Engineering with Computers (2024) 40:45–60
https://doi.org/10.1007/s00366-023-01782-2

ORIGINAL ARTICLE

Parametric model order reduction by machine learning for fluid–structure interaction analysis

SiHun Lee[1] · Kijoo Jang[1] · Sangmin Lee[1] · Haeseong Cho[2] · SangJoon Shin[1,3]

Received: 9 August 2022 / Accepted: 30 December 2022 / Published online: 10 January 2023

Abstract

An improved nonintrusive parametric model order reduction (pMOR) approach is proposed for the flow field interpolation regarding fluid–structure interaction (FSI) objects. Flow field computation using computational fluid dynamics (CFD) requires excessive computational time and memory. Nonintrusive and data-driven MOR schemes have been proposed to overcome such limitations. The present methodology is implemented by both proper orthogonal decomposition (POD) and a modified Nouveau variational autoencoder (mNVAE). POD attempts to reduce the number of degrees of freedom (DOFs) on the precomputed series of the full-order model parametric result. The reduced DOF yields parametrically independent reduced bases and dependent coefficients. Then, mNVAE is employed for the interpolation of POD coefficients, which will be combined with POD modes for parametrically interpolated flow field generation. The present approach is assessed on the benchmark problem of a two-dimensional plunging airfoil and the highly nonlinear FSI phenomenon of the limit cycle oscillation. The comparison was executed against other POD-based generative neural network approaches. The proposed methodology demonstrates applicability on highly nonlinear FSI objects with improved accuracy and efficiency.

Keywords Nonintrusive parametric reduced-order modeling · Machine learning · Fluid–structure interaction · Proper orthogonal decomposition · Variational autoencoder

1 Introduction

Advancements in computing hardware and software have enabled a comprehensive analysis of complicated flow fields using computational fluid dynamics (CFD). However, the high nonlinearity inherent in both Navier–Stokes and Euler equations leads to an extensive number of degrees of freedom (DOF), resulting in a huge amount of computational time and memory. To overcome such limitations, various model order reductions (MOR) have been proposed.

MOR aims at converting a higher dimensional representation into a lower dimensional one while preserving the main feature of the original. Over the past decades, MOR schemes, such as proper orthogonal decomposition (POD) [1, 2], balanced method [3], and empirical Gramian [4], have been proposed. POD is one of the most widely used approaches, owing to its robustness and optimality [5]. POD, also known as the Karhunen–Loéve theorem or principal component analysis, was introduced by Lumley [1] to analyze a turbulent flow field. This was later improved by Sirovich [2] using the snapshot method. POD extracts the reduced basis, referred to as the POD mode, using singular value decomposition (SVD) on the snapshot. An efficient surrogate representation is then constructed by combining a finite, small number of POD modes that will contain the majority of properties.

✉ SangJoon Shin
ssjoon@snu.ac.kr

SiHun Lee
leesihun@snu.ac.kr

Kijoo Jang
prastins@snu.ac.kr

Sangmin Lee
sj7714@snu.ac.kr

Haeseong Cho
hcho@jbnu.ac.kr

[1] Department of Aerospace Engineering, Seoul National University, Seoul 08226, Republic of Korea

[2] Department of Aerospace Engineering, Jeonbuk National University, Jeonju 54896, Republic of Korea

[3] Institute of Advanced Aerospace Technology, Seoul National University, Seoul 08226, Republic of Korea

There are two approaches to MOR: intrusive and nonintrusive. POD in an intrusive MOR is combined with Galerkin projection [6–8], where the governing equation is projected onto the reduced subspace. Because of the dependence on the governing equation, multiparametric analysis becomes tedious when the specific CFD algorithm is not explicitly established [9]. To address these difficulties, a nonintrusive MOR (NIMOR) was proposed. NIMOR is a purely data-driven approach based on high-fidelity analysis results without knowledge of the governing equation. Similar to the intrusive method, POD for NIMOR extracts reduced bases. However, additional treatment will be performed in the POD mode or coefficient to approximate the full order model (FOM), rather than projecting the higher dimensional representation.

NIMOR utilizes the latest developments in machine-learning techniques. Gaussian process regression (GPR) has been used by several researchers [10, 11] owing to its simplicity and decent performance. However, GPR is known to be more inaccurate than the latest machine-learning techniques [12]. An artificial neural network (ANN) was employed for NIMOR, including a simple multilayer perceptron [13–17], long short-term memory networks [18–20], and generative adversarial networks [21, 22].

Attempts to replace POD as a MOR method are found. One of the most popular choices is the autoencoder for nonlinear MOR [20, 23–25]. Extension of the conventional autoencoder, such as SINDy autoencoder [26], and combined POD-autoencoder methodology [27] were investigated. Using the autoencoders, parametric MOR was attempted by identifying its latent space [28, 29]. Other methods include the physics-informed neural networks (PINN). In PINN, neural networks are designed to be universal function approximators to describe the partial differential equations [30, 31].

In this study, an ANN exploiting unsupervised learning method based on the modified Nouveau variational autoencoder (mNVAE) was adopted. Unlike supervised learning method, an ANN exploiting unsupervised learning method generates the result based on the pattern and feature of the given dataset, which is learned without a label. Autoencoder [32], VAE [33], and generative adversarial network (GAN) [34] are the most recognized unsupervised learning approaches. Several variations of these networks have been proposed to improve the performance. Based on pattern searching and clustering, they are applied to diverse tasks, including dimensionality reduction. In particular, they are widely used for image denoising, restoration, generating novel realistic data via interpolation, super-resolution of images, and anomaly detection [35–40].

Among generative neural networks, VAE is known to be much more stable during training than GAN. During the training of GAN, phenomena such as mode collapse and imbalance of the generator/critic network performance may occur. A situation referred to as posterior collapse occurs in vanilla VAE. This issue was alleviated by applying a technique known as Kullback–Leibler divergence (KL divergence) annealing [41–44]. Deep hierarchical VAE such as Ladder VAE (LVAE) [42] and Nouveau VAE (NVAE) [44] were proposed to stack multiple layers in VAE for a better formulation. These approaches force the encoder and decoder networks in VAE to produce analogous distribution results.

Several attempts have been made to apply unsupervised learning method to MOR. Phillips et al. [45] developed a reduced-order model (ROM) known as the SVD-autoencoder in which SVD was operated on the input of the autoencoder. Xu et al. [46] developed a multistage MOR for parametric estimation of the flow field based on a coarsely constructed overset grid and convolutional autoencoder. However, autoencoders are known to exhibit inferior performance because of their sparse latent space and wide range of latent variables [47]. Lee et al. [21] employed WGAN-GP and POD for unobserved parametric values of the flow field. Although WGAN-GP is known for generating "sharper" details, it is also known for slower and unstable training. In contrast, VAE is characterized by stable and fast training. The authors found that VAE would also be capable of generating "sharp" details if the dataset was one-dimensional (1D). Cheng et al. [48] developed a hybrid VAE-GAN for parameterized flow analysis. However, its applications were limited to a small number of DOFs and VAE was not optimal for MOR use when the number of DOF exceeded a few hundred thousand. The reconstruction error of VAE will become significantly large, and stability during training will degrade owing to the regularizing term, KL divergence.

In this study, a combination of POD and variation to VAE, referred to as the modified NVAE (mNVAE), was adopted to construct a nonintrusive parametric MOR (pMOR) for the analysis of FSI. The present methodology is henceforth referred to as POD-mNVAE. POD is used to reduce the order of the higher dimensional representation, and mNVAE will be used for interpolation. Two examples were examined: the flow field around a plunging airfoil, flow around a highly nonlinear FSI phenomenon, and limit cycle oscillation (LCO) of an airfoil. The computational time and accuracy were evaluated for both examples, and a comparison with those previously proposed was performed.

The remainder of this paper is organized as follows. Section 2 presents the formulation of POD and modification to its snapshot matrix for parametric information. The formulation of VAE and modifications are provided in Sect. 3. Section 4 consists of the assumptions and procedures to execute POD-mNVAE. Section 5.1 demonstrates the applications of POD-mNVAE on two FSI objects. A comparison against previously proposed POD-based ANN methods is

also presented. The conclusion and a relevant discussion are presented in Sect.6.

2 Proper orthogonal decomposition (POD)

The proposed scheme first constructs a lower dimensional representation of FOM using a reduced-order basis. The snapshot matrix is an ensemble of FOM analysis results. On the snapshot matrix, POD extracts the characteristics of a higher dimensional FOM using SVD. Using SVD, the POD modes are collected based on the dynamic energy ratio. Then, the POD modes are used as the basis vectors for MOR.

A snapshot matrix with a size of $N \times S$ is expressed in Eq. (1). Here, v denotes the physical variable of interest from the flow field, N corresponds to the number of DOF, and S corresponds to the number of time steps as follows:

$$\boldsymbol{W} = \begin{bmatrix} v_1^{(1)} & v_1^{(2)} & \cdots & v_1^{(S-1)} & v_1^{(S)} \\ v_2^{(1)} & v_2^{(2)} & \cdots & v_2^{(S-1)} & v_2^{(S)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ v_{N-1}^{(1)} & v_{N-1}^{(2)} & \cdots & v_{N-1}^{(S-1)} & v_{N-1}^{(S)} \\ v_N^{(1)} & v_N^{(2)} & \cdots & v_N^{(S-1)} & v_N^{(S)} \end{bmatrix} \tag{1}$$

The snapshot matrix was modified to accommodate parametric variations. The snapshot matrices for each parametric value are concatenated in the row direction. A single appended snapshot matrix, W_{total} will be constructed, and it contains parametric as well as spatio-temporal information. Equation (2) represents the snapshot matrix modified by parameter N_p in which the second superscript denotes the parameter as follows:

$$\boldsymbol{W_{total}} = \left[\begin{bmatrix} v_1^{(1,1)} & \cdots & v_1^{(S,1)} \\ \vdots & \ddots & \vdots \\ v_N^{(1,1)} & \cdots & v_N^{(S,1)} \end{bmatrix} \cdots \begin{bmatrix} v_1^{(1,N_p)} & \cdots & v_1^{(S,N_p)} \\ \vdots & \ddots & \vdots \\ v_N^{(1,N_p)} & \cdots & v_N^{(S,N_p)} \end{bmatrix} \right] \tag{2}$$

After the construction of the snapshot matrix was completed, SVD was performed. Then, the POD modes ϕ_i were extracted using the process in Eq. (3) as follows:

$$\begin{aligned} \boldsymbol{C} &= \boldsymbol{W}^T \boldsymbol{W} \\ \boldsymbol{C}\boldsymbol{V} &= \lambda \boldsymbol{V} \\ \phi_i &= \frac{1}{\sqrt{\lambda_i}} \boldsymbol{W} V_i \end{aligned} \tag{3}$$

The number of POD modes to be considered for MOR is determined by the energy ratio λ_i. The accumulated energy ratio E is defined by Eq. (4). The number of POD modes, N_m, is determined by the total energy considered, E:

$$E = \frac{\sum_{i=1}^{N_m} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \tag{4}$$

The POD coefficients for time t and parameter p were obtained by projecting the FOM solutions onto the space generated by the POD modes. The variable of interest is then generated by the combination of POD coefficients and modes. Then, the original variable of interest, $v^* \approx v$, is obtained by the sum of the averaged value, $\bar{v}$, and perturbations, $\hat{v}$, as expressed in Eq. (6) as follows:

$$a_i(t,p) = \boldsymbol{W}\boldsymbol{\phi}_i^T \tag{5}$$

$$\begin{aligned} v^*(x,t,p) &= \bar{v}(x) + \hat{v}(x,t,p) \\ &= \bar{v}(x) + \sum_{i=1}^{N_m} a_i(t,p)\phi_i(x) \end{aligned} \tag{6}$$

POD modes $\Phi_i(x)$ in Eq. (6) contain spatial information that is invariant with respect to the parameter and time, whereas POD coefficients, $a_i(t,p)$, contain the parametric and temporal information invariant with respect to the location. Naturally, POD coefficients comprise temporal information appended with respect to the parameter. These are decomposed later for mNVAE training.

3 Modified Nouveau variational autoencoder (mNVAE)

3.1 Variational autoencoder (VAE)

The current neural network, mNVAE, is an improved version of the VAE. VAE largely comprises two components: an encoder and decoder. The output of the encoder is expressed by the mean μ and the variance σ which will be used to generate the latent code, z. VAE aims to infer an intractable posterior distribution efficiently. As it is difficult to determine the true posterior, $p(z \mid x)$, VAE utilizes an approximate posterior, $q(z \mid x)$. Regarding the structure of VAE, $q(z \mid x)$ indicates an encoder or an inference network. $p(x \mid z)$ denotes the decoder network. A typical structure of VAE is found in Fig. 1.

The objective function of VAE was formulated as expressed in Eq. (7). The first term denotes the negative log-likelihood that operates as a reconstruction error. It is designed to minimize the discrepancy between the given input and the generated output. The second term is KL

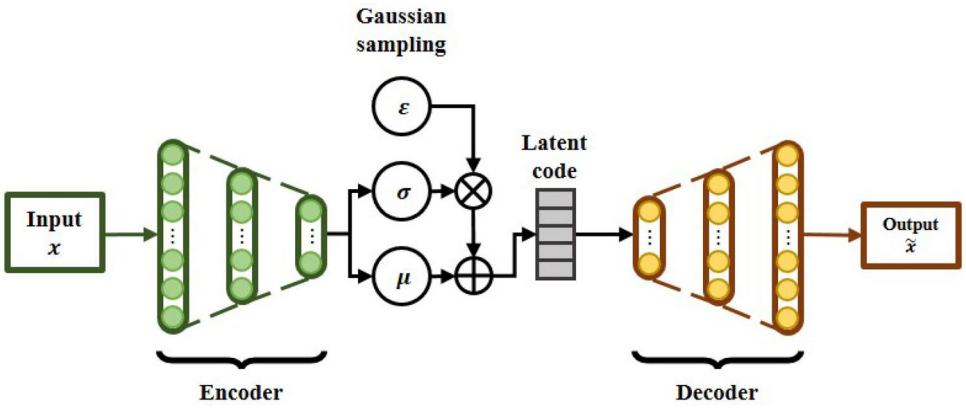


Fig. 1 Structure of a typical VAE

divergence, which is the distance between two distributions. KL divergence forces the approximate posterior to become closer to the prior, $p(z)$. During the optimization process, KL divergence acts as a regularization term:

$$\min[-\mathbb{E}_{q(z|x)}[\log p(x \mid z)] + D_{KL}(q(z \mid x)\|p(z))] \tag{7}$$

Usually, KL divergence term in the loss function can be integrated analytically [33]. The reconstructed error on the other hand, is not directly differentiable since z is sampled from μ and σ. To enforce it to be differentiable, reparameterization trick was adopted for the latent code sampling. First, Gaussian-sampled random noise, ϵ was introduced to the latent code. The latent code z, was formulated by μ, σ, and ϵ, so that, $z = \mu + (\sigma \times \epsilon)$. The new z enabled reconstruction loss to be differentiable by Monte Carlo method. Thus, the reconstruction loss would be backpropagated.

3.2 Hierarchical network

The conventional VAE is constrained to a shallow model with only a few layers of stochastic latent variables. Such constraints result in a restrictive mean-field approximation and degrade the VAE performance [42]. Deep hierarchical VAE such as LVAE [42] and NVAE [44] have been proposed to overcome these limitations. The ladder network connects the inference and generative networks via bidirectional information transfer [49]. The network structure is shown in Fig. 2.

The latent variables in the deep hierarchical VAE are partitioned. The latent variables are sampled from the layers in both bottom-up and top-down network as shown in Fig. 2. Especially, to generate ith latent variable, z_i, ith layer in the top-down network is used. The ith layer in the top-down network originates from predecessing latent variable, z_{i+1}, thus the network contains top-down information. Similarly, z_i also requires ith layer in the bottom-up network.

Since both bottom-up and top-down layers are used to sample latent variables in each layer, bidirectional information will be shared. Conventional deep VAE lack long-range correlations as first few encoder layers affect very little on the output. For the ladder VAEs, even the first encoder layer will be used to sample latent variable enabling better quality latent variables.

The latent variables are separated into L groups: $z = \{z_1, z_2, z_3, \cdots, z_L\}$. The prior and posterior can be fully factorized, as expressed in Eq. (8) and subsequently specified as expressed in Eq. (9), using a Gaussian distribution for each group:

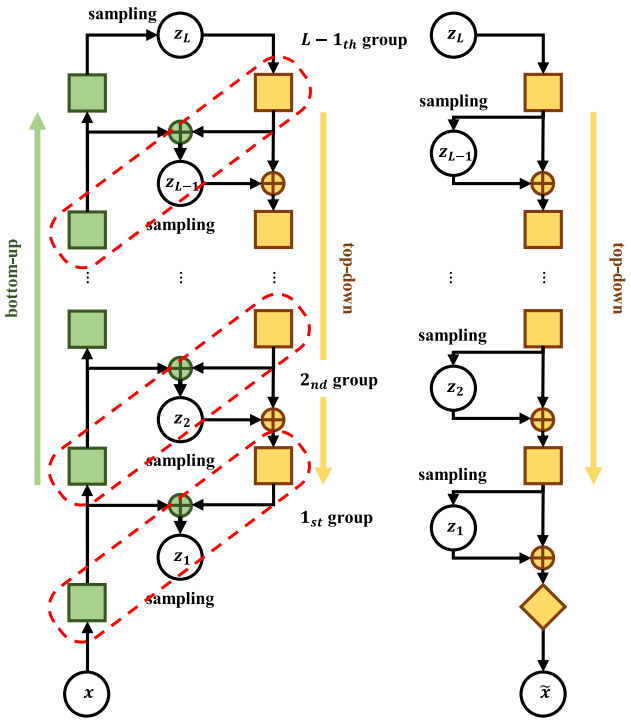


Fig. 2 Structure of deep hierarchical VAE

$$
\begin{aligned}
p(z) &= p(z_L)\prod_{i=1}^{L-1} p(z_i \mid z_{i+1}) \\
q(z \mid x) &= q(z_1 \mid x)\prod_{i=2}^{L} q(z_i \mid z_{i-1})
\end{aligned}
\tag{8}
$$

$$
\begin{aligned}
p(z_i \mid z_{i+1}) &= \mathcal{N}(z_i \mid \mu(z_{i+1}), \sigma^2(z_{i+1})) \\
p(z_L) &= \mathcal{N}(z_L \mid 0, I) \\
q(z_i \mid z_{i-1}) &= \mathcal{N}(z_i \mid \mu(z_{i-1}), \sigma^2(z_{i-1})) \\
q(z_1 \mid x) &= \mathcal{N}(z_1 \mid \mu(x), \sigma^2(x))
\end{aligned}
\tag{9}
$$

From the factorized distribution, the objective is specified in Eq. (10) in which the KL divergence is obtained individually. As aforementioned, KL divergence will force the approximate posterior to approach the prior. Therefore, splitting the KL divergence into groups will lead to bidirectional information transfer between the inference and generative networks. A detailed explanation of the ladder network for the VAE can be found in [42]:

$$
\begin{aligned}
&\min[-\mathbb{E}_{q(z|x)}[\log p(x \mid z)] + D_{KL}(q(z \mid x)\|p(z)) \\
&+ \sum_{i=1}^{L-1} \mathbb{E}_{q(z_{<i}|x)}[D_{KL}(q(z_i \mid z_{<i}, x)\|p(z_i \mid z_{>i}))]]
\end{aligned}
\tag{10}
$$

3.3 Modified NVAE (mNVAE)

The present ANN, mNVAE, is a modified version of NVAE [44] for improved accuracy when used with a 1D transient dataset. The network comprises 1D convolutional layers for the temporal continuity of the transient dataset. Instead of the conventional binary cross-entropy KL divergence, which is widely used for VAE, a hybrid weighted mean squared error KL divergence loss function is considered. The hybrid weighted loss function empirically shows better results regarding continuous data. The loss function for mNVAE is expressed as in Eq. (11) as follows:

$$
\begin{aligned}
&\min_{\phi,\theta}[\alpha MSE(x, \tilde{x}) + \beta D_{KL}(q(z \mid x)\|p(z)) \\
&+ \sum_{i=1}^{L-1} \alpha MSE(x, \tilde{x})[\beta D_{KL}(q(z_i \mid z_{<i}, x)\|p(z_i \mid z_{>i}))]]
\end{aligned}
\tag{11}
$$

In Eq. (11), the reconstruction loss, $\mathbb{E}$, is replaced by the mean squared error, MSE, between the input and output datasets. α and β denote the weight functions of MSE and KL divergence losses, respectively. The weight ratio for each loss function was set as approximately 1,000 $\alpha : \beta_{target} \approx 1,000 : 1$. For the mNVAE, KL annealing [41] is used, and β is expressed as in Eq. (12) as follows:

$$
\beta = \begin{cases} 1 \times 10^{-4}\beta_{target} & \text{if epoch } < 0.3n_{epochs} \\ \beta_{target}\frac{epoch}{n_{epochs}} & \text{if epoch } > 0.3n_{epochs} \end{cases}
\tag{12}
$$

KL annealing prevents a posterior collapse in which some of latent variables become inactive. At the start of training, the weight for KL divergence will be quite small and will act like an autoencoder. Then, the weight will be increased gradually introducing regularization term, resulting in a VAE. For interpolation in the latent space, spherical linear interpolation (slerp) is considered. Gaussian sampling forms latent space of mNVAE into a multi-dimension hypersphere. Linear interpolation within a hypersphere usually results in poor interpolation quality, where the interpolated vector length is ignored. Instead, linear interpolation along the sphere will be performed. By slerp interpolation, arc length will be interpolated linearly.

4 Framework

The proposed scheme makes the following two assumptions:

1. A sufficient number of the samplings is performed within the parametric space.
2. Change in the physical dynamics should be semi-continuous, i.e., no divergence or shock within the parametric space.

These assumptions are adopted such that the combination of POD modes may be sufficient to represent the flow field. The second assumption signifies that there should not exist any drastic change in the physical properties with respect to the parameter. Discontinuity in the phenomenon will lead to incapability of using POD modes semi-universally within the parametric space. Since the proposed scheme does not interpolate POD modes, the physical properties will be semi-continuous to ensure accuracy of the interpolation. Otherwise, collected POD modes will not be sufficient to construct the target flow field. A further explanation is discussed in Lee et al. [21].

4.1 Present POD approach

The snapshot matrix for the variable of interest, $\boldsymbol{W}_{p_j}$, are collected from the FOM result for each parameter. The snapshot matrices for each parameter are appended in the row direction to create a single snapshot matrix $\boldsymbol{W}_{total} = \left[\boldsymbol{W}_{p_1}, \boldsymbol{W}_{p_2}, \cdots, \boldsymbol{W}_{p_{N_p}}\right]$. Then, SVD is performed on $\boldsymbol{W}_{total}$ to obtain the spatially dependent POD modes, ϕ_i. The corresponding POD coefficients, a_i, in contrast, are temporally and parametrically dependent. Based on the aforementioned assumptions, the POD modes are

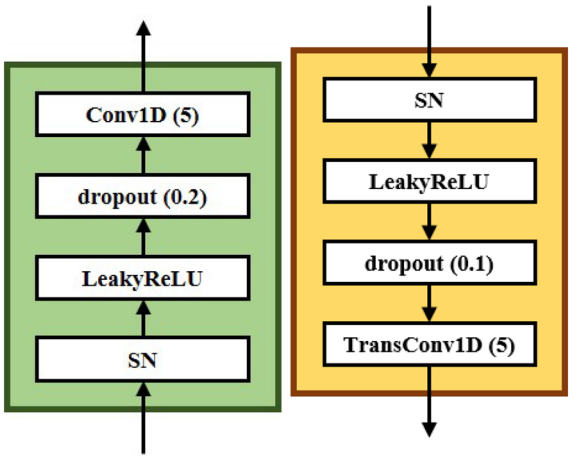
quasi-universal with respect to the parameter. The POD coefficients are then partitioned with regard to the parameter, as in Eq. 13:

$$\alpha_i = \left\{ \left(\alpha_1^1, \alpha_2^1, \cdots, \alpha_S^1\right), \cdots, \left(\alpha_1^{N_p}, \alpha_2^{N_p}, \cdots, \alpha_S^{N_p}\right) \right\} \tag{13}$$

The number of POD modes and coefficients was selected based on the energy ratio. By aligning the energy ratio of the POD modes to its size, the higher POD modes will exhibit a small value. These higher modes are neglected for MOR construction, and only a few lower modes are used.

4.2 Present mNVAE

The mNVAE encoder and decoder blocks are shown in Fig. 3. The current network comprises a bidirectional structure: bottom-up or top-down. The entire network of mNVAE comprises the blocks shown in Fig. 3. The encoder block contained a series of SN-LeakyReLU-dropout-Conv1D layers, where SN denotes the spectral normalization and Conv1D represents a 1D convolutional layer. The decoder block comprises an identical block, except for TransConv1D. TransConv1D is a transposed version that replaces Conv1D in the encoder for reconstruction. The numbers in parentheses next to Conv1D and TransConv1D layers are the kernel sizes. The dropout rates for the encoder and decoder were set as 0.2 and 0.1, respectively. This was designed to prevent overfitting and the SN layer was designed to stabilize the training process. Conv1D was employed to consider the temporal continuity of the POD coefficients.



(a) Encoder block **(b)** Decoder block

Fig. 3 Block construction in mNVAE

4.3 Training mNVAE and the flow field reconstruction

POD coefficients a_i are collected for $N_m \times N_p$ sets with a length of S time steps. The input dataset for mNVAE comprises 20 POD coefficients per network. The batch size for the training is denoted by N_b. The input dataset was reshaped into $(N_b, S, 20)$ and normalized to [0,1] for each POD coefficient. After the encoder, the shape of the latent code becomes (N_b, N_l), where N_l denotes the latent dimension size. The change in the data dimension in mNVAE is expressed in Eq. (14) as follows:

$$(N_b, S, 20) \xrightarrow{\text{encoder}} (N_b, N_l) \xrightarrow{\text{decoder}} (N_b, S, 20) \tag{14}$$

mNVAE is trained for POD coefficient interpolation, as shown in Algorithm 1. In Algorithm 1, a standard Adam optimizer is used and the ladder network architecture is adopted.

Algorithm 1 Training of mNVAE

$q_{\phi,i}^{enc}$: i_{th} encoder layer of mNVAE
$p_{\theta,i}^{dec}$: i_{th} decoder layer of mNVAE
Require: Hyperparameter, $\alpha_{LR}, \beta_1, \beta_2$
$x = N(a^m) = \bar{a}^m \rightarrow [0,1]\,(m = 1, \cdots, N_m)$
for $i = 1, N_{epoch}$ **do**
  for $j = 1, L$ **do**
    if $j = 1$ **then**
      $q_{\phi,1}^{enc}(x) = u_1 \rightarrow \mu_{q,1}, \sigma_{q,1}$
    else if $j > 1$ **then**
      $q_{\phi,j}^{enc}(u_{j-1}) = u_j \rightarrow \mu_{q,j}, \sigma_{q,j}$
    end if
  end for
  $z_L = \mu_{q,L} + \sigma_{q,L}\varepsilon, \varepsilon \sim \mathcal{N}(0,1)$
  for $k = L, -1, 1$ **do**
    if $k = L$ **then**
      $p_{\theta,k}^{dec}(z_L) = d_k \rightarrow \mu_{p,k}, \sigma_{p,k}$
    else if $1 \leq k < L$ **then**
      $p_{\theta,k}^{dec}(d_{k+1}, z_{k+1}) = d_k \rightarrow \mu_{p,k}, \sigma_{p,k}$
    end if
    $f(u_k, d_k) = \hat{u}_k \rightarrow \mu_{\hat{q},k}, \sigma_{\hat{q},k}$
    $z_k = \mu_{\hat{q},k} + \sigma_{\hat{q},k}\varepsilon, \varepsilon \sim \mathcal{N}(0,1)$
  end for
  $\Delta\mu_l = \mu_{\hat{q},l} - \mu_{p,l}, \quad \Delta\sigma_l = \sigma_{\hat{q},l}/\sigma_{p,l}\,(l = 1, 2, \cdots, L-1)$
  $\mathrm{Adam}(\nabla_{\phi,\theta}\mathcal{L}_{VAE}, \alpha_{LR}, \beta_1, \beta_2) \rightarrow \phi, \theta$
end for

The training process of the mNVAE can be divided into two parts: encoder and decoder training. First, the input dataset x is the normalized POD coefficient for each POD mode. In Algorithm 1, $N(a^m)$ denotes the normalization function of the POD coefficient. The normalized POD coefficient is provided to the first layer of L-layer encoder $q_{\phi,1}^{enc}$. The first encoder layer outputs u_1 and transforms it into $\mu_{q,1}, \sigma_{q,1}$. Then, for the remaining encoder layers, the output of the former layer, u_{j-1}, is provided as the input. Finally, the L_{th} encoder layer, the latent code z_L, is formulated by the mean and variance $\mu_{q,L}, \sigma_{q,L}$. In the formulation of the latent code, a Gaussian normally distributed random number, ε, is added. The decoder is

trained in a similar manner, where the input of the first decoder layer is substituted by z_L. A detailed explanation of the training of a VAE using a ladder network can be found in [42, 44].

Beyond the encoder, the latent variables are distributed over the latent dimension. Since Gaussian random noise was introduced, certain latent codes may not accurately represent the target POD coefficient. Therefore, an adequate latent code is sought for each parameter and coefficient prior to interpolation. Latent code search is performed by first providing the input dataset to the encoder. Latent code z_L is sampled from the encoder and then fed to the decoder. The discrepancy between the input and output of mNVAE is estimated. If the discrepancy is sufficiently small, z_L is stored as an adequate latent code. The latent code search process is summarized in Algorithm 2. Here, N_{iter} represents the number of iterations required to search for the latent code, which is set as 1,000.

Algorithm 2 Latent code searching

Require: Trained q_ϕ^{enc}, p_θ^{dec} from Algorithm 1
  for $i = 1, N_p$ **do**
    $q_\phi^{enc}\left(\bar{\alpha}^i\right) \rightarrow \mu_{q,L}, \sigma_{q,L}$
    $\epsilon = 1$
    for $j = 1, N_{iter}$ **do**
      $z_L = \mu_{q,L} + \sigma_{q,L}\varepsilon, \quad \varepsilon \sim \mathcal{N}(0,1)$
      if $\left(p_\theta^{dec}(z_L) - \bar{\alpha}^i\right)^2 < \epsilon$ **then**
        $\epsilon = p_\theta^{dec}(z_L) - \bar{\alpha}^i$
        $z^i = z_L$
      end if
    end for
  end for

In Algorithm 2, ϵ is the discrepancy between the input and the output. Interpolation will be performed to obtain adequate latent codes. Slerp is performed in the latent dimension, as it has been widely accepted for Gaussian-sampled generative models owing to its accuracy [50]. The interpolated latent code is then sent to the decoder to generate the interpolated POD coefficient. The relevant procedure is summarized in Algorithm 3, where *slerp* denotes slerp. The interpolated flow field is then constructed by the combination of $a_{i,p_{tgt}}$ and the pre-obtained POD modes, ϕ_i, as in Eq. (15):

$$q^*_{p_{tgt}}(x,t) = \bar{q}(x) + \sum_{i=1}^{N_m} a_{i,p_{tgt}}(t)\phi_i(x) \tag{15}$$

Algorithm 3 Interpolation of POD coefficients

Require: $N()$ from Algorithm 1, z^i from Algorithm 2
  for $n = 1, N_m$ **do**
    Set the target parametric value, p_{tgt}, and two adjacent values, p_1 and p_2
    Find the ratio, k, such that $p_{tgt} = k \times p_1 + (1-k) \times p_2$
    $z_{n,p_{tgt}} = slerp(z_{n,p_1}, z_{n,p_2}, k) = p_1 \frac{\sin(1-k)\theta}{\sin\theta} + p_2 \frac{\sin k\theta}{\sin\theta}, \theta = cos^{-1}(p_1 \cdot p_2)$
    $a_{n,p_{tgt}} = N^{-1}(f_\phi^{dec}(z_{n,p_{tgt}}))$
  end for

The present pMOR procedure comprises two stages: offline and online. The offline stage includes FOM construction, POD, mNVAE training, and latent code searching. The offline stage must only be performed once. In contrast, the online stage is executed repeatedly for each parametric estimation. It includes latent code interpolation and the construction of the interpolated flow field. Because the online stage requires a relatively smaller amount of computational time, the POD-mNVAE pMOR scheme becomes efficient. Figure 4 illustrates a flowchart of the proposed methodology.

5 Numerical results

This section presents the application of the proposed POD-mNVAE for two FSI situations. First, it was applied to the flow field of a plunging airfoil. The plunging airfoil demonstrated the variation in the flow field with respect to the deformed grid. The results obtained by POD-mNVAE were compared with those of other POD-based ANNs in terms of accuracy and efficiency. Then, the POD-mNVAE is examined by the highly nonlinear FSI phenomenon, LCO. The LCO examines the applicability of the POD-mNVAE to various engineering problems with nonlinearity.

For both applications, Navier–Stokes CFD analysis was performed. CFD analysis and POD were performed using an AMD 3950X CPU at 4.11 GHz. The ANN model was constructed using Tensorflow 2.7.0 and the training was

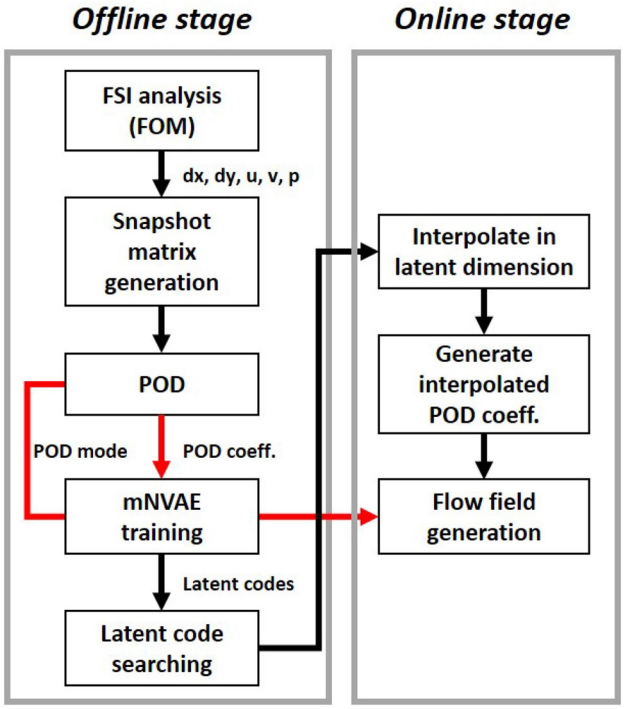


Fig. 4 Flowchart for POD-mNVAE

performed using an NVIDIA GeForce GTX 3090 GPU. The results obtained by the POD-mNVAE were evaluated by comparison with those obtained by FOM in terms of accuracy and computational time.

5.1 Plunging airfoil

5.1.1 Problem description

First, the prescribed plunging airfoil was analyzed. An airfoil with a chord length of 0.156 m was subjected to standard atmospheric inflow of 1 m/s. The airfoil plunges at a frequency of 10 rad/s while its amplitude varies. The plunging amplitude changes in ten variations, $h = [0.05, 0.06, \cdots, 0.13, 0.14]$ m, as illustrated in Fig. 5. The flow field was discretized using 28,315 three-node triangular elements with 14,300 nodes. An open-source Navier–Stokes CFD solver, OpenFOAM v1912, was used to construct the snapshot.

The snapshot matrix for various plunging amplitudes was obtained using the fully converged CFD results. 5s of FOM results with an interval of 0.01s were collected for the snapshot matrix. The total snapshot matrix W_{total} was then constructed by appending the snapshot matrices for each parameter in the row direction. After POD on W_{total} was completed, 100 POD modes were collected for the velocity, and one mode was collected for the grid deformation. The accumulated energy ratios of the POD modes are 99.9% and 99.9%, respectively. The first two POD mode shapes for the x-direction velocity u are shown in Fig. 6.

The current mNVAE for the plunging airfoil comprises six blocks in the encoder and decoder. The encoder comprises Conv1D blocks with [800, 400, 200, 100, 50] filters in a bottom-up network. The decoder comprises TransConv1D blocks with the same filters as in a top-down network. The latent dimensions for interpolation were set as 64. Detailed hyperparameters used for the present mNVAE training are summarized in Table 1.

After the training of mNVAE is completed, an adequate latent code for each parameter is sought. Then, *slerp* is performed, and the latent code for the target value, $h = 0.095$ m, is acquired. The interpolated POD

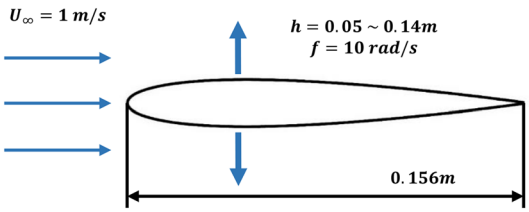
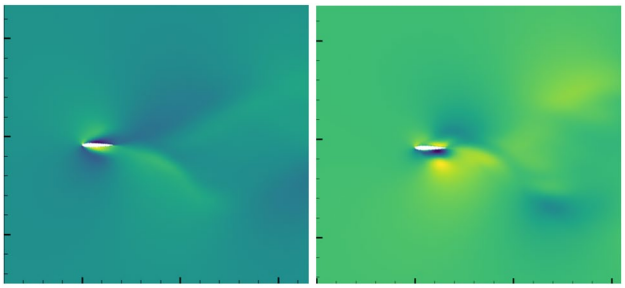


Fig. 5 Schematic of the plunging airfoil



(a) POD mode #1 **(b)** POD mode #2

Fig. 6 POD modes for the plunging airfoil, velocity component, u

Table 1 Hyperparameters for the present mNVAE training for the plunging airfoil

Criterion	Value	Criterion	Value
Epochs	20,000	Latent dim.	64
α	1,000	Batch size	1
β_{target}	5	Latent epochs	50
Learning rate	5×10^{-5}	Coeff. per network[a]	20

[a]Number of POD coefficients interpolated by a single mNVAE network

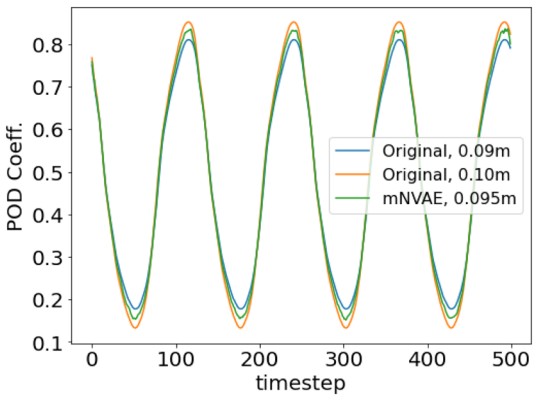
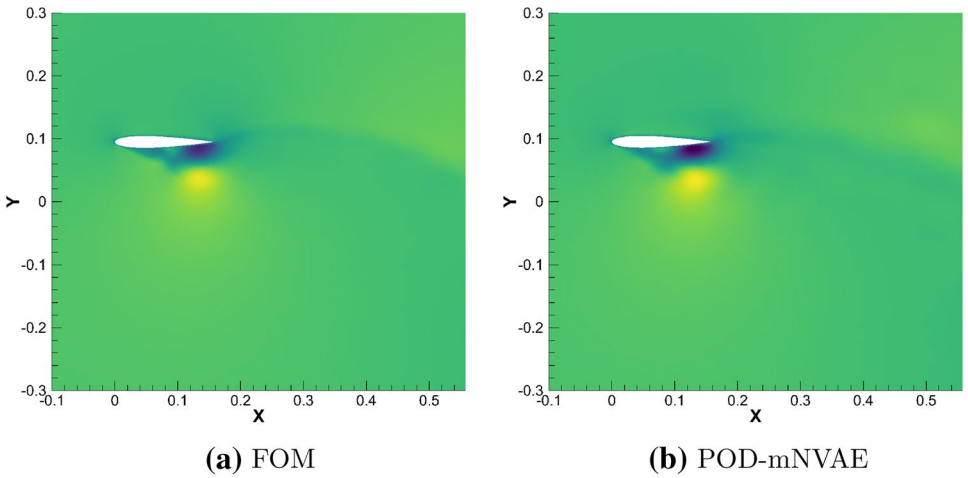


Fig. 7 Interpolated POD coefficient #1, velocity for the plunging airfoil

coefficients were generated by the decoder, as shown in Fig. 7. Using Eq. (15), an interpolated flow field is created for the target parameter. The resultant interpolated and FOM flow fields are shown in Fig. 8.

5.1.2 Accuracy and efficiency of POD-mNVAE

The accuracy of POD-mNAVE was evaluated with respect to the following seven categories:



(a) FOM **(b)** POD-mNVAE

Fig. 8 Original and interpolated flow field around the plunging airfoil for $h = 0.095$ m at $t = 4.5$ s

- ΔV_{avg}: Average velocity discrepancy.
- $\Delta f_{V,\text{avg}}$: Oscillation frequency discrepancy of the velocity component.
- ΔV_{pp}: Peak-to-peak (oscillation amplitude) discrepancy of the velocity component.
- ΔV_{point}: Velocity discrepancy at the 15 points of interest.
- $\Delta f_{x,\text{avg}}$: Oscillation frequency discrepancy of the grid deformation.
- Δx_{pp}: Peak-to-peak (oscillation amplitude) discrepancy of the grid deformation.
- Δx_{point}: Grid deformation discrepancy in the 15 points of interest.

Among the seven categories, the preceding four are discrepancies in the velocity components. The latter three are discrepancies in grid deformation. The 15 points of interest described in ΔV_{point} and Δx_{point} were placed in the wake region. These points were placed where the changes in the physical variables were expected to be the largest. The locations of the 15 points are shown in the Appendix A, Fig. 16. The formulations for these seven categories are included in Appendix A. Table 2 summarizes the discrepancies in the plunging airfoil.

In Table 2, the accuracy of the POD-mNVAE is determined to be significantly small, as most of them are less than 1%. The largest discrepancy was determined as 6.40%,

Table 2 Discrepancies between POD-mNVAE and FOM for the plunging airfoil

Category	ΔV_{avg}	$\Delta f_{V,\text{avg}}$	ΔV_{pp}	ΔV_{point}
Discrepancy	0.11%	< 0.01%	6.40%	0.03%
Category	–	$\Delta f_{x,avg}$	Δx_{pp}	Δx_{point}
Discrepancy	–	0%	0.90%	0.10%

Table 3 Computational time result for the plunging airfoil

	Procedure		Computational time [h]
Offline	FOM		75.6
	POD	velocity	0.22
		Mesh	0.21
	Algorithm 1	Velocity	3.98
		Mesh	2.38
	Algorithm 2	Velocity	0.01
		Mesh	0.01
	Total online stage		82.4
Online	Algorithm 3		<0.01
	Flow field construction/write		0.29
	Total offline stage		0.29
Total sum			82.7

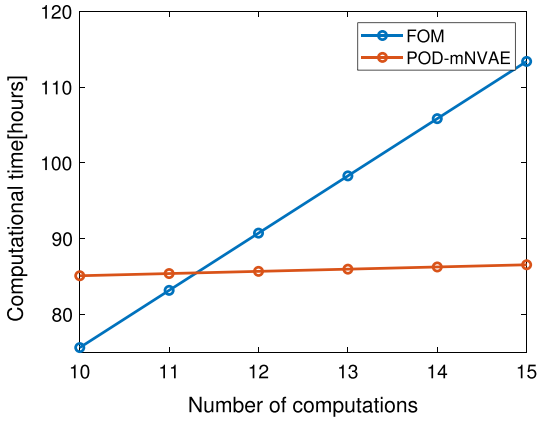


Fig. 9 Computational time in terms of the number of the computations for the plunging airfoil

which was ΔV_{pp}. A large discrepancy in ΔV_{pp} was caused by the tendency to underestimate the oscillation amplitude. However, for the other categories, the discrepancies were smaller. In particular, for $\Delta f_{x,avg}$, the discrepancy was determined as zero.

The computational procedure for POD-mNVAE comprises six steps. The time required for each computational step is summarized in Table 3. The entire interpolation process for POD-mNVAE consumes 82.7 h regarding the ten parameters. It can be divided into two components, 82.4 h for the offline stage and 0.29 h for the online stage. With the appropriate execution of the offline stage, the interpolation process is predicted to consume only 0.29 h. In conclusion, POD-mNVAE is capable of reducing 96.2% of the computational time for each novel parametric estimation. POD-mNVAE will be efficient if repeated computations exceeding 12 times are required. The expected computational time in terms of the number of computations is as shown in Fig. 9.

5.1.3 Comparison against the other ANN methods

The proposed POD-mNVAE was then compared with other POD-based ANN methods. A comparison was performed between WGAN-GP [21], the previous version of mNVAE [51], and Gaussian process regression (GPR). The accuracy of various interpolation methods are summarized in Table 4. In this study, POD-mNVAE was found to be superior. The discrepancy in the current POD-mNVAE is the smallest, except for ΔV_{avg}, ΔV_{PP}, and Δx_{PP}. For these categories, the WGAN-GP [21] and GPR performed better. However, the current POD-mNVAE was the most accurate overall. Improved accuracy for the current mNVAE was observed because of the hybrid weighted mean squared error-Kullback–Leibler divergence (MSE-KLD) loss function. The current loss function was empirically determined to significantly enhance accuracy

Table 4 Accuracy of interpolation models for the plunging airfoil

Factors	Current	mNVAE [51]	WGAN-GP [21]	GPR
ΔV_{avg}	0.11%	0.32%	0.09%	0.15%
$\Delta f_{V,avg}$	<0.01%	0.03%	0.26%	<0.01%
ΔV_{PP}	6.40%	N/A	3.68%	12.08%
ΔV_{point}	0.03%	3.83%	0.23%	0.07%
$\Delta f_{x,avg}$	**0%**	0.27%	0.14%	0%
Δx_{PP}	0.90%	N/A	1.37%	0.61%
Δx_{point}	0.10%	1.72%	0.42%	0.13%

when used for a continuous dataset. The computational time required was obtained as the sum of the times required for Algorithms 1, 2, and 3. Notably, the proposed method was significantly efficient compared to other ANN methods. The current mNVAE required 6.37 h whereas the previous version of mNVAE needed 17.82 h, and WGAN-GP used 77.33 h for training. However, it is noteworthy that GPR was the most efficient as it took less than 0.1 h. The current mNVAE reduced the training time by more than 74 % compared with the previous version of the mNVAE [51]. It is also capable of reducing the training time by more than 91 % compared with WGAN-GP [21]. The current mNVAE was the most efficient, as it trains 20 POD coefficients per network. In contrast, the previous version of mNVAE trained a single POD coefficient per network, and WGAN-GP trained 10 POD coefficients per network. Training more POD coefficients per network leads to a smaller number of networks required for interpolation. The current mNVAE trains more POD coefficients per network with accuracy because the modified loss function enables a further compactly constructed latent space. However, the training speed of the WGAN-GP was slow owing to the inherent instability. To ensure stable training, a gradient penalty was adopted for the generator (similar to the decoder for the GAN). For the same reason, the critic network (similar to the encoder) was trained for five iterations per epoch [21]. In general, the current mNVAE is the most efficient yet accurate among the previously introduced unsupervised learning methods.

5.2 Limit cycle oscillation (LCO)

5.2.1 Problem description

The LCO of an aircraft is a periodic, nondiverging oscillation that may lead to structural fatigue and failure. It is an FSI phenomenon caused by either or both nonlinearities in the fluid and structural dynamics. An accurate LCO analysis is typically performed using a high-fidelity nonlinear FSI analysis. Generally, for fluid analysis, either Euler or Navier–Stokes CFD, is implemented owing to the high

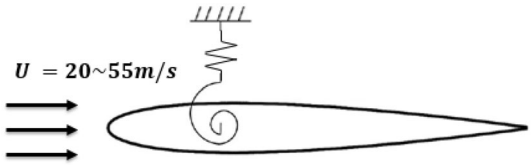


Fig. 10 Schematic of the airfoil under LCO

nonlinearity. Consequently, LCO analysis is challenging and tedious. Because the LCO amplitude and frequency differ by flight speed, iterative computation is required to determine the safe flight speed limit of an aircraft. In this section, the POD-mNVAE is examined for a realistic engineering problem with nonlinearity.

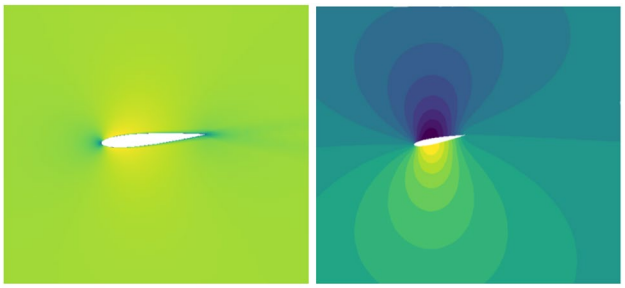
The analysis used in this section was derived from O'Neil et al. [52]. An airfoil with a chord length of 0.2128 m was subjected to standard atmospheric conditions as in Fig. 10. The inflow speed ranged from 20 to 45 m/s at 5 m/s intervals. The airfoil had two DOFs: pitch and heave. Both the pitch and heave stiffnesses are nonlinear in their cubic terms. The equations for the pitch and heave stiffnesses are expressed in Eq. 16 as follows:

$$\begin{aligned} K_\alpha &= 2.57(\alpha + 500\alpha^3) \\ K_h &= 0.09(h + 2860h^3)) \end{aligned} \tag{16}$$

The parameter to be interpolated for the current analysis was flight speed. The flight speed of the airfoil will change in six variations, $U = [20, 25, 30, 35, 40, 45]$m/s, as shown in Fig. 14. The relevant flow field was discretized using 19,543 quadrilateral elements comprising 19,381 nodes. For CFD, the Navier–Stokes solver ANSYS was employed. To model the structural nonlinearity, a user-defined function was used. The preliminary LCO analysis exhibited a good correlation with the wind tunnel test [52]. The LCO onset speed was determined as 16 m/s for ANSYS FSI, whereas it was "slightly higher than 15 m/s" for the wind tunnel test [52].

The snapshot matrix for various flight speeds was obtained from the fully converged CFD result. $2s$ FOM results with an interval of $0.01s$ were collected for the snapshot matrix. The total snapshot matrix W_{total} was then constructed by appending the snapshot matrices for each parameter in the row direction. After POD on W_{total} was completed, 40 POD modes were collected for the velocity, and two modes were collected for the grid deformation. The first POD modes for velocity and grid deformation are shown in Fig. 11. The accumulated energy ratio for both POD modes was determined as 99.9

The current-interpolating mNVAE comprises eight blocks in the encoder and decoder. The encoder comprises Conv1D



(a) POD mode #1, u **(b)** POD mode #1, dx

Fig. 11 POD modes for LCO analysis

Table 5 Hyperparameters for mNVAE training for LCO analysis

Criterion	Value	Criterion	Value
Epochs	5,000	Latent dim.	256
α	1,000	Batch size	1
β_{target}	1	Latent epochs	50
Learning rate	5×10^{-5}	Coeff. per network[a]	20

[a]Number of POD coefficients interpolated by a single mNVAE network

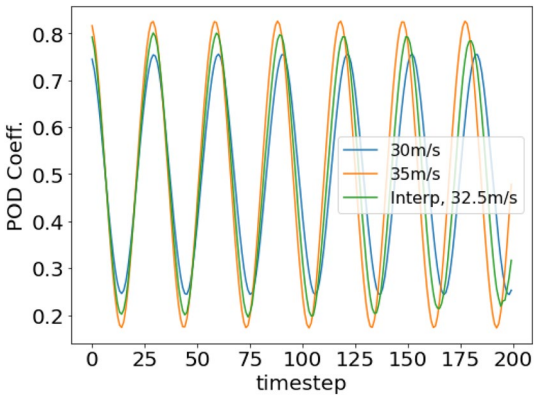
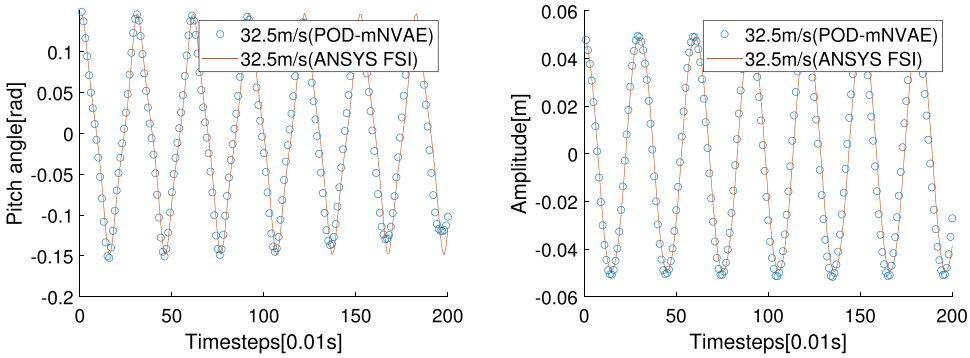


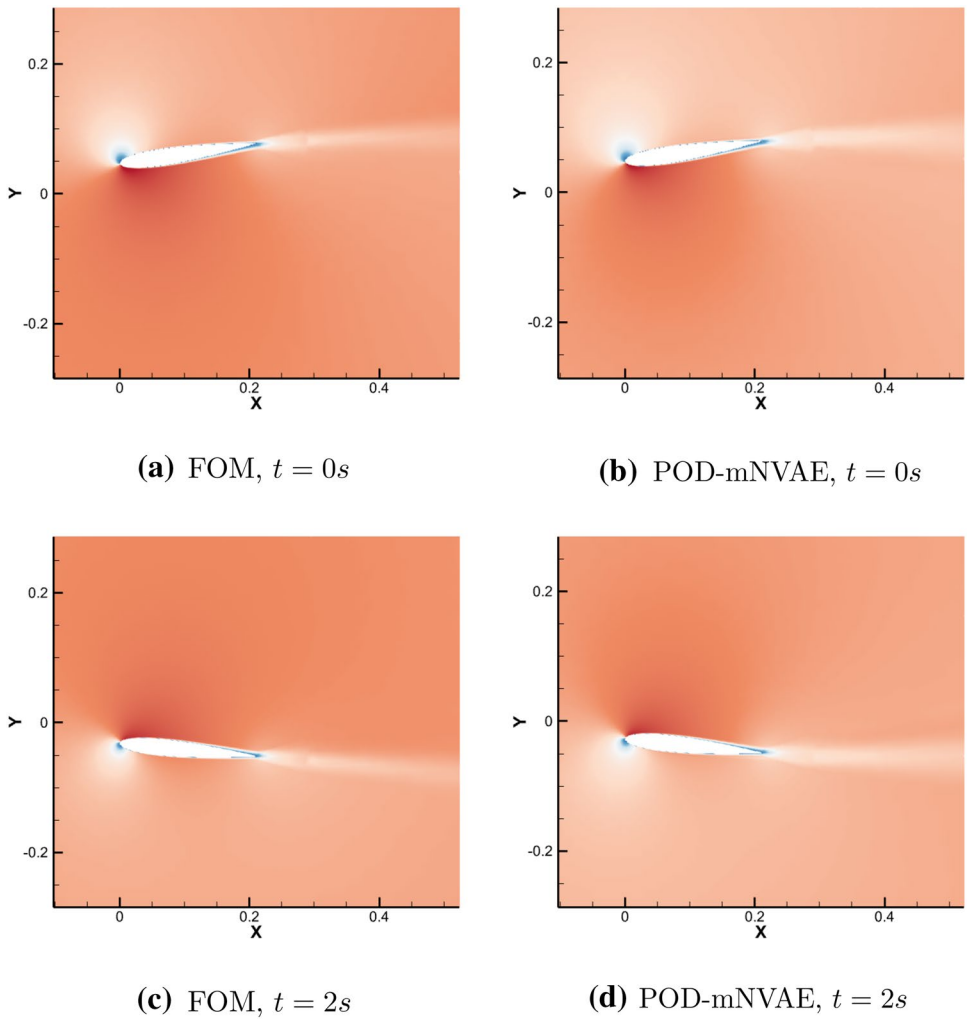
Fig. 12 Interpolated POD coefficient #1, dx, of LCO analysis

blocks with [800, 400, 200, 100, 50, 20, 10] filters in a bottom-up network. The decoder comprises TransConv1D blocks with the same filters as in a top-down network. The latent dimension for the interpolation was set as 256 to accommodate intricate pattern recognition. Detailed hyperparameters used for mNVAE training are summarized in Table 5. After the training of mNVAE was completed, an adequate latent code for each parameter was sought. Then, *slerp* was performed, and the latent code for the target value, $U = 32.5$ m/s, was acquired. The interpolated POD coefficients were



(a) Pitch angle **(b)** Heave amplitude

Fig. 13 Original and interpolated movement of the airfoil undergoing LCO



(a) FOM, $t = 0s$ **(b)** POD-mNVAE, $t = 0s$

(c) FOM, $t = 2s$ **(d)** POD-mNVAE, $t = 2s$

Fig. 14 Original and interpolated flow field around the airfoil undergoing LCO

generated by the decoder, as shown in Fig. 12. Using Eq. (15), the interpolated flow field was generated for $U = 32.5$ m/s. The resultant airfoil movement of the interpolated flow field and FOM is illustrated in Fig. 13. The fully interpolated flow field is illustrated in Fig. 14.

5.2.2 Accuracy and efficiency of POD-mNVAE

The accuracy of the POD-mNVAE was evaluated using the seven categories mentioned in the plunging airfoil. Fifteen points of interest were placed behind the airfoil, where the

Table 6 Discrepancies between POD-mNVAE and FOM for LCO analysis

Factors	ΔV_{avg}	$\Delta f_{V,\text{avg}}$	ΔV_{pp}	ΔV_{point}
Discrepancy	3.20%	0.73%	2.39%	3.21%
Factors	–	$\Delta f_{x,avg}$	Δx_{pp}	Δx_{point}
Discrepancy	–	0%	0.27%	0.32%

changes in the physical variables were expected to be the largest. The 15 locations are shown in Appendix A, Fig. 16. Table 6 summarizes the discrepancies in the POD-mNVAE for LCO analysis.

In Table 6, the accuracy of the POD-mNVAE was significantly small. The discrepancy in the velocity component was 3.21%. The other categories displayed similar discrepancies, except for frequency, which was as small as 0.73%. The discrepancies in grid deformation were smaller, ranging from 0% to 0.32%.

The computational procedure for POD-mNVAE comprises six steps. The time required for each computational step is summarized in Table 7. The entire interpolation process using POD-mNVAE consumes 284.4 h for the six parameters. It may be divided into two components, 283.8 h for the offline stage and 0.59 h for the online stage. With the appropriate execution of the offline stage, the interpolation process is predicted to consume only 0.59 h. In conclusion, the proposed POD-mNVAE is capable of reducing by 98.7% of the computational time for each novel parametric estimation. The proposed POD-mNVAE is efficient if the number of repeated computations exceeds seven. The expected computational time in terms of the number of computations is shown in Fig. 15.

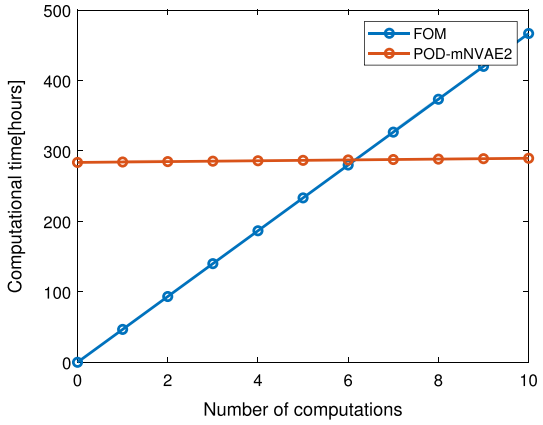


Fig. 15 Computational time in terms of the number of computations for LCO analysis

6 Conclusions

In this study, an improved data-driven pMOR scheme was proposed to construct an accurate ROM. The present methodology, referred to as POD-mNVAE, combines an interpolating neural network, mNVAE, and POD. POD is used to reduce the number of DOFs in the FOM result, whereas mNVAE is used to compress the temporal information inherent in the POD output, that is, the POD coefficient. The POD-mNVAE was capable of accurately constructing the ROM while significantly improving the computational time.

The POD-mNVAE was applied to two FSI situations: the flow field surrounding a prescribed plunging airfoil and LCO. The evaluation was performed with regard to accuracy and computational time. The present POD-mNVAE produced accurate results. The plunging airfoil exhibited a discrepancy of less than 1% and that of LCO was approximately 3%. The proposed POD-mNVAE achieves a reduction in computational time of 96% for the plunging airfoil

Table 7 Computational time result for LCO analysis

	Procedure		Computational time[hr]
Offline	FOM (including the baseline parameter)		280.1
	POD	Velocity	0.04
		Mesh	0.04
	Algorithm 1 (mNVAE training)	Velocity	1.79
		Mesh	1.78
	Algorithm 2 (latent code searching)	Velocity	0.01
		Mesh	0.01
	Total online stage		283.8
Online	Algorithm 3 (POD Coeff. interpolation)		<0.01
	Flow field construction/write on disk		0.59
	Total offline stage		0.59
Total sum			284.4

and 98% for LCO at the cost of the pre-executed offline stage. Furthermore, the current mNVAE was compared with the previous versions of mNVAE, WGAN-GP, and GPR to assess its superiority. The current mNVAE produced the most accurate results. The present methodology is applicable to other fields, such as structural dynamics. In particular, the present approach was used to construct the ROM of a highly nonlinear FSI.

However, it has shortcomings. The present method may not be used for problems in which different parameters will exhibit different dynamics. Rapid change in physical dynamics will lead to incapability of using POD modes universally across the parameters. To overcome such limitation, adopting the local or on-the-fly MOR methods may be considered as in [53, 54]. The present method generally cannot extrapolate beyond the prescribed parametric space. It is due to the universal use of POD modes that are constrained within the parametric space. Regarding the accuracy of the present method, sufficient number of sampling need to be conducted. Empirically, more than 5 sampling will be desirable. mNVAE requires more than 5 data points to recognize nonlinear pattern accurately. Also, those sampling should be conducted so that the change in physical dynamics may be captured in POD modes. Finally, the present method cannot extrapolate beyond FOM computation duration. Simple modification such as inserting LSTM network will be considered to enable such temporal extrapolation.

In the future, the current POD-mNVAE will be evaluated for an extremely large three-dimensional (3D) full FSI situation. It will be investigated for a full CFD–CSD combination for large structures, such as a full 3D passenger jet aircraft analysis in maneuver. For large three-dimensional FSI problems, the present framework will be used as it is. However, MOR stage is expected to be a challenge since POD will require extensive memory and computational time. To mitigate such limitation, other dimensionality reduction methods such as the local POD and autoencoder will be investigated.

Appendix A: Accuracy evaluation

The 15 points of interest used for the accuracy evaluation are located where the changes in the variables are expected to be the largest. Those points are shown in Fig. (16).

The seven categories used to evaluate the accuracy of the POD-mNVAE were formulated as follows:

1. Average velocity discrepancy, ΔV_{avg}:

$$\Delta V_{avg} = \frac{\bar{u}_{fom} - \bar{u}_{rom}}{\bar{u}_{fom}} \tag{A1}$$

where

$$\bar{u} = \frac{1}{N}\sum_{i=1}^{N} u_i \tag{A2}$$

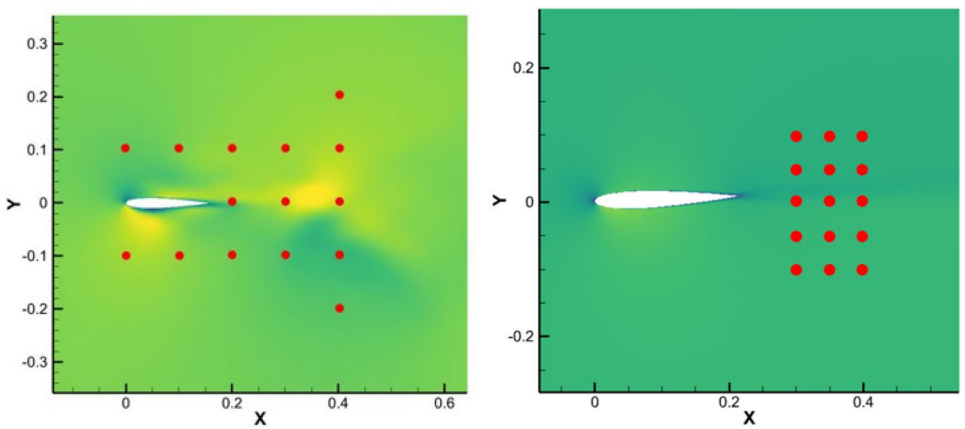
2. Oscillation frequency discrepancy of the velocity component, $\Delta f_{V,avg}$:

$$\Delta f_{V,avg} = \frac{\bar{f}_{fom} - \bar{f}_{rom}}{\bar{f}_{fom}} \tag{A3}$$

where f_{fom} is obtained for the 15 locations of interest.

$$\bar{f} = \frac{1}{n_p}\sum_{i=1}^{n_p} f_i \tag{A4}$$

3. Peak-to-peak (oscillation amplitude) velocity discrepancy, ΔV_{pp}:



(a) The plunging airfoil analysis **(b)** LCO analysis

Fig. 16 Fifteen points of interest for the accuracy evaluations

$$\Delta V_{pp} = \frac{u_{pp,fom} - u_{pp,rom}}{u_{pp,fom}} \tag{A5}$$

where $u_{pp,fom}$ is obtained for the 15 locations of interest.

$$u_{pp,fom} = \frac{1}{n_p}\sum_{i=1}^{n_p}(max(u_{,fom}) - min(u_{,fom})) \tag{A6}$$

4. Velocity discrepancy in the 15 points of interest, ΔV_{point}:

$$\Delta V_{point} = \frac{1}{n_p}\sum_{i=1}^{n_p}\frac{\|u_{rom,i} - u_{fom,i}\|}{\|u_{fom,i}\|} \tag{A7}$$

5. Oscillation frequency discrepancy of the grid deformation, $\Delta f_{x,avg}$:

$$\Delta f_{x,avg} = \frac{\bar{f}_{fom} - \bar{f}_{rom}}{\bar{f}_{fom}} \tag{A8}$$

where

$$\bar{f} = \frac{1}{n_p}\sum_{i=1}^{n_p} f_i \tag{A9}$$

6. Peak-to-peak (oscillation amplitude) discrepancy of the grid deformation, Δx_{pp}:

$$\Delta x_{pp} = \frac{x_{pp,fom} - x_{pp,rom}}{x_{pp,fom}} \tag{A10}$$

where $x_{pp,fom}$ is obtained for the 15 locations of interest.

$$x_{pp,fom} = \frac{1}{n_p}\sum_{i=1}^{n_p}(max(x_{,fom}) - min(x_{,fom})) \tag{A11}$$

7. Grid deformation discrepancy in the 15 points of interest, Δx_{point}:

$$\Delta x_{point} = \frac{1}{n_p}\sum_{i=1}^{n_p}\frac{\|x_{rom,i} - x_{fom,i}\|}{\|x_{fom,i}\|}. \tag{A12}$$

Acknowledgements This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science, ICT and Future Planning, Republic of Korea (2021R1A2C1007352). This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No.2021R1A5A1031868).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Lumley JL (1967) The structure of inhomogeneous turbulent flows. Atmos Turbul Radio Wave Propag
2. Sirovich L (1987) Turbulence and the dynamics of coherent structures, parts I, II and III. Quart Appl Math 45:561–590
3. Moore B (1981) Principal component analysis in linear systems: controllability, observability, and model reduction. IEEE Trans Autom Control 26(1):17–32
4. Lall S, Marsden JE, Glavaški S (1999) Empirical model reduction of controlled nonlinear systems. IFAC Proc Volumes 32(2):2598–2603
5. Berkooz G, Holmes P, Lumley JL (1993) The proper orthogonal decomposition in the analysis of turbulent flows. Annu Rev Fluid Mech 25(1):539–575
6. Rowley CW, Colonius T, Murray RM (2004) Model reduction for compressible flows using pod and galerkin projection. Physica D 189(1–2):115–129
7. Couplet M, Basdevant C, Sagaut P (2005) Calibrated reduced-order pod-Galerkin system for fluid flow modelling. J Comput Phys 207(1):192–220
8. Quarteroni A, Rozza G, Manzoni A (2011) Certified reduced basis approximation for parametrized partial differential equations and applications. J Math Ind 1(1):1–49
9. Chen H, et al (2012) Blackbox stencil interpolation method for model reduction. PhD thesis, Massachusetts Institute of Technology
10. Xiao D (2019) Error estimation of the parametric non-intrusive reduced order model using machine learning. Comput Methods Appl Mech Eng 355:513–534
11. Moosavi A, Ştefănescu R, Sandu A (2018) Multivariate predictions of local reduced-order-model errors and dimensions. Int J Numer Meth Eng 113(3):512–533
12. Krauth K, Bonilla EV, Cutajar K, Filippone M (2016) Autogp: Exploring the capabilities and limitations of gaussian process models. arXiv Preprint arXiv:1610.05392
13. Hesthaven JS, Ubbiali S (2018) Non-intrusive reduced order modeling of nonlinear problems using neural networks. J Comput Phys 363:55–78
14. Wang Q, Hesthaven JS, Ray D (2019) Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem. J Comput Phys 384:289–307
15. Li T, Deng S, Zhang K, Wei H, Wang R, Fan J, Xin J, Yao J (2021) A nonintrusive parametrized reduced-order model for periodic flows based on extended proper orthogonal decomposition. Int J Comput Methods 18(09):2150035
16. Kneifl J, Grunert D, Fehr J (2021) A nonintrusive nonlinear model reduction method for structural dynamical problems based on machine learning. Int J Numer Meth Eng 122(17):4774–4786
17. Hoang C, Chowdhary K, Lee K, Ray J (2022) Projection-based model reduction of dynamical systems using space-time subspace and machine learning. Comput Methods Appl Mech Eng 389:114341

18. Mohan AT, Gaitonde DV (2018) A deep learning based approach to reduced order modeling for turbulent flow control using lstm neural networks. arXiv preprint arXiv:1804.09269
19. Wiewel S, Becher M, Thuerey N (2019) Latent space physics: towards learning the temporal evolution of fluid flow. Computer graphics forum. Wiley Online Library, London, pp 71–82
20. Gonzalez FJ, Balajewicz M (2018) Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems. arXiv preprint arXiv:1808.01346
21. Lee S, Jang K, Cho H, Kim H, Shin S (2021) Parametric non-intrusive model order reduction for flow-fields using unsupervised machine learning. Comput Methods Appl Mech Eng 384:113999
22. Kadeethum T, O'Malley D, Fuhg JN, Choi Y, Lee J, Viswanathan HS, Bouklas N (2021) A framework for data-driven solution and parameter estimation of PDES using conditional generative adversarial networks. Nature Comput Sci 1(12):819–829
23. Kadeethum T, Ballarin F, Choi Y, O'Malley D, Yoon H, Bouklas N (2022) Non-intrusive reduced order modeling of natural convection in porous media using convolutional autoencoders: comparison with linear subspace techniques. Adv Water Resour 160:104098
24. Kadeethum T, Ballarin F, O'Malley D, Choi Y, Bouklas N, Yoon H (2022) Reduced order modeling with barlow twins self-supervised learning: Navigating the space between linear and nonlinear solution manifolds. arXiv preprint arXiv:2202.05460
25. Kim H, Cheon S, Jeong I, Cho H, Kim H (2022) Enhanced model reduction method via combined supervised and unsupervised learning for real-time solution of nonlinear structural dynamics. Nonlinear Dyn 110:2165–2195
26. Champion K, Lusch B, Kutz JN, Brunton SL (2019) Data-driven discovery of coordinates and governing equations. Proc Natl Acad Sci 116(45):22445–22451
27. Fresca S, Manzoni A (2022) Pod-dl-rom: enhancing deep learning-based reduced order models for nonlinear parametrized PDES by proper orthogonal decomposition. Comput Methods Appl Mech Eng 388:114181
28. Fries WD, He X, Choi Y (2022) Lasdi: parametric latent space dynamics identification. Comput Methods Appl Mech Eng 399:115436
29. He X, Choi Y, Fries WD, Belof J, Chen JS (2022) glasdi: parametric physics-informed greedy latent space dynamics identification. arXiv preprint arXiv:2204.12005
30. Raissi M, Perdikaris P, Karniadakis GE (2019) Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 378:686–707. https://doi.org/10.1016/j.jcp.2018.10.045
31. Chen W, Wang Q, Hesthaven JS, Zhang C (2021) Physics-informed machine learning for reduced-order modeling of nonlinear problems. J Comput Phys 446:110666
32. Kramer MA (1991) Nonlinear principal component analysis using autoassociative neural networks. AIChE J 37(2):233–243
33. Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114
34. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. Adv neural Inf Process Syst
35. Milano M, Koumoutsakos P (2002) Neural network modeling for near wall turbulent flow. J Comput Phys 182(1):1–26
36. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
37. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning, pp 1096–1103
38. Cho K (2013) Simple sparsification improves sparse denoising autoencoders in denoising highly corrupted images. In: International Conference on Machine Learning, PMLR, pp 432–440
39. Bergmann P, Löwe S, Fauser M, Sattlegger D, Steger C (2018) Improving unsupervised defect segmentation by applying structural similarity to autoencoders. arXiv preprint arXiv:1807.02011
40. An J, Cho S (2015) Variational autoencoder based anomaly detection using reconstruction probability. Spec Lect IE 2(1):1–18
41. Bowman SR, Vilnis L, Vinyals O, Dai AM, Jozefowicz R, Bengio S (2015) Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349
42. Sønderby CK, Raiko T, Maaløe L, Sønderby SK, Winther O (2016 Ladder variational autoencoders. Adv Neural Inf Process Syst
43. Fu H, Li C, Liu X, Gao J, Celikyilmaz A, Carin L (2019) Cyclical annealing schedule: A simple approach to mitigating kl vanishing. arXiv preprint arXiv:1903.10145
44. Vahdat A, Kautz J (2020) Nvae: a deep hierarchical variational autoencoder. Adv Neural Inf Process Syst 33:19667–19679
45. Phillips TR, Heaney CE, Smith PN, Pain CC (2021) An autoencoder-based reduced-order model for eigenvalue problems with application to neutron diffusion. Int J Numer Meth Eng 122(15):3780–3811
46. Xu J, Duraisamy K (2020) Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics. Comput Methods Appl Mech Eng 372:113379
47. Spinner T, Körner J, Görtler J, Deussen O (2018) Towards an interpretable latent space: an intuitive comparison of autoencoders with variational autoencoders. In: IEEE VIS
48. Cheng M, Fang F, Pain C, Navon I (2020) An advanced hybrid deep adversarial autoencoder for parameterized nonlinear fluid flow modelling. Comput Methods Appl Mech Eng 372:113375
49. Rasmus A, Berglund M, Honkala M, Valpola H, Raiko T (2015) Semi-supervised learning with ladder networks. Adv Neural Inf Process Syst
50. White T (2016) Sampling generative networks. arXiv preprint arXiv:1609.04468
51. Jang K (2022) Parametric interpolation of flow field based on the proper orthogonal decomposition and unsupervised machine learning.
52. O'Neil T, Strganac TW (1998) Aeroelastic response of a rigid wing supported by nonlinear springs. J Aircr 35(4):616–622
53. Choi Y, Boncoraglio G, Anderson S, Amsallem D, Farhat C (2020) Gradient-based constrained optimization using a database of linear reduced-order models. J Comput Phys 423:109787
54. Choi Y, Oxberry G, White D, Kirchdoerfer T (2019) Accelerating design optimization using reduced order models. arXiv preprint arXiv:1909.11320