Robustness evaluation of large-scale machine learning-based reduced order models for reproducing flow fields

Higashida, Aito

Ando, Kazuto

Rüttgers, Mario

Lintermann, Andreas

Tsubokura, Makoto

(Citation)
Future Generation Computer Systems,159:243-254

(Issue Date)
2024-10

(Resource Type)
journal article

(Version)
Version of Record

Future Generation Computer Systems 159 (2024) 243–254

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Robustness evaluation of large-scale machine learning-based reduced order models for reproducing flow fields

Aito Higashida [a,b], Kazuto Ando [a,b,c,*], Mario Rüttgers [d], Andreas Lintermann [d], Makoto Tsubokura [a,b]

[a] Department of Computational Science, Graduate School of System Informatics, Kobe University, 1-1, Rokkodai-cho, Nada-ku, Kobe, 657-8501, Hyogo, Japan
[b] Complex Phenomena Unified Simulation Research Team, RIKEN Center for Computational Science, 7-1-26, Minatojima-minami-machi, Chuo-ku, Kobe, 650-0047, Hyogo, Japan
[c] Operations and Computer Technologies Division, RIKEN Center for Computational Science, 7-1-26, Minatojima-minami-machi, Chuo-ku, Kobe, 650-0047, Hyogo, Japan
[d] Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Wilhelm-Johnen-Straße, 52428, Jülich, Germany

ARTICLE INFO

Keywords:
Computational fluid dynamics
Distributed machine learning
Convolutional autoencoder
Reduced order model
Model robustness evaluation

ABSTRACT

The robustness of an artificial neural network that performs model order reduction for flow field data is studied. The network is trained with a large-scale distributed learning approach using up to 6259 nodes of the supercomputer Fugaku. Flow around two square cylinders with a varying distance between their centers is investigated. The network is trained and tested with data from numerical simulations. First, the capability to reproduce flow fields with 2, 12, and 24 modes is investigated by comparing the reconstructed flow data to simulated data. It is shown, that reconstructions based on 2 modes cannot capture both, low- and high-frequency flow structures correctly, whereas predictions based on 12 and 24 modes yield improved flow fields, especially in the case of high-frequency waves in the vicinity of the square cylinders. Reconstructions with 24 modes provide smooth velocity fields that reproduce all relevant low- and high-frequency waves for all variations of the distance between the two square cylinders. Second, the performance of the machine learning-based reconstructions are compared to proper orthogonal decomposition, which is a commonly used reduced order model technique. The comparison only includes flow fields based on 24 modes. For all geometric variations, the mean squared errors of the reconstructions by the conventional method are higher than those of the machine learning model. This underlines the advantage of artificial neural networks over linear methods like proper orthogonal decomposition for tasks like reconstructing flow fields that are characterized by non-linear governing equations.

1. Introduction

Computational Fluid Dynamics (CFD) has been a significant application of high-performance computing for a long time and an indispensable tool for manufacturing. For example, multi-objective vehicle shape design requires many fluid simulations with different shapes to evaluate and optimize the aerodynamic properties of the vehicle body. However, executing many simulations that resolve the smallest turbulent scales according to Kolmogorov's $-5/3$ power law at a Strouhal number (St) of $St = fL/U_{ref} = 100$ [1] is nearly impossible considering a realistic computing budget, where f is the frequency, L is the vehicle length, and U_{ref} is the uniform inflow velocity.

To handle problems that require a large number of numerical flow simulations, various model order reduction techniques have been proposed in the CFD community. Proper orthogonal decomposition (POD) [2] is the most commonly used method and offers ways to find optimal lower-dimensional approximations for a given data set, e.g., data of flow fields. First, the hyperplane passing through the nearest space of most of the original data is constructed with the basis finding method. After identifying the basis, the original data is represented approximately using the data that is projected onto the hyperplane (hereinafter called "reduced variables"). The ratio of the number of bases that can reproduce the original data within an allowable error range to the number of dimensions of the original data corresponds to the data compression ratio.

* Corresponding author at: Department of Computational Science, Graduate School of System Informatics, Kobe University, 1-1, Rokkodai-cho, Nada-ku, Kobe, 657-8501, Hyogo, Japan.
E-mail address: kazuto.ando@riken.jp (K. Ando).

https://doi.org/10.1016/j.future.2024.05.005
Received 4 January 2024; Received in revised form 1 May 2024; Accepted 4 May 2024
Available online 10 May 2024

However, because POD is a linear reduction method, the nonlinear behavior of high Reynolds number (Re) flow fields cannot be successfully reduced into a few variables. To address this problem, recently, a nonlinear model reduction technique using neural networks has gained attention. Murata et al. proposed the mode decomposing convolutional neural network autoencoder (MD-CNN-AE) that reduces the data of the two-dimensional flow around a circular cylinder [3]. They successfully reduced the Kármán vortex street at $Re = 100$ into only two variables without significant reproduction loss.

Ando et al. extended the MD-CNN-AE to three-dimensional flow fields with a higher Reynolds number ($Re = 1000$) [4,5] using large-scale distributed machine-learning on the Supercomputer Fugaku [6,7]. A three-dimensional flow around a cylinder, which was calculated using 28 million cells, was reproduced using 64 variables, and the time series of the reduced variables were predicted using a long-short-term memory (LSTM) network [8]. Their implementation of distributed machine learning scales up to 25,250 computational nodes ($1,212,000$ cores) on Fugaku, and the convolution routine indicates over 100 PFLOPS as a single-precision floating-point arithmetic performance.

Hasegawa et al. evaluated the robustness of a neural network-based reduction model [9]. That is, the performance of the model for reproducing flow field characteristics was examined for flow around bluff bodies with varying shapes. The shapes were created by employing trigonometric functions with random amplitudes. The model was trained with data of two-dimensional flow around random variations of the bodies, and the flow around a variation that did not belong to the training data was successfully reproduced.

1.1. Related works

Neural network-based robustness analyses have not only been conducted for model order reduction techniques, but also for flow field reconstruction in general. Yu et al. investigated the model robustness for reproducing two-dimensional flow around a bridge body [10]. Shape variations were realized by changing the width and depth of the initial bridge body. They introduced the laws of mass and momentum conservation as loss functions into the training process. Their method demonstrated that the flow around an unknown bridge body that has not been part of the training data can be successfully reproduced. Moreover, Morimoto et al. studied the reproduction performance of the wake behind two parallel circular cylinders with varying distances and radii using a super-resolution technique [11]. Morimoto et al. also indicated the limitation of the robustness of a convolutional autoencoder [12]. They demonstrated that a model trained with flow fields around two square cylinders fails to predict the flow field around a single square cylinder, and vice-versa.

Data-driven techniques for analyzing highly resolved physical domains require large-scale distributed machine learning. Such machine learning approaches have recently been demonstrated on the top-ranked systems listed in the TOP500 ranking [13]. For example, Patton et al. utilized massively parallel deep learning to extract structural information from raw microscopy data with atomic resolution [14]. They used nearly all of the resources of the Oak Ridge National Laboratory's Summit system [15], and achieved practically perfect weak scaling up to 4200 nodes and showed a floating point arithmetic performance of 152 PFLOPS. Kurth et al. applied Exa-scale machine learning to image segmentation for climate data to specify extreme weather patterns. They achieved 90.7% of the weak scaling performance using 27,360 graphics processing units (GPUs) on Summit and won the Gordon Bell prize at the Supercomputing conference (SC) 2018 [16]. Yang et al. reported the performance of a physics-informed generative adversarial network (GAN) on Summit. Their calculations scaled up to 4584 nodes (27,500 GPUs) with 93.1% of the weak scaling performance and showed 1.2 EFLOPS with half-precision calculation [17]. Jia et al. were honored with the Gordon Bell prize at the SC 2020 for machine-learning-enhanced ab initio molecular dynamics simulations [18]. They achieved 91 PFLOPS in double-precision and 162/275 PFLOPS in mixed-single/half-precision using Summit.

1.2. Contributions

Neural network-based reduced order models can only develop their full potential if their predictive capabilities are robust to different flow configurations. Whereas the previously mentioned studies focused on variations of single flow parameters, the current study investigates the robustness of the model for varying execution conditions. Specifically, in contrast to Hasegawa et al. who evaluated the robustness for shape variations of a bluff body, which leads to a moderate difference in flow characteristics, this study investigates the robustness for various flow characteristics produced around two square cylinders separated by varying distances, utilizing large-scale distributed machine learning on Fugaku.

The paper is structured as follows. Section 2 describes the computational methods used to train the reduced order models. In Section 3, the results of the robustness analysis are presented, followed by concluding remarks in Section 4.

2. Methods

This section presents the computational methods that are used to conduct the robustness study. First, the implementation of the distributed machine learning approach on Fugaku is explained in Section 2.1. This contains general information about Fugaku, as well as details about the scalability of the implemented algorithm. Second, the numerical methods and the computational domain used to generate training data in terms of flow fields are described in Section 2.2. Finally, the architecture and hyperparameters of the machine learning model are explained in Section 2.3.

2.1. Distributed machine learning on Fugaku

The supercomputer Fugaku was developed by the RIKEN Center for Computational Science in Kobe, Japan.[1] At the time of writing, Fugaku is ranked number four among the HPC systems listed in the TOP500 [13]. Fugaku has a single arm-based central processing unit (CPU) named A64FX™, developed by Fujitsu.[2] The A64FX™ is equipped with four core memory groups (CMGs), equivalent to non-uniform memory access (NUMA) nodes, and each CMG has 12 computational cores. Each computational core runs at 2.0 GHz in normal mode and 2.2 GHz in boost mode. The peak arithmetic performance of the CPU when operating in normal mode is 3.072 TFLOPS for double-precision, 6.144 TFLOPS for single-precision, and 12.288 TFLOPS for half-precision. In boost mode, these increase to 3.3792 TFLOPS, 6.7584 TFLOPS, and 13.5168 TFLOPS, respectively. Double-precision matrix–matrix multiplication (DGEMM) operations achieve an efficiency of greater than 90%. Each CMG has 8 GiB of HBM2 memory and a 1024 GB/s node throughput. The Stream Triad performance is 830+ GB/s.

The Fugaku nodes are connected with TofuD interconnects [19]. The bandwidth is 6.8 GB/s per link, with six links. Consequently, the injection bandwidth is 40.8 GB/s per node. The system has 158,976 nodes and $7,630,848$ cores. The theoretical full system peak performance is 488 PFLOPS in normal mode and 537 PFLOPS in boost mode [7]. Intel's OneAPI Deep Neural Network Library (oneDNN) [20] has been optimized for the A64FX™ CPU. In this study, the PyTorch framework [21] is used that is ported to Fugaku [22].

The scalable distributed machine learning implementation that has been used by Ando et al. in previous studies is employed for the current robustness study [4,5]. This implementation incorporates a hybrid parallelization scheme combining data and model parallelism. The performance of distributed learning owes to the performance of data movement required for such parallelism. On the other hand,

[1] RIKEN Center for Computational Science, https://www.r-ccs.riken.jp/en/
[2] FUJITSU: https://www.fujitsu.com/global/

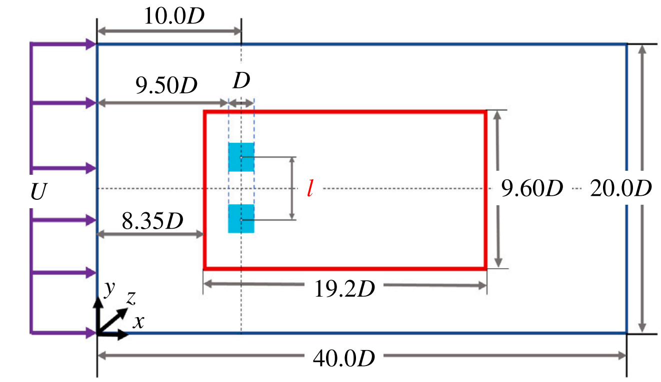

Fig. 1. Computational domain.

the serial execution performance is dependent on the performance of matrix–matrix multiplication in the convolution calculation because the method's neural network architecture is constituted with multiple blocks of convolutional and pooling (or interpolation) layers, as detailed in the following chapter. Using one CMG, the single-precision floating-point arithmetic performance of the entire training loop is 370.31 GFLOPS (24.28% of the peak performance). This corresponds to 1.5 TFLOPS for one node (4 CMGs). A convolution kernel indicates 753 TFLOPS (49.29% of the peak performance). This corresponds to 3.0 TFLOPS in terms of one node [23]. When the distributed machine learning implementation is scaled up to 25,250 nodes (1,212,000 cores), the single-precision floating-point arithmetic performance of the entire training procedure is 7.8 PFLOPS, which is 72.9% of the weak scaling performance relative to 750 nodes. In addition, the convolution routines reach approximately 100 PFLOPS at 25,250 nodes. The speedup of flow simulation execution performance using the reduced order model over the full order model (conventional CFD code) is four orders of magnitude in the case of three-dimensional cylinder flow (Re=1000) [4].

2.2. Numerical methods

The training data are obtained by simulating the flow field around two square cylinders using the simulation framework CUBE [24]. CUBE is a unified simulation framework that is based on the building cube method (BCM) [25,26] and immersed boundary method (IBM) [27,28]. The governing equations are the three-dimensional incompressible fluid continuity equation and Navier–Stokes equations

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}. \tag{2}$$

Here, $\boldsymbol{u}$, ρ, p, μ and f stand for the velocity vector, density, pressure, kinematic viscosity, and the vector for an external force, respectively. The constants are set to $\rho = 1$ kg/m^3 and $\mu = 0.01$ Pa s.

The computational domain is shown in Fig. 1. The uniform velocity $U = 1$ m/s is prescribed at the inflow boundary, a convective boundary condition was used at the outflow boundary, the free-slip condition is imposed on the top and bottom boundaries, and periodic boundaries were used on the front and back sides.

The computational grid is generated using CUBE. The characteristic length is the edge length D of a square cylinder and set to $D = 1$ m. The spatial discretization is performed with the finite volume method, which is uniform in both streamwise x, transverse y and height z directions with the grid size $\Delta x/D = \Delta y/D = 6.250 \times 10^{-4}$, $\Delta z/D = 1.875 \times 10^{-3}$. The number of computational cells is $(N_x, N_y, N_z) = (1600, 800, 2)$, and the domain size is $(40D, 20D, 0.05D)$. Two square cylinders are placed in the flow domain. The centers of the square cylinders are located at $10D$ in the streamwise direction from the inflow boundary. The distance between the prism centers is defined as l.

The time integration is done using the Adams–Bashforth scheme [29]. The time step is set to $\Delta t = 2.0 \times 10^{-3}$ s. The Reynolds number is defined by

$$Re = \frac{\rho U D}{\mu} = 100. \tag{3}$$

These conditions result in two asynchronous von-Karman vortices that interfere with each other at the center line behind the two square cylinders. By changing the value of l as the initial condition of the simulation, the interference pattern changes and the robustness of the proposed method can be tested with various patterns.

To focus on the 2D flow around the square cylinders, the velocities (u, v) in the region enclosed by the red rectangle in Fig. 1 are extracted to be used as training data for the machine learning algorithm. These vectors indicate the velocity in the streamwise direction and the transverse direction, respectively. The number of cells of this cutout is $(\Delta N_x, \Delta N_y, \Delta N_z) = (768, 384, 1)$, which corresponds to a cutout size of $(19.2D, 9.6D, 0.025D)$.

Data pre-processing such as normalization or standardization is not applied because the order of magnitude is unity for all the quantities. However, to compare the predicted flow fields with data generated by a POD approach, the input data for the machine learning model $\mathbf{x}^l_{input}(t)$ consists of the original instantaneous flow field $\mathbf{x}^l(t)$ minus the averaged flow field $\mathbf{x}^l_{average}$

$$\mathbf{x}^l_{input}(t) = \mathbf{x}^l(t) - \mathbf{x}^l_{average}, \tag{4}$$

This is because mode decomposition using POD solves the problem by reducing it to finding the eigenvectors and eigenvalues of the variance–covariance matrix. To obtain the variance–covariance matrix, it is necessary to subtract the mean value from the original data.

As shown in Fig. 1, the square cylinders are placed such that they are vertically symmetric. A total number of 50 simulations were conducted with an increment of $\Delta l = 0.0625D$, such that the distance between the centers of the prisms range from $1D$ to $4.0625D$. Snapshots of the flow fields, which serve as training data, were saved with a time interval of $\Delta t = 0.25$ s. For each simulation, 10,000 snapshots were captured for $2,500$ s.

2.3. Machine learning techniques

Fig. 2 shows a schematic of the network structure. The network has two levels. First, highly resolved snapshots of the flow fields function as input to the encoder of the network, where the dimensions are gradually decreased. A fully connected layer with $\mathcal{N}$ neurons yields the latent vector containing information of the reduced variables. Second, the decoder of the network splits up into $\mathcal{N}$ branches, one for each mode of the decomposition. The elements of the latent vector function as input to each branch. In each branch, the dimension is successively increased to output the decomposed flow field for the corresponding mode. Finally, these decomposed flow fields are combined to output a flow field that reproduces the original flow field.

The MD-CNN-AE is trained to minimize the error between the original flow field $\mathbf{x}(t)$ and reconstructed flow field $\mathbf{x}_R(t)$, with

$$\mathbf{x}_R(t) = \sum_{j=1}^{r} \mathcal{F}_{dec,j}\left(\left[\mathcal{F}_{enc}\left(\mathbf{x}(t)\right)\right]_j\right), \tag{5}$$

where $\mathcal{F}_{enc}$ is the encoder branch, $\mathcal{F}_{dec,j}$ stands for the decoder branch of the jth mode, and r represents the number of modes. This optimization problem is formulated as

$$\left\{\mathbf{w}_j, \mathbf{b}_j\right\}_{j=1}^r = argmin_{\{\tilde{\mathbf{w}}_j, \tilde{\mathbf{b}}_j\}_{j=1}^r} \int_{t_{min}}^{t_{max}} \left\|\mathbf{x}(t) - \mathbf{x}_R(t)\right\|^2 dt, \tag{6}$$

where $\left\{\mathbf{w}_j, \mathbf{b}_j\right\}_{j=1}^r$ are the weights and biases of the neural network.

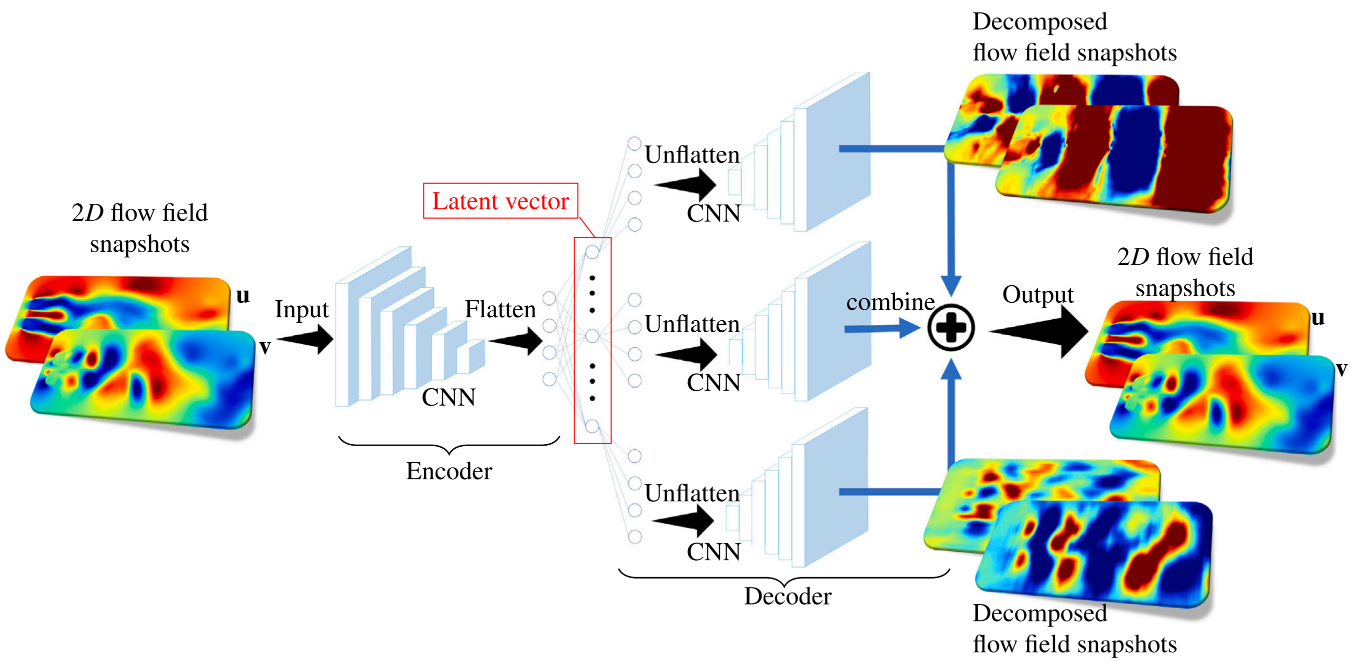

Fig. 2. Concept of the MD-CNN-AE.

Table 1
Network structure of the encoder of the MD-CNN-AE.

Encoder	
Layer	Data size
Input	(786, 384, 2)
1st Conv. (3, 3, 16)	(786, 384, 16)
1st MaxPooling	(384, 192, 16)
2nd Conv. (3, 3, 8)	(384, 192, 8)
2nd MaxPooling	(192, 96, 8)
3rd Conv. (3, 3, 8)	(192, 96, 8)
3rd MaxPooling	(96, 48, 8)
4th Conv. (3, 3, 8)	(96, 48, 8)
4th MaxPooling	(48, 24, 8)
5th Conv. (3, 3, 4)	(48, 24, 4)
5th MaxPooling	(24, 12, 4)
6th Conv. (3, 3, 4)	(24, 12, 4)
6th MaxPooling	(12, 6, 4)
Fully-connected (Latent vector)	($\mathcal{N}$, 1, 1)

Table 2
Network structure of a decoder branch of the MD-CNN-AE.

Decoder	
Layer	Data size
1st Value	(1, 1, 1)
Fully-connected	(12, 6, 4)
1st Upsampling	(24, 12, 4)
7th Conv. (3, 3, 4)	(24, 12, 4)
2nd Upsampling	(48, 24, 4)
8th Conv. (3, 3, 4)	(48, 24, 8)
3rd Upsampling	(96, 48, 8)
9th Conv. (3, 3, 4)	(96, 48, 8)
4th Upsampling	(192, 96, 8)
10th Conv. (3, 3, 4)	(192, 96, 8)
5th Upsampling	(384, 192, 8)
11th Conv. (3, 3, 4)	(796, 384, 16)
6th Upsampling	(796, 384, 2)
12th Conv. (3, 3, 2) (Decomposed field)	(786, 384, 2)

The types and size of layers of the encoder and the n decoder branches are listed in Tables 1 and 2. The encoder decomposes the input into modes by repeating six alternating 2D convolutional layers and max-pooling layers, and finally by inputting it into a fully connected layer representing the latent space. In contrast, a decoder reconstructs the flow field by first passing the encoder's input through a fully connected layer, then passing it through a convolution layer and an upsampling layer alternately six times.

The hyperprameters used to train the network are listed in Table 3. For encoder and decoder training, the Adaptive Moment Estimation Optimizer (ADAM) [30] is employed to find the optimized parameters. This algorithm calculates the moving average of the slope and the squared slope. Parameters $\beta 1$ and $\beta 2$ control the decay rate of these moving averages. In the training process, the 400,000 data snapshots selected in 3.1 are used as training data, and a total of 100,000 snapshots as test data.

Table 3
Hyperparameters used for the MD-CNN-AE.

Parameter	Value
CNN filter size	3×3
CNN pooling size	2×2
Number of layers	28
Number of data	500 000
Percentage of training data	80%
Time interval of data	0.25
Number of epochs	2000
Batch size	100
Optimizer for network	Adam
Learning rate of Adam	0.001
$\beta 1$ of Adam	0.9
$\beta 2$ of Adam	0.999
Learning rate decay of Adam	0

3. Results and discussion

In this section, the predictive capabilities of the MD-CNN-AE are analyzed. First, the ability of the neural network to reproduce flow fields depending on the number of decomposition modes is described in Section 3.1. Second, in Section 3.2, the neural network's decomposition accuracy is compared to the accuracy of a conventional method, i.e., flow fields reproduced by POD. These are described using the mean squared error (MSE). This is because, unlike metrics such as the

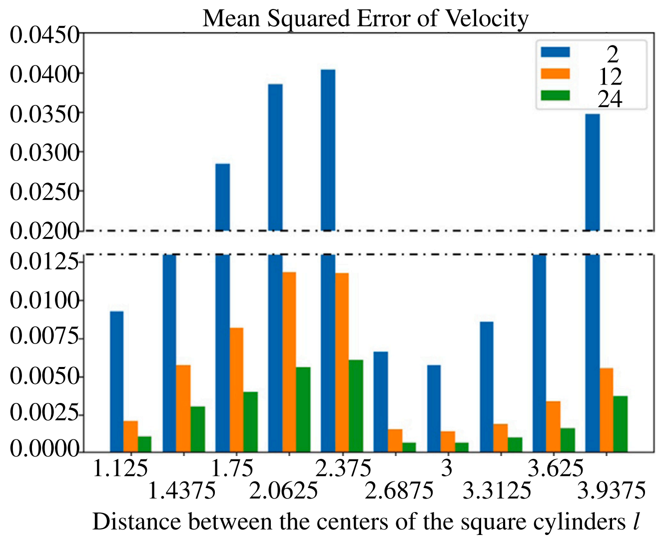

Fig. 3. Mean squared errors over all time steps between the reconstructed and simulated velocity fields for 2, 12, and 24 modes.

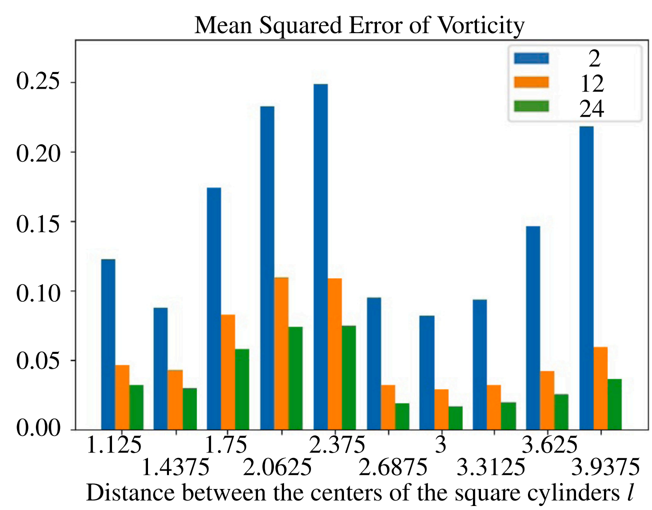

Fig. 4. Mean squared errors over all time steps between the reconstructed and simulated vorticity for 2, 12, and 24 modes.

structural similarity index (SSIM) [31], MSE does not have a weight parameter that can be freely set by the evaluator, so the results will not be biased or exaggerated.

3.1. Mode decomposition performance of the MD-CNN-AE

The dimensionality reduction performance of the MD-CNN-AE is investigated for several numbers of modes and a varying distance l. The distances l_{Train} between the centers of the square cylinders of the training data, and the distances l_{Test} between the centers of the square cylinders of the test data are computed as follows

$$l_{Train} = 1 + 0.0625n \qquad (n = 0, 1, \dots, 49), \tag{7}$$

$$l_{Test} = 1.125 + 0.3125n \qquad (n = 0, 1, \dots, 9). \tag{8}$$

Fig. 3 shows the MSE of velocities over all time steps for a varying distance l and different numbers of modes. Fig. 4 shows the mean squared error of vorticities over all time steps for a varying distance l and different numbers of modes. Generally, increasing the number of modes results in lower MSEs. Averaging the MSE for all variations of l

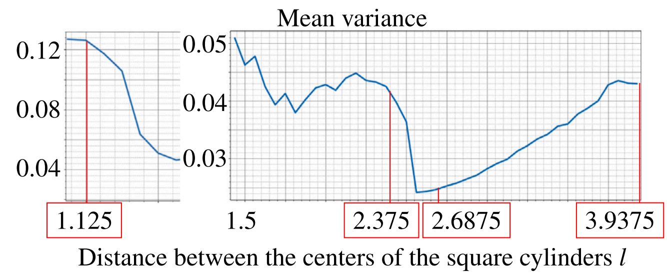

Fig. 5. Mean variance of the velocity fields of all test cases and training cases.

yields a reduction by 75% when the number of modes is increased from 2 to 12, and a reduction of another 50% when the number of modes is increased from 12 to 24. When varying l, the same general trend is observed for all numbers of modes. That is, the largest errors occur near $l = 2.375$. Relatively low MSEs are observed for $l = 1.125$, followed by a steady increase until $l = 2.375$, an abrupt decrease at $l = 2.6875$, and again an increase until $l = 3.9375$.

Fig. 5 shows the mean variance $\bar{\boldsymbol{U}}_{var}$ of the velocity fields, computed based on the variance $\boldsymbol{U}^l_{var}$ averaged for each simulation case according to

$$\boldsymbol{U}^l_{var} = \frac{\sum_{t\in N}\left(\boldsymbol{U}^l_t - \bar{\boldsymbol{U}}^l\right)^2}{N}, \tag{9}$$

where $\boldsymbol{U}^l_t$ indicates the velocity matrix $[u, v]$ of a snapshot at time t, $N = 10{,}000$ the number of snapshots per case, and $\bar{\boldsymbol{U}}^l$ the average of the velocity matrices of all snapshots of a case. In general, the smaller $\bar{\boldsymbol{U}}_{var}$ is, the more the low-frequency structure becomes dominant in the flow field, which improves the flow reconstruction accuracy by MD-CNN-AE. The case $l = 1.125$ has the largest variances, but relatively low MSEs for all modes. The reason for that is explained with the help of Figs. 6 and 7. The figures show the velocity components u and v of the simulation results of 10 test cases and the reconstruction results using the MD-CNN-AE with 2, 12, and 24 modes. The large variance comes from the fact that $l = 1.125$ is the only case with strongly alternating low-frequency vortices. The square cylinders are so close to each other that they nearly act as a single obstacle. Predicting such low-frequency vortices seems to be a relatively easy task for the MD-CNN-AE since the vortex shedding is visible even when solely using 2 modes.

The variance in Fig. 5 decreases from $l = 1.125$ to $l = 2.375$, but is generally still high, compared to the cases with $l > 2.375$. Figs. 6 and 7 indicate that although l increases, the separation region behind the square cylinders is still mainly characterized by low-frequency vortex shedding. However, with an increasing distance between the square cylinders, high-frequency structures start to develop in the flow field. These small scale vortices make predictions of the main flow structures more and more challenging and the MD-CNN-AE trained with 2 or 12 modes struggles to reproduce the flow field correctly. Only training with 24 modes yields an accurate reproduction. This is essentially visible for $l = 2.375$ where vortices with high-frequency shedding start to evolve behind each of the square cylinders.

Between $l = 2.375$ and $l = 2.6875$, Fig. 5 shows an abrupt change in the mean variance. This is due to the sudden disappearance of high-frequency structures from the flow field. Fig. 3 also shows the significant impact of the reconstruction of the velocity field due to the sudden change in the frequency structure of the flow field. The flow fields from $l = 2.6875$ to $l = 3.9375$ have separated vortex shedding behind each of the square cylinders in common. Especially the flow fields from $l = 2.6875$ to $l = 3.625$ are similar, which explains the generally low mean variance of these cases shown in Fig. 5. Although the MD-CNN-AE trained with 2 modes is still not capable of reproducing the main flow features correctly, predictions based on 12 and 24 modes manage to reconstruct the flow fields.

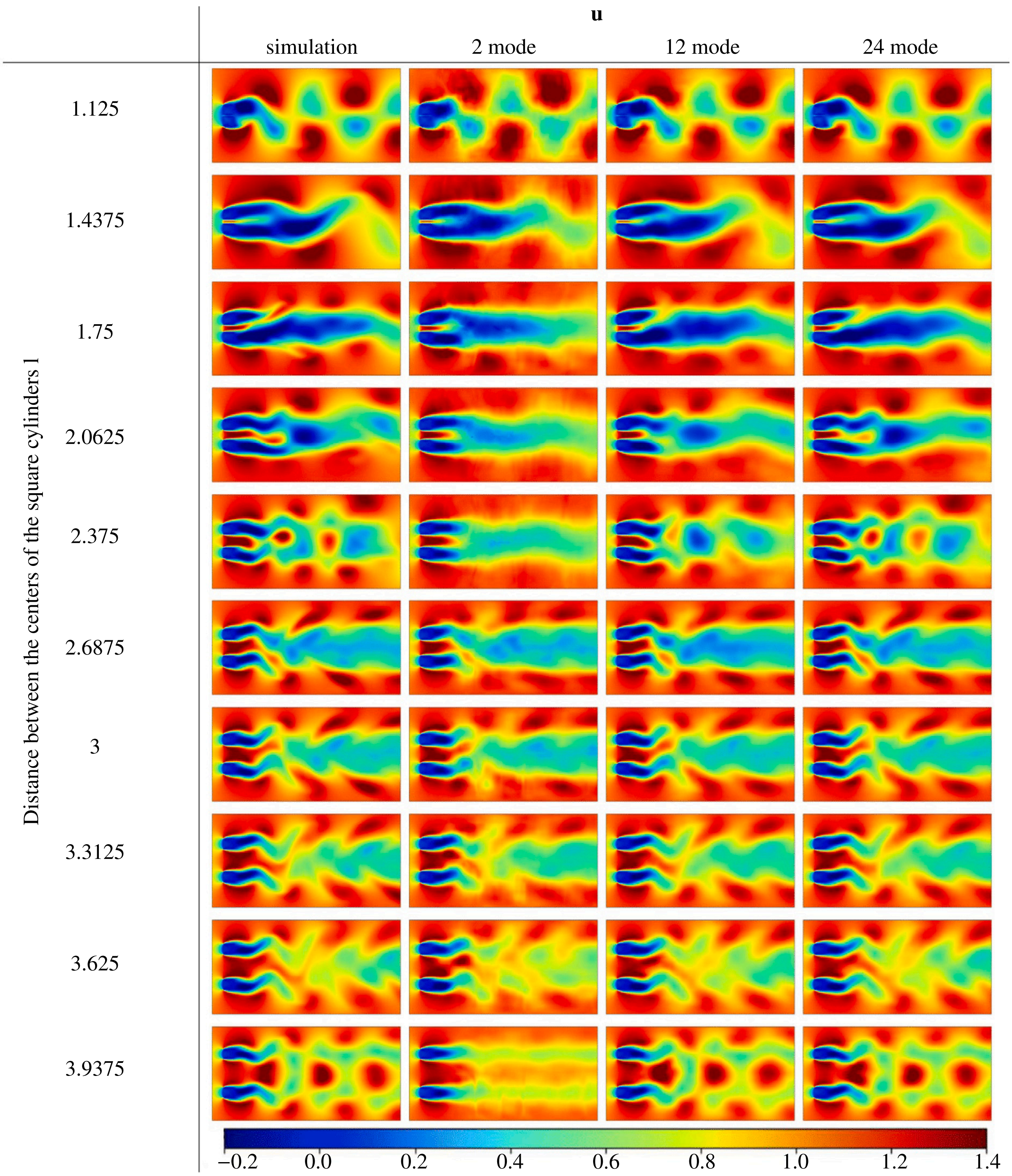

Fig. 6. The velocity component u of the instantaneous flow fields with various distances between the center of the square cylinders at $t = 1,350$ s. The simulated and reconstructed flow fields by the MD-CNN-AE are juxtaposed.

The tide turns for the case with $l = 3.9375$. Whereas the cases from $l = 2.6875$ to $l = 3.625$ are characterized by synchronous vortex shedding behind the two square cylinders, the flow for $l = 3.9375$ reveals asynchronous shedding. The MD-CNN-AE trained with two modes interprets this as a an averaged flow, as shown in Figs. 6 and 7. Yet, predictions based on 12 and 24 modes seem not to have these difficulties. However, the singularity of this case yields an increased variance in Fig. 5.

The previously presented results show that the cases $l = 1.125, l = 2.3750, l = 2.6875$, and $l = 3.9375$ play the major role for reconstructing the flow fields in the current robustness study. They are investigated in detail in Fig. 8, which shows the time-averaged streamwise velocity along the centerline between the square cylinders, simulated and reconstructed for 2, 12, and 24 modes and each of the four cases.

For all variations of l, the lowest errors occur when rebuilding the flow fields based on 24 modes. The MD-CNN-AE trained with 12 modes faces challenges in the central part of the centerlines for $l = 1.125$, $l = 2.6875$, and $l = 3.9375$. This indicates inaccurate predictions in the wake regions behind the square cylinders. Predictions based on 2 modes show less deviations in these regions compared to those based on 12 modes. This means that predictions with 2 modes are already close to reproductions of the time-averaged flow field. Reproducing the flow for

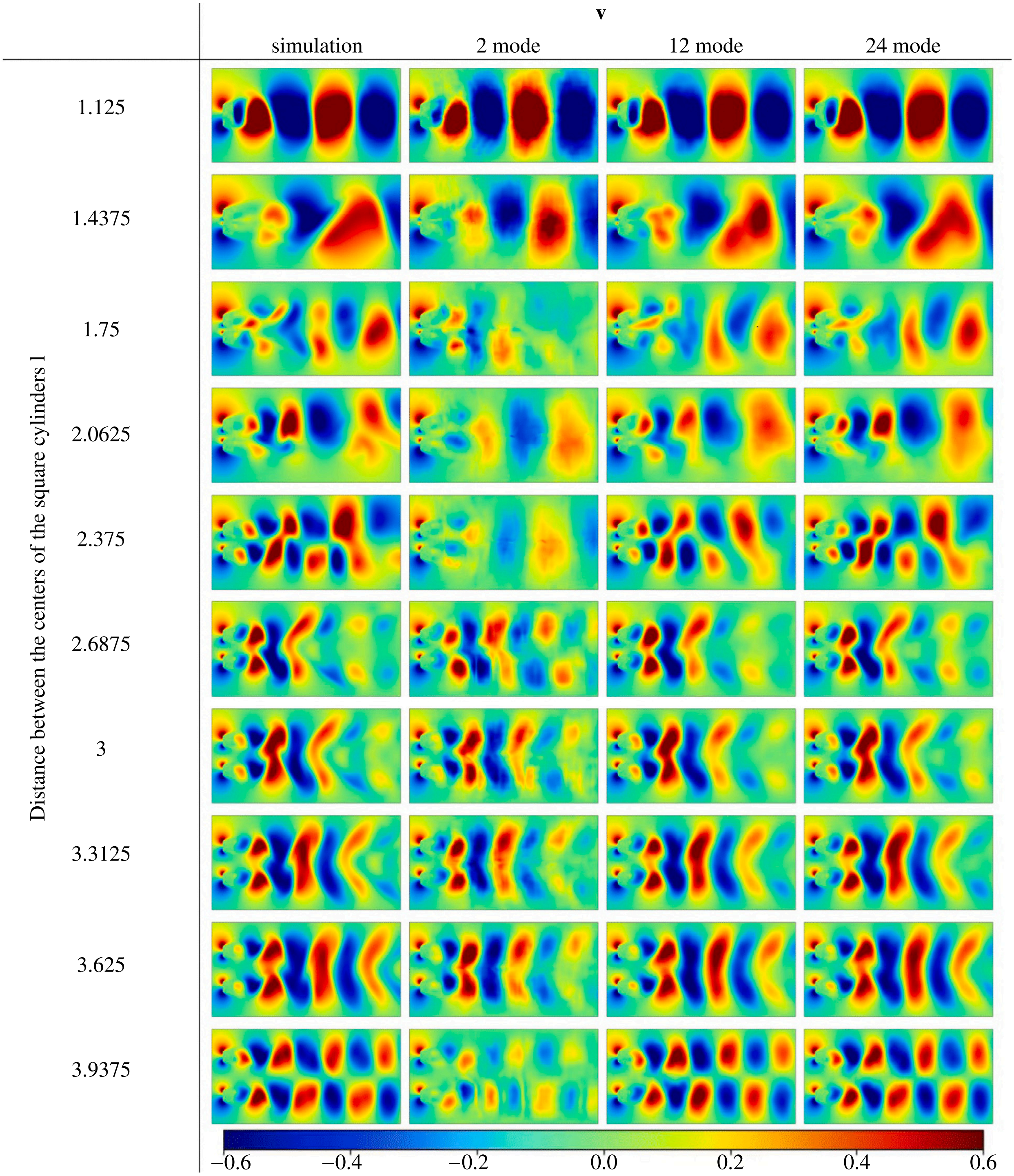

Fig. 7. The velocity component v of the instantaneous flow fields with various distances between the square cylinders at $t = 1,350$ s. The simulated and reconstructed flow fields by the MD-CNN-AE are juxtaposed.

$l = 3.9375$ shows inaccuracies with all numbers of modes. This again stresses the difficulty of predicting temporally high-frequency flow structures with two asynchronous von-Karman vortices that interfere with each other at the centerline.

3.2. Mode decomposition performance of the MD-CNN-AE compared to POD

In this section, the mode decomposition accuracy of the proposed method (MD-CNN-AE) is compared to a conventional method (POD). The comparison is carried out for 24 modes to reconstruct the flow field with sufficient accuracy. The test data is the same as in Section 3.1.

Fig. 9 shows the MSE for each method compared to the simulation results. In all cases, the MSE of the reconstructions based on POD is larger than the one based on the MD-CNN-AE. The mean MSE of the POD results is 3.61×10^{-3}, compared to 2.71×10^{-3} in case of the MD-CNN-AE, yielding a deviation of nearly 25%.

Similarly, Fig. 10 shows the MSE for each method compared to the vorticity results. In all cases, the MSE of the reconstructions based on POD is larger than the one based on the MD-CNN-AE. The mean MSE

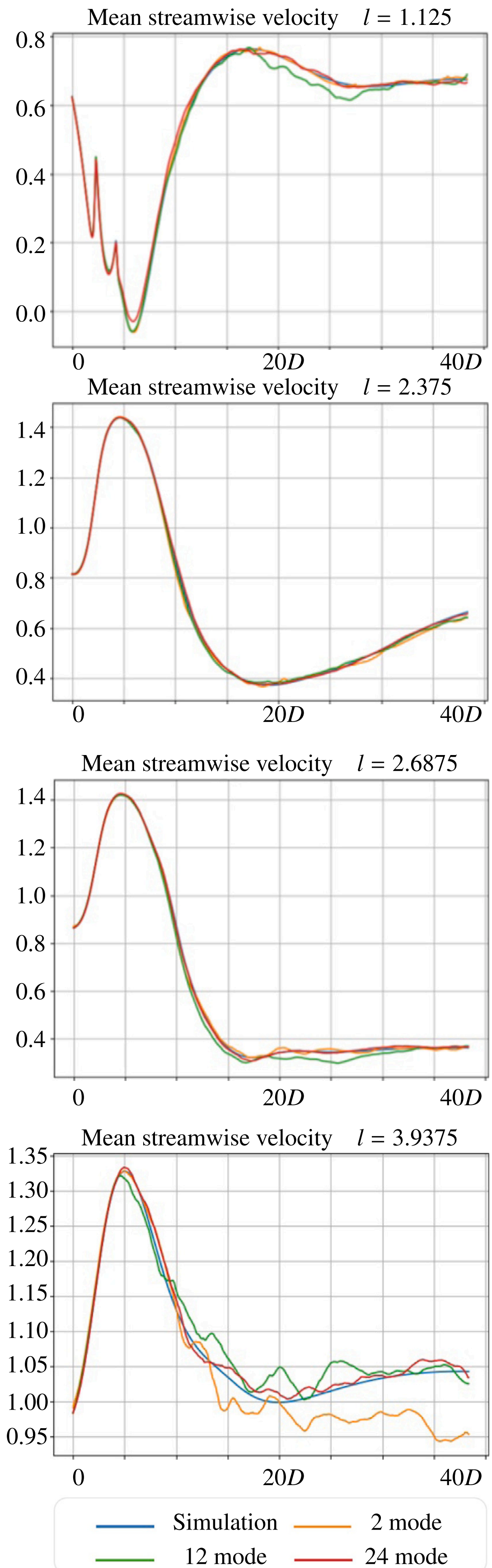

Fig. 8. Mean streamwise velocities along the centerline for l = 1.125, l = 2.3750, l = 2.6875, and l = 3.9375, simulated and reconstructed for 2, 12, and 24 modes.

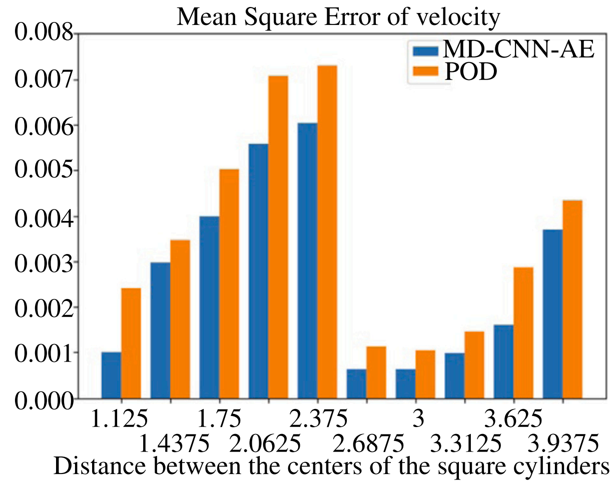

Fig. 9. Mean squared errors of reconstructed velocities by the MD-CNN-AE and POD compared to the reference simulation data.

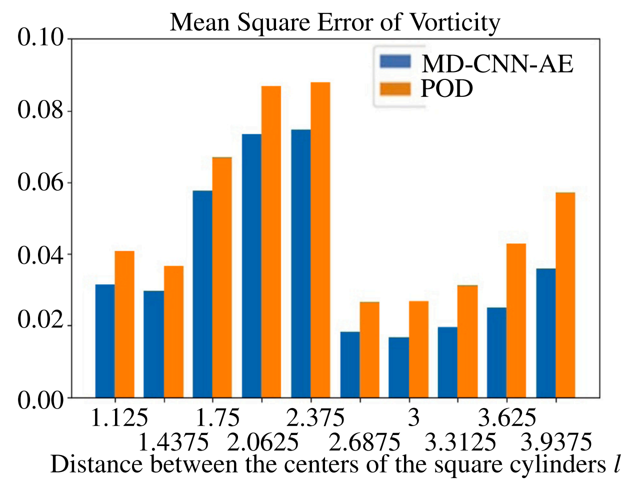

Fig. 10. Mean squared errors of reconstructed vorticities by the MD-CNN-AE and POD compared to the reference simulation data.

of the POD results is 5.04×10^{-2}, compared to 3.83×10^{-2} in case of the MD-CNN-AE, yielding again a deviation of nearly 25%.

Fig. 11 shows the deviation of the streamwise components of the time-averaged centerline velocity reconstructed by the MD-CNN-AE and POD to the simulation results. For l = 1.125, l = 2.375, and l = 2.6875, the reconstructed velocities of both reduced order methods show only minor differences to the simulation results. In contrast, the MD-CNN-AE has a clearly better accuracy than POD for l = 3.9375.

Figs. 12 and 13 show the time-averaged squared errors between the simulation results and the velocity fields reconstructed by the MD-CNN-AE and POD for l = 1.125, l = 2.3750, l = 2.6875, and l = 3.9375. For l = 1.125. The u-predictions based on the POD show large errors in the vicinity of the squared cylinders, and the v-predictions show inaccuracies in the complete wake region behind the obstacles. In contrast, the predictions by the MD-CNN-AE of both velocity components are characterized by low errors. This again indicates that the reconstruction of the low-frequency structures behind two closely located squared cylinders is a suitable task for the neural network.

For l = 2.3750, reconstructing the flow fields of both velocity components becomes more challenging for both reduced order methods. However, the reconstructions based on POD have generally higher errors than those of the MD-CNN-AE, especially in the two near-wall wake

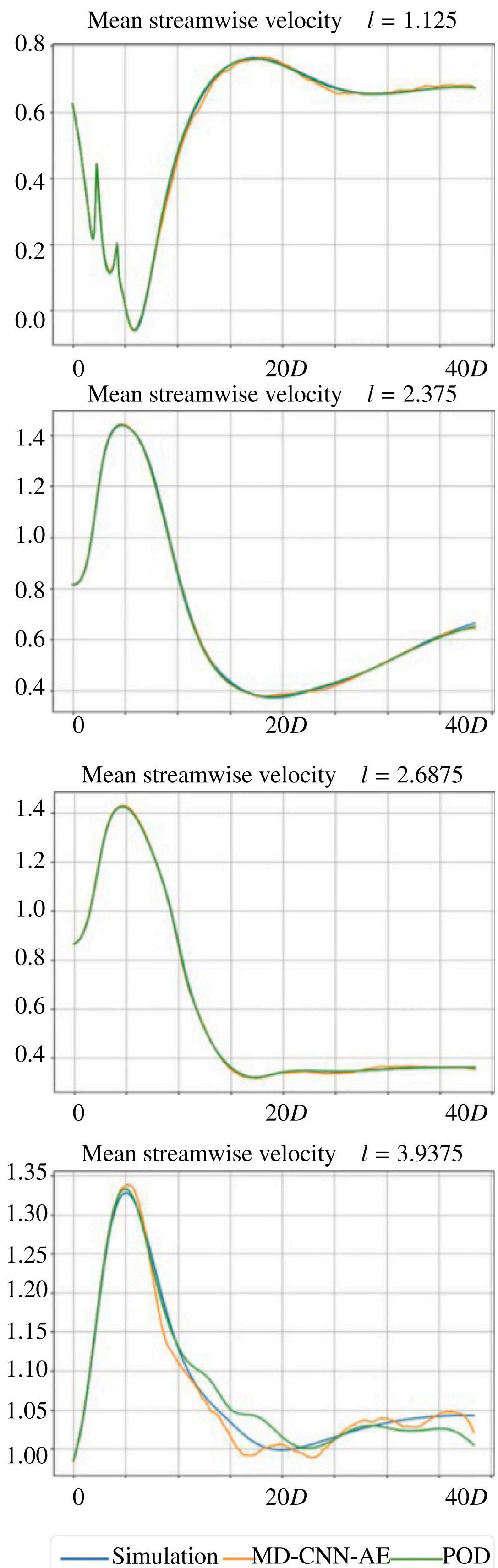

Fig. 11. Mean streamwise velocities along the centerline.

regions behind the obstacles which are dominated by high-frequency vortex shedding.

Predictions for $l = 2.6875$ were already attributed as the least challenging case with the lowest variance in Fig. 5 and the lowest MSEs in Figs. 3 and 9. Therefore, it is not surprising that both reduced order models have low errors when reconstructing both velocity components in Figs. 12 and 13.

For $l = 3.9375$, the only case with asynchronous vortex shedding, the errors for both velocity components tend to increase again. However, because the MD-CNN-AE is better at reconstructing high-frequency components, regions with large errors are less frequent compared to the errors illustrated for the POD.

Fig. 14 shows the time-averaged squared errors between the simulation results and the vorticity fields reconstructed by the MD-CNN-AE and POD for $l = 1.125$, $l = 2.3750$, $l = 2.6875$, and $l = 3.9375$. These results are consistent with the trend in the accuracy of the velocity field.

3.3. Computational costs of the MD-CNN-AE compared to POD

The implementation costs were compared between MD-CNN-AE and POD. It took 980 h using 11 nodes to run the 40-case simulation used for training data. Training MD-CNN-AE using these data took 32 h using 6250 nodes. On the other hand, flow field decomposition using POD took 3.5 h. There is a trade-off between accuracy and implementation cost.

4. Summary and conclusions

In this study, the robustness of a modal decomposition unsupervised neural network (MD-CNN-AE) applicable to large-scale machine learning-based reduced modeling of two-dimensional flow fields at $Re = 100$ is investigated. The flow domain is characterized by two square cylinders with a varying distance l.

First, neural network-based reconstructions based on 2, 12, and 24 modes have been investigated. For the predictions based on 2 modes, neither high-frequency waves, nor low-frequency flow structures are reproduced correctly. In addition, the velocity fields behind the square cylinders are not smooth and characterized by noise. In case of 12 modes, the low-frequency waves are reproduced much better than in the previous case, but still challenging from $l = 1.125$ to $l = 2.375$. From $l = 2.68875$ to $l = 3.9375$ the high-frequency recirculation zones in the vicinity of each square cylinders are captured well. However, parts of the high-frequency waves cannot fully be reproduced. To improve this, 24 modes were used for the reconstruction, which provided a smoother velocity field that reproduced all relevant low and high frequencies for all variations of l.

Second, neural network-based and POD-based reconstructions for 24 modes have been compared to flow fields computed by numerical simulations. For all variations of l, the MSE of the POD reconstructions is higher than the MSE of the flow fields reproduced by the MD-CNN-AE. Analyzing the L2 error of the flow fields for $l = 1.125$, $l = 2.375$, $l = 2.6875$, and $l = 3.9375$ yields a superiority of the MD-CNN-AE over POD, especially in the two near-wall wake regions behind the obstacles which are dominated by high-frequency vortex shedding.

These results suggest that the proposed method is applicable to general flow phenomena that have unknown flow characteristics (for example, flows that occur when the shape, location, and number of objects placed in the fluid are changed or when the Reynolds number is changed). The method proposed in this paper was developed through the Joint Laboratory for Extreme Scale Computing (JLESC) project "Deep Neural Networks for CFD Simulations"[3] through a collaboration between researchers from the RIKEN Center for Computational Science (R-CCS) and the Jülich Supercomputing Centre (JSC).

[3] https://jlesc.github.io/projects/dnn_cfd/

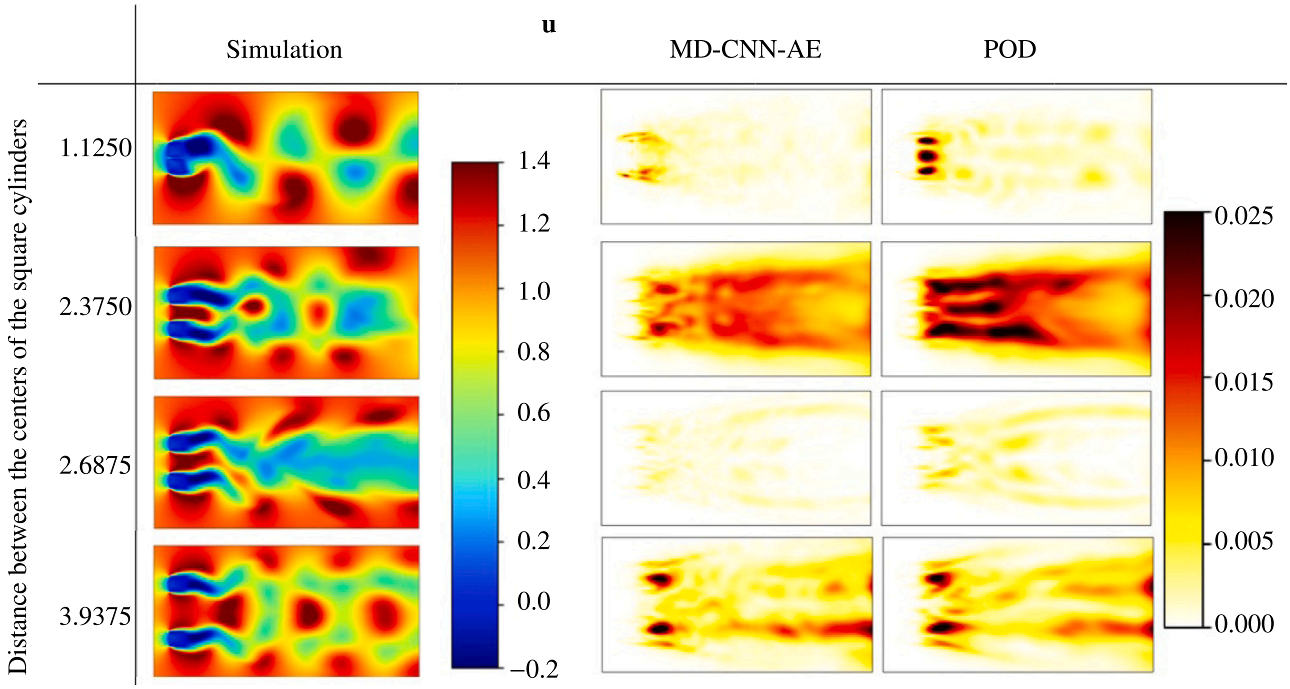

Fig. 12. Comparison of the time-averaged L2 error between reconstructed velocity by the MD-CNN-AE and POD, and simulation results. The flow fields are snapshots of **u** at t = 1,350 s.

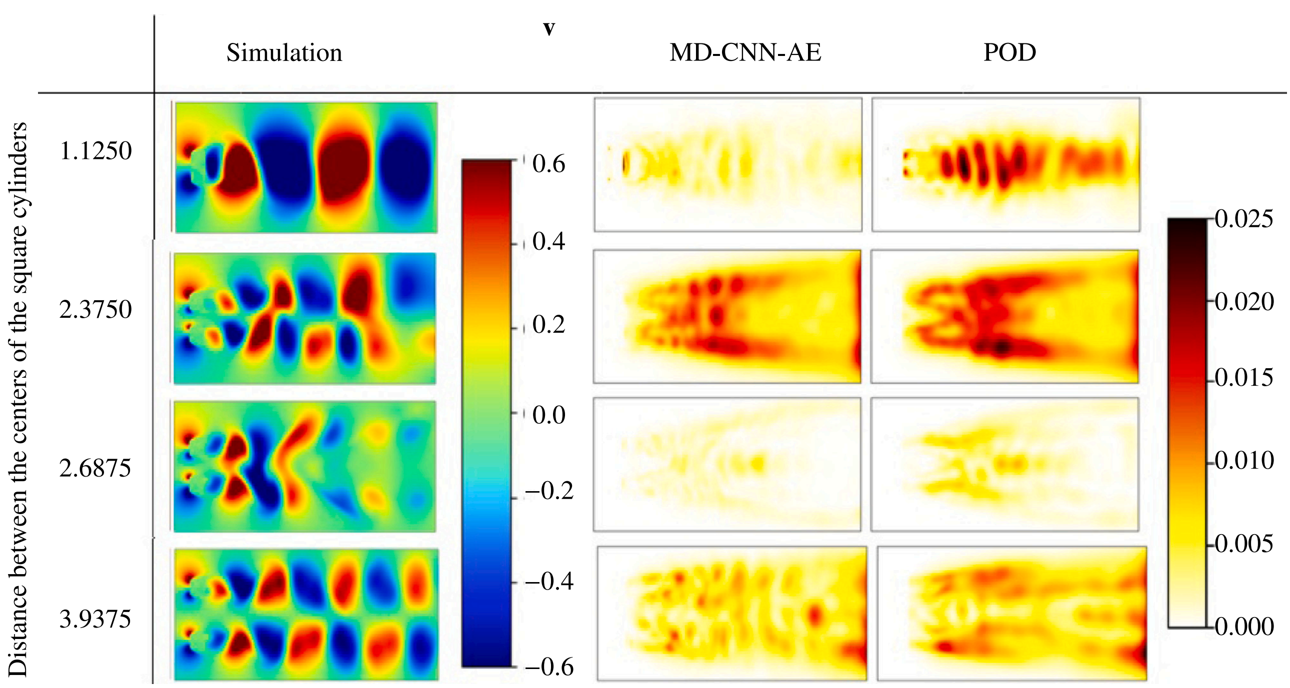

Fig. 13. Comparison of the time-averaged L2 error between reconstructed velocity fields by the MD-CNN-AE and POD, and simulation results. The flow fields are snapshots of **v** at t = 1,350 s.

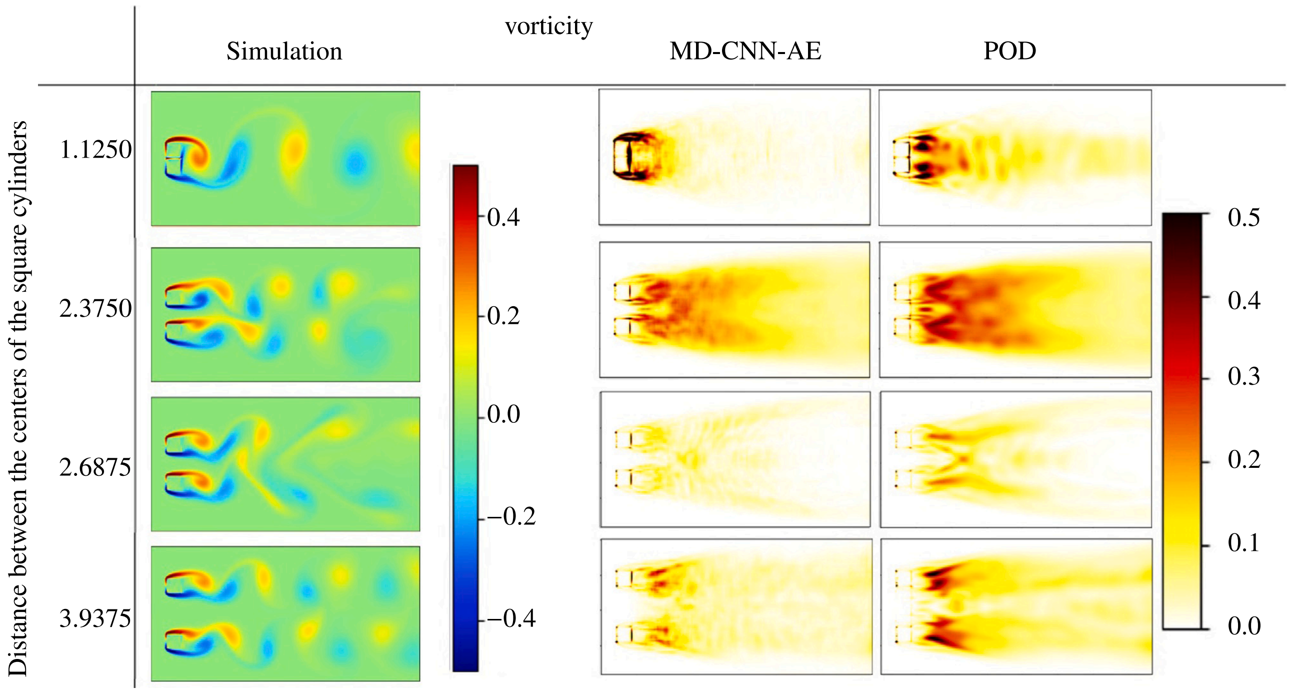

Fig. 14. Comparison of the time-averaged L2 error between reconstructed vorticity fields by the MD-CNN-AE and POD, and simulation results. The vorticity fields are snapshots at t = 1,350 s.

In future research, the nonlinear order reduction modeling method will be applied to more complex flow fields, such as flows around vehicle bodies. Future challenges include examining robustness in three-dimensional flow fields and at higher Reynolds numbers.

CRediT authorship contribution statement

Aito Higashida: Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Kazuto Ando:** Writing – original draft, Supervision, Software, Methodology. **Mario Rüttgers:** Writing – review & editing. **Andreas Lintermann:** Writing – review & editing. **Makoto Tsubokura:** Supervision, Resources, Project administration.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Mario Ruttgers reports financial support was provided by Horizon Europe. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

Computational resources of Fugaku were provided through the HPCI System Research Project (Project ID: hp230193). The research leading to these results has been conducted in the CoE RAISE project, which receives funding from the European Union's Horizon 2020 – Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733.

References

[1] M. Rüttgers, J. Park, D. You, Large-eddy simulation of turbulent flow over the DrivAer fastback vehicle model, J. Wind Eng. Ind. Aerodyn. 186 (2019) 123–138.
[2] Lumley, The structure of inhomogeneous turbulent flows, in: Atmospheric Turbulence and Radio Wave Propagation, 1967.
[3] T. Murata, K. Fukami, K. Fukagata, Nonlinear mode decomposition with convolutional neural networks for fluid dynamics, J. Fluid Mech. 882 (2020) A13.
[4] K. Ando, K. Onishi, R. Bale, M. Tsubokura, A. Kuroda, K. Minami, Nonlinear mode decomposition and Reduced-Order modeling for Three-Dimensional cylinder flow by distributed learning on Fugaku, in: High Performance Computing, Springer International Publishing, 2021, pp. 122–137.
[5] K. Ando, K. Onishi, R. Bale, A. Kuroda, M. Tsubokura, Nonlinear reduced-order modeling for three-dimensional turbulent flow by large-scale machine learning, Comput. & Fluids 266 (2023) 106047.
[6] T. Yoshida, Fujitsu high performance CPU for the Post-K computer, in: Hot Chips, Vol. 30, IEEE Computer Society, 2018, p. 22.
[7] R-CCS, About Fugaku, 2021, https://www.r-ccs.riken.jp/en/fugaku/about/. (Accessed 12 September 2023).
[8] S. Hochreiter, J. Schmidhuber, Long Short-Term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[9] K. Hasegawa, K. Fukami, T. Murata, K. Fukagata, Machine-learning-based reduced-order modeling for unsteady flows around bluff bodies of various shapes, Theor. Comput. Fluid Dyn. 34 (4) (2020) 367–383.
[10] X. Yu, T. Wu, Simulation of unsteady flow around bluff bodies using knowledge-enhanced convolutional neural network, J. Wind Eng. Ind. Aerodyn. 236 (2023) 105405.
[11] M. Morimoto, K. Fukami, K. Zhang, K. Fukagata, Generalization techniques of neural networks for fluid flow estimation, Neural Comput. Appl. (2022).
[12] M. Morimoto, K. Fukami, K. Fukagata, Experimental velocity data estimation for imperfect particle images using machine learning, Phys. Fluids (2021).
[13] TOP500.org, TOP500 supercomputer sites, 2023, https://www.top500.org/. (Accessed 14 September 2023).
[14] R.M. Patton, J.T. Johnston, S.R. Young, C.D. Schuman, D.D. March, T.E. Potok, D.C. Rose, S.-H. Lim, T.P. Karnowski, M.A. Ziatdinov, S.V. Kalinin, 167-PFlops deep learning for electron microscopy: from learning physics to atomic manipulation, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis, SC '18, (Article 50) IEEE Press, 2018, pp. 1–11.
[15] OLCF, Summit system overview, 2018, https://www.olcf.ornl.gov/wp-content/uploads/2018/05/Intro_Summit_System_Overview.pdf. (Accessed 07 November 2023).

[16] T. Kurth, S. Treichler, J. Romero, M. Mudigonda, N. Luehr, E. Phillips, A. Mahesh, M. Matheson, J. Deslippe, M. Fatica, Prabhat, M. Houston, Exascale deep learning for climate analytics, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis, SC '18, (Article 51) IEEE Press, 2018, pp. 1–12.
[17] L. Yang, S. Treichler, T. Kurth, K. Fischer, D. Barajas-Solano, J. Romero, V. Churavy, A. Tartakovsky, M. Houston, M. Prabhat, G. Karniadakis, Highly-scalable, Physics-Informed GANs for learning solutions of stochastic PDEs, in: 2019 IEEE/ACM Third Workshop on Deep Learning on Supercomputers, DLS, 2019, pp. 1–11.
[18] W. Jia, H. Wang, M. Chen, D. Lu, L. Lin, R. Car, W. E, L. Zhang, Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '20, (Article 5) IEEE Press, 2020, pp. 1–14.
[19] Y. Ajima, T. Kawashima, T. Okamoto, N. Shida, K. Hirai, T. Shimizu, S. Hiramoto, Y. Ikeda, T. Yoshikawa, K. Uchida, T. Inoue, The tofu interconnect D, in: 2018 IEEE International Conference on Cluster Computing, CLUSTER, 2018, pp. 646–654.
[20] Intel Corporation, oneAPI: A new era of heterogeneous computing, 2023, https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html. (Accessed 14 September 2023).
[21] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, 2017, https://openreview.net/forum?id=BJJsrmfCZ. (Accessed 14 September 2023).
[22] Fujitsu Limited, Deep neural network library for AArch64, 2019, https://github.com/fujitsu/dnnl_aarch64. (Accessed 14 September 2023).
[23] K. Ando, R. Bale, A. Kuroda, M. Tsubokura, Scalable reduced-order modeling for three-dimensional turbulent flow, in: Proceedings for SC23: International Conference for High Performance Computing, Networking, Storage and Analysis, to be published.
[24] N. Jansson, R. Bale, K. Onishi, M. Tsubokura, CUBE: a scalable framework for large-scale industrial simulations, Int. J. High Perform. Comput. Appl. 33 (4) (2019) 678–698.
[25] K. Nakahashi, Building-Cube method for flow problems with broadband characteristic length, in: Computational Fluid Dynamics 2002, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 77–81.
[26] K. Onishi, M. Tsubokura, Topology-free immersed boundary method for incompressible turbulence flows: an aerodynamic simulation for "dirty" CAD geometry, Comput. Methods Appl. Mech. Eng. 378 (2021) 113734.
[27] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[28] A.P.S. Bhalla, R. Bale, B.E. Griffith, N.A. Patankar, A unified mathematical framework and an adaptive numerical method for fluid-structure interaction with rigid, deforming, and elastic bodies, J. Comput. Phys. 250 (2013) 446–476.
[29] S.P.N. Ernst Hairer, Solving Ordinary Differential Equations I, Springer Berlin, Heidelberg, 1993.
[30] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, CoRR. arXiv:1412.6980.
[31] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612, http://dx.doi.org/10.1109/TIP.2003.819861.

Aito Higashida is a Master's Student in Kobe University, Japan. His research interests are Computational Fluid Dynamics, and Artificial Intelligence. He received his B.S. degree in engineering from Kobe University, Japan.

Kazuto Ando is a senior technical staff in the Operations and Computer Technologies Division and Complex Phenomena Unified Simulation Research Team at RIKEN Center for Computational Science (R-CCS), Kobe, Japan. His research interests are High-Performance Computing (specifically computational performance optimization on HPC systems), Computational Fluid Dynamics, and Artificial Intelligence. He had also worked in the application area of earthquake and tsunami simulation at JAMSTEC. He received his B.S. and M.S. degree in Physics from Tokyo Metropolitan University, Japan.

Mario Rüttgers is working as a doctoral researcher at the Simulation and Data Laboratory "Highly Scalable Fluid & Solids Engineering" (SDL FSE) of the Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich. He is associated with the Helmholtz School for Data Science in Life, Earth and Energy (HDS-LEE). His research focuses on combining computational fluid dynamics and machine learning techniques to improve diagnoses and treatments of respiratory diseases.

Andreas Lintermann is a postdoctoral researcher and group leader of the SDL FSE at JSC, Forschungszentrum Jülich. He is coordinating the European Center of Excellence in Exascale Computing "Research on AI- and Simulation-Based Engineering at Exascale" (CoE RAISE), leads the activities in the EuroCC/EuroCC2 and interTwin projects from Jülich's side, and is involved in the Industry Relations Team of the institute. His research focuses on high-performance computing, heterogeneous hardware, modular supercomputing, artificial intelligence, bio-fluidmechanical analyses of respiratory diseases, lattice-Boltzmann methods, high-scaling meshing methods, and efficient multi-physics coupling strategies.

Makoto Tsubokura has been the professor of computational fluid dynamics at the Graduate School of System Informatics, Kobe University since 2015, and the leader of Complex Unified Simulation Research Team at the Center of Computational Science, RIKEN since 2012. After earning a Ph.D. from the University of Tokyo in 1997, he first became a lecturer at the Tokyo Institute of Technology in 1999, then associate professor at the University of Electro-Communications in 2003, then at Hokkaido University in 2007.

He specializes in research and development of mathematical modeling and simulation algorithms for thermal fluids in turbulent state, and their expansion to massively parallel environment on high-end supercomputers. Until now, he has published more than 90 research articles, 30 review papers at distinguished international journals, and more than 50 plenary/invited talks, 110 conference papers and 110 presentations at international conferences. He is a fellow of the Japan Society of Mechanical Engineers, the Japan Society of Fluid Mechanics, and Society of Automotive Engineers of Japan. He has been a member of the Engineering Academy of Japan since 2020. Since 2023, he has been an associate member of Science Council of Japan.