Oral presentation | Data science and AI
Data science and AI-II
Thu. Jul 18, 2024 10:45 AM - 12:45 PM Room C

[10-C-03] *Improving RANS turbulence models using random forests and neural networks*

*Pedro Stefanin Volpiani[1] （1. ONERA）

Keywords: RANS modeling, machine learning, random forests, neural networks

Twelfth International Conference on
Computational Fluid Dynamics (ICCFD12),
Kobe, Japan, July 14-19, 2024

Improving RANS turbulence models using random forests and neural networks

Pedro Stefanin Volpiani*
Corresponding author: pedro.stefanin_volpiani@onera.fr
* ONERA, France.

Abstract: Machine-learning (ML) techniques have 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 flows around bumps. The ML-based models were trained in four configurations presenting attached flow, small and moderate separations and tested in a configuration presenting large separation. The output quantity of the machine-learning model is the normalized turbulent viscosity as done in [1]. 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 certain input features. These oscillations can be reduced if the RF model is employed. One 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. At the end, we propose a mixed NN-RF model that combines the strengths of each method and, as a result, improves considerably the RANS prediction capability, even for a case with strong separation where the Boussinesq hypothesis (and therefore the eddy-viscosity assumption) lacks accuracy.

Keywords: RANS modeling, machine learning, random forests, neural networks.

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 can be powerful when applied to improve Reynolds-averaged Navier-Stokes (RANS) models [2, 3, 4]. 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 [5] and Singh et al. [6] proposed to correct the source terms in turbulence transport equations using data assimilation and machine learning. Volpiani et al. [7] 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.

A different approach consists of learning directly the unknown terms in the RANS equations based on a high-fidelity training set. Many authors proposed to directly predict the Reynolds stress (more specifically, its deviatoric part) using machine learning [8, 9, 10, 11]. Others focused on the discrepancies between the exact and the RANS modeled Reynolds stresses [12, 13, 14] or even on the divergence of the Reynolds stress tensor, also called the Reynolds force vector, as a target for the machine learning procedure [15, 16]. More recently, Volpiani et al. [1] used machine-learning techniques to infer the 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 turbulence-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 a tensor (especially if we consider three-dimensional configurations).

From the list of works mentioned above, the majority of them employed artificial neural networks to model the key quantity. Fewer studies focused on random forest as the ML strategy. This work is a continuation of Volpiani et al. [1]'s study and in this paper, we compare the performance and address

Table 1: Summary of configurations studied in this work. The reference data was performed by [20].

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

pros and cons of two types of supervised-learning methodologies: artificial neural networks (NN) and random forests (RF). Careful attention is also drawn to the choice of input features used by the ML algorithm in order to have a more generic model for RANS computations. The ultimate goal is to try to answer the question: "Are random forests better suited than neural networks to augment RANS turbulence models?". The paper is organized as follows: in 2, we present the RANS equations and the configuration we simulate; in 3, we expose the two types of supervised-learning strategies used in this work: NN and RF; section 4 discusses the list of inputs and outputs considered in this study; in 5, a posteriori results are presented using different implementations of the ML-based model and we address the limitations of each method; in 6, an improved ML-based model is proposed and we finally end with a conclusion. Note that most of the results and methods presented in this paper have been published in [17].

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 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 use 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 turbulence model [18]:

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

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:

$$P_{\tilde{\nu}} = c_{b1} \widetilde{S} \widetilde{\nu}, \quad D_{\tilde{\nu}} = c_{w1} f_w \left[\frac{\widetilde{\nu}}{d} \right]^2, \quad C_{\tilde{\nu}} = \frac{c_{b2}}{\sigma} \frac{\partial \widetilde{\nu}}{\partial x_k} \frac{\partial \widetilde{\nu}}{\partial x_k} . \tag{4}$$

More details about the model variables and the physical definitions of each term are found in ref. [18]. Numerical implementation in the finite-element software FreeFEM was done following [19].

We simulate the flows over a family of bidimensional bumps for which a reference dataset from Matai and Durbin [20] is available. Large-eddy simulations were performed for five bump heights: 20, 26, 31, 38 and 42 mm (see figure 1). 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.

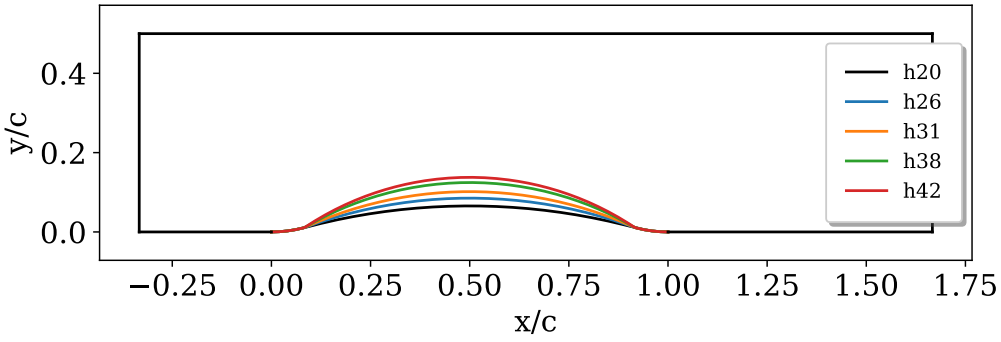

Figure 1: Computational domain for the flows over the family of bumps. The baseline geometry is shown in black.

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 hyper-parameter 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 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.

Table 2: Set #1 of local input features.

Feature	Description	Formula
q_1	Q-criterion	$\frac{\left\|\mathbf{\Omega}\right\|^2 - \left\|\mathbf{S}\right\|^2}{\left\|\mathbf{\Omega}\right\|^2 + \left\|\mathbf{S}\right\|^2}$
q_2	Ratio of pressure normal stresses to shear stresses	$\frac{\sqrt{\frac{\partial \overline{P}}{\partial x_i}\frac{\partial \overline{P}}{\partial x_i}}}{\sqrt{\frac{\partial \overline{P}}{\partial x_j}\frac{\partial \overline{P}}{\partial x_j}} + \frac{1}{2}\frac{\partial \bar{u}_k^2}{\partial x_k}}$
q_3	Modified [24] marker	$\frac{\left\|\bar{u}_k\bar{u}_l\frac{\partial \bar{u}_k}{\partial x_l}\right\|}{\left\|\bar{u}_i\bar{u}_j\frac{\partial \bar{u}_i}{\partial x_j}\right\| + \sqrt{\bar{u}_n\bar{u}_n\bar{u}_i\frac{\partial \bar{u}_i}{\partial x_j}\bar{u}_m\frac{\partial \bar{u}_m}{\partial x_j}}}$
q_4	Streamline pressure gradient	$\frac{\bar{u}_k\frac{\partial \overline{P}}{\partial x_k}}{\left\|\bar{u}_l\frac{\partial \overline{P}}{\partial x_l}\right\| + \sqrt{\frac{\partial \overline{P}}{\partial x_j}\frac{\partial \overline{P}}{\partial x_j}\bar{u}_i\bar{u}_i}}$
q_5	Viscosity ratio	$\frac{\nu_t}{\nu_t + 100\nu}$
q_6	SA ratio of production to destruction	$\frac{c_{b1}\widetilde{S}\widetilde{\nu}}{\left\|c_{b1}\widetilde{S}\widetilde{\nu}\right\| + c_{w1}f_w\left(\frac{\widetilde{\nu}}{d}\right)^2}$
q_7	SA ratio of production to diffusion	$\frac{c_{b1}\widetilde{S}\widetilde{\nu}}{\left\|c_{b1}\widetilde{S}\widetilde{\nu}\right\| + \frac{c_{b2}}{\sigma}\frac{\partial \widetilde{\nu}}{\partial x_k}\frac{\partial \widetilde{\nu}}{\partial x_k}}$
q_8	Turbulence intensity	$\frac{k_{qcr}}{k_{qcr} + \frac{1}{2}\bar{u}_i^2}$

4 Input and output features

For the neural-network model, the input features are the same from [1] and they are summarized in Table 2. They were inspired by [21, 12, 7]. Concerning the random forest algorithm, two sets of inputs were tested: set 1, which is the same one used in [1] and set 2, which uses non-normalized inputs and is more generic. 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 [22]. 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 [23] 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 [22] 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 3.

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 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. Figures 2 and 3 show the feature importance using the MDI and permutation methods for RF1 and RF2, respectively. A clear conclusion arises from these images: input features related to the SA eddy-viscosity, ν_t^{SA}, and the modeled turbulence 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):

Table 3: 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})	$\frac{3}{2}c_{cr2}\nu_t\sqrt{2S_{ij}S_{ij}}$

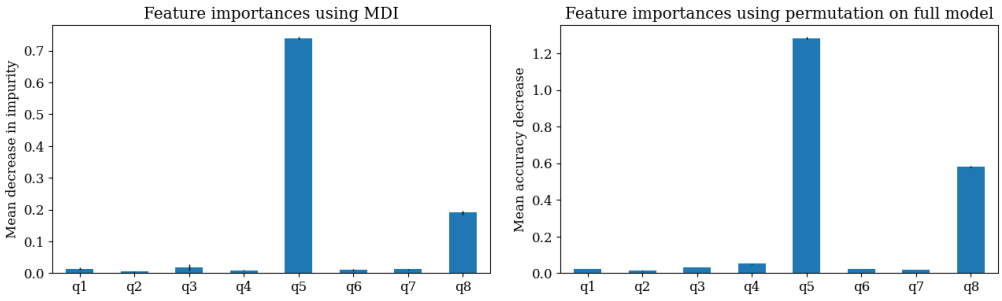

Figure 2: Feature importance concerning model RF1.

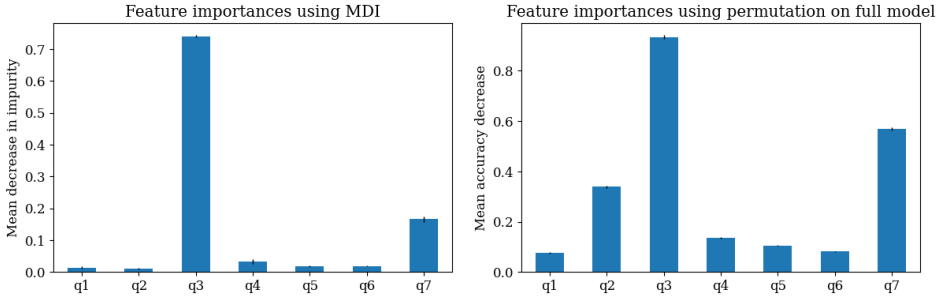

Figure 3: Feature importance concerning model RF2.

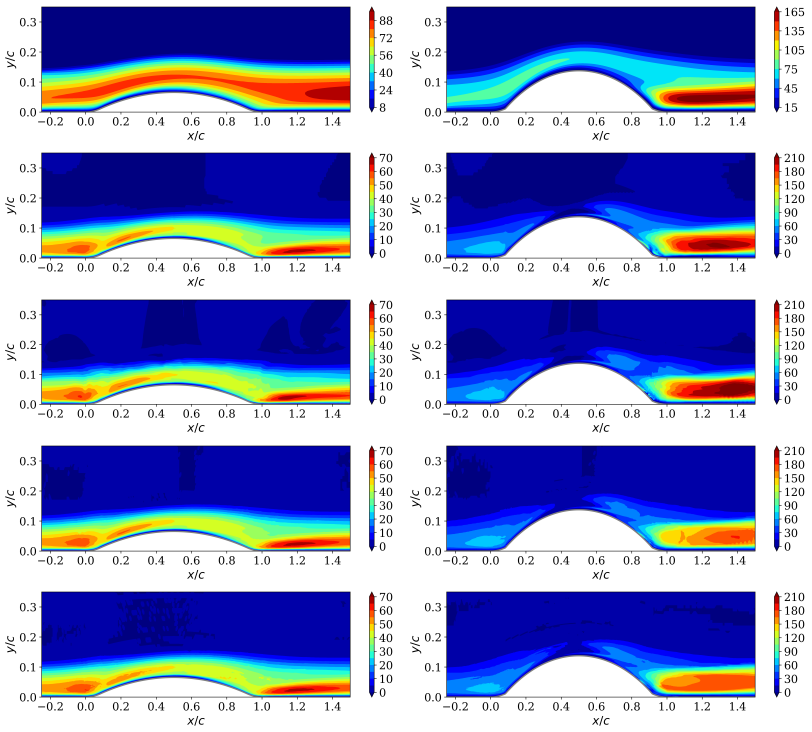

Figure 4: Normalized turbulent viscosity ν_t/ν computed from the SA, LES, NN, RF1 and RF2 models (from top to bottom), case h20 (left) and h42 (right).

$$\nu_t^{LES} = \frac{\max\left(0, -a_{ij}\ \partial_j \overline{u_i}\right)}{\max\left(0, 2S_{ij}S_{ij}\right) + \epsilon} \tag{5}$$

where ϵ is a small parameter. Figure 4 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 turbulent 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 5 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 4. 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 four configurations: cases h20 (attached flow), 26 (incipient separation), h31 (small separation), and h38 (moderate separation); and tested in case h42 (large separation). This setup allows us to evaluate the new ML models in scenarios of extrapolation. Figure

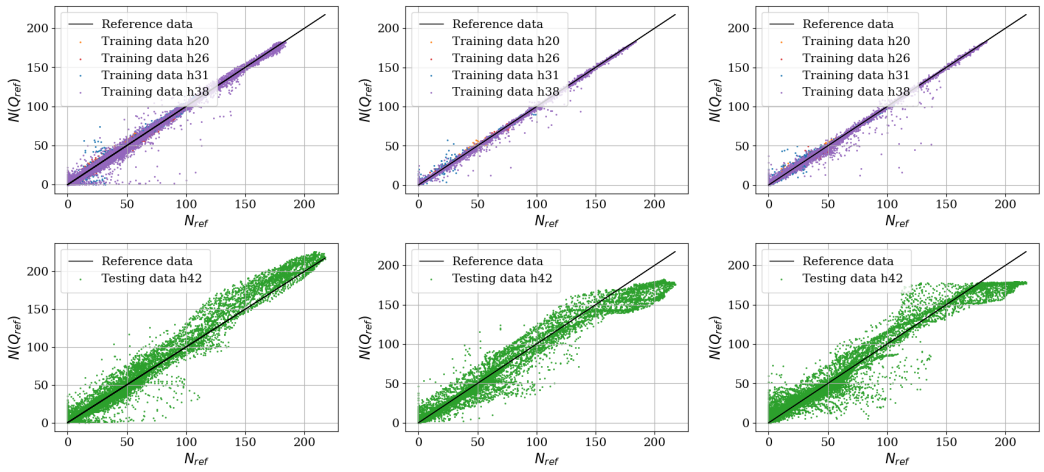

Figure 5: Normalized output quantities from NN (left), RF1 (middle) and RF2 (right). The scatter points should approximate to the solid line plotted as reference.

6 (top) displays the wall-pressure and the skin-friction coefficients for testing case h42 using the NN1 model. We note a considerable improvement when predicting the skin-friction distribution when using the new ML-based model. Conversely, we observe significant noise in the near wall region affecting the C_f profiles. This noise is particularly present in the first boundary cells and is not a generalized behavior. The wall pressure distribution presents a smooth signal. Figure 6 (middle and bottom) shows the same quantities using the RF1 and RF2 model respectively. 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. On the other hand, the RF2 model degrades the pressure distribution in the separation zone. Globally, the ML-based model manages to augment the RANS prediction, even for a case in which the eddy-viscosity formula is known to lack accuracy [1].

Now we focus our attention in the velocity field as a whole. In Figure 7, we plot the error

$$e(\mathbf{x}) = \left[\left(\frac{\overline{u}_{RANS}(\mathbf{x}) - \overline{u}_{LES}(\mathbf{x})}{u_\infty} \right)^2 + \left(\frac{\overline{v}_{RANS}(\mathbf{x}) - \overline{v}_{LES}(\mathbf{x})}{u_\infty} \right)^2 \right]^{1/2} \tag{6}$$

given by the difference between the reference and modeled velocities for testing case h42. We note that, in the baseline simulation, the error is concentrated in the boundary-layer region, especially after the bump. The discrepancy between LES and RANS for case h42 is 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 $0.8 < x/c < 1.0$ where it lacks precision. Model RF1 presents a similar behavior than the NN-based one but the error is slightly amplified in the same region. The error given by RF2 is similar than RF1 and for brevity is not shown. It is important to note that the error given by the ML models can also come from other factors: (i) it could be due to the choice of input features, and/or (ii) from the strong approximation made to compute the turbulence eddy viscosity Eq. (5), 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 (5) and the ML models presented herein are sufficient. In the next section, we propose an alternative to improve the capabilities of ML-based models.

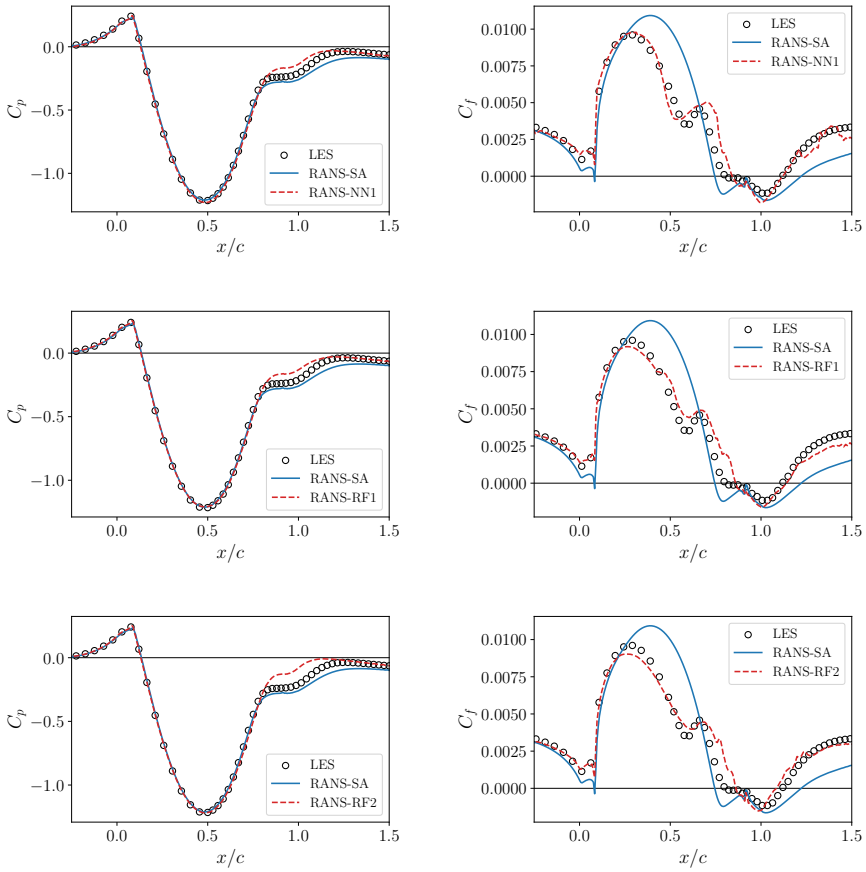

Figure 6: Skin-friction distribution for case h42.

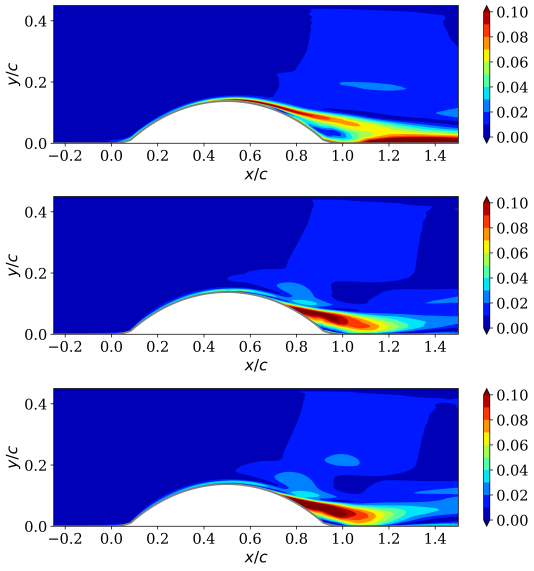

Figure 7: Velocity error computed for the baseline SA, NN and RF1 models (from top to bottom), case 42.

Table 4: Set #3 of local input features.

Feature	Description	Formula
q_1	Q-criterion	$\frac{\|\mathbf{\Omega}\|^2 - \|\mathbf{S}\|^2}{\|\mathbf{\Omega}\|^2 + \|\mathbf{S}\|^2}$
q_2	f_d function from [25]	$1 - \tanh((8r_d)^3)$
q_3	$f_d^{'}$ function from [26]	$1 - \tanh((r_d)^{0.5})$
q_4	Boundary-layer variable	$\frac{1}{1+r_d}$
q_5	Viscosity ratio	$\frac{\nu_t}{\nu_t + 100\nu}$
q_6	SA ratio of production to destruction	$\frac{c_{b1}\widetilde{S}\widetilde{\nu}}{\left\|c_{b1}\widetilde{S}\widetilde{\nu}\right\| + c_{w1}f_w\left(\frac{\widetilde{\nu}}{d}\right)^2}$
q_7	SA ratio of production to diffusion	$\frac{c_{b1}\widetilde{S}\widetilde{\nu}}{\left\|c_{b1}\widetilde{S}\widetilde{\nu}\right\| + \frac{c_{b2}}{\sigma}\frac{\partial\widetilde{\nu}}{\partial x_k}\frac{\partial\widetilde{\nu}}{\partial x_k}}$
q_8	Turbulence intensity	$\frac{k_{qcr}}{k_{qcr} + \frac{1}{2}\bar{u}_i^2}$

6 Mixed model

In the last sections, we saw that NN are more efficient in extrapolating the output quantity than RF. The eddy-viscosity field predicted by the NN for case h42, for instance, was closer to reality than the one given by the RF algorithm. On the other hand, the NN-based model presented non-physical oscillations near the wall when analyzing the skin-friction profiles. In the near-wall region, the RF seemed to agree better with the reference LES solution. Therefore, it is natural to think of a new model that combines NN and RF and that incorporates the advantages of each method. To design this new model, it is important to define a new set of input features that can be used by both algorithms. The new set 3 is composed of normalized features. The first feature is the ratio of excess rotation rate to strain rate (Q-criterion). We included the f_d-function used by [6] and originally proposed by [25] in the framework of Detached Eddy Simulations, a modified $f_d^{'}$ quantity proposed by [26] and a new quantity based on $r_d = (\widetilde{\nu} + \nu)/(\sqrt{u_{i,j}u_{i,j}}\kappa^2 d^2)$. In this expression, $\widetilde{\nu}$ is the modified eddy viscosity, ν the molecular viscosity, $u_{i,j}$ the velocity gradients, κ the Kármán constant, and d the distance to the wall. We kept the ratio of turbulence and laminar viscosities, the SA ratio of production to destruction, the SA ratio of production to diffusion and the turbulence intensity based on $k_{qcr} = \frac{3}{2}c_{cr2}\nu_t\sqrt{2S_{ij}S_{ij}}$ from [27]. The normalized turbulence intensity (quantity q_8 in 4) is not Galilean invariant. At the same time, the turbulence kinetic energy is a statistical quantity, making it adequate to be used in the context of steady RANS computations. In the last sections, we showed that k_{qcr} was an important feature, and therefore we kept it in the new list of input features. Input features of set 3 are summarized in Table 4. To blend our NN and RF models we use a blending function that is based on r_d:

$$\nu_t^{new} = [\tanh(\alpha r_d^{\beta})]\nu_t^{RF} + [1 - \tanh(\alpha r_d^{\beta})]\nu_t^{NN} \ . \tag{7}$$

Parameter r_d defines the boundary layer region. It equals 1 in the logarithmic layer, and falls to 0 gradually towards the edge of the boundary layer. Therefore, Equation (7) favors the RF model close to the wall and the NN model far away from it. The shape of the blending function can be regulated by adjusting the parameters α and β. Here, both are kept equal to unity.

Figure 8 shows the distribution of the pressure and skin-friction coefficients at the wall for simulations based on the NN alone (NN3), the RF alone (RF3) and the mixed model (NNRF). We observe that the pressure distribution is well predicted for simulation RANS-NN3. The skin friction was also much improved. The strong oscillations previously observed disappeared, indicating that the set of inputs 3 improves the solution. With respect to simulation RANS-RF3, the C_f is well captured, but C_p is slightly overpredicted in the separated region. The mixed RANS-NNRF model, on the other hand, manages to improve consistently both skin-friction and pressure distribution if compared to the baseline SA model.

To further the analysis, we plot in Figure 9 the velocity error between the reference solution and the

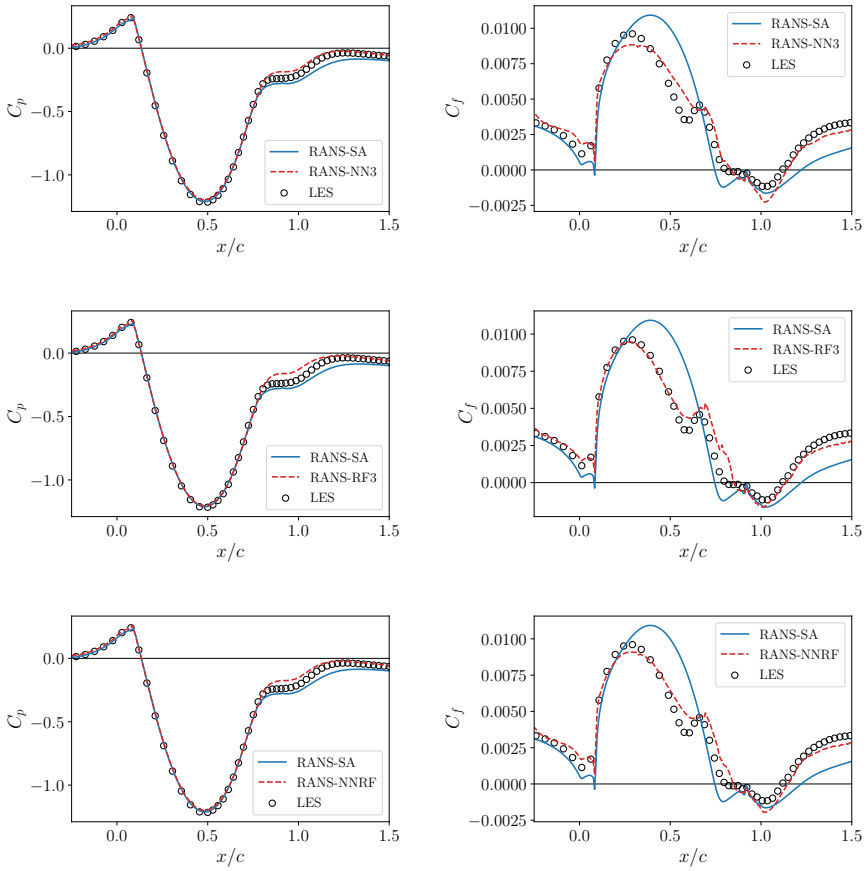

Figure 8: Pressure and skin-friction distribution (right) for testing case h42.

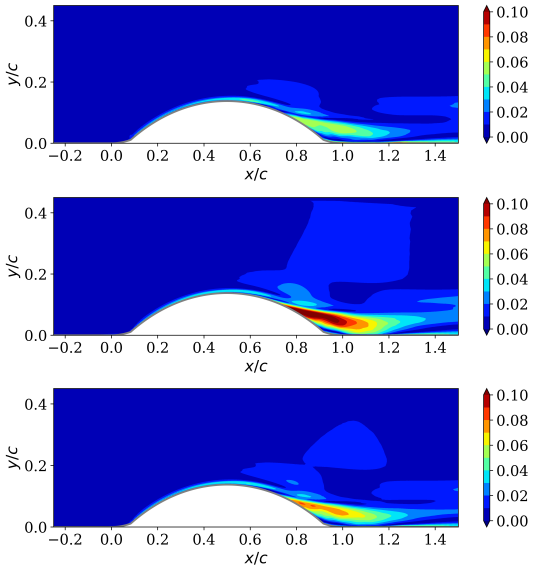

Figure 9: Velocity error computed for the NN3 (top), RF3 (middle) and mixed NNRF (bottom) models for case h42.

ones obtained using models NN3, RF3 and NNRF. We observe that the error indicator was drastically reduced for simulation using NN3 (and consequently the mixed NNRF) model if compared to the ones presented in Figure 7. The error did not change much for case based on the RF algorithm. Therefore, we conclude that the new set of input features helps to enhance the neural-network predictions. The mixed eddy-viscosity formulation using the blending function (7) takes the best of each ML method: close to the wall boundary, the RF output dominates and far away from it the NN output does. Therefore, we managed to create a new mixed RANS-NNRF model that takes advantages of each method, where it is supposed to. The new mixed model is more universal, thanks to the new set of inputs, and outperforms our previous model [1] in predicting the real mean quantities. Although not shown, we point out that the new mixed model has good performances even when estimating the eddy-viscosity discrepancy, $\Delta\nu_t$, which was discarded in the previous study.

Table 5 compares the errors between the baseline and the new ML-based models in terms of C_p and C_f distributions. If the error E_{Cp} or E_{Cf} is less than unity, it indicates an improvement in predicting these important quantities. Globally, the results are improved if compared to the baseline one. The only exception is the RF2 model, which managed to improve considerably the skin-friction distribution but degraded the pressure distribution in the region of separation. Models RF3 and NN3 both improved RANS predictions. The advantage of the mixed model is that it improves both the C_p distribution with respect to model RF3 and the C_f distribution with respect to model NN3.

Table 5: C_p error computed as $E_{Cp} = \int (C_p^{ML} - C_p^{LES})^2 dx / \int (C_p^{SA} - C_p^{LES})^2 dx$ and C_f error computed as $E_{Cf} = \int (C_f^{ML} - C_f^{LES})^2 dx / \int (C_f^{SA} - C_f^{LES})^2 dx$ for case h42.

Model	NN1	RF1	RF2	RF3	NN3	NNRF
E_{Cp}	0.742	0.737	1.896	0.833	0.460	0.574
E_{Cf}	0.668	0.673	0.589	0.668	0.674	0.671

Note that the objective of the models generated in this study was to accurately predict the flow field around bumps. So, what happens if we apply the model on a different flow configuration? We tried to employ the new ML-based model on the smooth backward-facing step simulated by [28] using DNS and LES and by [29] using RANS. The result was a considerable degradation of the baseline RANS model even on regions where it was accurate. Therefore, it is extremely important to keep in mind the limits of validity of your new data-driven model. If the configuration is too far away from the training scenario, we have high chances that it is not going to perform well. Machine-learning is not miraculous, and we need to understand that we should try to predict only the types of flows learned during the training phase. So, in this case, we do not have (and we are not aiming at having) a general model. Nevertheless, a model designed for a specific need is also valuable. During the initial design phase and optimization process of a wing, for example, the flow conditions (Mach and Reynolds numbers) and the type of flow do not change; only the geometry does. The methodology developed here could be used to treat this type of problem. Thus, a data-driven model should be built having in mind the final application.

7 Conclusion and perspectives

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 four configurations presenting attached flow, incipient, small and moderate separation and curvature (cases h20, h26, h31 and h38 respectively) and tested in a configuration presenting large separation (case h42). 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). We saw that the noise can be reduced by playing

with the input features. Another way to overcome this issue is by taking advantage of the RF method. We saw that this 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 highpoints of the RF method is the fact that non-normalized inputs can be fed to the RF algorithm, contrarily to NN. A new set of inputs was also derived based on a more generic ML framework. At the end, instead of answering which method is better, we proposed to combine them to take the advantages of each method where needed. We generated a new ML model based on both approaches (NN and RF). We showed that the new NNRF model was able to improve the pressure and skin-friction profiles and the velocity field with respect to the baseline SA model and other ML models already published in the literature. 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.

Acknowledgments

This work was supported by the French National Research Agency under project ANR-22-FAI2-0002-01 and by ONERA under projects MODDA and MASSIPH.

References

[1] Pedro Stefanin Volpiani, Raphaella Fusita Bernardini, and Lucas Franceschini. Neural network-based eddy-viscosity correction for rans simulations of flows over bi-dimensional bumps. *International Journal of Heat and Fluid Flow*, 97:109034, 2022.

[2] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508, 2020.

[3] Karthik Duraisamy. Perspectives on machine learning-augmented Reynolds-averaged and large eddy simulation models of turbulence. *Physical Review Fluids*, 6(5):050504, 2021.

[4] Richard D Sandberg and Yaomin Zhao. Machine-learning for turbulence and heat-flux model development: A review of challenges associated with distinct physical phenomena and progress to date. *International Journal of Heat and Fluid Flow*, 95:108983, 2022.

[5] Eric J Parish and Karthik Duraisamy. A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, 305:758–774, 2016.

[6] Anand Pratap Singh, Shivaji Medida, and Karthik Duraisamy. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA Journal*, pages 2215–2227, 2017.

[7] Pedro Stefanin Volpiani, Morten Meyer, Lucas Franceschini, Julien Dandois, Florent Renac, Emeric Martin, Olivier Marquet, and Denis Sipp. Machine learning-augmented turbulence modeling for RANS simulations of massively separated flows. *Physical Review Fluids*, 6(6):064607, 2021.

[8] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

[9] Jack Weatheritt and Richard Sandberg. A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship. *Journal of Computational Physics*, 325:22–37, 2016.

[10] Mikael LA Kaandorp and Richard P Dwight. Data-driven modelling of the reynolds stress tensor using random forests with invariance. *Computers & Fluids*, 202:104497, 2020.

[11] Chao Jiang, Ricardo Vinuesa, Ruilin Chen, Junyi Mi, Shujin Laima, and Hui Li. An interpretable framework of data-driven turbulence modeling using deep neural networks. *Physics of Fluids*, 33(5):055133, 2021.

[12] Jian-Xun Wang, Jin-Long Wu, and Heng Xiao. Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. *Physical Review Fluids*, 2(3):034603, 2017.

[13] Jin-Long Wu, Heng Xiao, and Eric Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Physical Review Fluids*, 3(7):074602, 2018.

[14] Yuhui Yin, Pu Yang, Yufei Zhang, Haixin Chen, and Song Fu. Feature selection and processing of turbulence modeling based on an artificial neural network. *Physics of Fluids*, 32(10):105117, 2020.

[15] Matheus A Cruz, Roney L Thompson, Luiz EB Sampaio, and Raphael DA Bacchi. The use of the Reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling. *Computers & Fluids*, 192:104258, 2019.

[16] Stefano Berrone and Davide Oberto. An invariances-preserving vector basis neural network for the closure of reynolds-averaged navier–stokes equations by the divergence of the reynolds stress tensor. *Physics of Fluids*, 34(9):095136, 2022.

[17] Pedro Stefanin Volpiani. Are random forests better suited than neural networks to augment rans turbulence models? *International Journal of Heat and Fluid Flow*, 107:109348, 2024.

[18] Philippe Spalart and Steven Allmaras. A one-equation turbulence model for aerodynamic flows. *La Recherche Aérospatiale*, (1):5–21, 1994.

[19] Arthur Shiniti Cato, Pedro Stefanin Volpiani, Vincent Mons, Olivier Marquet, and Denis Sipp. Comparison of different data-assimilation approaches to augment rans turbulence models. 2023.

[20] Racheet Matai and Paul Durbin. Large-eddy simulation of turbulent flow over a parametric set of bumps. *Journal of Fluid Mechanics*, 866:503–525, 2019.

[21] Julia Ling and J Templeton. Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty. *Physics of Fluids*, 27(8):085103, 2015.

[22] Philippe R Spalart. Philosophies and fallacies in turbulence modeling. *Progress in Aerospace Sciences*, 74:1–15, 2015.

[23] PR Spalart and M Shur. On the sensitization of turbulence models to rotation and curvature. *Aerospace Science and Technology*, 1(5):297–302, 1997.

[24] C Gorlé, Johan Larsson, Michael Emory, and Gianluca Iaccarino. The deviation from parallel shear flow as an indicator of linear eddy-viscosity model inaccuracy. *Physics of Fluids*, 26(5):055105, 2014.

[25] Philippe R Spalart, Shur Deck, Michael L Shur, Kyle D Squires, M Kh Strelets, and Andrei Travin. A new version of detached-eddy simulation, resistant to ambiguous grid densities. *Theoretical and computational fluid dynamics*, 20(3):181–195, 2006.

[26] Andrea Ferrero, Angelo Iollo, and Francesco Larocca. Field inversion for data-augmented rans modelling in turbomachinery flows. *Computers & Fluids*, 201:104474, 2020.

[27] Mortaza Mani, Deric Babcock, Chad Winkler, and Philippe Spalart. Predictions of a supersonic turbulent flow in a square duct. In *51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, page 860, 2013.

[28] Julien Dandois, Eric Garnier, and Pierre Sagaut. Numerical simulation of active separation control by a synthetic jet. *Journal of Fluid Mechanics*, 574:25–58, 2007.

[29] Lucas Franceschini, Denis Sipp, and Olivier Marquet. Mean-flow data assimilation based on minimal correction of turbulence models: Application to turbulent high Reynolds number backward-facing step. *Physical Review Fluids*, 5(9):094603, 2020.