# Nonlinear model order reduction of engineering turbulence using data-assisted neural networks

Chuanhua Zhu[a,c], Jinlong Fu[b,c,*], Dunhui Xiao[d], Jinsheng Wang[e]

[a]*School of Environment and Energy Engineering, Anhui Jianzhu University, Hefei, 230601, China*
[b]*School of Engineering and Materials Science, Faculty of Science and Engineering, Queen Mary University of London, London, E1 4NS, UK*
[c]*Zienkiewicz Centre for Modelling, Data and AI, Faculty of Science and Engineering, Swansea University, Swansea, SA1 8EN, UK*
[d]*School of Mathematical Sciences, Key Laboratory of Intelligent Computing and Applications (Ministry of Education), Tongji University, Shanghai, 200092, China*
[e]*School of Civil Engineering, University of Birmingham, Birmingham, B15 2TT, UK*

**Abstract**

Conducting repeated high-fidelity simulations of complex turbulent flows entails substantial computational costs in engineering applications. Reduced-order modeling (ROM) seeks to derive low-dimensional representations from full-order numerical systems, thereby facilitating rapid forecasting of future flow states. This study presents a novel data-assisted computational framework that employs deep neural networks for nonlinear ROM of engineering turbulent flows. Specifically, the Stacked Auto-Encoder (SAE) network is utilized for nonlinear dimensionality reduction and feature extraction; the resulting latent features subsequently serve as inputs to the Long Short-Term Memory (LSTM) network for predictive ROM of turbulent fluid dynamics. A comparative analysis is conducted between SAE and proper orthogonal decomposition regarding dimensionality reduction, and the performance of LSTM in time-series forecasting is also evaluated against dynamic mode decomposition, where two different training strategies are applied for LSTM within the reduced-order latent space. The proposed SAE-LSTM-based ROM approach is tested on two typical turbulent flow problems for non-intrusive model order reduction. The results demonstrate that the constructed surrogate models possess significant capability in predicting the evolution of turbulent flows by preserving essential nonlinear characteristics inherent in fluid dynamics. This innovative method shows great promise in addressing computational challenges associated with high-resolution numerical modeling applied to complex large-scale flow problems.

*Keywords:* Reduced-order modelling; Turbulent flows; Computational fluid dynamics; Stacked auto-encoder; Long short-term memory; Non-intrusive

## 1. Introduction

Due to the computational complexity and large scale of engineering fluid systems, predicting unsteady and turbulent flow behaviors through high-fidelity computational fluid dynamics (CFD) remains a formidable challenge [1, 2, 3]. The high dimensionality, non-linearity, and chaotic nature of turbulent fluid flows necessitate the development of rapid models capable of accurately extrapolating evolutionary behaviors beyond the observation range [4, 5, 6]. Reduced-order modeling (ROM) [7, 8, 9] is a technique designed to significantly reduce the computational burden associated with high-resolution numerical simulations. This approach allows for low-dimensional approximations to be derived from full-order numerical models while preserving essential flow characteristics. These reduced-order models hold substantial practical significance for fast and nearly real-time predictions of high-dimensional quantities such as velocity components, pressure, and temperature fields.

Nevertheless, traditional intrusive reduced-order modeling (ROM) methods, such as those based on proper orthogonal decomposition (POD) or Galerkin projection, often encounter challenges in long-term predictions and struggle to capture complex dynamics in out-of-sample scenarios [10, 11, 12, 13]. These approaches typically construct reduced-order models in an intrusive manner, necessitating access to the source codes of full-order systems or numerical solvers. This requirement significantly restricts their applicability in many contexts [14, 15, 16].

*Corresponding author
*Email address:* jinlong.fu@qmul.ac.uk (Jinlong Fu)

Given the inherent connection between non-intrusive ROM and artificial intelligence (AI), various machine learning and deep learning algorithms [9, 17, 18, 19, 20, 21, 22, 23] have been integrated into model order reduction. These methodologies leverage AI's robust capabilities for complex data analysis and hidden rule exploration. Recent advancements [24, 25, 26, 27, 28, 29] in deep learning demonstrate considerable promise in addressing the aforementioned limitations—particularly through the integration of Stacked Auto-Encoders and Long Short-Term Memory networks for non-intrusive ROM and enhanced extrapolation predictions of turbulent fluid dynamics.

*Stacked Auto-Encoder* (SAE) [30], which involves the sequential stacking of multiple autoencoder layers, is particularly effective in learning hierarchical and compressed representations of high-dimensional data. SAE models can be trained to encode the complex spatial structures of fluid flows into low-dimensional latent representations while preserving the essential features necessary for accurate reconstruction and prediction [31]. Dimensionality reduction is crucial for ROM, as it facilitates a more efficient representation of flow dynamics, thereby making subsequent forecasting computationally feasible. The effectiveness of autoencoders in fluid dynamics has been demonstrated across various studies. For instance, Gonzalez and Balajewicz [32] employed deep convolutional recurrent autoencoders to capture the low-dimensional feature dynamics of fluid systems, achieving significant improvements in modeling unsteady flows. Similarly, Mardt et al. [33] introduced a deep learning approach that integrates convolutional neural networks (CNNs) with autoencoders to capture the nonlinear dynamics inherent in fluid flows, illustrating how these models can surpass traditional methods in both accuracy and computational efficiency. Zhu et al. [34] employed SAE to extract low-dimensional latent features from unsteady/turbulent flows, and the extracted features were then utilized as inputs for constructing a predictive reduced-order model through Dynamic Mode Decomposition (DMD). Furthermore, Fu et al. [35] combined self-attention deep learning techniques with SAE to develop non-intrusive reduced-order models for complex fluid flows while effectively preserving nonlinear features.

*Long Short-Term Memory* (LSTM) [36, 37], a specialized type of recurrent neural network (RNN), is designed to analyze time-series data and effectively capture long-term dependencies. This capability is essential for predicting fluid dynamics, where the temporal evolution of flow is influenced by past states over extended periods. The memory cells within LSTM networks allow the model to retain information across long sequences, which proves particularly advantageous when addressing fluid flows that exhibit temporal correlations across multiple scales. For example, Karniadakis et al. [38] investigated the application of LSTM networks in modeling turbulent flows and demonstrated their proficiency in capturing long-range temporal dependencies—an aspect crucial for accurate extrapolation. Moreover, several studies have underscored the potential benefits of integrating LSTM with autoencoder-based models to enhance predictive performance further. Wiewel et al. [39] introduced a model that employed latent space physics alongside LSTM networks to predict the temporal evolution of fluid flows, illustrating that their approach could significantly improve extrapolation accuracy compared to traditional methods. Vlachas et al. [40] developed a framework that combined convolutional autoencoders with LSTM networks for predicting high-dimensional spatial-temporal chaotic systems, demonstrating that their model could generalize effectively to previously unseen flow regimes. Furthermore, Solera-Rico et al. [21] explored this combination and showed that $\beta$-variational autoencoders ($\beta$-VAEs) integrated with transformers could substantially enhance both extrapolation and prediction capabilities of reduced-order models by balancing reconstruction fidelity with latent space regularization.

Besides, the application of AI in model order reduction extends beyond academic research and is actively being investigated for practical engineering applications. For instance, LSTM-based models have been effectively utilized for early anomaly detection in industrial systems such as induction heating processes, where precise and timely predictions are crucial for preventive maintenance [41]. The integration of autoencoders with LSTM networks has demonstrated significant potential in enhancing the accuracy of nonlinear model order reduction for unsteady flows, which is vital for real-time control and optimization within engineering systems [37]. Additionally, Zhu et al. [34] advanced a data-driven computational framework by adeptly combining SAE with DMD, successfully applying this approach to non-intrusive ROM of turbulent flows around bridge piers.

The aforementioned studies illustrate that a strategic integration of deep learning architectures can yield more robust and generalizable models for fluid dynamics. The combination of SAE and LSTM networks holds significant promise in establishing a powerful framework for the extrapolative prediction of turbulent fluid dynamics. Initially, the SAE network compresses high-dimensional fluid data into low-dimensional latent features, effectively capturing the spatial structures inherent in the flow. Subsequently, the LSTM network operates on these latent representations to learn temporal dynamics, thereby enabling the model to predict future states of the fluid system with enhanced accuracy and stability, even beyond the range of training data.

This study aims to further explore and validate the extrapolation prediction capabilities of the combined SAE-LSTM architecture in the context of reduced-order modeling for turbulent fluid dynamics. By leveraging the

hierarchical feature extraction of SAE and the temporal learning capabilities of LSTM, we seek to develop a robust data-assisted ROM framework that not only improves predictive accuracy within the training range, but also extends the model's ability to accurately forecast fluid behavior in unobserved conditions, thereby broadening the applicability of reduced-order models to more complex and turbulent fluid flows. The remainder of this paper is structured as follows: In Section 2, the full-order numerical model and the governing equations of turbulent fluid flows are briefly introduced; Nonlinear dimensionality reduction of high-dimension snapshots using SAE is detailedly described in Section 3; In Section 4, the intricate methodology underpinning the data-assisted SAE-LSTM framework for predictive ROM is elucidated; Numerical investigations with two typical cases are carried out in Section 5, aiming to validate the effectiveness of the newly proposed SAE-LSTM-based ROM method; Finally, the key findings and main contributions of this work are discussed and summarized in Section 6.

## 2. Full-order numerical model

The evolutionary behavior of a nonlinear dynamical system is governed by its fundamental physical laws, which are frequently represented in the form of partial differential equations (PDEs). In this context, the fluid dynamics of turbulent and unsteady flows can be characterized by the incompressible Navier-Stokes (N-S) equations, which mathematically encapsulate mass conservation and momentum balance for Newtonian fluids:

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v}\right) = -\nabla p + \mu\Delta\mathbf{v}\,, \tag{1}$$

$$\nabla\cdot\mathbf{v} = 0\,, \tag{2}$$

where $\mathbf{v} \equiv (v_x, v_y, v_z)^{\mathrm{T}}$ denotes the flow velocity components in $x$-, $y$- and $z$-directions, $p$ is the fluid pressure, $t$ denotes time, $\rho$ denotes the constant fluid density, and $\mu$ represents the dynamic viscosity.

Analytically solving the aforementioned N-S equations is exceedingly challenging and often infeasible in most scenarios. The conventional approach involves approximating the solutions of these nonlinear PDEs through numerical methods [42], such as the finite element method. In this process, the N-S equations are discretized and reformulated into a solvable system of algebraic equations that effectively characterize the nonlinear fluid dynamics. The full-order numerical model representing this nonlinear dynamical system can generally be expressed as follows:

$$\mathcal{M}\frac{\partial \boldsymbol{v}}{\partial t} + \mathcal{A}(\boldsymbol{v})\boldsymbol{v} + \mathcal{K}(\mu)\boldsymbol{v} + C\boldsymbol{p} = \mathbf{0}, \tag{3}$$

$$\mathcal{D}(\boldsymbol{v}) = \mathbf{0}, \tag{4}$$

where $\mathcal{M}$ signifies the mass matrix, $\mathcal{A}(\cdot)$ denotes the solution-dependent streaming operator, and $\mathcal{K}(\mu)$ refers to the matrix associated with the remaining linear velocity terms; The vector $\boldsymbol{v}$ encompasses the nodal values of all three velocity components; $C$ is identified as a pressure gradient matrix; $\boldsymbol{p}$ constitutes a vector containing the nodal values of pressure; $\mathcal{D}(\cdot)$ represents the divergence operator.

To achieve precise numerical approximations, continuous physical fields are typically discretized into fine meshes comprising a large number of elements, each possessing a specific number of degrees of freedom. Consequently, the full-order numerical model that incorporates the N-S equations is generally characterized by high dimensionality. This implies that the matrices and variable vectors in Eqs (3) and (4) are substantial in size. Solving such a large-scale system of equations with an extensive number of unknowns can be extremely computationally intensive. Therefore, model order reduction techniques hold significant practical value in uncovering latent low-dimensional representations of full-order numerical models, thereby facilitating efficient resolution of the primary fluid dynamics within nonlinear systems.

## 3. Nonlinear dimensionality reduction

The full-order numerical model characterizes the nonlinear fluid dynamical system that maps spatial, temporal, and system parameters—such as fluid viscosity, Reynolds number, initial conditions, and boundary conditions—onto physical field variables including velocity components and pressure. The solutions for these physical field variables, denoted by $\mathbf{u}(\mathbf{x},\ t;\ \boldsymbol{\mu})$, are functions of space $\mathbf{x} \in \mathcal{X}$ and time $t \in \mathcal{T}$ while also depending on the system parameters $\boldsymbol{\mu} \in \mathcal{P}$. This relationship can be expressed through the following function:

$$\mathbf{u}(\mathbf{x},\ t;\ \boldsymbol{\mu}):\ \ \mathcal{X}\times\mathcal{T}\times\mathcal{P}\rightarrow\mathbb{R}, \tag{5}$$

where $\mathcal{X}$, $\mathcal{T}$, and $\mathcal{P}$ represent the domains of space, time, and parameters respectively.

This study aims to derive reduced-order approximate models of $\mathbf{u}(\mathbf{x},\ t;\ \boldsymbol{\mu})$ from high-fidelity numerical solutions. In this context, stacked autoencoder (SAE) is employed to perform nonlinear dimensionality reduction. The high-fidelity dynamical system is projected onto a nonlinear latent subspace while preserving the essential dynamical characteristics.

### *3.1. Data preparation*

Consider a collection of high-fidelity numerical solutions for the physical field variable $\mathbf{u}(\cdot\ ,\ t;\ \boldsymbol{\mu})$ at various time points $t = \{t_1,\ t_2,\ ...,\ t_i,\ ...,\ t_n\} \in \mathcal{T}$, where $t_i = T_0 + i\Delta t$, and $T_0$ represents the initial time point. Each numerical solution $\mathbf{u}(\cdot\ ,\ t_i;\ \boldsymbol{\mu})$ is referred to as a "snapshot", which can be reshaped into a column vector denoted by $\boldsymbol{u}_i \in \mathbb{R}^{\mathcal{N}_x}$. A dataset $\boldsymbol{U}$ is constructed by aggregating all column vectors:

$$\boldsymbol{U} = \Big\{\boldsymbol{u}_1,\ \boldsymbol{u}_2,\ ...,\ \boldsymbol{u}_i,\ ...,\ \boldsymbol{u}_n\Big\} \in \mathbb{R}^{\mathcal{N}_x \times n},\ (1 \le i \le n), \tag{6}$$

where $\mathcal{N}_x$ denotes the dimension of the numerical discretization of the spatial domain.

Before conducting data dimensionality reduction, it is essential to standardize the snapshot data by eliminating the fixed component $\overline{\boldsymbol{u}}$, referred to as the "particular solution" or "static correction." This process yields a standardized dataset $\widetilde{\boldsymbol{U}}$:

$$\begin{aligned}\widetilde{\boldsymbol{U}} &= \Big\{\boldsymbol{u}_1 - \overline{\boldsymbol{u}},\ \boldsymbol{u}_2 - \overline{\boldsymbol{u}},\ ...,\ \boldsymbol{u}_i - \overline{\boldsymbol{u}},\ ...,\ \boldsymbol{u}_n - \overline{\boldsymbol{u}}\Big\} \\ &= \Big\{\widetilde{\boldsymbol{u}}_1,\ \widetilde{\boldsymbol{u}}_2,\ ...,\ \widetilde{\boldsymbol{u}}_i,\ ...,\ \widetilde{\boldsymbol{u}}_n\Big\} \in \mathbb{R}^{\mathcal{N}_x \times n},\end{aligned} \tag{7}$$

where $\overline{\boldsymbol{u}}$ is calculated as the mean of all column vectors reshaped from the snapshots, as noted in Hall et al. [43].

$$\overline{\boldsymbol{u}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{u}_i, \tag{8}$$

It is important to consider that this mean value is independent of the time dependence inherent in fluid dynamics.

### *3.2. Stacked auto-encoder for nonlinear dimensionality reduction*

High-fidelity snapshots typically exhibit high data dimensionality due to the fine mesh grid utilized in full-order numerical simulation models. In this study, we employ *auto-encoder* [44, 30] for data compression by learning nonlinear latent representations from these high-dimensional snapshots. Auto-encoder is a type of unsupervised artificial neural network that consists of two main components: an encoder and a decoder. The encoder compresses the input into a lower-dimensional code, while the decoder reconstructs the output from this latent space representation. The primary objective of an autoencoder is to minimize the discrepancy between the original input and its reconstructed output. Multiple basic autoencoders can be stacked together to form a hierarchical deep model, where the output of each individual autoencoder serves as the input for subsequent layers. This configuration, known as a *Stacked Auto-encoder*, enables the learning of more complex representations from raw inputs, allowing for hierarchical features to be compressed, organized, and extracted effectively. A graphical illustration of a SAE model composed of multiple layers is presented in Figure 1, wherein each encoder/decoder layer's output feeds into its corresponding successive layer.

The encoder and decoder serve as the fundamental components of a SAE model, with their mathematical formulations detailed in the subsequent sections. Let $\boldsymbol{\alpha}^{(l)} = \left(\alpha_1^{(l)},\ \alpha_2^{(l)},\ \ldots,\ \alpha_{m_l}^{(l)}\right)^{\mathrm{T}}$ represent the latent feature vector acquired by the $l$-th encoder layer within the SAE framework, where $m_l$ indicates the total number of hidden units in this particular encoder layer. The $l$-th encoder $\phi_{\check{\theta}}^{(l)}(\cdot)$ compresses its input $\boldsymbol{\alpha}^{(l-1)}$ into the hidden representation denoted as $\boldsymbol{\alpha}^{(l)}$, governed by the following equation:

$$\boldsymbol{\alpha}^{(l)} = \phi_{\check{\theta}}^{(l)}\Big(\boldsymbol{\alpha}^{(l-1)}\Big) = g\Big(\boldsymbol{W}_{\mathrm{e}}^{(l)}\boldsymbol{\alpha}^{(l-1)} + b_{\mathrm{e}}^{(l)}\Big),\quad (l = 1, 2, 3, ..., L), \tag{9}$$

where the encoder $\phi_{\check{\theta}}^{(l)}(\cdot)$ is parameterized by $\check{\theta} = \left[\boldsymbol{W}_{\mathrm{e}}^{(l)}, b_{\mathrm{e}}^{(l)}\right]$, $\boldsymbol{W}_{\mathrm{e}}^{(l)}$ and $b_{\mathrm{e}}^{(l)}$ denote the weight matrix and the bias of the encoder respectively, and $L$ is the total number of layers in the stacked encoder. In this study, the original input $\boldsymbol{\alpha}^{(0)}$ is the standardized snapshot data $\widetilde{\boldsymbol{u}} \in \mathbb{R}^{\mathcal{N}_x}$ (see Eq. (7)). The final latent feature $\boldsymbol{\alpha}^{(L)} \in \mathbb{R}^{m_L}$, often referred to as a "code", will be simplified in subsequent discussions as $\boldsymbol{\alpha}$ for ease of reference; its dimensionality $m_L$ will be denoted simply as $m$.

The $l$-th decoder, denoted by $\psi_{\hat{\theta}}^{(l)}(\cdot)$, transforms the hidden representation $\widehat{\boldsymbol{\alpha}}^{(l)}$ into the input space $\widehat{\boldsymbol{\alpha}}^{(l-1)}$ through the following equation:

$$\widehat{\boldsymbol{\alpha}}^{(l-1)} = \psi_{\hat{\theta}}^{(l)}\Big(\widehat{\boldsymbol{\alpha}}^{(l)}\Big) = g\Big(\boldsymbol{W}_{\mathrm{d}}^{(l)}\,\widehat{\boldsymbol{\alpha}}^{(l)} + b_{\mathrm{d}}^{(l)}\Big),\quad (l = 1, 2, 3, ..., L), \tag{10}$$

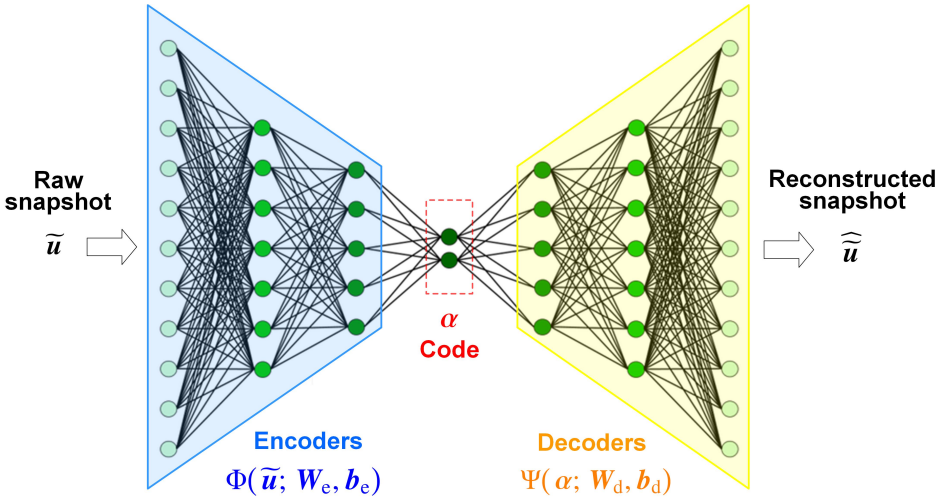



Figure 1: The architecture of a typical stacked autoencoder (SAE) model.

where the decoder $\psi_{\hat{\theta}}^{(l)}(\cdot)$ is parameterized by $\hat{\theta} = \left[\boldsymbol{W}_{\mathrm{d}}^{(l)}, b_{\mathrm{d}}^{(l)}\right]$, and $\boldsymbol{W}_{\mathrm{d}}^{(l)}$ and $b_{\mathrm{d}}^{(l)}$ denote the weight matrix and the bias of the decoder respectively. It should be noted that $\widehat{\boldsymbol{\alpha}}^{(L)} = \boldsymbol{\alpha}^{(L)}$ here. Regarding the activation functions $g(\cdot)$, the ReLU function is adopted for the intermediate hidden layers, the Tanh function is also an alternative option, and the Sigmoid function is used for the final output layer, which can be mathematically expressed as follows:

$$\text{ReLU function:} \quad g(x) = ReLU(x) = \max(0, x), \tag{11}$$

$$\text{Tanh function:} \quad g(x) = \tanh(x) = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}}, \tag{12}$$

$$\text{Sigmoid function:} \quad g(x) = \sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}}. \tag{13}$$

Training a SAE model involves minimizing the discrepancy between the initial input $\widetilde{\boldsymbol{u}}_i$ and the final output $\widehat{\widetilde{\boldsymbol{u}}}_i$. This is achieved by optimally adjusting the parameters of the neural network, which include weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$. The aforementioned discrepancy is quantified by a cost function comprising three terms, expressed as follows:

$$\mathcal{L}\left(\boldsymbol{W},\ \boldsymbol{b}\right) = \frac{1}{n}\sum_{i=1}^{n} L\left(\widetilde{\boldsymbol{u}}_i,\ \widehat{\widetilde{\boldsymbol{u}}}_i\right) + \beta \sum_{j=1}^{m} KL\left(\kappa \,\|\, \widehat{\kappa}_j\right) + \gamma \left\|\boldsymbol{W}\right\|_2^2 \ , \tag{14}$$

where $\beta$ and $\gamma$ are constant coefficients. The first term of the loss function is the mean sum of squares error that accounts for the discrepancy between the input $\widetilde{\boldsymbol{u}}_i$ and the output $\widehat{\widetilde{\boldsymbol{u}}}_i$ across the entire data set containing the information of the snapshots $n$:

$$\frac{1}{n}\sum_{i=1}^{n} L\left(\widetilde{\boldsymbol{u}}_i,\ \widehat{\widetilde{\boldsymbol{u}}}_i\right) = \frac{1}{n}\sum_{i=1}^{n} \left\|\widetilde{\boldsymbol{u}}_i - \widehat{\widetilde{\boldsymbol{u}}}_i\right\|^2 = \frac{1}{n}\sum_{i=1}^{n} \left\|\widetilde{\boldsymbol{u}}_i - \psi_{\hat{\theta}}\left[\phi_{\hat{\theta}}(\widetilde{\boldsymbol{u}}_i)\right]\right\|^2, \tag{15}$$

Encouraging sparsity in autoencoders facilitates the discovery of latent representations within input data while minimizing redundant information. To achieve this, a sparsity cost can be incorporated into the loss function to penalize activations of hidden layers, with the objective of activating fewer hidden neurons without compromising data processing performance. In this context, the Kullback-Leibler ($KL$) divergence is employed to impose a constraint on the output sparsity from hidden layers. This is represented by the second term in Eq. (14), which delineates the sparsity cost (where $m$ denotes the number of neurons in the hidden layer and index $j$ iterates over all hidden neurons within the network):

$$KL\left(\kappa \,\|\, \widehat{\kappa}_j\right) = \kappa \log \frac{\kappa}{\widehat{\kappa}_j} + (1-\kappa) \log \frac{1-\kappa}{1-\widehat{\kappa}_j}, \tag{16}$$

where $\widehat{\kappa}_j$ is the average activation value of the hidden neuron $j$ throughout the data set, and $\kappa$ is the desired value of average activation value. The third term in Eq. (14) defines the regularization cost (also called the weight decay

term), which tends to decrease the magnitude of the weights to prevent overfitting:

$$\|\boldsymbol{W}\|_2^2 = tr\big(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}\big). \tag{17}$$

In summary, the appropriately trained SAE model yields a stacked encoder (SE) function for nonlinear dimensionality reduction of high-fidelity snapshots, given by:

$$\Phi(\widetilde{\boldsymbol{u}}_i;\, \boldsymbol{W}_{\mathrm{e}}, \boldsymbol{b}_{\mathrm{e}}):\ \ \widetilde{\boldsymbol{u}}_i \in \mathbb{R}^{\mathcal{N}_x} \longrightarrow \boldsymbol{\alpha}_i \in \mathbb{R}^m, \tag{18}$$

where $\Phi(\cdot)$ denotes the SE function. A standardized snapshot data $\widetilde{\boldsymbol{u}}_i$ is greatly compressed into a low-dimensional code $\boldsymbol{\alpha}_i$ through SE, where $\mathcal{N}_x \gg m$. Furthermore, the stacked decoder (SD) function can recover the initial input $\widehat{\widetilde{\boldsymbol{u}}}_i$ from the code $\boldsymbol{\alpha}_i$, given by:

$$\Psi(\boldsymbol{\alpha}_i;\, \boldsymbol{W}_{\mathrm{d}}, \boldsymbol{b}_{\mathrm{d}}):\ \ \boldsymbol{\alpha}_i \in \mathbb{R}^m \longrightarrow \widehat{\widetilde{\boldsymbol{u}}}_i \in \mathbb{R}^{\mathcal{N}_x}, \tag{19}$$

where $\Psi(\,\cdot\,)$ signifies the SD function.

## 4. Predictive reduced-order modeling

*Long short-term memory* (LSTM) network is employed to construct predictive reduced-order models for time-series forecasting of turbulent fluid flow dynamics. The latent features (see Figure 1) derived from high-dimensional snapshots using SAE are utilized to train the predictive LSTM models via data-driven modeling. Here, two distinct strategies are introduced to train these LSTM models: Bunch training and Individual training.

### *4.1. Basic theory of LSTM*

LSTM is a specialized variant of Recurrent Neural Network (RNN), introduced by Hochreiter and Schmidhuber [36]. It is specifically designed to capture long-term dependencies through the incorporation of memory cells and gating mechanisms. These gates effectively regulate the flow of information within the network, thereby alleviating the vanishing gradient problem. As graphically illustrated in Figure 2, each LSTM cell consists of three primary gates: the forget gate, the input gate, and the output gate, and their roles can be mathematically represented as follows.

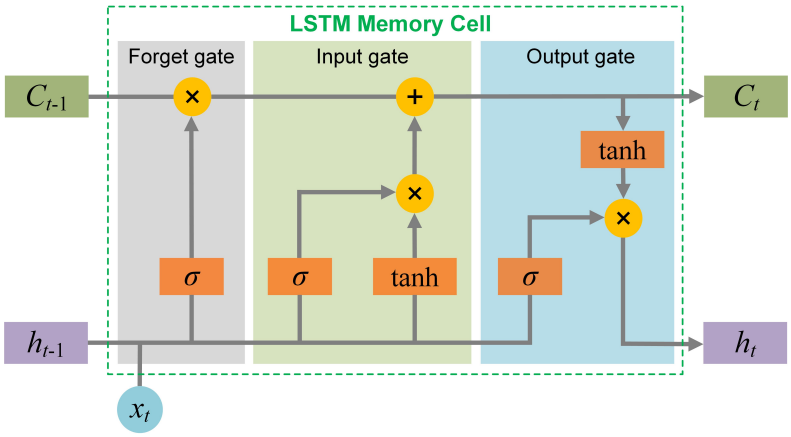



Figure 2: The architecture of a long short-term memory (LSTM) cell consisting of three primary gates: forget gate, input gate, and output gate.

The forget gate plays a crucial role in determining which information should be discarded from the cell state, which is mathematically represented as follows:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big), \tag{20}$$

where $f_t$ denotes the forget gate, $h_{t-1}$ signifies the previous hidden state, $x_t$ represents the current input, $W_f$ is the weight matrix, and $b_f$ indicates the bias term. The Sigmoid function $\sigma(\cdot)$ ensures that the output of the forget gate remains within a range of 0 to 1.

The input gate determines which new information will be incorporated into the cell state. This is represented mathematically as follows:

$$i_t = \sigma\Big(W_i \cdot [h_{t-1}, x_t] + b_i\Big), \tag{21}$$

$$\widetilde{C}_t = \tanh\Big(W_c \cdot [h_{t-1}, x_t] + b_c\Big), \tag{22}$$

where $i_t$ denotes the input gate, $\widetilde{C}_t$ signifies the candidate cell state, and $\tanh(\cdot)$ represents the hyperbolic tangent function. The weight matrices $W_i$ and $W_c$, along with the bias terms $b_i$ and $b_c$, define how the network processes both the current input and the previous hidden state.

The output gate regulates the extent to which the cell state is transmitted to the subsequent time step:

$$o_t = \sigma\Big(W_o \cdot [h_{t-1}, x_t] + b_o\Big), \tag{23}$$

$$h_t = o_t \cdot \tanh(C_t), \tag{24}$$

where $h_t$ represents the new hidden state and $C_t$ denotes the updated cell state, achieved by integrating both previous and new information:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C}_t\,. \tag{25}$$

The gating mechanisms of LSTM network facilitate the retention of pertinent information across extended sequences, effectively addressing the limitations associated with traditional RNNs [36]. These dynamic properties have rendered LSTM a widely favored option for a variety of tasks, including time-series prediction, speech recognition, and natural language processing [45].

### *4.2. Training strategy*

In this work, two different strategies are employed to train the LSTM models: *Bunch* training and *Individual* training. In the bunch training strategy, all elements within a latent feature vector $\boldsymbol{\alpha}$ (refer to Figure 1) are processed simultaneously by LSTM, where all feature elements are treated as an integrated whole. The input data, consisting of the latent feature vectors $\boldsymbol{\alpha}$ derived from the first $(n-1)$ snapshots using SAE, is structured as a tensor with dimensions $(n-l,\ l,\ m)$, where $n$ denotes the number of feature vectors utilized for model training, $l$ indicates the sequence length (i.e., the LSTM model is trained with a sequence length of 1), and $m$ represents the length of a latent feature vector $\boldsymbol{\alpha}$. By leveraging the feature vector in its entirety, the LSTM model can effectively capture shared temporal patterns among feature elements, thereby potentially enhancing prediction performance.

The individual training strategy means to train separate LSTM models for each element $\alpha_i$ in a latent feature vector $\boldsymbol{\alpha}$, with all LSTM models being trained using the same fixed sequence length. The input data, composed of individual feature elements $\alpha_i$, is structured as a tensor with dimensions $(n-l,\ l,\ 1)$, where $n$ denotes the number of snapshots utilized for model training, $l$ indicates the sequence length (i.e., the LSTM model is trained with a sequence length of 90), and 1 represents the length of a single feature element. In this setup, the LSTM models are tasked with predicting the next time step for each sequence of feature elements. This training strategy enables LSTM models to specialize in capturing the temporal dependencies of individual feature elements, which may lead to improved performance when the signals are independent from one another.

In summary, the primary distinction between the bunch and individual training strategies lies in the manner in which LSTM processes input data. In the bunch strategy, the LSTM model captures relationships among different feature elements by training on all feature elements simultaneously, which can enhance performance when there is correlation among feature elements. In contrast, the individual strategy concentrates exclusively on each feature element's internal temporal dependencies, which may prove more effective when feature elements are independent. In scenarios where feature elements exhibit high correlation, the LSTM-bunch model can leverage inter-code information to improve forecasting accuracy. However, when feature elements display distinct patterns, the LSTM-individual model offers greater granularity by focusing on each feature's unique temporal dynamics.

## 5. Numerical investigation

In order to validate the effectiveness of integrating the SAE network with the LSTM network for model order reduction of turbulence dynamics, the proposed methodology is employed to construct predictive reduced-order models for two representative engineering problems: turbulent flows passing around a single cylinder, and turbulent flows passing around multiple bridge pillars.

### 5.1. Performance metrics

The performance of the SAE-LSTM models is compared against that of the SAE-DMD models [34] and full-order numerical models, in terms of prediction accuracy and computational efficiency. The discrepancy between the full-order solution $\boldsymbol{u}(t)$ and the reduced-order solution $\widehat{\boldsymbol{u}}(t)$ is quantified using two indicators: root-mean-square error (RMSE) and correlation coefficient (CC). The RMSE quantifies the difference between $\boldsymbol{u}(t)$ and $\widehat{\boldsymbol{u}}(t)$, calculated as follows:

$$RMSE\Big(\boldsymbol{u}(t), \widehat{\boldsymbol{u}}(t)\Big) = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\Big(\boldsymbol{u}(t,k) - \widehat{\boldsymbol{u}}(t,k)\Big)^2}, \tag{26}$$

where $k$ denotes the index of a meshing node in the computational domain, and $K$ is the total number of meshing nodes. The CC computes the strength of the linear relationship between $\boldsymbol{u}(t)$ and $\widehat{\boldsymbol{u}}(t)$, and it can be expressed as :

$$CC\Big(\boldsymbol{u}(t), \widehat{\boldsymbol{u}}(t)\Big) = \frac{E\Big[\Big(\boldsymbol{u}(t) - \mu_{\boldsymbol{u}(t)}\Big)\Big(\widehat{\boldsymbol{u}}(t) - \mu_{\widehat{\boldsymbol{u}}(t)}\Big)\Big]}{\sigma_{\boldsymbol{u}(t)}\sigma_{\widehat{\boldsymbol{u}}(t)}}, \tag{27}$$

where $E(\cdot)$ denotes the expectation operator, $\mu_{\boldsymbol{u}(t)}$ and $\sigma_{\boldsymbol{u}(t)}$ are the mean and the standard deviation of $\boldsymbol{u}(t)$ respectively; $\mu_{\widehat{\boldsymbol{u}}(t)}$ and $\sigma_{\widehat{\boldsymbol{u}}(t)}$ are the mean and the standard deviation $\widehat{\boldsymbol{u}}(t)$ respectively.

### 5.2. Case 1: Turbulent flow passing around a circular cylinder

#### 5.2.1. Full-order numerical model

The first case study aims to investigate the turbulent flow passing around a circular cylinder in a 2D channel. The schematic diagram of the geometry is illustrated in Figure 3. The computational domain measures 22.0 m in length and 4.1 m in width, incorporating a circular cylinder with a diameter of $d = 1.0$ m positioned at the coordinates (2.0 m, 2.0 m). The open-source solver *Fluidity* [46] is employed to build the computational fluid dynamics (CFD) model utilizing large eddy simulation techniques. As shown in Figure 4, the finite element method (FEM) with unstructured meshes is applied to construct the numerical simulation model for turbulent flow, which comprises a total of 84,180 mesh nodes.

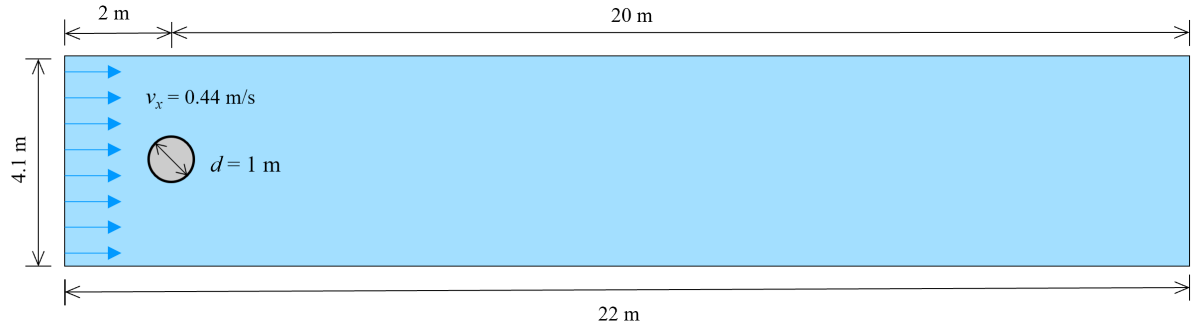



Figure 3: Schematic diagram of the turbulent flow passing around a circular cylinder.

Regarding the fluid-solid boundary conditions, the Dirichlet (no slip) boundary condition is imposed on the wall of the circular pillar, and zero outward flow boundaries are applied to both the lower and upper edges within the computational domain. Driven by an inlet horizontal velocity $v_x = 0.44$ m/s, fluid flows into the 2D channel from the leftmost side and circulates around the circular cylinder. The fluid kinematic viscosity is $\nu = \mu/\rho = 1.1 \times 10^{-4}$ m$^2$/s, and the Reynolds number of the fluid flow near the pillar is calculated to be $Re = v_x d/\nu = 4,000$. This high Reynolds number indicates that turbulent flow occurs inside the 2D channel. The numerical simulation lasts for 500 seconds with a time step of 0.01s, resulting in a total of 5,001 snapshots (full-order solutions of velocity components) obtained at equal time intervals $\Delta t = 0.10$s. During offline processing, SAE-LSTM models and SAE-DMD models are constructed using the first set of snapshots ranging from 0 to 460.0s; while performance evaluation utilizes remaining snapshots taken between 460.1s to 500.0s consisting of 400 data points (time instances) for extrapolation prediction.

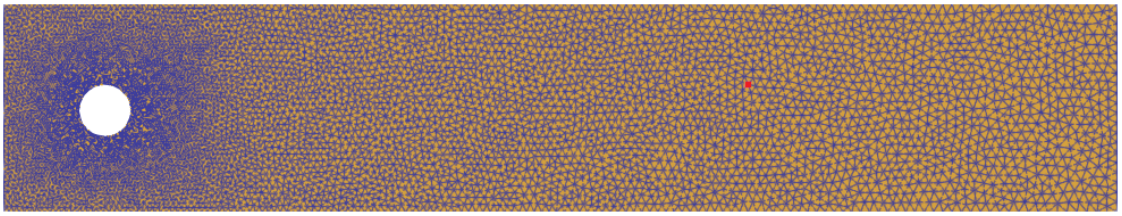


Figure 4: The numerical simulation model of turbulent flow passing around a circular cylinder, which is constructed by using the FEM with unstructured meshes (the red point will be used for further analysis).

### 5.2.2. *Dimensionality reduction*

As explained in Section 3, dimensionality reduction is an crucial step for deriving the low-dimensional codes (latent features $\boldsymbol{\alpha}$) from numerical solutions of velocity components $v_x$ and $v_y$ (from 0 to 460.0s). Here, both SAE and proper orthogonal decomposition (POD) are utilized to reduce dimensionality. Subsequently, a comparative analysis is conducted to demonstrate the advantages of SAE over POD within this case study. As presented in Table 1, the architectures and hyperparameters of the employed SAE model are outlined, with the Adam optimizer being applied alongside a predefined learning rate schedule to train the SAE models effectively.

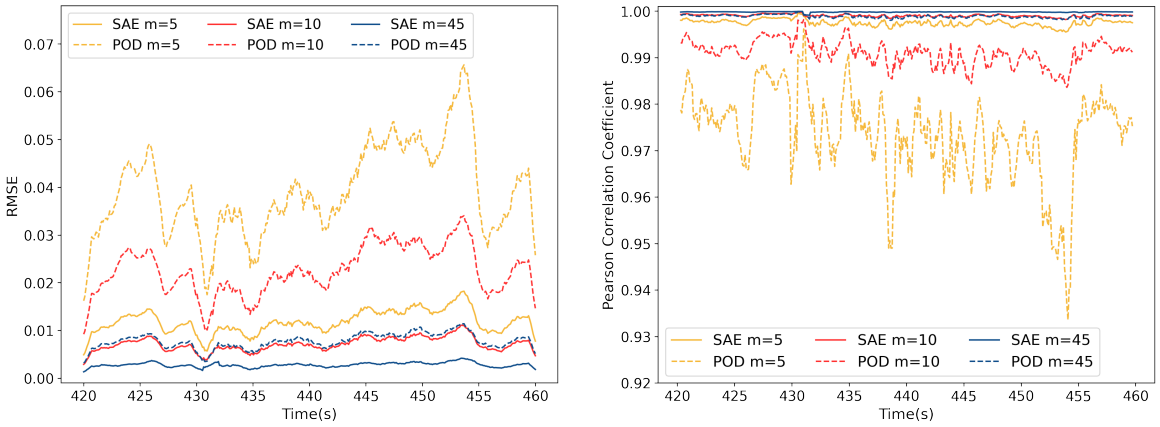
Table 1: The influence of code length $m$ (the dimension of latent space) on the results of dimensionality reduction using SAE

| Code length ($m$) | Captured energy (%) | Hyper-parameters of SAE models | |
|---|---|---|---|
| | | Hidden layers | Neurons |
| 5 | 90.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |
| 10 | 95.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |
| 45 | 99.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |

To systematically evaluate the performances of SAE and POD, standardized snapshot data is compressed into latent features of varying lengths (different values of $m$). The discrepancy between the raw input and the reconstructed data serves as an indicator for assessing the accuracy of dimensionality reduction, which can be quantified using RMSE and CC. As illustrated in Table 1, more and more energy of fluid flow dynamics can be captured by the SAE model, as the code length $m$ increases. When the code length $m$ reaches 45, over 99.0% of the flow dynamical energy can be well preserved after dimensionality reduction using SAE, where the high-dimensional snapshots (the original data length is 168,360 = 84,180 $\times$ 2) are significantly compressed into low-dimensional codes (the latent length is 45). Here, it is worthy of noting that the proportion of energy captured is calculated using principal component analysis. As illustrated in Figure 5, SAE models demonstrate significantly higher accuracy compared to POD regarding data compression. The main reason is that POD decomposes the raw data in a linear manner while SAE can well capture the nonlinear components. The comparative results indicate that SAE is more suitable than POD for dimensionality reduction of turbulent flow data, as it adeptly preserves the nonlinear characteristics inherent in fluid flow dynamics.

### 5.2.3. *Predictive reduced-order modeling*

As explained in Section 4, the latent features $\boldsymbol{\alpha}$ resulted from SAE serve as the inputs to train the LSTM models for reduced-order predictive modeling of turbulent fluid dynamics. Both the bunch training strategy and individual training strategy are applied to train the LSTM models, and the corresponding hyper-parameters are summarized in Tables 2 and 3 respectively. The Adam optimizer is employed to optimize the parameters of these LSTM models, where the learning rate starts at $1 \times 10^{-3}$ and is decayed via the plateau scheme. Here, both the LSTM-bunch and LSTM-individual models are trained over 500 epochs with the batch size of 32, and the entire time series dataset is split into training set (90%) and validation set (10%). Furthermore, a parametric study is carried out to identify the optimal sequence length $l$ for training the LSTM-individual models based on the latent space of Encoder when $m = 5$, and the optimal value is $l = 90$ for this case study, as illustrated in Figure 6.

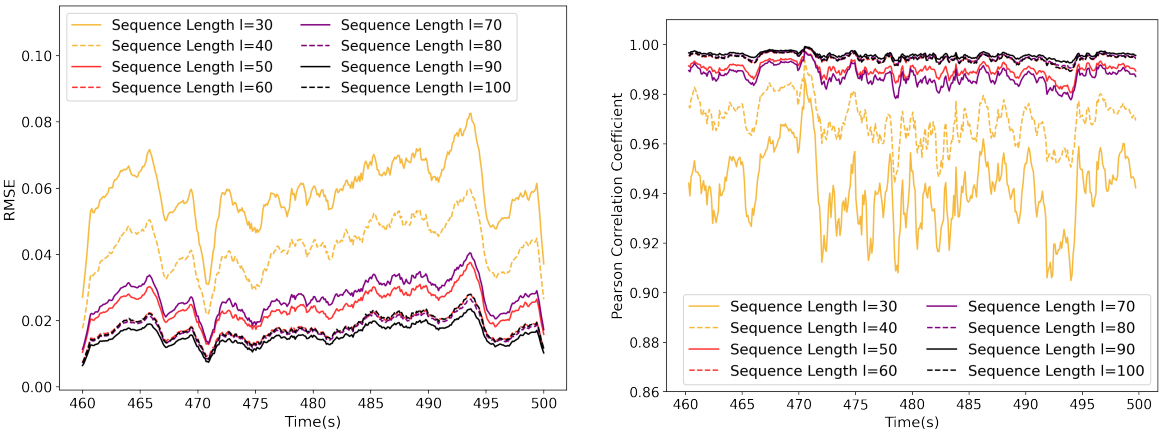



(a) RMSE (b) Correlation coefficient

Figure 5: Performance comparison between SAE and POD in terms of dimensionality reduction: (a) RMSE and (b) Correlation coefficient are used as the indicators to quantify the discrepancy between the raw input and the reconstructed output.

Table 2: The architecture and hyperparameters of the LSTM model using the bunch training strategy

| **Layer** | **Layer type** | **Output dimension** | **Activation function** | **Adam optimizer** | |
|---|---|---|---|---|---|
| | | | | **Initial learning rate** | **Learning rate schedule** |
| Input | - - | (32, 1, $m$) | - - | | |
| Hidden 1 | LSTM cell | (32, 100) | tanh | | |
| Hidden 2 | RepeatVector | (32, 1, 100) | - - | 0.001 | Reduce learning rate on plateau |
| Hidden 3 | LSTM cell | (32, 1, 100) | tanh | | |
| Output | TimeDistributed | (32, 1, $m$) | - - | | |




(a) RMSE (b) Correlation coefficient

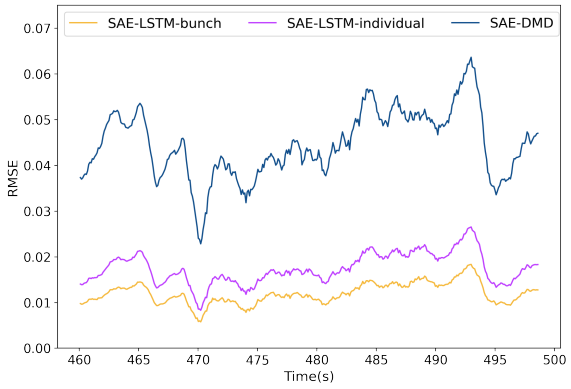
Figure 6: Forecasting performance of the LSTM-individual models with different sequence lengths $l$ (the optimal value of $l$ is 90).

The latent features (code data $\boldsymbol{\alpha}$) corresponding to high-dimensional snapshots from 0 to 460.0s are utilized to construct SAE-LSTM models (both bunch and individual) as well as SAE-DMD models [34] for predictive reduced-order modeling. A comparative study is conducted between the SAE-LSTM and SAE-DMD models to investigate their extrapolation prediction performance, where different code lengths $m$ are used. As illustrated in Figure 7, the velocity magnitude $|v|$ over the time period from 460.1s to 500.0s is predicted using these models, with prediction errors quantified against full-order numerical solutions through RMSE and correlation coefficients. Generally, the forecasting performances of all models improve as the code length $m$ increases. The SAE-LSTM-bunch models demonstrate significantly superior performance compared to the SAE-LSTM-individual models, while the prediction accuracy of SAE-DMD models falls between those two types of predictive frameworks when

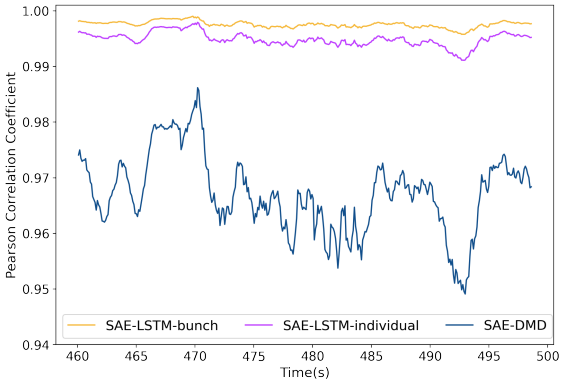
Table 3: The architecture and hyperparameters of the LSTM model using the individual training strategy

| **Layer** | **Layer type** | **Output dimension** | **Activation function** | **Adam optimizer** | |
|---|---|---|---|---|---|
| | | | | **Initial learning rate** | **Learning rate schedule** |
| Input | - - | (32, $l$, 1) | - - | 0.001 | Reduce learning rate on plateau |
| Hidden 1 | LSTM cell | (32, $l$, 100) | tanh | | |
| Hidden 2 | LSTM cell | (32, 100) | tanh | | |
| Output | Dense | (32, 1) | - - | | |

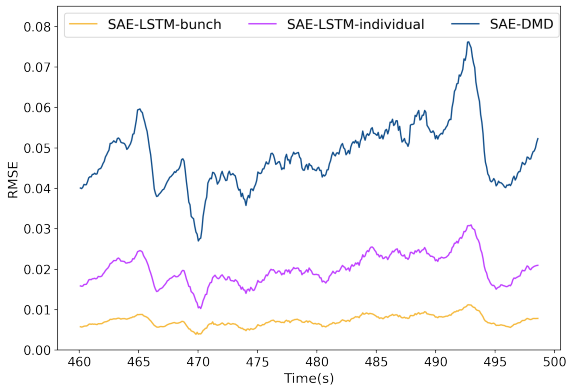
$m$ gradually increased to 45. When the code length $m = 45$, the reduced-order solutions predicted by the SAE-LSTM-bunch model closely align with the full-order numerical solutions, exhibiting RMSE values consistently below 0.005 (see Figure 7e).



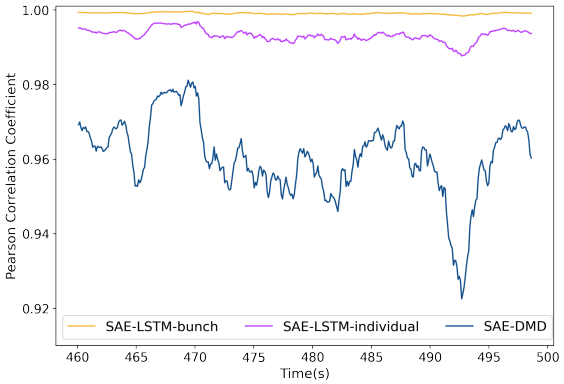

(a) RMSE ($m = 5$)



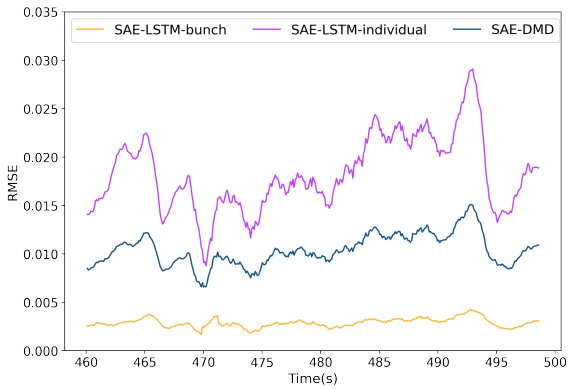

(b) Correlation coefficient ($m = 5$)



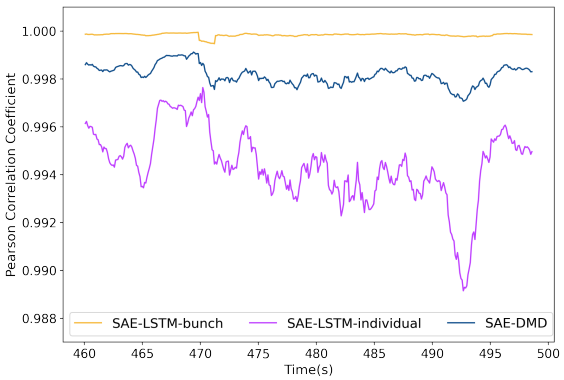

(c) RMSE ($m = 10$)




(d) Correlation coefficient ($m = 10$)

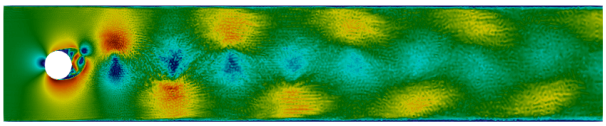



(e) RMSE ($m = 45$)

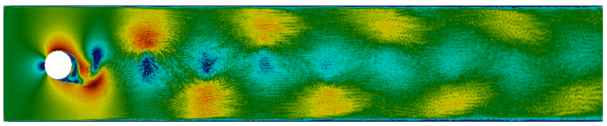



(f) Correlation coefficient ($m = 45$)

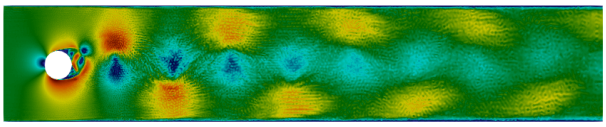
Figure 7: Performance comparison between the SAE-LSTM models (bunch and individual) and the SAE-DMD models in terms of extrapolation prediction of velocity magnitude $|v|$, where the full-order numerical solutions are used as the ground truth.



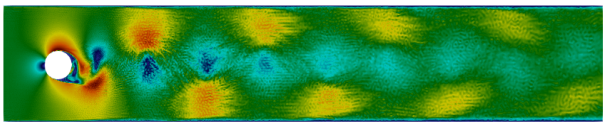
(a) The numerical simulation model, $t$ = 480s



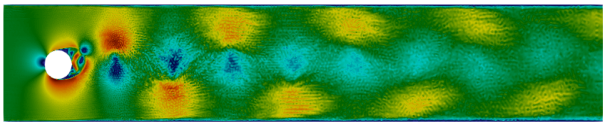
(b) The numerical simulation model, $t$ = 499.5s



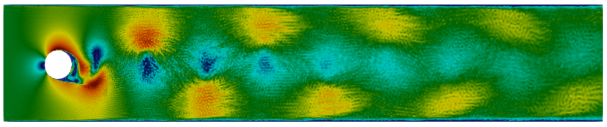
(c) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s



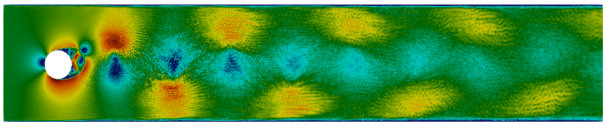
(d) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s



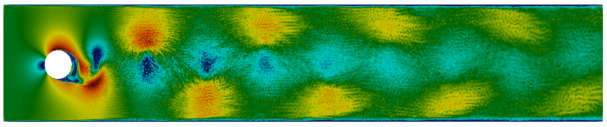
(e) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s



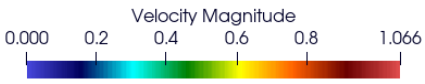
(f) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s



(g) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s



(h) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s



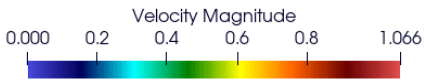
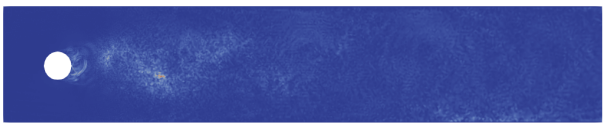
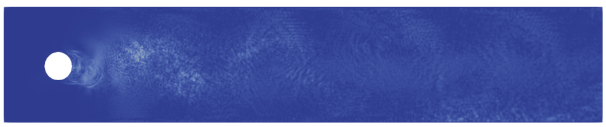




Figure 8: Comparison of velocity magnitude $|v|$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.

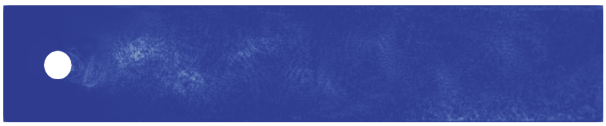


(a) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s



(b) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s

(c) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s



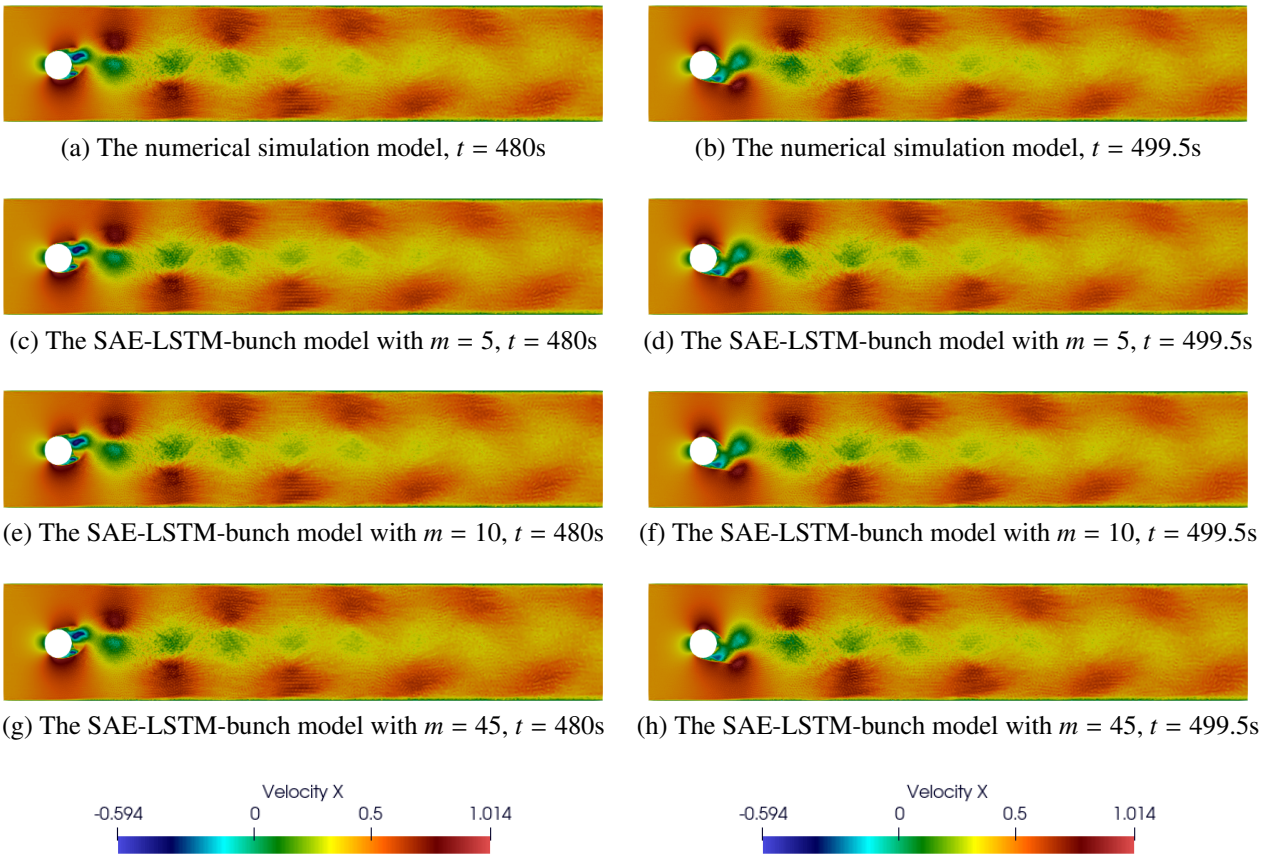
(d) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s

(e) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s

(f) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s



Figure 9: The absolute errors of the velocity magnitude $|v|$ predicted from the SAE-LSTM-bunch models with different code lengths $m$.




(a) The numerical simulation model, $t$ = 480s

(b) The numerical simulation model, $t$ = 499.5s

(c) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s

(d) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s

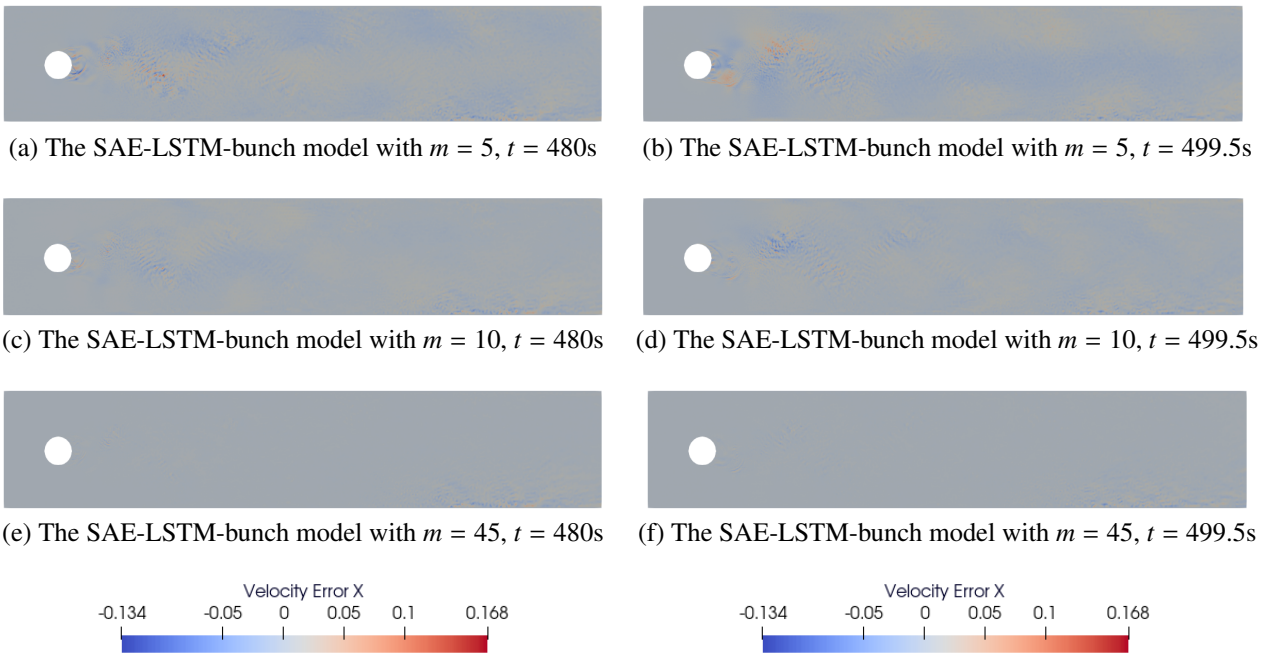
(e) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s

(f) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s

(g) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s

(h) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s

Figure 10: Comparison of velocity $v_x$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.




(a) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s

(b) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s

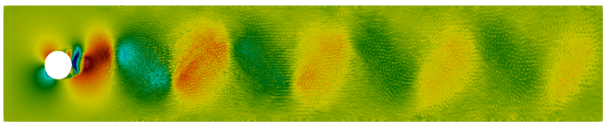
(c) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s

(d) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s

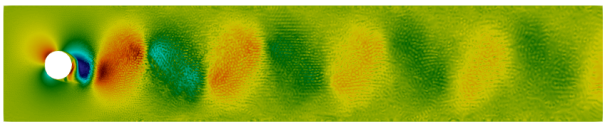
(e) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s

(f) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s

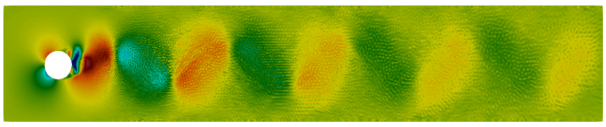
Figure 11: The absolute errors of the velocity component $v_x$ predicted by the SAE-LSTM-bunch models with different code lengths $m$.



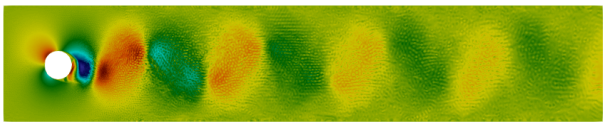
(a) The numerical simulation model, $t$ = 480s



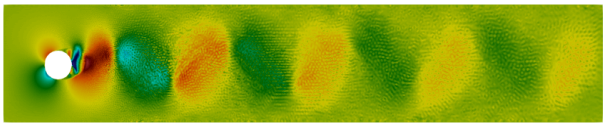
(b) The numerical simulation model, $t$ = 499.5s



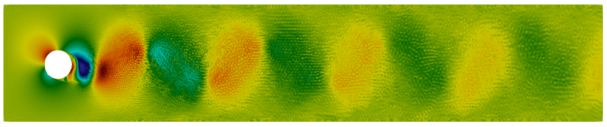
(c) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s



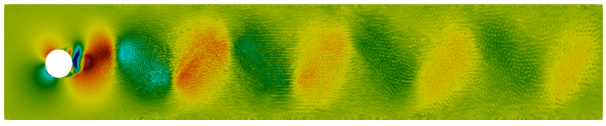
(d) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s



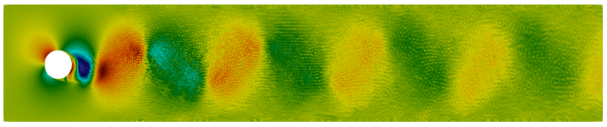
(e) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s



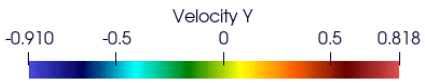
(f) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s



(g) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s



(h) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s



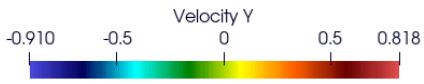




Figure 12: Comparison of velocity $v_y$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.

(a) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 480s

(b) The SAE-LSTM-bunch model with $m$ = 5, $t$ = 499.5s

(c) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 480s

(d) The SAE-LSTM-bunch model with $m$ = 10, $t$ = 499.5s

(e) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 480s

(f) The SAE-LSTM-bunch model with $m$ = 45, $t$ = 499.5s

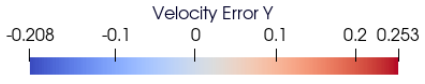
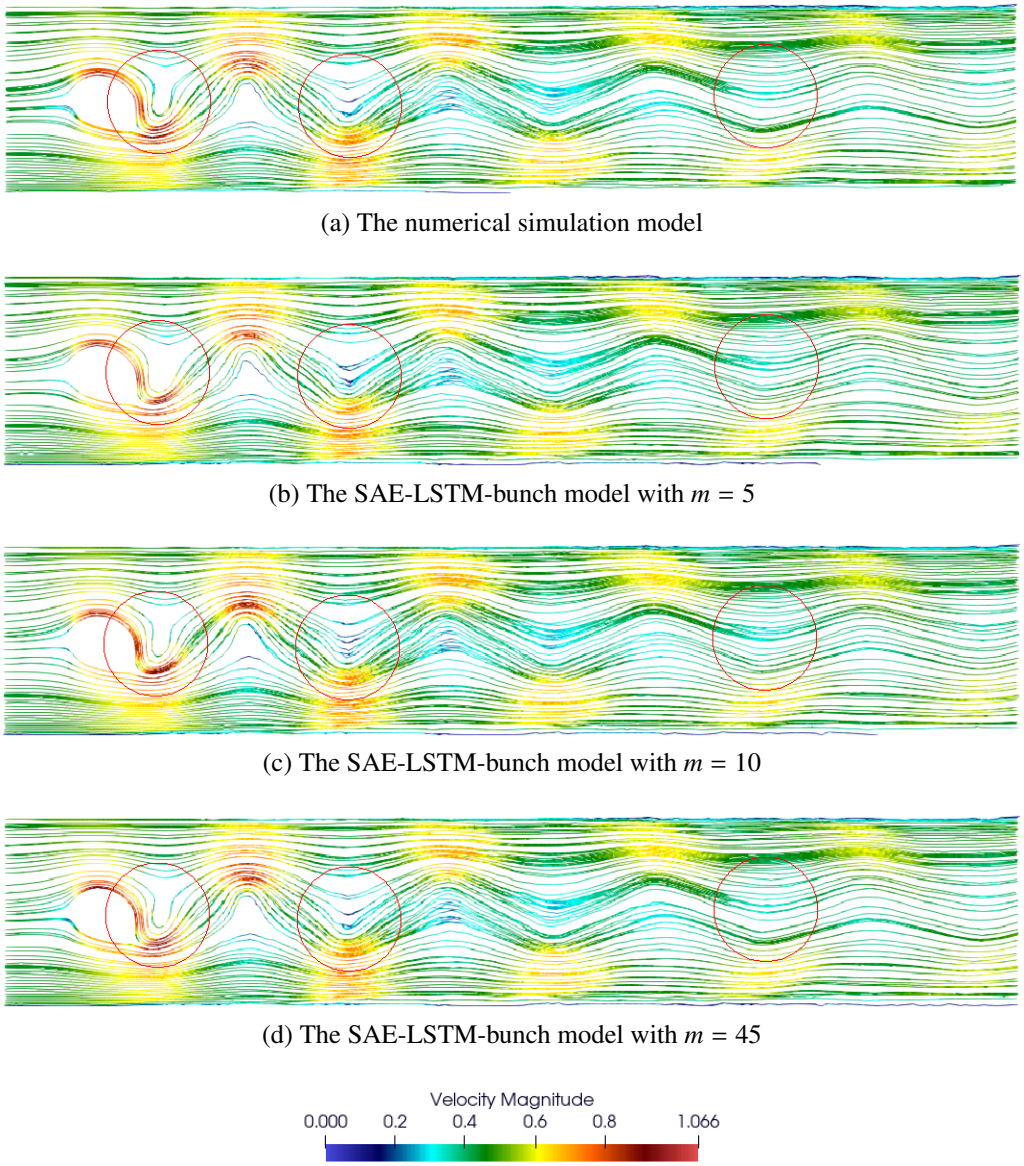




Figure 13: The absolute errors of the velocity component $v_y$ predicted from the SAE-LSTM-bunch models with different code lengths $m$.

As illustrated in Figures 8, 10, and 12, the reduced-order solutions of velocity fields at $t$ = 480s and 499.5s

are derived from the SAE-LSTM-bunch models with varying code lengths $m$. These reduced-order solutions are compared against the full-order numerical solutions obtained from CFD models, and the absolute errors between them are calculated, as depicted in Figures 9, 11, and 13. It is evident that the extrapolation performance of the SAE-LSTM-bunch models improves as the code length $m$ increases. When the code length $m$ reaches 45, the reduced-order solutions become indistinguishable from their full-order counterparts, resulting in an almost negligible absolute error between them. This exceptional predictive capability further substantiates that the proposed SAE-LSTM framework can effectively construct accurate reduced-order models for predicting previously unseen scenarios of turbulent flow dynamics, when a sufficiently large code length $m$ is employed.




(a) The numerical simulation model

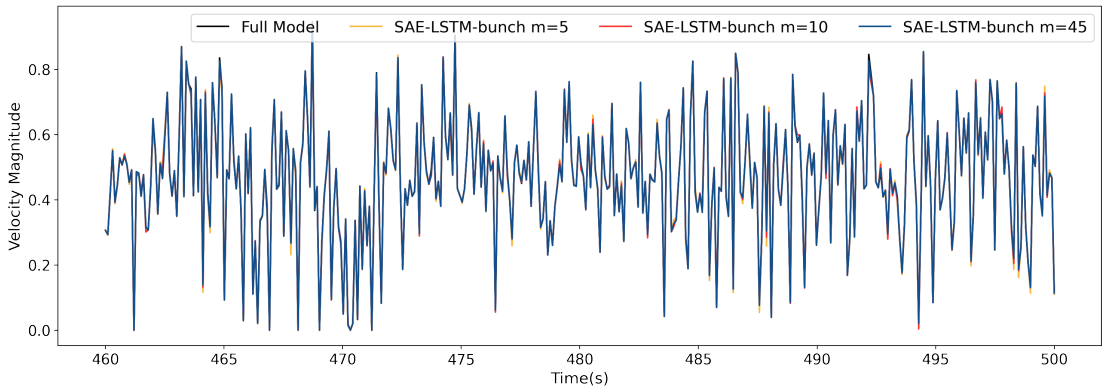
(b) The SAE-LSTM-bunch model with $m = 5$

(c) The SAE-LSTM-bunch model with $m = 10$

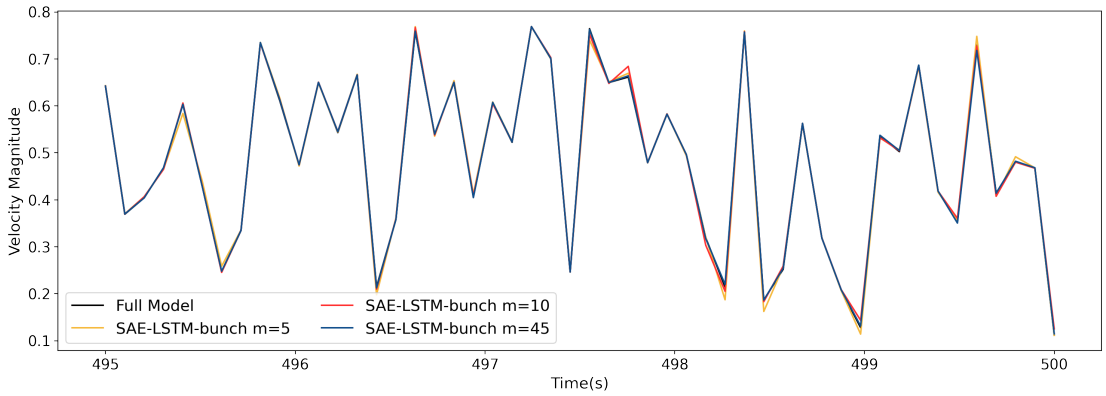
(d) The SAE-LSTM-bunch model with $m = 45$

Figure 14: The streamlines derived from the full-order solutions and the reduced-order solutions predicted from the SAE-LSTM-bunch models at $t = 499.5$s.

To provide a more detailed understanding of turbulent flow dynamics, the streamlines corresponding to both full-order and reduced-order solutions predicted by the SAE-LSTM-bunch models at $t = 499.5$s are illustrated in Figure 14. It is evident that the discrepancy between the full-order and reduced-order solutions diminishes as the code length $m$ increases. When the code length $m$ reaches 45, there is virtually no discernible difference between these two sets of solutions. These findings underscore the robust capability of the proposed SAE-LSTM-based ROM method in accurately capturing intricate characteristics of turbulent fluid flows.




(a) Predicted results of velocity magnitude during the time period from $t$ = 460.1s to 500.0s




(b) Predicted results of velocity magnitude during the time period from $t$ = 495.0s to 500.0s

Figure 15: The reduced-order solutions of velocity magnitude $|v|$ at the point ($x$ = 15.3109m, $y$ = 1.96666m) over time, where the SAE-LSTM-bunch models with different code lengths $m$ are used for extrapolation prediction.

In addition, a specific nodal point located at ($x$ = 15.3109m, $y$ = 1.96666m) has been selected from the computational domain for further analysis, with its position indicated by the red node in Figure 4. The velocity magnitude solutions $|v|$ obtained from both the full-order numerical model and the reduced-order models (the SAE-LSTM-bunch models) are illustrated in Figure 15. The velocity magnitude at this point exhibits significant temporal variation, highlighting the turbulent nature of fluid flow dynamics. When the code length $m \geq 10$, the velocity magnitude curves derived from reduced-order solutions align closely with those of the reference curve from full-order solutions. In Figure 16, absolute errors are calculated to evaluate the predictive performance of these reduced-order models; it is evident that prediction accuracy improves as code length $m$ increases. When $m$ reaches 45, the average magnitude of absolute errors over the time interval from 460.1s to 500s falls below 0.02. These findings confirm that the proposed SAE-LSTM-based ROM method is capable of constructing reliable reduced-order models to accurately represent turbulent fluid dynamics when an appropriate code length $m$ is chosen.

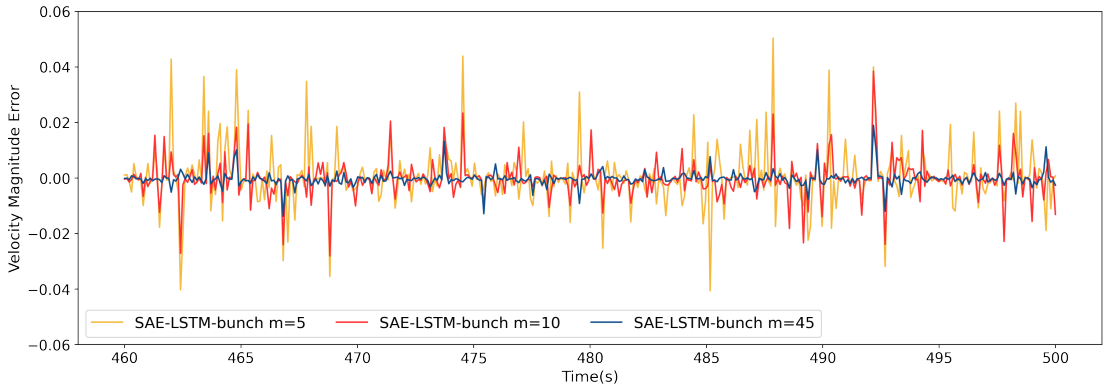



Figure 16: The absolute errors of the reduced-order solutions of velocity magnitudes $|v|$ at the point ($x$ = 15.3109m, $y$ = 1.96666m) over time, where the SAE-LSTM-bunch models with different code lengths $m$ are used for extrapolation prediction.

### *5.3. Case 2: Turbulent river flow passing around multiple bridge pillars*

#### *5.3.1. Full-order numerical model*

Compared to Case 1, the turbulent fluid flow interacting with multiple pillars presents a more intricate scenario, posing a greater challenge for the proposed SAE-LSTM-based ROM method. As illustrated in Figure 17, the three-dimensional computational domain measures 48.0 m in length, 24.0 m in width, and 1.073312 m in depth. This domain encompasses two bridge circular pillars with a diameter of 1.2 m positioned at coordinates (12.0 m, 12.0 m) and (17.6 m, 12.0 m), respectively. The open-source solver *OpenFoam* [47] is employed to numerically simulate the turbulent river flow around these two bridge pillars, utilizing the finite volume method (FVM) with a non-uniform meshing grid configuration adopted for this purpose. The three-dimensional numerical simulation model comprises a total of 55,268 computational elements, as depicted in Figures 18 and 19.

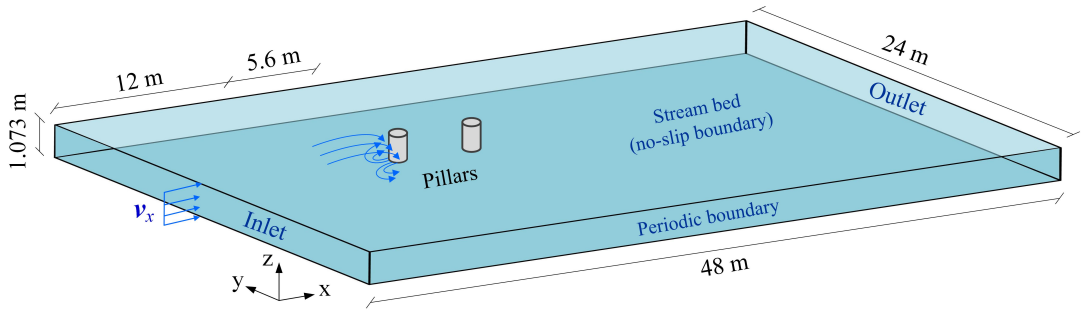



Figure 17: Schematic diagram of turbulent river flow passing along two circular bridge pillars (the diameter $d$ of each bridge pillar is 1.2 m).

Regarding the boundary conditions in this CFD simulation model, periodic boundaries are applied to both the lower and upper borders of the computational domain. Additionally, Dirichlet (no slip) boundary conditions are applied to the walls of bridge pillars. The muddy river flow is driven by an inlet velocity $v_x$ = 3.0 m/s from the leftmost border, and it passes around two circular bridge pillars in succession. The fluid kinematic viscosity is $\nu = \mu/\rho = 9.0 \times 10^{-4}$ m$^2$/s, and the Reynolds number of the river flow near the left bridge pillar is calculated to be $Re = v_x d/\nu = 4000$, indicating turbulent flow characteristics. The numerical simulation of fluid flow lasts for 200 seconds with a time step equal to 0.1s, resulting in 2001 snapshots obtained at equal time intervals $\Delta t$ = 0.1s for velocity components and pressure. During the offline phase, the SAE-LSTM models and SAE-DMD models are constructed using 1601 snapshots corresponding to a time period from $t$ = 0 to 160s. In contrast, during the online phase, their extrapolation prediction performances are tested using final 400 snapshots (from $t$ = 160.1s to 200.0s).

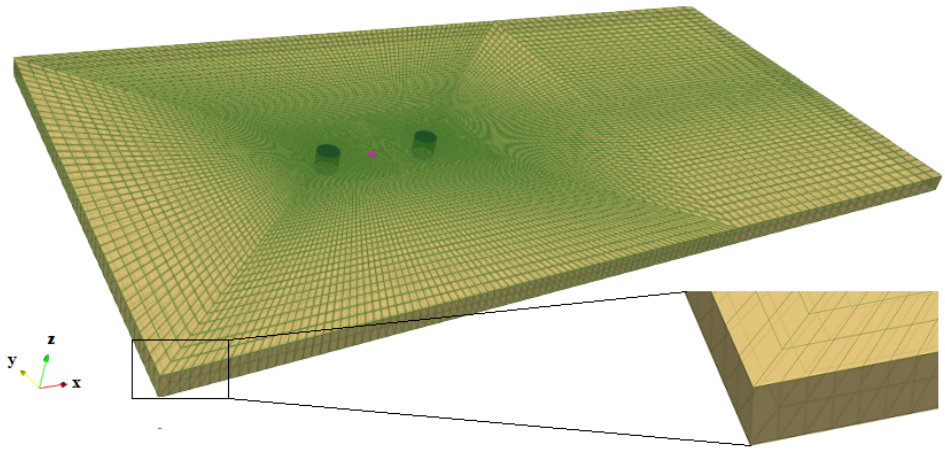



Figure 18: The CFD model for numerical simulation of turbulent muddy river flow passing along two bridge pillars, which is constructed by using the FVM with non-uniform meshes (the red point will be used for further analysis).

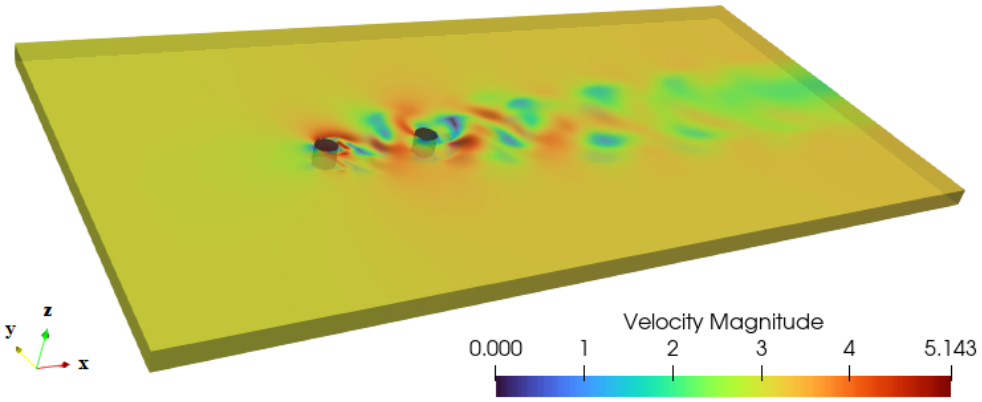


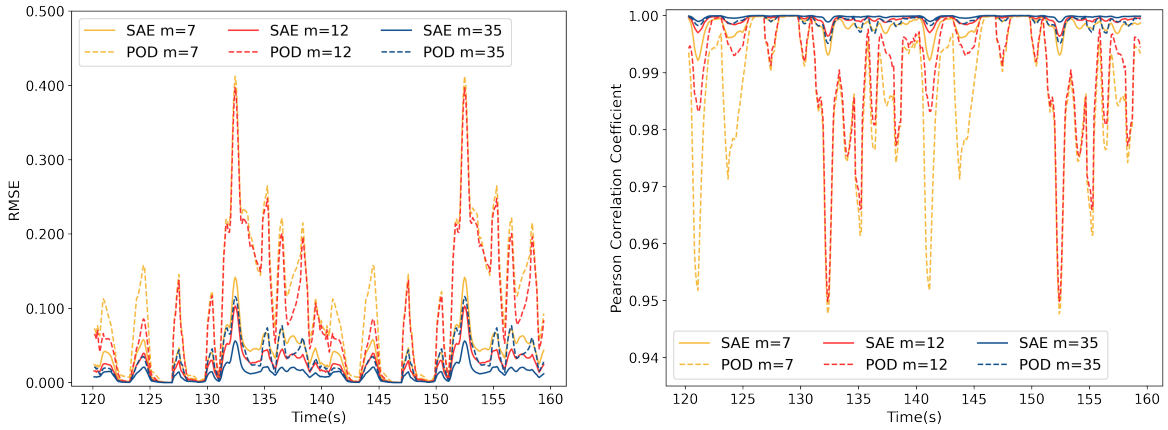

Figure 19: The numerical simulation result of flow velocity obtained from the 3D CFD model.

*5.3.2. Dimensionality reduction*

After acquiring numerical simulation data, the initial step of model order reduction involves compressing high-dimensional snapshots of velocity components into low-dimensional codes, as detailed in Section 3. Here, both SAE and POD are employed for dimensionality reduction. The RMSE and correlation coefficient are calculated to evaluate the accuracy of data compression by quantifying the discrepancy between raw input snapshots and their reconstructed counterparts. The architectures and hyperparameters of the SAE models are outlined in Table 4. As illustrated in Figure 20, SAE demonstrates significantly superior performance compared to POD in terms of dimensionality reduction, primarily due to its ability to effectively preserve the nonlinear components in turbulent fluid dynamics. Additionally, a parametric study is conducted to examine how code length $m$ influences the outcomes of dimensionality reduction using SAE. As shown in Table 4, an increasing code length $m$ allows for greater capture of energy associated with fluid flow dynamics by the SAE model. When $m$ reaches 35, over 99.0% of the dynamical energy within the turbulent flow can be preserved following dimensionality reduction via SAE; this represents a significant compression from high-dimensional snapshots (originally sized at 110,536 = 55,268 $\times$ 2) down to low-dimensional codes with a latent length of 35. It is important to note that this percentage of captured energy is estimated through principal component analysis. These findings further substantiate that SAE is more adept than POD for performing dimensionality reduction on turbulent flow data since it proficiently retains the inherent nonlinear characteristics present within fluid flow dynamics.

Table 4: The influence of code length $m$ (the dimension of latent space) on the results of dimensionality reduction using SAE

| Code length ($m$) | Captured energy (%) | Hyper-parameters of SAE models | |
|---|---|---|---|
| | | Hidden layers | Neurons |
| 7 | 90.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |
| 12 | 95.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |
| 35 | 99.0 | 4 | $m \times 8$; $m \times 4$; $m \times 2$; $m$ |




(a) RMSE (b) Correlation coefficient

Figure 20: Performance comparison between SAE and POD in terms of dimensionality reduction: (a) RMSE and (b) Correlation coefficient are used as the indicators to quantify the discrepancy between the raw input and the reconstructed output.

### *5.3.3. Predictive reduced-order modeling*

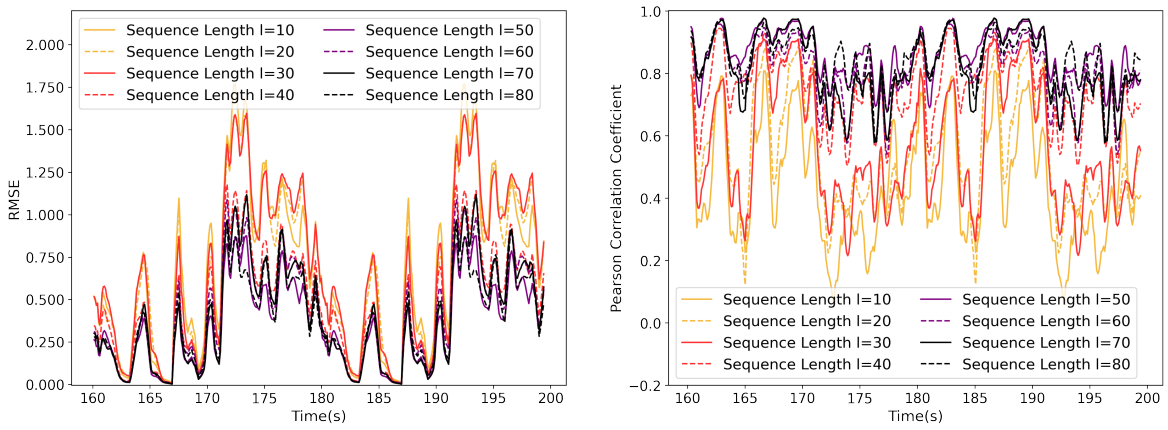
The latent features $\boldsymbol{\alpha}$ derived from SAE, as discussed in Section 4, serve as inputs for training LSTM models aimed at reduced-order predictive modeling of turbulent flow dynamics. Both the bunch training strategy and the individual training strategy are employed to train these LSTM models, with their corresponding hyperparameters summarized in Tables 5 and 6, respectively. The Adam optimizer is utilized to optimize the parameters of these LSTM models, beginning with a learning rate of $1\times10^{-3}$ which decays according to a plateau scheme. In this study, both LSTM-bunch and LSTM-individual models are trained over 500 epochs using a batch size of 32, while the entire time series dataset is partitioned into a training set comprising 90% and a validation set constituting 10%. Additionally, a parametric study is conducted to identify the optimal sequence length $l$ for training the LSTM-individual models based on the latent space of Encoder when $m = 7$, and it is determined that an optimal value of $l = 70$ pertains for this case study, as illustrated in Figure 21.

Table 5: The architecture and hyperparameters of SAE-LSTM-bunch model

| **Layer** | **Layer type** | **Output dimension** | **Activation function** | **Adam optimizer** | |
|---|---|---|---|---|---|
| | | | | **Initial learning rate** | **Learning rate schedule** |
| Input | - - | $(32, 1, m)$ | - - | | |
| Hidden 1 | LSTM cell | $(32, 100)$ | tanh | | |
| Hidden 2 | RepeatVector | $(32, 1, 100)$ | - - | 0.001 | Reduce learning rate on plateau |
| Hidden 3 | LSTM cell | $(32, 1, 100)$ | tanh | | |
| Output | TimeDistributed | $(32, 1, m)$ | - - | | |

Table 6: The architecture and hyperparameters of SAE-LSTM-individual model

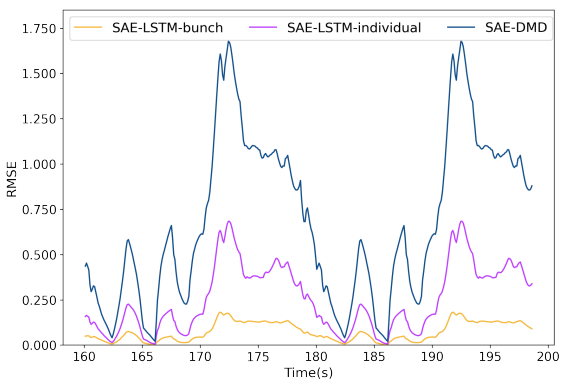
| Layer | Layer type | Output dimension | Activation function | Adam optimizer | |
|---|---|---|---|---|---|
| | | | | Initial learning rate | Learning rate schedule |
| Input | - - | $(32, l, 1)$ | - - | 0.001 | Reduce learning rate on plateau |
| Hidden 1 | LSTM cell | $(32, l, 100)$ | tanh | | |
| Hidden 2 | LSTM cell | $(32, 100)$ | tanh | | |
| Output | Dense | $(32, 1)$ | - - | | |



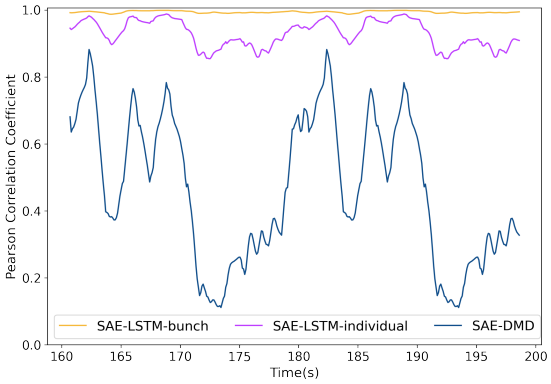

(a) RMSE (b) Correlation coefficient

Figure 21: Forecasting performance of the LSTM-individual models with different sequence lengths $l$ (the optimal value of $l$ is 70).

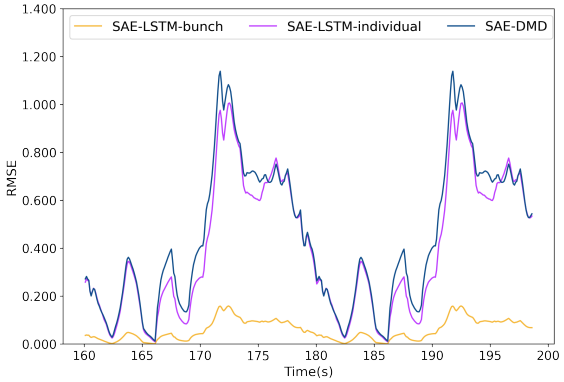
The latent features (code data $\boldsymbol{\alpha}$) corresponding to high-dimensional snapshots from 0 to 160.0s are utilized to construct SAE-LSTM models (both bunch and individual) as well as SAE-DMD models [34] for reduce-order predictive modeling. A comparative study is performed between the SAE-LSTM and SAE-DMD models to examine their extrapolation prediction performance, utilizing various code lengths $m$. As depicted in Figure 22, the velocity magnitude $|v|$ over the time interval from 160.1s to 200.0s is predicted using these models, with prediction errors assessed against full-order numerical solutions through RMSE and correlation coefficients. Overall, the forecasting performances of all models improve with an increase in code length $m$. The SAE-LSTM-bunch models exhibit significantly enhanced performance compared to the SAE-LSTM-individual models; meanwhile, the prediction accuracy of a variety of SAE-DMD models resides between these two frameworks. When the code length $m = 35$, the reduced-order solutions obtained from the SAE-LSTM-bunch model demonstrate a close alignment with full-order numerical solutions, as evidenced by correlation coefficients consistently approaching 1.0 (see Figure 22f).



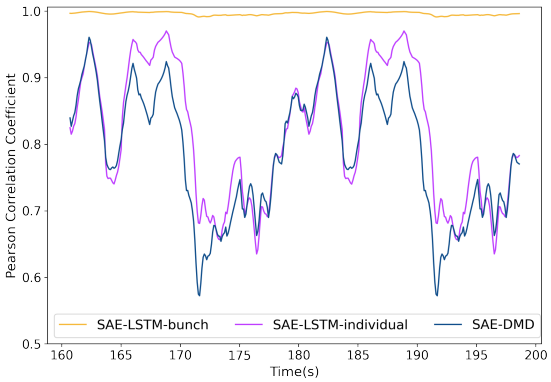

(a) RMSE ($m$ = 7)



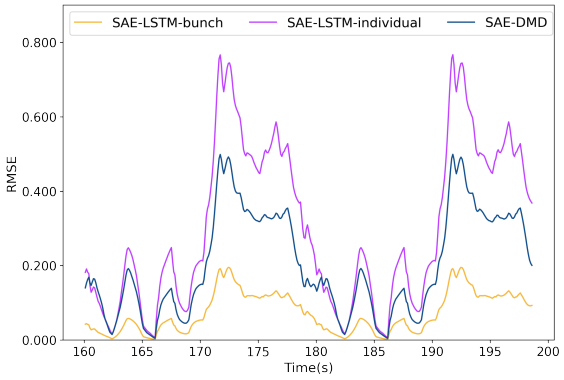

(b) Correlation coefficient ($m$ = 7)



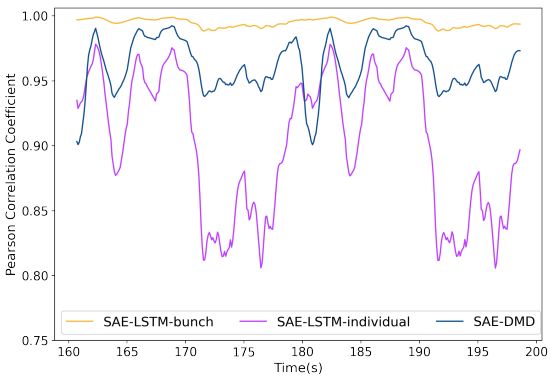

(c) RMSE ($m$ = 12)




(d) Correlation coefficient ($m$ = 12)



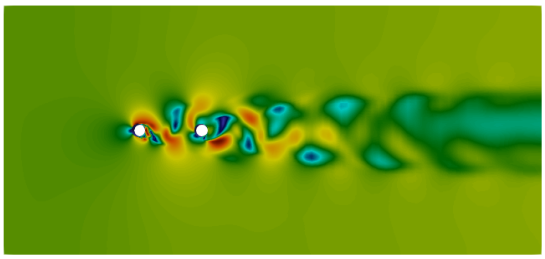

(e) RMSE ($m$ = 35)



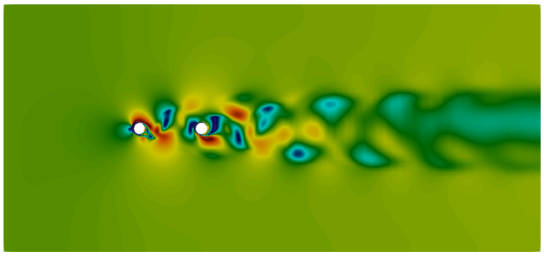

(f) Correlation coefficient ($m$ = 35)

Figure 22: Performance comparison between the SAE-LSTM models (bunch and individual) and the SAE-DMD models in terms of extrapolation prediction of velocity magnitude $|v|$, where the full-order numerical solutions are used as the ground truth.

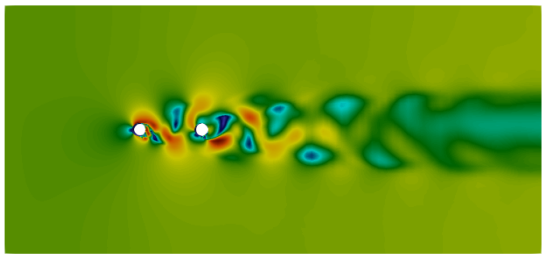
As elucidated in Figures 23, 25 and 27, the reduced-order solutions for the velocity fields at $t$ = 180.0s and 199.5s are derived from the SAE-LSTM-bunch models employing varying code lengths $m$. These reduced-order outcomes are juxtaposed against full-order numerical solutions procured from the CFD model, with absolute errors meticulously calculated, as illustrated in Figures 24, 26 and 28. It becomes abundantly clear that the extrapolation capabilities of the SAE-LSTM-bunch models exhibit marked enhancement in conjunction with an increase in code length $m$. Upon reaching a code length of 35, these reduced-order solutions become virtually indistinguishable from their full-order counterparts, yielding an almost imperceptible absolute error between them. This remarkable predictive prowess fortifies our assertion that the proposed SAE-LSTM framework can adeptly construct precise reduced-order models for forecasting previously unencountered scenarios within turbulent flow dynamics, given that a sufficiently substantial code length $m$ is employed.



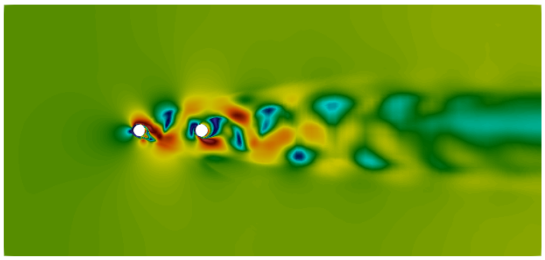
(a) The numerical simulation model, $t$ = 180s



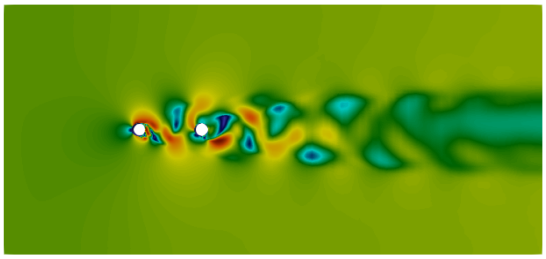
(b) The numerical simulation model, $t$ = 199.5s



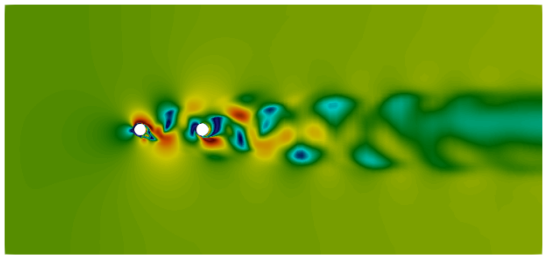
(c) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 180s



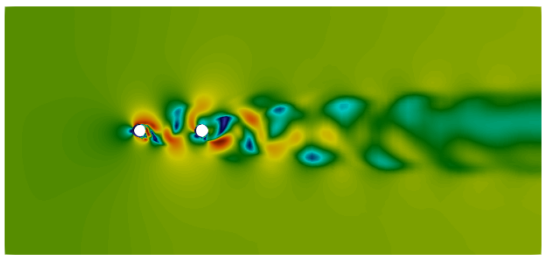
(d) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 199.5s



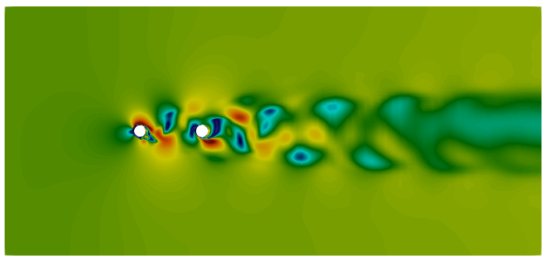
(e) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 180s



(f) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 199.5s



(g) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 180s



(h) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 199.5s

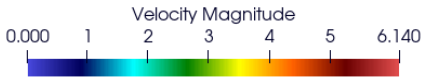
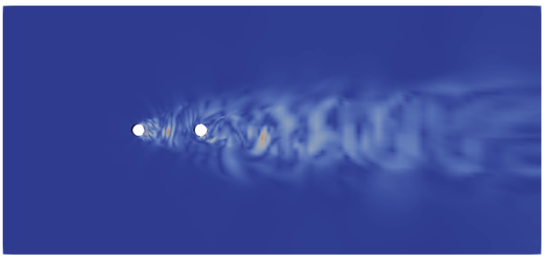
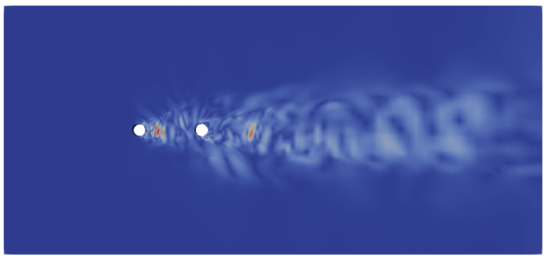



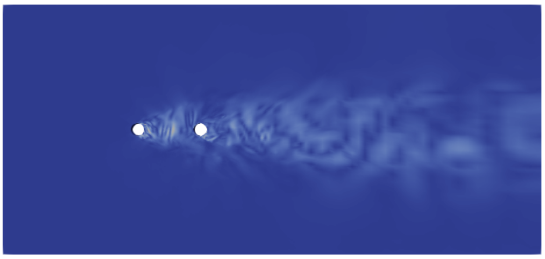

Figure 23: Comparison of velocity magnitude $|v|$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.



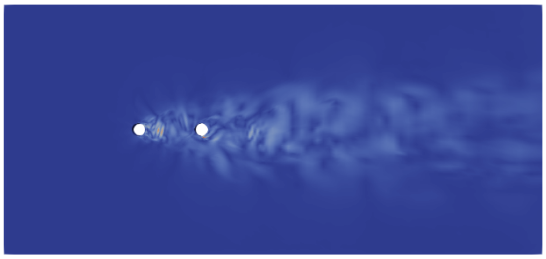
(a) The SAE-LSTM-bunch model with $m = 7$, $t = 180$s



(b) The SAE-LSTM-bunch model with $m = 7$, $t = 199.5$s

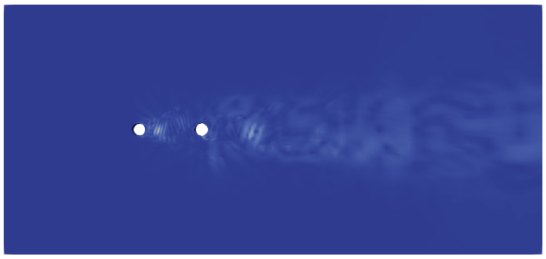


(c) The SAE-LSTM-bunch model with $m = 12$, $t = 180$s



(d) The SAE-LSTM-bunch model with $m = 12$, $t = 199.5$s

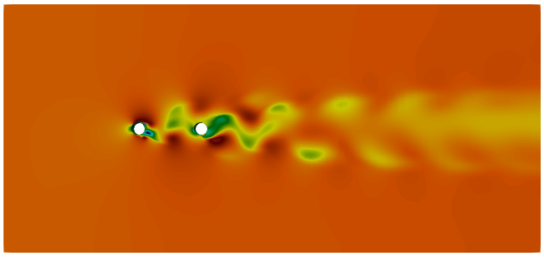
(e) The SAE-LSTM-bunch model with $m = 35$, $t = 180$s



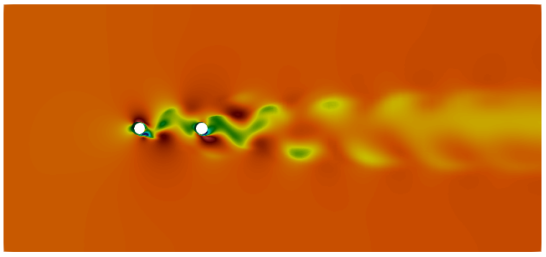
(f) The SAE-LSTM-bunch model with $m = 35$, $t = 199.5$s



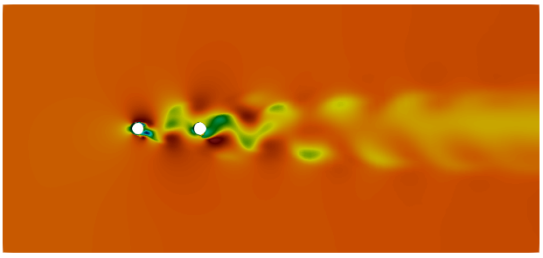
Figure 24: The absolute errors of the velocity magnitude $|v|$ predicted from the SAE-LSTM-bunch models with different code lengths $m$.



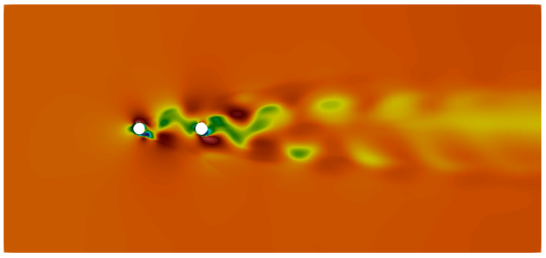
(a) The numerical simulation model, $t$ = 180s



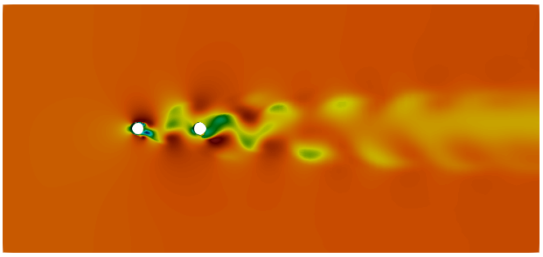
(b) The numerical simulation model, $t$ = 199.5s



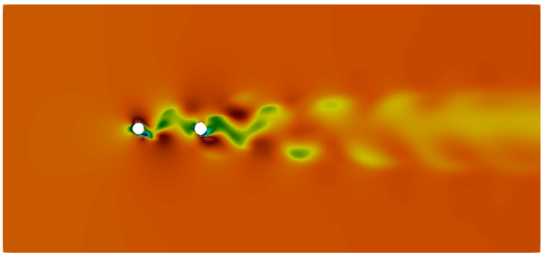
(c) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 180s



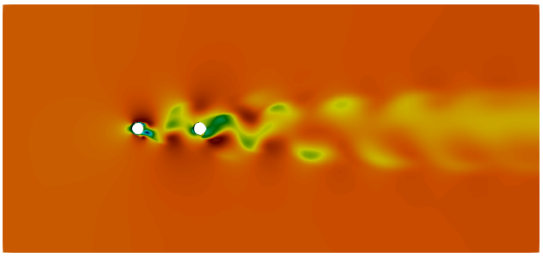
(d) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 199.5s



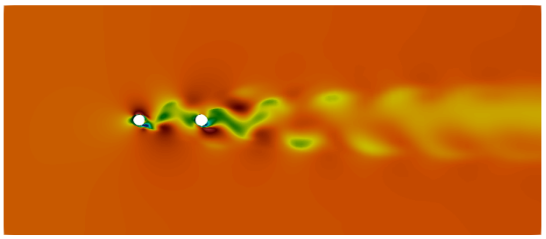
(e) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 180s



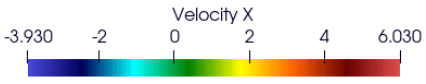
(f) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 199.5s



(g) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 180s



(h) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 199.5s



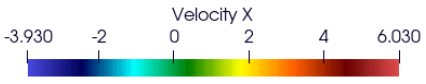
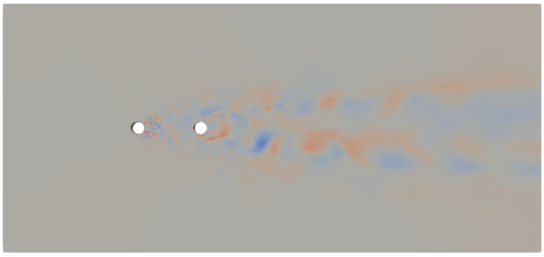
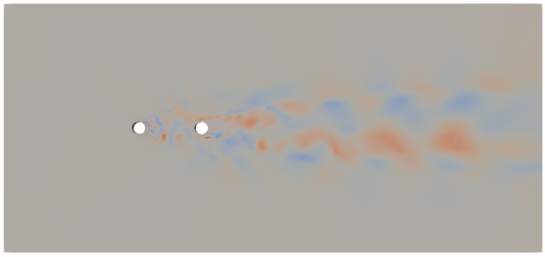




Figure 25: Comparison of velocity $v_x$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.

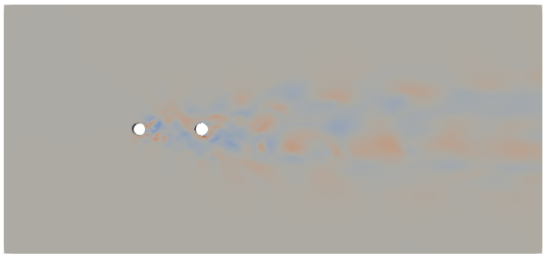


(a) The SAE-LSTM-bunch model with $m = 7$, $t = 180$s

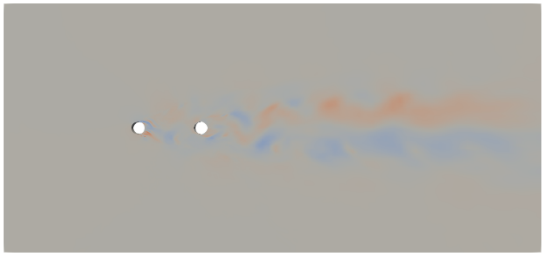


(b) The SAE-LSTM-bunch model with $m = 7$, $t = 199.5$s

(c) The SAE-LSTM-bunch model with $m = 12$, $t = 180$s

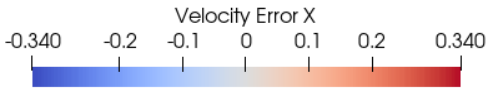


(d) The SAE-LSTM-bunch model with $m = 12$, $t = 199.5$s



(e) The SAE-LSTM-bunch model with $m = 35$, $t = 180$s

(f) The SAE-LSTM-bunch model with $m = 35$, $t = 199.5$s



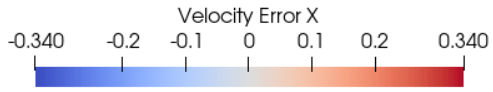
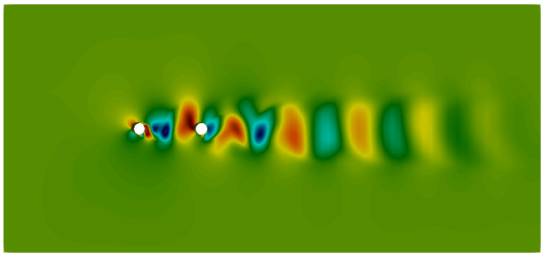
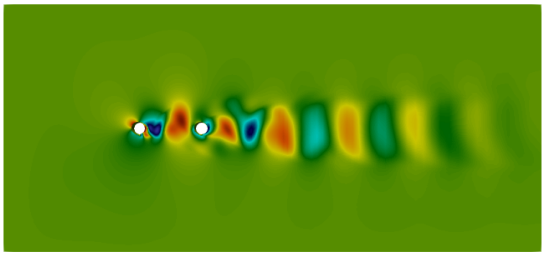



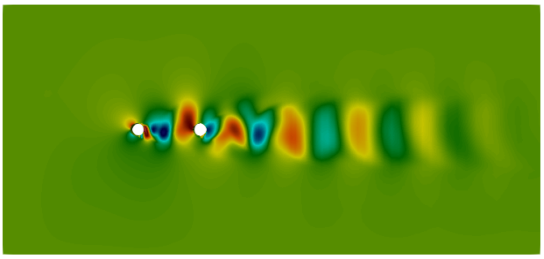

Figure 26: The absolute errors of the velocity component $v_x$ predicted from the SAE-LSTM-bunch models with different code lengths $m$.



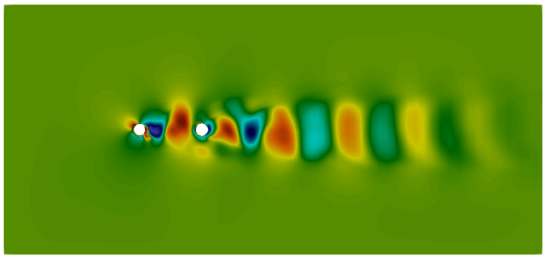
(a) The numerical simulation model, $t$ = 180s



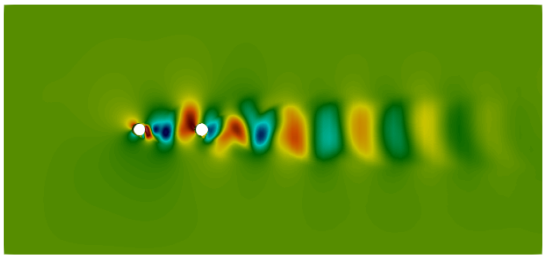
(b) The numerical simulation model, $t$ = 199.5s



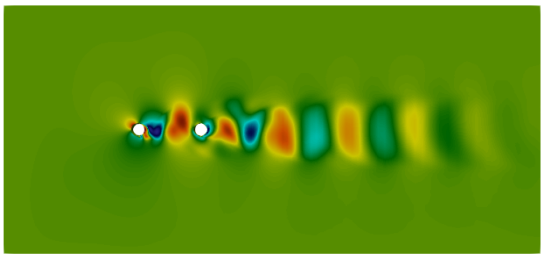
(c) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 180s



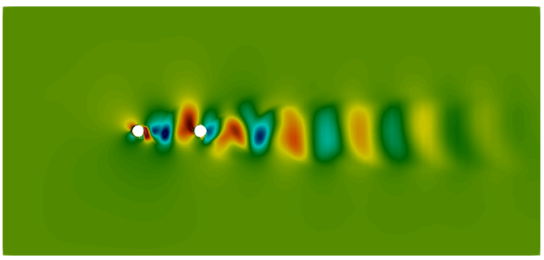
(d) The SAE-LSTM-bunch model with $m$ = 7, $t$ = 199.5s



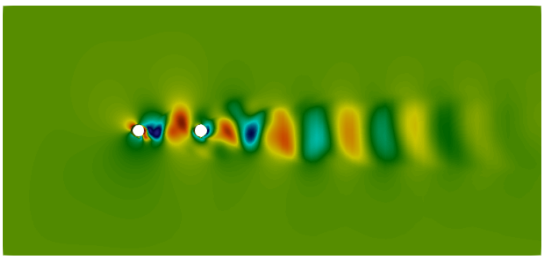
(e) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 180s



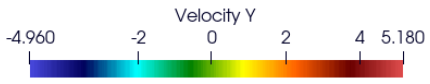
(f) The SAE-LSTM-bunch model with $m$ = 12, $t$ = 199.5s



(g) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 180s



(h) The SAE-LSTM-bunch model with $m$ = 35, $t$ = 199.5s



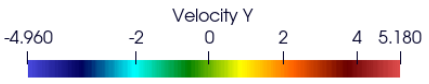
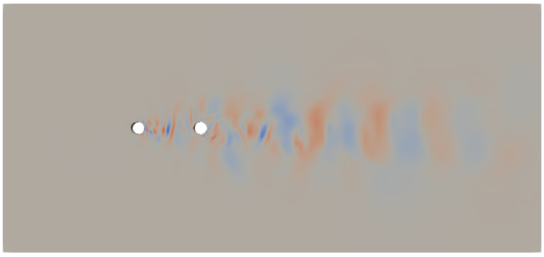
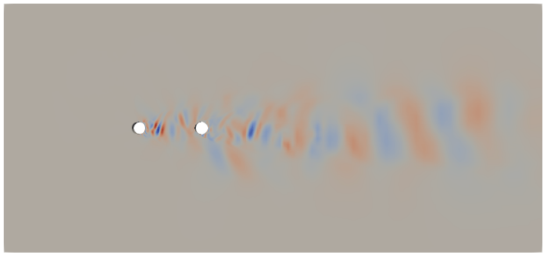



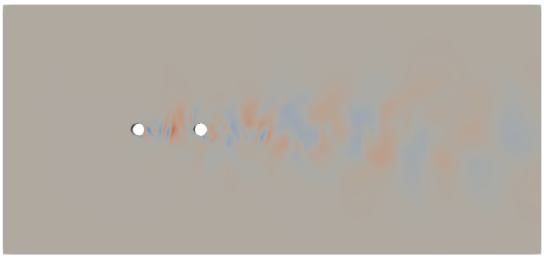

Figure 27: Comparison of velocity $v_y$ between the full-order solutions obtained from the numerical CFD model and the reduced-order solutions predicted from the SAE-LSTM-bunch models with different code lengths $m$.



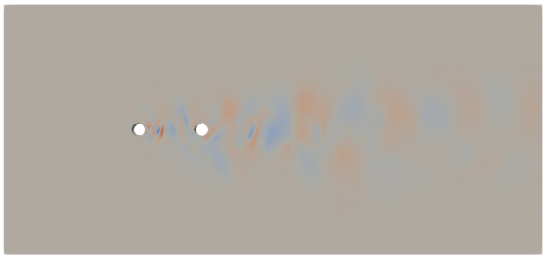
(a) The SAE-LSTM-bunch model with $m = 7$, $t = 180$s



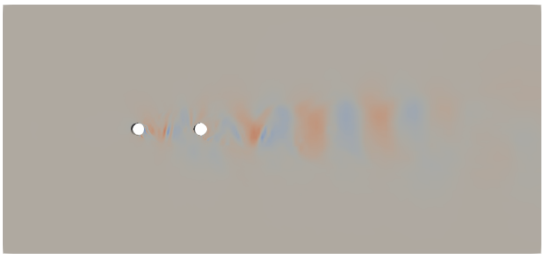
(b) The SAE-LSTM-bunch model with $m = 7$, $t = 199.5$s



(c) The SAE-LSTM-bunch model with $m = 12$, $t = 180$s

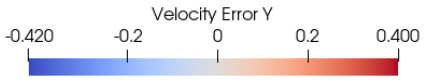


(d) The SAE-LSTM-bunch model with $m = 12$, $t = 199.5$s



(e) The SAE-LSTM-bunch model with $m = 35$, $t = 180$s

(f) The SAE-LSTM-bunch model with $m = 35$, $t = 199.5$s



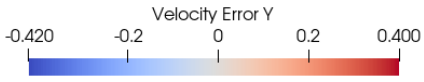
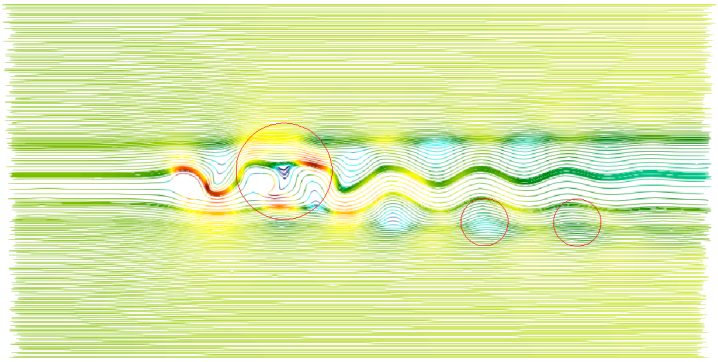
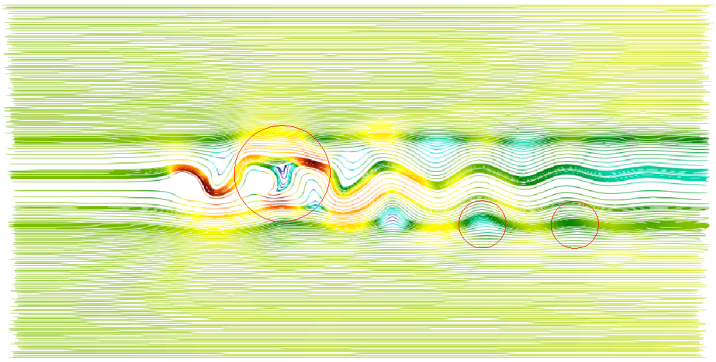




Figure 28: The absolute errors of the velocity component $v_y$ predicted from the SAE-LSTM-bunch models with different code lengths $m$.

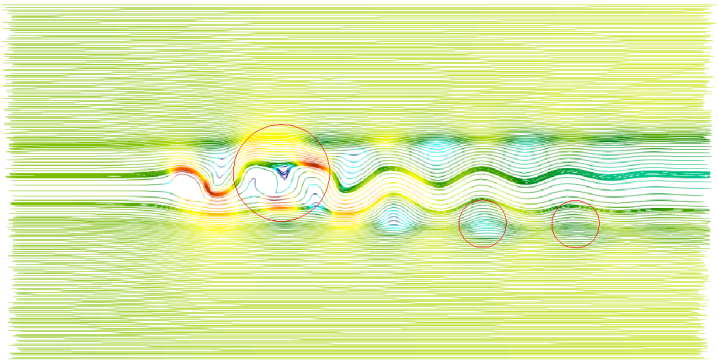
To furnish a more nuanced comprehension of turbulent flow dynamics, the streamlines corresponding to both full-order solutions from the CFD model and the reduced-order solutions predicted by the SAE-LSTM-bunch models at $t = 199.5$s are depicted in Figure 29. It becomes increasingly evident that the divergence between the full-order and reduced-order solutions lessens significantly with an escalation in code length $m$. When this code length ascends to 35, there exists virtually no perceptible distinction between these two sets of solutions. These revelations illuminate the formidable capability of the proposed SAE-LSTM-based ROM methodology in astutely capturing the intricate characteristics inherent to turbulent fluid flows.



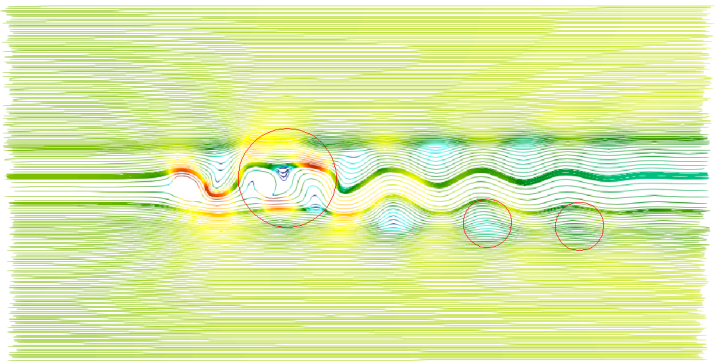
(a) The numerical simulation model



(b) The SAE-LSTM-bunch model with $m = 7$



(c) The SAE-LSTM-bunch model with $m = 12$



(d) The SAE-LSTM-bunch model with $m = 35$

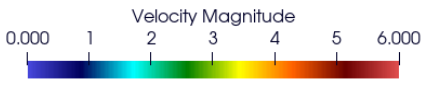
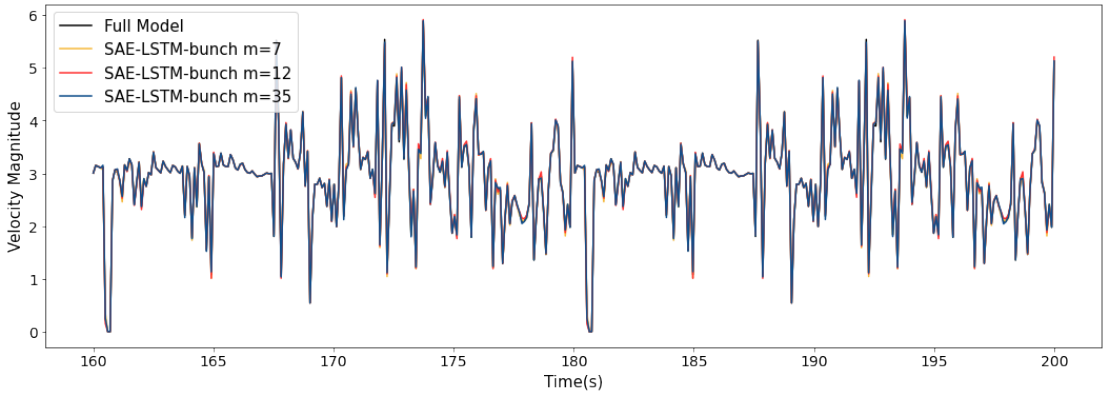
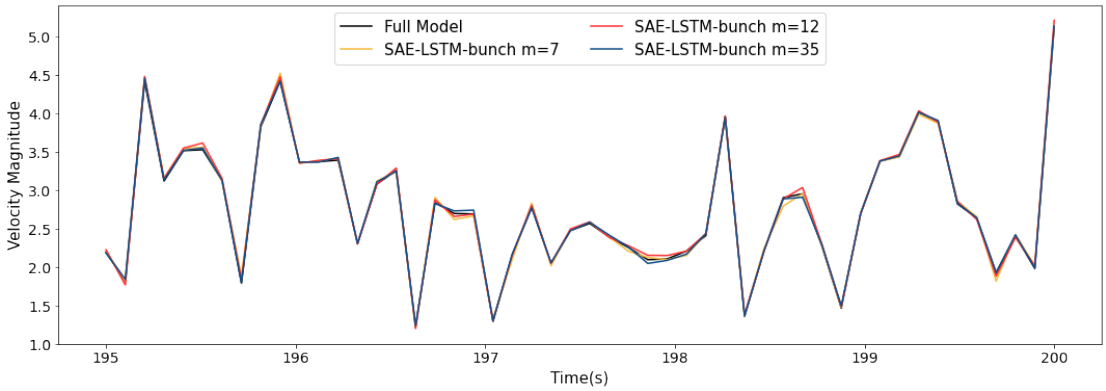



Figure 29: The streamlines derived from the full-order solutions and the reduced-order solutions at $t = 199.5$s.




(a) Predicted results of velocity magnitude during the time period from $t$ = 160.1 to 200.0s




(b) Predicted results of velocity magnitude during the time period from $t$ = 195.0 to 200.0s

Figure 30: The reduced-order solutions of velocity magnitude at the point ($x$ = 13.9708 m, $y$ = 11.154182 m, $z$ = 1.073312 m) over time, where the SAE-LSTM-bunch models with different code lengths $m$ are used for extrapolation prediction.

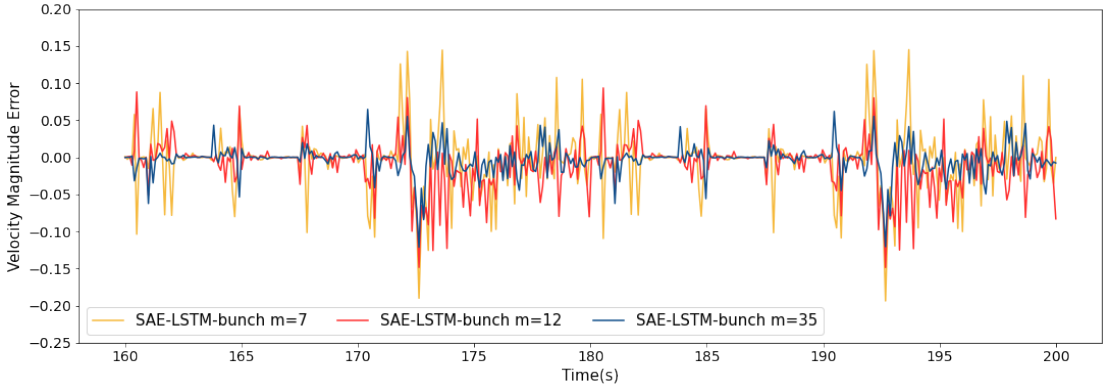



Figure 31: The absolute errors of the reduced-order solutions of velocity magnitudes at the point ($x$ = 13.9708 m, $y$ = 11.154182 m, $z$ = 1.073312 m) over time, where the SAE-LSTM-bunch models with different code lengths $m$ are used for extrapolation prediction.

Moreover, a distinctive nodal point situated at ($x$ = 13.9708 m, $y$ = 11.154182 m, $z$ = 1.073312 m) has been meticulously chosen from the computational domain for in-depth analysis, with its location denoted by the red node in Figure 18. The velocity magnitude solutions $|v|$ derived from both the comprehensive full-order numerical model and the sophisticated reduced-order models (the SAE-LSTM-bunch models) are depicted in Figure 30. At this pivotal point, the velocity magnitude showcases substantial temporal fluctuations, thereby underscoring the inherently turbulent characteristics of fluid flow dynamics. When the code length $m \geq 12$, the velocity magnitude trajectories obtained from reduced-order solutions closely conform to those presented by the reference curve

originating from full-order solutions. In Figure 31, absolute errors have been computed to assess the predictive efficacy of these reduced-order models; it becomes increasingly apparent that prediction accuracy enhances as code length $m$ escalates. Upon reaching a threshold where $m$ attains 35, the mean magnitude of absolute errors over time intervals spanning from 160.1s to 200s dips below 0.05. These observations ascertain that the proposed SAE-LSTM-based ROM methodology possesses remarkable capacity for constructing dependable reduced-order representations that accurately embody turbulent fluid dynamics when an appropriate code length $m$ is judiciously selected.

## 6. Discussion and conclusion

### *6.1. Discussion*

In the above section, the proposed SAE-LSTM-based ROM method is validated via two representative case studies. The results demonstrate its capability to construct accurate and reliable reduced-order models by capturing complicated turbulent flow dynamics. Given the goal of model order reduction, computational efficiency emerges as a crucial metric for assessing the performance of this innovative non-intrusive ROM approach. Table 7 outlines the computational costs associated with various numerical predictive models, including the recently developed SAE-DMD-based model [34] and the newly introduced SAE-LSTM-based models for each test case. In this study, numerical simulation models are generated using open-source solvers, specifically *Fluidity* and *OpenFoam*. Meanwhile, both the SAE-LSTM and SAE-DMD models are implemented in *Python*. All relevant programs are executed on a high-performance computing desktop equipped with 13$^{\text{th}}$ Gen Intel® Core® i5-13400 processors (2.50 GHz base frequency and 4.60 GHz maximum turbo frequency) and 128 GB of RAM.

Table 7: Performance comparisons between different predictive models in terms of computational cost (and memory) requirement

| **Test case** | **Predictive models** | **Number of nodes** | **Order reduction time cost** (s)* | **Time cost** (s) | |
|---|---|---|---|---|---|
| | | | | **Offline training**⊛ | **Online prediction**⁕ |
| Case 1 | Numerical simulation model⋆ | 84,180 | - - | - - | 7.100 |
| | SAE-DMD model ($r = 45$) | | 281.490 | - - | $3.854 \times 10^{-5}$ |
| | SAE-LSTM-bunch model ($m = 45$) | | 281.490 | 42.285 | $1.687 \times 10^{-2}$ |
| | SAE-LSTM-bunch model ($m = 10$) | | 72.550 | 40.157 | $1.645 \times 10^{-2}$ |
| | SAE-LSTM-bunch model ($m = 5$) | | 44.100 | 38.121 | $1.633 \times 10^{-2}$ |
| Case 2 | Numerical simulation model† | 55, 268 | - - | - - | 3.500 |
| | SAE-DMD model ($r = 35$) | | 48.410 | - - | $1.800 \times 10^{-5}$ |
| | SAE-LSTM-bunch model ($m = 35$) | | 48.410 | 7.122 | $1.668 \times 10^{-2}$ |
| | SAE-LSTM-bunch model ($m = 12$) | | 18.870 | 7.048 | $1.655 \times 10^{-2}$ |
| | SAE-LSTM-bunch model ($m = 7$) | | 13.040 | 6.822 | $1.639 \times 10^{-2}$ |

* The time cost of SAE model training for 10 epochs;
⊛ The time cost of LSTM-bunch model training for 100 epochs;
⁕ The time cost of predicting one snapshot at each time step;
⋆ The finite element model constructed by using *Fluidity*;
† The finite volume model constructed by using *OpenFoam*.

Compared to the full-order numerical models, the SAE-LSTM-bunch models can rapidly perform online predictions of high-dimensional quantities (e.g., velocity component fields), exhibiting a greater accuracy than the SAE-DMD models, albeit at a slightly slower speed. This advantage in prediction time becomes even more significant when addressing complex and large-scale problems. Nevertheless, the proposed SAE-LSTM-bunch method requires approximately 80 seconds per 10 epochs for training the SAE models aimed at nonlinear dimensionality reduction. The efficiency of this training process is heavily dependent on the computational power available, particularly on GPU capabilities. High-performance GPU-based parallel computing can significantly expedite this training process by hundreds or even thousands of times. Furthermore, once adequately trained, the SAE models can be stored for future use, thereby conserving computational resources over the long term.

### *6.2. Conclusion*

The primary contribution of this work is to present a novel data-assisted computational framework that employs deep neural networks for nonlinear reduced-order modeling (ROM) of engineering turbulent flows. This

is accomplished through the effective integration of a Stacked Auto-Encoder (SAE) network and a Long Short-Term Memory (LSTM) network. Specifically, the SAE network facilitates nonlinear dimensionality reduction and feature extraction, while the resulting latent features are subsequently input into the LSTM network to enable predictive ROM for turbulent fluid dynamics. The proposed SAE-LSTM-based ROM methodology has been applied to two representative turbulent flow problems. Its performance has been systematically compared with that of the recently developed SAE-DMD method [34] as well as high-fidelity numerical simulations via computational fluid dynamics, focusing on modeling accuracy and forecast efficiency. These case studies demonstrate that combining SAE with LSTM for non-intrusive ROM can significantly enhance computational speed and reduce computational complexity while accurately preserving key nonlinear characteristics inherent in turbulent flow dynamics. This innovative approach shows considerable promise in addressing the computational challenges associated with high-resolution numerical modeling applicable to complex large-scale flow scenarios. Future research will involve extending this new methodology for parametric model order reduction of large-scale turbulent fluid flows, aiming to achieve rapid predictions concerning engineering turbulence under varying initial conditions, fluid properties, geometries, or dynamic boundary conditions.

### Data availability statement

All data, codes, and numerical models that support the findings of this study are available on GitHub through the following link: https://github.com/ihorizon2018/SAE-LSTM-ROM-for-Turbulence.


### Acknowledgments

The authors would like to acknowledge the support received from the Swansea University FSE IMPACT Fund, the EPSRC Grant (PURIFY: $EP/V000756/1$), the Shanghai Universities-Class I Top Discipline Plan and the Shanghai Municipal Science and Technology Major Project (No. 2021SHZDZX0100), and the National Key *R&D* Program of China (No. 2022YFE0208000).


### Declarations

The authors declare that they have no conflict of interest in this paper.

## References


[1] Philippe R Spalart. Comments on the feasibility of les for wings and on the hybrid rans/les approach. In *Proceedings of the First AFOSR International Conference on DNS/LES, 1997*, pages 137–147, 1997.

[2] Parviz Moin and Krishnan Mahesh. Direct numerical simulation: a tool in turbulence research. *Annual Review of Fluid Mechanics*, 30(1):539–578, 1998.

[3] 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.

[4] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1993.

[5] Dunhui Xiao, Fangxin Fang, Christopher Pain, and Guangwei Hu. Non-intrusive reduced-order modelling of the navier–stokes equations based on rbf interpolation. *International Journal for Numerical Methods in Fluids*, 79(11):580–595, 2015.

[6] Zheng Wang, Dunhui Xiao, Fangxin Fang, Rajesh Govindan, Christopher C Pain, and Yike Guo. Model identification of reduced order fluid dynamics systems using deep learning. *International Journal for Numerical Methods in Fluids*, 86(4):255–268, 2018.

[7] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

[8] Philip Holmes. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 2012.

[9] Jinlong Fu, Dunhui Xiao, Rui Fu, Chenfeng Li, Chuanhua Zhu, Rossella Arcucci, and Ionel M Navon. Physics-data combined machine learning for parametric reduced-order modelling of nonlinear dynamical systems in small-data regimes. *Computer Methods in Applied Mechanics and Engineering*, 404:115771, 2023.

[10] 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.

[11] Clarence W Rowley and Scott TM Dawson. Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49:387–417, 2017.

[12] Sebastian Grimberg, Charbel Farhat, Radek Tezaur, and Charbel Bou-Mosleh. Mesh sampling and weighting for the hyperreduction of nonlinear petrov–galerkin reduced-order models with local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 122(7):1846–1874, 2021.

[13] S Ares de Parga, JR Bravo, JA Hernández, R Zorrilla, and Riccardo Rossi. Hyper-reduction for petrov–galerkin reduced order models. *Computer Methods in Applied Mechanics and Engineering*, 416:116298, 2023.

[14] Bernd R Noack, Konstantin Afanasiev, Marek Morzyński, Gilead Tadmor, and Frank Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363, 2003.

[15] Clarence W Rowley, Tim Colonius, and Richard M Murray. Model reduction for compressible flows using pod and galerkin projection. *Physica D: Nonlinear Phenomena*, 189(1-2):115–129, 2004.

[16] Michele Girfoglio, Annalisa Quaini, and Gianluigi Rozza. A pod-galerkin reduced order model for the navier–stokes equations in stream function-vorticity formulation. *Computers & Fluids*, 244:105536, 2022.

[17] Sibo Cheng, Jianhua Chen, Charitos Anastasiou, Panagiota Angeli, Omar K Matar, Yi-Ke Guo, Christopher C Pain, and Rossella Arcucci. Generalised latent assimilation in heterogeneous reduced spaces with machine learning surrogate models. *Journal of Scientific Computing*, 94(1):11, 2023.

[18] Luca Pegolotti, Martin R Pfaller, Natalia L Rubio, Ke Ding, Rita Brugarolas Brufau, Eric Darve, and Alison L Marsden. Learning reduced-order models for cardiovascular simulations with graph neural networks. *Computers in Biology and Medicine*, 168:107676, 2024.

[19] Helin Gong, Sibo Cheng, Zhang Chen, Qing Li, César Quilodrán-Casas, Dunhui Xiao, and Rossella Arcucci. An efficient digital twin based on machine learning svd autoencoder and generalised latent assimilation for nuclear reactor physics. *Annals of nuclear energy*, 179:109431, 2022.

[20] Jiaxin Wu, Min Luo, Dunhui Xiao, Christopher C Pain, and Boo Cheong Khoo. Koopman dynamic-oriented deep learning for invariant subspace identification and full-state prediction of complex systems. *Computer Methods in Applied Mechanics and Engineering*, 429:117071, 2024.

[21] Alberto Solera-Rico, Carlos Sanmiguel Vila, Miguel Gómez-López, Yuning Wang, Abdulrahman Almashjary, Scott TM Dawson, and Ricardo Vinuesa. $\beta$-variational autoencoders and transformers for reduced-order modelling of fluid flows. *Nature Communications*, 15(1):1361, 2024.

[22] Souvik Chakraborty and Nicholas Zabaras. Efficient data-driven reduced-order models for high-dimensional multiscale dynamical systems. *Computer Physics Communications*, 230:70–88, 2018.

[23] Caili Zhong, Sibo Cheng, Matthew Kasoar, and Rossella Arcucci. Reduced-order digital twin and latent data assimilation for global wildfire prediction. *Natural hazards and earth system sciences*, 23(5):1755–1768, 2023.

[24] Jiaxin Wu, Dunhui Xiao, and Min Luo. Deep-learning assisted reduced order model for high-dimensional flow prediction from sparse data. *Physics of Fluids*, 35(10), 2023.

[25] X Wu, P Gan, J Li, F Fang, X Zou, CC Pain, X Tang, J Xin, Z Wang, and J Zhu. A long short-term memory neural network-based error estimator for three-dimensional dynamically adaptive mesh generation. *Physics of Fluids*, 35(10), 2023.

[26] Paolo Conti, Giorgio Gobat, Stefania Fresca, Andrea Manzoni, and Attilio Frangi. Reduced order modeling of parametrized systems through autoencoders and sindy approach: continuation of periodic solutions. *Computer Methods in Applied Mechanics and Engineering*, 411:116072, 2023.

[27] Anastasiia Nazvanova, Muk Chen Ong, and Guang Yin. A data-driven reduced-order model based on long short-term memory neural network for vortex-induced vibrations of a circular cylinder. *Physics of Fluids*, 35(6), 2023.

[28] Ashton Hetherington, Adrián Corrochano, Rodrigo Abadía-Heredia, Eneko Lazpita, Eva Muñoz, Paula Díaz, Egoitz Maiora, Manuel López-Martín, and Soledad Le Clainche. Modelflows-app: data-driven post-processing and reduced order modelling tools. *Computer Physics Communications*, 301:109217, 2024.

[29] Clément Scherding, Georgios Rigas, Denis Sipp, Peter J Schmid, and Taraneh Sayadi. An adaptive learning strategy for surrogate modeling of high-dimensional functions-application to unsteady hypersonic flows in chemical nonequilibrium. *Computer Physics Communications*, page 109404, 2024.

[30] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

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

[32] Francisco J Gonzalez and Maciej Balajewicz. Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems. *arXiv preprint arXiv:1808.01346*, 2018.

[33] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. Vampnets for deep learning of molecular kinetics. *Nature Communications*, 9(1):5, 2018.

[34] Chuanhua Zhu, Dunhui Xiao, Jinlong Fu, Yuntian Feng, Rui Fu, and Jinsheng Wang. A data-driven computational framework for non-intrusive reduced-order modelling of turbulent flows passing around bridge piers. *Ocean Engineering*, 308:118308, 2024.

[35] R Fu, D Xiao, IM Navon, F Fang, Liang Yang, C Wang, and S Cheng. A non-linear non-intrusive reduced order model of fluid flow by auto-encoder and self-attention deep learning methods. *International Journal for Numerical Methods in Engineering*, 124:3087–3111, 2023.

[36] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[37] Hamidreza Eivazi, Hadi Veisi, Mohammad Hossein Naderi, and Vahid Esfahanian. Deep neural networks for nonlinear model order reduction of unsteady flows. *Physics of Fluids*, 32(10), 2020.

[38] 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.

[39] Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent space physics: Towards learning the temporal evolution of fluid flow. In *Computer Graphics Forum*, volume 38, pages 71–82. Wiley Online Library, 2019.

[40] Pantelis R Vlachas, Wonmin Byeon, Zhong Y Wan, Themistoklis P Sapsis, and Petros Koumoutsakos. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213):20170844, 2018.

[41] Mohammed H Qais, Seema Kewat, Ka Hong Loo, Cheung-Ming Lai, and Aldous Leung. Lstm-based stacked autoencoders for early anomaly detection in induction heating systems. *Mathematics*, 11(15):3319, 2023.

[42] Jiri Blazek. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[43] 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.

[44] Jinlong Fu, Shaoqing Cui, Song Cen, and Chenfeng Li. Statistical characterization and reconstruction of heterogeneous microstructures using deep neural network. *Computer Methods in Applied Mechanics and Engineering*, 373:113516, 2021.

[45] Greg Van Houdt, Carlos Mosquera, and Gonzalo Nápoles. A review on the long short-term memory model. *Artificial Intelligence Review*, 53(8):5929–5955, 2020.

[46] CC Pain, MD Piggott, AJH Goddard, F Fang, GJ Gorman, DP Marshall, MD Eaton, PW Power, and CRE De Oliveira. Three-dimensional unstructured mesh ocean modelling. *Ocean Modelling*, 10(1-2):5–33, 2005.

[47] Hrvoje Jasak, Aleksandar Jemcov, Zeljko Tukovic, et al. Openfoam: A c++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, volume 1000, pages 1–20, 2007.