PHYSICAL REVIEW FLUIDS **9**, 044603 (2024)

Editors' Suggestion

Machine-learning-augmented domain decomposition method for near-wall turbulence modeling

Shiyu Lyu,[1,*] Jiaqing Kou,[2,†] and Nikolaus A. Adams[1,3]
[1]*Chair of Aerodynamics and Fluid Mechanics, School of Engineering and Design, Technical University of Munich, Boltzmannstrasse 15, 85748 Garching, Germany*
[2]*Institute of Aerodynamics, RWTH Aachen University, Wüllnerstrasse 5a, 52062 Aachen, Germany*
[3]*Munich Institute of Integrated Materials, Energy and Process Engineering, Technical University of Munich, Lichtenbergstrasse 4a, 85748 Garching, Germany*

(Received 31 October 2023; accepted 7 March 2024; published 5 April 2024)

To tackle the challenging near-wall turbulence modeling while preserving low computational cost, the near-wall nonoverlapping domain decomposition (NDD) method is proposed, incorporating the machine-learning technique. Using recently proposed implicit NDD (INDD), the solution can be calculated with a Robin-type (slip) wall boundary condition on a relatively coarse mesh and then corrected in the near-wall region at every iteration through an estimated turbulent viscosity profile obtained from a neural network. To maintain a reasonable complexity with acceptable accuracy, only the near-wall field properties, i.e., wall-normal distance, near-wall velocities, streamwise pressure gradients, and one near-wall scale parameter, are employed as the input features for the neural network. The benefit of incorporating machine learning is twofold. First, the near-wall turbulent viscosity is predicted more accurately than by the traditional algebraic functions used in the approximate approach. Second, similarly to the conventional NDD, the present simulations save one order of computational cost over the fully resolved one-block simulations. The accuracy and efficiency of the method are demonstrated on test cases with the k-ε model, including turbulent flows in a channel and an asymmetric diffuser at different Reynolds numbers. Results compare favorably with the one-block benchmark solutions and show a better agreement when compared with approximate NDD predictions. In the latter case, a variation of diffuser geometry is also considered to test the model performance on engineering design tasks with another turbulent model (i.e., Spalart-Allmaras), showing good generalization capability on different turbulence closures.

DOI: 10.1103/PhysRevFluids.9.044603

I. INTRODUCTION

Developing effective near-wall turbulence modeling methods remains a popular topic in modern fluid dynamics. Very fine meshes are required to fully resolve the near-wall flow, thus incurring large computational costs. The thickness of the turbulent boundary layer shrinks as the Reynolds number (Re) increases, leading to the excessive computational effort of handling wall layers [1]. Specifically, the computing cost for direct numerical simulation (DNS) scales as Re^3, and for large eddy simulation (LES) it is proportional to $\mathrm{Re}^{2.4}$ [2] or to the friction Reynolds number Re_τ^2 [3].

---

[*] shiyu.lyu@tum.de
[†] Corresponding author: j.kou@aia.rwth-aachen.de

The Reynolds-averaged Navier-Stokes (RANS) models are still widely deployed in the industrial community, due to their good compromise between accuracy and cost. RANS models that resolve the entire computational domain, including the near-wall layer, are called low-Reynolds number (LRN) models (e.g., the LRN k-ε model by Launder and Sharma [4] and by Chien [5], and the LRN k-ω by Wilcox [6]), where damping functions are commonly introduced into the turbulence equation to predict the variation of each function in the vicinity of the wall. Although the LRN solution is often considered as the benchmark result for its high accuracy, LRN corrections significantly increase the complexity of turbulence models. More importantly, greatly fine meshes are required to sufficiently resolve the near-wall sublayer.

One of the most common ways to save computational effort is by applying high-Reynolds number (HRN) models. The near-wall sublayer is deliberately not resolved with this type of model, but is described via wall functions. Specifically, the wall functions represent an approximate solution of the near-wall region and formulate Dirichlet-type boundary conditions at the cell nearest to the wall [7]. However, assuming the logarithmic relationship between the wall distance and wall-parallel velocity inevitably fails if the near-wall cell center is located outside the log-law sublayer. Thereafter, the scalable wall function [8] is established to improve the validity of the logarithmic law and to reduce mesh dependence. The analytical wall function [9] and the later numerical wall function [10] perform the corresponding analytical and numerical integration of the simplified boundary-layer equations using a fairly simple formula for the turbulent viscosity. This class of wall functions reduces the limitation of the cell size nearest to the wall and takes the buoyancy and pressure gradient into account, but it is difficult to further generalize due to stability issues [11]. Similarly, the compound wall treatment [12] uses a one-dimensional boundary-layer equation with the inclusion of source terms. Since the source terms are assumed to be unchanged across the sublayer, this approach lacks validation and accuracy, especially in complex flows.

Another efficient approach to solve the challenging problem is the near-wall nonoverlapping domain decomposition (NDD) method [13]. With the NDD approach, the computational domain is split into two subdomains: inner domain and outer domain. Both domains are related by a common boundary patch named the interface. The calculation of the near-wall domain with high gradients can be excluded by exactly transferring the wall boundary condition to an interface boundary [14]. The obtained interface boundary conditions (IBCs) are proved to be mesh-independent and, more importantly, do not require any free parameters tuned for different cases, which can be generalized to other turbulence models without significant modification [15]. Compared with the LRN models, the NDD method drastically reduces the computational cost by one order of magnitude with required resolutions [16,17], even for complex simulations [18,19]. The computational time reduces as the distance from the interface to the wall increases, which is mainly because of the utilization of simplified governing equations, i.e., the thin boundary-layer equations (TBLEs), in the near-wall region. Another reason is that the variation and its derivative are taken into the Robin-type boundary conditions at the same iteration, leading to faster convergence. Conversely, when the interface is sufficiently close to the wall, the interface boundary conditions approach the physical wall boundary conditions and tend to the LRN solution.

However, mesh generation for cases with complex geometries and optimal problems can be time-consuming, due to its relative algorithmic complexity and the necessity of splitting the entire domain explicitly. The implicit NDD is further proposed in [20] to ameliorate this issue and has been successfully applied to the LRN k-ε turbulence model in the channel flow and 2D asymmetrical impinging jet cases. Within the framework of implicit NDD, the IBC for the tangential velocity is transferred back to the wall with frozen coefficients, and subsequently a Robin-type boundary condition is introduced at the wall to replace the original wall boundary condition. The obtained Robin boundary condition for velocity is similar in form to the dynamic slip boundary condition for LES in [21,22], while following considerably different principles and computing procedures.

For the NDD method, a particular challenge is related to the turbulent viscosity. In the approximate NDD method, the turbulent viscosity across the inner domain is approximated in advance by semiempirical functions, such as the piecewise linear profile [23], the exponential profile with a damping function [24–26] and further improved with dynamic coefficients [27,28], and the nonlinear profile [29], which largely restricts the range of applications. This issue becomes more pronounced when solving new problems because of the uncertainty of the obtained results with the selected profile [30]. Alternatively, in the exact NDD approach, a more accurate profile of turbulent viscosity can be computed directly from the inner turbulent model, while costing extra computation time. It is worth noting that, generally, the NDD methods fall into two categories: one is based on the decomposition method, i.e., explicit or implicit; the second one is according to whether the governing equations for the inner region are simplified or not, i.e., approximate or exact, respectively [18,31].

With the recent progress in artificial intelligence, machine-learning (ML) approaches have been applied to tackle various turbulence problems [32,33], for example flow simulation [34], shock capturing [35,36], turbulence reconstruction [37,38], etc. Particularly, ML has been utilized to develop data-driven turbulence models. Promising activities include using neural networks to improve the Reynolds stress predictions from high-fidelity DNS data [39,40] and introducing the field inversion approach and ML to infer better functional forms in RANS models [41]. In addition, gene expression programming (GEP) has been applied to enhance explicit algebraic Reynolds stress models, showing good performance in practical RANS calculations [42,43]. Different ML techniques are also proposed for LES closures [44,45]. Given the practical importance, the wall models for LES are trained using neural networks to achieve an accurate wall shear stress [46,47]. More recently, an advanced wall model was proposed for LES by reinforcement learning to properly predict key quantities with substantially lower cost in extreme Re flows [48] and by reconstructing the near-wall field as a collection of simple building-block flows to reproduce the correct wall stress [49]. Data-driven approaches for developing machine-learning augmented turbulence models in RANS have been proven to give reliable mapping between the flow features and eddy viscosity [50,51] and improved prediction of separated flows [52]. Thanks to the rapid growth of ML techniques, the eddy viscosity profile can apparently be learned from a large set of data in the near-wall region, instead of calculating from the inner governing equations as in the conventional NDD method.

The combined technique can properly overcome the limitation induced by the prescribed turbulent viscosity while securing a great reduction in computation time. In particular, the wall boundary conditions are formally identical to the Robin-type boundary conditions in implicit NDD. With machine learning, the calculation of the turbulence model can be entirely avoided in the subgrid region of implicit NDD. This is because the turbulent viscosity is predicted by a pretrained machine-learning model straightforwardly, to form the Robin boundary conditions at the wall. In this case, lower computational cost than both the LRN solution and the original implicit NDD method is expected.

The rest of the paper is organized as follows: In Sec. II, the original NDD method and the slip wall boundary condition are described and briefly derived. Next, in Sec. III, the machine-learning technique used in this work is described. The accuracy and efficiency of the new approach are demonstrated on a one-dimensional channel flow and two-dimensional diffuser flow in Sec. IV, where the numerical results of the test cases are discussed in detail. This is followed by the conclusion in Sec. V.

II. NEAR-WALL DOMAIN DECOMPOSITION

A. Derivation of approximate NDD

In NDD, the entire computational domain in the wall-normal direction $\Omega := [0, y_e]$ is separated by the interface at $y = y^*$, leading to two types of regions: the inner and outer regions. Specifically, the widely applied LRN k-ε model of Launder-Sharma [4] is chosen to close the governing equations over the outer region $\Omega_o := [y^*, y_e]$, where the full equations for turbulent kinetic energy k and its dissipation rate ε are given in Appendix A. The TBLE utilized in the inner region $\Omega_i := [0, y^*]$

can be written as follows:

$$\begin{aligned}
\frac{\partial}{\partial y}\left[(\nu+\nu_t)\frac{\partial u}{\partial y}\right] &= \frac{1}{\rho}\frac{\partial p}{\partial x},\\
\frac{\partial}{\partial y}\left[\left(\nu+\frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial y}\right] &= \tilde{\varepsilon}+\varepsilon_0-\nu_t\left(\frac{\partial u}{\partial y}\right)^2,\\
\frac{\partial}{\partial y}\left[\left(\nu+\frac{\nu_t}{\sigma_\varepsilon}\right)\frac{\partial \tilde{\varepsilon}}{\partial y}\right] &= C_{\varepsilon 2}f_{\varepsilon 2}\frac{\tilde{\varepsilon}^2}{k}-C_{\varepsilon 1}f_{\varepsilon 1}\frac{\tilde{\varepsilon}}{k}\nu_t\left(\frac{\partial u}{\partial y}\right)^2-E.
\end{aligned} \tag{1}$$

Here, u, k, and $\tilde{\varepsilon}$ are the wall-parallel velocity component, turbulent kinetic energy, and its dissipation, respectively; x is the coordinate along the wall; y is the wall-normal coordinate; p is the pressure; ν denotes the dynamic viscosity coefficient; and ν_t denotes the eddy viscosity coefficient. More definitions of model coefficients are given in Appendix A.

The reduced form of governing Eqs. (1) for a variable Φ can be written compactly:

$$\frac{\partial}{\partial y}\left(\Gamma_\Phi\frac{\partial \Phi}{\partial y}\right)=R_\Phi. \tag{2}$$

Here, the variable Φ can represent the velocity u, turbulent kinetic energy k, and its dissipation $\tilde{\varepsilon}$. Meanwhile, the corresponding source term R_Φ on the right-hand side is specified according to the selected equations. Γ_Φ represents the effective viscosity, $\Gamma_\Phi=\nu+\nu_t/\sigma_\Phi$, where ν and ν_t are the laminar viscosity and turbulent viscosity, respectively. The σ_Φ is the turbulent Prandtl number for Φ.

After integrating Eq. (2) from y to y^*, we obtain

$$\frac{\partial \Phi}{\partial y}=\frac{\Gamma_\Phi(y^*)}{\Gamma_\Phi(y)}\left.\frac{\partial \Phi}{\partial y}\right|_{y^*}-\frac{\int_y^{y^*}R_\Phi dy'}{\Gamma_\Phi(y)}. \tag{3}$$

The second integration is between the wall ($y=0$) and the interface ($y=y^*$), giving

$$\Phi(y^*)=\left.\frac{\partial \Phi}{\partial y}\right|_{y^*}\int_0^{y^*}\frac{\Gamma_\Phi(y^*)}{\Gamma_\Phi(y)}dy-\int_0^{y^*}\frac{\int_y^{y^*}R_\Phi dy'}{\Gamma_\Phi(y)}dy+\Phi(0). \tag{4}$$

Further, we express Eq. (4) compactly via introducing f_1, f_2 as

$$\Phi(y^*)=f_1\left.\frac{\partial \Phi}{\partial y}\right|_{y^*}+f_2, \tag{5}$$

where

$$\begin{aligned}
f_1 &= \int_0^{y^*}\frac{\Gamma_\Phi(y^*)}{\Gamma_\Phi(y)}dy,\\
f_2 &= -\int_0^{y^*}\frac{\int_y^{y^*}R_\Phi dy'}{\Gamma_\Phi(y)}dy+\Phi(0).
\end{aligned}$$

Therefore, recalling the right-hand side of the inner governing equations (1), the source terms for u, k, and $\tilde{\varepsilon}$ are expressed as

$$\begin{aligned} R_u &= \frac{1}{\rho}\frac{\partial p}{\partial x}, \\ R_k &= \varepsilon - \nu_t\left(\frac{\partial u}{\partial y}\right)^2, \\ R_{\tilde{\varepsilon}} &= C_{\varepsilon 2} f_{\varepsilon 2}\frac{\tilde{\varepsilon}^2}{k} - C_{\varepsilon 1} f_{\varepsilon 1}\frac{\tilde{\varepsilon}}{k}\nu_t\left(\frac{\partial u}{\partial y}\right)^2 - E. \end{aligned} \tag{6}$$

The interface boundary conditions have been formulated in a Robin-type (5), which is applicable to all key variables, i.e., u, k, and $\tilde{\varepsilon}$. The corresponding source terms are presented in Eq. (6), which is essential for calculating the coefficient f_2. The right-hand side R_u is asserted to be unchanged along the wall-normal direction within the inner region and evaluated at the interface ($y = y^*$) every iteration. Unlike in the outer region, the pressure-velocity decoupling or the Poisson equation is not required in the inner region. Note that k and $\tilde{\varepsilon}$ are coupled in terms of the governing equations. Thus, the solution in the inner region is required to compute every iteration, so as to update the interface boundary conditions.

It should be noted that an interface condition of Robin-type for the normal component of velocity is deployed, stemming from the no-flux boundary condition and continuity equation using the Taylor expansion [18]:

$$v^* = \frac{y^*}{3}\frac{\partial v_c}{\partial y}, \tag{7}$$

where v^* and v_c are the values of the normal velocity at the interface boundary and interface-adjacent cell center in the outer region, respectively.

B. Near-wall turbulent viscosity profile

To close the formulation of approximate NDD, the only parameter required to be imposed is an approximate profile of ν_t for the near-wall regions. The popular exponential profile [29] that takes into account the streamwise pressure gradient is used here:

$$\nu_t(\xi) = \nu\kappa\xi^+[\alpha + \xi^+(1-\alpha)^{3/2}]^\beta\left[1 - \exp\left(\frac{-\xi^+}{1 + A\alpha^3}\right)\right]^2, \tag{8}$$

where $\kappa = 0.41$, $A = 17$, $\beta = 0.78$, $\alpha = u_\tau^2/u_{\tau p}^2$, $\xi^+ = yu_{\tau p}/\nu$, $u_{\tau p} = \sqrt{u_\tau^2 + u_p^2}$, $u_\tau = \sqrt{|\tau_w|/\rho}$, $u_p = |\nu\partial p/\partial x/\rho|$.

The turbulent viscosity profile (8) obviously depends on the surface friction. The parameter τ_w can be updated every iteration via (3) for $\Phi = u$ with setting $y = 0$. This explicit NDD with an approximate turbulent viscosity profile for the inner regions is termed ENDD-A for brevity hereafter in this paper.

C. Derivation of slip boundary condition for implicit NDD

With implicit NDD [20], the Robin-type interface boundary condition for velocity is transferred back to the wall. First, the solution $\widetilde{U}$ with the slip boundary condition in the inner region can be interpreted using the Taylor series,

$$\widetilde{U}(y) \approx \widetilde{U}(0) + y\left.\frac{\partial\widetilde{U}}{\partial y}\right|_0 + \frac{y^2}{2}\left.\frac{\partial^2\widetilde{U}}{\partial y^2}\right|_0. \tag{9}$$

Evaluation at the interface ($y = y^*$) and replacing the second derivative according to the simplified momentum equation, we obtain

$$\widetilde{U}(y^*) = \widetilde{U}(0) + y^* \left.\frac{\partial \widetilde{U}}{\partial y}\right|_0 + \frac{y^{*2} R_u^*}{2\Gamma_u^*}, \tag{10}$$

where $R_u^* = R_u(y^*)$ and $\Gamma_u^* = \Gamma_u(y^*)$.

Having differentiated (10), we arrive at

$$\left.\frac{\partial \widetilde{U}}{\partial y}\right|_{y^*} = \left.\frac{\partial \widetilde{U}}{\partial y}\right|_0 + \frac{y^* R_u^*}{\Gamma_u^*}. \tag{11}$$

The new solution must satisfy the interface boundary condition (5):

$$\widetilde{U}(y^*) = f_1 \left.\frac{\partial \widetilde{U}}{\partial y}\right|_{y^*} + f_2. \tag{12}$$

Finally, the Robin-type boundary condition at the wall for variable $\widetilde{U}$ can be formulated:

$$\widetilde{U}(0) = f_{w1} \left.\frac{\partial \widetilde{U}}{\partial y}\right|_0 + f_{w2}, \tag{13}$$

where $f_{w1} = f_1 - y^*$, $f_{w2} = f_2 + \frac{y^* R_u^*}{\Gamma_u^*}(f_1 - \frac{y^*}{2})$.

Evidently, the slip wall boundary condition (13) is free of *a priori* specified coefficients, except for the turbulent viscosity. In contrast to the approximate NDD approach, in the implicit NDD we determine the turbulent viscosity from the trained neural network. Additionally, the implicit NDD is further performed on the Spalart-Allmaras model [53] (refer to Appendix B for a detailed formulation) in the diffuser turbulent flow to capture the flow separation.

III. CONSTRUCTION OF DATA-DRIVEN NDD METHOD

A. Neural network architecture

Within the framework of the implicit NDD, the neural network acts as a supplement to predict the turbulent viscosity in the inner region, rather than obtaining it by explicit expression or solving partial differential equations. Such a model is termed INDD-D (implicit NDD with data-driven prediction of turbulent viscosity) for brevity hereafter in this paper. A schematic diagram for the entire algorithm is depicted in Fig. 1. It can be seen that a feed-forward neural network (FNN) is deployed for predicting the near-wall eddy viscosity of every iteration. The neural network

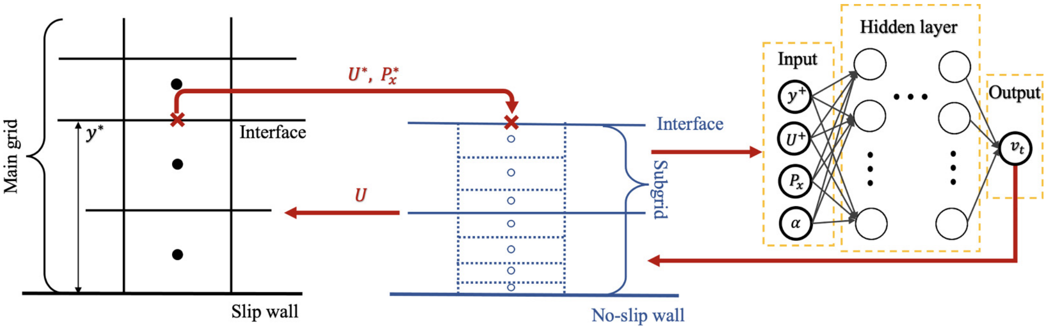


FIG. 1. Sketch of the data-driven implicit NDD.

architecture is built based on the mutual connection among brain neurons, which generally contains one input layer, several hidden layers, and one output layer. Assuming a neural network has L layers, where each layer has K_n $(n = 1, 2, \ldots, L)$ neurons, the output of the mth node in the nth layer is constructed by taking a weighted sum of the inputs X_m^{n-1} and then transforming them through nonlinear activation functions as follows:

$$Y_m^n = a^n\left(\sum_{i=1}^{K_n} \omega_m^n(i)X_m^{n-1}(i) + b_m^n\right). \tag{14}$$

Here, a^n and ω_m^n are the activation function and weights associated with the nth hidden layer, respectively. b_m^n denotes one bias term at the mth neuron in the nth layer. The activation function utilized in this study is the rectifier linear unit (ReLU), which has been widely used and defined as $a(x) = \max(0, x)$. The loss function is defined as the mean-square error (MSE) between forecast values Y_{pr} from FNN output and actual values Y_{tr} from the training set:

$$\mathcal{L}_{\text{NN}} = \frac{1}{N_s}\sum_{i=1}^{N_s}[Y_{\text{pr}}(i) - Y_{\text{tr}}(i)]^2, \tag{15}$$

where N_s is the number of training samples. The optimal hyperparameters, i.e., the number of hidden layers and neurons, and the value of the learning rate, are chosen to reach the lowest error on the validation set. Considering the balance of accuracy and generalization, one hidden layer should be adequate and suitable for the current prediction [50]. Additionally, the results indicate that 40 neurons are enough to reach the optimal solution. In the training process, the node weights were iteratively tuned using the Adam optimizer [54] to reach the lowest mean-squared error between the predicted eddy viscosity and the true counterparts.

B. Training data

For each test case, the one-block LRN RANS simulations are performed first at various Reynolds numbers to generate the training data set. Generally, 80% of training data are used for the training data set, and the remaining 20% are used as the validation data set. The flow boundary conditions of training and validation cases are identical to those in the test cases. Specifically in this work, only the flow data in the near-wall regions with wall-normal distance smaller than the chosen interface position y^* are further processed for training the data-driven model.

For each point in the fine subgrid, the wall-normal distance η, the velocity components U in the wall-tangential direction, the streamwise pressure gradient $\partial p/\partial x$, and the near-wall scale parameter $\alpha = u_\tau^2/u_{\tau p}^2$ are employed as the input features. Physically, the nondimensional parameter α qualifies the predominant effect of shear stress and streamwise pressure gradient near the wall [55], where $u_{\tau p} = \sqrt{u_\tau^2 + u_p^2}$, $u_\tau = \sqrt{(\tau_w/\rho)}$ with an additional velocity based on streamwise pressure gradient $u_p = |\nu\partial p/\partial x/\rho|^{1/3}$. Obviously, the eddy viscosity ν_t is required to be the unique output feature. Following the near-wall treatments of NDD, the streamwise pressure gradient is assumed to be constant at the wall-normal direction and evaluated instantly along the interface position y^* from the outer region. For the wall-tangential velocity component, we can easily solve the momentum equation in (1), or obtain straightforwardly

$$U(y) = \left.\frac{\partial U}{\partial y}\right|_{y^*}\int_0^y \frac{\Gamma_u(y^*)}{\Gamma_u(y)}dy - \int_0^y \frac{\int_y^{y^*} R_u dy'}{\Gamma_u(y)}dy + U(0). \tag{16}$$

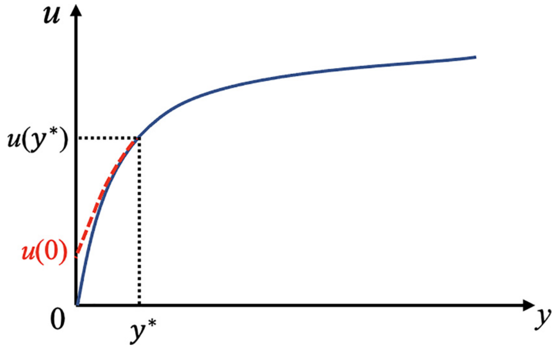


FIG. 2. Sketch of the solution obtained in two stages of implicit NDD.

To generalize the capacity of the trained profile and take into account the classical dimensionless parameters [29,55], the wall-normal distance is normalized using the near-wall scale and written as y^+, where $y^+ = \eta u_{\tau p}/\nu$. The pressure gradient along the wall is multiplied by η/h as utilized in [47]. The output label ν_t is treated as ν_t/ν to improve training efficiency. Moreover, all the input and output features are normalized using the min-max scaling,

$$X_{\text{scale}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}. \tag{17}$$

The training results of the mapping function between input and output features, including the selection of different features, are shown in Appendix C for different turbulence models.

C. Numerical implementation of implicit NDD

The Robin boundary condition (13) is imposed for the tangential velocity at the wall, while the no-flux wall boundary condition remains for the normal velocity component. At the first stage, the solution of the original full governing equations in the entire region can be obtained on a relatively coarse mesh with the slip boundary condition at the wall. This can be achieved because the slip boundary condition is imposed at the wall, resulting in a smaller velocity gradient near the wall. At the next stage, the TBLE model on a very fine subgrid, tailored to the underresolved region of the first stage, is designed to recalculate immediately with the original wall boundary condition and instantaneous interface boundary condition. Then, the inner solution can be corrected accordingly. The final composite solution from inner TBLE and outer RANS is smooth up to the first derivative [20]. These two stages are schematically shown in Fig. 2, where the red dotted line corresponds to the solution at the preliminary stage with the slip wall boundary, and the blue solid line represents the composite solution.

The implicit NDD approach has been implemented within the framework of OpenFOAM [56], advanced open-source software for computational fluid dynamics based on the finite volume method. Spatial discretization uses the second-order central difference scheme for solving incompressible flows on collocated grids. The velocity-pressure coupling is handled by the SIMPLEC algorithm [57,58], with a variant of the Rhie-Chow interpolation to avoid checkerboard oscillations. The sequentially algebraic equations have been solved using the Gauss-Seidel iterative method. The Poisson equation for pressure was solved using a geometric-algebraic multigrid (GAMG) solver. With the help of the popular machine-learning library *PyTorch* [59] for approximating the cell-centered fields, i.e., turbulent viscosity field, the original and simplified governing equations are separately solved within the framework of multiregion functionality in OpenFOAM.

The proposed implicit NDD algorithm with machine learning is summarized in Algorithm 1.

ALGORITHM 1. The implicit NDD algorithm with neural network.

```
Initialize the flow fields for both the main (coarse) grid and sub-grid, see Fig. 1;
for each time step do
    1: Compute the coefficients $f_{w1}$ and $f_{w2}$ based on the sub-grid solution to impose the slip boundary condition at the wall;
    2: Solve the full governing equations on the main grid with the updated slip wall boundary condition;
    3: Recalculate the solution in the inner region on the sub-grid region with the interface boundary value obtained from the main grid;
    4: Obtain the turbulent viscosity from the trained neural network model based on the corrected flow features;
end
```

IV. TEST CASES

Next, the proposed algorithm is assessed on the one-dimensional channel flow and two-dimensional diffuser flow. In each simulation, mesh independence is first achieved on a grid for the benchmark one-block solution to verify the numerical results. The one-block simulation represents the solution of the governing equations in the entire single domain without decomposition. Meanwhile, the first layer height of the grid in boundary layer $y^+ < 1$ is satisfied for the LRN turbulent model. In the outer region, the same mesh strategy is retained for both explicit NDD (ENDD-A) and implicit NDD (INDD-D). The same grid distribution is deployed for the inner region of the ENDD-A. In contrast, a very coarse mesh, along with slip boundary conditions, is used in the inner region of the INDD at the first stage. Then, a local fine mesh is utilized to recalculate the inner solution [20].

A. Plane channel flow

The fully developed channel flow with various friction Reynolds numbers Re_τ is considered as the first test case. Here, Re_τ is defined in terms of the friction velocity u_τ and the half of the channel height h: $\mathrm{Re}_\tau = u_\tau h/\nu$. The periodic boundary conditions are deployed at the inlet and outlet boundaries to exactly map the variations at the outlet back to the inlet, eventually reaching a fully developed flow. The pressure gradient along the streamwise direction is fixed as a constant driving force.

The fully developed channel flows with the friction Reynolds number $\mathrm{Re}_\tau = 395$, 1000, 3950, and 8000 are utilized for the neural network training. Then, the simulations of $\mathrm{Re}_\tau = 590$, 1995, 5186, and 9000 are tested and compared with the corresponding one-block LRN data in Fig. 3, where $\nu_t^+ = \nu_t/(u_\tau)^2$ and $u^+ = u/u_\tau$. Note that the last test case is the extrapolation state. The interface is located at $y^{*+} = 100$ for all test cases. The predicted velocities at different Re_τ are shifted along the vertical axis for clarity. The lowest curves are the results of $\mathrm{Re}_\tau = 590$, while the highest curves are for the results of $\mathrm{Re}_\tau = 9000$. The prediction of turbulent viscosity ν_t agrees well with the one-block solutions in the inner region. Subsequently, the velocity profiles in subfigure (b) are observed to have an overall good agreement with the benchmark solutions. The discrepancy between INDD-D and one-block solutions does not vary too much for different Re_τ for two reasons:

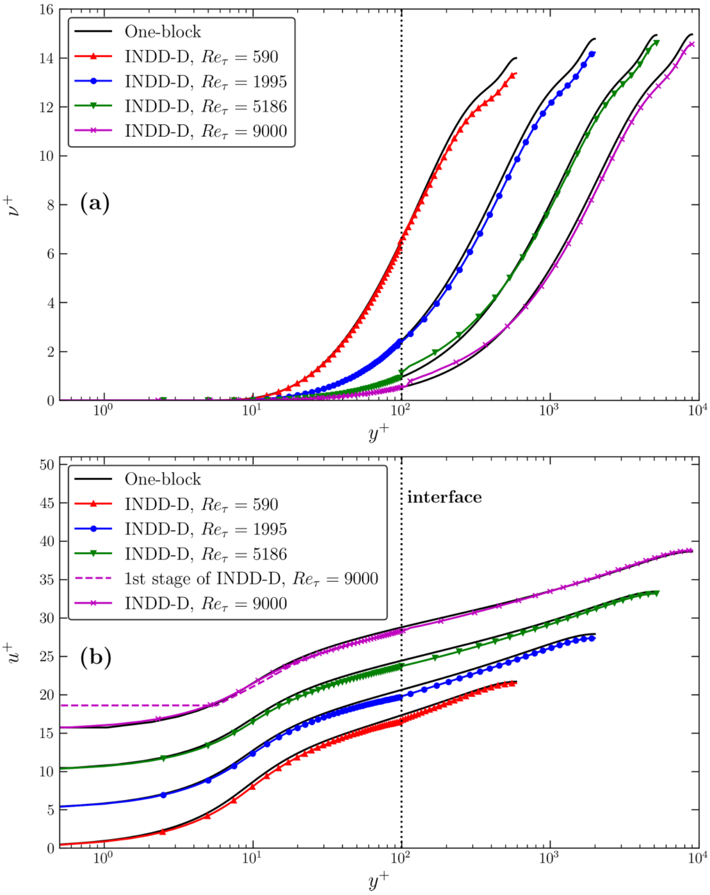


FIG. 3. Channel flow, interface position $y^{*+} = 100$. Comparison of nondimensional turbulent viscosity ν_t^+ (a) and streamwise velocity u^+ (b) between the results from implicit NDD ($\mathrm{Re}_\tau = 590$ in the red line, 1995 in the blue line, 5180 in the green line, and 9000 in the magenta line) and the one-block LRN solution.

(i) the strong physics relations between ν_t and mean flow features within the near-wall region; (ii) the truncated TBLE model does not introduce any essential error for one-dimensional channel flow.

B. Two-dimensional diffuser flow with the k-ε model

The geometry of the 2D asymmetric diffuser, according to the layout in [60], is portrayed in Fig. 4. The inlet is located at $x/h = -10$ and the outlet at $x/h = 75$. The corners of the diffuser situated at $x/h = 0$ and 21 are smoothed with the radius of curvature equal to $9.7H$. The expansion

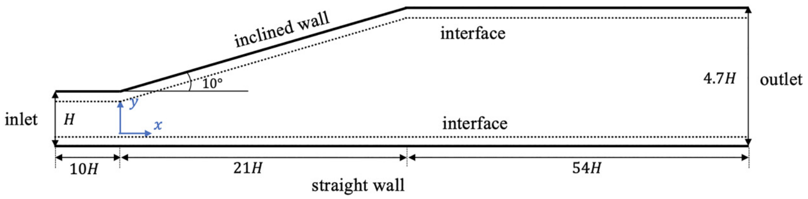


FIG. 4. Geometry of asymmetric diffuser.

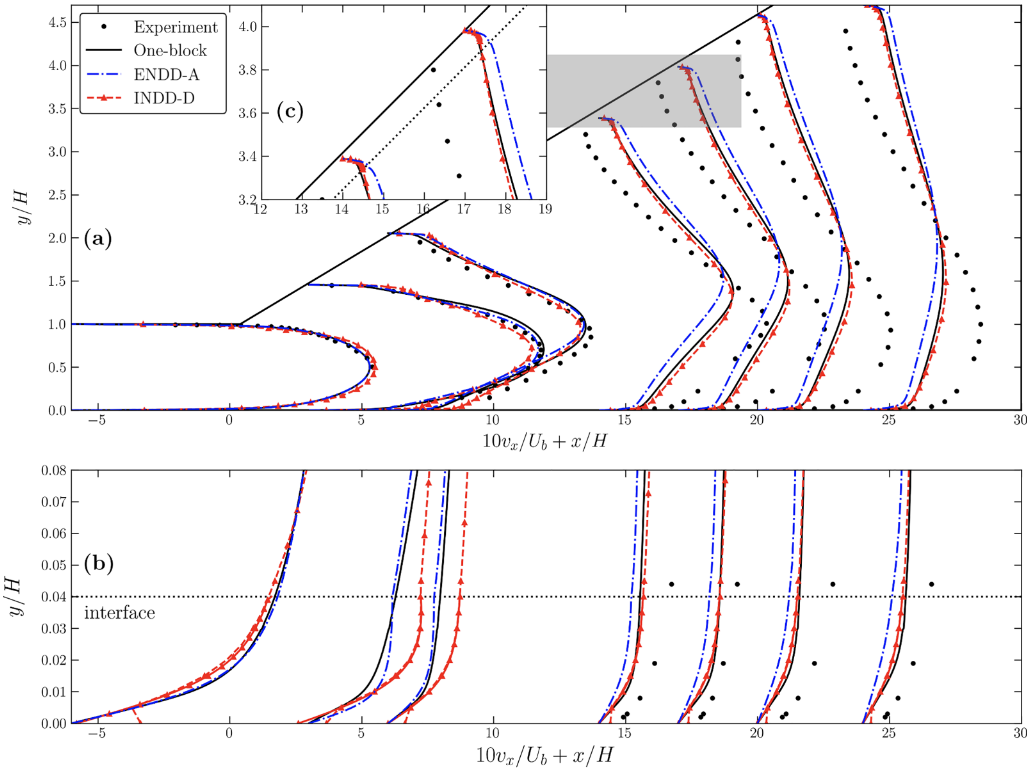


FIG. 5. Diffuser flow, Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$. (a) Comparison of mean streamwise velocities v_x between the results from experiment [60], one-block solution, explicit (approximate), and implicit NDD. (b) Enlargement of the region from the straight wall to $y/H = 0.06$. (c) Enlargement of the shaded region in (a).

ratio of the diffuser is 4.7, and its opening angle θ is 10°. The inlet boundary condition is imposed as the fully developed channel flow corresponding to Reynolds number Re $= U_b H/\nu$, where U_b is the bulk velocity and H is the channel height.

The one-block solutions of diffuser flow with the LRN k-ε model at Reynolds numbers Re $= 2 \times 10^4$ and 4×10^4 are utilized for the neural network training. After training, the neural network model for eddy viscosity is further applied to diffuser flows with the same geometry at Re $= 1.8 \times 10^4$ and 3.6×10^4.

As plotted in Fig. 5, the predictions of the streamwise velocity component v_x with the explicit and implicit NDD are compared with the one-block LRN results and experimental data [60] at $x/H = -5.87$, 2.59, 5.98, 13.56, 16.93, 20.32, and 23.71. The near-wall regions close to straight and inclined walls are enlarged in Figs. 5(b) and 5(c), respectively. The interface boundary for NDD methods is located at $y^*/H = 0.04$, corresponding to $y^{*+} = 40$. The velocity profiles obtained with the INDD-D are in overall good agreement with the benchmark LRN solutions. This means that using this TBLE model in the near-wall region of the diffuser flow is satisfied with good accuracy. A slight difference is near the interface area at $x/H = 2.59$ and 5.98 of the straight wall. This happens because the current NN model, which directly takes the streamwise pressure gradient as an input feature, is more sensitive to the larger pressure gradient in this region. Conversely, a relatively larger discrepancy exists between the ENDD-A predictions and the benchmark ones, which is mainly driven by the inaccurate turbulent viscosity approximation.

The velocity component along the y-axis v_y is shown in Fig. 6, where the sampling positions are in conformity with v_x. Similarly to the streamwise velocity, the velocity v_y obtained from

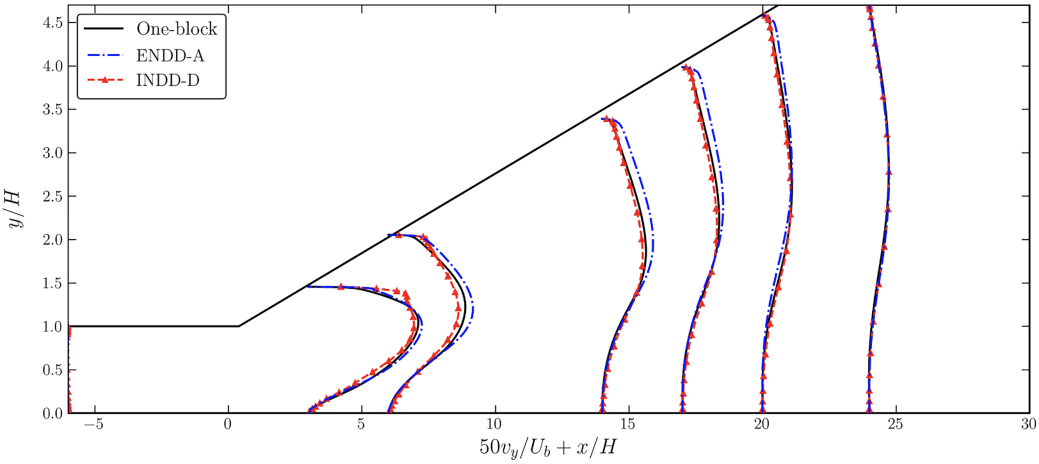


FIG. 6. Diffuser flow, Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$. Comparison of spanwise velocities v_y between the results from the one-block solution, explicit and implicit NDD.

the implicit NDD has a noticeably better match with the one-block curves than the ENDD-A ones.

Two dimensionless parameters (the pressure coefficient C_p and the skin friction coefficient C_f) are used to quantify the near-wall results between ENDD-A and INDD-D. The corresponding definitions are

$$\begin{aligned} C_p &= \frac{p - p_{\text{ref}}}{\frac{1}{2}\rho U_b^2}, \\ C_f &= \frac{\tau_w}{\frac{1}{2}\rho U_b^2}, \end{aligned} \tag{18}$$

where p_{ref} is the pressure at the midpoint of the inlet channel, and τ_w represents the wall shear stress.

As can be seen from Figs. 7(a) and 7(b), the skin friction profiles with implicit NDD for both straight and inclined walls are well improved compared to the predictions obtained with ENDD-A. The reason for the improvement is that imposing the turbulent viscosity through a pretrained neural network model is superior to the traditional exponential estimations. According to the experimental data, there is a secondary peak of C_f along the straight wall in subfigure (a) and a recirculation region along the inclined wall in subfigure (b). However, due to the limitation that the original LRN k-ε model is generally deficient in capturing the flow recirculation in the near-wall region, as also noted in [61], the current INDD-D and ENDD-A fail to predict any of them. For this reason, the current implicit NDD method is further extended to the Spalart-Allmaras model in the next section.

As for the C_p curves in Figs. 7(c) and 7(d), the value of C_p shows less discrepancy among the two NDD solutions and LRN solution, in contrast to C_f. This is because the pressure is computed only in the outer region, and the pressure gradient does not vary considerably near the wall, making the form of C_p analogous between NDD and LRN solutions. The figures show that the pressure gradient reduces as x increases, leading to another collapse of the wall shear stress downstream of the secondary peak point around $x/H = 22$ in subfigure (b).

Figures 8(a) and 8(b) demonstrate the prediction of velocity component v_x and v_y at Re $= 3.6 \times 10^4$, respectively. With larger y^{*+}, the results with ENDD-A deviate noticeably from the LRN one-block solutions, while the distributions computed from implicit NDD are in overall better agreement with reference LRN data than the counterparts from the ENDD-A. The reason for this improvement is that the superior prediction of wall shear stress and turbulent viscosity provides more accurate mean velocity profiles.

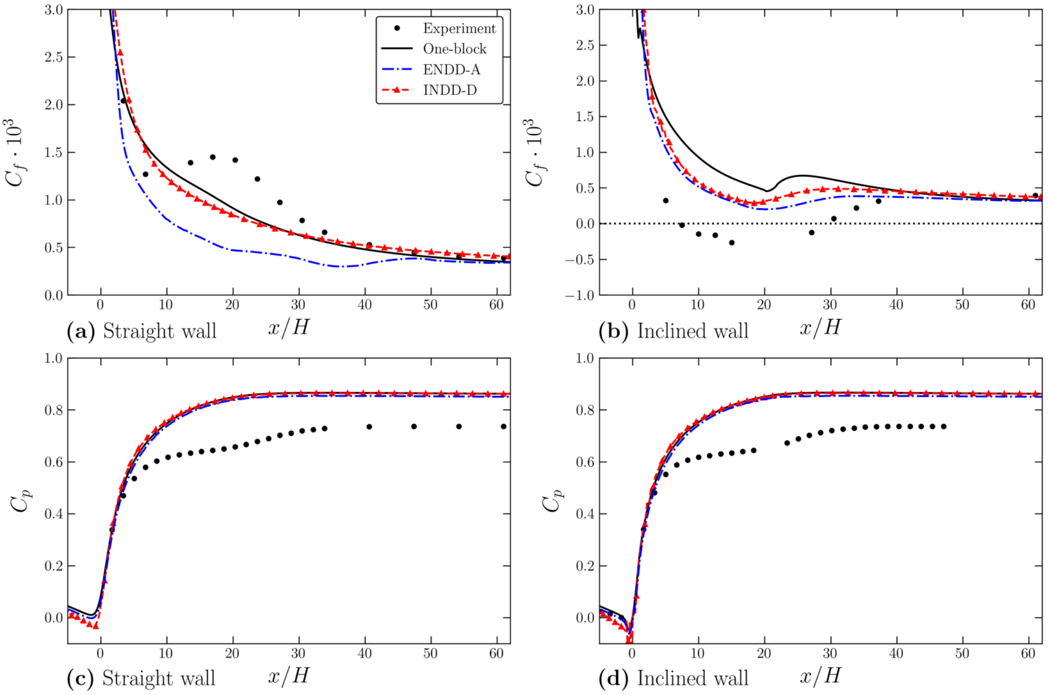


FIG. 7. Diffuser flow, Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$. Comparison of C_f and C_p between the results from experiment [60], one-block solution, explicit and implicit NDD. The subfigures (b), (c), and (d) share the same legend as (a).

Note that the data-driven INDD has been implemented straightforwardly to the LRN k-ε model by solely solving the momentum equation in the inner region. However, in the original INDD [20], the simplified turbulent equations corresponding to k and ε are additionally implemented inside numerical code and concurrently solved along with the momentum equation. That is to say, the implementation of data-driven INDD is independent of the specific turbulent model, as long as a pretrained viscosity profile based on an accurate data set is given. Thus, the data-driven INDD is relatively easier to implement and extend to different turbulent models. The comparisons of accuracy and efficiency between the original INDD [20] and the data-driven INDD are briefly demonstrated in Appendix D.

C. Two-dimensional diffuser flows with Spalart-Allmaras model

To demonstrate the generalization and capability of the proposed framework, the aforementioned diffuser flow with the Spalart-Allmaras (written as SA for brevity hereafter) model is further investigated, corresponding to various Reynolds numbers and opening angles. It is highly desirable to efficiently predict the separation of diffuser flows under different conditions [62]. The variation in geometry (as shown in Fig. 9) keeps the ratio of the width between the outlet and inlet the same. Three training cases of diffuser flows were sampled through the Latin hypercube sampling (LHS) method with the Reynolds number (Re) from 2×10^4 to 8×10^4 and the angle of the inclined wall (θ) from 8° to 14°. The predicted cases were chosen as the circumcenter of the three training sets and three points around the circumcircle; see Fig. 10. The predicted examples include both interpolation and extrapolation, by which the generalization of the proposed model is demonstrated.

The skin friction and pressure coefficients on the inclined wall with the SA model for different Re and θ are shown in Fig. 11. For C_p, all of the cases are identical with the one-block solutions. The

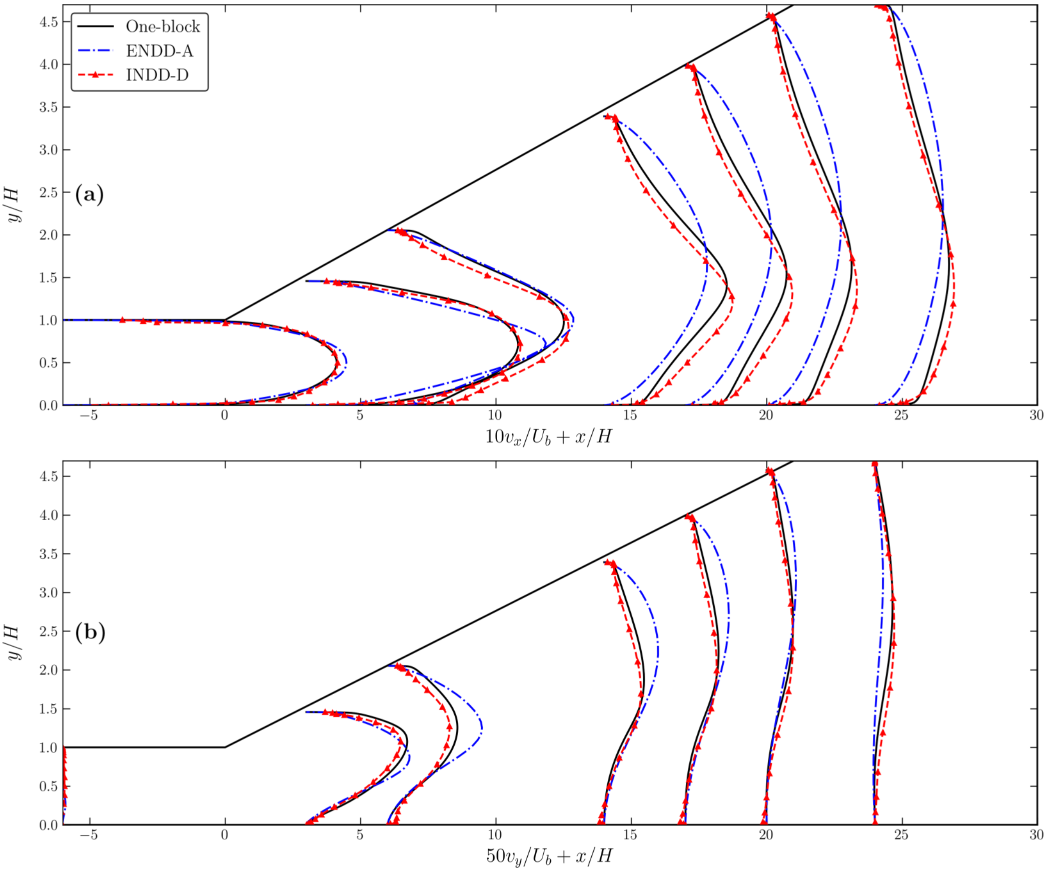


FIG. 8. Diffuser flow, $\mathrm{Re} = 3.6 \times 10^4$, interface position $y^{*+} = 60$. (a) Comparison of mean stream-wise velocities v_x between the results from one-block solution, explicit (approximate), and implicit NDD. (b) Comparison of velocity component v_y between the results from one-block solution, explicit and implicit NDD.

pressure gradient does not change considerably in the vicinity of the solid wall. The one equation SA model provides improved performance relative to the LRN k-ε models for the diffuser flows with adverse pressure gradients. Notably, data-driven INDD successfully predicts flow separation, albeit over a smaller region than the benchmark solution. As expected, the flow separation grows from mild to vast, as the opening angle increases. However, a relatively larger discrepancy exists between the predicted C_f and the benchmark ones, especially for the largest opening angle $\theta = 15^\circ$. The reason for this is twofold: one is because of the simplifications in TBLE, and the other is due to

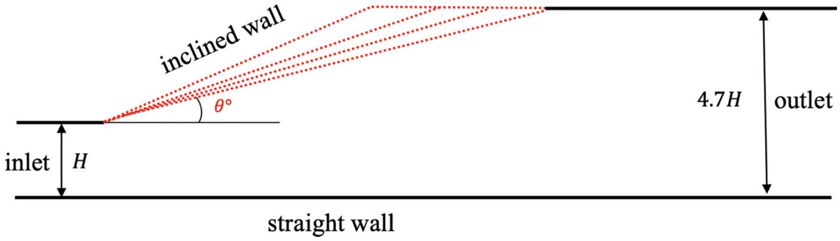


FIG. 9. The variation in opening angle θ of the diffuser flow cases.

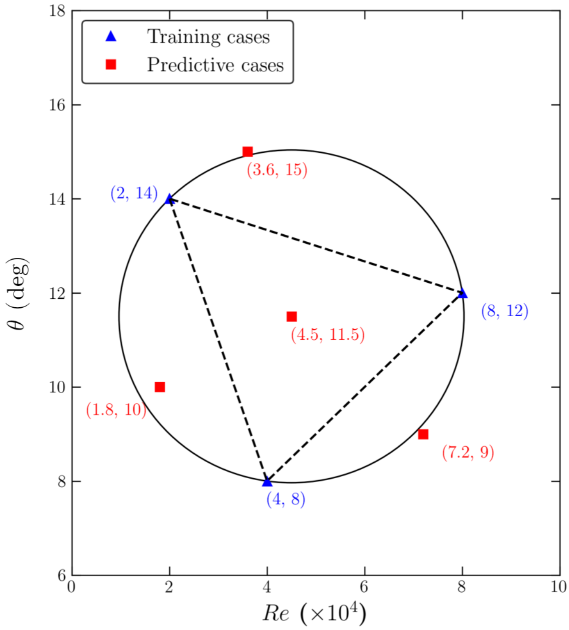
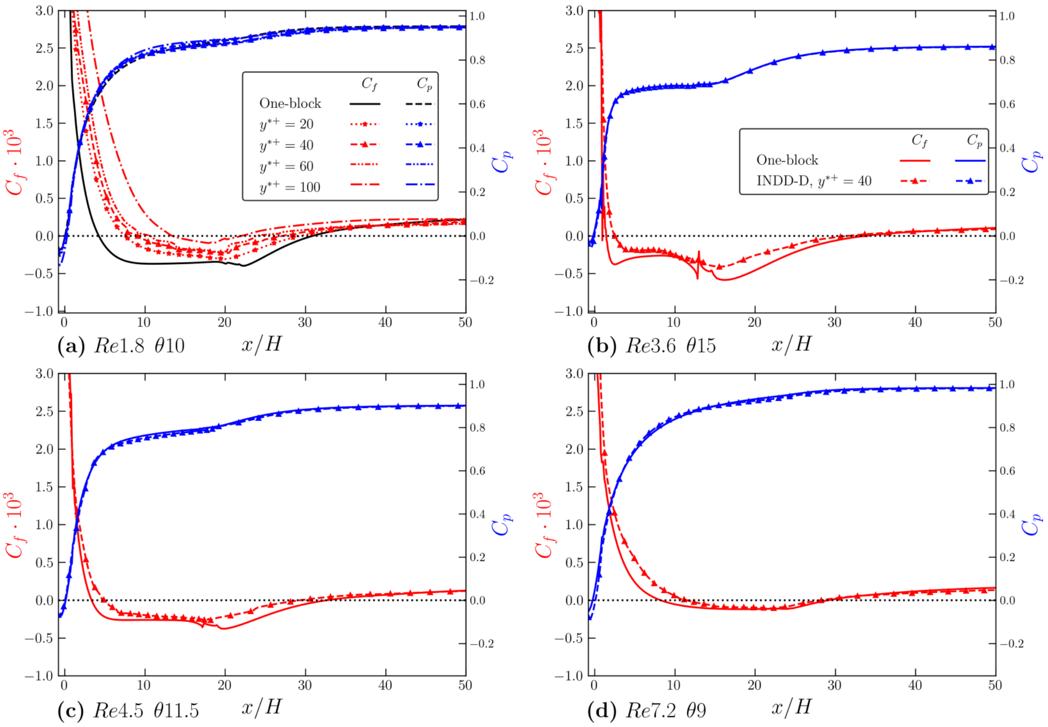


FIG. 10. The adopted training and predictive cases.



(a) $Re1.8\ \theta 10$

(b) $Re3.6\ \theta 15$

(c) $Re4.5\ \theta 11.5$

(d) $Re7.2\ \theta 9$

FIG. 11. Comparison of C_f and C_p on the inclined wall of the diffuser flow for different cases with *Re* and θ specified at the left bottom of respective subfigures. Subfigures (c) and (d) share the same legend as (b).

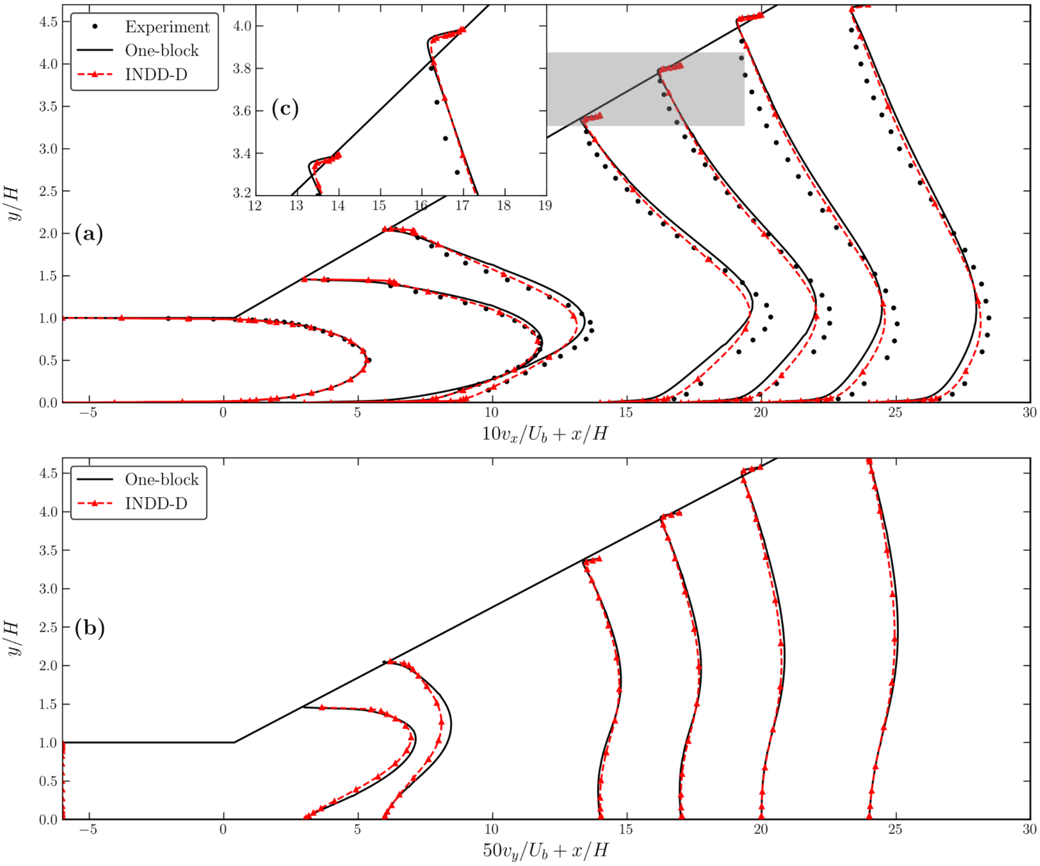


FIG. 12. Diffuser flow, Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$. (a) Comparison of mean stream-wise velocities v_x between the results from experiment [60], one-block solution, and implicit NDD with Spalart-Allmaras model. (b) Comparison of velocity component v_y between the results from one-block solution and implicit NDD with Spalart-Allmaras model. (c) Enlargement of the shaded region in (a).

the larger opening angle of the diffuser that requires extrapolation of the training model. Moreover, with the increase of the interface locations y^{*+}, the TBLE assumptions become less strictly valid. Eventually, it almost fails to capture the flow recirculation when the interface is far from the wall until $y^{*+} = 100$, as can be seen in subfigure (a). It is clear that the accuracy of the INDD, even with more accurate eddy viscosity profiles from the NN model, is largely determined by the aptitude of the thin-layer model in the inner region.

In Fig. 12, a noticeable recirculation region is present near the inclined wall of the diffuser with the SA model, which is not predicted by the LRN k-ε model under the same flow condition and diffuser geometry, thus giving a better agreement with the experimental data. This is basically because data-driven INDD is determined from the corresponding turbulence model and pretrained from their one-block data. The curves of velocity component v_x and v_y are consistent with the LRN results in subfigures (a) and (b), respectively.

D. Comparison of computation time

Apart from accuracy, high efficiency is also one of the targets in this work. We listed the computing time of diffuser flow cases with two turbulence models at Re $= 1.8 \times 10^4$ and $\theta = 10^\circ$ in Fig. 13. In particular, the error evolution of C_f along both walls with INDD-D at $y^{*+} = 40$ is

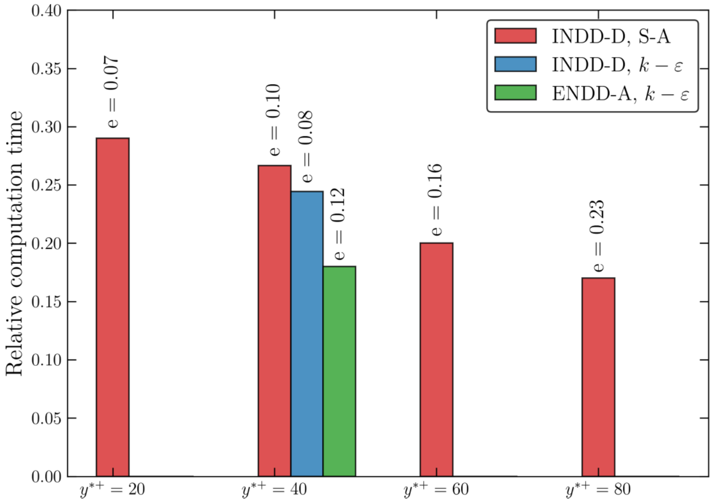


FIG. 13. Relative computational times with the explicit (approximate) and implicit NDD for both k-ε and Spalart-Allmaras models. The times are normalized by that of companion one-block solution.

compared with those with one-block solutions; see Appendix E. It should be noted that the model training time is not included. It can be seen that the computational cost of both ENDD-A and INDD-D is significantly lower than that of one-block simulations. This is because the proposed and conventional NDD have inherited the advantages of the NDD technique. In contrast to the full RANS equations for the outer layer, the computational cost for solving the boundary-layer equations in the wall vicinity is insignificant, where the Poisson equation for pressure is excluded. More importantly, the remaining momentum equations are highly simplified in the locally orthogonal wall-layer coordinates. In the data-driven implicit NDD, there is no need to solve k and ε equations or $\tilde{\nu}$ equation in the subgrid, which saves extra time. On the other hand, additional time is required to obtain the inner viscosity coefficient from the trained neural network model at every iteration. Overall, the INDD-D requires slightly more time than the approximate NDD with an algebraic function for eddy viscosity.

In addition, the maximum errors in C_f for the NDD simulations are calculated in terms of the LRN solutions and normalized by the skin friction at the inlet channel as

$$e = \frac{\max\left|C_f^{\mathrm{NDD}} - C_f^{\mathrm{LRN}}\right|}{\left|C_f^{\mathrm{inlet}}\right|}. \tag{19}$$

A smaller e indicates a closer agreement with the benchmark one-block data. It is observed that the accuracy of the NDD solution increases as the y^{*+} decreases. This is because errors by the simplification of equations and by the estimation of the turbulent viscosity are reduced, making the NDD solution converge to the LRN solution.

V. CONCLUSIONS

In this work, we developed a machine-learning-augmented implicit NDD method for near-wall turbulence modeling based on the feed-forward neural networks and the LRN data at selected conditions. The proposed algorithm has several advantages: (i) it benefits from the theoretical background of a well-established domain decomposition method; (ii) the machine-learning intrusion into the numerical scheme is kept to a minimum, thus guaranteeing a physics-informed scheme that provides comprehensible and interpretable results.

The prediction accuracy and generalization capacity of the proposed method were examined by comparing the predicted wall shear stresses with the corresponding one-block LRN data. For the test case of turbulent channel flow, overall good agreement is observed for the mean streamwise velocity and the wall shear stress. For the flow through the asymmetric diffuser, key flow quantities are reproduced with acceptable accuracy for both the k-ε model and the Spalart-Allmaras model. Using limited training data, the proposed method with the Spalart-Allmaras model can be generalized to different Reynolds numbers and geometries, even for the extrapolation states. Furthermore, with the NDD technique, the near-wall region can be efficiently decoupled from the outer main region, which, in return, strongly reduces the overall computational cost. The tradeoff between accuracy and computational time still remains.

This paper presents work to explore the feasibility of combining the NDD with data-driven methods. In principle, the proposed approaches can be easily extended to other turbulent models and more complex turbulent flows. A possible improvement of the method would be to include more high-fidelity data, like LES and DNS, during the training to generate a universal eddy viscosity profile for the specific flow. These topics will be investigated in future studies.

ACKNOWLEDGMENTS

S.L. acknowledges the China Scholarship Council (Grant No. 201708060474) for funding his Ph.D. study. J.K. would like to gratefully acknowledge the support of the Alexander von Humboldt Foundation (Ref. 3.5-CHN-1227287-HFST-P).

APPENDIX A: THE LAUNDER-SHARMA *k*-*ε* MODEL

The k-ε model by Launder and Sharma [4] is a two-equation low-Re (LRN) turbulence model, where the damping functions (f_ν, $f_{\varepsilon 1}$, $f_{\varepsilon 2}$) and extra source terms (ε_0, E) are introduced to properly solve k and ε equations down to the viscous sublayer. For a steady and incompressible turbulent flow, the k and ε equations can be written as

$$\begin{aligned} \nabla \cdot (Uk) - \nabla \cdot \left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\nabla k\right] &= P_k - (\tilde{\varepsilon} + \varepsilon_0), \\ \nabla \cdot (U\tilde{\varepsilon}) - \nabla \cdot \left[\left(\nu + \frac{\nu_t}{\sigma_\varepsilon}\right)\nabla \tilde{\varepsilon}\right] &= C_{\varepsilon 1} f_{\varepsilon 1}\frac{\tilde{\varepsilon}}{k}P_k - C_{\varepsilon 2} f_{\varepsilon 2}\frac{\tilde{\varepsilon}^2}{k} + E, \end{aligned} \tag{A1}$$

where

$$\nu_t = C_\mu f_\mu k^2/\tilde{\varepsilon}, \quad f_\mu = e^{\left(\frac{-3.4}{(1+\mathrm{Re}_\tau/50)^2}\right)}, \quad \mathrm{Re}_\tau = \frac{k^2}{\tilde{\varepsilon}\nu}, \quad f_{\varepsilon 2} = 1 - 0.3e^{-\mathrm{Re}_t^2},$$

$$\varepsilon_0 = 2\nu\left(\frac{\partial\sqrt{k}}{\partial y}\right)^2, \quad E = -2\nu\nu_t\left(\frac{\partial^2 u}{\partial y^2}\right)^2, \quad P_k = -\overline{u_i'u_j'}\frac{\partial u_j}{\partial x_i}.$$

The k-ε model constants are shown in Table I.

TABLE I. The constants for the Launder-Sharma k-ε model.

σ_k	σ_ε	$f_{\varepsilon 1}$	C_μ	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$
1.0	1.3	1.0	0.09	1.44	1.92

TABLE II. The constants for the Spalart-Allmaras model.

σ_ν	κ	C_{b1}	C_{b2}	C_{w1}	C_{w2}	C_{w3}	$C_{\nu 1}$
2/3	0.41	0.1355	0.622	$\frac{C_{b1}}{\kappa^2} + \frac{1+C_{b2}}{\sigma_{\nu_t}}$	0.3	2	7.1

APPENDIX B: THE SPALART-ALLMARAS MODEL

The Spalart-Allmaras model [53] is a one-equation turbulence model that solves for a variable $\tilde{\nu}$ and sequentially uses this quantity to modulate the turbulent viscosity ν_t. It should be noted that the implemented equation within the framework of OpenFOAM is identical to the standard version, except that trip f_{t1} and f_{t2} terms are dropped. The equation that solves $\tilde{\nu}$ for steady and incompressible flows is

$$\nabla \cdot (U\tilde{\nu}) - \frac{1}{\sigma_\nu}[\nabla \cdot (\nu + \tilde{\nu})\nabla\tilde{\nu} + C_{b2}|\nabla\tilde{\nu}|^2] = C_{b1}\tilde{S}\tilde{\nu} - C_{w1}f_w\frac{\tilde{\nu}^2}{\tilde{d}^2}, \tag{B1}$$

where

$$\tilde{S} = \Omega + \frac{\tilde{\nu}}{\kappa^2\tilde{d}^2}, \quad \Omega = \sqrt{2W_{ij}W_{ij}}, \quad W_{ij} = \frac{1}{2}\left(\frac{\partial u_j}{\partial x_i} - \frac{\partial u_i}{\partial x_j}\right),$$

$$f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu},$$

$$f_w = g\left[\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right]^{1/6}, \quad g = r + C_{w2}(r^6 - r), \quad r = \min\left(\frac{\tilde{\nu}}{\tilde{S}\kappa^2\tilde{d}^2}, 10\right),$$

and $\tilde{d}$ is the distance to the closest wall.

After that, the turbulence viscosity is obtained via

$$\nu_t = \tilde{\nu} f_{v1}. \tag{B2}$$

The model constants are given in Table II.

APPENDIX C: DETAILS OF THE NEURAL NETWORKS

Figure 14 shows the loss variations with the training epochs for training and validation data sets for the two turbulence models, i.e., the k-ε and SA model in subfigures (a) and (b), respectively. During the training, the weight and bias coefficients are progressively adjusted, and consequently the loss dramatically decreases during the first 1000 training epochs and then approaches a steady small value. Three different types of input features, i.e., NN-1, NN-2, and NN-3, in Table III, which include classical near-wall scale parameters $\alpha = u_\tau^2/u_{\tau p}^2$ and damping function $\mathcal{D} = 1 - \exp(-y^+/17)$, are further investigated. It should be noted that the flow characteristics that can be used as input features are limited, as the governing equations in the near-wall flow field are reduced to the truncated TBLE. The discrepancy among the three NN models in Figs. 14(a) and 14(b) indicates the NN-3 with additional α predicts slightly better, which is further applied to the test

TABLE III. Details of the neural networks. The tabulated hidden layer (HL) size contains the number of hidden layers and the number of neurons within each hidden layer.

NN	HL size	Input	Output
NN-1	1(40)	y^+, u^+, P_x	ν_t/ν
NN-2	1(40)	$y^+, u^+, P_x, \mathcal{D}$	ν_t/ν
NN-3	1(40)	y^+, u^+, P_x, α	ν_t/ν

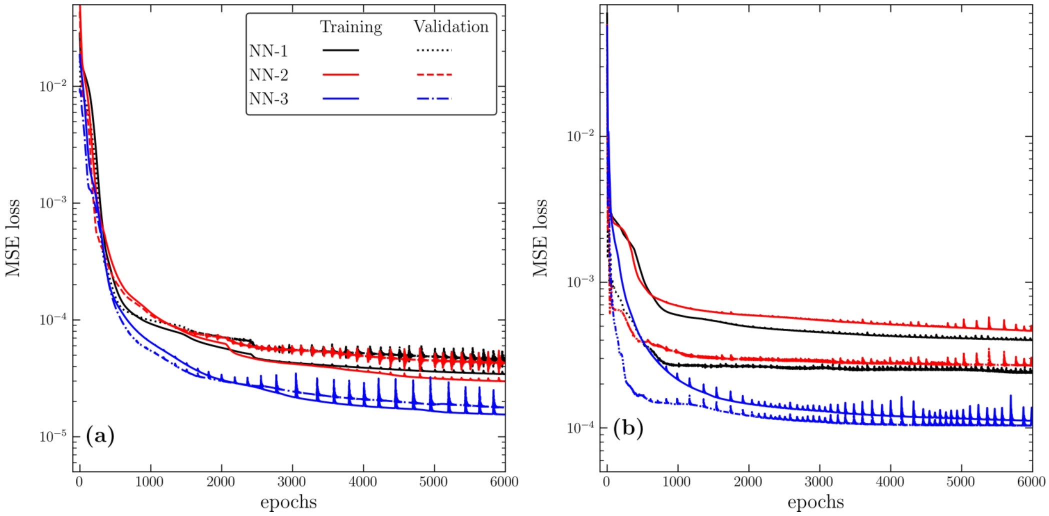


FIG. 14. The variations of training and validation loss with training epochs for different cases with the k-ε model (a) and the Spalart-Allmaras model (b).

cases in this study. The mapping performance between input and output features is markedly good in subfigure (a), as the test flow geometry is consistent with the training one. However, for the SA model, both the inlet flow condition (Re) and the flow geometry (opening angle θ) vary for different cases. More prominently, there exists a recirculation flow region near the inclined wall. Thus, the training process is more challenging, providing a relatively larger loss value in subfigure (b).

APPENDIX D: COMPARISON BETWEEN ORIGINAL INDD [20] AND THE PROPOSED DATA-DRIVEN INDD

The comparison of skin friction coefficient C_f and pressure coefficient C_p along the inclined wall of the diffuser (Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$), obtained from a one-block solution, original INDD, and data-driven INDD, can be seen in Fig. 15. The overall solutions with data-driven

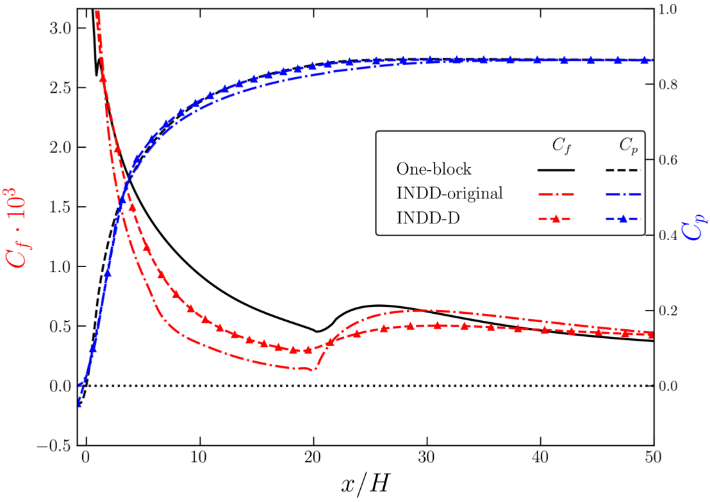


FIG. 15. Diffuser flow, Re $= 1.8 \times 10^4$, interface position $y^{*+} = 40$. Comparison of C_f and C_p along the inclined wall between the results from the one-block solution, original INDD [20], and data-driven INDD.

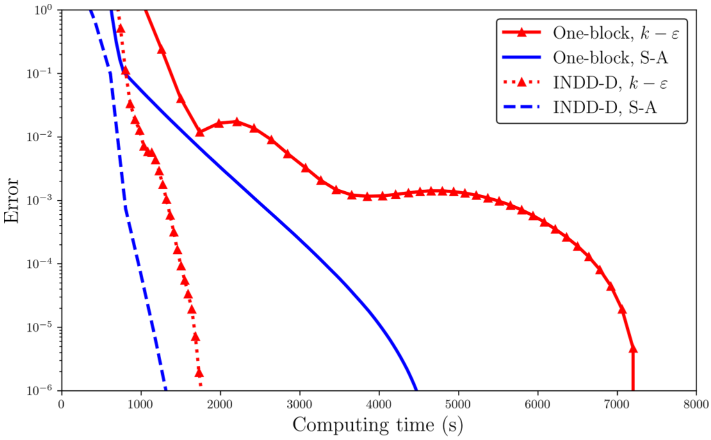


FIG. 16. Error evolution of the cases ($\mathrm{Re} = 1.8 \times 10^4,\ \ \theta = 10^o$) at $y^{*+} = 40$ with the k-ε model and the Spalart-Allmaras model.

INDD have a slightly better agreement with the one-block solutions than the counterparts with original INDD (based on solving the turbulent equations in a TBLE form). This is because, in the proposed data-driven INDD, the one-block results with the corresponding LRN k-ε turbulent model have been used as the training data set. In addition, compared to the one-block solution, the reduction of computation time with data-driven INDD is similar to that with original INDD. Although there is no need to solve the k and ε equations in the inner regions of data-driven INDD, additional time is required for the viscosity profiles output from the neural network at every iteration.

APPENDIX E: ERROR EVOLUTION

The convergence is evaluated by the mean-square relative error of key parameter C_f along the walls:

$$\mathcal{J}_{\text{mse}} = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{C_f(i) - C_f^{\text{final}}(i)}{C_f^{\text{final}}(i)}\right]^2, \tag{E1}$$

where C_f^{final} is the final converged values of C_f (Fig. 16).

[1] A. J. Smits, B. J. McKeon, and I. Marusic, High–reynolds number wall turbulence, Annu. Rev. Fluid Mech. **43**, 353 (2011).

[2] U. Piomelli, Wall-layer models for large-eddy simulations, Prog. Aerosp. Sci. **44**, 437 (2008).

[3] J. S. Baggett, J. Jimenez, and A. G. Kravchenko, Resolution requirements in large-eddy simulations of shear flows, Annu. Res. Briefs 51 (1997).

[4] B. E. Launder and B. I. Sharma, Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc, Lett. Heat Mass Transf. **1**, 131 (1974).

[5] K.-Y. Chien, Predictions of channel and boundary-layer flows with a low-reynolds-number turbulence model, AIAA J. **20**, 33 (1982).

[6] D. C. Wilcox *et al.*, *Turbulence Modeling for CFD* (DCW Industries La Canada, CA, 1998), Vol. 2.

[7] B. Launder and D. Spalding, The numerical computation of turbulent flows, Comput. Methods Appl. Mech. Eng. **3**, 269 (1974).

[8] H. Grotjans and F. Menter, Wall functions for general application CFD codes, Comput. Fluid Dyn. Eur. Conf. **1**, 1112 (1998).

[9] T. Craft, A. Gerasimov, H. Iacovides, and B. Launder, Progress in the generalization of wall-function treatments, Int. J. Heat Fluid Flow **23**, 148 (2002).

[10] T. Craft, S. Gant, H. Iacovides, and B. Launder, A new wall function strategy for complex turbulent flows, Numer. Heat Transf., Pt. B **45**, 301 (2004).

[11] S. Gant, Development and application of a new wall function for complex turbulent flows, Ph.D. thesis, University of Manchester, 2003.

[12] M. Popovac and K. Hanjalic, Compound wall treatment for rans computation of complex turbulent flows and heat transfer, Flow Turbul. Combust. **78**, 177 (2007).

[13] S. Utyuzhnikov, Some new approaches to building and implementation of wall-functions for modeling of near-wall turbulent flows, Comput. Fluids **34**, 771 (2005).

[14] S. Utyuzhnikov, Robin-type wall functions and their numerical implementation, Appl. Numer. Math. **58**, 1521 (2008).

[15] A. Jones and S. Utyuzhnikov, Application of a near-wall domain decomposition method to turbulent flows with heat transfer, Comput. Fluids **119**, 87 (2015).

[16] S. Utyuzhnikov, Domain decomposition for near-wall turbulent flows, Comput. Fluids **38**, 1710 (2009).

[17] S. Utyuzhnikov, Towards development of unsteady near-wall interface boundary conditions for turbulence modeling, Comput. Phys. Commun. **185**, 2879 (2014).

[18] M. Petrov, S. Utyuzhnikov, A. Chikitkin, and V. Titarev, On extension of near-wall domain decomposition to turbulent compressible flows, Comput. Fluids **210**, 104629 (2020).

[19] A. Jones and S. Utyuzhnikov, Efficient computation of turbulent flow in ribbed passages using a non-overlapping near-wall domain decomposition method, Comput. Phys. Commun. **217**, 1 (2017).

[20] S. Lyu and S. Utyuzhnikov, A computational slip boundary condition for near-wall turbulence modeling, Comput. Fluids **246**, 105628 (2022).

[21] S. T. Bose and P. Moin, A dynamic slip boundary condition for wall-modeled large-eddy simulation, Phys. Fluids **26**, 015104 (2014).

[22] H. J. Bae, A. Lozano-Durán, S. T. Bose, and P. Moin, Dynamic slip wall model for large-eddy simulation, J. Fluid Mech. **859**, 400 (2019).

[23] S. Utyuzhnikov, Generalized wall functions and their application for simulation of turbulent flows, Int. J. Numer. Methods Fluids **47**, 1323 (2005).

[24] E. Balaras, C. Benocci, and U. Piomelli, Two-layer approximate boundary conditions for large-eddy simulations, AIAA J. **34**, 1111 (1996).

[25] W. Cabot and P. Moin, Approximate wall boundary conditions in the large-eddy simulation of high reynolds number flow, Flow Turbul. Combust. **63**, 269 (2000).

[26] S. Kawai and J. Larsson, Wall-modeling in large eddy simulation: Length scales, grid resolution, and accuracy, Phys. Fluids **24**, 015105 (2012).

[27] M. Wang and P. Moin, Dynamic wall modeling for large-eddy simulation of complex turbulent flows, Phys. Fluids **14**, 2043 (2002).

[28] S. Kawai and J. Larsson, Dynamic non-equilibrium wall-modeling for large eddy simulation at high reynolds numbers, Phys. Fluids **25**, 015105 (2013).

[29] C. Duprat, G. Balarac, O. Métais, P. M. Congedo, and O. Brugière, A wall-layer model for large-eddy simulations of turbulent flows with/out pressure gradient, Phys. Fluids **23**, 015101 (2011).

[30] A. Jones and S. Utyuzhnikov, A near-wall domain decomposition approach in application to turbulent flow in a diffuser, Appl. Math. Modell. **40**, 329 (2016).

[31] A. Chikitkin, S. Utyuzhnikov, M. Petrov, and V. Titarev, Non-overlapping domain decomposition for modeling essentially unsteady near-wall turbulent flows, Comput. Fluids **202**, 104506 (2020).

[32] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. **52**, 477 (2020).

[33] J. Kou and W. Zhang, Data-driven modeling for unsteady aerodynamics and aeroelasticity, Prog. Aerosp. Sci. **125**, 100725 (2021).

[34] B. D. Tracey, K. Duraisamy, and J. J. Alonso, A machine learning strategy to assist turbulence model development, in *53rd AIAA Aerospace Sciences Meeting* (AIAA, Kissimmee, Florida, 2015), p. 1287.

[35] D. A. Bezgin, S. J. Schmidt, and N. A. Adams, A data-driven physics-informed finite-volume scheme for nonclassical undercompressive shocks, J. Comput. Phys. **437**, 110324 (2021).

[36] A. D. Jagtap, Z. Mao, N. Adams, and G. E. Karniadakis, Physics-informed neural networks for inverse problems in supersonic flows, J. Comput. Phys. **466**, 111402 (2022).

[37] M. Milano and P. Koumoutsakos, Neural network modeling for near wall turbulent flow, J. Comput. Phys. **182**, 1 (2002).

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

[39] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, J. Fluid Mech. **807**, 155 (2016).

[40] J.-X. Wang, J.-L. Wu, and H. Xiao, Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on DNS data, Phys. Rev. Fluids **2**, 034603 (2017).

[41] E. J. Parish and K. Duraisamy, A paradigm for data-driven predictive modeling using field inversion and machine learning, J. Comput. Phys. **305**, 758 (2016).

[42] J. Weatheritt and R. Sandberg, A novel evolutionary algorithm applied to algebraic modifications of the rans stress–strain relationship, J. Comput. Phys. **325**, 22 (2016).

[43] Y. Zhao, H. D. Akolekar, J. Weatheritt, V. Michelassi, and R. D. Sandberg, Rans turbulence model development using CFD-driven machine learning, J. Comput. Phys. **411**, 109413 (2020).

[44] A. Beck, D. Flad, and C.-D. Munz, Deep neural networks for data-driven les closure models, J. Comput. Phys. **398**, 108910 (2019).

[45] R. Maulik, O. San, J. D. Jacob, and C. Crick, Sub-grid scale model classification and blending through deep learning, J. Fluid Mech. **870**, 784 (2019).

[46] X. I. A. Yang, S. Zafar, J.-X. Wang, and H. Xiao, Predictive large-eddy-simulation wall modeling via physics-informed neural networks, Phys. Rev. Fluids **4**, 034602 (2019).

[47] Z. Zhou, G. He, and X. Yang, Wall model based on neural networks for les of turbulent flows over periodic hills, Phys. Rev. Fluids **6**, 054610 (2021).

[48] H. J. Bae and P. Koumoutsakos, Scientific multi-agent reinforcement learning for wall-models of turbulent flows, Nat. Commun. **13**, 1443 (2022).

[49] A. Lozano-Durán and H. J. Bae, Machine learning building-block-flow wall model for large-eddy simulation, J. Fluid Mech. **963**, A35 (2023).

[50] L. Zhu, W. Zhang, J. Kou, and Y. Liu, Machine learning methods for turbulence modeling in subsonic flows around airfoils, Phys. Fluids **31**, 015105 (2019).

[51] P. S. Volpiani, R. F. Bernardini, and L. Franceschini, Neural network-based eddy-viscosity correction for RANS simulations of flows over bi-dimensional bumps, Int. J. Heat Fluid Flow **97**, 109034 (2022).

[52] A. P. Singh, S. Medida, and K. Duraisamy, Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils, AIAA J. **55**, 2215 (2017).

[53] P. Spalart and S. Allmaras, A one-equation turbulence model for aerodynamic flows, in *30th Aerospace Sciences Meeting and Exhibit* (Reno, NV, USA, 1992), p. 439.

[54] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[55] M. Manhart, N. Peller, and C. Brun, Near-wall scaling for turbulent boundary layers with adverse pressure gradient: A priori tests on dns of channel flow with periodic hill constrictions and dns of separating boundary layer, Theor. Comput. Fluid Dyn. **22**, 243 (2008).

[56] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. **12**, 620 (1998).

[57] S. V. Patankar and D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, in *Numerical Prediction of Flow, Heat Transfer, Turbulence and Combustion* (Elsevier, Amsterdam, 1983), pp. 54–73.

[58] J. P. Van Doormaal and G. D. Raithby, Enhancements of the simple method for predicting incompressible fluid flows, Numer. Heat Transf. **7**, 147 (1984).

[59] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, Pytorch: An imperative style, high-performance deep learning library, Adv. Neural Inf. Proc. Syst. **32** (2019).

[60] C. U. Buice and J. K. Eaton, Experimental investigation of flow through an asymmetric plane diffuser, Centre Turbul. Res. Annu. Res. Brief. **1995**, 117 (1995).

[61] A. Hellsten, P. Rautaheimo, and M. Orpana, 8th Ercoftac/Iahr/Cost Workshop on Refined Turbulence Modelling, ERCOFTAC, Helsinki University of Technology, Espoo, Finland, 1999.

[62] J. Gullman-Strand, O. Törnblom, B. Lindgren, G. Amberg, and A. V. Johansson, Numerical and experimental study of separated flow in a plane asymmetric diffuser, Int. J. Heat Fluid Flow **25**, 451 (2004).