R}^{\mathcal{N}} \tag{9}$$

where $\mathbf{u}_\mathrm{r}(\cdot\,,\cdot\,;\boldsymbol{\mu}_i)$ denotes the reduced-order solution, and $\alpha_k(\boldsymbol{\mu}_i)$ is the k-th POD expansion coefficient. Besides, the POD expansion coefficients $\boldsymbol{\alpha}(\boldsymbol{\mu}_i)$ can also be computed by projecting the snapshots onto the subspace:

$$\alpha(\boldsymbol{\mu}_i) = \mathbf{\Phi}^\mathrm{T}\,\widetilde{\mathbf{u}}(\cdot\,,\cdot\,;\boldsymbol{\mu}_i) = \mathbf{\Phi}^\mathrm{T}\Big[\mathbf{u}(\cdot\,,\cdot\,;\boldsymbol{\mu}_i) - \overline{\mathbf{u}}\Big] \in \mathbb{R}^{\mathcal{N}_\mathrm{r}} \tag{10}$$

Obviously, there is a mapping relation between the system parameter $\boldsymbol{\mu}$ and the POD expansion coefficients $\boldsymbol{\alpha}$, which is the low-dimensional representation of the parametric dependency of the nonlinear dynamical system. To physically characterise the linkage between $\boldsymbol{\mu}$ and $\boldsymbol{\alpha}$, the reduced-order numerical model that embeds the governing parameterized PDEs can be derived from the full-order numerical model and the POD basis functions. Substituting the low-dimensional approximation of the physical field $\mathbf{u}_\mathrm{r}(\cdot\,,\cdot\,;\boldsymbol{\mu})$ into the full-order numerical model in Eq. (2) and then projecting the obtained result onto the subspace yield the reduced-order numerical model:

$$\mathbf{\Phi}^\mathrm{T}\mathcal{F}\Big(\overline{\mathbf{u}} + \mathbf{\Phi}\alpha,\ \mathbf{x},\ t,\ \boldsymbol{\mu}\Big) = \mathbf{\Phi}^\mathrm{T} s(\mathbf{x},\ t;\ \boldsymbol{\mu}) \tag{11}$$

Traditional projection-based methods [11] assemble the above reduced-order model in an intrusive manner, where high-fidelity system operators are projected onto the subspace. Generally, traditional projection-based ROM can well maintain the underlying structure of the full-order model, and the system-theoretic error can also be derived in a rigorous way. On the other side of the coin, the level of access to system operators often limits the usage of traditional projection-based methods in many cases. To decouple the online stage from the high-fidelity system operators, a non-intrusive ROM framework is developed in the following section, by embedding the reduced-order numerical system into feedforward neural networks.

3. Construction of reduced-order models

Essentially, non-intrusive projection-based ROM is to construct the mapping relation between the varying parameter(s) $\boldsymbol{\mu}$ and the POD expansion coefficients $\boldsymbol{\alpha}$, thereby achieving a low-dimensional representation of the parametric dependence for the studied dynamical system. To some extent, the reduced-order numerical system (Eq. (11)) can be considered as a representation of the *physical essence* behind the $\boldsymbol{\mu}$-$\boldsymbol{\alpha}$ mapping relation. Considering the high cost to obtain a large number of high-fidelity snapshots, this study aims to develop a cost-effective ROM method for nonlinear problems in small-data regimes, where physical principles combined with the limited available data are integrated into the machine learning model to enhance its prediction performance and generalization capacity. The methodology of the novel physics-data combined ROM is graphically illustrated in Figure 2, and detailed explanations are provided in the following subsections.

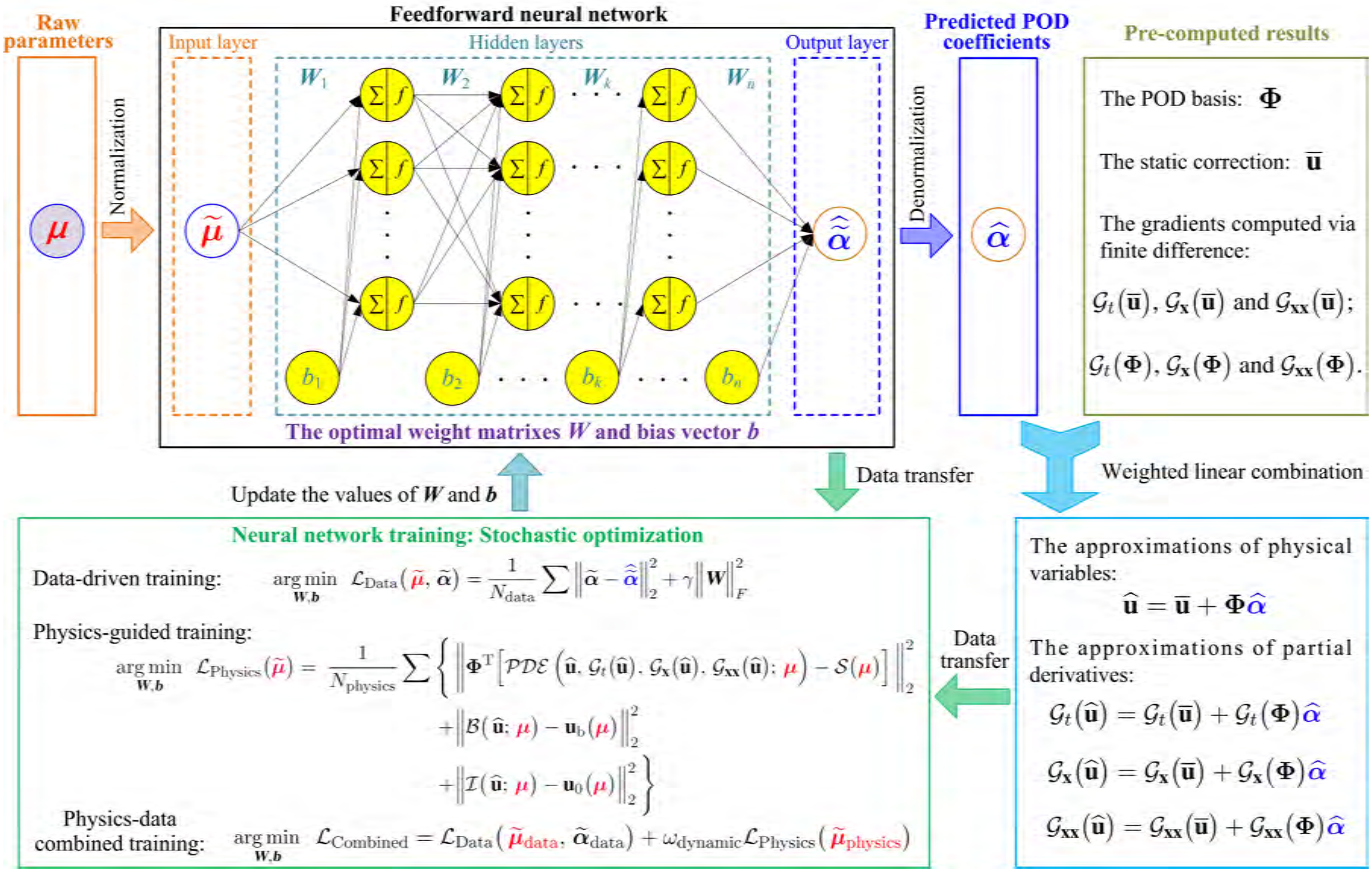

Raw parameters
μ
Normalization
Feedforward neural network
Input layer
Hidden layers
Output layer
$\tilde{\mu}$
W_1
W_2
W_k
W_n
Σ f
b_1
b_2
b_k
b_n
$\hat{\tilde{\alpha}}$
The optimal weight matrixes W and bias vector b
Denormalization
Predicted POD coefficients
$\hat{\alpha}$
Pre-computed results
The POD basis: $\mathbf{\Phi}$
The static correction: $\overline{\mathbf{u}}$
The gradients computed via finite difference:
$\mathcal{G}_t(\overline{\mathbf{u}}), \mathcal{G}_{\mathbf{x}}(\overline{\mathbf{u}})$ and $\mathcal{G}_{\mathbf{xx}}(\overline{\mathbf{u}})$;
$\mathcal{G}_t(\mathbf{\Phi}), \mathcal{G}_{\mathbf{x}}(\mathbf{\Phi})$ and $\mathcal{G}_{\mathbf{xx}}(\mathbf{\Phi})$.
Weighted linear combination
Data transfer
Update the values of W and b
Neural network training: Stochastic optimization
Data-driven training: $\underset{\mathbf{W},\boldsymbol{b}}{\arg\min}\ \mathcal{L}_{\text{Data}}(\tilde{\mu}, \tilde{\alpha}) = \frac{1}{N_{\text{data}}}\sum \left\|\tilde{\alpha} - \hat{\tilde{\alpha}}\right\|_2^2 + \gamma\left\|\mathbf{W}\right\|_F^2$
Physics-guided training: $\underset{\mathbf{W},\boldsymbol{b}}{\arg\min}\ \mathcal{L}_{\text{Physics}}(\tilde{\mu}) = \frac{1}{N_{\text{physics}}}\sum\left\{\left\|\mathbf{\Phi}^{\mathrm{T}}\left[\mathcal{PDE}\left(\hat{\mathbf{u}}, \mathcal{G}_t(\hat{\mathbf{u}}), \mathcal{G}_{\mathbf{x}}(\hat{\mathbf{u}}), \mathcal{G}_{\mathbf{xx}}(\hat{\mathbf{u}}); \mu\right) - \mathcal{S}(\mu)\right]\right\|_2^2 + \left\|\mathcal{B}(\hat{\mathbf{u}}; \mu) - \mathbf{u}_{\text{b}}(\mu)\right\|_2^2 + \left\|\mathcal{I}(\hat{\mathbf{u}}; \mu) - \mathbf{u}_0(\mu)\right\|_2^2\right\}$
Physics-data combined training: $\underset{\mathbf{W},\boldsymbol{b}}{\arg\min}\ \mathcal{L}_{\text{Combined}} = \mathcal{L}_{\text{Data}}(\tilde{\mu}_{\text{data}}, \tilde{\alpha}_{\text{data}}) + \omega_{\text{dynamic}}\mathcal{L}_{\text{Physics}}(\tilde{\mu}_{\text{physics}})$
Data transfer
The approximations of physical variables:
$\hat{\mathbf{u}} = \overline{\mathbf{u}} + \mathbf{\Phi}\hat{\alpha}$
The approximations of partial derivatives:
$\mathcal{G}_t(\hat{\mathbf{u}}) = \mathcal{G}_t(\overline{\mathbf{u}}) + \mathcal{G}_t(\mathbf{\Phi})\hat{\alpha}$
$\mathcal{G}_{\mathbf{x}}(\hat{\mathbf{u}}) = \mathcal{G}_{\mathbf{x}}(\overline{\mathbf{u}}) + \mathcal{G}_{\mathbf{x}}(\mathbf{\Phi})\hat{\alpha}$
$\mathcal{G}_{\mathbf{xx}}(\hat{\mathbf{u}}) = \mathcal{G}_{\mathbf{xx}}(\overline{\mathbf{u}}) + \mathcal{G}_{\mathbf{xx}}(\mathbf{\Phi})\hat{\alpha}$

Figure 2: The workflows of non-intrusive parametric ROM of nonlinear dynamical systems via the data-driven method, the physics-guided method and the physics-data combined method.

3.1. Data preparation

Consider the scenario where high-fidelity snapshots are very limited but sufficient to yield an orthogonal reduced basis for a certain parameter domain. The available snapshot data $\mathbf{U}$ corresponds to a set of properly-sampled parameters $\boldsymbol{\mu}_{\text{data}}$, from which a set of POD basis functions $\boldsymbol{\Phi}$ can be derived by performing POD. The projection coefficients $\boldsymbol{\alpha}_{\text{data}}$ of the snapshots $\mathbf{U}$ onto to the reduced space can be computed according to Eq. (10). The obtained data pairs $(\boldsymbol{\mu}_{\text{data}}, \boldsymbol{\alpha}_{\text{data}})$ are the only available labelled data to construct a reduced-order model via non-intrusive projection-based ROM methods.

For ROM using machine learning, the scale of training data can affect both the training speed and quality. Data normalisation is thus required to standardise the input and output. As the input features, parameters $\boldsymbol{\mu} \supset \boldsymbol{\mu}_{\text{data}}$ within a range of interest can be scaled by using the minimum and maximum values, given by:

$$\widetilde{\boldsymbol{\mu}} = \frac{\boldsymbol{\mu} - \frac{\boldsymbol{\mu}_{\max}+\boldsymbol{\mu}_{\min}}{2}}{\frac{\boldsymbol{\mu}_{\max}-\boldsymbol{\mu}_{\min}}{2}} = \frac{2\boldsymbol{\mu} - (\boldsymbol{\mu}_{\max} + \boldsymbol{\mu}_{\min})}{\boldsymbol{\mu}_{\max} - \boldsymbol{\mu}_{\min}} \tag{12}$$

As to the output labels, POD expansion coefficients $\boldsymbol{\alpha}_{\text{data}}$ can only be statistically normalised based on the available projection data, because their value ranges are unknown.

$$\widetilde{\boldsymbol{\alpha}}_{\text{data}} = \frac{\boldsymbol{\alpha}_{\text{data}} - \overline{\boldsymbol{\alpha}}_{\text{data}}}{\sigma_{\boldsymbol{\alpha}}} \tag{13}$$

where $\overline{\boldsymbol{\alpha}}_{\text{data}}$ and $\sigma_{\boldsymbol{\alpha}}$ are the mean value and standard deviation of $\boldsymbol{\alpha}_{\text{data}}$ respectively.

3.2. Feedforward neural network

Feedforward neural network (FNN) is an ideal tool to deal with nonlinear regression problems for model order reduction, due to its powerful capacity in analysing complex data and exploring hidden rules. An FNN model can be considered as a function approximation to map the inputs to the outputs through many interconnected computation elements called neurons [38]. Each elementary neuron possesses a certain degree of approximation ability, and a powerful learning capacity can be achieved by cohesively connecting many neuron nodes. According to the universal approximation theorem, FNN can be properly designed and trained to approximate any given continuous function to desired accuracy [39].

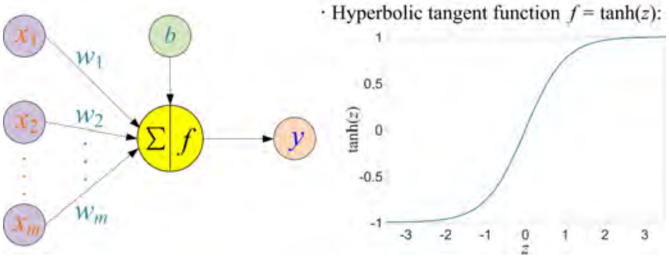


Figure 3: Schematic illustration of an elementary neuron.

The working mechanism of an elementary neuron is illustrated in Figure 3, where the weighted sum of inputs x_i plus the bias b is calculated through a transfer function Σ, and then this summation is fed to a nonlinear activation function f to yield an output y:

$$y = f\left(\sum_{i=1}^{m} w_i x_i + b\right) \tag{14}$$

where m is the number of inputs, and w_i denotes a weight. The activation function f can be a sigmoid function, hyperbolic tangent function, rectified linear unit or others. Here, hyperbolic tangent function $\tanh(z)$ is adopted to transform the continuous data z into a value between -1 and 1 (as illustrated in Figure 3):

$$\tanh(z) = \frac{\mathrm{e}^{z} - \mathrm{e}^{-z}}{\mathrm{e}^{z} + \mathrm{e}^{-z}} \tag{15}$$

An FNN model is a multi-layer architecture with many connected neurons, in which the output from a prior neuron is used as the input of the following neuron. A graphic illustration of a multi-layer neural network model is provided in Figure 2.

To establish the nonlinear linkage between the varying parameter and corresponding expansion coefficients via FNN, the normalised parameter $\widetilde{\boldsymbol{\mu}}$ plays the role of input features, expecting to obtain an accurate prediction of the normalised data $\widetilde{\boldsymbol{\alpha}}$. As an estimation of $\widetilde{\boldsymbol{\alpha}}$, the final FNN output $\widehat{\widetilde{\boldsymbol{\alpha}}}$ is computed through a series of forward-propagation equations occurring at particular layers, given by:

$$\begin{cases} \boldsymbol{y}_0 = \widetilde{\boldsymbol{\mu}} \\ \boldsymbol{y}_k = \tanh\left(\boldsymbol{W}_k^{\mathrm{T}} \boldsymbol{y}_{k-1} + b_k\right), \quad k \in [1, n-1] \\ \boldsymbol{y}_n = \boldsymbol{W}_n^{\mathrm{T}} \boldsymbol{y}^{n-1} + b_n \\ \widehat{\widetilde{\boldsymbol{\alpha}}} = \boldsymbol{y}_n \end{cases} \tag{16}$$

where $\boldsymbol{y}_0$, $\boldsymbol{y}_k$ and $\boldsymbol{y}_n$ denote the outputs of the input layer, the k-th hidden layer and the final output layer respectively; $n + 1$ is the total number of layers contained in the FNN architecture; $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight matrixes and the biases respectively.

Therefore, this FNN model can be considered as a vector-valued network surrogate to approximate the input-output relation of the $\widetilde{\boldsymbol{\mu}}$-$\widetilde{\boldsymbol{\alpha}}$ mapping, which can be mathematically expressed as follows:

$$\mathcal{FNN}\left(\widetilde{\boldsymbol{\mu}};\ \boldsymbol{W},\ \boldsymbol{b}\right):\ \widetilde{\boldsymbol{\mu}} \in \mathbb{R}^{d_\mu} \to \widetilde{\boldsymbol{\alpha}} \in \mathbb{R}^{\mathcal{N}_{\mathrm{r}}} \tag{17}$$

where $\mathcal{FNN}(\,\cdot\,)$ denotes the approximation function of the FNN model; d_μ and $\mathcal{N}_{\mathrm{r}}$ denote the dimensions of a normalised parameter $\widetilde{\boldsymbol{\mu}}$ and the normalised POD expansion coefficients $\widetilde{\boldsymbol{\alpha}}$ respectively.

Once the FNN model is properly trained, it can be used to predict the values of $\widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{new}}$ by inputting a new normalised parameter $\widetilde{\boldsymbol{\mu}}_{\mathrm{new}}$. Performing data denormalisation on $\widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{new}}$, the predictions of POD expansion coefficients $\widehat{\boldsymbol{\alpha}}_{\mathrm{new}}$ can then be obtained:

$$\widehat{\boldsymbol{\alpha}}_{\mathrm{new}} = \mathcal{D}\Big(\mathcal{FNN}(\widetilde{\boldsymbol{\mu}}_{\mathrm{new}};\ \boldsymbol{W},\ \boldsymbol{b})\Big) = \mathcal{D}\left(\widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{new}}\right) = \widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{new}} \sigma_{\boldsymbol{\alpha}} + \overline{\boldsymbol{\alpha}}_{\mathrm{data}} \tag{18}$$

where $\mathcal{D}(\,\cdot\,)$ denotes the denormalisation operation on the normalised POD expansion coefficients. Submitting the estimated expansion coefficients $\widehat{\boldsymbol{\alpha}}_{\mathrm{new}}$ into Eq. (9), the reduced-order solution $\mathbf{u}_{\mathrm{r}}(\cdot\,,\cdot\,;\boldsymbol{\mu}_{\mathrm{new}})$ of the physical field variable can be instantly acquired from the linear combination of POD basis functions.

However, the key issue here is how to optimally adjust the weight matrixes $\boldsymbol{W}$ and the bias vector $\boldsymbol{b}$ of the neural network, by making full use of both the available labelled data and the known physical principles. Crucially, the optimization objective to train the neural network should be carefully chosen, which is important to reach the optimum state that best represents the $\widetilde{\boldsymbol{\mu}}$-$\widetilde{\boldsymbol{\alpha}}$ mapping relation. In the following subsections, three types of FNN-based model order reduction methods are detailedly explained, including data-driven ROM, physics-guided ROM, and physics-data combined ROM.

3.3. Data-driven ROM

After performing POD and data normalisation on the snapshot data, a limited number of data pairs $(\widetilde{\boldsymbol{\mu}}_{\mathrm{data}},\ \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}})$ can be obtained. Naturally, these data pairs can be used as the labelled training data to fit an FNN model via supervised learning, although they could be too sparse to yield a reliable reduced-order model in the small-data regime. The methodology of data-driven ROM using FNN is straightforward, and the training process is also simple and efficient. The main procedures to construct a reduced-order model via data-driven machine learning are graphically summarised in Figure 2.

In essence, data-driven ROM is to optimise the weight matrixes $\boldsymbol{W}$ and the bias vector $\boldsymbol{b}$ of the neural network by minimising of the discrepancy between the output $\widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{data}}$ and the target $\widetilde{\boldsymbol{\alpha}}_{\mathrm{data}}$. This optimization problem can be mathematically expressed as follows:

$$\begin{aligned} \underset{\boldsymbol{W},\boldsymbol{b}}{\arg\min}\ \ \mathcal{L}_{\mathrm{Data}}\left(\widetilde{\boldsymbol{\mu}}_{\mathrm{data}},\ \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}}\right) &= \frac{1}{N_{\mathrm{data}}} \sum_{\widetilde{\boldsymbol{\alpha}}_{\mathrm{data}}} \left\| \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}} - \widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{data}} \right\|_2^2 + \gamma \left\| \boldsymbol{W} \right\|_F^2 \\ &= \frac{1}{N_{\mathrm{data}}} \sum_{\widetilde{\boldsymbol{\alpha}}_{\mathrm{data}}} \left\| \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}} - \mathcal{FNN}\left(\widetilde{\boldsymbol{\mu}}_{\mathrm{data}};\ \boldsymbol{W},\ \boldsymbol{b}\right) \right\|_2^2 + \gamma \left\| \boldsymbol{W} \right\|_F^2 \end{aligned} \tag{19}$$

where $\mathcal{L}_{\mathrm{Data}}(\widetilde{\boldsymbol{\mu}}_{\mathrm{data}},\ \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}})$ denotes the data-driven loss function to train the neural network, N_{data} is the number of the training data pairs $(\widetilde{\boldsymbol{\mu}}_{\mathrm{data}},\ \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}})$, and $\gamma \geq 0$ is the weight regulation constant. The first term $\frac{1}{N_{\mathrm{data}}} \sum \left\| \widetilde{\boldsymbol{\alpha}}_{\mathrm{data}} - \widehat{\widetilde{\boldsymbol{\alpha}}}_{\mathrm{data}} \right\|_2^2$

of the loss function is the mean square error to represent the discrepancy between $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ and $\widehat{\widetilde{\boldsymbol{\alpha}}}_{\text{data}}$. The second term $\gamma\left\|\boldsymbol{W}\right\|_F^2$ is the L_2 weight regulation term, also called weight decay, which can force the network response to be smoother and thus greatly reduce overfitting.

3.4. Physics-guided ROM

Data-driven ROM fits the surrogate model and works in a 'black-box' manner, and thus interpretable knowledge can scarcely be inferred from the training data. Besides, an important fact that is often ignored by data-driven ROM is that the governing PDEs behind high-fidelity numerical simulations are clearly known. As explained in Section 2.3, the the reduced-order numerical system in Eq. (11) physically characterises the intrinsic relation of $\boldsymbol{\mu}$-$\boldsymbol{\alpha}$ mapping. Such scientific knowledge should be leveraged on model order reduction to compensate for data deficiency. How to incorporate this physical knowledge into machine learning-based ROM is a critical issue to cope with the shortage of labelled training data ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$).

Traditional intrusive projection-based ROM [11] assembles reduced-order model by projecting the high-fidelity system operators onto the reduced subspace, as illustrated in Eq. (11). The level of access to system operators usually limits the use of traditional projection-based ROM, despite its various advantages. In this part, a physics-guided ROM framework is developed by embedding governing physical laws into feedforward neural networks, where the parameterized reduced-order system is solved through stochastic optimization. This physics-guided ROM framework fully decouples the online stage from the high-fidelity numerical simulation method, and the neural network training process is also completely label-free. To clearly explain the methodology, a flow chart is drawn in Figure 2 for graphical illustration.

In contrast to data-driven training, the optimization objective of physical-guided training is not to minimise the discrepancy between the outputs and the targets, but to force the learned input-output relation to follow the physical principle. In this work, the governing parameterized PDEs are used as the physical principle for physics-guided training, which can be expressed as follows:

$$\begin{cases} \mathcal{PDE}\left(\mathbf{u},\, \frac{\partial \mathbf{u}}{\partial t},\, \frac{\partial \mathbf{u}}{\partial \mathbf{x}},\, \frac{\partial^2 \mathbf{u}}{\partial \mathbf{x}^2};\, \boldsymbol{\mu}\right) = \mathcal{S}(\boldsymbol{\mu}) \\ \mathcal{B}(\mathbf{u};\, \boldsymbol{\mu}) = \mathbf{u}_{\text{b}}(\boldsymbol{\mu}) \\ \mathcal{I}(\mathbf{u};\, \boldsymbol{\mu}) = \mathbf{u}_0(\boldsymbol{\mu}) \end{cases} \tag{20}$$

where $\mathcal{PDE}$ denotes the basic arithmetic operations in the governing PDEs, $\mathcal{B}$ denotes the boundary condition operator, and $\mathcal{I}$ denotes the initial condition operator; $\mathcal{S}(\boldsymbol{\mu})$ is the source term, $\mathbf{u}_{\text{b}}(\boldsymbol{\mu})$ represents the boundary conditions, and $\mathbf{u}_0(\boldsymbol{\mu})$ denotes the initial conditions. High-order partial derivatives can also be considered in the above PDEs, although the testing cases in this work are quadratically nonlinear PDEs with the first- and second-order partial derivatives, such as the Navier-Stocks equations.

With the FNN model, a physical field $\widehat{\mathbf{u}}(\boldsymbol{\mu})$ can be predicted by inputting an arbitrary parameter $\boldsymbol{\mu}$ to it, and this procedure can be expressed by substituting Eq. (18) into Eq. (9):

$$\widehat{\mathbf{u}}(\boldsymbol{\mu}) = \overline{\mathbf{u}} + \boldsymbol{\Phi}\widehat{\alpha}(\boldsymbol{\mu}) = \overline{\mathbf{u}} + \boldsymbol{\Phi}\mathcal{D}\left(\mathcal{FNN}(\widetilde{\boldsymbol{\mu}};\, \boldsymbol{W},\, \boldsymbol{b})\right) \tag{21}$$

where $\widetilde{\boldsymbol{\mu}}$ denotes the normalised data of $\boldsymbol{\mu}$ (as defined by Eq. (12)). The predicted physical field $\widehat{\mathbf{u}}(\boldsymbol{\mu})$ can then be reshaped to the same dimensions as the raw snapshots. Here, the high-fidelity snapshots are assumed to be in the regular mesh grid, which is common for numerical simulations using finite difference, finite volume and lattice Boltzmann methods. Even for unstructured mesh grids, interpolation techniques can be used for data transformation. The regularly meshed snapshots are highly efficient and convenient for data computation in machine learning, since the neighbourhood relationships are defined by storage arrangement.

Due to the regular mesh in snapshots, finite difference [40, 41] can be performed on the predicted physical field $\widehat{\mathbf{u}}(\boldsymbol{\mu})$ to quickly compute the gradients with respect to space $\mathbf{x}$ and time t. These gradients of different orders are the reliable approximation of the partial derivatives in the governing PDEs (Eq. (20)). In this work, central finite differences with high accuracy orders are adopted, and more information about using the finite difference to approximate partial derivatives is provided in Appendix I. Taking account of Eq. (21), the following linear relationships hold:

$$\mathcal{G}_t\left(\widehat{\mathbf{u}}(\boldsymbol{\mu})\right) = \mathcal{G}_t\left(\overline{\mathbf{u}}\right) + \mathcal{G}_t\left(\boldsymbol{\Phi}\right)\widehat{\alpha}(\boldsymbol{\mu}) = \mathcal{G}_t\left(\overline{\mathbf{u}}\right) + \mathcal{G}_t\left(\boldsymbol{\Phi}\right)\mathcal{D}\left(\mathcal{FNN}(\widetilde{\boldsymbol{\mu}};\, \boldsymbol{W},\, \boldsymbol{b})\right) \tag{22}$$

$$\mathcal{G}_{\mathbf{x}}\left(\widehat{\mathbf{u}}(\boldsymbol{\mu})\right) = \mathcal{G}_{\mathbf{x}}\left(\overline{\mathbf{u}}\right) + \mathcal{G}_{\mathbf{x}}\left(\boldsymbol{\Phi}\right)\widehat{\alpha}(\boldsymbol{\mu}) = \mathcal{G}_{\mathbf{x}}\left(\overline{\mathbf{u}}\right) + \mathcal{G}_{\mathbf{x}}\left(\boldsymbol{\Phi}\right)\mathcal{D}\left(\mathcal{FNN}(\widetilde{\boldsymbol{\mu}};\, \boldsymbol{W},\, \boldsymbol{b})\right) \tag{23}$$

$$\mathcal{G}_{\mathbf{xx}}\Big(\widehat{\mathbf{u}}(\boldsymbol{\mu})\Big) = \mathcal{G}_{\mathbf{xx}}\Big(\overline{\mathbf{u}}\Big) + \mathcal{G}_{\mathbf{xx}}\Big(\mathbf{\Phi}\Big)\widehat{\alpha}(\boldsymbol{\mu}) = \mathcal{G}_{\mathbf{xx}}\Big(\overline{\mathbf{u}}\Big) + \mathcal{G}_{\mathbf{xx}}\Big(\mathbf{\Phi}\Big)\mathcal{D}\Big(\mathcal{FNN}(\widetilde{\boldsymbol{\mu}};\ \boldsymbol{W},\ \boldsymbol{b})\Big) \tag{24}$$

where $\mathcal{G}_t(\cdot)$, $\mathcal{G}_{\mathbf{x}}(\cdot)$ and $\mathcal{G}_{\mathbf{xx}}(\cdot)$ denote the finite difference operators to compute the 1st-order gradient with respect to t, the 1st-order gradient with respect to $\mathbf{x}$ and the 2nd-order gradient with respect to $\mathbf{x}$, respectively. The gradient results of $\mathcal{G}_t\Big(\widehat{\mathbf{u}}(\boldsymbol{\mu})\Big)$, $\mathcal{G}_{\mathbf{x}}\Big(\widehat{\mathbf{u}}(\boldsymbol{\mu})\Big)$ and $\mathcal{G}_{\mathbf{xx}}\Big(\widehat{\mathbf{u}}(\boldsymbol{\mu})\Big)$ are the accurate approximations of the partial derivatives $\frac{\partial \mathbf{u}}{\partial t}$, $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$ and $\frac{\partial^2 \mathbf{u}}{\partial \mathbf{x}^2}$ in the governing PDEs, respectively.

As as explained in Section 2.2, the static correction $\overline{\mathbf{u}}$ and the POD basis $\mathbf{\Phi}$ have already been obtained at the very beginning. The gradients including $\mathcal{G}_t\Big(\overline{\mathbf{u}}\Big)$, $\mathcal{G}_{\mathbf{x}}\Big(\overline{\mathbf{u}}\Big)$, $\mathcal{G}_{\mathbf{xx}}\Big(\overline{\mathbf{u}}\Big)$, $\mathcal{G}_t\Big(\mathbf{\Phi}\Big)$, $\mathcal{G}_{\mathbf{x}}\Big(\mathbf{\Phi}\Big)$ and $\mathcal{G}_{\mathbf{xx}}\Big(\mathbf{\Phi}\Big)$ can thus be precomputed through finite difference approximation, and corresponding results can be also stored for repeated use. More importantly, the partial derivative terms (including $\frac{\partial \mathbf{u}}{\partial t}$, $\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$ and $\frac{\partial^2 \mathbf{u}}{\partial \mathbf{x}^2}$) in the governing PDEs can be instantly computed through a simple linear combination of these precomputed gradient results, which can significantly speed up the training process of the physics-guided neural network. In summary, Eq. (21-24) provide a rapid way to approximate the physical variable(s) and the corresponding partial derivatives through simple linear combinations of precomputed results.

Therefore, the loss function of the physics-guided neural network for ROM can be set as the residual of the governing numerical system projected onto the subspace (Galerkin projection). The physics-guided training process of a neural network can be mathematically expressed as follows:

$$\begin{aligned}\underset{\boldsymbol{W},\boldsymbol{b}}{\arg\min}\ \ \mathcal{L}_{\text{Physics}}\Big(\widetilde{\boldsymbol{\mu}}_{\text{physics}}\Big) = \frac{1}{N_{\text{physics}}}\sum_{\boldsymbol{\mu}_{\text{physics}}}\Bigg\{&\Bigg\|\mathbf{\Phi}^{\mathrm{T}}\Big[\mathcal{PDE}\Big(\widehat{\mathbf{u}},\ \mathcal{G}_t\Big(\widehat{\mathbf{u}}\Big),\ \mathcal{G}_{\mathbf{x}}\Big(\widehat{\mathbf{u}}\Big),\ \mathcal{G}_{\mathbf{xx}}\Big(\widehat{\mathbf{u}}\Big);\ \boldsymbol{\mu}_{\text{physics}}\Big) - \mathcal{S}\Big(\boldsymbol{\mu}_{\text{physics}}\Big)\Big]\Bigg\|_2^2\\ &+\Bigg\|\mathcal{B}\Big(\widehat{\mathbf{u}};\ \boldsymbol{\mu}_{\text{physics}}\Big) - \mathbf{u}_{\mathrm{b}}\big(\boldsymbol{\mu}_{\text{physics}}\big)\Bigg\|_2^2\\ &+\Bigg\|\mathcal{I}\Big(\widehat{\mathbf{u}};\ \boldsymbol{\mu}_{\text{physics}}\Big) - \mathbf{u}_0\big(\boldsymbol{\mu}_{\text{physics}}\big)\Bigg\|_2^2\Bigg\}\end{aligned} \tag{25}$$

where $\mathcal{L}_{\text{Physics}}\Big(\widetilde{\boldsymbol{\mu}}_{\text{physics}}\Big)$ denotes the loss function of the physics-guided neural network, and N_{physics} is the number of parameter points $\widetilde{\boldsymbol{\mu}}_{\text{physics}}$ used for physics-guided training. Substituting Eq. (21-24) into Eq. (25), the physics-guided training process can be re-expressed as follows:

$$\begin{aligned}\underset{\boldsymbol{W},\boldsymbol{b}}{\arg\min}\ \ \mathcal{L}_{\text{Physics}}\Big(\widetilde{\boldsymbol{\mu}}_{\text{physics}}\Big) = \frac{1}{N_{\text{physics}}}\sum_{\widetilde{\boldsymbol{\mu}}_{\text{physics}}}\Bigg\{&\Bigg\|\mathbf{\Phi}^{\mathrm{T}}\Big[\mathcal{PDE}\Big(\overline{\mathbf{u}}, \mathbf{\Phi}, \mathcal{G}_t\Big(\overline{\mathbf{u}}\Big), \mathcal{G}_t\Big(\mathbf{\Phi}\Big), \mathcal{G}_{\mathbf{x}}\Big(\overline{\mathbf{u}}\Big), \mathcal{G}_{\mathbf{x}}\Big(\mathbf{\Phi}\Big), \mathcal{G}_{\mathbf{xx}}\Big(\overline{\mathbf{u}}\Big), \mathcal{G}_{\mathbf{xx}}\Big(\mathbf{\Phi}\Big);\ \ \mathcal{FNN}\Big(\widetilde{\boldsymbol{\mu}}_{\text{physics}}; \boldsymbol{W}, \boldsymbol{b}\Big)\Big)\\ &-\mathcal{S}\Big(\mathcal{H}(\widetilde{\boldsymbol{\mu}}_{\text{physics}})\Big)\Big]\Bigg\|_2^2\\ &+\Bigg\|\mathcal{B}\Big(\overline{\mathbf{u}}, \mathbf{\Phi};\ \mathcal{FNN}(\widetilde{\boldsymbol{\mu}}_{\text{physics}}; \boldsymbol{W}, \boldsymbol{b})\Big) - \mathbf{u}_{\mathrm{b}}\Big(\mathcal{H}(\widetilde{\boldsymbol{\mu}}_{\text{physics}})\Big)\Bigg\|_2^2\\ &+\Bigg\|\mathcal{I}\Big(\overline{\mathbf{u}}, \mathbf{\Phi};\ \mathcal{FNN}(\widetilde{\boldsymbol{\mu}}_{\text{physics}}; \boldsymbol{W}, \boldsymbol{b})\Big) - \mathbf{u}_0\Big(\mathcal{H}(\widetilde{\boldsymbol{\mu}}_{\text{physics}})\Big)\Bigg\|_2^2\Bigg\}\end{aligned} \tag{26}$$

where $\mathcal{H}(\ \cdot\ )$ denotes the denormalisation operation, given by

$$\boldsymbol{\mu} = \mathcal{H}\Big(\widetilde{\boldsymbol{\mu}}\Big) = \frac{\widetilde{\boldsymbol{\mu}}(\boldsymbol{\mu}_{\max} - \boldsymbol{\mu}_{\min}) + (\boldsymbol{\mu}_{\max} + \boldsymbol{\mu}_{\min})}{2} \tag{27}$$

Obviously, the parameter points $\widetilde{\boldsymbol{\mu}}_{\text{physics}}$ are the only inputs to train the physics-guided neural network, and such label-free ROM method can greatly alleviate the shortage of labelled training data ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\alpha}_{\text{data}}$).

It must be emphasised that the proposed physics-guided neural network is significantly different from the physics-informed neural network (PINN) in terms of derivative computation. In PINN [25, 29], the derivatives of the network outputs with respect to the network inputs are computed by using automatic differentiation. During the training process of PINN, derivatives of different orders are required to be repeatedly computed for all calibration points at each iteration step. Derivative computation is the most computationally intensive module to train PINN, and it is the main reason why training a PINN model is much more time-consuming than fitting a data-driven model.

In contrast, the proposed physics-guided neural network for ROM uses a high-efficient way to compute derivatives. As explained in Eq. (22-24), the partial derivative terms in the governing PDEs can be rapidly obtained through simple linear combinations of precomputed gradient results. Compared to PINN, the proposed physics-guided neural network is much computationally cheaper. Essentially, physics-guided training optimises the weight matrixes $\boldsymbol{W}$ and the bias vector $\boldsymbol{b}$ of a neural network by minimising the "discrepancy" between the learned input-output mapping and the intrinsic $\boldsymbol{\mu}$-$\boldsymbol{\alpha}$ relation characterised by the reduced-order numerical system. This physics-guided ROM framework is completely label-free, which is of great significance for common situations when high-fidelity snapshots are very limited.

3.5. Physics-data combined ROM

Obviously, neither the data-driven ROM nor the physics-guided ROM can simultaneously make full use of both the known physical principle and the labelled training data. It is of great necessity to develop a physics-data combined method that can seamlessly integrate both the physical principle and the labelled data into the feedforward neural network, thereby achieving model order reduction more cost-effectively. On the one hand, the physical principle can compensate for the shortage of labelled data; on the other hand, the labelled training data can accelerate the convergence of physics-guided training. Ideally, this physics-data combined ROM method is expected to inherit the comparative advantages of both the data-driven ROM and the physics-guided ROM methods.

In essence, it is a multi-objective optimization problem to jointly embed the reduced-order numerical system and the labelled training data into a feedforward neural network. The data objective and the physics objective can be significantly different in magnitude and temporarily conflicting during the optimization process. Here, a step-by-step training scheme is specially developed for this multi-objective optimization problem, during which a dynamic weighting factor is used to balance the data objective and the physics objective:

- **Step 1** is a purely data-driven training process, with the goal of establishing a preliminary mapping by using the limited amount of labelled training data ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$). The data-driven loss function $\mathcal{L}_{\text{Data}}\left(\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widetilde{\boldsymbol{\alpha}}_{\text{data}}\right)$ has been explained in Eq. (19), and it is often a simple optimization problem to embed the small data pairs ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) into the neural network by minimising the data-driven loss function. However, the obtained FNN model usually possesses a poor generalisation capacity, and it is thus required to be further enhanced in terms of interpolation and extrapolation performances.

- **Step 2** is aiming to enhance the *interpolation* performance of the preliminary FNN model obtained in Step 1 by embedding the parametrized reduced-order numerical system into it. A set of normalised parameter points $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}$ are selected within the scope of $\widetilde{\boldsymbol{\mu}}_{\text{data}}$, and they are then used to compute the physics-guided loss function $\mathcal{L}_{\text{physics}}\left(\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}\right)$ of the FNN model, as defined in Eq. (26). During the training process, the evolution of this FNN model is primarily dominated by the physics-guided loss function $\mathcal{L}_{\text{physics}}\left(\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}\right)$. But meanwhile, the input-output relation ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$-$\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ mapping) learned in Step 1 should be maintained by adding a data-driven loss term $\mathcal{L}_{\text{Data}}\left(\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widetilde{\boldsymbol{\alpha}}_{\text{data}}\right)$ as the data constraint. This constrained optimisation problem is mathematically expressed as follows:

$$\underset{\boldsymbol{W},\boldsymbol{b}}{\arg\min}\ \ \mathcal{L}^{(\text{inter})}_{\text{Combined}} = \mathcal{L}_{\text{Data}}\left(\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widetilde{\boldsymbol{\alpha}}_{\text{data}}\right) + \omega_{\text{inter}}\mathcal{L}_{\text{Physics}}\left(\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}\right) \tag{28}$$

  where $\mathcal{L}^{(\text{inter})}_{\text{Combined}}$ is named as the physics-data combined loss function, and ω_{inter} is a dynamic weight coefficient to adjust the physics-data proportion in the loss function $\mathcal{L}^{(\text{inter})}_{\text{Combined}}$, which is a step function defined in Eq. (30). An enhanced FNN model will be obtained after this training stage, from which $\widehat{\widetilde{\boldsymbol{\alpha}}}_{\text{data}}$ and $\widehat{\widetilde{\boldsymbol{\alpha}}}^{(\text{inter})}_{\text{physics}}$ can be estimated by inputting $\widetilde{\boldsymbol{\mu}}_{\text{data}}$ and $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}$ respectively. This enhanced FNN model together with the newly obtained data pairs $\left(\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widehat{\widetilde{\boldsymbol{\alpha}}}_{\text{data}}\right)$ and $\left(\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}, \widehat{\widetilde{\boldsymbol{\alpha}}}^{(\text{inter})}_{\text{physics}}\right)$ are transferred to Step 3 for further improvement.

- **Step 3** is devoted to reinforcing the *extrapolation* capacity of the FNN model by solving the following optimization problem:

$$\underset{\boldsymbol{W},\boldsymbol{b}}{\arg\min}\ \ \mathcal{L}^{(\text{extra})}_{\text{Combined}} = \mathcal{L}_{\text{Data}}\left(\left[\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}\right], \left[\widehat{\widetilde{\boldsymbol{\alpha}}}_{\text{data}}, \widehat{\widetilde{\boldsymbol{\alpha}}}^{(\text{inter})}_{\text{physics}}\right]\right) + \omega_{\text{extra}}\mathcal{L}_{\text{Physics}}\left(\widetilde{\boldsymbol{\mu}}^{(\text{extra})}_{\text{physics}}\right) \tag{29}$$

  where $\mathcal{L}^{(\text{extra})}_{\text{Combined}}$ is a new physics-data combined loss function, and ω_{extra} is a new weighting function to balance the physics objective and the data constraint, as defined in Eq. (30). The data-driven loss term

$\mathcal{L}_{\text{Data}}\Big(\big[\widetilde{\boldsymbol{\mu}}_{\text{data}}, \widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}\big], \big[\widehat{\widetilde{\boldsymbol{\alpha}}}_{\text{data}}, \widehat{\widetilde{\boldsymbol{\alpha}}}^{(\text{inter})}_{\text{physics}}\big]\Big)$ plays the role of data constraint to maintain the input-output relation learned in Step 1 and Step 2, as defined in Eq. (19). The physics-guided loss term $\mathcal{L}_{\text{physics}}\big(\widetilde{\boldsymbol{\mu}}^{(\text{extra})}_{\text{physics}}\big)$ defined in Eq. (26) is the primary optimization objective, and it is computed for a set of normalised parameter points $\widetilde{\boldsymbol{\mu}}^{(\text{extra})}_{\text{physics}}$ that are beyond the scope of $\widetilde{\boldsymbol{\mu}}_{\text{data}}$, in order to broaden the learning/mapping scope. After this step, a generalised surrogate model with a wide learning scope can be expected. Besides, the effective parameter range of this FNN model can be further broadened through more steps of physics-data combined training, as long as the basis functions $\boldsymbol{\Phi}$ are still applicable.

The dynamic weight coefficients ω_{inter} and ω_{extra} in Eq. (28) and (29) vary with the epoch number, in order to dynamically adjust the physics-data proportion in the total loss functions. The value of ω_{inter} or ω_{extra} should be sufficiently large to ensure that the training procedure is dominated by the physics-guided loss term. On the other side, the weight coefficient ω_{inter} or ω_{extra} cannot be too large because the data-driven loss term needs to play the role of optimization constraint, so as to preserve the learned input-output relation in the previous step(s). Here, a step function is used to define the dynamic weight coefficient as a function of the epoch number, given by:

$$\omega_{\text{dynamic}}(k) = 10^{\text{int}\left(\frac{k-K_{\text{s}}}{L_{\text{e}}}\right)+c} \tag{30}$$

where k is the current epoch number during the training process, $\text{int}(\cdot)$ is a function that rounds a number down to the nearest integer, K_{s} is the starting epoch number of Step 2 or Step 3, L_{e} denotes the epoch length during which the value of $\omega_{\text{dynamic}}(k)$ is fixed, and c is a constant.

In summary, this physics-data combined ROM flexibly integrates the available data and the known physics into the neural network through a special step-by-step scheme. In the beginning step, purely data-driven training is conducted to rapidly fit the neural network by using the small labelled data. In the following steps, the reduced-order numerical system is embedded into the neural network to compensate for data sparsity, thereby improving the model accuracy and generalisation capacity. The data-driven loss terms in Step 2 and Step 3 are used to maintain the learned input-output relation in the previous step(s), while the physics-guided loss terms are used to enhance/widen this input-output mapping in aspects of both interpolation and extrapolation. Besides, the final obtained FNN model can still be sustainably evolved through further training steps, and the POD basis $\boldsymbol{\Phi}$ can also be adaptively changed at different stages of the training process.

4. Results and comparison

In this section, four types of model order reduction methods (including the data-driven ROM, the physics-guided ROM the physics-data combined ROM and the traditional projection-based ROM methods) are applied to a series of nonlinear problems. For all these test cases, high-fidelity snapshot data is of limitation and only sufficient to yield a set of POD basis functions that are able to capture the primary dynamics of the studied systems. Based on the derived POD basis, four ROM methods are used to construct reduced-order models to approximately represent the parametric dependence. A comprehensive comparison between the ROM results is then carried out, in terms of modelling accuracy and construction efficiency. To quantitatively assess ROM quality, the relative L_2-norm error $\varepsilon(\boldsymbol{\mu})$ can be calculated to measure the discrepancy between a reduced-order solution $\mathbf{u}_{\text{r}}(\cdot\,,\cdot\,;\boldsymbol{\mu})$ and the high-fidelity numerical (analytical) solution $\mathbf{u}(\cdot\,,\cdot\,;\boldsymbol{\mu})$:

$$\varepsilon(\boldsymbol{\mu}) = \frac{\left\|\mathbf{u}(\cdot\,,\cdot\,;\boldsymbol{\mu}) - \mathbf{u}_{\text{r}}(\cdot\,,\cdot\,;\boldsymbol{\mu})\right\|_2}{\left\|\mathbf{u}(\cdot\,,\cdot\,;\boldsymbol{\mu})\right\|_2} \tag{31}$$

Besides, it should be mentioned that Adam optimiser [42] is adopted to train the feedforward neural networks in this study. As a frequently-used stochastic gradient descent algorithm, Adam optimization is based on the adaptive estimations of the first-order and second-order moments. The purpose of the training process is to let the neural network evolve into its optimal state that best represents the input-output relation. More details about Adam optimiser can be found in the relevant reference [42].

4.1. Case 1: 1D Burgers' equation

As a nonlinear PDE, one-dimensional (1D) parameterized Burgers' equation is widely used as a benchmark for validation purposes in various studies [4, 29]. Here, it is used to validate the proposed ROM methods, and it is mathematically expressed as:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \mu\frac{\partial^2 u}{\partial x^2} = 0 \tag{32}$$

Table 1: The hyper-parameters to train feedforward neural networks for reduced-order modelling of nonlinear dynamical systems

Test case	ROM method	Model hyper-parameters		Algorithm hyper-parameters					
								Dynamic weight coefficients	
		Hidden layers	Neurons	Mini-batch size	Maximum epoch number	Learning rate	Regulation constant γ	ω_{inter}	ω_{extra}
Case 1	'Data'	3	60	18	10,000	$\frac{0.04}{1+0.0025k}$	10^{-5}	$--$	$--$
Case 1	'Physics'	3	60	64	60,000	$\frac{0.05}{1+0.005k}$	$--$	$--$	$--$
Case 1	'Combined'	3	60	64	90,000	$\frac{0.05}{1+0.005k}$	10^{-5}	$10^{\text{int}(\frac{k-5000}{10000})+2}$ (2 steps)	$10^{\text{int}(\frac{k-25000}{12500})}$ (4 steps)
Case 2	'Data'	3	50	16	10,000	$\frac{0.1}{1+0.005k}$	10^{-5}	$--$	$--$
Case 2	'Physics'	3	50	34	50,000	$\frac{0.1}{1+0.005k}$	$--$	$--$	$--$
Case 2	'Combined'	3	50	64	60,000	$\frac{0.05}{1+0.005k}$	10^{-5}	$10^{\text{int}(\frac{k-5000}{7500})+1}$ (2 steps)	$10^{\text{int}(\frac{k-20000}{10000})+1}$ (3 steps)
Case 3	'Data'	3	50	32	10,000	$\frac{0.05}{1+0.005k}$	10^{-5}	$--$	$--$
Case 3	'Physics'	3	50	64	80,000	$\frac{0.1}{1+0.002k}$	$--$	$--$	$--$
Case 3	'Combined'	3	50	64	160,000	$\frac{0.1}{1+0.003k}$	10^{-5}	$10^{\text{int}(\frac{k-2000}{20000})+1}$ (4 steps)	$10^{\text{int}(\frac{k-120000}{10000})+3}$ (2 steps)
Case 4	'Data'	3	80	32	10,000	$\frac{0.02}{1+0.005k}$	10^{-5}	$--$	$--$
Case 4	'Physics'	3	80	64	50,000	$\frac{0.05}{1+0.001k}$	$--$	$--$	$--$
Case 4	'Combined'	3	80	64	230,000	$\frac{0.02}{1+0.005k}$	10^{-5}	$10^{\text{int}(\frac{k-5000}{40000})+1}$ (3 steps)	$10^{\text{int}(\frac{k-140000}{30000})+3}$ (3 steps)

Note: 'Data', 'Physics' and 'Combined' denote the data-driven ROM, physics-guided ROM and physics-data combined ROM methods respectively; k denotes the current epoch number during the training processes.

where $u(x, t)$ is the flow velocity, and μ denotes the kinematic viscosity of the fluid. The computational space-time domain is $x \in \mathcal{X} : [-1, 1] \times t \in \mathcal{T} : [0, 1]$. The initial condition is:

$$u(x, 0) = -\sin\pi x \tag{33}$$

and the boundary conditions are:

$$u(\pm 1, t) = 0 \tag{34}$$

Kinematic viscosity μ is the dimensionless parameter of interest that affects the solution of the 1D Burgers' equation. In this study, the value of μ uniformly varies over a predefined parameter space $\mathcal{P} : \left[\frac{1}{100\pi}, \frac{750}{100\pi}\right]$ with a interval of $\frac{1}{100\pi}$. The entire parameter space $\mathcal{P}$ containing 750 parameter points is divided into two sub-domains, which are $\mathcal{P}_{\text{inter}} : \left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ and $\mathcal{P}_{\text{extra}} : \left[\frac{501}{100\pi}, \frac{750}{100\pi}\right]$. The Smolyak sparse grid method [1, 2, 43] is applied to select parameter points from $\mathcal{P}$. The key idea of the Smolyak sparse grid is to select a small number of nodes on the full tensor-product grid according to the potential importance of the nodes, thereby greatly improving the computational efficiency. More information about Smolyak sparse grid is provided in Appendix II.

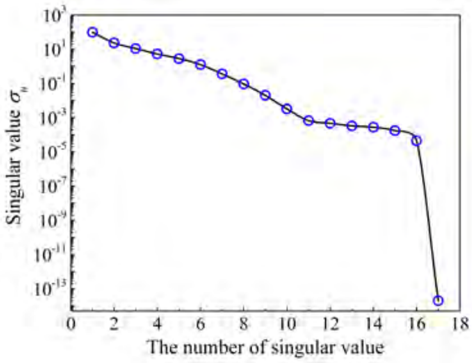
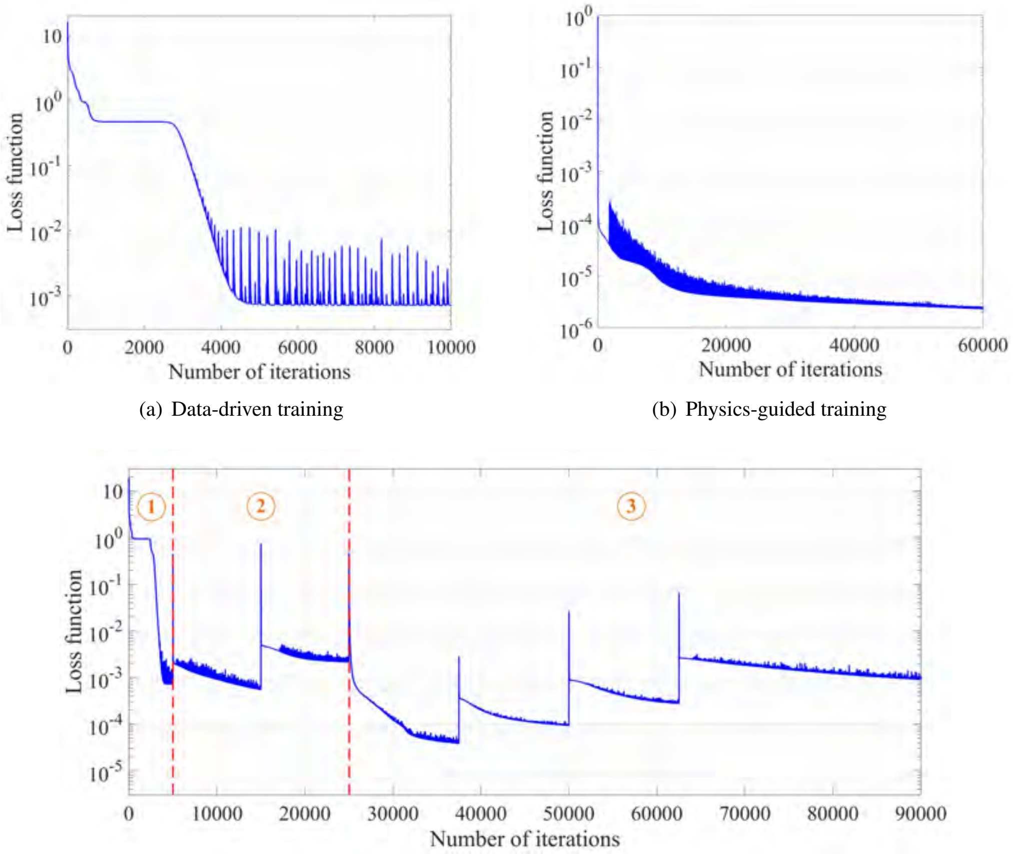


Figure 4: The singular values σ_u plotted in descending order for the physical variable u.

A set of 17 parameter points are selected from $\mathcal{P}_{\text{inter}} : \left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 4, which are denoted by $\boldsymbol{\mu}_{\text{data}}$ here. Finite Difference Method (FDM) [44] is then adopted to numerically approximate the 1D Burger's equation on a regular mesh grid 151×101, from which a set of 17 high-fidelity snapshots corresponding to the selected parameters are obtained. Performing singular value decomposition on the snapshot data according to Eq. (5), a set of singular values can be computed, as plotted in Figure 4. According to the criterion defined by Eq. (8), the first 12 left-singular vectors can be chosen as the POD basis functions because the value of β is sufficiently large ($> 99.999\%$). A great value of β means that the primary dynamical characteristics of the 1D Burger's equation can be approximately captured by these 12 POD modes. Besides, the POD expansion coefficients $\boldsymbol{\alpha}_{\text{data}}$ can also be computed by projecting the set of snapshots onto the reduced space, as explained in Eq. (10).



(a) Data-driven training

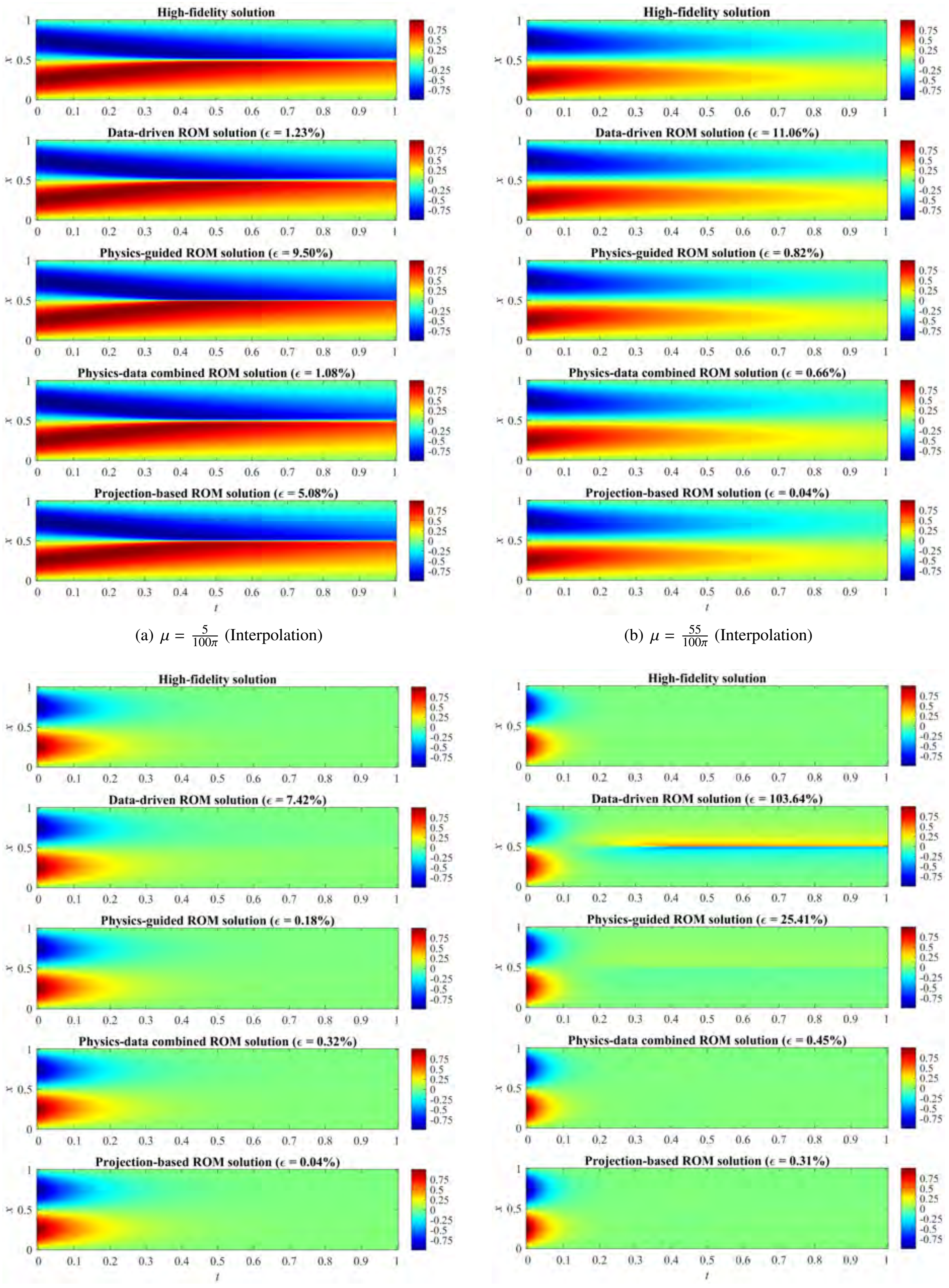
(b) Physics-guided training

(c) Physics-data combined training

Figure 5: The convergence curves of the neural networks' loss functions using (a) Data-driven ROM method, (b) Physics-guided ROM method and (c) Physics-data combined ROM method.

A set of data pairs ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) can be obtained by performing data normalisation on the raw data pairs ($\boldsymbol{\mu}_{\text{data}}$, $\boldsymbol{\alpha}_{\text{data}}$), and they are the only available labelled training data to approximate the $\widetilde{\mu}$-$\widetilde{\alpha}$ mapping relation via the data-driven ROM. To compare the data-driven ROM, physics-guided ROM and physics-data combined ROM methods, all these three methods are performed on the feedforward neural networks with the same architecture, and the corresponding hyper-parameters are listed in Table 1. The convergence curve of the data-driven training is plotted in Figure 5a, from which one can see that the neural network evolves into a stable state very fast. This data-driven FNN model can then be used to predict POD expansion coefficients for unseen parameters, based on which reduced-order solutions of the physical field can be rapidly approximated according to Eq. (9). As shown in Figure 6, the reduced-order solutions corresponding to four representative parameters are estimated from the data-driven FNN model.

To quantitatively assess the prediction accuracy of this FNN model, the relative L_2-norm errors of the reduced-order solutions corresponding to all 750 parameter points within $\mathcal{P}$ are computed, as plotted in Figure 7a. Generally, the data-driven FNN model shows poor performances in terms of both interpolation and extrapolation. The largest and the average interpolation errors are 12.44% and 3.04% respectively, which may imply that the labelled training data is too sparse to yield a reliable model via data-driven training. As to extrapolation results, the relative



(a) $\mu = \frac{5}{100\pi}$ (Interpolation)

(b) $\mu = \frac{55}{100\pi}$ (Interpolation)

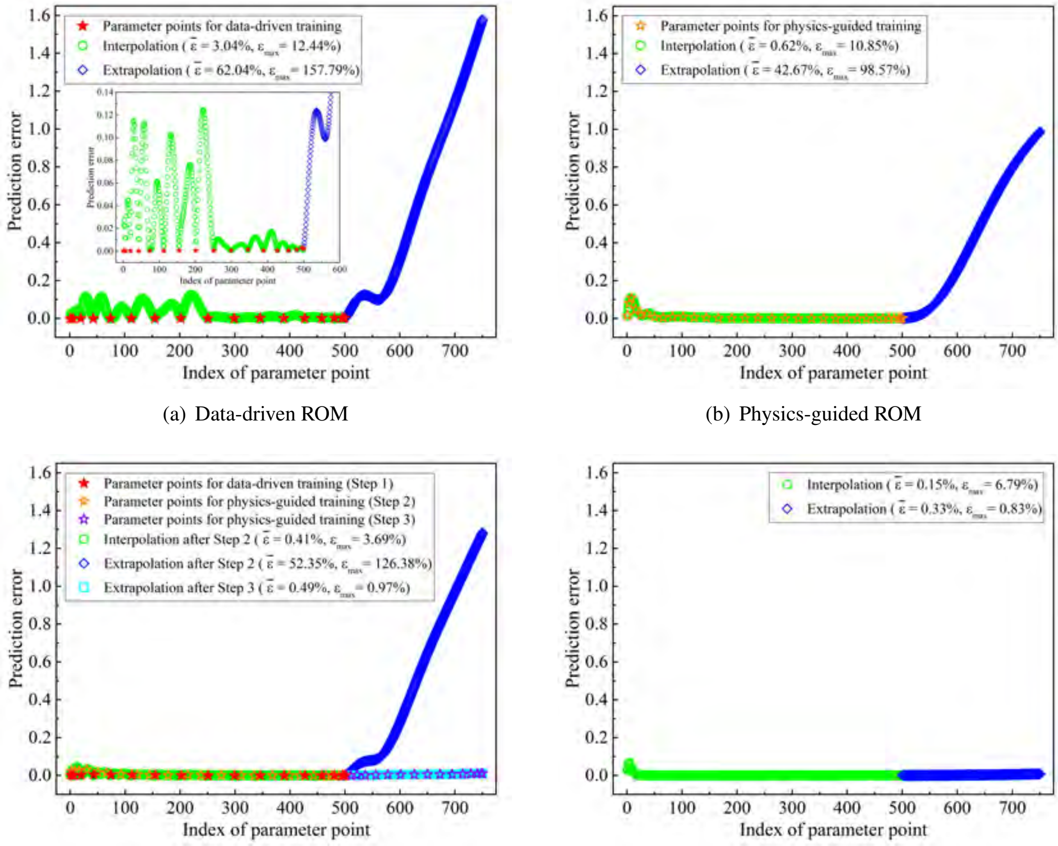
(c) $\mu = \frac{225}{100\pi}$ (Interpolation)

(d) $\mu = \frac{600}{100\pi}$ (Extrapolation)

Figure 6: Comparisons between the high-fidelity numerical solution and the reduced-order solutions of the physical field u predicted from the data-driven FNN, the physics-guided FNN, physics-data combined FNN, and the intrusive projection-based models for different parameters μ (the numbers in parentheses are the relative L_2-norm errors ε).

L_2-norm error increases rapidly as the parameter point becomes far away from the scope of training data. The average extrapolation error is up to 157.79%, indicating that the data-driven FNN model has poor extrapolation performance. In summary, purely data-driven ROM is unable to accurately represent the intrinsic $\widetilde{\mu}$-$\widetilde{\alpha}$ relation and to yield a reliable reduced-order model in the small-data regime.

To compensate for the shortage of labelled training data, the physical principle is involved in the loss function of the neural network for physics-guided training. A set of 63 parameter points are selected from $\mathcal{P}_{\text{inter}}$: $\left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 6, which are denoted by $\boldsymbol{\mu}^{(\text{inter})}_{\text{physics}}$ here. It should be noted that the parameter set $\boldsymbol{\mu}_{\text{data}}$ is completely contained in $\boldsymbol{\mu}^{(\text{inter})}_{\text{physics}}$ (mathematically expressed as $\boldsymbol{\mu}_{\text{data}} \subset \boldsymbol{\mu}^{(\text{inter})}_{\text{physics}}$), because they are chosen by using the same data sampling method. After data normalisation, the parameter points in $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}$ are used for physics-guided training, as explained in Eq. (26). Central finite differences with 4-order accuracy (as explained in Appendix I) are used to approximate the partial derivatives in Burgers' equation, based on which the physics-guided loss function can be computed for these selected parameter points. The convergence curve of the physics-guided loss function is plotted in Figure 5b, from which one can see that it converges much slower than the data-driven loss function. The computation cost of each iteration in physics-guided training is also higher than that of data-driven training.



(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Traditional projection-based ROM

Figure 7: The relative L_2-norm errors ε of the reduced-order solutions estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model, and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε over a specific parameter range).

Once the physics-guided FNN model is properly trained, reduced-order solutions of the physical field responding to varying parameters can be estimated from it, as shown in Figure 6. The relative L_2-norm errors of the reduced-order solutions corresponding to all parameter points in $\mathcal{P}$ are plotted in Figure 7b. Clearly, the physics-guided FNN model exhibits a much better performance in terms of interpolation, compared to the data-driven FNN model. The average interpolation error is 0.62%, but the maximum interpolation error is remarkable (up to 10.85%) when $\mu \leq \frac{5}{100\pi}$, which can be seen in Figure 7b. The physical field corresponding to a small value of

μ usually contains unsmooth/sharp regions with steep gradients, as shown in Figure 6a. Computing gradients by using finite difference in these unsmooth/sharp areas can bring errors, and such estimation errors can cause the physics-guided loss function to converge to a sub-optimal state. Besides, the physics-guided FNN model also has a poor extrapolation capacity, especially when parameter points go far away from the learning scope, as illustrated in Figure 7b.

To take full advantage of both the labelled data and the physical principle, the physics-data combined ROM is applied to this test case. The 1st step of the physics-data combined ROM is a purely data-driven training process by using the normalised data pairs ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) to yield a preliminary FNN model. Step 2 is to enhance the model interpolation performance by using a physics-data combined loss function, where the training process is mainly controlled by the physics-guided loss term but constrained by the $\widetilde{\boldsymbol{\mu}}_{\text{data}}$-$\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ relation learned in Step 1. The physics-guided loss term here is computed for the complementary set of $\widetilde{\boldsymbol{\mu}}_{\text{data}}$ in $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}$, which is mathematically denoted by $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}} - \widetilde{\boldsymbol{\mu}}_{\text{data}}$. To reinforce the model extrapolation capacity in the 3rd step, another set of 33 parameter points are selected from $\mathcal{P}_{\text{extra}} : \left[\frac{501}{100\pi}, \frac{750}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 5, and they are denoted by $\boldsymbol{\mu}^{(\text{extra})}_{\text{physics}}$ here. After data normalisation, $\widetilde{\boldsymbol{\mu}}^{(\text{extra})}_{\text{physics}}$ is obtained to calculate the physics-guided loss. The training objective is to minimise this physic-guided loss by preserving the input-output relations learned in Step 1 and Step 2.

Following the procedures explained in Section 3.5, a surrogate model is obtained by integrating the labelled training data and the reduced-order numerical system into the neural network. The converge curve of the physics-combined loss function over the entire training process is plotted in Figure 5c. Because a step function is used as the dynamic weight coefficient (as defined in Eq. (30)), the convergence curve of the loss function fluctuates. The relative L_2-norm errors of the reduced-order solutions corresponding to all parameter points are recorded in Figure 7c. Compared to the data-driven model, this physics-data combined model exhibits a much better interpolation performance after the training process in Step 2. Compared to the physics-guided model, the training process containing Step 1 and Step 2 is also much quicker, and the average interpolation error is reduced from 0.62% to 0.41%. Moreover, the physics-data combined model can still work well when μ is small (where the physical fields contain sharp regions), as can be observed in Figure 6a. After completing the training process in Step 3, the extrapolation capacity of the physics-data combined model is greatly enhanced as well, with an average extrapolation error of 0.49%. This could be the overwhelming superiority of the physics-data combined ROM method over the purely data-driven ROM and physics-guided ROM methods.

To provide a reliable reference, a projection-based model is constructed through the traditional intrusive ROM method [10, 11], where the orthogonal basis with 12 POD modes is used. Here, the projection-based model is built by following the procedure explained in [10], and the corresponding open-source code for parametrized ROM is used. As can be seen in Figure 7d, the projection-based model exhibits excellent performance in terms of both interpolation and extrapolation predictions. Similar to the physics-guided model, this projection-based model also has relatively larger errors (up to 6.79%) for small values of μ, and such a problem can be well solved by using another set of properly-selected POD basis functions. Overall, the physics-data combined model is comparable to the projection-based model in predicting accurate solutions for the entire parameter range, which further confirms the effectiveness of the proposed non-intrusive ROM method.

4.2. Case 2: 2D Burgers' equation

The two-dimensional (2D) Burgers' equations [45, 46] are fundamental PDEs to model various physical problems, such as fluid dynamics, heat conduction and acoustic transmission. To test the applicability of the proposed ROM method on 2D nonlinear problems with two physical variables, the unsteady 2D Burgers' equations are considered over a square spatial domain $x \in \mathcal{X} : [0, 1] \times y \in \mathcal{Y} : [0, 1]$, and they are mathematically described as follows:

$$\begin{cases} \dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} - \mu\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) = 0 \\ \dfrac{\partial v}{\partial t} + u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} - \mu\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right) = 0 \end{cases} \tag{35}$$

subject to the initial conditions:

$$\begin{cases} u(x, y, 0) = \dfrac{3}{4} - \dfrac{1}{4(1 + \mathrm{e}^{(y-x)/(8\mu)})} \\ v(x, y, 0) = \dfrac{3}{4} + \dfrac{1}{4(1 + \mathrm{e}^{(y-x)/(8\mu)})} \end{cases} \tag{36}$$

and the boundary conditions:

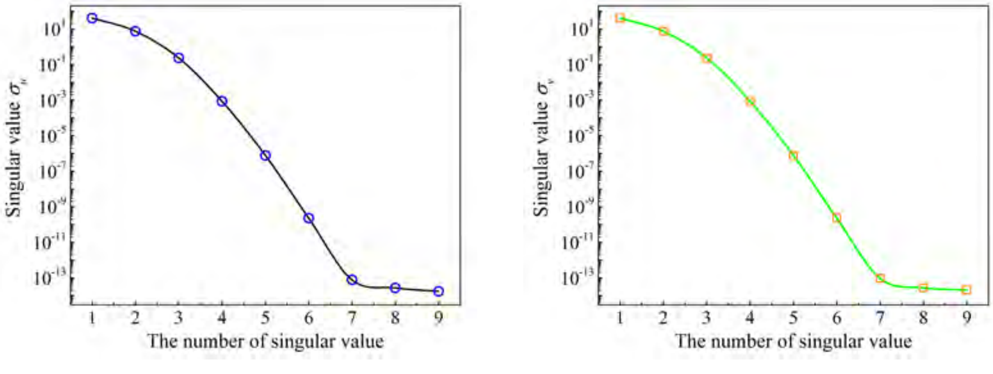
$$\begin{cases} u(0,y,t) = \dfrac{3}{4} - \dfrac{1}{4(1+\mathrm{e}^{(4y-t)/(32\mu)})}; & v(0,y,t) = \dfrac{3}{4} + \dfrac{1}{4(1+\mathrm{e}^{(4y-t)/(32\mu)})}; \\ u(1,y,t) = \dfrac{3}{4} - \dfrac{1}{4(1+\mathrm{e}^{(4y-4-t)/(32\mu)})}; & v(1,y,t) = \dfrac{3}{4} + \dfrac{1}{4(1+\mathrm{e}^{(4y-4-t)/(32\mu)})}; \\ u(x,0,t) = \dfrac{3}{4} - \dfrac{1}{4(1+\mathrm{e}^{(-4x-t)/(32\mu)})}; & v(x,0,t) = \dfrac{3}{4} + \dfrac{1}{4(1+\mathrm{e}^{(-4x-t)/(32\mu)})}; \\ u(x,1,t) = \dfrac{3}{4} - \dfrac{1}{4(1+\mathrm{e}^{(4-4x-t)/(32\mu)})}; & v(x,1,t) = \dfrac{3}{4} + \dfrac{1}{4(1+\mathrm{e}^{(4-4x-t)/(32\mu)})}; \end{cases} \tag{37}$$

where u and v are the velocity components along x-axis and y-axis respectively, and μ denotes the kinematic viscosity of the fluid.

The analytical solutions [45, 47] to the above 2D Burgers' Equations are given by:

$$\begin{cases} u(x,y,t) = \dfrac{3}{4} - \dfrac{1}{4(1+\mathrm{e}^{(4y-4x-t)/(32\mu)})} \\ v(x,y,t) = \dfrac{3}{4} + \dfrac{1}{4(1+\mathrm{e}^{(4y-4x-t)/(32\mu)})} \end{cases} \tag{38}$$

Kinematic viscosity μ is the varying parameter of interest that can lead to different solutions. Here, the value of parameter μ uniformly varies over a predefined space $\mathcal{P} : \left[\frac{1}{100\pi}, \frac{1000}{100\pi}\right]$ with the interval of $\frac{1}{100\pi}$. The entire parameter space $\mathcal{P}$ contains 1000 parameter points, and it is divided into two sub-domains which are denoted by $\mathcal{P}_{\text{inter}} : \left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ and $\mathcal{P}_{\text{extra}} : \left[\frac{501}{100\pi}, \frac{1000}{100\pi}\right]$ respectively.

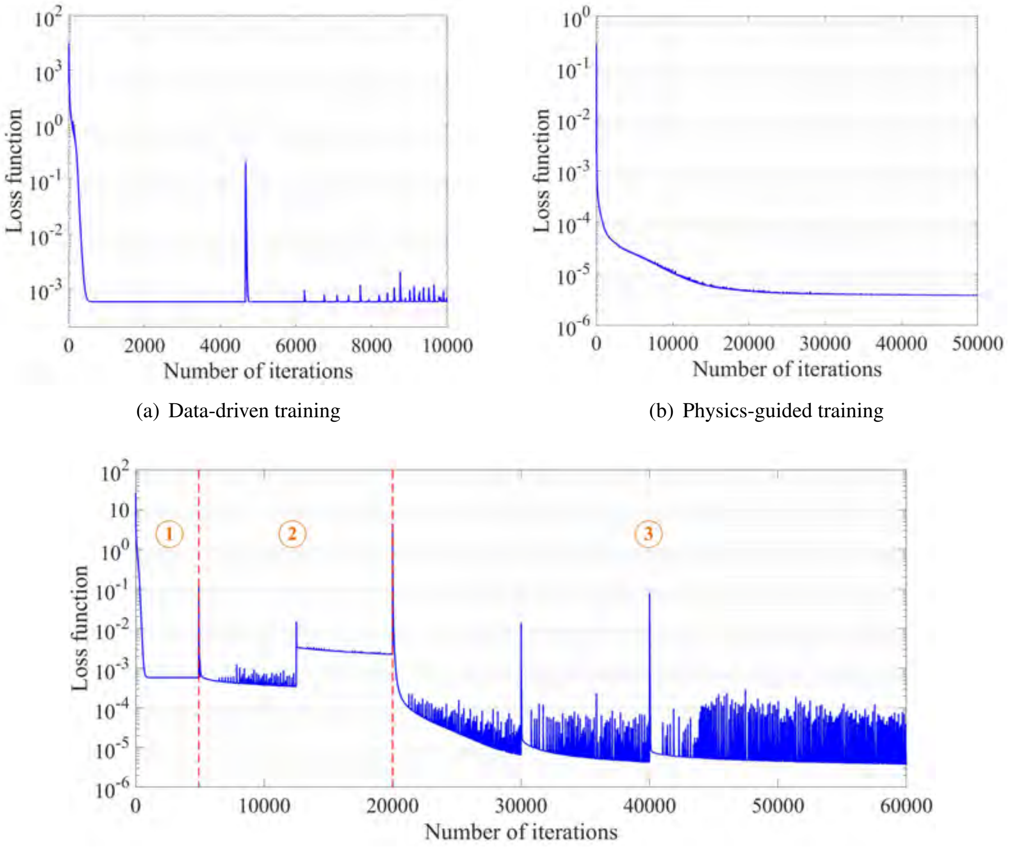


(a) σ_u

(b) σ_v

Figure 8: The singular values σ_u and σ_v plotted in descending orders for the physical variables u and v.

A set of 9 parameter points are selected from $\mathcal{P}_{\text{inter}} : \left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 3, which are denoted by μ_{data} here. The analytical solutions corresponding to μ_{data} are computed in a predefined time domain $t \in \mathcal{T} : [0, 1]$ by using Eq. (38). The analytical solutions are recorded on a uniform structured Cartesian grid 51 × 51 × 51, and then a set of 9 snapshots of u and v are thus obtained. Proper orthogonal decomposition is performed on the snapshot data according to Eq. (5), and two sets of singular values (σ_u and σ_v) are computed separately, as plotted in Figure 8. To capture as much information on high-fidelity solutions as possible, all 9 left-singular vectors are chosen as the POD modes for model reductions. Additionally, the POD expansion coefficients $\boldsymbol{\alpha}_{\text{data}}$ can be quickly computed by projecting the set of snapshots onto the reduced space, as explained in Eq. (10). The obtained data pairs ($\boldsymbol{\mu}_{\text{data}}, \boldsymbol{\alpha}_{\text{data}}$) are the only labelled training data for model order reduction via data-driven training.



(a) Data-driven training

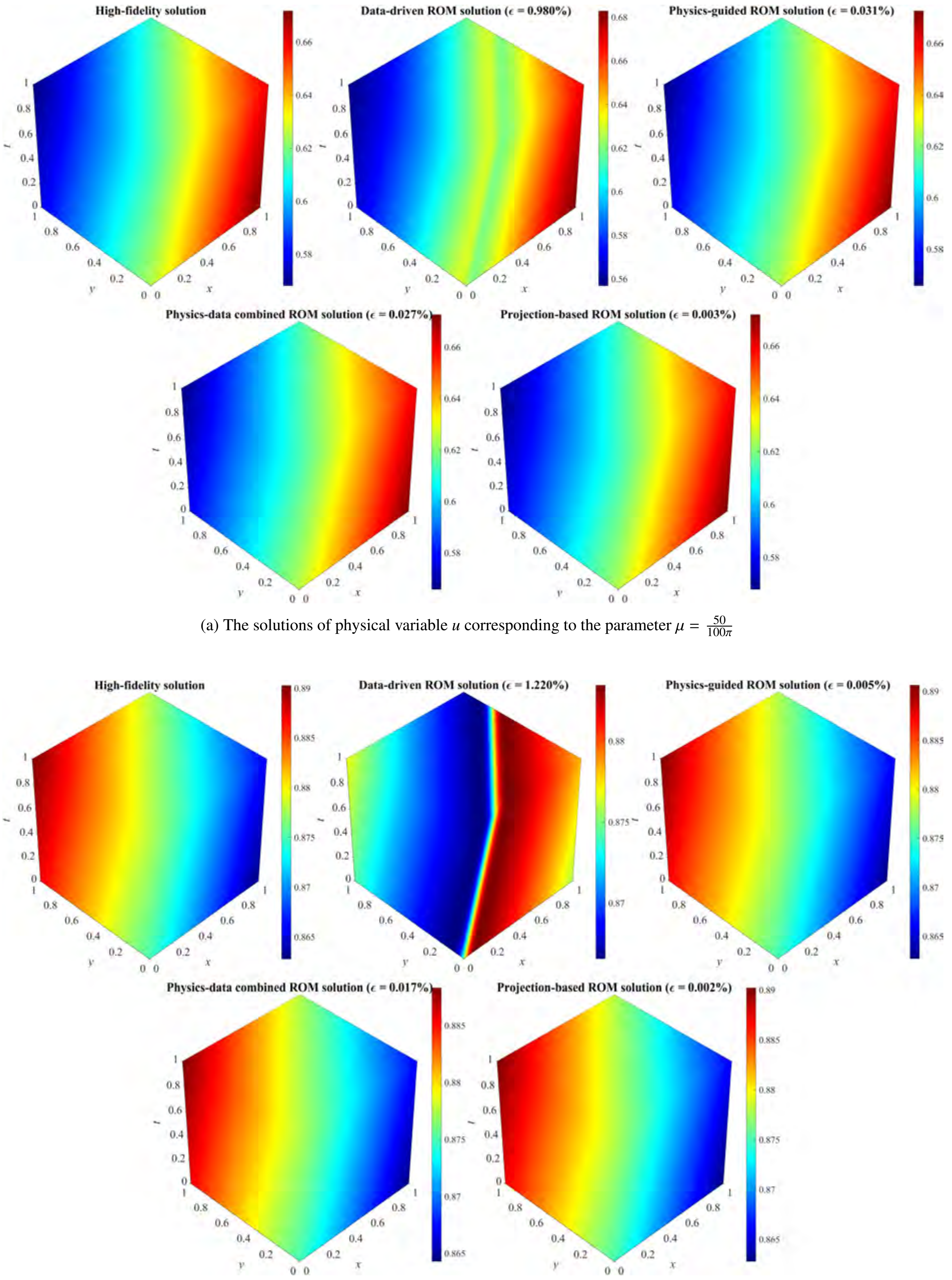
(b) Physics-guided training

(c) Physics-data combined training

Figure 9: The convergence curves of the neural networks' loss functions using (a) Data-driven ROM method, (b) Physics-guided ROM method and (c) Physics-data combined ROM method.

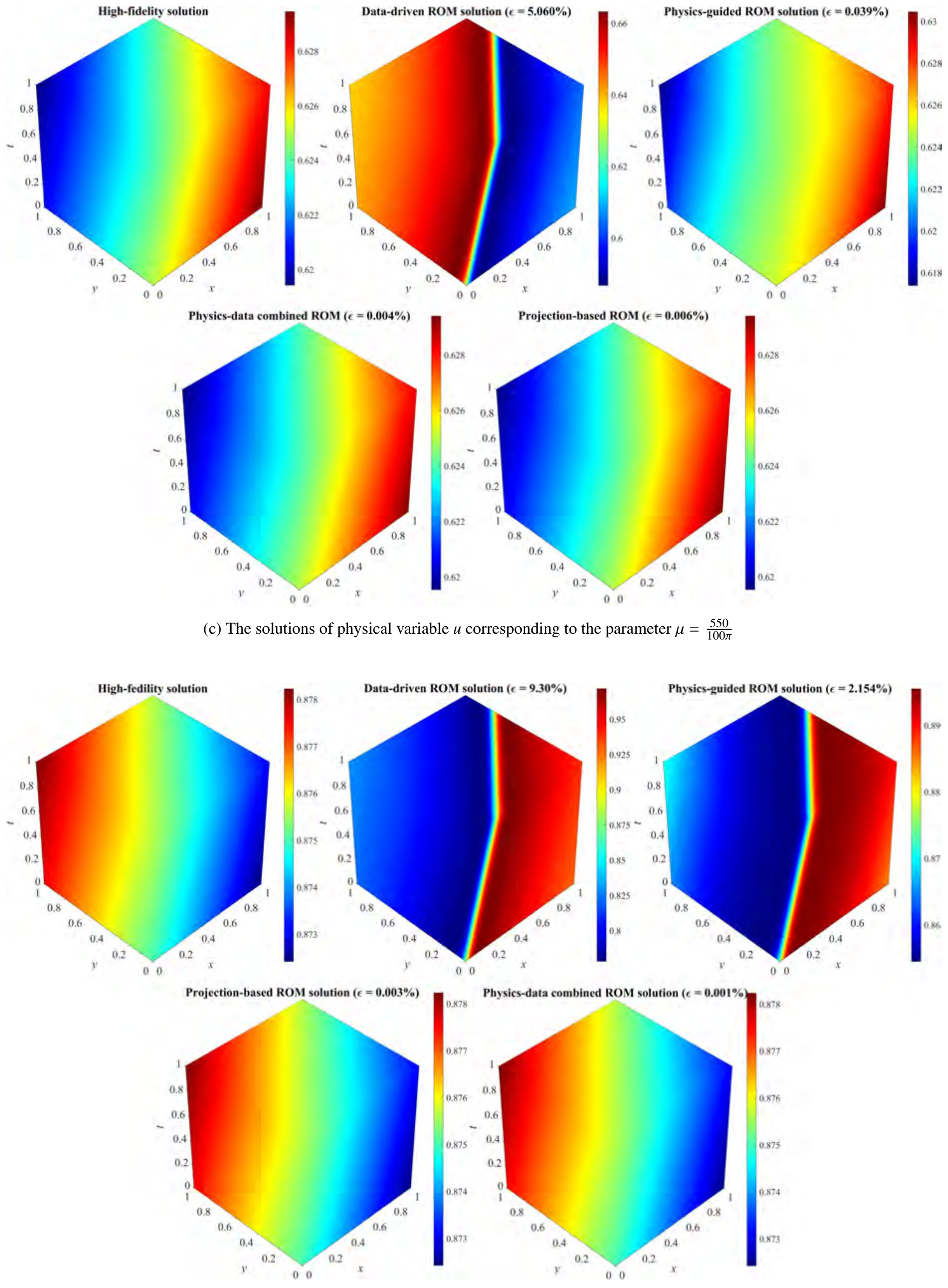
Performing data normalisation on the data pairs ($\boldsymbol{\mu}_{\text{data}}, \boldsymbol{\alpha}_{\text{data}}$) yields ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$), the latter is the only labelled training data to construct the $\widetilde{\mu}$-$\widetilde{\boldsymbol{\alpha}}$ linkage via the data-driven ROM. The hyper-parameters of the feedforward neural network are recorded in Table 1. Because the labelled training data ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) is small, the neural network rapidly converges to a stable state, which can be seen from the convergence curve in Figure 9a. Once the neural network has been properly fitted, reduced-order solutions corresponding to different parameters μ can be predicted from this data-driven model, and representative results are provided in Figure 10. Generally, this data-driven FNN model exhibits poor performance in both interpolation and extrapolation, which can be seen from the comparisons between the high-fidelity and reduced-order solutions. Relative L_2-norm error is used to assess the prediction results, as recorded in Figure 12a and b. The largest interpolation errors of u and v are 4.91% and 3.31% respectively, and the largest extrapolation errors of u and v are up to 13.04% and 9.30% respectively, which may imply that the data-driven ROM method is unable to yield reliable surrogate models for nonlinear problems in small-data regimes.

To overcome the lack of labelled data, the physics-guided ROM method is thus applied to this 2D nonlinear problem. A set of 33 parameter points are selected from $\mathcal{P}_{\text{inter}} : \left[\frac{1}{100\pi}, \frac{500}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 5, which are denoted by $\mu^{(\text{inter})}_{\text{physics}}$ here. After data normalisation, the parameter points in $\widetilde{\mu}^{(\text{inter})}_{\text{physics}}$ are the only input to compute the physics-guided loss function of the neural network, as defined in Eq. (26). Here, central finite differences are used to approximate partial derivatives to 6-order of accuracy, as explained in Appendix I. The convergence curve of the physics-guided loss function is plotted in Figure 9b, from which one can see that physics-guided training requires more iteration steps to reach the stable state compared to data-driven training. Once the neural network is properly trained, the reduced-order solutions of u and v corresponding to varying parameters can then be predicted from it, and representative results are provided in Figure 10. Besides, the surface plots of u and v corresponding to different parameters $\boldsymbol{\mu}$ and time points t are provided in Figure 11.



(a) The solutions of physical variable u corresponding to the parameter $\mu = \frac{50}{100\pi}$

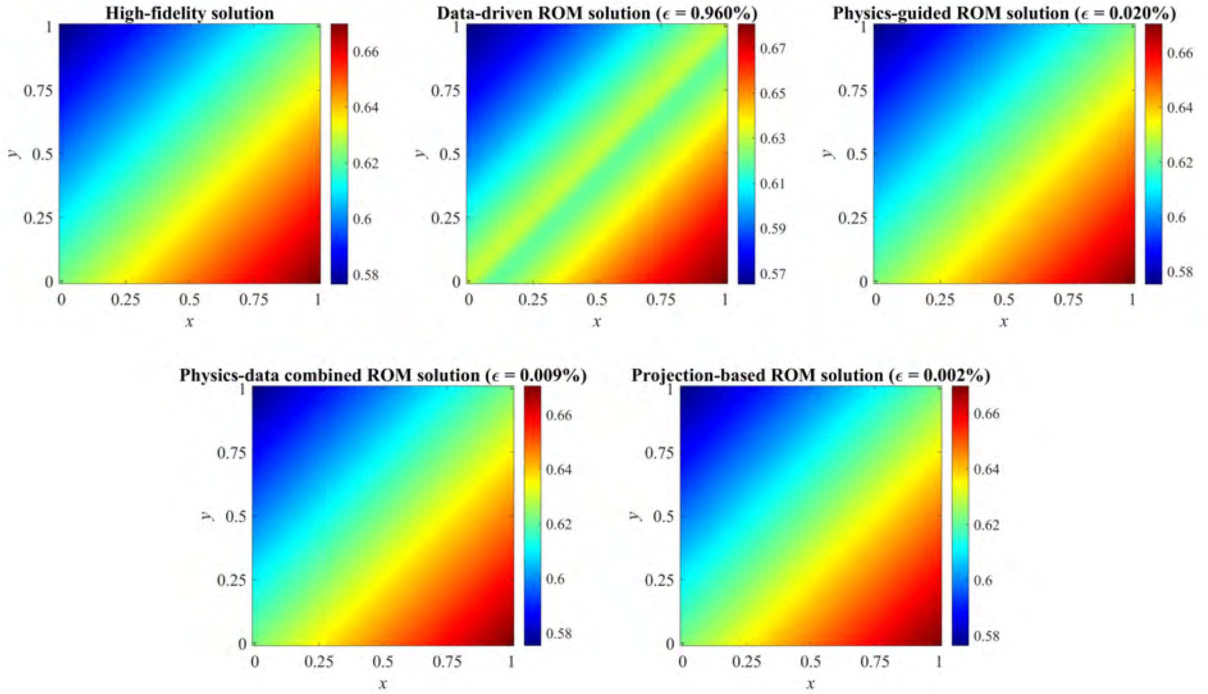
(b) The solutions of physical variable v corresponding to the parameter $\mu = \frac{200}{100\pi}$



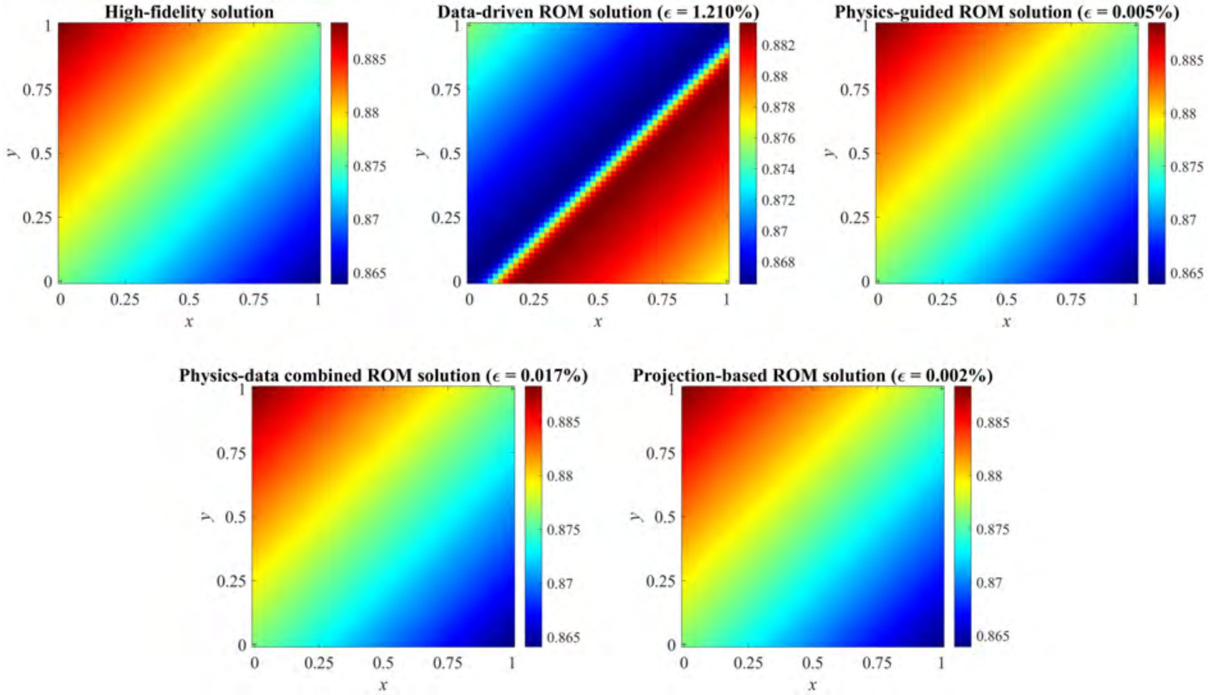
(c) The solutions of physical variable u corresponding to the parameter $\mu = \frac{550}{100\pi}$

(d) The solutions of physical variable v corresponding to the parameter $\mu = \frac{950}{100\pi}$

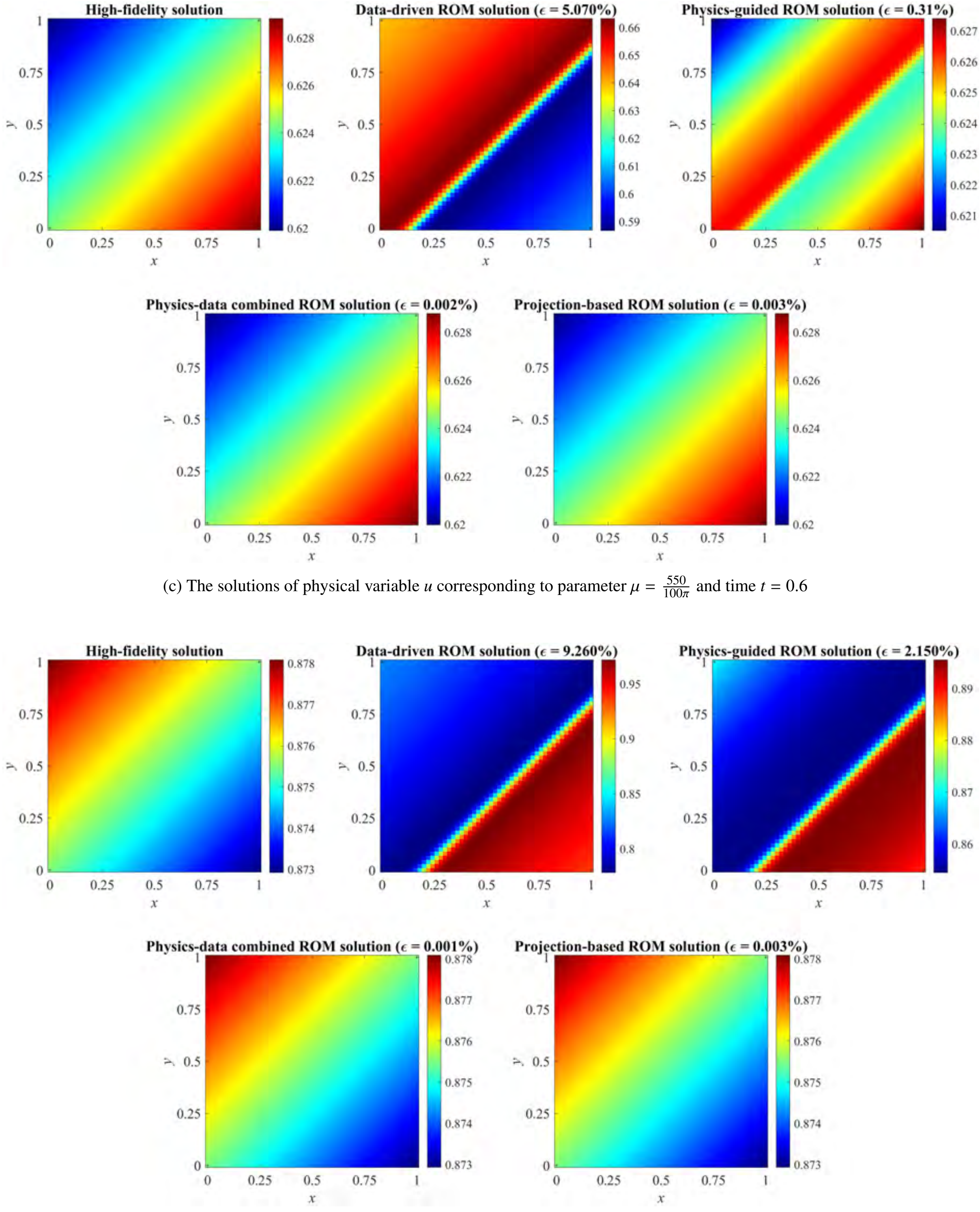
Figure 10: The reduced-order solutions of the physical fields u and v corresponding to different parameters μ (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



(a) The solutions of physical variable u corresponding to parameter $\mu = \frac{50}{100\pi}$ and time $t = 0.2$



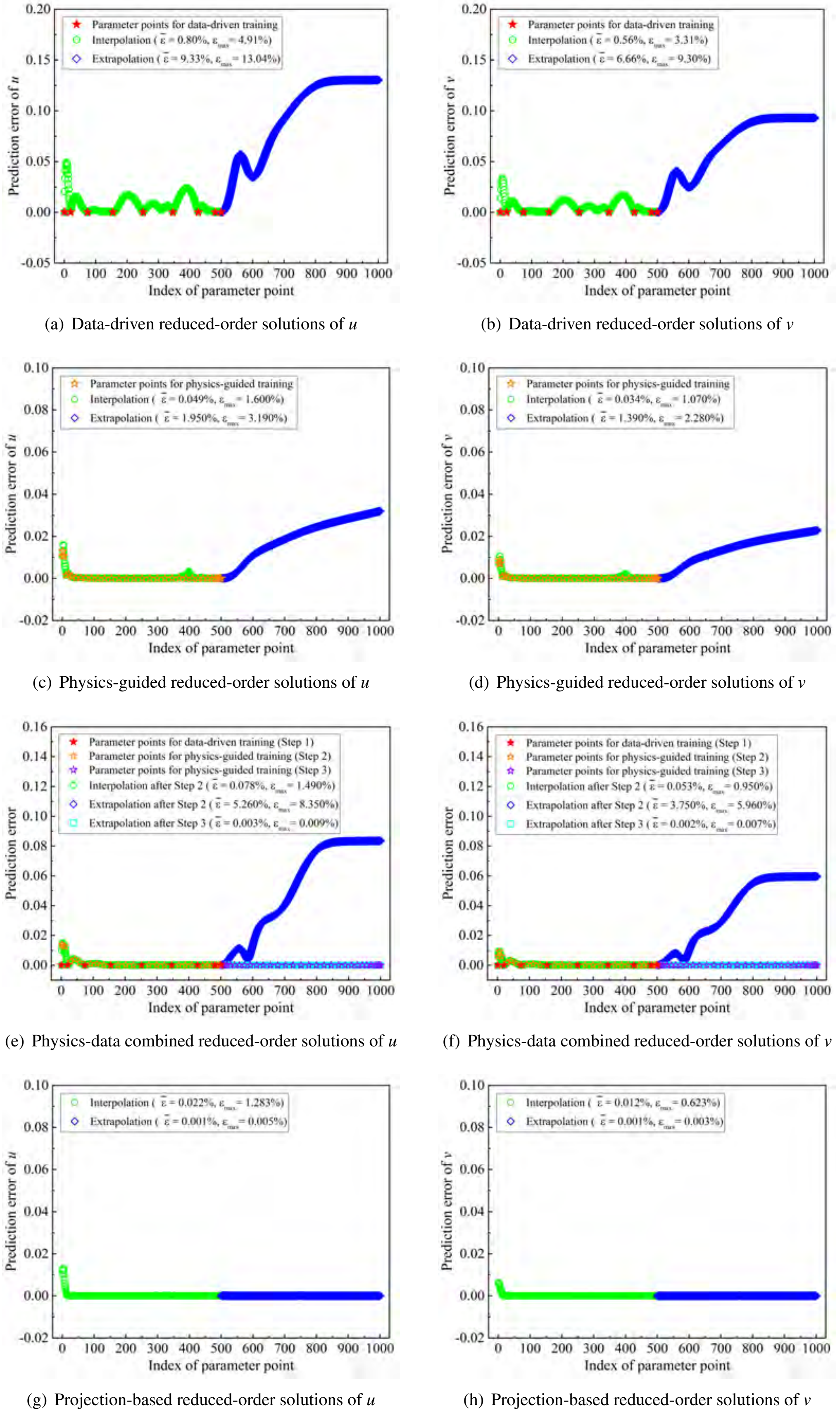
(b) The solutions of physical variable v corresponding to parameter $\mu = \frac{200}{100\pi}$ and time $t = 0.4$



(c) The solutions of physical variable u corresponding to parameter $\mu = \frac{550}{100\pi}$ and time $t = 0.6$

(d) The solutions of physical variable v corresponding to parameter $\mu = \frac{900}{100\pi}$ and time $t = 0.8$

Figure 11: The reduced-order solutions of the physical fields u and v corresponding to different parameters μ and time t (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



(a) Data-driven reduced-order solutions of u

(b) Data-driven reduced-order solutions of v

(c) Physics-guided reduced-order solutions of u

(d) Physics-guided reduced-order solutions of v

(e) Physics-data combined reduced-order solutions of u

(f) Physics-data combined reduced-order solutions of v

(g) Projection-based reduced-order solutions of u

(h) Projection-based reduced-order solutions of v

Figure 12: The relative L_2-norm errors ε of the reduced-order solutions estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε over a specific parameter range).

In general, the reduced-order solutions predicted from the physics-guided FNN model are much more accurate than that predicted from the data-driven FNN model. As recorded in Figure 12c and d, the largest interpolation errors of u and v are 1.60% and 1.07% respectively. It seems that the physics-guided neural network doesn't evolve into an accurate state for the small parameter values ($\boldsymbol{\mu} \leq \frac{5\pi}{100}$). One reason can be that the POD basis is inadequate to capture the detailed features of the physical fields corresponding to small values of μ. Another reason can be that these physical fields usually contain steep-gradient areas, which makes the finite difference approximation of partial derivatives less accurate, thereby hindering the physics-guided training process to reach an accurate state. Besides, the physics-guided FNN model also shows a better extrapolation performance, compared to the data-driven FNN model. As shown in Figure 12c and d, the average extrapolation errors of u and v are 3.19% and 2.28% respectively.

According to the above analyses, neither the data-driven ROM nor the physics-guided ROM method is able to construct a reliable reduced-order model for the 2D Burgers' equations in the small-data regime. The physics-data combined ROM method is thus applied to this 2D nonlinear problem, and the methodology is detailedly explained in Section 3.5. In Step 1, a preliminary FNN model is fitted through data-driven training by using the labelled training data ($\widetilde{\boldsymbol{\mu}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$). Because the labelled training data is too sparse, this preliminary FNN model needs to be further improved in terms of generalization capacity. In Step 2, a physics-data combined loss function (as defined in Eq. (28)) is used to replace the data-driven loss function, and this training process is primarily controlled by the physics-guided loss but constrained by the $\widetilde{\boldsymbol{\mu}}_{\text{data}}$-$\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ relation learned in Step 1. The physics-guided loss here is computed for the complementary set of $\widetilde{\boldsymbol{\mu}}_{\text{data}}$ in $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}}$, which is denoted by $\widetilde{\boldsymbol{\mu}}^{(\text{inter})}_{\text{physics}} - \widetilde{\boldsymbol{\mu}}_{\text{data}}$. Further, Step 3 is carried out to reinforce the extrapolation capacity of the FNN model obtained in Step 2. A set of 33 parameter points are sampled from $\mathcal{P}_{\text{extra}} : \left[\frac{501}{100\pi}, \frac{1000}{100\pi}\right]$ by using the 1D Smolyak sparse grid method with approximation level 5, which are denoted by $\boldsymbol{\mu}^{(\text{extra})}_{\text{physics}}$ here. After data normalisation, $\widetilde{\boldsymbol{\mu}}^{(\text{extra})}_{\text{physics}}$ is obtained to compute the physics-guided loss. The training objective is to minimise the physics-guided loss with the preservation of the input-output mappings learned in Step 1 and Step 2.

The converge curve of the physics-data combined loss function over the entire training process is plotted in Figure 9c, from which one can see that the loss function fluctuates because of the dynamic weight coefficients. Representative reduced-order solutions are presented in Figure 10, and one can see that the physics-data combined model shows the best prediction performance, compared to the data-driven and the physics-guided models. As recorded in Figure 12e and f, the relative L_2-norm errors of the reduced-order solutions predicted from the physics-data combined model are provided. The average interpolation errors of u and v are 0.078% and 0.053% respectively, and the average extrapolation errors of u and v are also significantly reduced to 0.003% and 0.002% respectively. Besides, this physics-data combined model can still work well when μ is small, where the physical field contains steep-gradient regions. In summary, the physics-data combined FNN model is reliable for the entire parameter range, and it exhibits prominent advantages over the data-driven and physics-guided FNN models.

In addition, the finite difference method is used to numerically solve 2D Burgers' equations, from which 9 snapshots corresponding to μ_{data} are obtained. Following the procedure explained in [10], a projection-based model is constructed in an intrusive ROM manner, and representative predictions of the reduced-order solutions are provided in Figure 10. As recorded in Figure 12g and h, the prediction errors of this projection-based model are very small over the entire parameter range. Due to the inadequacy of the POD basis, this projection-based model also exhibits less reliable performance for the small parameter values ($\boldsymbol{\mu} \leq \frac{5\pi}{100}$). Overall, the prediction accuracy of the physics-data combined FNN model is comparable to that of the projection-based model, which further confirms the effectiveness of the proposed non-intrusive ROM method.

4.3. Case 3: Steady lid-driven cavity flow

Lid-driven cavity flow is an important benchmark problem in various studies [4, 48, 49]. Here, a case study of steady incompressible flow in the 2D lid-driven square cavity is investigated, to verify the applicability of the proposed ROM method to the nonlinear problems containing three physical variables. The evolution of nonlinear fluid dynamics is governed by the steady Navier-Stokes equations, which are mathematically expressed as follows:

$$\begin{cases} \dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} = 0 \\ u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} + \dfrac{\partial p}{\partial x} - \dfrac{1}{Re}\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) = 0 \\ u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} + \dfrac{\partial p}{\partial y} - \dfrac{1}{Re}\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right) = 0 \end{cases} \tag{39}$$

where u and v are the velocity components along x-axis and y-axis respectively, p is the pressure, and Re denotes the Reynolds number.

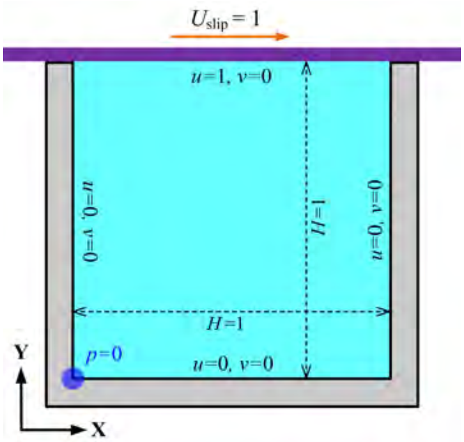
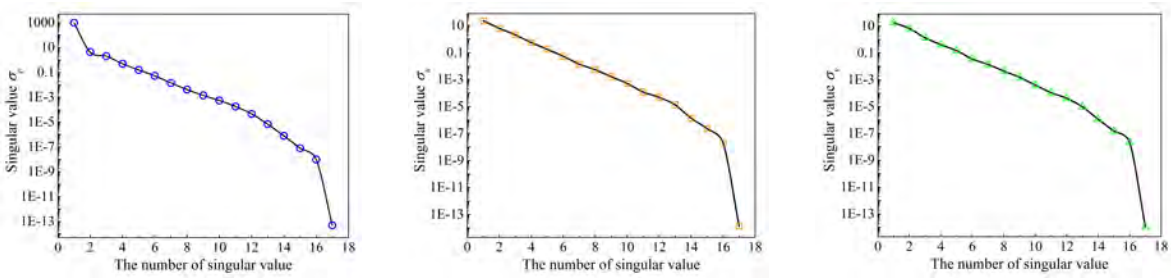


Figure 13: Schematic diagram of lid-driven flow in a 2D square cavity.

Finite Volume Method (FVM) [50] is adopted to simulate the classical lid-driven cavity flow by numerically solving the steady Navier-Stokes equations with varying Reynolds numbers Re. The geometry and the boundary conditions of the 2D square cavity are graphically illustrated in Figure 13, where the sidewall length H is equal to 1. The slip lid on top moves along the positive x direction with unit velocity U_{slip}=1, and no-slip boundary conditions are applied to the other three fixed sidewalls. The dimensionless spatial domain is $x \in \mathcal{X} : [0, 1] \times y \in \mathcal{Y} : [0, 1]$. FVM simulation with the primitive variable formulation is performed on a uniform staggered grid 131 × 131, and SIMPLE algorithm [50] is used to decouple velocity and pressure. More information on FVM and its advantages can be found in the relevant references [50]. It should be noted that the scalar variable (pressure p) is stored in the cell centres for the uniform staggered grid, while the momentum variables (velocity components u and v) are stored at the cell faces. Such grid arraignment can avoid the odd-even decoupling error between pressure and velocity [50]. However, the obtained numerical solutions of u, v and p have different data dimensions, which are 132 × 131, 131 × 132 and 131 × 131 respectively. Linear interpolation is thus applied to the snapshots of u and v to estimate the velocity values in the cell centres, through which the raw snapshots of u and v are rearranged into new snapshot data with the same dimension (131 × 131) of snapshots p.



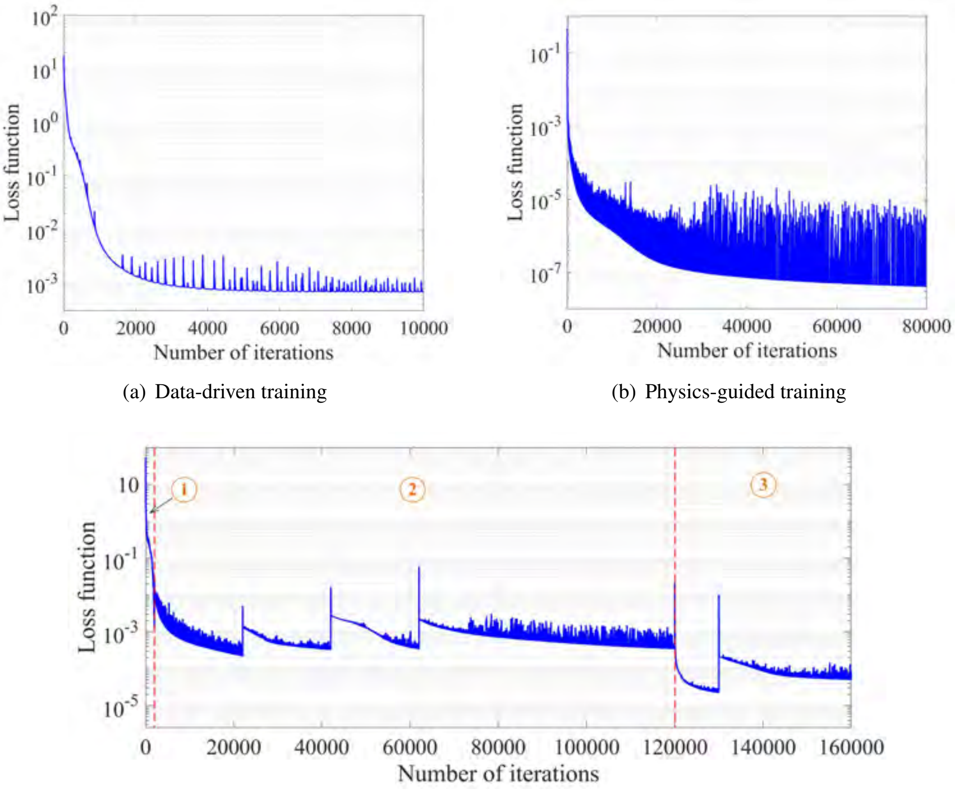
(a) σ_p

(b) σ_u

(c) σ_v

Figure 14: The singular values σ_p, σ_u and σ_v plotted in descending orders for the physical variables of p, u and v.

Reynolds number Re is the dimensionless parameter of interest in this test case, and it uniformly varies over a predefined parameter space $\mathcal{P} : \left[1,\, 500\right]$ with an interval of 1. The entire parameter space $\mathcal{P}$ contains 500 parameter points, which is then partitioned into two sub-domains denoted by $\mathcal{P}_{\text{inter}} : \left[101,\, 500\right]$ and $\mathcal{P}_{\text{extra}} : \left[1,\, 100\right]$ respectively. The 1D Smolyak sparse grid method with approximation level 4 is performed on $\mathcal{P}_{\text{inter}} : \left[101,\, 500\right]$ for data sampling, and a set of 17 parameter points are thus selected, which are denoted by $\boldsymbol{Re}_{\text{data}}$. High-fidelity solutions corresponding to the selected parameter points can be obtained from FVM simulations by numerically solving the steady Navier-Stokes equations in Eq. (39). Separately performing POD on the obtained snapshots of p, u and v yields three sets of singular values and singular vectors. As plotted in Figure 14a-c, singular values σ_p, σ_u and σ_v are recorded in descending orders respectively. Three sets of POD modes can be obtained for physical variables p, u and v by choosing the first 8 left-singular vectors to represent the primary dynamics of fluid flow, according to the criterion defined in Eq. (8). Besides, the POD expansion coefficients $\boldsymbol{\alpha}_{\text{data}}$ can also be quickly computed by separately projecting the snapshots of p, u and v to the corresponding subspace.



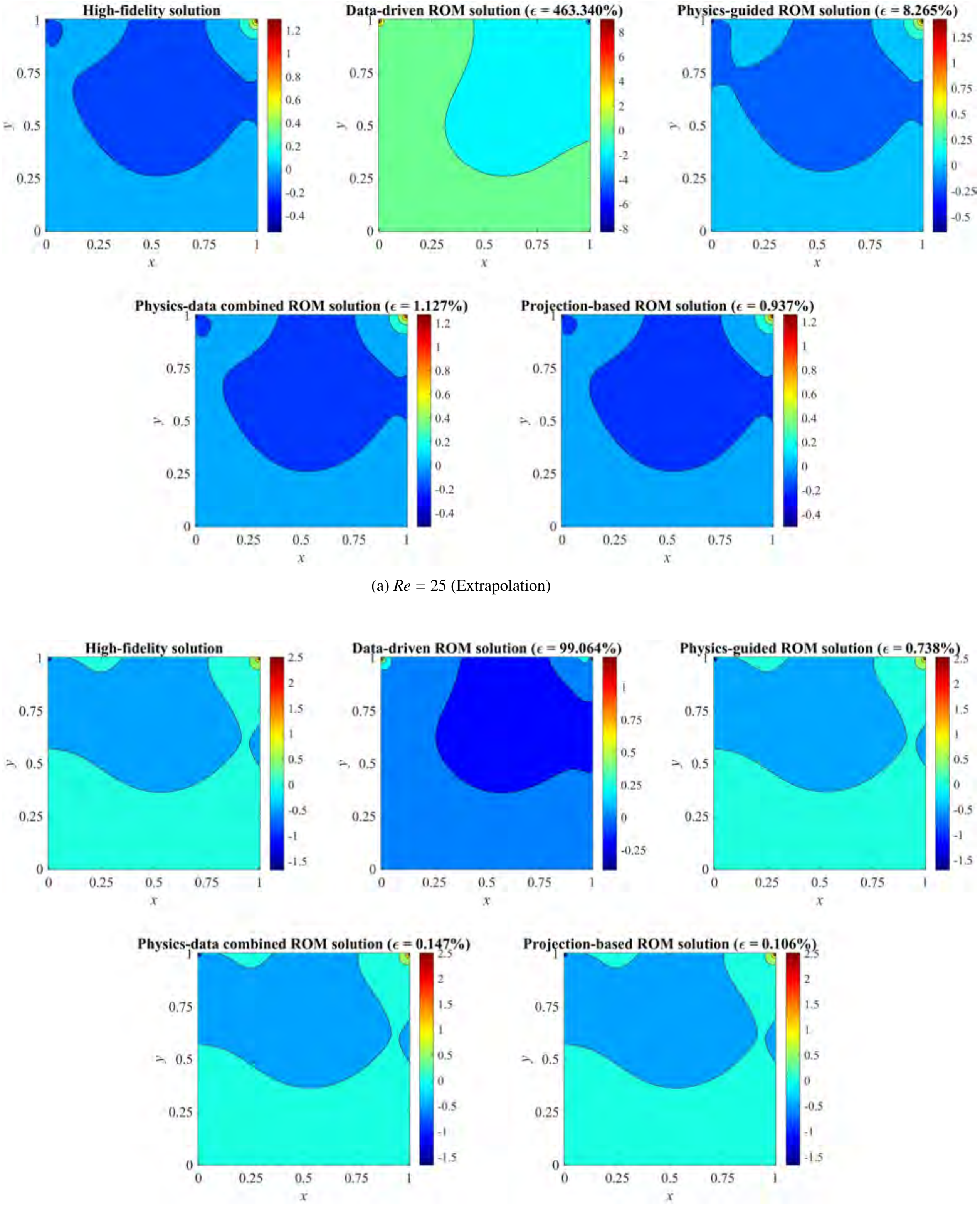
(a) Data-driven training

(b) Physics-guided training

(c) Physics-data combined training

Figure 15: The convergence curves of the neural networks' loss functions using (a) Data-driven ROM method, (b) Physics-guided ROM method and (c) Physics-data combined ROM method.

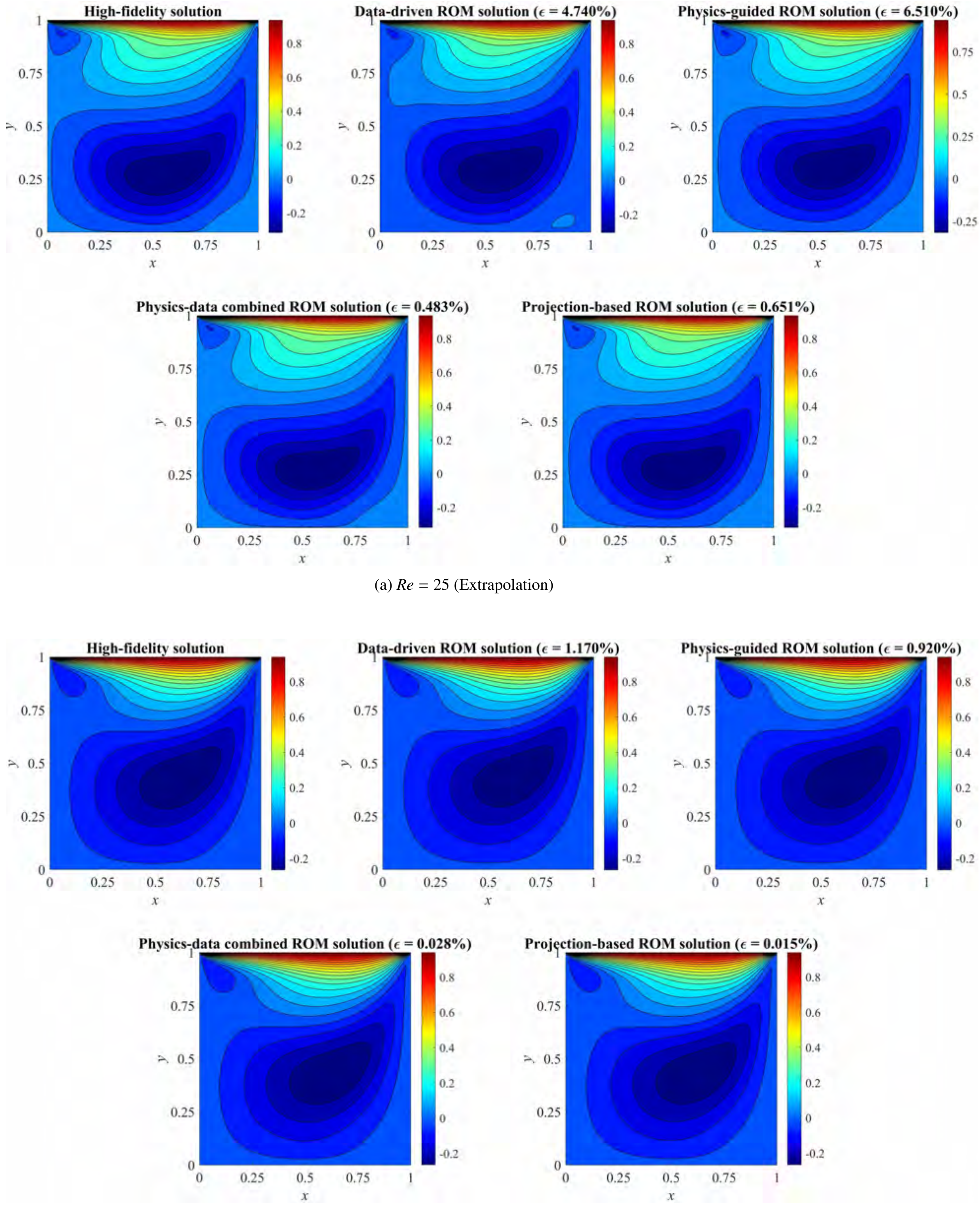
After performing data normalisation on $(\boldsymbol{Re}_{\text{data}}, \boldsymbol{\alpha}_{\text{data}})$, the obtained data pairs $\left(\widetilde{\boldsymbol{Re}}_{\text{data}}, \widetilde{\boldsymbol{\alpha}}_{\text{data}}\right)$ are the only labelled training data for data-driven ROM. Because the training data is small, the feedforward neural network rapidly evolves to a stable state, as illustrated in Figure 15a. Once the neural network has been properly fitted, reduced-order solutions of p, u and v can be predicted from it, and representative results can be found in Figure 16, 17 and 18 respectively. Compared to high-fidelity solutions, the reduced-order solutions predicted from the data-driven FNN model are inaccurate, especially for pressure p. Compared to velocity fields, the pressure field is much more sensitive to the varying Re, which may explain why the data-driven model is less accurate in predicting pressure solutions. The relative L_2-norm error of the reduced-order solutions are recorded in Figure 19a, 20a and 21a, from which one can see that this data-driven model shows poor performances on both interpolation and extrapolation. The largest interpolation error and extrapolation error of pressure p are up to 153.30% and 533.95% respectively, although the prediction errors of velocity components u and v are insignificant.



(a) Re = 25 (Extrapolation)

(b) Re = 325 (Interpolation)

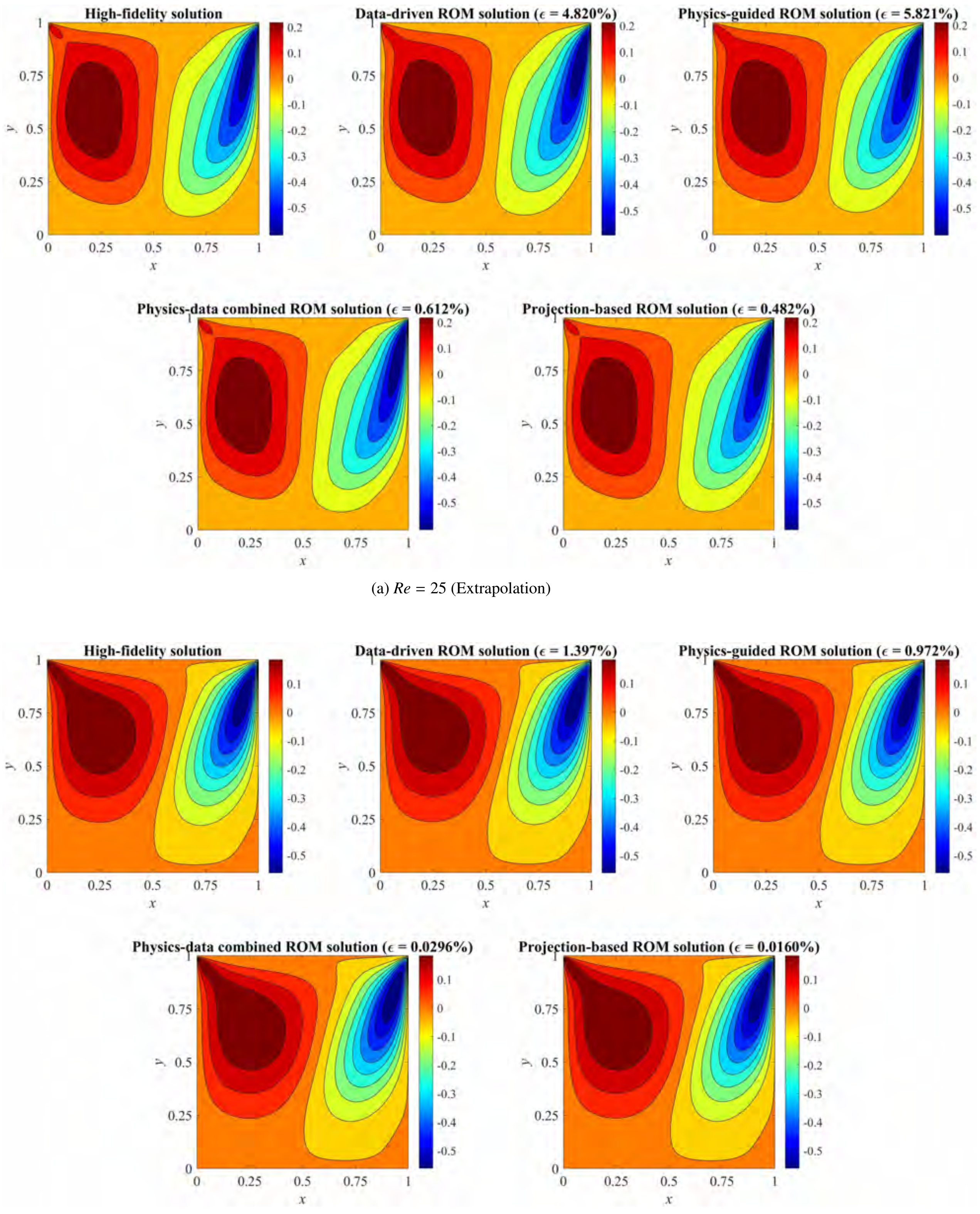
Figure 16: The reduced-order solutions of pressure p corresponding to different parameters Re (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



(a) Re = 25 (Extrapolation)

(b) Re = 325 (Interpolation)

Figure 17: The reduced-order solutions of velocity component u corresponding to different parameters Re (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



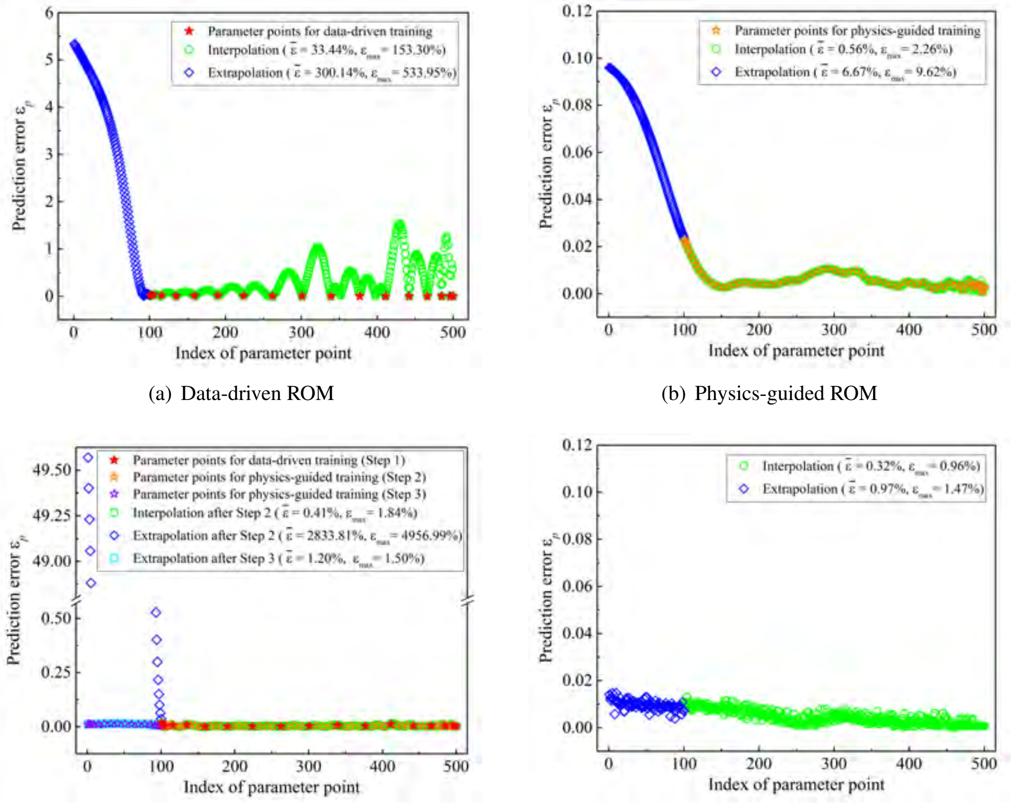
(a) Re = 25 (Extrapolation)

(a) Re = 325 (Interpolation)

Figure 18: The reduced-order solutions of velocity component v corresponding to different parameters Re (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).

As a label-free method, the physics-guided ROM can well deal with the shortage of labelled training data by embedding the physical laws into the neural network. A set of 33 parameter points are selected from $\mathcal{P}_{\text{inter}}$:

$\left[101,\, 500\right]$ by using the 1D Smolyak sparse grid method with approximation level 5, which are denoted by $\boldsymbol{Re}^{(\text{inter})}_{\text{physics}}$. After data normalisation, the parameter points in $\widetilde{\boldsymbol{Re}}^{(\text{inter})}_{\text{physics}}$ are the only inputs to compute the physics-guided loss functions, as defined in Eq. (26). Here, the central finite difference is adopted to approximate the partial derivatives to 6-order accuracy. The convergence curve of the physics-guided loss function is plotted in Figure 15b, from which one can see that physics-guided training requires more iterative steps than data-driven training. After the neural network is properly trained, reduced-order solutions of physical variables p, u and v can be rapidly predicted from it, and representative results are provided in Figure 16, 17 and 18 respectively. Roughly, the physics-guided FNN model exhibits a satisfactory performance in terms of interpolation. As illustrated in Figure 19b, 20b and 21b, the average interpolation errors of physical variables p, u and v are 0.59%, 0.62% and 0.58% respectively. However, the maximum extrapolation errors of p, u and v are up to 9.62%, 8.01% and 7.94% respectively, which means this physic-guided FNN model is unable to provide accurate reduced-order solutions for parameters that are beyond the training scope.



(a) Data-driven ROM

(b) Physics-guided ROM

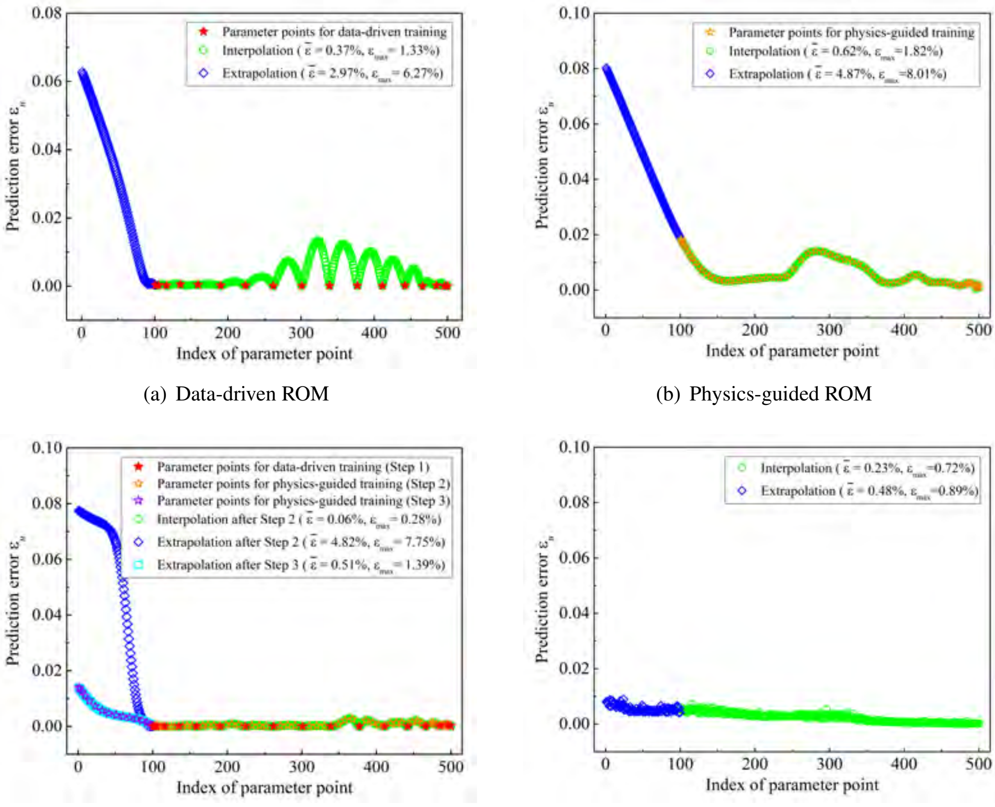
(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 19: The relative L_2-norm errors ε_p of the reduced-order solutions of p estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε_p over a specific parameter range).

To construct a surrogate model with a strong generalization capacity, the physics-data combined ROM method is applied to this 2D nonlinear problem. As explained in Section 3.5, the reduced-order numerical system and the available data are jointly integrated into the feedforward neural network through a step-by-step scheme. Step 1 is to fit a preliminary FNN model through data-driven training, during which the labelled training data ($\widetilde{\boldsymbol{Re}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) is embedded into the neural network. Due to data sparsity, this preliminary FNN model required to be enhanced in terms of interpolation and extrapolation. Step 2 is to improve the interpolation performance by replacing the data-driven loss function with a physics-data combined loss function, as defined in Eq. (28). The training process is dominated by the physics-guided loss term but constrained by the $\widetilde{\boldsymbol{Re}}_{\text{data}}$-$\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ mapping learned in the 1st step. The physics-guided loss term here is computed for the complementary set of $\widetilde{\boldsymbol{Re}}_{\text{data}}$ in $\widetilde{\boldsymbol{Re}}^{(\text{inter})}_{\text{physics}}$, which is denoted by $\widetilde{\boldsymbol{Re}}^{(\text{inter})}_{\text{physics}} - \widetilde{\boldsymbol{Re}}_{\text{data}}$. Step 3 is to reinforce the extrapolation capacity of the FNN model obtained in

Step 2. A set of 17 parameter points are sampled from $\mathcal{P}_{\text{extra}}$: [1, 100] by using the 1D Smolyak sparse grid method with approximation level 4, which are denoted by $\boldsymbol{Re}^{(\text{extra})}_{\text{physics}}$ here. After data normalisation, $\widetilde{\boldsymbol{Re}}^{(\text{extra})}_{\text{physics}}$ is obtained to compute the physics-guided loss. The training objective is to minimise the physics-guided loss with the preservation of the input-output mappings learned in Step 1 and Step 2.



(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

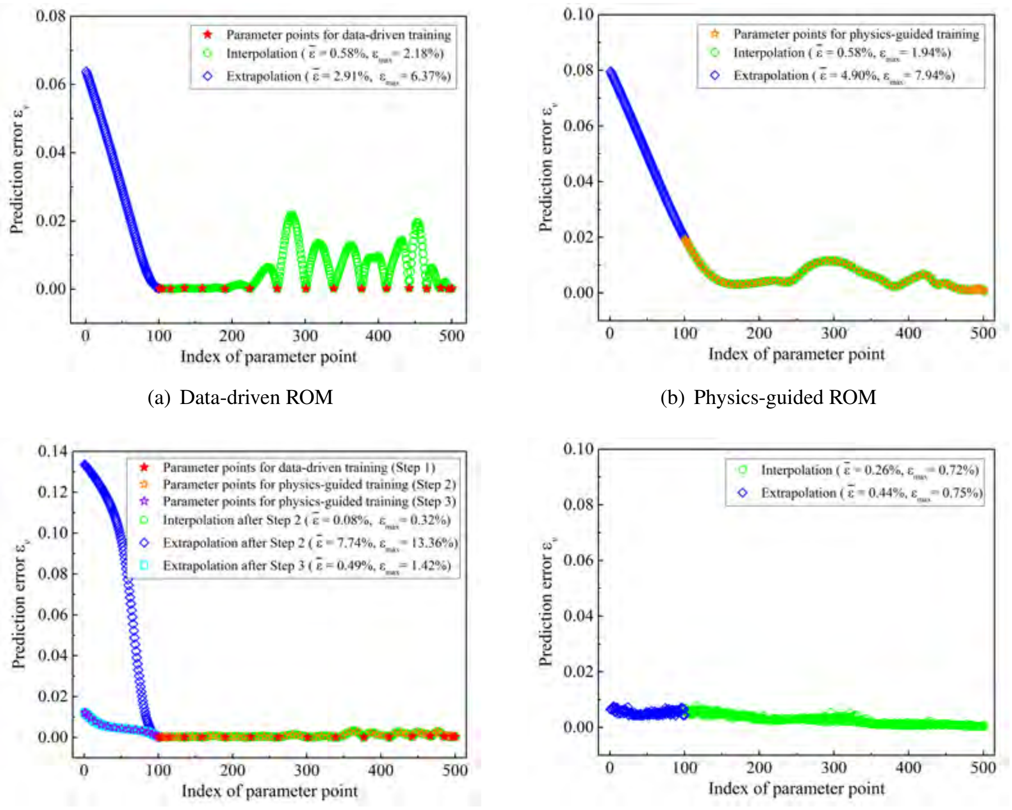
(d) Projection-based ROM

Figure 20: The relative L_2-norm errors ε_u of the reduced-order solutions of u estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε_u over a specific parameter range).

The converge curve of the physics-data combined loss function over the entire training process is plotted in Figure 15c, and its fluctuations mainly result from the usage of dynamic weight coefficients. As presented in Figure 16-18, reduced-order solutions of p, u and v corresponding to two representative parameter points are predicted from the physics-data combined FNN model. To quantitatively assess the ROM quality, relative L_2-norm errors of the reduced-order solutions corresponding to all 500 parameter points are calculated according to Eq. (31), as plotted in Figure 19-21. By comparison, the prediction accuracy of the physics-data combined model is much higher than that of the data-driven and physics-guided models. The average interpolation errors of p, u and v are 0.41%, 0.06% and 0.08% respectively, and the average extrapolation errors of p, u and v are also greatly reduced to 1.20%, 0.51% and 0.49% respectively. Therefore, this physics-data combined FNN model possesses a strong generalization capacity to provide reliable reduced-order solutions for the entire parameter range.

Furthermore, the traditional projection-based method [10, 11] is applied to this steady lid-driven cavity flow problem to construct an intrusive reduced-order model by using the same orthogonal bases containing 8 POD modes. More information about this projection-based method and the corresponding open-source code can be found in the relevant reference [10]. As shown in Figure 19, 20 and 21, the projection-based model exhibits excellent performance in predicting reduced-order solutions of p, u and v. The relative L_2-norm errors of reduced-order solutions over the entire parameter range are recorded in Figure 19, 20 and 21. This projection-based model is more accurate than the physical-data combined model in terms of extrapolation prediction, but the latter has a better performance than the former in aspects of interpolation prediction. Both the projection-based model and the physical-data combined model could be significantly improved, if adequate POD bases are properly determined.

Overall, the comparable performance between the projection-based model and the physical-data combined model further confirms that the physics-data combined ROM method is reliable for the ROM of nonlinear dynamical systems in small-data regimes.



(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 21: The relative L_2-norm errors ε_v of the reduced-order solutions of v estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε_v over a specific parameter range).

4.4. *Case 4: Steady Rayleigh-Bénard convection*

Natural convection is a classical problem containing important mechanisms of heat and mass transfer, which has been extensively investigated in numerous studies [4, 51, 52]. Here, Steady 2D Rayleigh-Bénard convection that contains 4 physical variables is chosen as the fourth case to verify the proposed ROM method. As graphically illustrated in Figure 22, a layer of fluid is confined in a 2D enclosure with isothermal horizontal boundaries, and the enclosure is heated from the bottom and cooled from the top. Buoyancy-driven flow occurs in the presence of a gravitational field, due to the unstable variation of fluid density caused by thermal inhomogeneity. To model the steady thermal convection in a 2D fluid layer, conservation of mass, conservation of energy and the Boussinesq approximation [51, 52] to the non-dimensional Navier-Stokes equations are used:

$$\begin{cases} \dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} = 0 \\ u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} + \dfrac{\partial p}{\partial x} - \sqrt{\dfrac{Pr}{Ra}}\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) = 0 \\ u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} + \dfrac{\partial p}{\partial y} - \sqrt{\dfrac{Pr}{Ra}}\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right) + T\mathbf{e}_g = 0 \\ u\dfrac{\partial T}{\partial x} + v\dfrac{\partial T}{\partial y} - \dfrac{1}{\sqrt{Pr\,Ra}}\left(\dfrac{\partial^2 T}{\partial x^2} + \dfrac{\partial^2 T}{\partial y^2}\right) = 0 \end{cases} \tag{40}$$

where u and v are the velocity components along $x-$axis and $y-$axis respectively, p is the pressure, T is the temperature, and $\mathbf{e}_g$ is the unit vector along the direction of gravitational acceleration. The Prandtl number Pr characterises the fluid properties:

$$Pr = \frac{\mu}{\kappa} \tag{41}$$

where μ and κ are the kinematic viscosity and the thermal diffusivity of the fluid. Here, $Pr = 0.71$, which is the Prandtl number of air at room temperature.

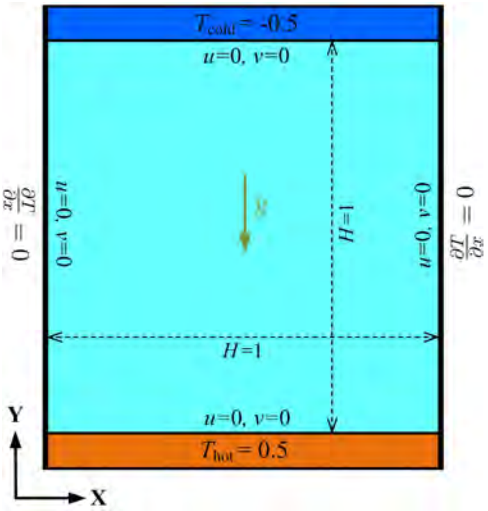


Figure 22: Schematic diagram of Rayleigh-Bénard convection in a 2D square enclosure.

The *Rayleigh number Ra* is the dimensionless parameter that can affect the solutions of the PEDs in Eq. (40), and it characterises the temperature difference across the enclosure:

$$Ra = \frac{g\lambda(T_{\text{hot}} - T_{\text{cold}})H^3}{\kappa\mu} \tag{42}$$

where g denotes gravitational acceleration, λ is the thermal expansion coefficient, H is the characteristic length of the 2D enclosure, and $T_{\text{hot}} - T_{\text{cold}}$ denotes the temperature difference across the 2D enclosure.

The Finite Volume Method is applied to simulate the Buoyancy-driven flow by numerically solving the steady Navier-Stokes equations with varying Rayleigh numbers Ra. As to the boundary conditions, the 2D enclosure consists of four non-slip and no penetration walls, which means $u = 0, v = 0$ on four walls. The thermal boundary conditions on the bottom and top walls satisfy:

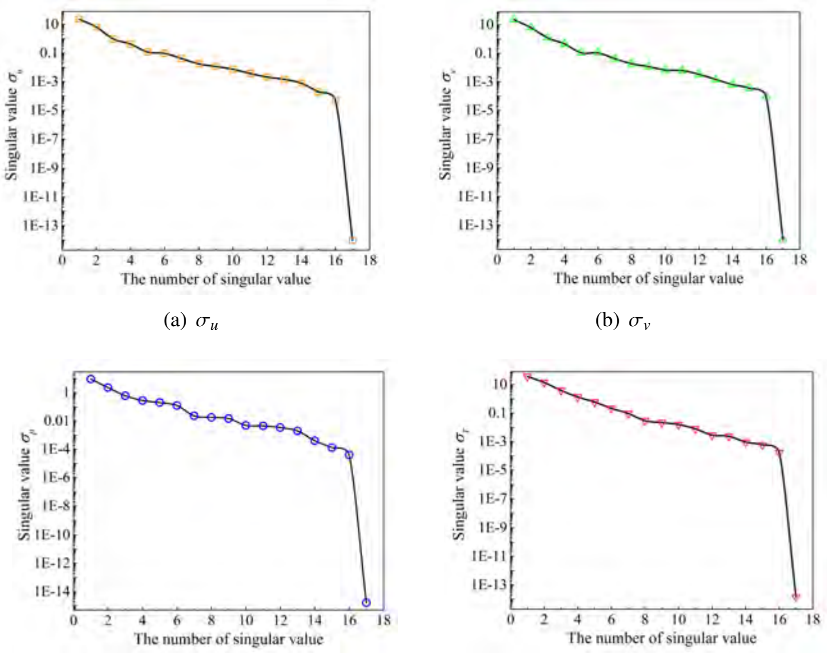
$$\begin{cases} T(x, 0) = +0.5 \\ T(x, 1) = -0.5 \end{cases} \tag{43}$$

The temperature on the other walls satisfies isothermal conditions:

$$\begin{cases} \left.\dfrac{\partial T}{\partial x}\right|_{x=0} = 0 \\ \left.\dfrac{\partial T}{\partial x}\right|_{x=1} = 0 \end{cases} \tag{44}$$

The dimensionless spatial domain is $x \in \mathcal{X} : [0, 1] \times y \in \mathcal{Y} : [0, 1]$. FVM simulation with the primitive variable formulation is performed on a uniform staggered grid 131 × 131, and the SIMPLE algorithm is used to decouple velocity and pressure/temperature. However, the obtained numerical solutions of u, v, p and T have different data dimensions, which are 132 × 131, 131 × 132, 131 × 131 and 131 × 131 respectively. Linear interpolation is thus applied to the snapshots of u and v to estimate the velocity values in the cell centres, through which the raw snapshots of u and v are rearranged into new snapshot data with the same dimension (131 × 131) of snapshots p and T.

Rayleigh number Ra uniformly varies over a predefined parameter space $\mathcal{P}$: [10000, 260000] with an interval of 500. The entire parameter space $\mathcal{P}$ contains 501 parameter points, which is then partitioned into two subdomains denoted by $\mathcal{P}_{\text{inter}}$: [10000, 210000] and $\mathcal{P}_{\text{extra}}$: [210500, 260000] respectively. The 1D Smolyak sparse grid method with approximation level 4 is performed on $\mathcal{P}_{\text{inter}}$ for data sampling, and a set of 17 parameter points are thus selected, which are denoted by $\boldsymbol{Ra}_{\text{data}}$. High-fidelity solutions corresponding to the selected parameter points can be obtained from FVM simulations. Separately performing POD on the obtained snapshots of u, v, p and T yields four sets of singular values and singular vectors. As plotted in Figure 23a-d, singular values σ_u, σ_v, σ_p and σ_T are recorded in descending orders respectively. Four sets of POD modes can be obtained for physical variables u, v, p and T by choosing all 17 left-singular vectors to represent the primary dynamics of fluid flow. Besides, the POD expansion coefficients $\boldsymbol{\alpha}_{\text{data}}$ can also be quickly computed by separately projecting the snapshots of u, v, p and T to the corresponding subspace.



(a) σ_u

(b) σ_v

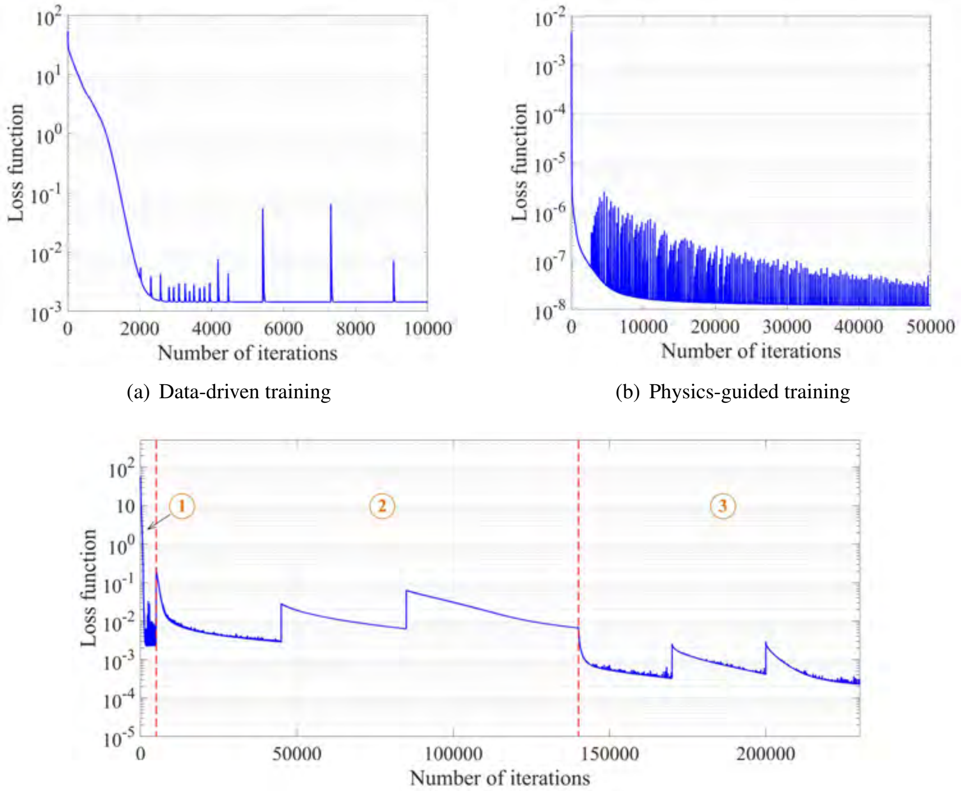
(c) σ_p

(d) σ_T

Figure 23: The singular values σ_u, σ_v, σ_p and σ_T plotted in descending orders for the physical variables of u, v, p and T.

After performing data normalisation on $(\boldsymbol{Ra}_{\text{data}}, \boldsymbol{\alpha}_{\text{data}})$, the obtained data pairs $\left(\widetilde{\boldsymbol{Ra}}_{\text{data}}, \widetilde{\boldsymbol{\alpha}}_{\text{data}}\right)$ are the only labelled training data for data-driven ROM. Because the training data is small, the FNN model rapidly evolves to a stable state, as illustrated in Figure 24a. Once the neural network has been properly fitted, reduced-order solutions of u, v, p and T can be predicted from it, and representative results can be found in Figure 25, 26, 27 and 28 respectively. Compared to high-fidelity solutions, the reduced-order solutions predicted from the data-driven FNN model are inaccurate, especially for pressure p and temperature T. The relative L_2-norm error of the reduced-order solutions are recorded in Figure 29a, 30a and 31a and 32a, from which one can see that this data-driven model shows poor performances on both interpolation and extrapolation. For example, the largest interpolation and extrapolation errors of temperature T are up to 63.21% and 17.73% respectively, and the prediction errors of velocity components and pressure are also very significant.

For the physics-guided ROM method, a set of 63 parameter points are selected from $\mathcal{P}_{\text{inter}}$: [10000, 210000] by using the 1D Smolyak sparse grid method with approximation level 6, which is denoted by $\boldsymbol{Ra}_{\text{physics}}^{(\text{inter})}$. After data normalisation, the parameter points in $\widetilde{\boldsymbol{Ra}}_{\text{physics}}^{(\text{inter})}$ are the only inputs to compute the physics-guided loss functions, as defined in Eq. (26). Here, the central finite difference is adopted to approximate the partial derivatives to 6-order accuracy. The convergence curve of the physics-guided loss function is plotted in Figure 24b. After the neural network is properly trained, reduced-order solutions of physical variables u, v, p and T can be rapidly predicted from it, and representative results are provided in Figure 25, 26, 27 and 28 respectively. Generally, the physics-guided FNN model exhibits a better prediction performance, compared with the data-driven model. As



(a) Data-driven training

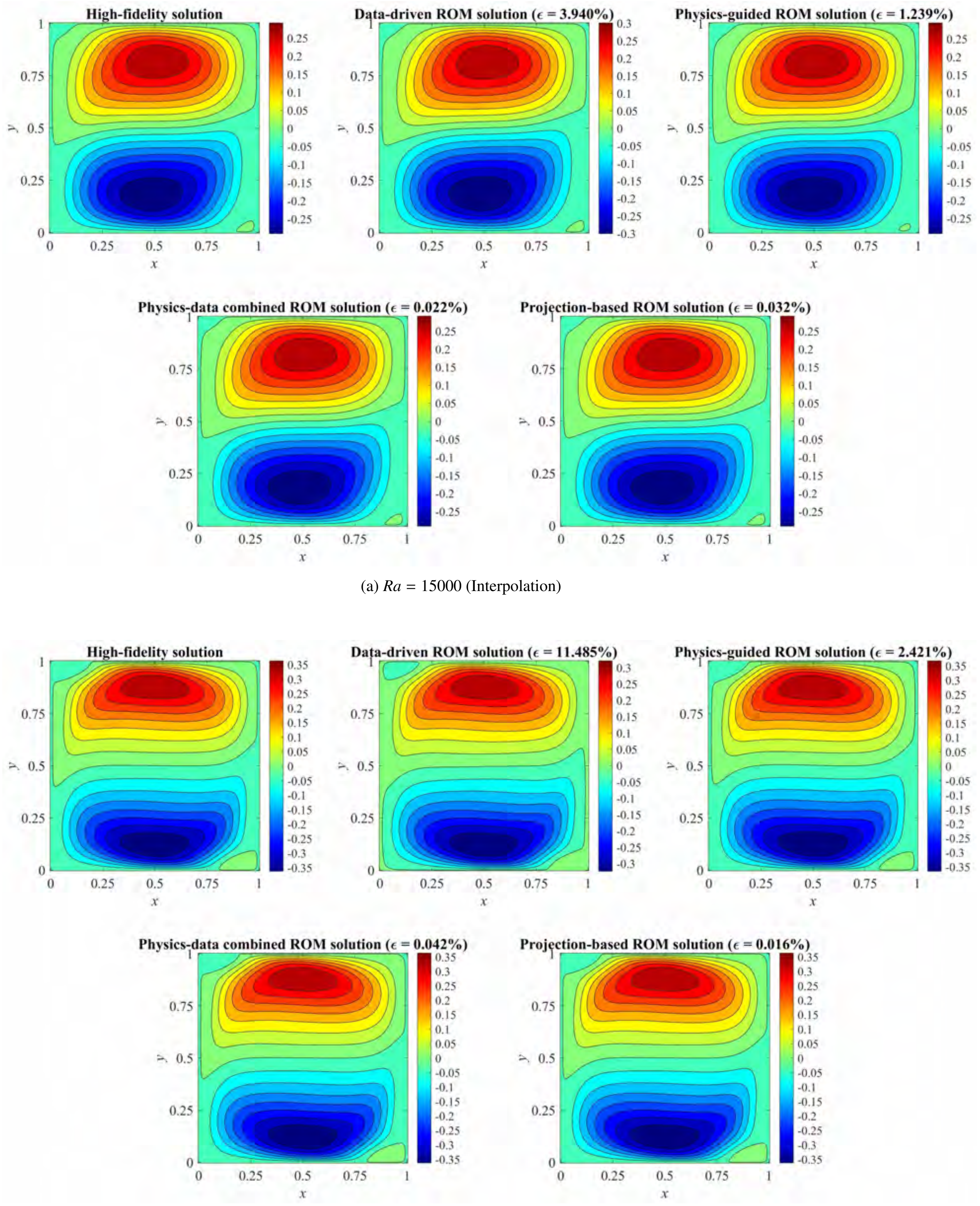
(b) Physics-guided training

(c) Physics-data combined training

Figure 24: The convergence curves of the neural networks' loss functions using (a) Data-driven ROM method, (b) Physics-guided ROM method and (c) Physics-data combined ROM method.

illustrated in Figure 29b, 30b, 31b and 32b, the average interpolation errors of physical variables u, v, p and T are 1.10%, 1.55%, 1.51% and 1.47% respectively. But the maximum extrapolation errors of u, v, p and T are up to 3.06%, 4.25%, 1.70% and 7.47% respectively, which may suggest this physic-guided FNN model is unable to provide accurate reduced-order solutions for parameters that are beyond the training scope.

For the physics-data combined ROM method, the 1st step is to fit a preliminary FNN model through data-driven training, during which the labelled training data ($\widetilde{\boldsymbol{Ra}}_{\text{data}}$, $\widetilde{\boldsymbol{\alpha}}_{\text{data}}$) is embedded into the neural network. Step 2 is to improve the interpolation performance by replacing the data-driven loss function with a physics-data combined loss function, as defined in Eq. (28). The training process is dominated by the physics-guided loss term but constrained by the $\widetilde{\boldsymbol{Ra}}_{\text{data}}$-$\widetilde{\boldsymbol{\alpha}}_{\text{data}}$ mapping learned in the 1st step. The physics-guided loss term here is computed for the complementary set of $\widetilde{\boldsymbol{Ra}}_{\text{data}}$ in $\widetilde{\boldsymbol{Ra}}^{(\text{inter})}_{\text{physics}}$, which is denoted by $\widetilde{\boldsymbol{Ra}}^{(\text{inter})}_{\text{physics}} - \widetilde{\boldsymbol{Ra}}_{\text{data}}$. Step 3 is to reinforce the extrapolation capacity of the FNN model obtained in Step 2. A set of 17 parameter points are sampled from $\mathcal{P}_{\text{extra}}$: [210500, 260000] by using the 1D Smolyak sparse grid method with approximation level 4, which are denoted by $\boldsymbol{Ra}^{(\text{extra})}_{\text{physics}}$ here. After data normalisation, $\widetilde{\boldsymbol{Ra}}^{(\text{extra})}_{\text{physics}}$ is obtained to compute the physics-guided loss. The training objective is to minimise the physics-guided loss with the preservation of the input-output mappings learned in Step 1 and Step 2.



High-fidelity solution

Data-driven ROM solution (ϵ = 3.940%)

Physics-guided ROM solution (ϵ = 1.239%)

Physics-data combined ROM solution (ϵ = 0.022%)

Projection-based ROM solution (ϵ = 0.032%)

(a) Ra = 15000 (Interpolation)

High-fidelity solution

Data-driven ROM solution (ϵ = 11.485%)

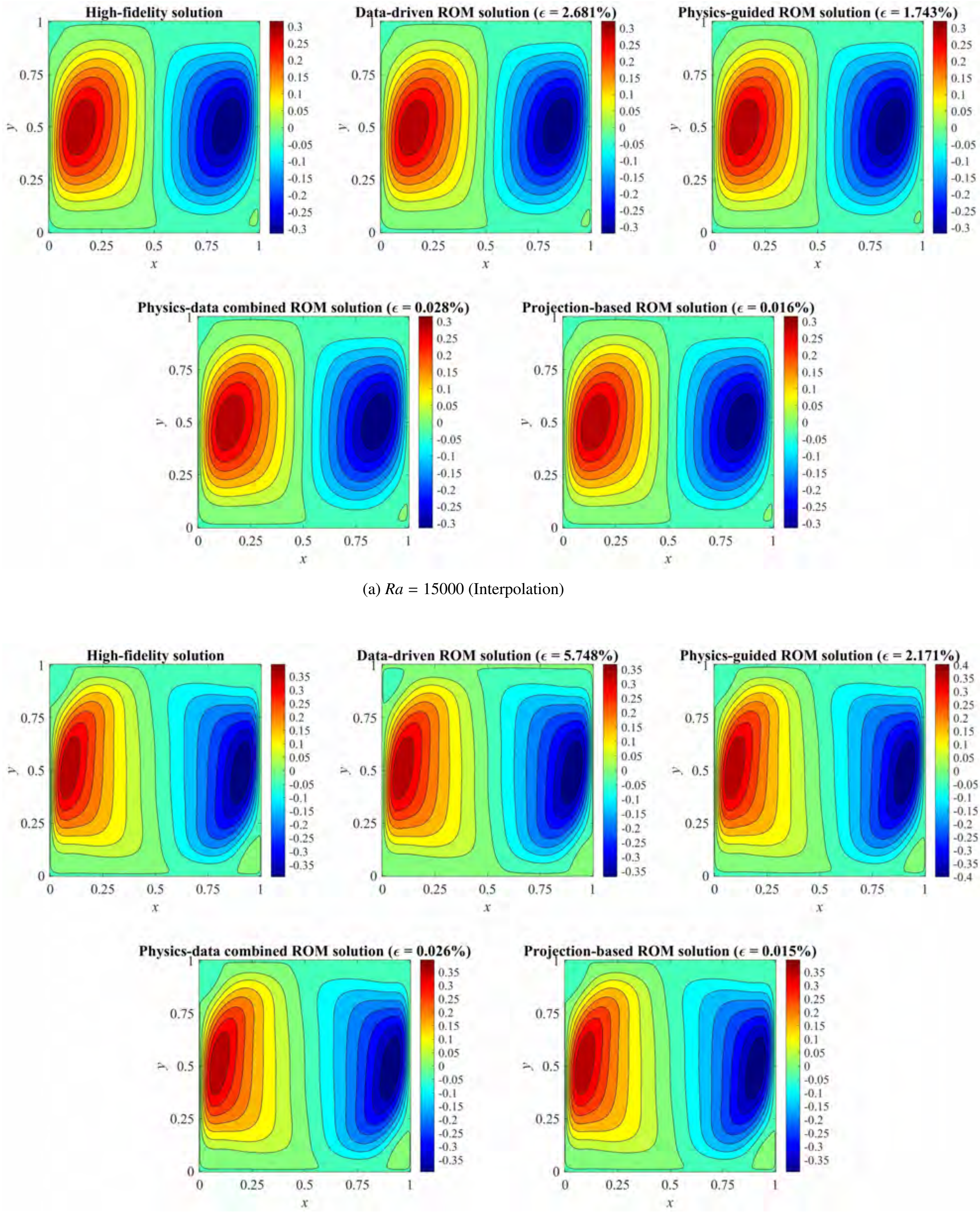
Physics-guided ROM solution (ϵ = 2.421%)

Physics-data combined ROM solution (ϵ = 0.042%)

Projection-based ROM solution (ϵ = 0.016%)

(b) Ra = 250000 (Extrapolation)

Figure 25: The reduced-order solutions of velocity component u corresponding to different parameters Ra (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).


High-fidelity solution

Data-driven ROM solution (ϵ = 2.681%)

Physics-guided ROM solution (ϵ = 1.743%)

Physics-data combined ROM solution (ϵ = 0.028%)

Projection-based ROM solution (ϵ = 0.016%)

(a) Ra = 15000 (Interpolation)

High-fidelity solution

Data-driven ROM solution (ϵ = 5.748%)

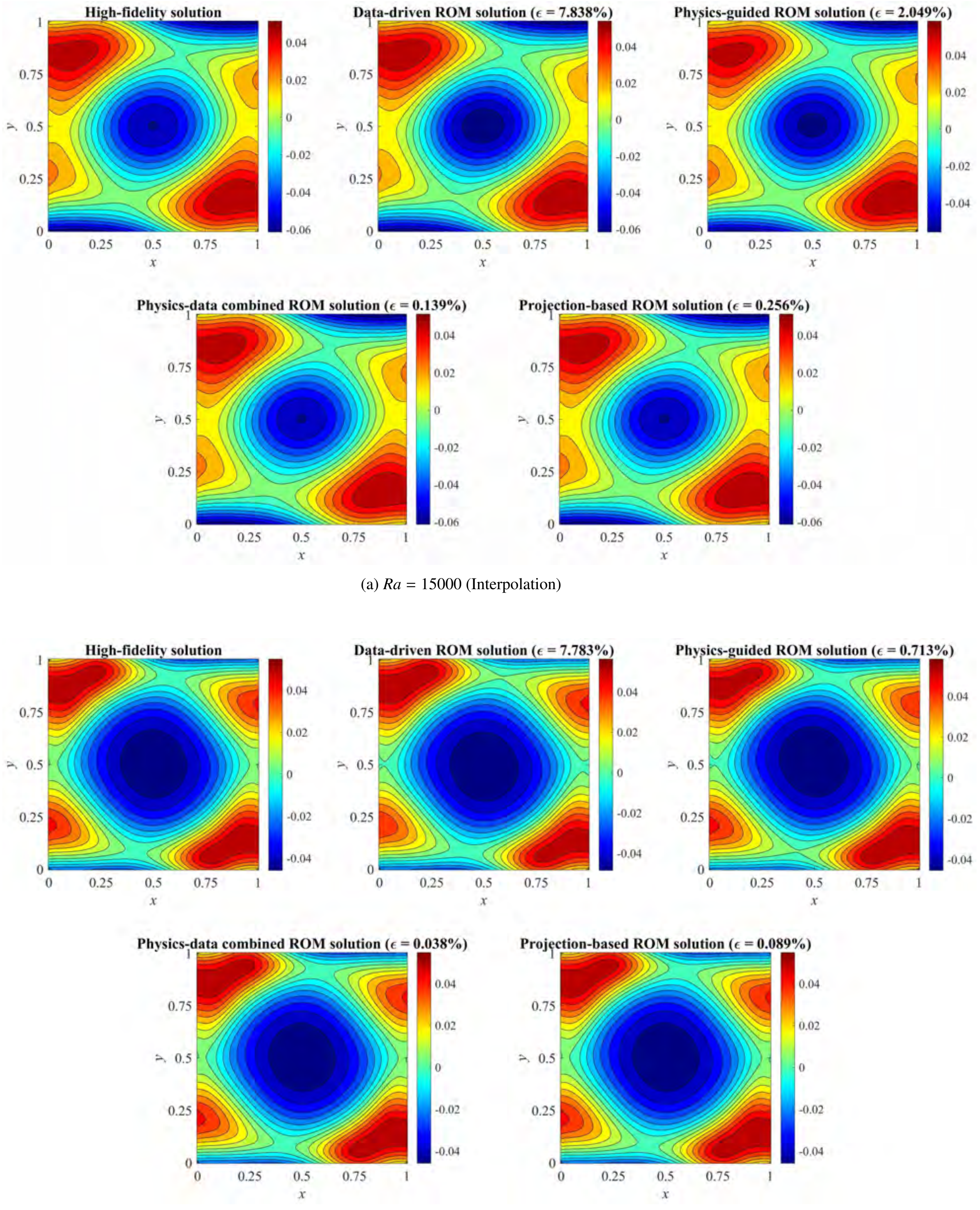
Physics-guided ROM solution (ϵ = 2.171%)

Physics-data combined ROM solution (ϵ = 0.026%)

Projection-based ROM solution (ϵ = 0.015%)

(b) Ra = 250000 (Extrapolation)

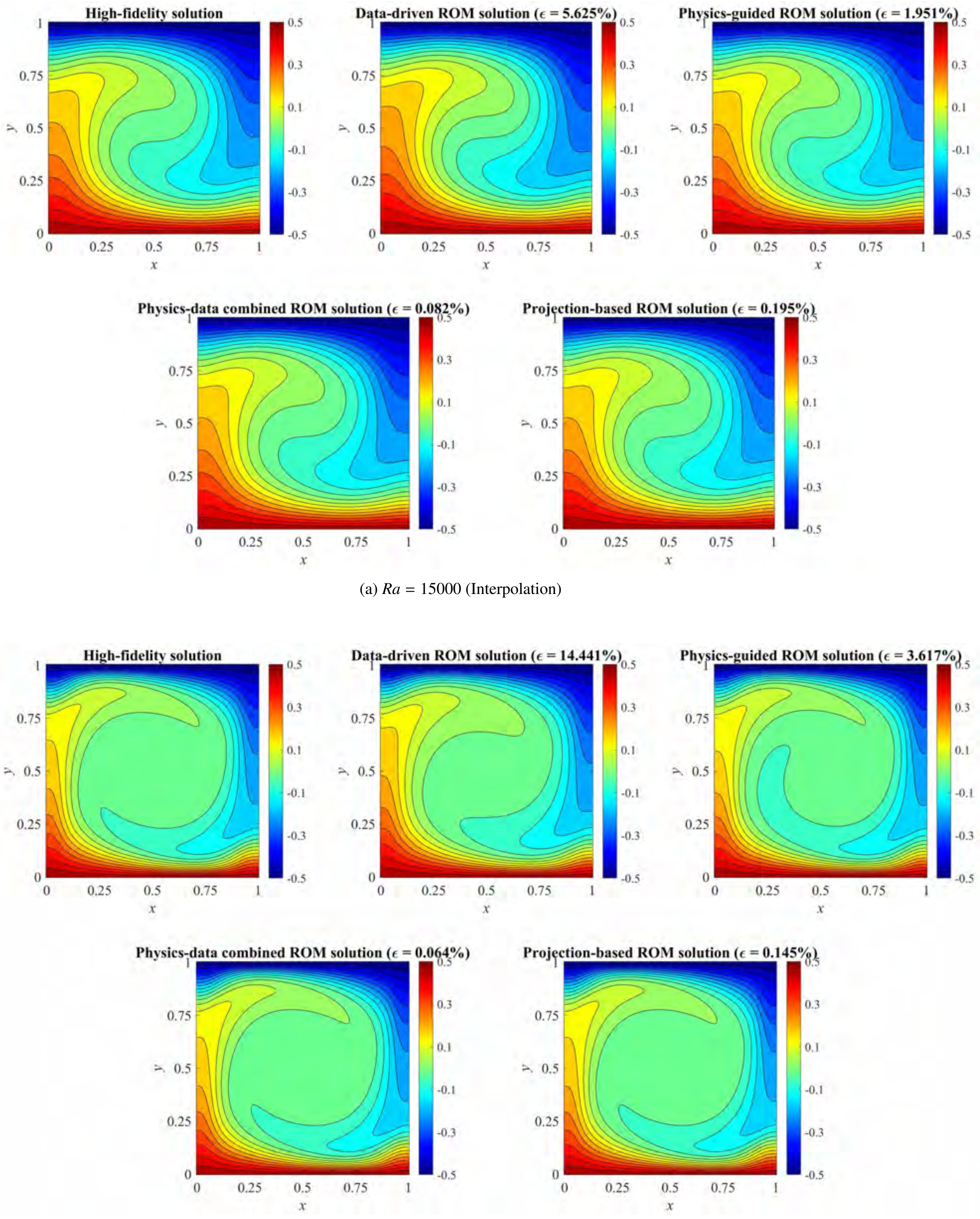
Figure 26: The reduced-order solutions of velocity component v corresponding to different parameters Ra (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



(a) Ra = 15000 (Interpolation)

(b) Ra = 250000 (Extrapolation)

Figure 27: The reduced-order solutions of pressure p corresponding to different parameters Ra (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).



(a) Ra = 15000 (Interpolation)

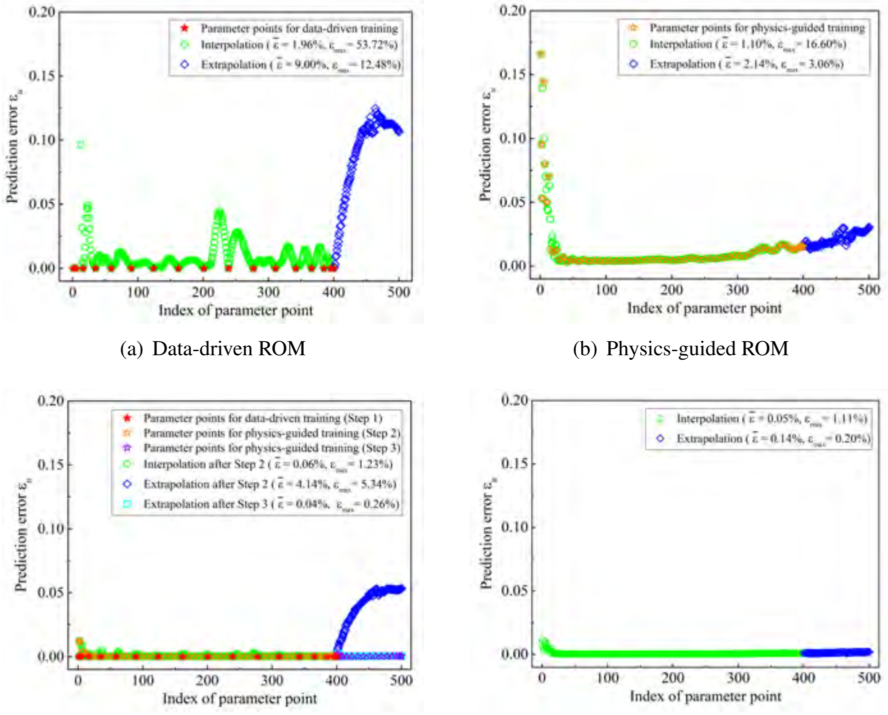
(b) Ra = 250000 (Extrapolation)

Figure 28: The reduced-order solutions of temperature T corresponding to different parameters Ra (the numbers in parentheses are the relative L_2-norm errors ε with respect to the high-fidelity solutions).

The converge curve of the physics-data combined loss function over the entire training process is plotted in Figure 24c, and its fluctuations mainly result from the usage of dynamic weight coefficients. As presented in

Figure 25-28, reduced-order solutions of u, v, p and T corresponding to two representative parameter points are predicted from the physics-data combined FNN model. To quantitatively assess the ROM quality, relative L_2-norm errors of the reduced-order solutions corresponding to all 501 parameter points are calculated according to Eq. (31), as plotted in Figure 29-32. By comparison, the prediction accuracy of the physics-data combined model is much higher than that of the data-driven and the physics-guided models. The average interpolation errors of u, v, p and T are 0.06%, 0.08%, 0.13% and 0.09% respectively, and the average extrapolation errors of u, v, p and T are also greatly reduced to 0.04%, 0.03%, 0.07% and 0.06% respectively. Therefore, this physics-data combined FNN model possesses a strong generalization capacity to provide reliable reduced-order solutions for the entire parameter range.

Furthermore, the traditional projection-based method [10, 11] is applied to construct an intrusive reduced-order model for this steady Rayleigh-B$\acute{e}$nard convection problem by using the same orthogonal bases containing 17 POD modes. More information about this projection-based method and the corresponding open-source code can be found in the relevant reference [10]. As shown in Figure 29, 30, 31 and 30, the projection-based model exhibits excellent performance in predicting reduced-order solutions of u, v, p and T. The relative L_2-norm errors of reduced-order solutions over the entire parameter range are recorded in Figure 29, 30, 31 and 32. This projection-based model has excellent performances in terms of both interpolation and extrapolation predictions, which are comparable to the physical-data combined ROM model. This comparable performance between them further confirms that the physics-data combined ROM method is reliable for the ROM of nonlinear dynamical systems in small-data regimes.



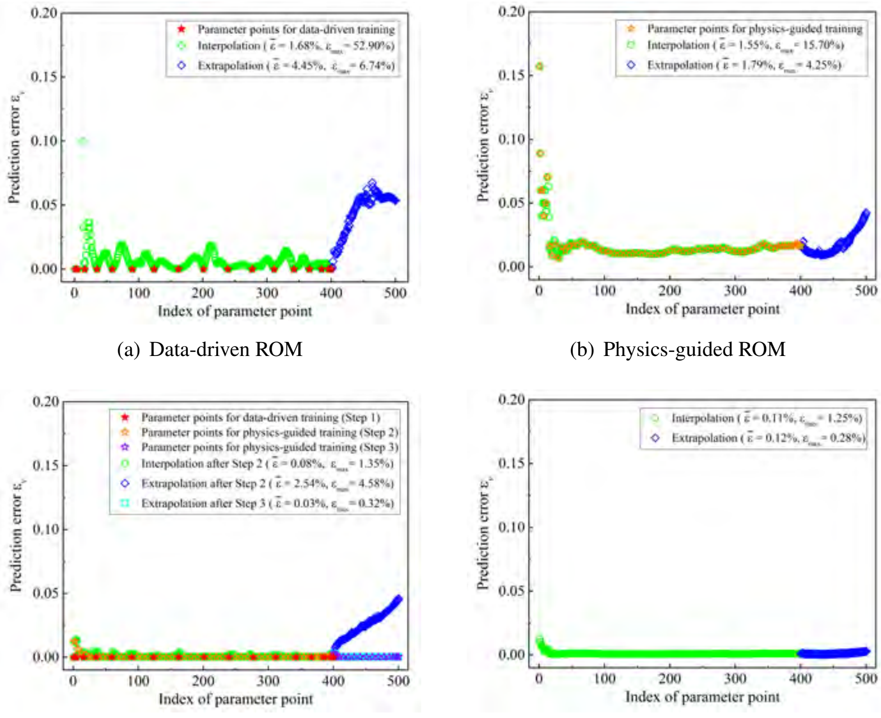
(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 29: The relative L_2-norm errors ε_u of the reduced-order solutions of u estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε_u over a specific parameter range).



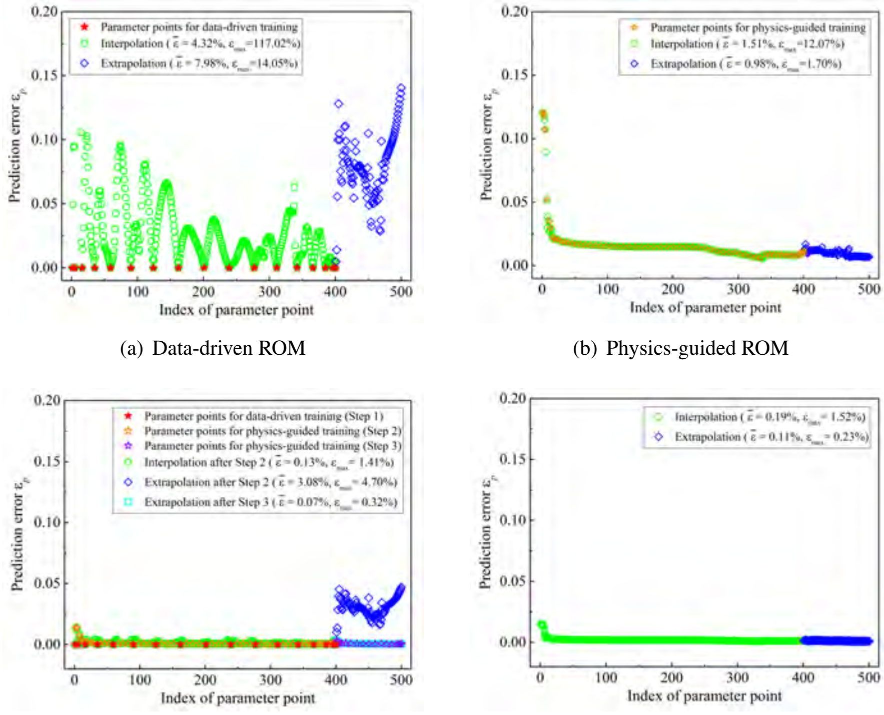
(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 30: The relative L_2-norm errors ε_v of the reduced-order solutions of v estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model.



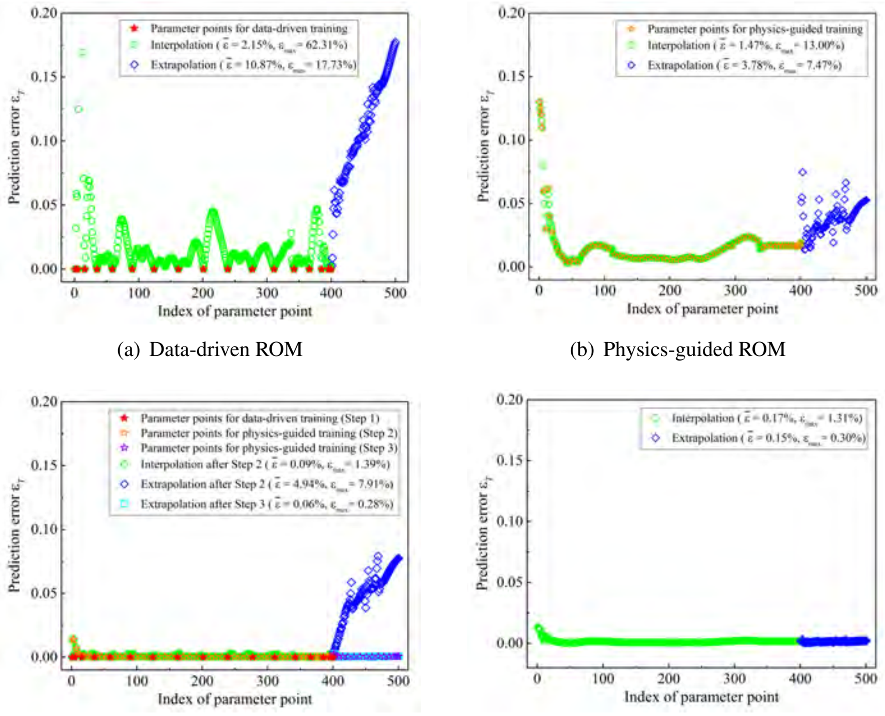
(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 31: The relative L_2-norm errors ε_p of the reduced-order solutions of p estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model.



(a) Data-driven ROM

(b) Physics-guided ROM

(c) Physics-data combined ROM

(d) Projection-based ROM

Figure 32: The relative L_2-norm errors ε_T of the reduced-order solutions of T estimated from the data-driven FNN model, the physics-guided FNN model, the physics-data combined FNN model and the intrusive projection-based model ($\bar{\varepsilon}$ denotes the average of ε_T over a specific parameter range).

5. Discussion

In the above section, a comparative study between the data-driven ROM, the physics-guided ROM, the physics-data combined ROM and the traditional projection-based ROM methods has been carried out in terms of modelling accuracy. The results demonstrate that the proposed physics-data combined ROM method is capable of constructing reliable reduced-order models for nonlinear dynamic systems in small-data regimes. Considering the purpose of model order reduction, efficiency is the other important performance index to evaluate the effectiveness of this new non-intrusive ROM method. In this work, three FNN-based ROM methods, the projection-based ROM method (open-source code) and corresponding high-fidelity numerical simulations are implemented by using MATLAB. Relevant programs are all run on the ASTUTE 2020 HPC cluster in Swansea University (72 nodes, 28-core CPUs per node, and two 32GB GPUs per node). High-fidelity numerical simulations are serially performed on the CPU, while feedforward neural networks are trained on the GPU.

As listed in Table 2, the time costs of high-fidelity computation and model order reduction for each test case are provided. Compared to the projection-based models, the properly trained FNN models can predict the reduced-order solutions of high-dimensional quantities almost instantly, and this advantage will become more pronounced for complicated and large-scale problems. However, it usually requires over a thousand seconds to train a physics-guided or physics-data combined FNN model. Generally, the training speed of neural networks relies heavily on the computing power of a computer, especially the GPU computing capacity. The training process can be easily accelerated ten times even a hundred times via high-performance GPU-based parallel computing [53]. Besides, the proposed physics-guided ROM framework computes partial derivatives through a simple linear combination of precomputed results, as illustrated in Figure 2. Basically, it is computationally cheap to construct reduced-order models via training physics-guided neural networks, which implies that there is still huge room for optimising our implementation code, thereby greatly speeding up the training process.

The popular data-driven ROM method only makes use of the magnitude information of the static correction $\overline{\mathbf{u}}$ and the POD basis $\mathbf{\Phi}$; While the proposed physics-data combined ROM method not only uses such magnitude information, but also takes advantage of the gradient information of $\overline{\mathbf{u}}$ and $\mathbf{\Phi}$, without increasing computational intensity much. Different from the traditional projection-based ROM method, this physics-data combined ROM

Table 2: The computational time costs of reduced-order modelling using different methods

Test case	Domain mesh	Average computational time cost (s)					
		High-fidelity solution	FNN-based ROM (Training & Prediction)				Projection-based ROM
			Data-driven	Physics-guided	Physics-data combined	FNN prediction	Online prediction
Case 1	151×101 $(\mathcal{X} \times \mathcal{T})$	1.013 (FDM)	6.50×10^1	9.750×10^2	1.058×10^3	9.240×10^{-6}	7.250×10^{-2}
Case 2	$51 \times 51 \times 51$ $(\mathcal{X} \times \mathcal{Y} \times \mathcal{T})$	7.306×10^{-3} (Analytical)	4.30×10^1	1.507×10^3	1.136×10^3	1.377×10^{-5}	2.685×10^{-1}
Case 3	131×131 $(\mathcal{X} \times \mathcal{Y})$	5.688×10^4 (FVM)	5.42×10^1	1.265×10^3	1.114×10^3	1.578×10^{-5}	2.410×10^{-1}
Case 4	131×131 $(\mathcal{X} \times \mathcal{Y})$	6.133×10^4 (FVM)	7.69×10^1	1.545×10^3	1.612×10^3	1.850×10^{-5}	3.650×10^{-1}

Note: The FNN models obtained from the data-driven ROM, physics-guided ROM and physics-data combined ROM methods have exactly the same architecture, so the time costs of prediction are also the same for them.

framework fully decouples the online stage from the high-fidelity system operators. From the perspectives of modelling accuracy and construction efficiency, the proposed physics-data combined machine learning (PDCML) method is a cost-effective approach to constructing non-intrusive parametric reduced-order models for nonlinear dynamical systems in small-data regimes.

Just as every coin has two sides, the novel PDCML method is not without limitations: (1) Regular data grid. Finite difference is adopted to approximate partial derivatives from physical fields to evaluate the physics loss of a neural network, which requires regular meshed snapshots. New derivative approximation methods that can be performed on arbitrary data grids should be found or developed to solve this problem. (2) The uniform POD basis. A uniform POD basis is derived for the entire computational domain, which may ignore some important information in complicated sub-regions. The domain decomposition technique can be used to build local POD bases for each subdomain. (3) Approximation errors of partial derivatives. Finite difference struggles to deal with the steep-gradient regions in physical fields, which can bring errors to the computation of physics-guided loss. Local grid refinement of the steep-gradient areas could be an effective way to solve this problem. (4) Parameter sampling. The Smolyak sparse grid method is adopted to select data points from the parameter space. A new sampling strategy should be used to select representative snapshots for the derivation of the POD basis. (5) Determination of FNN architectures. There is no clear relationship between problem complexity and FNN architecture, and the model hyper-parameters of neural networks are determined empirically in this work, which may bring uncertainties to FNN-based ROM. (6) Dynamic weight coefficient. A dynamic weight coefficient (step function) is used to adjust the physics-data proportion in the combined loss function. The parameters in this step function (see Eq. (30)) are determined after many failed trials, so an adaptive weighting function should be developed to solve this problem. The above limitations and unanswered questions are all worth further investigation, so as to improve ROM quality.

6. Conclusions

The main contribution of this work is to present a novel physics-data combined machine learning (PDCML) method for non-intrusive parametric ROM of nonlinear dynamic systems in small-data regimes. To overcome labelled data scarcity, a physics-guided ROM framework is specially developed to embed reduced-order numerical systems into the feedforward neural network (FNN), where the partial derivatives in the governing PDEs are rapidly approxiated through a simple linear combination of precomputed results. This physics-guided ROM framework is computationally cheap and completely label-free. A physics-data combined ROM method is further developed, because labelled data can accelerate the convergence of physics-guided training progress. Minimising the physics-data combined loss function is to solve a multi-objective optimization problem, where dynamic weighting functions are used to adjust the physics-data proportion of the loss function during the step-by-step training scheme. Furthermore, the obtained FNN model can be sustainably evolved to enhance/broaden its learning scope through more steps of physics-data combined training, as long as the POD basis is still applicable.

The proposed PDCML method is tested to construct reduced-order models for a series of nonlinear problems, and it is also compared with the data-driven ROM, the physics-guided ROM methods and the traditional projection-based ROM methods, in terms of modelling accuracy and construction efficiency. All case studies demonstrate that the combination of physics and data not only overcomes the shortage of labelled data, but also forces the physics loss to converge to a better state. The new PDCML method is proved to be a cost-effective approach to building reduced-order models with high prediction accuracy and strong generalization capability, and it is of practical significance for non-intrusive ROM of large-scale dynamical systems, where high-fidelity snapshot data is often very limited and expensive to obtain.

Acknowledgements

The authors would like to acknowledge the support of EPSRC grant: PURIFY ($EP/V000756/1$). Dr. Jinlong Fu acknowledges the enlightening discussion with Dr. Wenqian Chen from Xi'an Jiaotong University. We acknowledge the Fundamental Research Funds for the Central Universities. Miss Rui Fu acknowledges College of Engineering Centenary PhD Scholarship at Swansea University. We acknowledge the support of the Supercomputing Wales project, which is part-funded by the European Regional Development Fund (ERDF) via the Welsh Government.

Appendix I: Finite difference

As a common approach for numerical differentiation, finite difference [40, 41] is adopted to approximate partial derivatives on the regular/uniform data grid in this study. The partial derivatives in the PDEs at each grid point are approximated from neighbouring values by using Taylor's theorem. Consider a function $f(x)$ has the nth-order continuous derivative $f^{(n)}(x)$ at the point x, and thus $f(x+h)$ can be expressed by using the Taylor series expansion:

$$f(x+h) = f(x) + hf'(x) + h^2\frac{f''(x)}{2!} + ... + h^n\frac{f^{(n)}(x)}{(n)!} + O(h^{n+1}) \tag{45}$$

where h is a small finite value, and $O(h^{n+1})$ denotes the higher-order term.
Truncating Eq. (45) after the first derivative term gives,

$$f(x+h) = f(x) + hf'(x) + O(h^2) \tag{46}$$

Rearranging Eq. (46) and neglecting the higher-order term yields,

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \tag{47}$$

Because the truncation error is proportional to h to the power 1, so $\frac{f(x+h)-f(x)}{h}$ is a first-order approximation of the derivative $f'(x)$, and this approximation gets better as h gets smaller.

If $h > 0$, say $h = \Delta x$ where Δx is the grid spacing distance, then

$$f'(x) \approx \frac{f(x+\Delta x) - f(x)}{\Delta x} \tag{48}$$

is called the first-order *forward difference* approximation of $f'(x)$. If $h < 0$, say $h = -\Delta x$, then

$$f'(x) \approx \frac{f(x) - f(x-\Delta x)}{\Delta x} \tag{49}$$

is called the first-order *backward difference* approximation of $f'(x)$. Combining the Eq. (48) and (49) gives

$$f'(x) \approx \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} \tag{50}$$

where $\frac{f(x+\Delta x)-f(x-\Delta x)}{2\Delta x}$ is called *central difference*, and it is a second-order approximation of $f'(x)$.

If more terms in the Taylor expansion series (see Eq. (45)) are used, high-order approximation of $f'(x)$ can be derived. In this study, only central difference is used to approximate the partial derivatives in PDEs, so the second-, fourth- and sixth-order formulas for approximating $f'(x)$ and $f''(x)$ are provided here.

Second-order central difference approximations:

$$f''(x) \approx \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2} \tag{51}$$

Fourth-order central difference approximations:

$$f'(x) \approx \frac{-f(x+2\Delta x) + 8f(x+\Delta x) - 8f(x-\Delta x) + 1f(x-2\Delta x)}{12\Delta x} \tag{52}$$

$$f''(x) \approx \frac{-f(x+2\Delta x) + 16f(x+\Delta x) - 30f(x) + 16f(x-\Delta x) - f(x-2\Delta x)}{12\Delta x^2} \tag{53}$$

Sixth-order central difference approximations:

$$f'(x) \approx \frac{f(x+3\Delta x) - 9f(x+2\Delta x) + 45f(x+\Delta x) - 45f(x-\Delta x) + 9f(x-2\Delta x) - f(x-3\Delta x)}{60\Delta x} \tag{54}$$

$$f''(x) \approx \frac{2f(x+3\Delta x) - 27f(x+2\Delta x) + 270f(x+\Delta x) - 490f(x) + 270f(x-\Delta x) - 27f(x-2\Delta x) + 2f(x-3\Delta x)}{180\Delta x^2} \tag{55}$$

Appendix II: Smolyak sparse grid

Smolyak sparse grid method [1, 2, 43] is adopted to select a set of parameter points over a specific parameter space ($\mathcal{P} \in \mathbb{R}^P$) in this study. It is a numerical technique to tackle the problem of 'curse of dimensionality' by integrating high-dimensional functions based on a univariate quadrature rule. The core idea is to select a small number of nodes on the full tensor-product grid according to the potential importance of these nodes, aiming to significantly improve the computational efficiency. It means that only a small number of Smolyak nodes are involved in computation instead of all the nodes on the full tensor-product grid. The number of selected Smolyak nodes is controlled by the approximation level. The 1-dimensional (1D) Smolyak integral $Q_l^{(1)}$ of a function f can be expressed as follows:

$$Q_l^{(1)} f = \sum_{i=1}^{N_l} f\left(\boldsymbol{\mu}_l^i\right) \eta_l^i \tag{56}$$

where l denotes the approximation level of sparse grid, f is a function within the interval [0, 1], N_l is the number of selected Smolyak points, η_l^i denotes the weight corresponding to the Smolyak point $\boldsymbol{\mu}_l^i$, and i is the index of a Smolyak point.

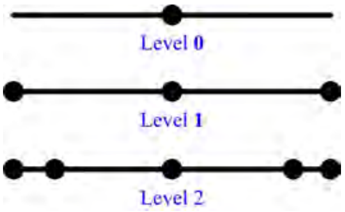


Figure 33: Schematic illustration the selected points in the 1D Smolyak grid with the approximation level 0, level 1 and level 2.

The higher the approximation level l is, the more Smolyak nodes will be chosen and higher approximation quality will be achieved. For 1D Smolyak sparse grid, the relationship between the number N_l of selected parameter points and the approximation level l is given by:

$$N_l = 2^l + 1 \quad (l = 0,\ 1,\ 2\ldots) \tag{57}$$

As graphically illustrated in Figure 33, the selected points in the 1D Smolyak grid with the approximation level 0, level 1 and level 2 are separately plotted.

References

[1] D Xiao, F Fang, CC Pain, and IM Navon. A parameterized non-intrusive reduced order model and error analysis for general time-dependent nonlinear partial differential equations and its applications. *Computer Methods in Applied Mechanics and Engineering*, 317:868–889, 2017.

[2] Jian Yu, Chao Yan, and Mengwu Guo. Non-intrusive reduced-order modeling for fluid problems: A brief review. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(16):5896–5912, 2019.

[3] Luning Sun, Han Gao, Shaowu Pan, and Jian-Xun Wang. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. *Computer Methods in Applied Mechanics and Engineering*, 361:112732, 2020.

[4] Wenqian Chen, Qian Wang, Jan S Hesthaven, and Chuhua Zhang. Physics-informed machine learning for reduced-order modeling of nonlinear problems. *Journal of Computational Physics*, 446:110666, 2021.

[5] Jinlong Fu, Jiabin Dong, Yongliang Wang, Yang Ju, D Roger J Owen, and Chenfeng Li. Resolution effect: An error correction model for intrinsic permeability of porous media estimated from lattice boltzmann method. *Transport in Porous Media*, 132(3):627–656, 2020.

[6] Jinlong Fu, Hywel R Thomas, and Chenfeng Li. Tortuosity of porous media: Image analysis and physical simulation. *Earth-Science Reviews*, 212:103439, 2021.

[7] Wing Kam Liu, Shaofan Li, and Harold Park. Eighty years of the finite element method: Birth, evolution, and future. *arXiv preprint arXiv:2107.04960*, 2021.

[8] Renee Swischuk, Laura Mainini, Benjamin Peherstorfer, and Karen Willcox. Projection-based model reduction: Formulations for physics-based machine learning. *Computers & Fluids*, 179:704–717, 2019.

[9] Peter Benner, Mario Ohlberger, Albert Cohen, and Karen Willcox. *Model reduction and approximation: theory and algorithms*. SIAM, 2017.

[10] Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced basis methods for partial differential equations: an introduction*, volume 92. Springer, 2015.

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

[12] Davide Papapicco, Nicola Demo, Michele Girfoglio, Giovanni Stabile, and Gianluigi Rozza. The neural network shifted-proper orthogonal decomposition: A machine learning approach for non-linear reduction of hyperbolic equations. *Computer Methods in Applied Mechanics and Engineering*, 392:114687, 2022.

[13] D Xiao, CE Heaney, F Fang, L Mottet, R Hu, DA Bistrian, E Aristodemou, IM Navon, and CC Pain. A domain decomposition non-intrusive reduced order model for turbulent flows. *Computers & Fluids*, 182:15–27, 2019.

[14] Diana Alina Bistrian and Ionel Michael Navon. Randomized dynamic mode decomposition for nonintrusive reduced order modelling. *International Journal for Numerical Methods in Engineering*, 112(1):3–25, 2017.

[15] Mathilde Chevreuil and Anthony Nouy. Model order reduction based on proper generalized decomposition for the propagation of uncertainties in structural dynamics. *International Journal for Numerical Methods in Engineering*, 89(2):241–268, 2012.

[16] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[17] Takaaki Murata, Kai Fukami, and Koji Fukagata. Nonlinear mode decomposition with convolutional neural networks for fluid dynamics. *Journal of Fluid Mechanics*, 882, 2020.

[18] R Fu, D Xiao, IM Navon, and C Wang. A data driven reduced order model of fluid flow by auto-encoder and self-attention deep learning methods. *arXiv preprint arXiv:2109.02126*, 2021.

[19] Rachel Cooper, Andrey A Popov, and Adrian Sandu. Investigation of nonlinear model order reduction of the quasigeostrophic equations through a physics-informed convolutional autoencoder. *arXiv preprint arXiv:2108.12344*, 2021.

[20] Dunhui Xiao. Error estimation of the parametric non-intrusive reduced order model using machine learning. *Computer Methods in Applied Mechanics and Engineering*, 355:513–534, 2019.

[21] Youngkyu Kim, Youngsoo Choi, David Widemann, and Tarek Zohdi. A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder. *arXiv preprint arXiv:2009.11990*, 2020.

[22] Sangseung Lee and Donghyun You. Data-driven prediction of unsteady flow over a circular cylinder using deep learning. *Journal of Fluid Mechanics*, 879:217–254, 2019.

[23] Muhammad Mohebujjaman, Leo G Rebholz, and Traian Iliescu. Physically constrained data-driven correction for reduced-order modeling of fluid flows. *International Journal for Numerical Methods in Fluids*, 89(3):103–122, 2019.

[24] Jiayang Xu and Karthik Duraisamy. Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics. *Computer Methods in Applied Mechanics and Engineering*, 372:113379, 2020.

[25] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[26] Dunhui Xiao, CE Heaney, L Mottet, F Fang, W Lin, IM Navon, Y Guo, OK Matar, AG Robins, and CC Pain. A reduced order model for turbulent flows in the urban environment using machine learning. *Building and Environment*, 148:323–337, 2019.

[27] D Xiao, F Fang, J Zheng, CC Pain, and IM Navon. Machine learning-based rapid response tools for regional air pollution modelling. *Atmospheric Environment*, 199:463–473, 2019.

[28] Suraj Pawar, Omer San, Aditya Nair, Adil Rasheed, and Trond Kvamsdal. Model fusion with physics-guided machine learning: Projection-based reduced-order modeling. *Physics of Fluids*, 33(6):067123, 2021.

[29] Maziar Raissi, Paris Perdikaris, and George 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*, 378:686–707, 2019.

[30] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[31] Ziqi Liu, Wei Cai, and Zhi-Qin John Xu. Multi-scale deep neural network (mscalednn) for solving poisson-boltzmann equation in complex domains. *arXiv preprint arXiv:2007.11207*, 2020.

[32] Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, page 110768, 2021.

[33] Sourav Saha, Zhengtao Gan, Lin Cheng, Jiaying Gao, Orion L Kafka, Xiaoyu Xie, Hengyang Li, Mahsa Tajdari, H Alicia Kim, and Wing Kam Liu. Hierarchical deep learning neural network (hidenn): An artificial intelligence (ai) framework for computational science and engineering. *Computer Methods in Applied Mechanics and Engineering*, 373:113452, 2021.

[34] Stein KF Stoter, Bernardo Cockburn, Thomas JR Hughes, and Dominik Schillinger. Discontinuous galerkin methods through the lens of variational multiscale analysis. *Computer Methods in Applied Mechanics and Engineering*, 388:114220, 2022.

[35] DM Causon and CG Mingham. *Introductory finite difference methods for PDEs*. Bookboon, 2010.

[36] George Stefanou, Dimitrios Savvas, and Manolis Papadrakakis. Stochastic finite element analysis of composite structures based on mesoscale random fields of material properties. *Computer Methods in Applied Mechanics and Engineering*, 326:319–337, 2017.

[37] Kenneth C Hall, Jeffrey P Thomas, and Earl H Dowell. Proper orthogonal decomposition technique for transonic unsteady aerodynamic flows. *AIAA journal*, 38(10):1853–1862, 2000.

[38] Bayya Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[39] Rastko R Selmic and Frank L Lewis. Neural-network approximation of piecewise continuous functions: application to friction compensation. *IEEE transactions on neural networks*, 13(3):745–751, 2002.

[40] Bengt Fornberg. Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of computation*, 51(184):699–706, 1988.

[41] John C Strikwerda. *Finite difference schemes and partial differential equations*. SIAM, 2004.

[42] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[43] Winfried Sickel and Tino Ullrich. *Smolyak's algorithm, sampling on sparse grids and function spaces of dominating mixed smoothness*. 2006.

[44] Cea Basdevant, M Deville, P Haldenwang, JM Lacroix, J Ouazzani, R Peyret, Paolo Orlandi, and AT Patera. Spectral and finite difference solutions of the burgers equation. *Computers & fluids*, 14(1):23–41, 1986.

[45] Liu Jianchun, Gary A Pope, and Kamy Sepehrnoori. A high-resolution finite-difference scheme for nonuniform grids. *Applied mathematical modelling*, 19(3):162–172, 1995.

[46] Lin Zhang, Jie Ouyang, Xiaoxia Wang, and Xiaohua Zhang. Variational multiscale element-free galerkin method for 2d burgers' equation. *Journal of computational physics*, 229(19):7147–7161, 2010.

[47] Jafar Biazar and Hossein Aminikhah. Exact and numerical solutions for non-linear burger's equation by vim. *Mathematical and Computer Modelling*, 49(7-8):1394–1400, 2009.

[48] Charles-Henri Bruneau and Mazen Saad. The 2d lid-driven cavity problem revisited. *Computers & fluids*, 35(3):326–348, 2006.

[49] Tamer A AbdelMigid, Khalid M Saqr, Mohamed A Kotb, and Ahmed A Aboelfarag. Revisiting the lid-driven cavity flow problem: Review and new steady state benchmarking results using gpu accelerated code. *Alexandria engineering journal*, 56(1):123–135, 2017.

[50] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.

[51] Nasreddine Ouertatani, Nader Ben Cheikh, Brahim Ben Beya, and Taieb Lili. Numerical simulation of two-dimensional rayleigh–bénard convection in an enclosure. *Comptes Rendus Mécanique*, 336(5):464–470, 2008.

[52] Baole Wen, David Goluskin, Matthew LeDuc, Gregory P Chini, and Charles R Doering. Steady rayleigh–bénard convection between stress-free boundaries. *Journal of Fluid Mechanics*, 905, 2020.

[53] Daniel Strigl, Klaus Kofler, and Stefan Podlipnig. Performance and scalability of gpu-based convolutional neural networks. In *2010 18th Euromicro conference on parallel, distributed and network-based processing*, pages 317–324. IEEE, 2010.