Comparing random forests and neural networks to augment RANS turbulence models

Pedro Stefanin Volpiani

COMPARING RANDOM FORESTS AND NEURAL NETWORKS TO AUGMENT RANS TURBULENCE MODELS

P.S. Volpiani[1]

[1] ***ONERA, The French Aerospace Lab.***

pedro.stefanin_volpiani@onera.fr

Abstract

Machine-learning (ML) techniques has bloomed in recent years, especially in fluid mechanics applications. In this paper, we trained, validated and compared two types of ML-based models to augment Reynolds-averaged Navier-Stokes (RANS) simulations. The methodology was tested in a series of flows around bumps, characterized by different levels of flow separation and curvatures. The ML-based models were trained in three configurations presenting attached flow, small and moderate separation and tested in two configurations presenting incipient and large separation. The output quantity of the machine-learning model is the turbulent eddy viscosity as done in Volpiani et al. (2022). The new models based on artificial neural networks (NN) and random forest (RF) improved the results if compared to the baseline Spalart-Allmaras model, in terms of velocity field and skin-friction profiles. We noted that NN has better extrapolation properties than RF, but the skin-friction distribution can present small oscillations when using this specific NN-based model. These oscillations can be reduced if the RF model is employed. One of the major advantages of RF is that raw quantities can be given as input features, avoiding normalization issues (such as division by zero) and allowing a larger number of universal inputs.

1 Introduction

Turbulence modeling based on artificial intelligence (AI) and machine learning (ML) has drawn a lot of interest in recent years, especially because these modern techniques were shown to be useful when applied to improve RANS models. Well-disseminated approaches consist of fixing existing models, such as the Spalart-Allmaras (SA) model, by solving an inverse problem and training an AI algorithm on a selected dataset and extrapolating to other cases that are not included in the training set. Parish and Duraisamy (2016) and Singh et al. (2017) proposed to correct the source terms in turbulence transport equations using data assimilation and machine learning. Volpiani et al. (2021) opted to introduce a correction to the Boussinesq-hypothesis by adding a forcing term in the momentum equations. They employed variational data assimilation to infer the vectorial source correction from high-fidelity numerical data and machine learning to reconstruct this quantity from the local mean-flow features. On the other hand, a different approach consists of learning directly the unknown terms in the RANS equations based on a high-fidelity training set. Ling et al. (2016) proposed to directly predict the Reynolds stress (more specifically, its deviatoric part) using machine learning. Wang et al. (2017) on the other hand, focused on the discrepancies between the exact and the RANS modeled Reynolds stresses. Cruz et al. (2019) explored a different venue; they proposed to work with the divergence of the Reynolds stress tensor, also called the Reynolds force vector, as a target for the machine learning procedure. More recently, Volpiani et al. (2022) used machine learning techniques to infer the turbulent eddy viscosity from high-fidelity simulations to correct the SA model and successfully improved RANS results of flows over bi-dimensional bumps. Despite the constraint of the Boussinesq hypothesis in the latter study, predicting a turbulent-eddy viscosity has two main advantages: first, we no longer need to transport a turbulent variable, i.e. we only need to solve for the mass and momentum equations since the problem is closed; and secondly, from a machine learning perspective, it is easier and faster to predict a scalar quantity, rather than a vector or tensor. This work is a continuation of Volpiani et al. (2022)'s study and in this paper, we compare the performance and address pros and cons of two types of supervised-learning methodologies: artificial neural networks (NN) and random forests (RF).

2 RANS equations and configuration

By using the Reynolds decomposition for the velocity $u_i = \overline{u}_i + u'_i$ and pressure $p = \overline{p} + p'$, the RANS equations for an incompressible steady flow can be written as

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0, \tag{1}$$

$$\overline{u}_i \frac{\partial \overline{u}_j}{\partial x_j} = -\frac{\partial \overline{P}}{\partial x_i} + \frac{\partial (2\nu S_{ij})}{\partial x_j} - \frac{\partial a_{ij}}{\partial x_j} \tag{2}$$

where the overbar stands for mean quantities and the prime for fluctuations. $S_{ij} = (\overline{u}_{i,j} + \overline{u}_{j,i})/2$ is the

Table 1: Summary of configurations studied in this work. The reference data was performed by Matai and Durbin (2019).

Case	Height (mm)	Characteristics	Usage
h20	20 (0.0659C)	No separation	training
h26	26 (0.0878C)	Incipient separation	testing
h31	31 (0.1032C)	Separated flow	training
h38	38 (0.1259C)	Separated flow	training
h42	42 (0.1377C)	Separated flow	testing

mean strain tensor and ν is the molecular viscosity. $\overline{P} = \overline{p} + 1/3\overline{u_i' u_i'}$ is the modified pressure and $a_{ij} = \overline{u_i' u_j'} - 1/3\overline{u_k' u_k'}\delta_{ij}$ is the deviatoric anisotropic part of the Reynolds stress tensor. In the RANS framework, common eddy-viscosity models uses the Boussinesq hypothesis and the tensor a_{ij} is approximated by $a_{ij} = -2\nu_t S_{ij}$. In this paper, the kinetic-eddy viscosity ν_t is estimated by the one equation Spalart-Allmaras (1994) turbulence model:

$$u_j \frac{\partial \widetilde{\nu}}{\partial x_j} - \nabla \cdot \left(\sigma^{-1}(\nu + \tilde{\nu}) \nabla \tilde{\nu} \right) = P_{\tilde{\nu}} - D_{\tilde{\nu}} + C_{\tilde{\nu}} \quad (3)$$

where the terms $P_{\tilde{\nu}}$, $D_{\tilde{\nu}}$ and $C_{\tilde{\nu}}$ are the production, destruction and cross-diffusion terms of the quantity $\tilde{\nu}$, and are given by:

$$P_{\tilde{\nu}} = c_{b1} \widetilde{S} \widetilde{\nu} \,,$$
$$D_{\tilde{\nu}} = c_{w1} f_w \left[\frac{\widetilde{\nu}}{d} \right]^2 \,,$$
$$C_{\tilde{\nu}} = \frac{c_{b2}}{\sigma} \frac{\partial \widetilde{\nu}}{\partial x_k} \frac{\partial \widetilde{\nu}}{\partial x_k} \,.$$

More details about the model variables and the physical definitions of each term are found in Spalart and Allmaras (1994).

We simulate the flows over a family of bidimensional bumps for which a reference dataset from Matai and Durbin (2019) is available. Large-eddy simulations were performed for five bumps heights: 20, 26, 31, 38 and 42 mm. This set of configurations is interesting because it is characterized by different levels of curvature, pressure gradient and flow separation. For the lowest bump height (h20), the flow remains attached all along the bottom wall. Case h26 presents incipient separation. The other configurations (h31, h38, h42) develop a recirculating bubble near the end of the bump and its length increases with the protuberance height. Details about the geometry and numerical conditions can be found in Matai and Durbin (2019) and Volpiani et al. (2022). A summary of simulations carried out in this study is given in Table 1.

3 Supervised-Learning techniques

Supervised learning is a machine learning paradigm for problems where the available data consists of labelled examples, meaning that each input data is associated with a known output. The goal of supervised learning algorithms is to learn a function that maps input features to labels (output). These algorithms are particularly employed to classify data or to predict outcomes accurately. Supervised learning uses a training set to teach models to predict the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through a loss function, which is adjusted until the error has been sufficiently decreased. In an ideal scenario, the algorithm is capable of estimating the correct output even for situations not present in the training phase. This requires the learning algorithm to generalize from the training data to unseen situations in a reasonable way. In this report, we employ two major techniques of supervised learning: artificial neural networks (NN) and random forest (RF).

Artificial Neural Networks is a subset of supervised learning that represents a structure of artificial neurons connected to each other. They are organized in one or multiple layers, through which information is transmitted successively from the input layer to the intermediate (hidden) layers, towards the output layer. Each node is made up of inputs, weights, a bias, and an output. To each neuron unit, it is assigned a function that represents how it will receive the information from the previous layer and transmit it to the next one, called the activation function σ. The activation outputs that come from each layer are usually treated by assigning them weights (w) and biases (b), generating a weighted input $z_i^l = \sum_j w_{ij}^l a_j^{l-1} + b_i^l$, for the i^{th} neuron at layer l, where j designates the j^{th} neuron at layer $l-1$. Thus, the activation output is given by $a_i^l = \sigma(z_i^l) = \sigma\left(\sum_j w_{ij}^l a_j^{l-1} + b_i^l\right)$. The training of a neural network is conducted by minimizing the error, given by the difference between the predicted output of the network and a correct (target) output. Successive adjustments of its weights and biases will cause the neural network to produce an output which is increasingly similar to the target output. After a sufficient number of adjustments (epochs) the training is paused based upon certain criteria. In this report, we employ the open-source Python library Pytorch to perform the training phase of our NN algorithm.

Random forest is a supervised machine learning algorithm used for classification and regression. The "forest" references a collection of uncorrelated decision trees, which are then merged together to reduce variance and create more accurate data predictions. There are several advantages associated to the RF technique: it offers a good performance when dealing with high-dimensional problems, it does not require hyperparameter tuning, it is simple to implement, and it has low computational overhead. However, we can cite a few inconveniences associated to this method as well: decision-tree learners can create over-complex

Table 2: Set #2 of non-normalized input features.

Feature	Description	Formula
q_1	Strain-rate magnitude	$\|\mathbf{S}\|$
q_2	Rotation-rate magnitude	$\|\mathbf{\Omega}\|$
q_3	SA eddy viscosity	ν_t^{SA}
q_4	SA production	$c_{b1}\widetilde{S}\widetilde{\nu}$
q_5	SA destruction	$c_{w1} f_w \left(\frac{\widetilde{\nu}}{d}\right)^2$
q_6	SA cross-diffusion	$\frac{c_{b2}}{\sigma}\frac{\partial\widetilde{\nu}}{\partial x_k}\frac{\partial\widetilde{\nu}}{\partial x_k}$
q_7	Turbulence intensity	k_{qcr}

trees that do not generalize the data well, predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated. In this study, the RF algorithm is based on the open-source Python library Scikit-Learn.

4 Input and output quantities

For the neural-network model, the input features are the same from Volpiani et al. (2022). Concerning the random forest algorithm, two sets of inputs were tested: set 1, which is the same one used in Volpiani et al. (2022) and set 2, which uses non-normalized inputs. Note that using set 2 for a NN model is not feasible, because this situation may lead to an imbalance in the input importance in the output prediction. Normalizing all features in the same range avoids this type of problem. The second choice of input features takes into consideration some philosophies and fallacies in turbulence modeling (see Spalart (2015)). For example, models should respect the rules of Galilean invariance and independence of the direction of the axes. Galilean invariance states that the laws of motion are the same in all inertial frames of reference. Therefore, in general, velocity should not be a valid entry in a model. Moreover, Spalart and Shur (1997) explain that even the derivative U_y itself is not Galilean invariant, because it is referred to axes of a reference frame, which is aligned with the velocity. Consequently, the streamline curvature itself is also an inadequate entry into a model. However, it is true that if we are dealing solely with steady flow problems, a unique reference frame can be identified and this limitation can be withdrawn. Spalart (2015) also highlights that acceleration and pressure-gradient dependence in models should be avoided, because they have no direct impact on the turbulence, and can be introduced to or removed from the equations by a simple change of reference frame. The list of inputs concerning set 2 is given in Table 2.

Since interpretability may be useful when designing a new model, two methods to shed some light in understanding the importance of each feature to predict the output were investigated: the mean decrease impurity (MDI, or Gini importance), and the mean decrease accuracy (or permutation importance). In the

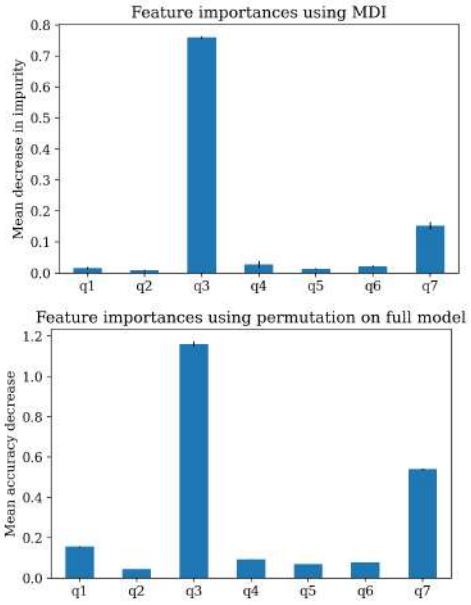

Figure 1: Feature importance concerning model RF2.

first method, each feature importance is calculated as the sum over the number of splits across all trees that include the feature, proportionally to the number of samples it splits. In the second method, we shuffle the entries of a specific variable in the test dataset and we compute the resulting increase in error. Figure 1 shows the feature importance using the MDI and permutation methods for RF2, but similar importance is observed for RF1 (not reported for brevity). A clear conclusion arises from these images: input features related to the SA eddy-viscosity, ν_t^{SA}, and the modeled turbulent kinetic energy, k_{qcr}, present great relevance in the estimation of the corrected eddy viscosity. The fact that ν_t^{SA} is the most important variable is not surprising, since it models the output quantity. The second most important quantity k_{qcr} also indicates that this quantity is of prime importance in modelling turbulence closure.

The output quantity is the eddy-viscosity estimated from the LES as done in Volpiani et al. (2022):

$$\nu_t^{LES} = \frac{\max\left(0, -a_{ij}\,\partial_j\overline{u_i}\right)}{\max\left(0, 2S_{ij}S_{ij}\right) + \epsilon} \tag{4}$$

where ϵ is a small parameter. Figure 2 shows the normalized eddy-viscosity fields coming from the baseline SA model, the reference simulation, the NN, RF1 and RF2 models for both extreme cases: h20, which presents no separation and belongs to the training set and h42, which presents large separation and belongs to the testing set. For case h20, the SA model tends to overpredict the eddy viscosity above and in the rear of the bump. We would like to emphasize that predicting the correct amount of ν_t for all configurations is not easy, and our goal is to use machine-learning algorithm to help in this task. We note that the NN model manages to reproduce the correct levels of tur-

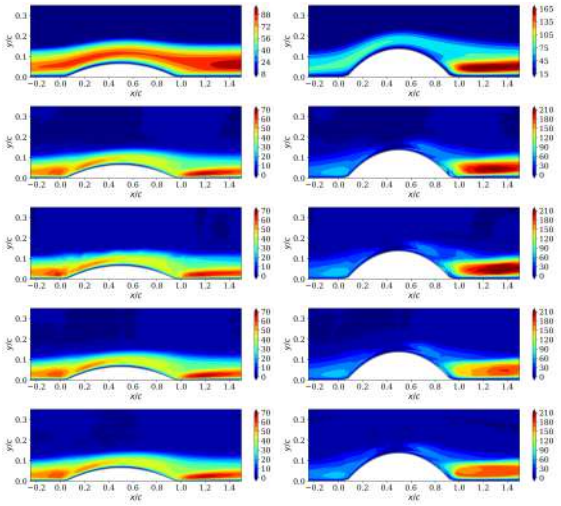

Figure 2: Normalized turbulent viscosity ν_t/ν computed from the SA, LES, NN, RF1 and RF2 models (from top to bottom), case h20 (left) and h42 (right).

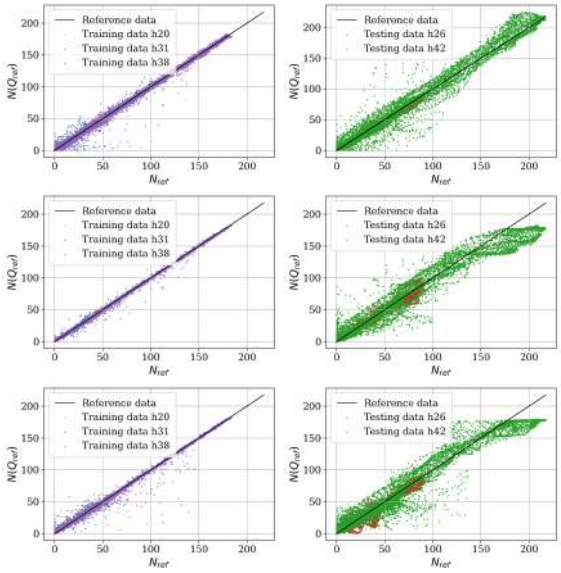

Figure 3: Normalized output quantities from NN (top), RF1 (middle) and RF2 (bottom). The scatter points should approximate to the solid line plotted as reference.

bulent eddy viscosity, despite some fluctuations in the frontier of the free stream. Models RF1 and RF2 present similar predictions. For case h42, the traditional SA model underpredicts the eddy viscosity in the boundary-layer recovery region. The NN model predicts precisely the eddy-viscosity field. However, it is possible to note oscillations after the bump and close to the wall region which can contribute to a noisy RANS solution. The RF models do not present such oscillatory behavior. For this test case, RF2 is superior than RF1. The drawback of the RF models is that the maximum value of the output quantity is bounded by the maximum value present in the training set, indicating that the RF method should be used with caution when dealing with extrapolations. These conclusions are also supported by figure 3 that plots the output of the ML models (NN, RF1 and RF2) as a function of the expected quantity. If the model works, the scatter points should approximate to the solid line plotted as reference. The NN model manages to predict a more realistic trend overall, despite its oscillatory behavior observed in figure 2. On the other hand, RF models are more stable, they predict extremely well the training set, but quantitatively are less precise than NN when extrapolating. In the next section, we present a posteriori RANS results obtained with the NN and RF models.

5 A posteriori results

The new ML assisted models were trained in three configurations: cases h20 (attached flow), h31 (small separation), and h38 (moderate separation); and tested in cases h26 (incipient separation) and h42 (large separation). This setup allows us to evaluate the new ML models in scenarios of interpolation and extrapolation. Figure 4 (top) displays the skin-friction coefficients for testing cases h26 and h42 using the NN1 model. Results are in agreement with the ones from Volpiani et al. (2022). We note a considerable improvement when predicting the skin-friction distribution. Conversely, we observe significant noise in the near wall region affecting the C_f profiles. For instance, there is a jump in the solution, in the region $0.7 < x/c < 0.8$, that is non-physical. This noise is particularly present in the first boundary cells and is not a generalized behavior. The wall pressure distribution presents a smooth signal (not shown for brevity). Figure 4 (middle) shows the same quantities using the RF1 model. Two conclusions stand out from this graphic: i) the first is that the C_f profiles present less oscillations than in the NN situation; and (ii) the second is that the results are closer to the reference LES results. At least for the skin-friction distribution, the RF model seems to be less sensitive than the NN one. Contrarily to NN, non-dimensionalized input features can be fed to a RF model. This means that more variables can be used to train the model and using crude quantities avoids divisions by small numbers, helping to improve the prediction capabilities. This time, we choose features that respect the recommendations from Spalart (2015) as shown in Table 2. In figure 4 (bottom), we compare the C_f profiles at the lower wall for the reference LES, baseline RANS-SA and the new RANS-RF2 simulation using non-dimensionalized input features. We note that the skin-friction profiles are smoother than in the previous cases (NN1 and RF1), confirming that the new inputs improve the learning process. Globally, the ML-based model manages to predict two types of flow conditions: an attached flow for case h26 and a separated flow for case h42. However, it is impressive

the good prediction of skin friction using RF2, especially for the extrapolation scenario h42, which was proven in Volpiani et al. (2022) to be a case where the eddy-viscosity formula is not precise.

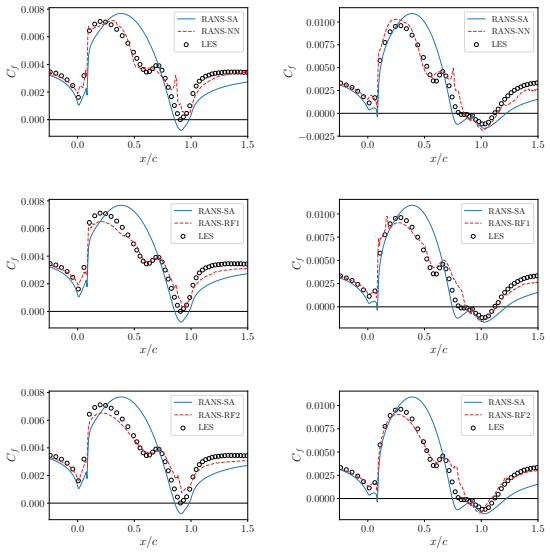

Figure 4: Skin friction distribution for cases h20 (left) and h42 (right).

Now we focus our attention in the velocity field as a whole. In Figure 5, we plot the error

$$e = \left[\left(\frac{\overline{u}_{RANS} - \overline{u}_{LES}}{u_\infty} \right)^2 + \left(\frac{\overline{v}_{RANS} - \overline{v}_{LES}}{u_\infty} \right)^2 \right]^{1/2}$$

given by the difference between the reference and modeled velocities for testing cases h26 and h42. We note that, in the baseline simulation, the error is concentrated in the boundary-layer region, especially after the bump. The error in the RANS simulation increases if we increase the hump height. For case h26, that presents incipient separation, the SA model does a good job predicting the overall flow field, except very close to the wall. The ML models increase the accuracy especially on top of the bump, but generally the results are very similar to the SA model. The NN might have a slightly advantage in the diverging section around $x/c \approx 1.0$ and the RF in the boundary-layer recovery region ($x/c > 1.1$), but the results are comparable. Concerning case h42, that presents large flow separation, the discrepancy between LES and RANS are more flagrant. The baseline SA simulation fails in the boundary-layer region and in a more extended region after the bump. The ML model that best corrects the velocity field is the NN-based one, but there is still a region around $0.8 < x/c < 1.0$ that it misses precision. Model RF1 presents a similar behavior than the NN-based one but the error is slightly amplified in the same region. The same is observed for model RF2. It was shown that the ML methods

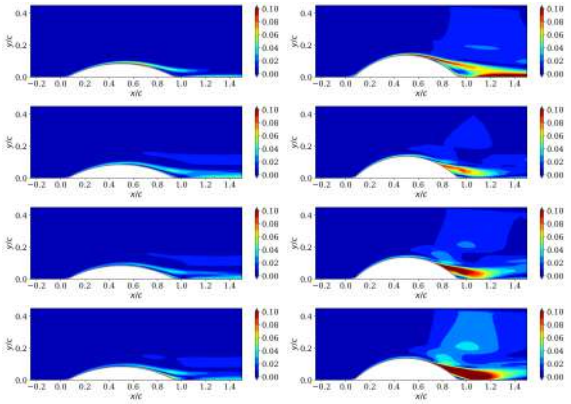

Figure 5: Velocity error computed for the baseline SA, NN, RF1 and RF2 models (from top to bottom), case h26 (left) and h42 (right).

manage to learn our key quantity. However, it is important to note that the error given by the ML models can also come from the strong approximation made to compute the turbulence eddy viscosity Eq. (4), which is known to be inaccurate in flows presenting separation. Therefore, a possible solution to correct the full resulting flow field would be to improve the estimate of ν_t through data-assimilation for example. On the other hand, if the CFD engineer is only interested in the skin-friction distribution, then approximation (4) and the ML models presented herein are sufficient.

6 Conclusions

In this paper, we compared two types of supervised-learning methodologies to correct RANS simulations of flows over a family of bumps. The ML-based models were trained in three configurations presenting attached flow, small and moderate separation and curvature (cases h20, h31 and h38 respectively) and tested in two configurations presenting incipient and large separation (cases h26 and h42 respectively). The new models based on artificial neural networks and random forest improved considerably the results if compared to the baseline SA model, in terms of velocity field and skin-friction profiles. One of the goals of the paper was to investigate which method outperforms the other, in other words, which method is better suited to augment RANS turbulence models. We concluded that each strategy has its pros and cons, which need to be taken into account when developing a data-driven model. We highlight that the conclusion here does not only apply to RANS models, but can also be generalized to other fields. We learned that NN are more efficient in interpolating and extrapolating the output quantity than RF. However, this technique when used to predict the turbulence-eddy viscosity can display some oscillatory behavior especially close to the wall boundaries (noticed by skin-friction profiles). One way to overcome this issue is to keep the turbulent-variable transport equation

and try to correct a term in this equation (as done in Parish and Duraisamy (2016) or Singh et a. (2017) through data assimilation for example). The advantage of the present method is that all the modelling is embedded in the turbulence-eddy viscosity given by the ML-based model. This is an extremely simple and straightforward way (both in terms of physical modelling and numerical implementation) to correct a RANS simulation. The oscillations in the skin-friction profiles are much reduced if RF are employed. Moreover, the RF method does a good job learning the training cases. However, it was shown that RF do not extrapolate well to configurations unseen during the training process. Nonetheless, the results obtained with the RF model for case h42 are still in excellent agreement with the reference data. One of the high-points of the RF method, in the author's opinion, is the fact that non-normalized inputs can be fed to the RF algorithm, contrarily to NN. The set 2 of inputs were derived based on a more generic framework that can be used in mixed configurations. ML-based turbulence models are still in early stages of development if compared to their traditional RANS model counterparts that went through decades of adjustments and tunning. However, as time goes by, more and more reference data will become available and taking into consideration this additional information in a model seems natural. So, developing a data-driven model that gives outstanding performance for flows around stator blades is good, but a model that deals with both stator and rotor blades is better. This work laid an additional brick in the development of a more general data-driven turbulence model.

Acknowledgments

This work was funded by ONERA under projects MODDA and MASSIPH.

References

Cruz, M.A., Thompson, R.L., Sampaio, L.E. and Bacchi, R.D. (2019). The use of the Reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling. *Comput. Fluids*, Vol. 192, pp. 104258.

Matai, R. and Durbin, P. (2019). Large-eddy simulation of turbulent flow over a parametric set of bumps. *J. Fluid Mech.*, Vol. 866, pp. 503-525.

Ling, J., Kurzawski, A. and Templeton, J. (2016), Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.*, Vol. 807, pp. 155-166.

Parish, E.J. and Duraisamy, K. (2016). A paradigm for data-driven predictive modeling using field inversion and machine learning. *J. Comput. Phys.*, 305, 758-774.

Volpiani, P.S., et al. (2021), Machine learning-augmented turbulence modeling for RANS simulations of massively separated flows. *Phys. Rev. Fluids*, Vol. 6, pp. 064607.

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

Singh, A.P., Shivaji, M. and Karthik, D. (2017), Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA J.*, Vol. 55, pp. 2215-2227.

Spalart, P. R. and Allmaras, S. R. (1994), A One-Equation Trubulence Model for Aerodynamic Flows. *Rech. Aerosp.*, Vol. 1, pp. 5-21.

Spalart, P. R. and Shur, M. (1997). On the sensitization of turbulence models to rotation and curvature. *Aerosp. Sci. Technol.*, Vol. 1, pp. 297-302.

Spalart, P.R. (2015). Philosophies and fallacies in turbulence modeling. *Prog. Aerosp. Sci.*, Vol. 74, pp. 1-15.

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