A comparative analysis of multi-machine learning algorithms for data-driven RANS turbulence modelling

Junjie Wu¹, Jiahua Li²*, Xiang Qiu¹, Yulu Liu¹
¹ School of Science, Shanghai Institute of Technology, Shanghai 201418, China
² College of Urban Construction and Safety Engineering, Shanghai Institute of Technology, Shanghai 201418, China
*Corresponding author’s e-mail: 13916592953@163.com

Abstract. To improve the speed or accuracy of Reynolds-averaged Navier-Stokes in numerical simulations of turbulence, five machine learning algorithms including Random Forests, eXtreme Gradient Boosting, K-Nearest Neighbours, Support Vector Machine and Artificial Neural Network are introduced in this paper. We establish the nonlinear mapping relationship between the average flow field and the steady-state eddy viscosity. The machine learning surrogate models for the Spalart-Allmaras turbulence model are constructed and used to solve the backward facing step problem. It is demonstrated that the machine learning surrogate models constructed by the five machine learning algorithms can effectively express the complex mapping relationship between the average turbulent flow field and the steady-state eddy viscosity. They can not only obtain reasonable simulation accuracy but also significantly accelerate the solution time of the flow field. The algorithms of eXtreme Gradient Boosting, K-Nearest Neighbours and Artificial Neural Network have better performance when considering the calculation accuracy, time cost and memory cost. It shows the great potential of applying machine learning algorithms to RANS turbulence model and also provides a new idea for industrial simulations of turbulent flows.

1. Introduction

It is now well established from a variety of studies, Reynolds-averaged Navier-Stokes simulation (RANS), large eddy simulation (LES) and direct numerical simulation (DNS) have been commonly used in simulations of turbulent flows. However, it is difficult to simulate a case which has an extremely high grid resolution in DNS or LES due to the limitation of computing power and algorithmic technology. Hence, RANS model will continue to play a vital role in industrial simulations of turbulent flows for its convenience and efficiency during the foreseeable future.

However, it is well known that the simulation accuracy of RANS will depend on the choice of a suitable turbulence model and solver. A key shortcoming of RANS turbulence models is their lack of stability and robustness, i.e. the difficulty in achieving convergence [1]. In this context, it makes sense to develop a method that can improve the simulations results of RANS. In recent years, with the rapid development of artificial intelligence technology, a large number of open-source machine learning (ML) and deep learning frameworks (such as Seikit-Learn and TensorFlow, etc.) [2-3] have sprung up, as well as the rapid improvement of GPU parallel computing capability, which has greatly reduced the threshold for researchers to engage in relevant work. At this time, the combination of ML algorithms...
and turbulence modelling has become a new direction, and the establishment of data-driven model to improve the simulations of RANS has also become a research hotspot [4]. Wang et al. [5] utilized the Random Forests and Neural Networks to propose a data-driven, physics-informed machine learning approach for reconstructing discrepancies in RANS modelled Reynolds stresses. Excellent predictive performances were observed in separated flows and secondary flows. Ling et al. [6] proposed a method of using Deep Neural Networks to learn a model for the Reynolds stress anisotropy tensor from high-fidelity simulations data. A novel Neural Network architecture was proposed which used a multiplicative layer with an invariant tensor basis to embed Galilean invariance into the predicted anisotropy tensor. The significant improvement versus baseline RANS linear eddy viscosity and nonlinear eddy viscosity models was demonstrated. Maulik et al. [7] introduced a surrogate model for the turbulent eddy viscosity was assessed for parametric interpolation, while solving for the pressure and velocity equations to steady-state, thus representing a framework that was hybridized with the Artificial Neural Network. Accurate steady-state results with significant reduction in solution time were achieved when compared to those obtained by the Spalart-Allmaras one-equation model. Although these data-driven turbulence model methods can effectively improve the flow field or accelerate the solution process. The research on the selection and performance of ML algorithms is still lacking. At present, most of the existing studies believe that Neural Networks and Random Forests are ideal algorithms for data-driven turbulence modelling [8]. The purpose of our work is to find the optimal ML algorithms to be applied to turbulence modelling through calculation and comparison of five supervised learning algorithms.

Five supervised learning algorithms of Random Forests, eXtreme Gradient Boost, K-Nearest Neighbours, Support Vector Machine and Artificial Neural Network are adopted in our work. The steady-state average flow field data under different Reynolds numbers is used to train separately and construct the nonlinear mapping relationship between the average flow field and steady-state eddy viscosity. Then, the ML surrogate turbulence models are constructed to utilize the frozen steady-state eddy viscosity predicted by the trained model to solve the Reynolds-averaged pressure equations and velocity equations. Finally, the simulation results of each ML surrogate models are compared and analysed. The whole process can be divided into two parts. The first part is to select the flow features, and the eddy viscosity will be treated as a training target by five ML regression algorithms to establish ML surrogate models separately. In the second part, the ML surrogate models are coupled to the solver, and the predictions and simulations are carried out in the process of solving the flow field.

2. Methodology

2.1. Summary of proposed approach

This section will introduce the overall architecture of the data-driven turbulence model in the current work. The regression model is constructed through five ML algorithms. A brief introduction of the selected algorithms are as follows:

a) Random Forests (RF): RF is an ensemble algorithm built from a number of decision trees that can be used for classification or regression problems [9]. Each decision tree is a learner. For an input sample, multiple decision trees have different predictions. Random Forests integrate the voting results of all decision trees. The randomness of Random Forests is reflected in two aspects: the first is the randomness formed by the internal decision tree, and the second is the randomness of data selection and feature selection.

b) eXtreme Gradient Boosting (XGB): XGB belongs to the Boosting algorithm, which is an ensemble algorithm composed of multiple base learners, and its model equations can be written as follows:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

(1)
where $f_k$ denotes the $k$ base learner, $K$ denotes the total number of base learners, $x_i$ is the $i$ input of the $i$ sample, and $\hat{y}_i$ is the prediction result of the $i$ sample. Compared with the traditional Boosting algorithm, the algorithm greatly improves the fitting ability by using the second derivative when calculating the loss function. Besides, it also adds a regular term to control the complexity of the model and control the over-fitting situation [10], which has strong learning performance.

c) K-Nearest Neighbours (KNN): KNN algorithm is a non-parametric statistical method that can be used for classification or regression problems [11]. The principle of the algorithm is to find the closest $K$ samples in the training set based on the Euclidean distance or Manhattan distance given the test data when the features and labels in the training set are known. The predicted label corresponding to the test data is among these $K$ samples most frequently occurring. Euclidean distance is adopted in our work, which is written in the following form:

$$d(X, Y) = \left( \sum_{k=1}^{n} (x_k - y_k)^2 \right)^{1/2}$$

where $X$ denotes the specific training set $X = (x_1, x_2, \ldots, x_n)$, $Y$ denotes the specific testing set $Y = (y_1, y_2, \ldots, y_n)$ with the same number of $X$.

d) Support Vector Machine (SVM): SVM is a supervised learning algorithm that can be used for classification and regression problems. The core of the algorithm is to find a hyperplane that can maximize the distance from the support vectors to the hyperplane. When dealing with regression problems, it is necessary to introduce a relaxation factor $\varepsilon$ to make the original loss function $\varepsilon$-insensitive, that is, the loss function is calculated only when the deviation between the predicted value and the true value is greater than $\varepsilon$, otherwise the loss function is 0 [12].

e) Artificial Neural Network (ANN): ANN is mainly composed of three parts, the input layer, the output layer and the hidden layer. The neurons in each layer are connected by weights and biases. The output of each layer is set with an activation function to introduce a nonlinear factor to the neuron. So that the neural network can approach any nonlinear function arbitrarily. The neural network learning process in our work is to build a loss function between the output and the true label of the sample data, according to the gradient descent algorithm of error back-propagation [13], train a multilayer feedforward neural network and iteratively reduce the loss function to update weights. Neural Networks have obvious advantages in dealing with nonlinear problems.

In our work, a backward facing step problem is selected as the research case. In this case, a turbulent boundary layer encounters a sudden back step, causing flow separation. The flow then reattaches and recovers downstream of the step. It is a typical fluid mechanics case used to validate turbulence models. The boundary conditions of the numerical simulation are employed based on the experiments of Driver and Seegmiller [14]. The geometry of the computational domain is shown in Figure 1(a). The airflow is simulated in our research and the step height $H$ is 0.00127 m. The height of the inlet is set as 8$H$, and the height of the outlet is set as 9$H$. The distance from the inlet to the step is set as 130$H$, and the distance from the step to the outlet is set to 50$H$ to ensure the full development of the flow. The black dash lines are the probe locations of $x/H = 1$ and $x/H = 4$ (denoted L1 and L2), which are used to evaluate the simulation results of each ML surrogate model. The total number of computing cells is 20540. To ensure that the separation phenomenon can be captured, the grid near the step is refined, as shown in Figure 1(b). The one-equation Spalart-Allmaras (SA) turbulence model [15] based on Boussinesq eddy viscosity hypothesis is adopted for simulation to obtain training and comparison data. The Reynolds number, based on the step height $H$ and the inlet free stream velocity $U$, is defined as $Re_H = \frac{U}{H}/\nu$, where $\nu$ is kinematic viscosity set as $1.56 \times 10^{-5}$ $m^2s^{-1}$. Different Reynolds numbers are achieved by varying the free stream velocity.

In our work, the training data comes from the average flow field of 10 kinds of Reynolds number simulation results, and the Reynolds number is controlled by the free stream velocity, respectively $Re_H = 32000, 33000, 34000, 35000, 37000, 38000, 39000, 40000, 41000$ and $42000$. All simulations
are carried out on the open-source computational fluid dynamics (CFD) platform OpenFOAM, and the built-in steady-state incompressible turbulence solver simpleFoam is adopted [16].

Figure 1. (a) The geometry of the backward facing step used for simulation and (b) the local refined mesh.

The test flow is carried out at a Reynolds number of 36000. The open-source Scikit-Learn [3], TensorFlow [3] and XGBoost [10] frameworks are used to build an ML surrogate model for the relationship between Reynolds number and average flow field. In general, the testing process can be divided into three steps. The first step is to generate initial conditions treated as the input of the test flow. The second step is to predict the steady-state eddy viscosity of the corresponding parameter space through the ML models that has been trained and deployed. The last step is to utilize the frozen steady-state eddy viscosity predicted by the trained model to solve the Reynolds-averaged pressure equations and velocity equations.

2.2. Input and output of Machine Learning models
In the construction of the ML algorithm models, the characteristic input and output of the model are also important components. Hence, the input features of the model should not only reflect the main characteristics of the local flow field but also ensure the averaged flow data to avoid the occurrence of extreme abnormal values. According to these requirements, the method proposed by Ling et al. [17] to construct another set of an invariant input set from a given set of tensors is adopted in our study. The final input set $Q$ is as follows:

$$Q = \{ S^*, \Omega^*, \nabla p^* \}$$

where $S$ and $\Omega$ indicate strain rate tensor and rotation rate tensor. In addition, considering the physical properties of the flow field, we add a pressure gradient term $\nabla p$ in the input features set. In the study of Wu et al. [18], the gradient term of turbulent kinetic energy $\nabla k$ is also considered. This term is omitted in our work for the reason that the RANS simulation flows with the same strain rate tensor $S$ and rotation rate tensor $\Omega$, the gradient of turbulent kinetic energy $\nabla k$ is also the same.

In order to improve the generalization ability of the model and reduce the redundancy of feature information, all data sets are normalized, which is represented by the superscript *. The normalization method is processed by zero-mean normalization. The input features normalized by this method conform to the standard normal distribution, which is conducive to train. The normalization equation can be written as:

$$x^* = (x - \mu)\sigma^{-1}$$

where $\mu$ represents the mean of all samples, and $\sigma$ represents the standard deviation of all samples. The training target of the model in our work is to select the corresponding parameter space steady-state turbulent eddy viscosity field $\nu_t$ under each Reynolds number. The entire training model can be simplified as:

$$\nu_t^* = f_{ML}(Q)$$

In this study, it should be noted that the training features $Q$ and the training target which is steady-state eddy viscosity field $\nu_t$ in the training data set are all based on the simulation results of the original SA turbulence model.
2.3. Construction and evaluation methods of ML models

Given the initial boundary conditions, five ML architectures are integrated into OpenFOAM to predict and froze the steady-state eddy viscosity field and simulate the backward facing step to convergence respectively. The comparison test results of the five ML surrogate models will be expanded in the flow of $Re_H = 36000$.

The optimal hyperparameters are determined using GridSearchCV [2] to build ML models. Meanwhile, 10-fold cross-validation is utilized to evaluate the accuracy of ML models according to the metrics between the true value and the predicted value. The principle is to divide the training set of 205,400 samples into ten parts, without repeating one of them as the test set, and the remaining nine parts as the training set to train the model. After calculating the evaluation metric $Err_i$ on the test set of each time, the final evaluation value of the ML model is obtained by averaging of ten metrics, which can be represented as:

$$CV(10) = \frac{1}{10} \sum_{i=1}^{10} Err_i$$  \hspace{1cm} (6)

The coefficient of determination $R^2$, the root mean square error $RMSE$ and the mean absolute error $MAE$ are selected as metrics to evaluate the model in our study. $R^2$ can be used to measure the effect of model fitting. Its value range is between 0 and 1, the larger the value, the higher the fitting accuracy. $RMSE$ can be used to measure the degree of deviation between the predicted value and the true value. The smaller the value, the stronger the agreement between the predicted value and the true value, reflecting the higher the accuracy of the model. $MAE$ more directly reflects the error of the predicted value. The smaller the value, the smaller the model prediction error. The calculation equations of the three metrics are as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$  \hspace{1cm} (7)

$$RMSE = \left[\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2\right]^{-1}$$  \hspace{1cm} (8)

$$MAE = \left[\sum_{i=1}^{n} |y_i - \hat{y}_i| / n\right]^{-1}$$  \hspace{1cm} (9)

where $n$ is the total number of samples, $y_i$ denotes the true value of the data sample, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the average of all the true value.

3. Results

3.1. Analysis of ML models accuracy

The ML models are deployed to OpenFOAM to construct surrogate SA turbulence models. The initial boundary conditions are set. Then the simple algorithm solver substitutes the frozen steady-state eddy viscosity field predicted by the ML model into the velocity and pressure equations to convergence. The test case will be simulated on the Reynolds number $Re_H = 36000$ (the free-stream velocity is set as $44.2 \text{ m/s}$). The evaluation metrics of $R^2$, $RMSE$ and $MAE$ are utilized to measure the accuracy of five ML models. Besides, statistics are made on the time and memory consumption of each ML model training and storage process, as shown in Table 1.

From the data in Table 1, it can be seen that the $R^2$ values of the five ML models are distributed between 0.818601 and 0.998993, and the fitting results of each model are relatively ideal. Except for SVM whose $R^2$ value is around 0.8, the $R^2$ values of the other four models are all above 0.99. RF has the highest fitting degree. The $RMSE$ value of each model is in the range of 0.010276 to 0.666347. The $RMSE$ value of SVM is the largest among the five ML models. On the contrary, the model with the smallest $RMSE$ value is RF. By comparing the $MAE$ values of each ML model, it is similar that SVM
is still the model with the maximum deviation, RF is still the model with the minimum deviation, and XGB is close to RF, with the deviation slightly greater than RF. It can be seen from Table 1 that SVM takes the longest time to build an ML model, which takes 2176.2s, and the model with the shortest construction time is KNN, which takes only 4.2s. In the statistics of the memory occupied by the model, RF occupies the largest with a size of about 8.3GB, followed by KNN, with a model size of about 15.9MB, and ANN with the smallest occupancy, which occupies only 444KB.

Table 1. Evaluation, building time and memory size of different ML models.

| ML models | R^2   | RMSE  | MAE   | Building time (s) | Memory size (KB) |
|-----------|-------|-------|-------|-------------------|------------------|
| RF        | 0.999893 | 0.010276 | 0.004502 | 253.1              | 8,785,011        |
| XGB       | 0.999809 | 0.013759 | 0.008464 | 46.2               | 3,832            |
| KNN       | 0.994876 | 0.071302 | 0.033890 | 4.2                | 16,303           |
| SVM       | 0.818601 | 0.666347 | 0.462479 | 2176.2             | 9,195            |
| ANN       | 0.998701 | 0.034641 | 0.014491 | 81.9               | 444              |

In order to better illustrate the correlation between the predicted results of each model and the true results, eddy viscosity field simulated by the SA turbulence model are compared with the predicted field of each ML model, as shown in Figure 2. The x-axis represents the true value of the eddy viscosity simulated by the SA turbulence model, and the y-axis represents the predicted value of each ML model. The distance between the scattered points in the figure and the black dashed line represents the absolute error between the two values, and the color distribution represents the density distribution of the scattered points. It is found that the prediction value of RF and XGB algorithms have the smallest deviation from the truth value, and the other three models can see obvious deviations. The maximum deviations are mainly distributed at the lower magnitudes zone, which can be caused by the prediction deviation in the transition from the near-wall separation zone to the free-stream zone.

On the whole, the five ML models can accurately predict the two peaks of the eddy viscosity field and can describe the complex relationship between the eddy viscosity field and the average flow field. Among them, the RF and XGB algorithms have a stronger ability to express the eddy viscosity field.

3.2. Numerical results

Figure 3 respectively shows the steady-state eddy viscosity field distributions obtained by the convergent simulations of the ML surrogate models constructed by the five ML models at the detection positions of L1 and L2. Obviously, the simulation results of the RF surrogate model and the XGB surrogate model are in good agreement with the original simulation values of SA, and there is only a slight deviation between the calculation results of the KNN surrogate model and the ANN surrogate model. Compared with the other four ML surrogate models, the SVM surrogate model has the largest simulation deviation, especially the peak near the wall. At both positions L1 and L2, it is less than the SA steady-state eddy viscosity value.

Figure 4 shows the comparison of velocity profile predictions simulated by the five ML surrogate models at L1 and L2 positions. In addition, the original SA simulation performed with OpenFOAM and the same working conditions simulation performed by Rumsey et al. [19] using NASA’s open-source program CFL3D as reference. For dimensionless reference velocity $U_{ref}$ is the free-stream velocity at the centerline of $x/H = -4$. It is clearly found that, compared with the simulation results of the original SA model, the simulations results of the RF and XGB surrogate models on the velocity distribution are in good agreement, and are the closest to the SA among all. Secondly, the KNN surrogate model only has some slight deviation near the wall area and other positions are basically in good agreement with the SA model. The last two models, SVM and ANN, have relatively large deviations compared with the SA model. At the L1 position, the simulation deviation of the SVM
surrogate model near the wall is significantly larger than the ANN surrogate model, but the position deviation above the steps is more obvious. At the L2 position, the simulation deviations of several ML surrogate models are similar to the L1 position.

In general, the simulation results of each ML surrogate model are basically satisfactory, showing that each ML algorithm has a strong ability to reflect the average flow field. Although there are slight deviations, there is still potential for improving traditional simulation methods.

Figure 2. Comparison and correlation between predicted and true values of eddy viscosity of five ML models, where (a) RF; (b) XGB; (c) KNN; (d) SVM; (e) ANN.

Figure 3. Average flow steady-state eddy viscosity predictions (in $m^2 s^{-1}$) at (a) L1 and (b) L2 positions for five ML surrogate models.

In general, the simulation results of each ML surrogate model are basically satisfactory, showing that each ML algorithm has a strong ability to reflect the average flow field. Although there are slight deviations, there is still potential for improving traditional simulation methods.
Compared with the simulation time of the original SA model, the ML surrogate models constructed by the five ML algorithms can all accelerate the process. The simulation process of the original SA turbulence model requires 3642 iterations to converge, and the surrogate models built by ML algorithms, except SVM, only need about 700 iterations to converge. The fastest ANN algorithm is only 687 times. The iterations number of RF and the iterations number of XGB are almost the same. While SVM requires 1148 iterations, but this is only 1/3 of the original SA simulation time, as shown in Figure 5. Therefore, the ML surrogate models can speed up the simulations process with reasonable accuracy, greatly reduce time costs, and thus reduce the cost of computing resources to a certain extent. It provides a new idea for engineering simulations.

Figure 4. Average flow free-stream velocity predictions at (a) L1 and (b) L2 positions for five ML surrogate models.

Figure 5. Comparison of the number of convergent iterations simulated by ML surrogate models.

4. Conclusion
Aiming at the closure problem of RANS, we utilize the existing simulated data and five ML algorithms: Random Forest, eXtreme Gradient Boosting, K-Nearest Neighbours, Support Vector Machine, and Artificial Neural Network are adopted to describe the complex and non-linear mapping relationship between the average flow field and steady-state eddy viscosity. The ML surrogate models for SA turbulence model are constructed and a backward facing step problem is analysed and compared in our work.

The results of calculating $R^2$, RMSE and MAE of each model show that RF has a better fitting ability, followed by XGB, and the worst is SVM. From the perspective of the memory occupied by each model, RF takes up a lot of memory space, and ANN takes up the least memory. The difference between other models is not obvious. In terms of time-consuming, SVM takes the most time, followed by RF, and the least time is KNN. According to the simulations results of the flow field using ML surrogate models, the results of the RF surrogate model and the XGB surrogate model are basically consistent with the results of the SA turbulence model. The results of the ANN surrogate model and
the KNN surrogate model only have slight deviations. Compared with the other four models, the SVM surrogate model has the largest deviation but it is still satisfactory overall. In the case of comprehensive consideration of simulation accuracy, time cost and memory cost, the XGB algorithm, KNN algorithm and ANN algorithm have more advantages.

We construct a comparative analysis of surrogate turbulence models through various machine algorithms, confirming the feasibility of applying machine learning to industrial simulations of turbulent flows, and also provide an idea for further research on the combination of machine learning and turbulence models in the selection of algorithms.

Acknowledgments
The research was sponsored by Shuguang Program supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission in China via project 18SG53 and the National Natural Science Foundation of China project 91952102.

References
[1] Basara, B., Jakirlic, S. (2003) A new hybrid turbulence modelling strategy for industrial CFD. International journal for numerical methods in fluids, 42(1): 89-116.
[2] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011) Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12: 2825-2830.
[3] Abadi, M., Agarwal, A., Barham, P., et al. (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint, arXiv:1603.04467.
[4] Kutz, J.N. (2017) Deep learning in fluid dynamics. Journal of Fluid Mechanics, 814: 1-4.
[5] Wang, J.X., Wu, J.L., Xiao, H. (2017) Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on DNS data. Phys.rev.fluids., 2(3):1-22.
[6] Ling, J., Kurzawski, A., Templeton, J. (2016) Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. Journal of Fluid Mechanics, 807: 155-166.
[7] Maulik, R., Sharma, H., Patel, S., et al. (2019) Accelerating rans turbulence modeling using potential flow and machine learning. arXiv preprint, arXiv:1910.10878.
[8] Duraisamy, K., Iaccarino, G., Xiao, H. (2019) Turbulence modeling in the age of data. Annual Review of Fluid Mechanics, 51: 357-377.
[9] Breiman, L. (2001) Random forests. Machine learning, 45(1): 5-32.
[10] Chen, T., Guestrin, C. (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco. pp. 785–794.
[11] Altman, N.S. (1992) An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3): 175-185.
[12] Cristianini, N., Shawe-Taylor, J. (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, Cambridge.
[13] Ruder, S. (2016) An overview of gradient descent optimization algorithms. arXiv preprint, arXiv:1609.04747.
[14] Driver, D.M., Seegmiller, H.L. (1985) Features of a reattaching turbulent shear layer in divergent channelflow. AIAA journal, 23(2): 163-171.
[15] Spalart, P., Allmaras, S. (1992) A one-equation turbulence model for aerodynamic flows. In: 30th aerospace sciences meeting and exhibit. Reno. pp. 439.
[16] Weller, H.G., Tabor, G., Jasak, H., et al. (1998) A tensorial approach to computational continuum mechanics using object-oriented techniques. Computers in physics, 12(6): 620-631.
[17] Ling, J., Jones, R., Templeton, J. (2016) Machine learning strategies for systems with invariance properties. Journal of Computational Physics, 318: 22-35.
[18] Wu, J.L., Xiao, H., Paterson, E. (2018) Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. Physical Review Fluids, 3(7): 074602.

[19] Rumsey, C.L., Thomas, J.L. (2008) Application of FUN3D and CFL3D to the Third Workshop on CFD Uncertainty Analysis. NASA/TM, 2008-215537.