Physics of Fluids

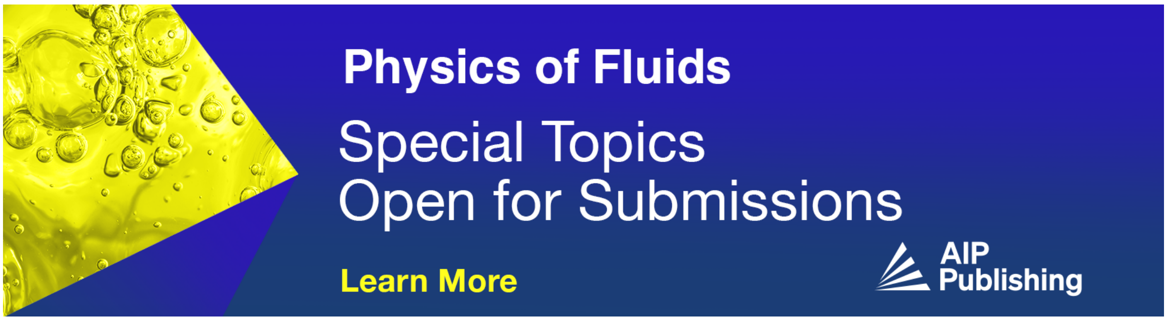
RESEARCH ARTICLE | JUNE 24 2024

Enhancing the accuracy of physics-informed neural networks for indoor airflow simulation with experimental data and Reynolds-averaged Navier–Stokes turbulence model

Special Collection: Flow and Civil Structures

Chi Zhang (张弛) ; Chih-Yung Wen (温志湧) ; Yuan Jia (贾媛) ; Yu-Hsuan Juan (阮于軒) ; Yee-Ting Lee (李宜庭) ; Zhengwei Chen (陈争卫) ; An-Shik Yang (楊安石) ; Zhengtong Li (李政桐)

Physics of Fluids 36, 065161 (2024)
https://doi.org/10.1063/5.0216394



AIP Publishing

Enhancing the accuracy of physics-informed neural networks for indoor airflow simulation with experimental data and Reynolds-averaged Navier–Stokes turbulence model

Cite as: Phys. Fluids **36**, 065161 (2024); doi: 10.1063/5.0216394
Submitted: 29 April 2024 · Accepted: 5 June 2024 ·
Published Online: 24 June 2024

View Online Export Citation CrossMark

Chi Zhang (张弛),[1,2] Chih-Yung Wen (温志湧),[3] Yuan Jia (贾媛),[3] Yu-Hsuan Juan (阮于軒),[4,5] Yee-Ting Lee (李宜庭),[6] Zhengwei Chen (陈争卫),[1,2] An-Shik Yang (楊安石),[7] and Zhengtong Li (李政桐)[3,a)]

AFFILIATIONS

[1]National Rail Transit Electrification and Automation Engineering Technology Research Center (Hong Kong Branch), Hong Kong, China
[2]Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong, China
[3]Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Hong Kong, China
[4]Department of Mechanical Engineering, National Taipei University of Technology, Taipei City 10608, Taiwan
[5]Research Center of Energy Conservation for New Generation of Residential, Commercial, and Industrial Sectors, National Taipei University of Technology, Taipei City 10608, Taiwan
[6]Department of Greenergy, National University of Tainan, Tainan City 70005, Taiwan
[7]Department of Energy and Refrigerating Air-Conditioning Engineering, National Taipei University of Technology, Taipei City 10608, Taiwan

Note: This paper is part of the special topic, Flow and Civil Structures.

[a)]**Author to whom correspondence should be addressed:** zhengtong.li@connect.polyu.hk

ABSTRACT

Physics-informed neural network (PINN) has aroused broad interest among fluid simulation researchers in recent years, representing a novel paradigm in this area where governing differential equations are encoded to provide a hybrid physics-based and data-driven deep learning framework. However, the lack of enough validations on more complex flow problems has restricted further development and application of PINN. Our research applies the PINN to simulate a two-dimensional indoor turbulent airflow case to address the issue. Although it is still quite challenging for the PINN to reach an ideal accuracy for the problem through a single purely physics-driven training, our research finds that the PINN prediction accuracy can be significantly improved by exploiting its ability to assimilate high-fidelity data during training, by which the prediction accuracy of PINN is enhanced by 53.2% for pressure, 34.6% for horizontal velocity, and 40.4% for vertical velocity, respectively. Meanwhile, the influence of data points number is also studied, which suggests a balance between prediction accuracy and data acquisition cost can be reached. Last but not least, applying Reynolds-averaged Navier–Stokes (RANS) equations and turbulence model has also been proved to improve prediction accuracy remarkably. After embedding the standard k–ε model to the PINN, the prediction accuracy was enhanced by 82.9% for pressure, 59.4% for horizontal velocity, and 70.5% for vertical velocity, respectively. These results suggest a promising step toward applications of PINN to more complex flow configurations.

I. INTRODUCTION

After years of development, traditional computational fluid dynamics (CFD) methods such as the finite volume method (FVM) now play a very important role in flow-related research, where they are competent for numerous indoor and outdoor flow field simulations. However, there are still a few limitations with these traditional methods and further development of new methods continues, where various machine learning methods with deep neural networks are expected to make a revolutionary contribution.

One constantly mentioned limitation of the traditional CFD methods is that they generally have a high computational cost and are considered time-consuming, unsuitable for some time-sensitive tasks like design space exploration, real-time prediction, and flow field control. Due to their strong fitting ability and fast inference feature, neural networks have been widely studied for such problems.[1–12] One common approach is to train the machine learning model to give flow field predictions directly from known information, e.g., geometries and inlet conditions, with a large amount of data, which can be thousands of times faster than computing with a traditional CFD method with a cost of accuracy.[1] Such methods can be helpful for aerodynamic configuration design, like fast prediction of aircraft aerodynamics[2] at an early stage. Other examples of potential applications include rapid flood inundation forecast,[3] real-time evaluation of CO_2 geological storage[4] and urban wind microclimate,[5–7] and fast prediction of indoor airflow distribution, which can be applied to real-time control of indoor flow field combined with air-conditioning systems.[8] As for the specific model architectures involved in these research types, apart from the simpler fully connected neural networks,[2,4] some of the more sophisticated neural network architectures include U-Net,[9] which was initially proposed for biomedical image segmentation,[13] graph neural networks (GNNs),[5,7] Fourier neural operators (FNOs),[3,4,6] and transformer networks.[10] Another data-driven approach to reduce computational costs is called super-resolution. One can first get low-fidelity data at a lower cost and then map them to high-fidelity with the trained model near real-time.[11,12] The model is trained with low-fidelity and high-fidelity data simultaneously to learn the mapping between them.

Despite the near real-time inference once trained, fully data-driven training for such surrogate models usually requires a large amount of data.[9] Data fidelity is also critical for the trained model's performance. These high-fidelity data can be acquired either from experiments or high-fidelity CFD results like direct numerical simulation (DNS). The high requirement for data on both quantity and quality can make the training very expensive, which can severely diminish its application value when the desired output space is small. For example, when using a transformer network trained with CFD data for flow field prediction,[10] the total time cost, including data preparation time and training time, is much larger than the calculation directly with CFD methods for a single case; the surrogate model only saves time when about over 1600 cases are inferenced. Thus, researchers proposed adding physical constraints for surrogate model training without labeled data,[14] which might be the key to addressing the data cost problem. Physics-informed neural network (PINN)[15] is one of the original works that has profound significance, where the residuals of governing partial differential equations (PDEs) can be taken as the physical loss for the training of fully connected neural networks. A purely physics-driven PINN can be similar to a CFD solver when solving forward problems. With a PINN, the flow field can be calculated when only the boundary and/or initial conditions are given together with governing equations without any extra data. Thus, it can sometimes also be referred to as a "neural solver."[16] A PINN with parameterized geometry inputs can accelerate a design optimization task by several orders of magnitude compared to traditional CFD solvers.[17] However, it must be clarified that, at current stage, training a single case with PINN normally does not provide advantage on computational speed compared to traditional CFD methods. Other examples of similar physics-informed methods include adding physical loss to a U-Net (named Spline-PINN),[18] physics-informed PointNet (PIPN),[19] where the PointNet is initially proposed for segmentation and classification of 3D point cloud data,[20] and physics-informed neural operator (PINO).[21]

Another problem with traditional CFD methods is that for incompressible flows, all the boundary conditions for a case must be explicitly determined. In contrast, PINN can be highly adaptive to some ill-posed boundary conditions.[22] For engineering applications that require determining the unknown inlet condition for CFD simulation, one might take several iterations for calibration based on measured data, which can be very time-consuming.[23] With its ability to utilize high-fidelity data,[24] a PINN can be trained with both governing PDEs and data points where they are available,[25–28] giving reasonable results even without knowing some of the boundary conditions explicitly.[27] This means that, for cases where the inlet condition can hardly be specified explicitly, by assimilating data collected from sensors, PINN holds the potential to simulate the flow field and infer the specific inlet condition with a single training, instead of assuming and adjusting the inlet condition while running CFD calculations repeatedly.

One more problem for traditional CFD methods like FVM is that, for researchers and engineers, the often handcrafted mesh generation involved in preprocessing can take enormous effort. However, the sophisticated mesh generation process can be completely bypassed for PINN due to its meshless nature. The handcrafted mesh generation is replaced by automatic sampling algorithms that sample residual points within the domain of interest, where automatic differentiation (AD) is applied to acquire the derivatives of physical variables for PDE residual calculation.[15] Intuitively, PINN's residual points are similar to the mesh grids in CFD, and different sampling schemes can significantly affect PINN's performance.[29] To summarize, the low dependency on training data, the ability to assimilate high-fidelity data during training, and its meshless feature make PINN a unique method worth to be studied.

However, despite all three advantages theoretically, the ability of PINN to simulate turbulent flow with high accuracy is still questionable,[30,31] yet traditional CFD methods with high accuracy can readily handle that. This can limit its application on built environment flow field study where turbulence is quite common. PINN has an inherent limitation for such multiscale problems, where the networks struggle to learn high-frequency components of flow field information and the steep gradients caused by these high-frequency features.[30] For CFD methods, the application of Reynolds-averaged Navier–Stokes (RANS) equations and various turbulence models[32] has been quite popular for the simulation of turbulence, especially for industrial applications since it achieves a good balance between accuracy and computational cost by omitting flow features in smaller scales, which can be inspiring for improving PINN to tackle turbulence. Despite PINN has already been applied for many turbulence-related research types,[22,25–28,33–36] only a few studies investigated it with the application of RANS equations. Eivazi *et al.*[35] and Hanrahan *et al.*[36] employed RANS equations with PINN to study flows in turbulent boundary layers and over a periodic hill. However, the Reynolds-stress components are required on the domain boundaries as training data, which are normally difficult to obtain, limiting the practical application of the method. Furthermore, Rui *et al.*[28] applied zero-equation models with PINN for flow field reconstruction. Their work demonstrated the feasibility of incorporating RANS equations and turbulence model into PINN. However, only the outdoor flow field was studied, and the influence of applying

RANS equations and turbulence models was not fully showcased for it lacked comparison with the situation where no turbulence model was used. As for other physics-informed models, for example, the Spline-PINN[18] studied the two-dimensional (2D) flow around a cylinder but lacked validation for Reynolds number over 100, the PIPN[19] only studied laminar flow, and the PINO[21] studied Kolmogorov flow and lid-driven cavity flow for solving Navier–Stokes equation only with the Reynolds number no higher than 500. Thus, simulating turbulence with high accuracy can be a common challenge for this category of physics-informed machine learning methods.

Recently, Wei *et al.*[27] employed PINN for the reconstruction of indoor airflow with the Reynolds number equal to 5000 and proved the PINN has a better performance on reconstruction accuracy than a purely data-driven fully connected neural network with the same amount of data, without embedding any turbulence models in the PINN. However, there is still a large error in their PINN result since turbulence is already present in the flow field. Based on their work, our research studies the competence of the PINN embedded with RANS equations and turbulence model on the same indoor airflow problem. Our research proves that it is vital for PINN to apply RANS equations and turbulence model as well as exploit its ability to utilize high-fidelity data for further accuracy improvement. With improved reliability, PINN can be applied for more complex flow cases involving turbulence, broadening its engineering application and development prospect. The influence of different numbers and locations of data together with the influence of different turbulence models are also studied. Our simulation is performed on the NVIDIA Modulus platform.[17]

II. METHODOLOGY

A. PINN configuration and loss equations

PINN is a deep learning model where boundary and initial conditions, residuals of governing equations, and possibly ground truth data if available can be put into loss terms for model training. Our PINN model embedded with the standard k–ε model is shown in Fig. 1.

For the 2D steady flow in our research, only spatial coordinates x and y are taken as inputs and there are no initial conditions in loss terms. The total loss is a weighted summation (weights denoted by λ) of the PDE loss calculated by residual equations within the domain, boundary loss between inference results and give boundary conditions, and the data loss between inference results and experimental data, represented by Eq. (1). The PDE loss and boundary loss are calculated on randomly sampled points, and the data loss is calculated on given data points

$$Loss_{total} = \sum_{m=1}^{5} \lambda_m Loss_{PDE,\ m} + \sum_{n=1}^{3} \lambda_n Loss_{boundary,\ n} + \lambda_{data} Loss_{data}, \tag{1}$$

where

$$Loss_{PDE,\ m} = \frac{1}{s}\sum_{i=1}^{s} r_m^2. \tag{2}$$

Here, s is the number of sampled points where the residuals are calculated. For our PINN model based on RANS equations with a standard k–ε model for 2D incompressible steady flow, the PDE residuals composing $Loss_{PDE,\ m}$ (m =1, 2, …, 5) are calculated with the following equations:

(1) Continuity equation

$$r_1 = \frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y}, \tag{3}$$

where $\bar{u}$ and $\bar{v}$ are time-averaged velocity components in the x and y directions.

(2) Momentum equations

$$r_2 = \bar{u}\frac{\partial \bar{u}}{\partial x} + \bar{v}\frac{\partial \bar{u}}{\partial y} + \frac{1}{\rho}\frac{\partial \bar{p}}{\partial x} - \frac{\partial}{\partial x}\left((\nu + \nu_t)\frac{\partial \bar{u}}{\partial x}\right) - \frac{\partial}{\partial y}\left((\nu + \nu_t)\frac{\partial \bar{u}}{\partial y}\right), \tag{4}$$

$$r_3 = \bar{u}\frac{\partial \bar{v}}{\partial x} + \bar{v}\frac{\partial \bar{v}}{\partial y} + \frac{1}{\rho}\frac{\partial \bar{p}}{\partial y} - \frac{\partial}{\partial x}\left((\nu + \nu_t)\frac{\partial \bar{v}}{\partial x}\right) - \frac{\partial}{\partial y}\left((\nu + \nu_t)\frac{\partial \bar{v}}{\partial y}\right), \tag{5}$$

where ν and ν_t are dynamic viscosity and turbulent viscosity.

(3) Turbulence kinetic energy (k) and dissipation rate (ε) equations

$$r_4 = \bar{u}\frac{\partial k}{\partial x} + \bar{v}\frac{\partial k}{\partial y} - \frac{\partial}{\partial x}\left(\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x}\right) - \frac{\partial}{\partial y}\left(\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial y}\right) - P_k + \varepsilon, \tag{6}$$

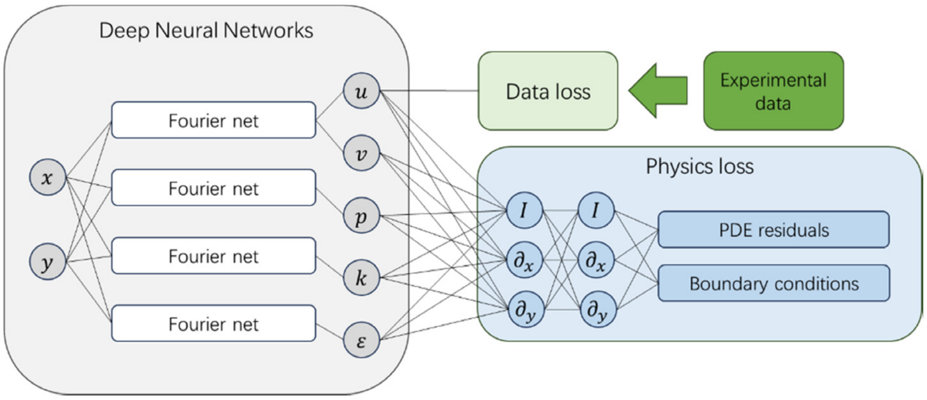


FIG. 1. Neural network architecture for PINN.

$$r_5 = \bar{u}\frac{\partial \varepsilon}{\partial x} + \bar{v}\frac{\partial \varepsilon}{\partial y} - \frac{\partial}{\partial x}\left(\left(\nu + \frac{\nu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x}\right) - \frac{\partial}{\partial y}\left(\left(\nu + \frac{\nu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial y}\right) - \frac{\varepsilon}{k}(C_1 P_k - C_2\varepsilon), \tag{7}$$

where turbulent viscosity and production term are

$$\nu_t = C_\mu \frac{k^2}{\varepsilon}, \tag{8}$$

$$P_k = \nu_t\left(2\left(\frac{\partial \bar{u}}{\partial x}\right)^2 + 2\left(\frac{\partial \bar{v}}{\partial y}\right)^2 + \left(\frac{\partial \bar{u}}{\partial y}\right)^2 + \left(\frac{\partial \bar{v}}{\partial x}\right)^2 + 2\frac{\partial \bar{u}}{\partial y}\frac{\partial \bar{v}}{\partial x}\right), \tag{9}$$

and the empirical constants and coefficients included are

$$\sigma_k = 1, \quad \sigma_\varepsilon = 1.3, \quad C_1 = 1.44, \quad C_2 = 1.92, \quad C_\mu = 0.09.$$

The $Loss_{boundary,n}$ (n =1, 2, 3) is the L2 loss between inference results and given boundary conditions for inlet condition, outlet condition, and no-slip wall boundaries separately. The $Loss_{data}$ is the L2 loss between the inference results and experimental data.

Neural networks tend to favor low-frequency solutions, a phenomenon that is known as "spectral bias,"[37] which can have a detrimental impact on training convergence and the accuracy of the model. Therefore, our study chooses the Fourier net to alleviate this problem. The Fourier net performs input encoding, thus transforming the inputs into a higher-dimensional feature space using high-frequency functions.[38] The Fourier net takes the following form:

$$u_{net}(\mathbf{x};\theta) = \mathbf{W}_n\{\phi_{n-1}\circ\phi_{n-2}\circ\cdots\circ\phi_1\circ\phi_E\}(\mathbf{x}) + \mathbf{b}_n, \tag{10}$$

where

$$\phi_i(\mathbf{x}_i) = \sigma(\mathbf{W}_i\mathbf{x}_i + \mathbf{b}_i). \tag{11}$$

Here, $u_{net}(\mathbf{x};\theta)$ is the approximate solution. $\mathbf{x}\in\mathbb{R}^{d_0}$ is the input to the network. $\phi_i \in \mathbb{R}^{d_i}$ is the ith layer of the network. $\mathbf{W}_i \in \mathbb{R}^{d_i\times d_{i-1}}$ **and** $\mathbf{b}_i \in \mathbb{R}^{d_i}$ are the weight and bias of each layer. $\theta = \{\mathbf{W}_1, \mathbf{b}_1, \ldots, \mathbf{b}_n, \mathbf{W}_n\}$ denotes the set of the network's trainable parameters. n is the number of layers. σ is the activation function. The Fourier net comprises standard feed-forward fully connected architecture, including the input encoding layer ϕ_E. We set each Fourier net composed of six hidden layers with layer size (d_i) equal to 512 (Fig. 1). The parameters of neural networks are optimized to minimize the total loss through backpropagation during training. The optimizer in our research is Adam. The learning rate is scheduled with Eq. (12) with the initial learning rate equal to 1×10^{-4}. Each model is trained with 400 000 steps with about 13 h. Our training takes place on an RTX 4090 GPU.

$$learning\ rate = 1\times10^{-4}\times0.95^{\frac{step}{8000}}. \tag{12}$$

B. Case description and CFD validation

The case features a forced convection flow in an empty room, which is proven to have a 2D nature[39] and can represent flows in mechanically ventilated rooms.[40] The experimental data of time-averaged horizontal velocity u are measured on four dashed lines $x = H$, $x = 2H$, $y = 0.5h_1$, and $y = H - 0.5h_1$ from the experiment conducted by Nielsen.[39] The 2D model is shown in Fig. 2, a long room with the height $H = 3$ m and the length $L = 3H$ where the air flows in through an inlet with the height $h_1 = 0.056H$ in the upper left corner, forming a jet, and flows out from an outlet with the height $h_2 = 0.16H$. The inlet velocity is parallel to the room ceiling with its magnitude $u_{in} = 0.455$ m/s. The Reynolds number for this case is 5000, taking h_1 as the characteristic length and u_{in} as the characteristic velocity. The outlet pressure is set as 0.

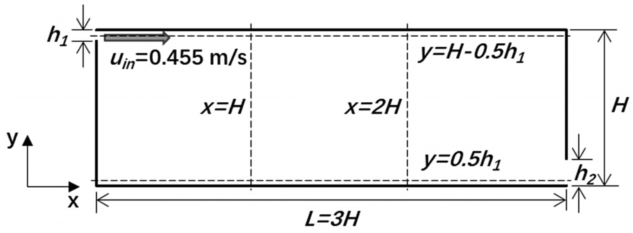


FIG. 2. Schematic of the indoor airflow case.

TABLE I. Configurations of the CFD simulation.

Item	Settings
Simulation software	ANSYS Fluent 2022
Simulation object	Air (incompressible, isothermal)
Turbulence model	Standard k–ε model
Cell number	905, 744
Inlet boundary conditions	Fixed velocity $u_{in} = 0.455$ m/s Turbulent intensity: 4%
Wall boundaries	No-slip wall boundary

We first calculated this case using CFD software ANSYS Fluent with settings in Table I. The results of CFD prediction will serve as the reference for PINN results. Therefore, we first need to validate CFD with the experiment. The results are validated with available horizontal velocity data from the experiment[39] along the four dashed lines in Fig. 2. The comparison is shown in Fig. 3, where the CFD results align well with experimental data on all four lines. Both the velocity variance within the jet area along the room ceiling and the reverse flow region along the room floor are well captured. Figure 5 further presents the predicted velocity and pressure fields by validated CFD simulation. Apart from an inlet jet along the room ceiling, other features include a significant downwash flow along the right-sidewall and a reverse flow along the room floor, forming a major vortex. There are also some minor vortices formed in the upper right corner and lower left corner, according to experimental data in Fig. 3. For the pressure field, the flow separations near the upper right corner and near the outlet in the lower right corner form two high-pressure zones, respectively, as can be seen in pressure field CFD result in Fig. 5.

C. Sampling method

In PINN, residual loss and boundary loss are calculated on residual points. It is encouraged to occasionally resample these residual points during training rather than using fixed points to achieve

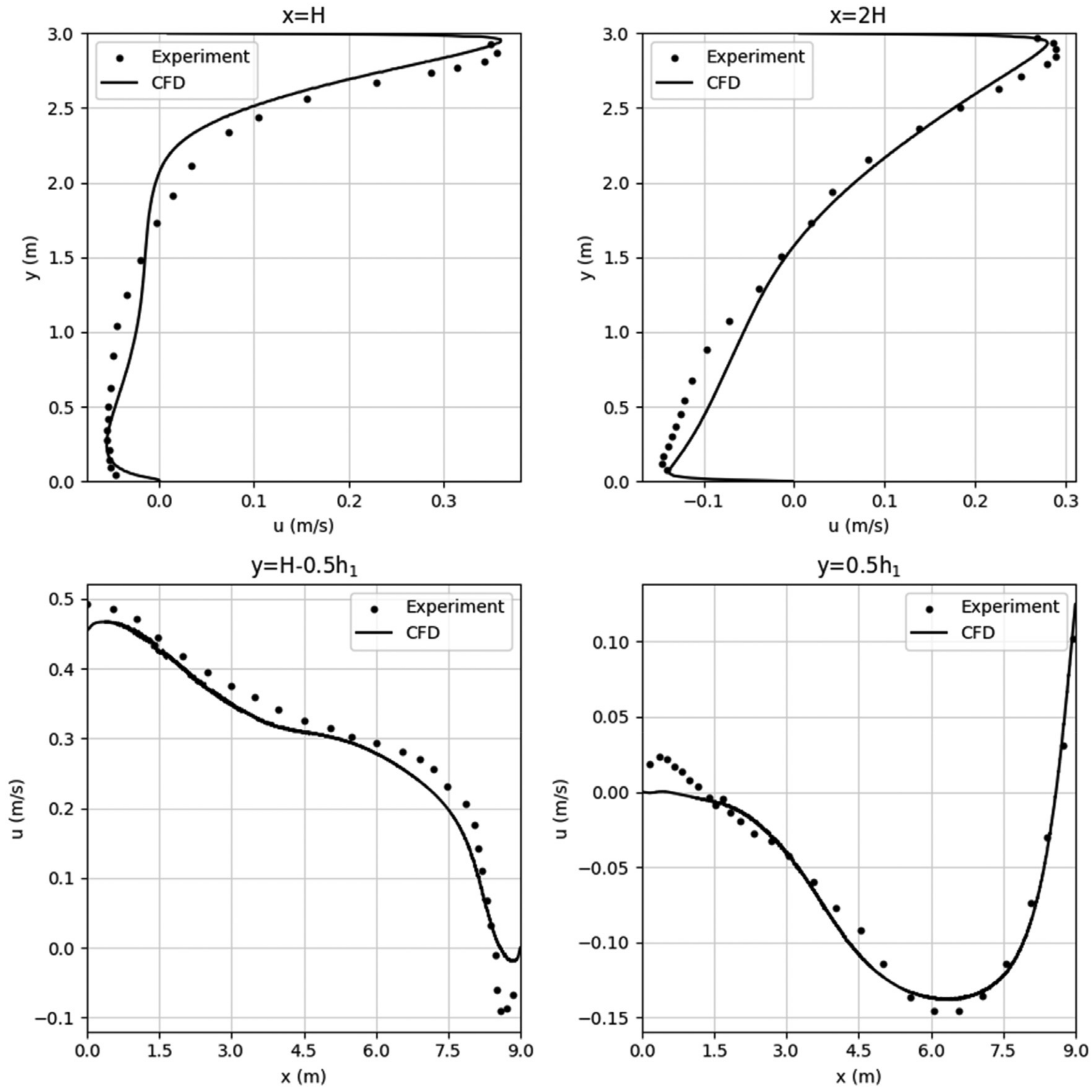


FIG. 3. CFD result validation.

better accuracy.[29] Here, in our research, we use a map-style dataset by default in Modulus.[17] For example, as shown in Fig. 4, for a batch per epoch equal to 1 and batch size equal to 10, there will be 10 points in total, and residual points will be fixed at every step during the training. However, for batch per epoch set to 2 and batch size equal to 10 with shuffling, it does not just generate two different sets of sampling, each with 10 residual points, but a combination of 10 points out of 20 in total. Despite some points being the same for two different training steps, like the two in the red circle and the three in the blue circle in Fig. 4(b), residual points for different steps can still hardly be the same. With this method, although the total residual points for sampling are fixed from the start of training and there is no real resampling, enough randomness is still guaranteed for the training to be practical.

Analogous to mesh independence test for CFD methods like the FVM, for PINN training, although a smaller batch size occupies less computational resource, e.g., GPU memory, it can lead to larger prediction error. To compare the prediction accuracy quantitatively, the root mean square errors (RMSEs) of flow field variables are calculated. Take horizontal velocity u for example,

$$RMSE_u = \sqrt{\frac{\sum_{i=1}^{n} |u_{pred} - u_{ref}|^2}{n}}. \tag{13}$$

The RMSEs are calculated on randomly sampled 468, 851 points where the CFD data are available as reference data. In Eq. (13), $n = 468, 851$, u_{pred} stands for PINN results, and u_{ref} stands for CFD results. To choose a suitable batch size, we tested three different cases

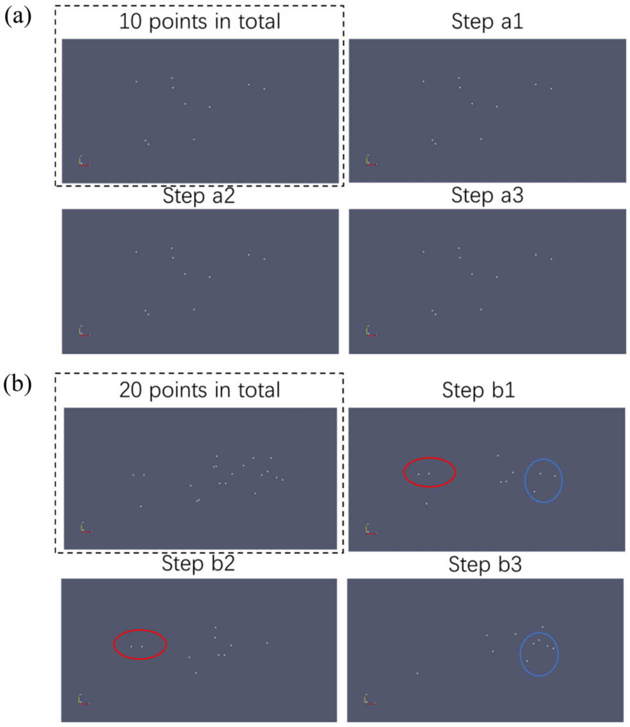


FIG. 4. Illustration of the residual points sampling method for PINN training with batch size = 10, where in (a) batch per epoch = 1 and in (b) batch per epoch = 2.

with the standard k–ε model embedded as listed in Table II. No experimental data are added during training. The batch per epoch is set to 1000.

With insufficient batch size (small batch size in Table II), the RMSE for horizontal velocity u is 30.4% higher than that of medium batch size, occupying 2800 MiB less GPU memory. However, compared to medium batch size, the RMSE is only reduced less than 5% when employing large batch size, while occupying 4480 MiB more GPU memory. We chose the medium batch size in Table II for all the following cases in Sec. III. The results for training with small batch size and large batch size without experimental data can be found in Fig. 14. The loss curves during training can be found in Fig. 15. Some steep fluctuations appeared especially in the early stage of training are likely caused by changed positions of residual points that are sampled for different steps.

III. RESULTS

A. Influence of assimilating experimental data

First, a PINN with the standard k–ε model is trained without experimental data. This pure physics-driven training aims to find a relatively suitable set of hyperparameters for the physical problem. Although the hyperparameters, especially the weights of different loss terms, are tuned so that the overall predictions are as close to the CFD results as possible, the attachment of reverse flow on the domain bottom is still poorly predicted by PINN as suggested by Fig. 5 no data point case. The reverse flow detached from the room floor somewhere near $x = 5.5$ m, a phenomenon that should not happen as suggested by CFD results in Fig. 5. Through the ability of PINN to utilize high-fidelity data, a total of 28 data points located on the $y = 0.5h_1$ line are added to the model, providing extra information. Informed by these additional data, the model can now give a better prediction as the prediction accuracy of the reverse flow attachment is significantly improved, as shown in the case of Fig. 5 with 28 data points. As a result of the enhancement, the absolute error of u prediction in the lower half domain is significantly reduced. In addition, the prediction error of v in the near right wall region is also lowered noticeably.

In general, for both scenarios, more significant prediction errors appear at the jet interface near the inlet for the horizontal velocity u, and the prediction error of pressure is higher in the right half domain (except the two corners), where the change of velocity can be more drastic and thus the flow structure more complex than in other regions.

The RMSEs for PINN prediction without any data are 0.040 257, 0.018 462, and 0.012 891 for u, v, and p predictions, respectively, while the RMSEs for scenario with 28 data points in total are 0.026 311, 0.011 008, and 0.006 031. By aiding the PINN with extra ground truth data, the RMSEs of prediction are greatly reduced, especially for

TABLE II. Comparison of training with different batch sizes (with no experimental data).

Case	Batch size		RMSE$_u$	GPU memory (MiB)
Small batch size	No-slip wall:	100	0.052 482	922
	Domain interior:	200		
	Inlet/outlet:	5		
	Total:	**310**		
Medium batch size	No-slip wall:	1000	0.040 257	3722
	Domain interior:	2000		
	Inlet/outlet:	50		
	Total:	**3100**		
Large batch size	No-slip wall:	2000	0.038 468	8202
	Domain interior:	4000		
	Inlet/outlet:	100		
	Total:	**6200**		

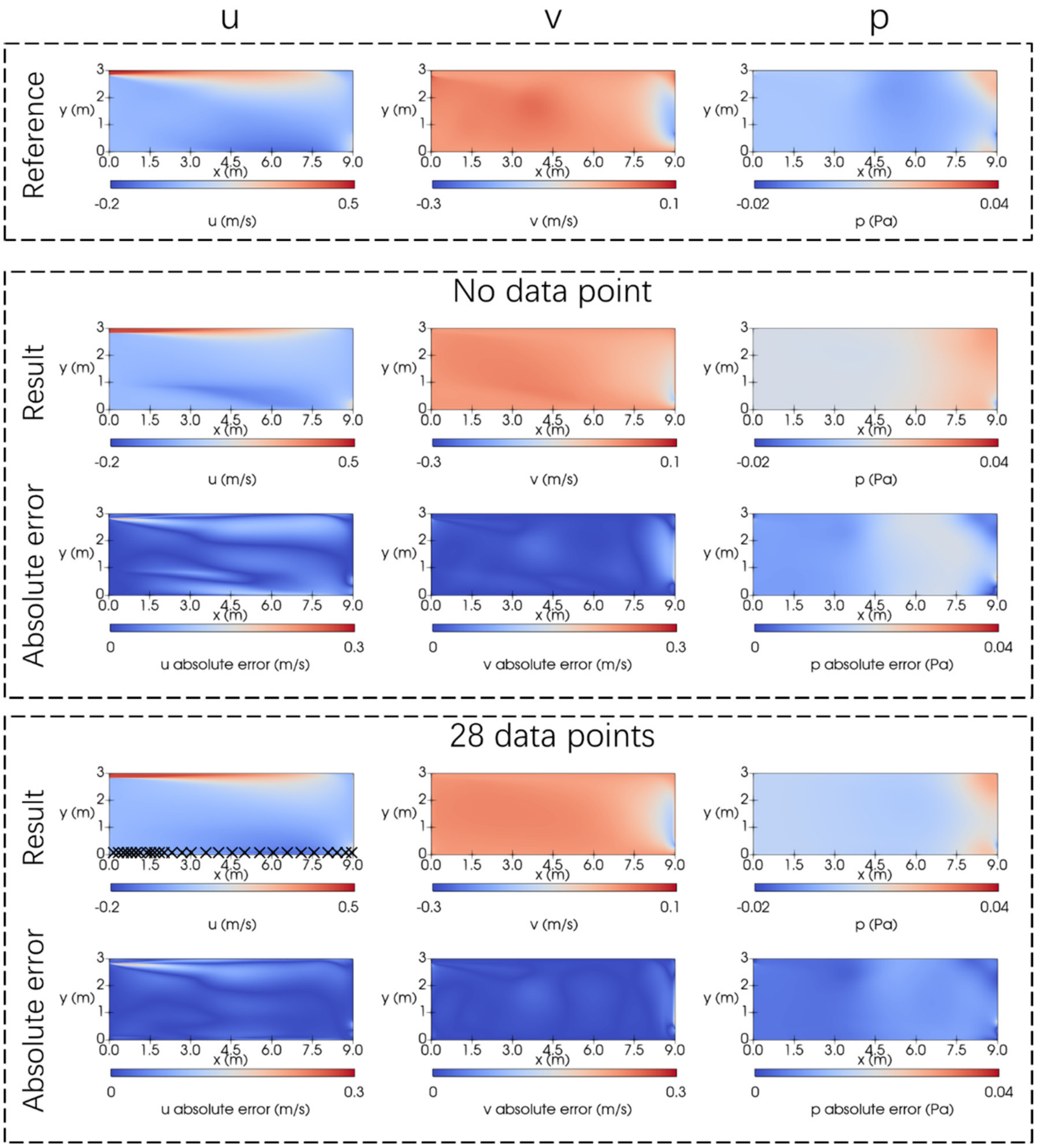


FIG. 5. Validated CFD results, PINN predictions, and absolute error with no data and 28 data points (marked with ×). *u* and *v* are the velocity components in the *x* and *y* directions; *p* is the pressure.

pressure prediction, for which the RMSE is reduced by more than 50%. The RMSE for *u* prediction is reduced by 34.6%, and for v prediction, it is reduced by 40.4%.

B. Influence of data points number

Despite the significant improvement in PINN prediction accuracy with experimental data in Sec. III A, considering it is just a simple 2D case, a total of 28 data points can still be somewhat unacceptable in number, for it is not without cost and hardship to install this number of sensors and collect data in the real world. Here, the PINN prediction accuracy with fewer data points is further studied with different numbers of data points. The number of data points is halved once a time by removing neighboring points, beginning with 28 points. The PINN prediction results and absolute error with 14, 7, 4, and 2 data points on the $y = 0.5h_1$ line are shown in Fig. 6, with the locations of data points

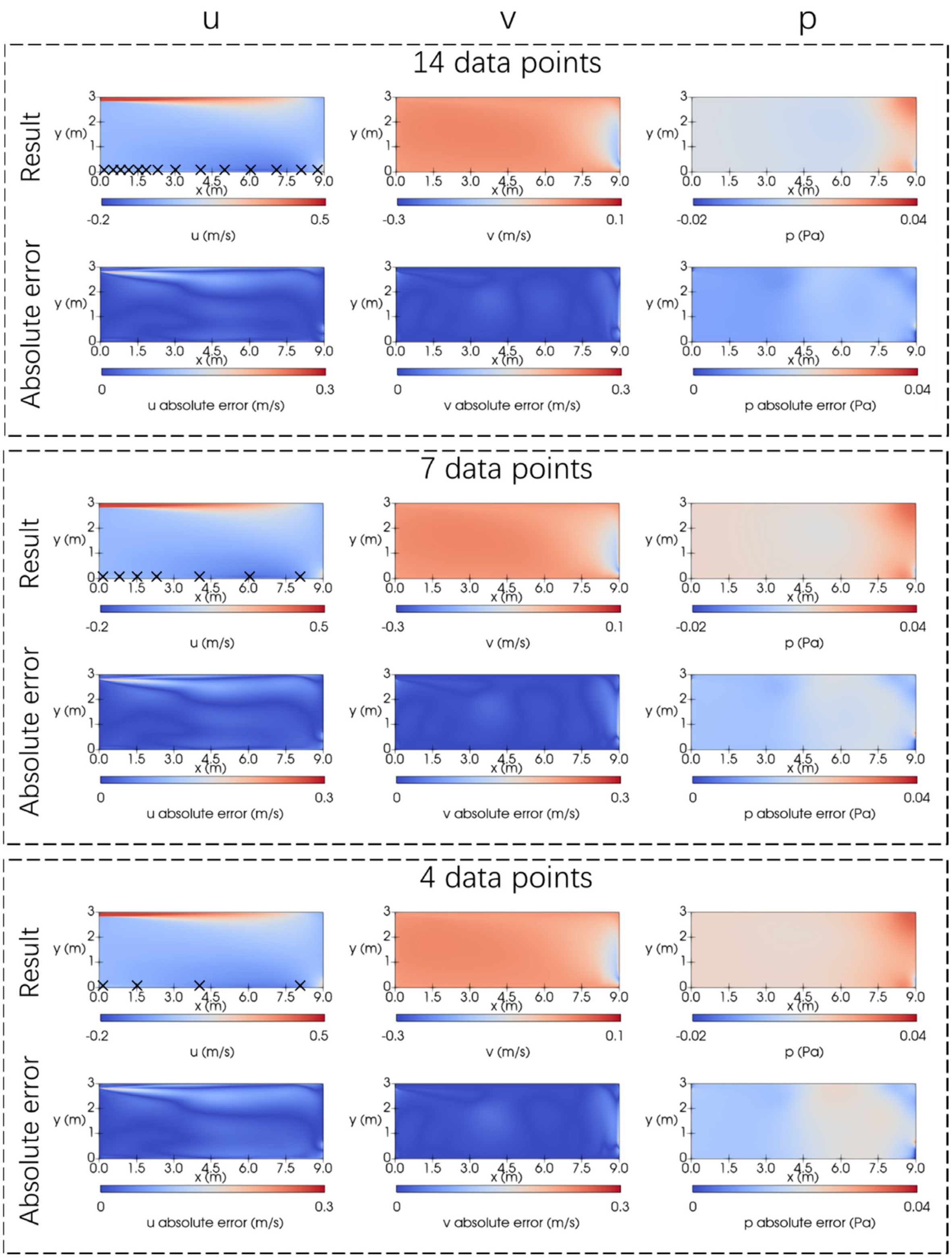


FIG. 6. PINN prediction and absolute error with 14, 7, 4, and 2 data points (marked with ×), respectively.

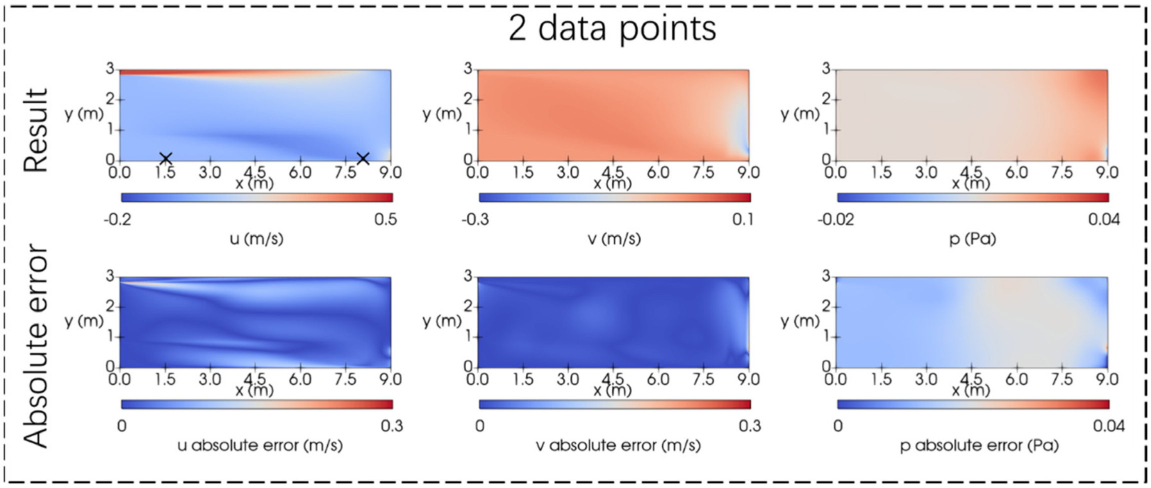


FIG. 6. (*Continued.*)

marked with ×. Generally, the overall error distributions are similar to those in Sec. III A. For velocity prediction, larger errors are still observed on the jet interface near the inlet for each case, and the prediction error for pressure is higher in the right half domain than in the left, with more minor errors appearing in the corners. Yet, one thing worth noticing is that, before the number of data points is reduced to 2, the predicted reverse flow attachments on the domain bottom are still relatively similar. However, for the two data points scenario in Fig. 6, the reverse flow attachment prediction suddenly worsens compared to the four data points in Fig. 6, with an erroneous detachment prediction similar to that using zero data. Consequently, the horizontal velocity prediction errors in the lower half domain become noticeably larger. The case with two data points will be further studied later in Sec. III C.

The RMSEs of predictions in each case are shown in Fig. 7. For velocity prediction, with relatively fewer data points, the prediction accuracy improvement is more significant by adding more data. For prediction with four data points, the RMSE is decreased by 24.7% for horizontal velocity u and 29.0% for vertical velocity v, compared with the case where no data are added. However, adding more data points becomes less helpful when enough data already exists. The RMSEs for both u and v are the same for the case with 28 and 14 data points, as seen in Fig. 7. This means a balance between more accurate prediction results and lower data acquisition costs exists. For pressure prediction, the results are more complex. As data points are added from 0, the RMSE first rises, then decreases. One possibility is that, by adding very few ground truth data only for velocity, the obedience of physical laws is compromised as the model tries to fit for these newly added data, making the pressure prediction a little worse. The RMSE for pressure prediction with four points is 30.1% higher than with no data. However, after the data are increased to a certain value, it becomes enough for the model to find an overall better solution, and the RMSE for pressure prediction tends to decrease as the number of data points increases. Eventually, the RMSE for pressure prediction with a total of 28 points is 53.2% lower than with no data.

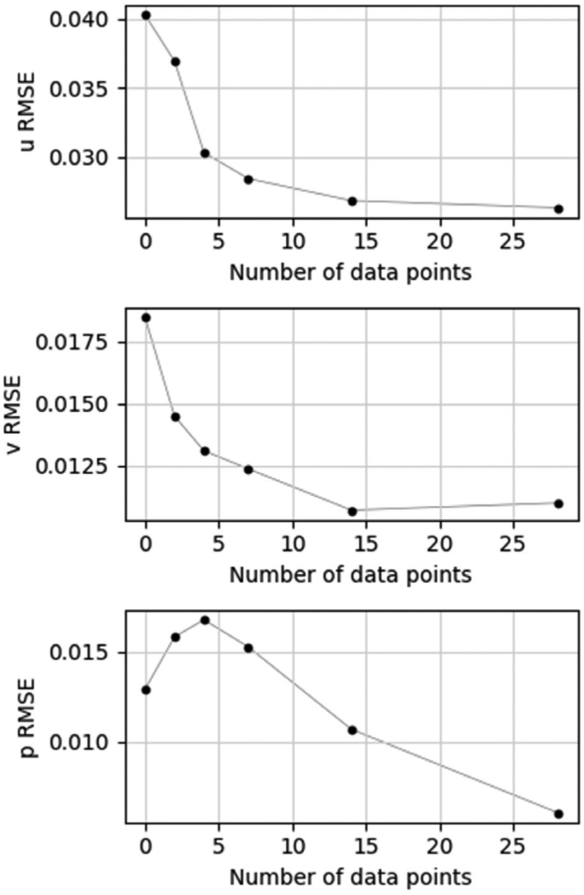


FIG. 7. Influence of data points number on PINN prediction accuracy.

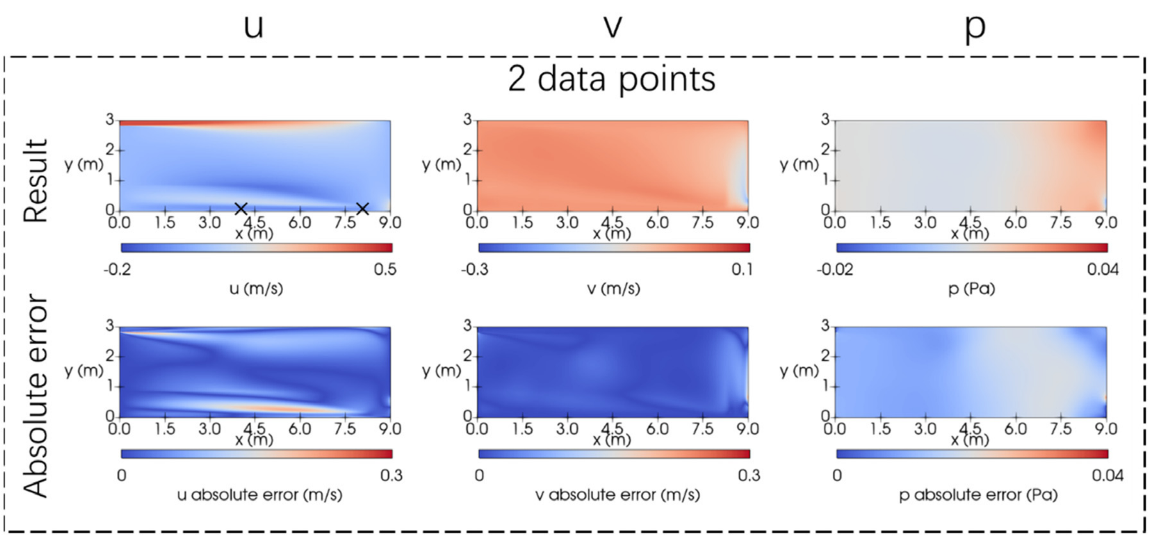


FIG. 8. PINN prediction and absolute error with two data points (marked with ×) at a different position.

C. Influence of data points position

In Sec. III B, we find the apparent erroneous prediction of reverse flow detachment on the domain bottom occurs when the number of data points is reduced from 4 to 2. On the one hand, it is likely because there is just not enough data to provide sufficient information for the PINN model. On the other hand, it is also possible that some data at specific locations might be more important for the prediction of the reverse flow, and our former trial with two data points in Sec. III B from four data points happens to remove the more important points, yielding a less ideal result. Here, we rechoose the two data points closer to the reverse flow region as marked with × in Fig. 8 from scenario of the four data points in Fig. 6 to see whether providing only two data points can still be effective on a more reasonable flow attachment prediction with better data locations.

The results prove the latter assumption to be negative. Despite the two data points being specifically chosen to be close to the reverse flow region, the reverse flow prediction is still far from ideal. Even larger prediction errors for horizontal velocity are observed near the domain bottom in Fig. 8. To fit the two data points, the model prediction deviated further from physics, causing the overall prediction error to rise instead. The RMSEs for u, v, and p predictions are 0.050 028, 0.017 317, and 0.013 444, respectively. The velocity prediction with the newly chosen two points has higher RMSEs than the case of two data points in Sec. III B, by 35.5% for horizontal velocity u and by 19.4% for vertical velocity v. In contrast, the horizontal velocity u prediction RMSE is even higher than the case without any data in Sec. III A by 24.3%. These findings show that although the importance of different data point locations can be intuitively different, a sufficient data number can still be more crucial for prediction result improvement.

D. Influence of turbulence models

This section studies the influence of applying the RANS-based turbulence model with PINN. On the basis of our PINN with the standard k–ε model, two more PINN models are studied for comparison, one with the Prandtl's one-equation model and the other one with no turbulence model embedded. Their residual equations for PDE loss can be found in Appendix. These two models have the same hyperparameter as the one used in the former results and utilize the same 28 data points as the case in Sec. III A.

Figure 9 shows the PINN prediction results and absolute error distribution for PINN with no turbulence model and one-equation model, respectively. Compared to the reference CFD results in Fig. 5, the PINN prediction with the standard k–ε model, as shown in Fig. 5, is much better than those with no turbulence model and the one-equation model. Intuitively, the results with no RANS model seem straightforward, without smoother transitions and many details. The possible reason is that due to the Reynolds number exceeding 5000, turbulent features and small-scale eddies appear in the flow domain of this model, resulting in more high-frequency turbulent signals. Without incorporating the RANS model, the PINN directly learns from the Navier–Stokes equations, similar to DNS, requiring the handling of full-frequency turbulent signals (Fig. 16).[41] Therefore, although Fourier net is employed in this study, the high-frequency signals introduced by turbulence reduce the predictive capability of the neural network. However, when the RANS model is incorporated, the fluctuating characteristics of the velocity and pressure fields are neglected, and the flow field is averaged, effectively excluding high-frequency turbulent signals (Fig. 16). Consequently, the predictive ability of the PINN is greatly enhanced when the RANS model is incorporated.

In addition, the results with Prandtl's one-equation model seem to be an intermediate state between those with no turbulence model and those with the standard k–ε model. This should be attributed to Prandtl's relatively poor ability to predict the turbulence viscous using Prandtl's one-equation model. Similar to the former results with the standard k–ε model, more significant errors appear on the jet surface near the flow inlet for horizontal velocity prediction, and the error is larger in the right half domain than in the left for pressure prediction.

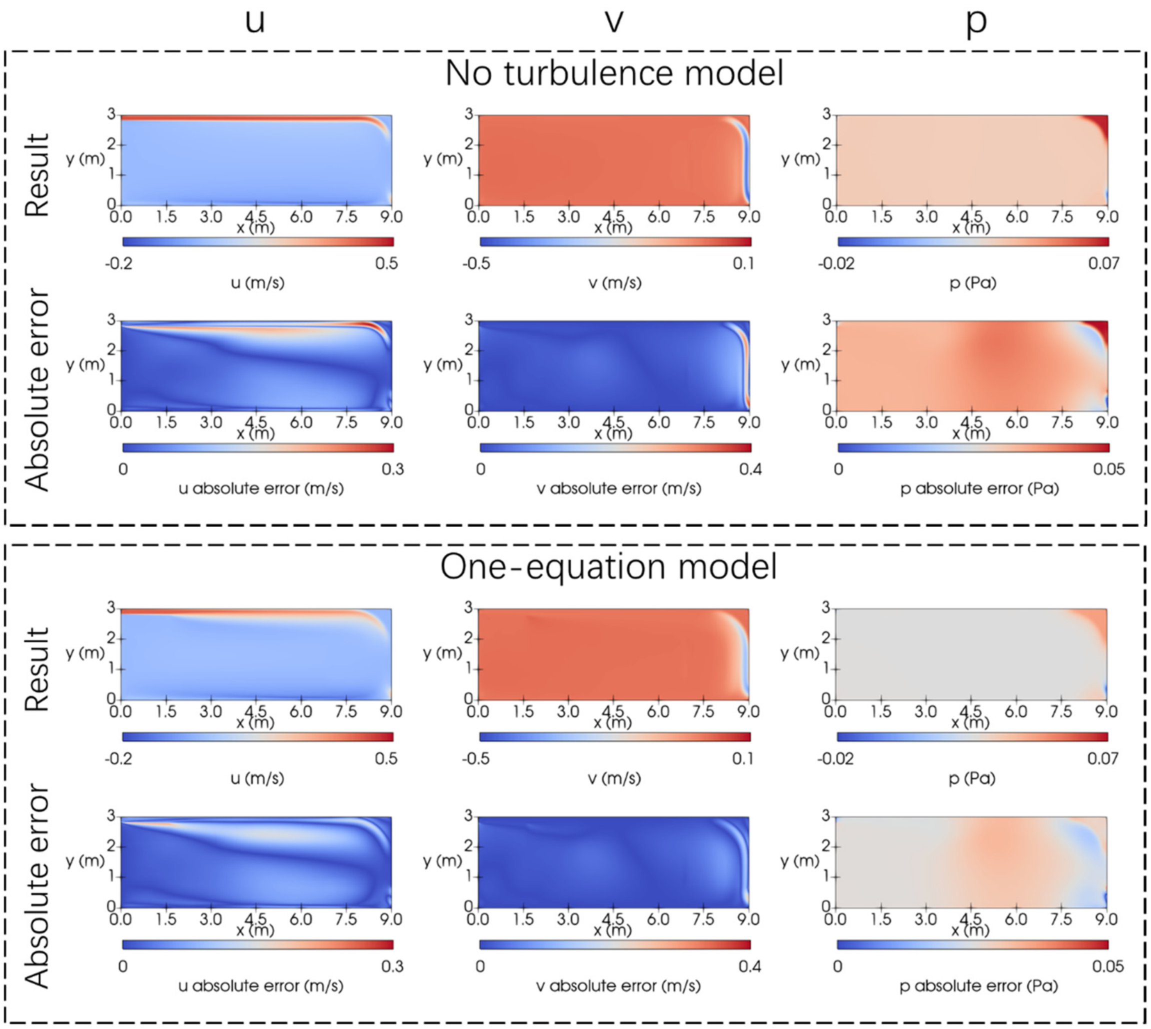


FIG. 9. PINN prediction and absolute error with no turbulence model and the Prandtl's one-equation model.

RMSE comparisons in Fig. 10 further illustrate the advantage of using turbulence models for PINN with experimental data, especially the standard k–ε model. After applying the turbulence model, the RMSE for horizontal velocity u prediction is 59.4% lower with the standard k–ε model and 22.1% lower with the Prandtl's one-equation model, respectively, than with no turbulence model. For vertical velocity v prediction, the RMSE is 70.5% lower with the standard k–ε model and 34.8% lower with the Prandtl's one-equation model. For pressure prediction, it is 82.9% lower with the standard k–ε model and 22.6% lower with the Prandtl's one-equation model.

We can see that, for the 2D indoor airflow case with a higher Reynolds number, the same as CFD, it is also necessary for the PINN to include a turbulence model in the loss functions for a better prediction of the turbulent flow and a complex turbulence model can give better results. It is strongly suggested that researchers in this area consider applying turbulence models in PINNs when dealing with turbulent flow problems.

The architecture for the PINN without turbulence model is shown in Fig. 17(a), and the other with Prandtl's one-equation model is shown in Fig. 17(b). For the one-equation model, there is one less Fourier net for the turbulent dissipation rate prediction. For the case with no turbulence model, there is further one less Fourier net for the turbulent kinetic energy. One might argue that, since for PINN models without turbulence model and with the Prandtl's one-equation model, the size of neural networks is also reduced (with fewer Fourier nets) compared to the one with standard k–ε model, so the different performance of different PINN models can also due to the model size. This will be further studied in Sec. III E.

E. Influence of architectures

To study the influence of architectures here in our research, the PINN model size with standard k–ε model and the one-equation model is reduced to keep the same as the one with no turbulence

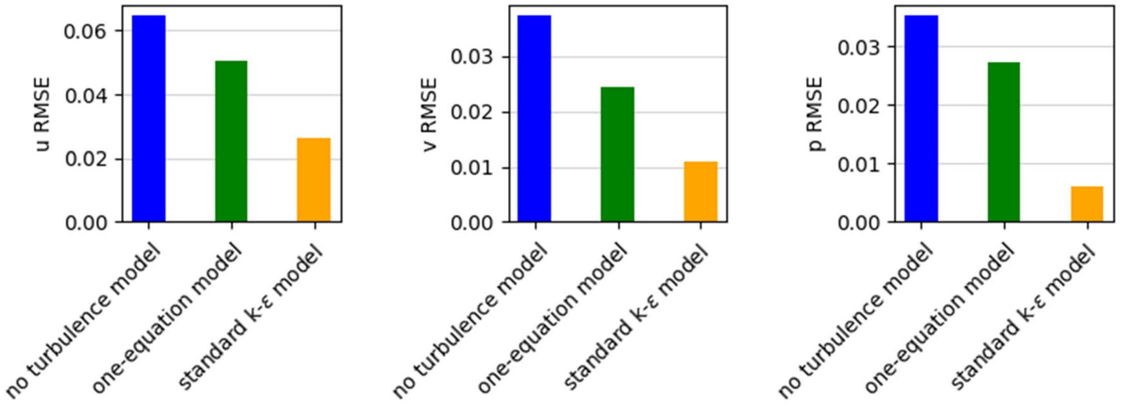


FIG. 10. Influence of turbulence models on PINN prediction accuracy.

model in Sec. III D, as shown in Fig. 11, where Fourier net 1 is for prediction of all the other variables except pressure, which is specially predicted by Fourier net 2, to keep the same with the PINN with no turbulence model. The same 28 data points of horizontal velocity on the $y = 0.5h_1$ are utilized.

There is basically no significant difference in prediction results compared to former trials in Secs. III A and III D, except the downwash flow near the right wall now floats a little further away to the left, as shown in Fig. 12. According to the RMSEs of predictions in Fig. 13, it can be seen the conclusion in Sec. III D still stands. PINNs with the turbulence model still have better prediction accuracy, while the one with the standard k–ε model is better than that with the Prandtl's one-equation model. Compared to RMSEs shown in Fig. 10, we can see that after the model size is reduced, the prediction accuracy of velocity is slightly better for the one with the Prandtl's one-equation model, while the prediction accuracy of pressure is slightly worse; for the PINN with the standard k–ε turbulence model, after the model size is reduced, the velocity prediction accuracy becomes worse while the pressure prediction accuracy remains almost the same.

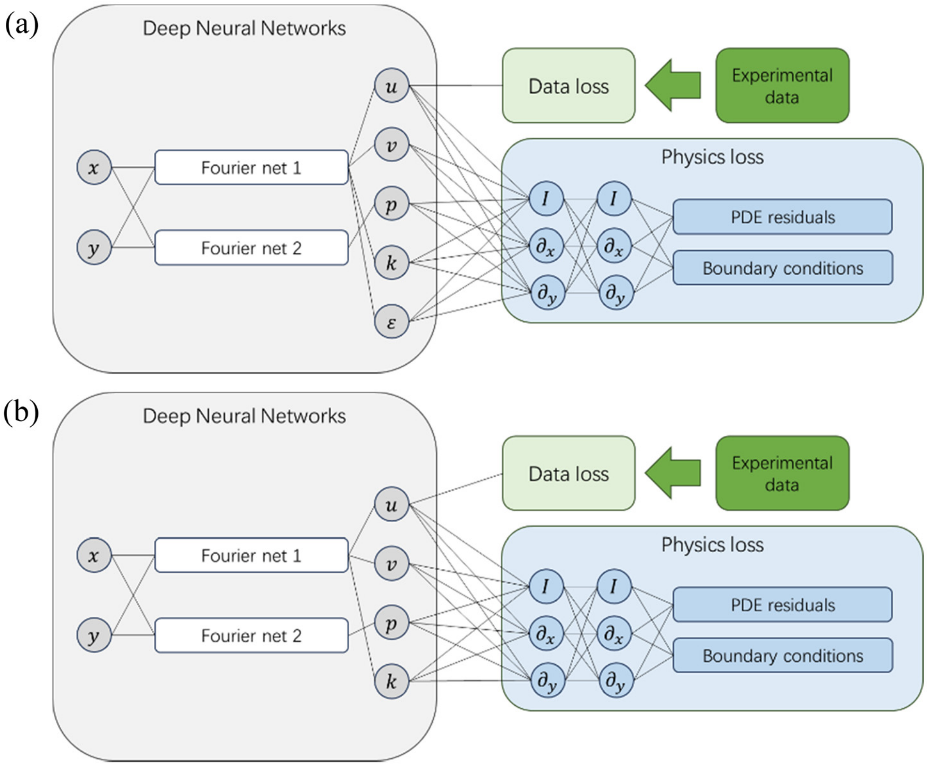


FIG. 11. Neural network architecture with reduced size for (a) the standard k–ε model and (b) the Prandtl's one-equation model.

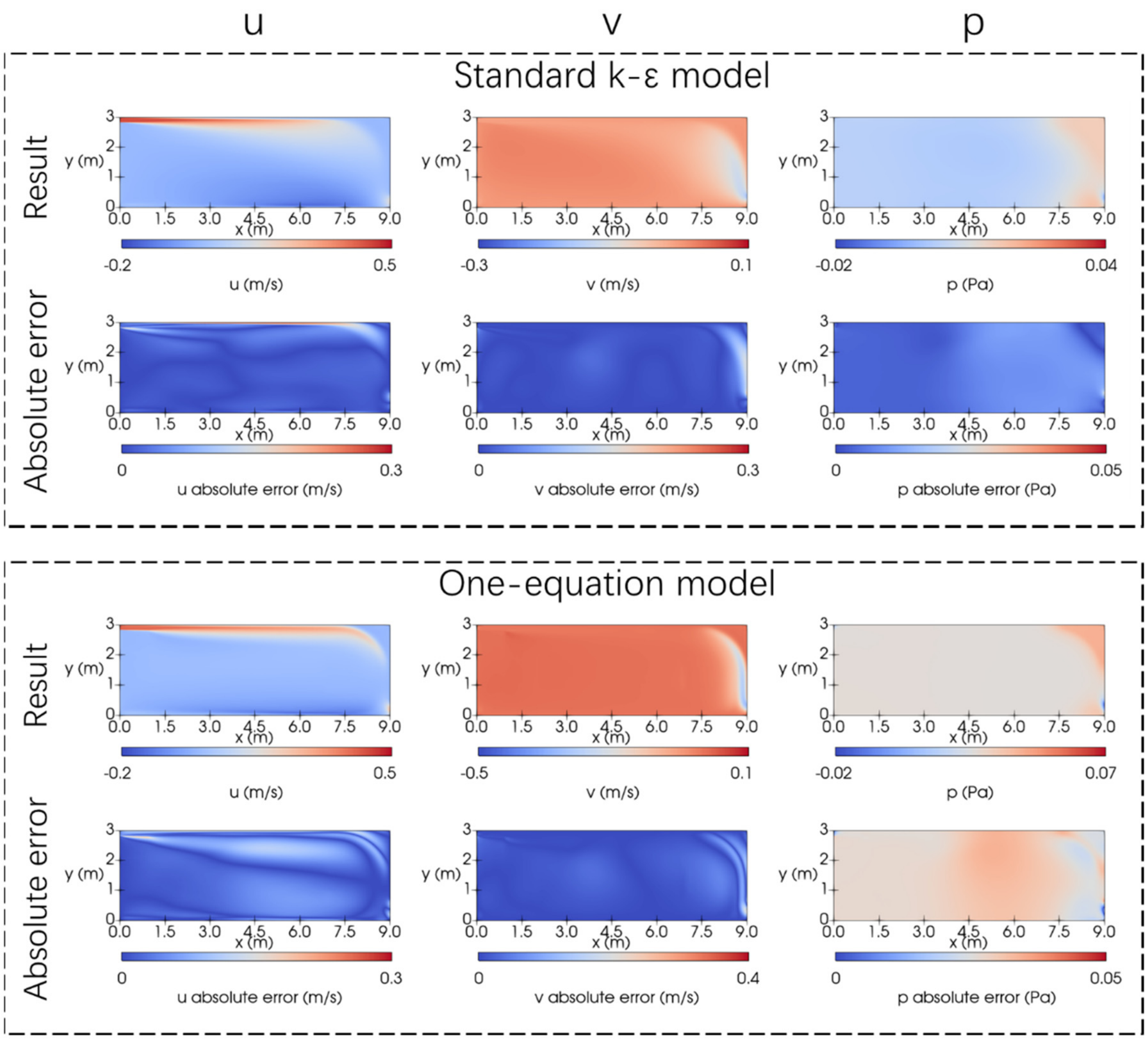


FIG. 12. PINN prediction and absolute error with the standard *k*–ε model and the Prandtl's one-equation model with reduced model size.

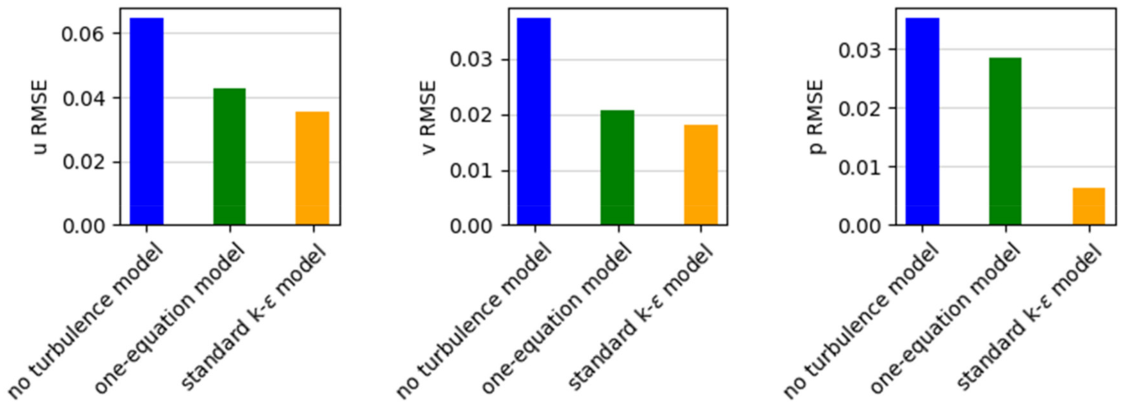


FIG. 13. Influence of turbulence models on PINN prediction accuracy with uniform model size.

IV. DISCUSSION

Despite the application of RANS equations, our PINN embedded with the standard k–ε model still fails to give more accurate results without the extra help of experimental data. Actually, classical PINN for flow field simulation can have two major technical limitations. Apart from the aforementioned "spectral bias" problem,[30,37,38] i.e., the difficulty for PINN to learn high-frequency features in multiscale problems, the other challenge is training with complicated loss functions where multiple terms are involved, making it a highly non-convex optimization problem.[30] For the 2D turbulent indoor airflow case in our research, while the implementation with RANS equations and turbulence model proved effective for the mitigation of the former problem, more terms and residual equations are introduced, making the non-convex optimization problem even more challenging. Multiple loss balancing algorithms (e.g., SoftAdapt,[42] GradNorm,[43] and Neural Tangent Kernel (NTK) approach[44]) might be helpful for dealing with the latter problem.

As for other physics-informed machine learning models (e.g., Spline-PINN,[18] PIPN,[19] and PINO[21]) with distinct architectures, although current studies on the spectral bias problem[30,37,38] mainly focused on the more traditional fully connected neural networks, these models are all trained with the same optimization algorithms (e.g., Adam). The steep gradients brought by high-frequency features can be a shared challenge during model training. It is expected the conclusion of this research can have reference value in a broader aspect.

V. CONCLUSION

To improve the reliability of PINN for turbulent flow and promote its application on more complex flow problems encountered in the built environment area, our research studies the PINN on a 2D indoor turbulent flow case and significantly enhances its prediction accuracy by exploiting its ability to utilize high-fidelity data and applying RANS equations and turbulence models. We also study the influence of using different numbers and positions of data points for utilizing experimental data and the impact of changing neural network architectures. Here are our conclusions.

(1) An accurate prediction of indoor turbulent flow remains a challenge for PINN with purely physics-driven training. By utilizing available experimental data during training, the overall prediction accuracy of PINN can be enhanced significantly. In our case, the prediction accuracy is improved by 53.2% for pressure and by 34.6% and 40.4% for horizontal velocity u and vertical velocity v, respectively.

(2) Using a sufficient number of data is essential for utilizing experimental data. Adding a small number of data points, four points in our case, can already be quite efficient in improving the overall prediction on flow field structure. The improvement will be less significant as more and more data points are added. Thus, a balance can exist between the improvement of prediction accuracy and the cost of data acquisition.

(3) Embedding RANS equations and turbulence model in PINN can be highly beneficial for model training, enhancing its prediction accuracy, while a more sophisticated turbulence model can have better results. After the standard k–ε model is applied, the prediction accuracy is improved by 82.9% for pressure and 59.4% and 70.5% for horizontal velocity u and vertical velocity v, respectively, compared with applying no turbulence model. Despite currently less studied by researchers, developing turbulence models for PINN can have great potential.

Our research is still limited for indoor airflow scenarios, for it only studies a 2D case with Reynolds number equal to 5000 and the PINN used in the research fails to give a more accurate result independently without the extra help of experimental data. For further engineering applications, we will dedicate to the continuous improvement of the PINN model in the future and test it on more 3D cases and also with higher Reynolds numbers.

ACKNOWLEDGMENTS

The authors would like to express their gratitude for the funding support provided by the Innovation and Technology Commission (Grant No. K-BBY1) and Environment and Conservation Fund (Grant No. ECF 29/2022) of the Hong Kong SAR Government.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Chi Zhang: Investigation (equal); Methodology (equal); Writing – original draft (equal). **Chih-Yung Wen:** Supervision (equal); Writing – review & editing (equal). **Yuan Jia:** Investigation (equal); Software (equal); Visualization (equal). **Yu-Hsuan Juan:** Conceptualization (equal). **Yee Ting Lee:** Conceptualization (equal); Writing – review & editing (equal). **Zheng-Wei Chen:** Investigation (equal); Methodology (equal). **An-Shik Yang:** Writing – review & editing (equal). **Zhengtong Li:** Conceptualization (equal); Supervision (equal).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

APPENDIX: SUPPLEMENTARY MATERIALS

Based on Prandtl's mixing length model, the loss equations for Prandtl's one-equation model are similar to those for the standard k–ε model in Sec. II A, except the turbulent dissipation ratio ε now does not have a separate equation but instead is calculated directly by

$$\varepsilon = C^* \frac{k^{3/2}}{l}. \tag{A1}$$

The turbulent viscosity and mixing length are

$$\nu_t = C_\mu l k^{1/2}, \tag{A2}$$

$$l = \min(0.419d, 0.09d_{max}), \tag{A3}$$

in which d is the normal distance from the wall and d_{max} is the maximum normal distance. The empirical coefficients are

$$C^* = 0.18, \quad C_\mu = 0.09.$$

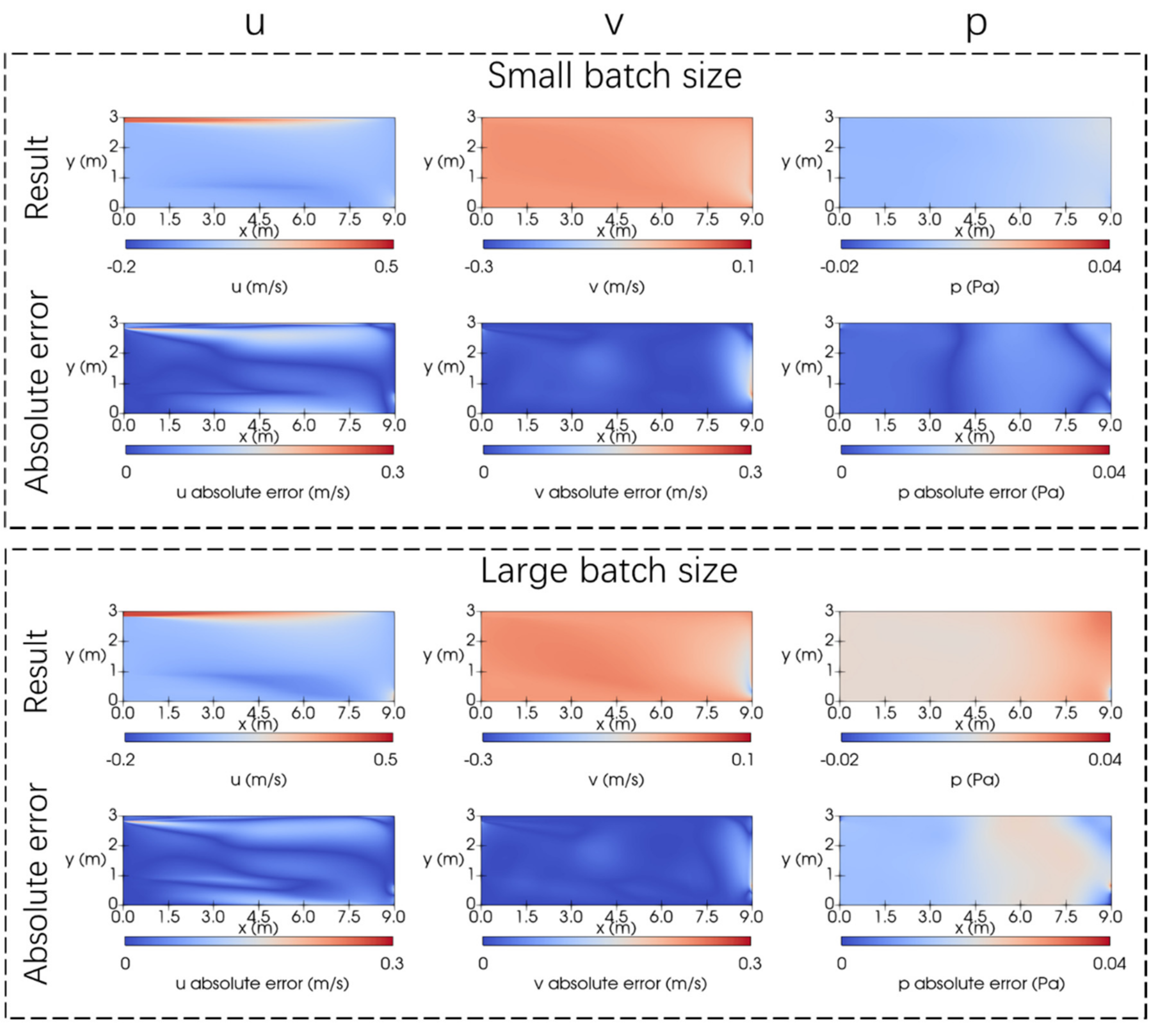


FIG. 14. PINN prediction and absolute error with small batch size and large batch size for testing.

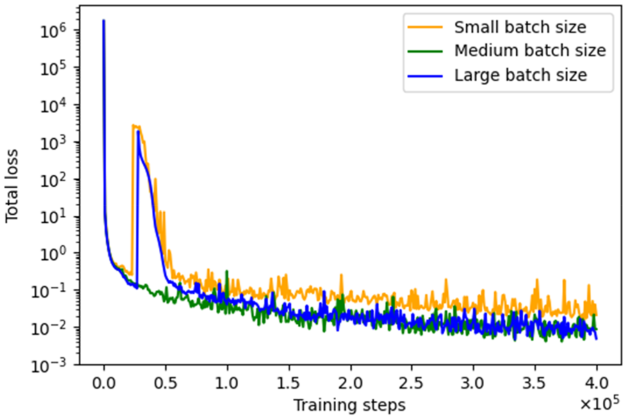


FIG. 15. Recorded loss fluctuation during training with different batch sizes.

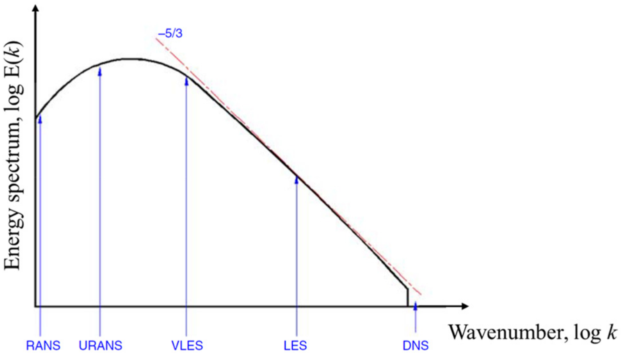


FIG. 16. Illustration of RANS-DNS energy spectra against wavenumber.[41]

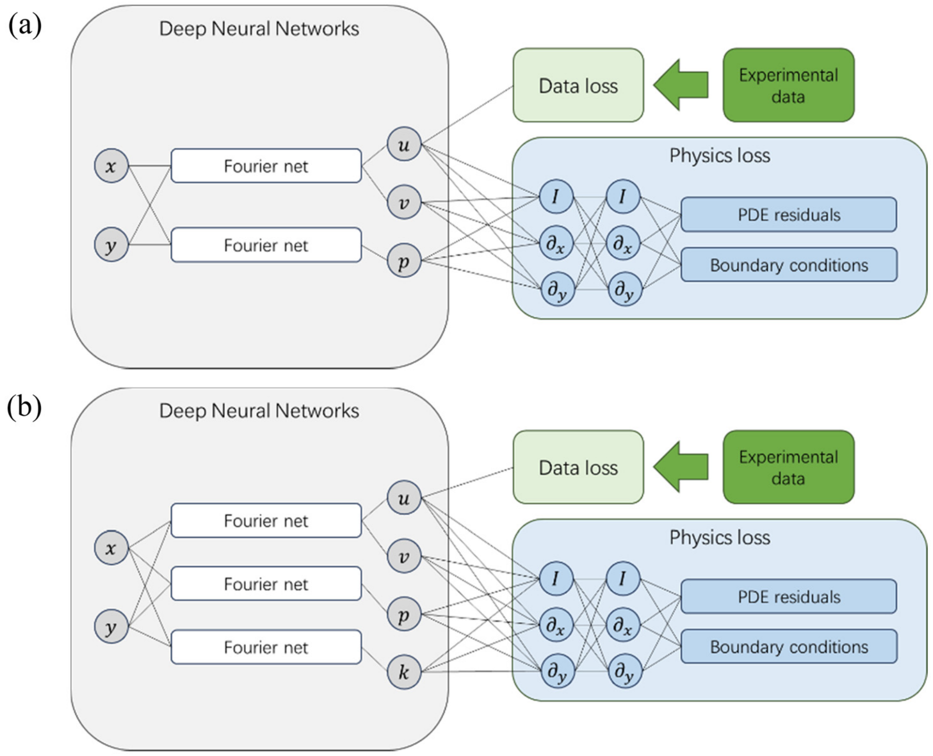


FIG. 17. Neural network architecture for (a) no turbulence model and (b) the Prandtl's one-equation model.

Navier–Stokes equations are directly used for governing PDEs without turbulence models and the residual equations are as follows:

(1) Continuity equation

$$r_1 = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}, \tag{A4}$$

where u and v are instantaneous velocity components in the x and y directions.

(2) Momentum equations

$$r_2 = u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial x} - \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \tag{A5}$$

$$r_3 = u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial y} - \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right). \tag{A6}$$

REFERENCES

[1] X. Guo, W. Li, and F. Iorio, "Convolutional neural networks for steady flow approximation," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2016), pp. 481–490.

[2] C. Sabater, P. Stürmer, and P. Bekemeyer, "Fast predictions of aircraft aerodynamics using deep-learning techniques," AIAA J. **60**(9), 5249–5261 (2022).

[3] A. Y. Sun, Z. Li, W. Lee, Q. Huang, B. R. Scanlon, and C. Dawson, "Rapid flood inundation forecast using Fourier neural operator," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (IEEE, 2023), pp. 3733–3739.

[4] G. Wen, Z. Li, Q. Long, K. Azizzadenesheli, A. Anandkumar, and S. M. Benson, "Real-time high-resolution CO_2 geological storage prediction using nested Fourier neural operators," Energy Environ. Sci. **16**(4), 1732–1741 (2023).

[5] X. Shao, Z. Liu, S. Zhang, Z. Zhao, and C. Hu, "PIGNN-CFD: A physics-informed graph neural network for rapid predicting urban wind field defined on unstructured mesh," Build. Environ. **232**, 110056 (2023).

[6] W. Peng, S. Qin, S. Yang, J. Wang, X. Liu, and L. L. Wang, "Fourier neural operator for real-time simulation of 3D dynamic urban microclimate," Build. Environ. **248**, 111063 (2024).

[7] Z. Liu, S. Zhang, X. Shao, and Z. Wu, "Accurate and efficient urban wind prediction at city-scale with memory-scalable graph neural network," Sustain Cities Soc. **99**, 104935 (2023).

[8] Y. Zheng and X. Xu, "Predicting indoor 3D airflow distribution using artificial neural networks with two different architectures," Energy Build. **303**, 113841 (2024).

[9] N. Thuerey, K. Weißenow, L. Prantl, and X. Hu, "Deep learning methods for Reynolds-averaged Navier–Stokes simulations of airfoil flows," AIAA J. **58**(1), 25–36 (2020).

[10] K. Zuo, Z. Ye, W. Zhang, X. Yuan, and L. Zhu, "Fast aerodynamics prediction of laminar airfoils based on deep attention network," Phys. Fluids **35**(3), 037127 (2023).

[11] K. Fukami, K. Fukagata, and K. Taira, "Super-resolution reconstruction of turbulent flows with machine learning," J. Fluid Mech. **870**, 106–120 (2019).

[12] B. Hu, Z. Yin, A. Hamrani, A. Leon, and D. McDaniel, "Super-resolution-assisted rapid high-fidelity CFD modeling of data centers," Build. Environ. **247**, 111036 (2024).

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *18th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015), Munich, Germany, October 5–9*, Part III 18 (Springer, 2015), pp. 234–241.

[14] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris, "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data," J. Comput. Phys. **394**, 56–81 (2019).

[15] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "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 (2019).

[16] Z. Li *et al.*, "Fourier neural operator for parametric partial differential equations," arXiv:2010.08895 (2020).

[17] O. Hennigh *et al.*, "NVIDIA SimNet$^{\text{TM}}$: An AI-accelerated multi-physics simulation framework," in *International Conference on Computational Science* (Springer, 2021), pp. 447–461.

[18] N. Wandel, M. Weinmann, M. Neidlin, and R. Klein, "Spline-PINN: Approaching PDEs without data using fast, physics-informed hermite-spline CNNs," in *Proceedings of the AAAI Conference on Artificial Intelligence* (AAAI, 2022), pp. 8529–8538.

[19] A. Kashefi and T. Mukerji, "Physics-informed PointNet: A deep learning solver for steady-state incompressible flows and thermal fields on multiple sets of irregular geometries," J. Comput. Phys. **468**, 111510 (2022).

[20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2017), pp. 652–660.

[21] Z. Li *et al.*, "Physics-informed neural operator for learning partial differential equations," ACM/IMS J. Data Sci. **1**, 1–27 (2021).

[22] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations," J. Comput. Phys. **426**, 109951 (2021).

[23] R. Wang, D. Xia, Z. Cao, Y. Wen, R. Tan, and X. Zhou, "Toward data center digital twins via knowledge-based model calibration and reduction," ACM Trans. Model. Comput. Simul. **33**(4), 1–24 (2023).

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

[25] B. Steinfurth and J. Weiss, "Assimilating experimental data of a mean three-dimensional separated flow using physics-informed neural networks," Phys. Fluids **36**(1), 015131 (2024).

[26] H. Wang, Y. Liu, and S. Wang, "Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network," Phys. Fluids **34**(1), 017116 (2022).

[27] C. Wei and R. Ooka, "Indoor airflow field reconstruction using physics-informed neural network," Build. Environ. **242**, 110563 (2023).

[28] E.-Z. Rui, Z.-W. Chen, Y.-Q. Ni, L. Yuan, and G.-Z. Zeng, "Reconstruction of 3D flow field around a building model in wind tunnel: A novel physics-informed neural network framework adopting dynamic prioritization self-adaptive loss balance strategy," Eng. Appl. Comput. Fluid Mech. **17**(1), 2238849 (2023).

[29] C. Wu, M. Zhu, Q. Tan, Y. Kartha, and L. Lu, "A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks," Comput. Methods Appl. Mech. Eng. **403**, 115671 (2023).

[30] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," Nat. Rev. Phys. **3**(6), 422–440 (2021).

[31] V. Kag, K. Seshasayanan, and V. Gopinath, "Physics-informed data based neural networks for two-dimensional turbulence," Phys. Fluids **34**(5), 055130 (2022).

[32] G. Alfonsi, "Reynolds-averaged Navier–Stokes equations for turbulence modeling," Appl. Mech. Rev. **62**(4), 040802 (2009).

[33] D. Lucor, A. Agrawal, and A. Sergent, "Simple computational strategies for more effective physics-informed neural networks modeling of turbulent natural convection," J. Comput. Phys. **456**, 111022 (2022).

[34] X.-D. Bai and W. Zhang, "Machine learning for vortex induced vibration in turbulent flow," Comput. Fluids **235**, 105266 (2022).

[35] H. Eivazi, M. Tahani, P. Schlatter, and R. Vinuesa, "Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations," Phys. Fluids **34**(7), 075117 (2022).

[36] S. Hanrahan, M. Kozul, and R. D. Sandberg, "Studying turbulent flows with physics-informed neural networks and sparse data," Int. J. Heat Fluid Flow **104**, 109232 (2023).

[37] N. Rahaman *et al.*, "On the spectral bias of neural networks," in *International Conference on Machine Learning* (PMLR, 2019), pp. 5301–5310.

[38] M. Tancik *et al.*, "Fourier features let networks learn high frequency functions in low dimensional domains," in *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Curran Associates, Inc., 2020), Vol. 33, pp. 7537–7547.

[39] P. V. Nielsen, "Specification of a two-dimensional test case: (IEA)," (Institut for Bygningsteknik, Aalborg Universitet, 1990).

[40] W. Zuo and Q. Chen, "Fast and informative flow simulations in a building by using fast fluid dynamics model on graphics processing unit," Build. Environ. **45**(3), 747–757 (2010).

[41] J. B. Perot and J. Gadebusch, "A stress transport equation model for simulating turbulence at any mesh resolution," Theor. Comput. Fluid Dyn. **23**, 271–286 (2009).

[42] A. A. Heydari, C. A. Thompson, and A. Mehmood, "SoftAdapt: Techniques for adaptive loss weighting of neural networks with multi-part loss functions," arXiv:1912.12355 (2019).

[43] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International Conference on Machine Learning* (PMLR, 2018), pp. 794–803.

[44] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," J. Comput. Phys. **449**, 110768 (2022).