11th International Symposium on Turbulence and Shear Flow Phenomena (TSFP11)
Southampton, UK, July 30 to August 2, 2019

MACHINE LEARNING MODELING FOR RANS TURBULENT KINETIC ENERGY TRANSPORT IN 3D SEPARATED FLOWS

David S. Ching
Department of Mechanical Engineering
Stanford University
Stanford, CA 94305
dching@stanford.edu

Andrew J. Banko
Department of Mechanical Engineering
Stanford University
Stanford, CA 94305
abanko@stanford.edu

Pedro M. Milani
Department of Mechanical Engineering
Stanford University
Stanford, CA 94305
pmmilani@stanford.edu

John K. Eaton
Department of Mechanical Engineering
Stanford University
Stanford, CA 94305
eatonj@stanford.edu

ABSTRACT

Neural network based turbulence models are developed to improve predictions of complex separated flows. The k-ε turbulent kinetic energy transport equation is modified by multiplying the production term by a factor predicted by a neural network. The neural network is trained separately on Large-Eddy Simulation (LES) data of a three-dimensional skewed bump and a wall-mounted cube and tested by applying the model to Reynolds Averaged Navier-Stokes (RANS) simulations of the same flows. A Gaussian mixture model is used to predict where the neural network is extrapolating and the standard RANS model is used in those regions. The neural network and Gaussian mixture model are coupled directly in the RANS solver so predictions are made at each iteration. The method improves TKE and mean velocity predictions in the separated flow cases tested while maintaining an accurate boundary layer profile.

INTRODUCTION

Reynolds Averaged Navier-Stokes (RANS) simulations are not predictive in separated flows, especially when separation occurs on a smooth surface. The precise location of separation is determined by a balance between mean momentum transport, Reynolds stress gradients, and pressure gradients. If the location of separation on a smoothly contoured surface changes slightly, the separation bubble size and downstream flow can change significantly (Cherry *et al.*, 2008). Furthermore, Reynolds stresses are typically high in the shear layer immediately after separation, and inaccurate predictions of those Reynolds stresses impact the RANS model accuracy in the separation bubble and wake.

The Reynolds stress tensor $-\overline{u_i'u_j'}$ can be split as:

$$-\frac{\overline{u_i'u_j'}}{2k} = b_{ij} - \frac{1}{3}\delta_{ij} \tag{1}$$

where b_{ij} is the dimensionless anisotropy tensor, k is turbulent kinetic energy (TKE), and δ_{ij} is the Kronecker delta function. One interpretation of this decomposition is that the turbulent kinetic energy represents the magnitude of the Reynolds stress and the anisotropy tensor represents the directionality. In a RANS model, both b_{ij} and k must be modeled. In linear eddy viscosity models such as the $k-\varepsilon$ or $k-\omega$ models, the anisotropy tensor is represented as:

$$b_{ij} = \frac{\nu_T}{k}\overline{S_{ij}} \tag{2}$$

where $\overline{S_{ij}}$ is the mean strain rate tensor and ν_T is the turbulent viscosity, which is formed from a combination of other RANS variables.

RANS models solve auxiliary partial differential equations in addition to the time-averaged Navier-Stokes equations. The $k-\varepsilon$ model solves two auxiliary equations for turbulent kinetic energy, k, and turbulent dissipation rate, ε. Both RANS auxiliary equations represent transport equations for k or ε, but employ approximate models for some terms. For example, the RANS TKE transport equation is based on the exact TKE transport equation, but the production, diffusion, and dissipation terms are modeled.

From Equation 1, it is clear that to correctly predict the Reynolds stress, both the anisotropy tensor and turbulent kinetic energy must be correctly predicted, so both terms are potentially sources of error. The assumption that the anisotropy tensor is linearly related to the strain rate tensor is inaccurate in complex flows (Slotnick *et al.*, 2014). Furthermore, the use of Equation 2 to model the anisotropy tensor assumes that anisotropy can be represented by the local flow gradients, implying that the timescale for anisotropy to adjust to the local flow is short. The modeled auxiliary equations are sources of error for k and ν_T. The present work focuses solely on error in k due to the modeled auxiliary equations.

Typically, RANS models are evaluated by their accuracy at predicting the mean velocity field, and accurate prediction of TKE is of secondary importance. Our philosophy

is that a robust, physics-based model must accurately represent TKE to accurately predict the mean velocity field for a wide range of flow topologies. If both TKE and anisotropy are accurate, it is conjectured that the mean velocities will also be accurate. The only way a RANS simulation could accurately predict the velocity field but have incorrect TKE is if errors in TKE cancel errors in anisotropy. A model tuned to provide such cancellation in one class of flows is unlikely to work well in other cases. Furthermore, methods that improve the anisotropy tensor may aim to make the anisotropy tensor physically accurate rather than optimizing the velocity field (Ling *et al.* (2016*b*), Wang *et al.* (2017)). For those methods, it is important that the RANS TKE also be physical to make the overall Reynolds stress accurate.

In recent years, researchers have begun using machine learning (ML) to improve RANS predictions. Tracey *et al.* (2015) found that neural networks could be used to learn terms in a turbulence model, which suggested that ML could be used to improve RANS models. Ling *et al.* (2016*b*) developed the Tensor-Basis Neural Network (TBNN) that preserves Galilean invariance for predicting the anisotropy tensor and found that RANS simulations with TBNN predictions for the anisotropy tensor were more accurate than standard linear or quadratic eddy viscosity models (Ling *et al.*, 2016*b*, 2016*a*). Duraisamy *et al.* (2015) modified the Spallart-Allmaras eddy viscosity equation by multiplying the production term by a factor β. They used inverse modeling to identify the optimal β as a function of space and machine learning to predict β from local mean flow features. In Parish & Duraisamy (2016) and Duraisamy *et al.* (2017) their method was extended to modifying the auxiliary equations in the $k-\omega$ model by multiplying either production term by a new factor. Xiao and colleagues used random forests to model discrepancies between a RANS and DNS as functions of local mean flow features (Wang *et al.* (2017), Wu *et al.* (2018)). In their method as well as in Ling *et al.* (2016*b*), an initial standard RANS is run and the solution of that RANS is used to predict discrepancies using ML. A second RANS is run using the predicted discrepancies to improve the Reynolds stress. Predicting model discrepancies is dependent on the accuracy of the standard RANS simulation and will not work well if the standard RANS simulation is highly inaccurate. With the method of Xiao and colleagues, both the TKE and anisotropy are improved. The TKE is improved by predicting the logarithm of the ratio of the true TKE to the RANS TKE, then using the predicted discrepancy to improve the TKE of the second RANS.

The equation modification in this work is inspired by Parish & Duraisamy (2016), but our methodology is different. Ling *et al.* (2016*b*) has already developed a method to improve predictions of the anisotropy tensor, and Duraisamy *et al.* (2015) developed a method to improve predictions of turbulent kinetic energy, but the two methods cannot be combined consistently. Any optimization that minimizes error in the velocity or turbulent kinetic energy is dependent on errors in the anisotropy tensor. Once the TBNN is used to correct the anisotropy tensor, the method of Duraisamy *et al.* (2015) is no longer optimal.

Here we develop an alternative method of identifying β that is independent from anisotropy tensor errors, so that the method can eventually be combined with a TBNN to improve both components of the Reynolds stress. We modify the k-ε turbulent kinetic energy transport equation using neural networks trained on Large Eddy Simulation (LES) data to model β from local flow features. The neural network is coupled in the RANS solver to update predictions during the simulation.

The training data features (inputs) can be described as data from a distribution in feature space. The neural network is well trained where the training data has high density in feature space, but is not well trained where the density of training data is low. In low-density regions, neural networks can extrapolate and give highly inaccurate results. When a neural network is used to improve RANS simulations, the flow configurations tested are different from the flow configurations from which training data is extracted. The distributions of training and test data may be different, allowing extrapolation to occur. Here we use a Gaussian mixture model to approximate the training data feature distribution so that potential neural network extrapolation can be identified at run time. The ML RANS then reverts back to a standard RANS model rather than using extrapolated values.

APPROACH

Neural Network

In an accurate RANS model, the RANS equations should give an accurate representation of TKE when the mean velocity is correct. In that case, solving the RANS equations with mean velocity and pressure from a high-fidelity simulation should give an accurate TKE. However, current RANS models give inaccurate predictions of TKE, especially in complex flows. This work adjusts the RANS TKE equation to predict more accurate TKE with the longer term goal of improving mean flow prediction.

We modify the production term of the realizable k-ε turbulent kinetic energy transport equation (Eqn. 3) by multiplying it by a new factor β, which is a function of local mean flow features.

$$\frac{\partial}{\partial x_j}(\rho k u_j) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma k}\right)\frac{\partial k}{\partial x_j}\right] + \beta\mu_T S_{ij}S_{ij} - \rho\varepsilon \tag{3}$$

This modification is similar to the method of Duraisamy *et al.* (2017), but the method of determining β is different. The procedure of identifying β is as follows: The RANS ε equation is calculated with the high-fidelity simulation velocity, pressure, and TKE held fixed. Using the calculated distribution of ε, all the variables in the RANS TKE equation are known, but the equation is not balanced. β is determined as a function of space such that Eqn. 3 is balanced. If those values of β were used to adjust the TKE equation for that particular configuration, the RANS TKE equation would be balanced when the RANS predicts the LES TKE and mean velocity.

β is determined as a function of space, but for a general model it should be parameterized as a function of local flow variables. Here, a neural network is used to convert β to a function of local flow invariants. Specifically, the neural network inputs are the invariants of the mean strain and rotation rate tensors S and R, the wall distance Reynolds number, and the turbulent viscosity ratio. These input features are listed in Table 1. The strain and rotation rate tensors in Table 1 are nondimensionalized by k/ε, as suggested by Pope (1975). The neural network loss function used during training is the square of the imbalance in Equation 3.

Table 1. Input features to the neural network and Gaussian mixture model.

$Tr(S^2)$	$Tr(R^2)$	$Tr(S^3)$	$Tr(R^2S)$
$Tr(R^2S^2)$	$\frac{yk^{0.5}}{\nu}$	$\frac{\nu_T}{\nu}$	

The neural network has 10 hidden layers, each with 15 nodes. The neural network size is chosen such that no overfitting occurs but the neural network is sufficiently large to create complex functional relations. Each neuron not in the last layer has a rectified linear unit (ReLU) activation function. The neural network is trained to convergence, with a learning rate of 10^{-5} at the end. Values of β are limited to the range between 0 and 4 before updating the RANS TKE equation.

When applied to a new flow, the neural network predicts β as a function of the local features computed by the RANS. For consistency, the neural network is coupled in the RANS solver to make updated predictions each iteration. This method was designed as a 'turbulence model' in the sense that the output (β) is a function only of local flow variables. The method is independent of errors in an initial RANS, unlike discrepancy-based methods. We use the RANS solver Fluent, developed by ANSYS, with the realizable $k-\varepsilon$ model. The neural network is coded into user-defined functions that calculate β each timestep. Each iteration takes approximately twice as long with the neural network. However, it is not necessary to update the neural network predictions each timestep. The method works equally well when the neural network predictions are updated every fifth iteration step, in which case the slowdown is approximately 20%.

Gaussian Mixture

Neural networks can extrapolate when used on data unlike the training data. When used to improve RANS simulations in engineering, it is difficult to prevent extrapolation from occuring. As a product is being designed, the flow configurations are different from the training data. Ideally, the ML algorithms are trained on a wide variety of flows so that the ML predictions are accurate in any engineering flow. Currently, no large database of high-fidelity simulations exists with which to train the algorithms. Furthermore, identifying when the ML algorithm is trained on enough different flows is still an open research topic and beyond the scope of the present work. It is unclear if it is even possible to create a sufficiently large database of flows to make a fully general data-driven RANS model.

Wu *et al.* (2017) used the Mahalanobis distance and kernel density estimation to assess if their ML algorithm was extrapolating based on the training data inputs. Here we use a Gaussian mixture model to estimate if the neural network is overfitting, in which case our model will revert back to the baseline RANS (set β to unity). The Gaussian mixture model is a method of approximating a density function from data with a sum of weighted Gaussian functions (Dempster *et al.*, 1977). The Gaussian mixture model is trained on the same input features as the neural network. During testing, the Gaussian mixture model returns a score for each point. If the score is lower than that of 98% of the training data, the ML model reverts back to the standard k-ε model for that point.

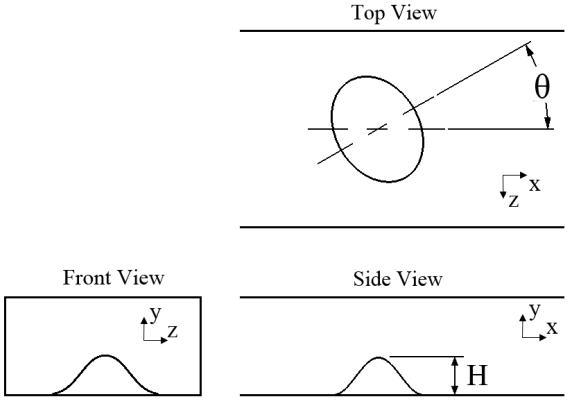

Figure 1. Schematic of wall-mounted bump flow.

FLOW CONFIGURATIONS

The ML models are trained using data from LES's of two different separated flows over a skewed three-dimensional wall-mounted bump with Reynolds number of 16,000 based on bump height from Ching & Eaton (2019). The bump has cosine cross-sections from the side and elliptical cross-sections from the top with 4/3 axis ratio, shown schematically in Figure 1. The incoming boundary layer thickness is half the bump height. LES's are run with the bump at angles of 10° and 50° with respect to the flow, producing a highly 3D separated flow. The LES's are wall resolved on the bottom surface and were carefully validated against experimental mean velocity data from Ching *et al.* (2018*a*) and Ching *et al.* (2018*b*).

The skewed bump flow is highly sensitive to the precise geometry. Vortex dynamics of the wake are very different at the two bump angles studied, which leads to significantly different mean fields and TKE in the wake. The bump flow is therefore a very challenging case for RANS models to predict. It is found that the realizable $k-\varepsilon$ model converges to a steady solution for the 10° bump angles, but does not converge for the 50° bump. We therefore use the 50° bump only for training.

The trained models are tested on the bump flows as well as flow over a wall-mounted cube and a flat-plate boundary layer. Rossi *et al.* (2010) ran a Direct Numerical Simulation (DNS) of a wall-mounted cube at Reynolds number of 5000 based on cube height. The cube has a small vertical jet on the top surface with diameter 1/12 of the cube height and uniform velocity 0.19 times the freestream velocity. The boundary layer thickness is 1/5 the cube height. The cube DNS has a uniform velocity inlet with no turbulence and the boundary layer Reynolds number based on distance from the inlet is 30,000. At this low Reynolds number, the boundary layer never transitions to turbulence. Due to the absence of a turbulent boundary layer, the cube case is not used for training models tested on the bump. To exclude laminar regions from the training data, only points with turbulence intensity greater than 3% and wall distance Reynolds number greater than 5 are included in the training data for all the flow configurations. The ML model is applied only to regions that satisfy the same conditions.

In addition to the separated flow cases, the ML model is tested for a flat plate boundary layer. Because the ML model is developed for 3D flows, the RANS is run in a 3D domain with symmetry side walls, resulting in a 2D boundary layer. The mesh has 100,000 elements and is a single element wide in the spanwise direction. The bottom wall is

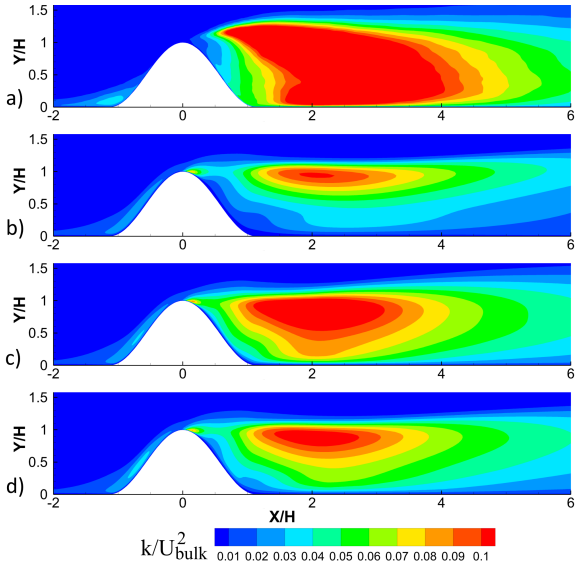

Figure 2. a) TKE contours on the centerplane of the 10° bump LES. b) Baseline RANS. c) ML RANS trained on 10° bump. d) ML RANS trained on 50° bump

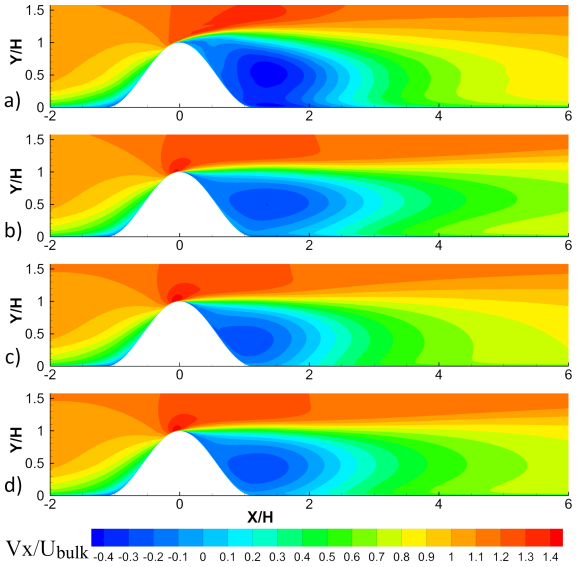

Figure 3. a) Streamwise mean velocity contours on the centerplane of the 10° bump LES. b) Baseline RANS. c) ML RANS trained on 10° bump. d) ML RANS trained on 50° bump

resolved with a y^+ of 0.3 at the outlet. Boundary layer profiles are compared to the experimental data of Degraff and Eaton (2000) at $Re_\theta = 2900$.

RESULTS

The ML algorithms are trained and tested on different flows, since when used in engineering the method would be applied to test flows different from the training flows. In practice, it is desirable that the ML algorithms are trained on flows similar to the test flows for best accuracy, but it is possible that a test flow contains features not found in the training data. In the present work, ML algorithms trained on one bump angle are applied to the other bump angle and to the cube flow to examine both scenarios.

Because this method aims to increase accuracy of the RANS TKE equation, but does not change anisotropy errors, the mean velocity fields are expected to have large error, particularly in areas that anisotropy is highly important. The main performance metric of this method should be improvement of RANS TKE. It should be noted that incorrect anisotropy leads to mean velocity errors and velocity gradients are inputs to both the TKE equation and the ML correction. Therefore, RANS TKE predictions are still coupled to incorrect velocity predictions, so the TKE is still expected to have error even after the ML correction.

Figure 2a shows TKE on the centerplane of the 10° bump from the LES, and Figure 2b shows TKE from the standard RANS. The RANS severely underpredicts TKE in the wake, particularly near the walls. Figure 2c shows the results when ML models trained on the identical 10° bump are used to modify the RANS TKE equation. Training and testing on the same flow is not a good metric of the model's performance, but it is informative of the best possible improvement from this model. The ML model improves TKE predictions significantly, especially near the bottom surface and far downstream. Figure 2d shows an ML RANS trained on the 50° bump and applied to the 10° bump. The TKE is significantly improved over the baseline RANS (Figure 2b), although not as much as the model trained on the 10° bump (Figure 2c). The differences between models trained on different bump angles is due to geometric sensitivity of the bump.

Figure 3 shows contours of streamwise velocity for the same cases. The LES (Figure 3a) has stronger backflow in the separation bubble and the wake decays faster than the standard RANS model (Figure 3b). Both ML RANS improve the mean velocity in the far wake, but do not improve the reattachment location or strength of reversed flow in the separation bubble.

It is important to note that no method is used to modify the anisotropy tensor. It is well known that in separated flows, the RANS model for anisotropy given in Equation 2 is inaccurate. Since TKE and anisotropy are coupled, errors in anisotropy affect the velocity field and thereby the TKE. It is therefore not surprising that the ML RANS do not improve the velocity field in the separation bubble, since there is a high degree of anisotropy in that region. Downstream of the separation bubble anisotropy is far less important and the converged ML RANS models improve the velocity field. It is promising that the far wake results can be improved solely by modifying the RANS TKE equation. While this method is designed to eventually be combined with a TBNN to improve the anisotropy tensor, adjusting the anisotropy tensor to improve overall predictions is beyond the scope of this work.

Figure 4 shows contours of β predicted by the neural network after the RANS simulation has converged for the two ML RANS simulations. Points where turbulence intensity is less than 3% or the Gaussian mixture model density function is lower than the 2% percentile of the training data are blanked. The ML model is only trained on points with turbulence intensity greater than 3%, so for consistency it is applied only to the same points. In regions where Figure 4 is blanked, the ML RANS reverts back to a standard RANS, which is equivalent to setting β to unity. Note that the values of β in Figure 4 vary widely, indicating large errors in the standard model. The predicted value of β is generally high near the bottom surface and in the shear layer above

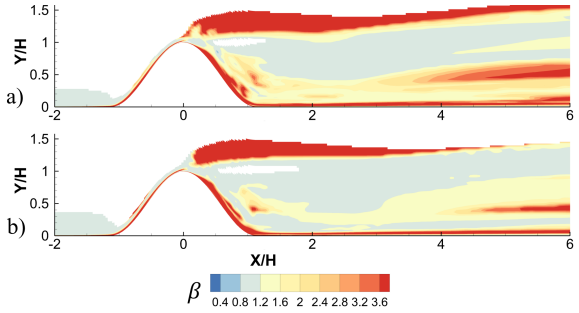

Figure 4. a) Contours of β in the 10° bump using the ML model trained on the 10° bump. Points where the Gaussian mixture model predicts potential extrapolation are blanked. b) ML model trained on the 50° bump.

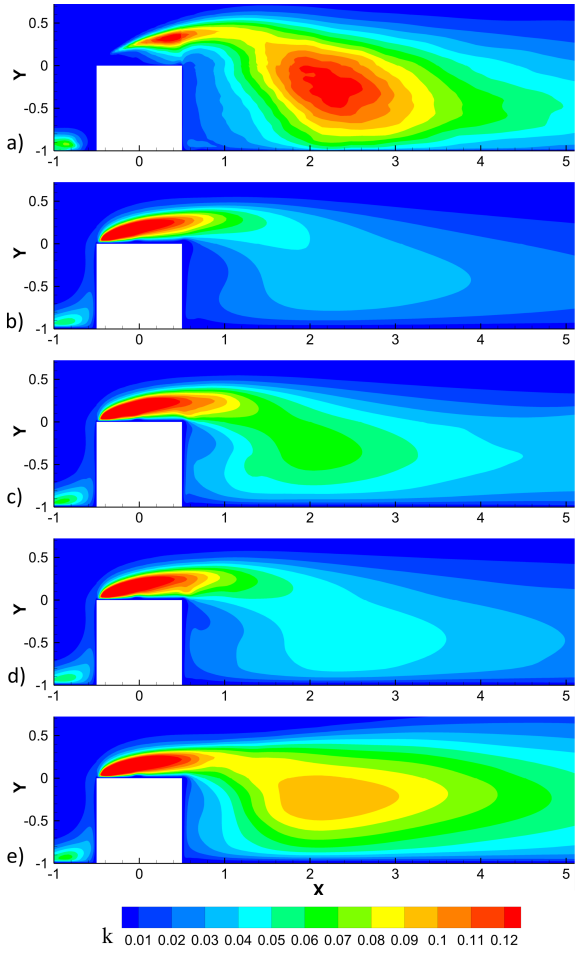

Figure 5. a) Contours of TKE on centerplane of cube DNS. b) Baseline RANS. c) ML RANS trained on cube. d) ML RANS trained on 50° bump. e) ML RANS trained on 10° bump.

the bump. This indicates that the standard RANS model production is too low or the dissipation is too high in those regions.

The machine learning algorithms were next used to predict the wall-mounted cube flow. Figure 5 shows TKE contours from each simulation. The baseline RANS (Figure 5b) severely underpredicts TKE in comparison to the DNS (Figure 5a). An ML model trained on the cube in Figure 5c significantly improves the wake. The ML models trained

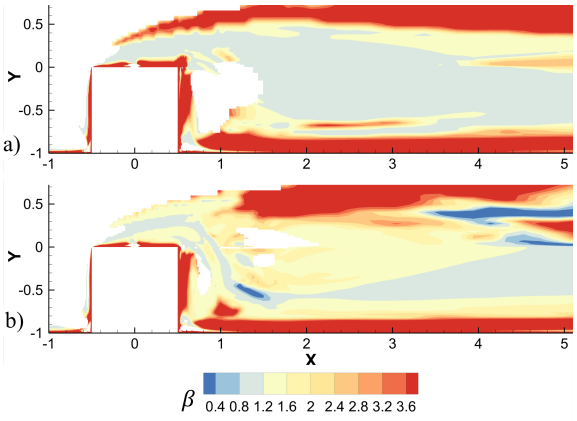

Figure 6. a) Centerplane contours of β from ML RANS trained on 50° bump. b) ML RANS trained on 10° bump.

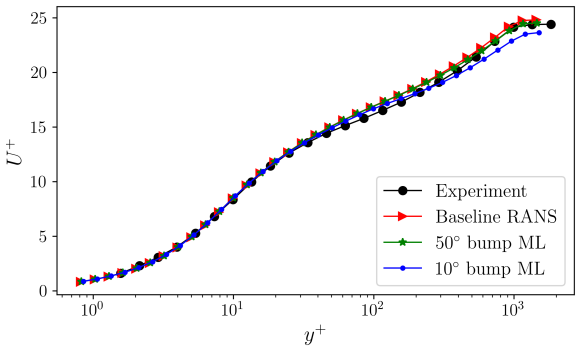

Figure 7. Streamwise velocity profiles at $Re_\theta = 2900$ from Degraaff and Eaton (2000), standard RANS, and ML RANS models trained on 10° bump and 50° bump.

on the 50° bump (Figure 5d) and the 10° bump (Figure 5e) give very different results, although both models are more accurate than the baseline RANS.

Figure 6 shows contours of β in the cube flow when trained separately on the 50° and 10° bumps. Not only are the contours of β significantly different, the blanking due to the Gaussian mixture model is different in the near wake. A large region of the near wake is blanked in the ML RANS trained on the 50° bump (Figure 6a) but not the 10° bump (Figure 6b). This suggests that the flow in the wake of the cube is more similar to the 10° bump than the 50° bump, so the 10° bump forms a better training case for the cube flow. It is likely that the TKE for the ML RANS trained on the 50° bump is low partially because the ML model reverts back to the standard RANS model in that region.

The original realizable $k-\varepsilon$ model was tuned to give a reasonably accurate boundary layer mean velocity profile. It is highly important that when the TKE equation is improved with machine learning, the modification does not make the boundary layer profile inaccurate. To test this, the ML models trained on the bumps are applied to the flat-plate boundary layer RANS. Figure 7 shows the streamwise velocity profiles at $Re_\theta = 2900$. The profiles are compared to experimental data from Degraaff and Eaton (2000) and show reasonably accurate velocity profiles.

Figure 8 shows TKE profiles for the same cases. Due to the absence of spanwise velocity fluctuations in Degraaff's data, we assume spanwise fluctuations are equal to the average of streamwise and wall-normal fluctuations when plot-

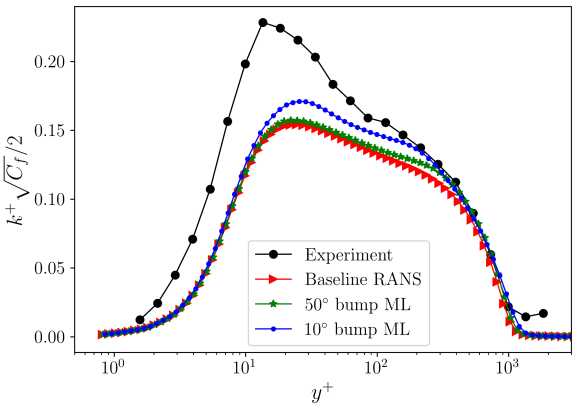

Figure 8. TKE profiles at $Re_\theta = 2900$ from Degraaff and Eaton (2000), standard RANS, and ML RANS models trained on 10° bump and 50° bump.

ting the experimental TKE. Both the standard RANS and ML RANS simulations underpredict TKE in the boundary layer, but the ML RANS simulations marginally improve the TKE profiles.

CONCLUSIONS

A neural network is used to modify the turbulent kinetic energy transport equation in the $k-\varepsilon$ RANS model to improve predictions of turbulent kinetic energy. The method is designed to be independent of errors in the RANS anisotropy model so that future work can combine this method with a Tensor Basis Neural Network (TBNN) (Ling *et al.*, 2016*b*). The ML RANS models are trained and tested on separated flows over a three-dimensional, wall-mounted bump and a wall-mounted cube. Because the training and test flows are different, neural networks trained on one flow may extrapolate when applied to a different flow. A Gaussian mixture model predicts where the neural network extrapolates and reverts the ML model back to the standard RANS model in those regions. Both the neural network and Gaussian mixture model are coupled in the RANS solver to make predictions each iteration so the converged ML RANS uses converged variables to make predictions. The ML models show significant improvement to the velocity and turbulent kinetic energy predictions when applied to both the bump and a wall-mounted cube. Tests show that the model maintains accurate boundary layer profiles.

It is important to note that as a standalone method, the modifications of Parish & Duraisamy (2016) are preferred to the present work. However, only this method can be combined with a TBNN to improve both TKE and anisotropy predictions.

REFERENCES

Cherry, Erica M, Elkins, Christopher J & Eaton, John K 2008 Geometric sensitivity of three-dimensional separated flows. *International Journal of Heat and Fluid Flow* **29** (3), 803–811.

Ching, David S & Eaton, John K 2019 Geometric sensitivity and unsteady wake dynamics on a skewed bump. *Journal of Fluid Mechanics, in revision* .

Ching, David S, Elkins, Christopher J & Eaton, John K 2018*a* Investigation of geometric sensitivity of a non-axisymmetric bump: 3D mean velocity measurements. *Experiments in Fluids* **59** (143).

Ching, David S, Elkins, Christopher J & Eaton, John K 2018*b* Unsteady vortex structures in the wake of non-axisymmetric bumps using spiral MRV. *Experiments in Fluids* **59** (144).

De Graaff, David B & Eaton, John K 2000 Reynolds-number scaling of the flat-plate turbulent boundary layer. *Journal of Fluid Mechanics* **422**, 319–346.

Dempster, Arthur P, Laird, Nan M & Rubin, Donald B 1977 Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* **39** (1), 1–22.

Duraisamy, Karthik, Singh, Anand Pratapand & Pan, Shaowu 2017 Augmentation of turbulence models using field inversion and machine learning. In *55th AIAA Aerospace Sciences Meeting*, p. 0993.

Duraisamy, Karthik, Zhang, ZJ & Singh, AP 2015 New approaches in turbulence and transition modeling using data-driven techniques. In *53rd AIAA Aerospace Sciences Meeting*, p. 1284.

Ling, Julia, Jones, Reese & Templeton, Jeremy 2016*a* Machine learning strategies for systems with invariance properties. *Journal of Computational Physics* **318**, 22–35.

Ling, J, Kurzawski, A & Templeton, J 2016*b* Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics* **807**, 155–166.

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

Pope, SB 1975 A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics* **72** (2), 331–340.

Rossi, R, Philips, DA & Iaccarino, G 2010 A numerical study of scalar dispersion downstream of a wall-mounted cube using direct simulations and algebraic flux models. *International Journal of Heat and Fluid Flow* **31** (5), 805–819.

Slotnick, Jeffrey, Khodadoust, Abdollah, Alonso, Juan, Darmofal, David, Gropp, William, Lurie, Elizabeth & Mavriplis, Dimitri 2014 Cfd vision 2030 study: a path to revolutionary computational aerosciences. *NASA Technical Reports* .

Tracey, Brendan D, Duraisamy, Karthikeyan & Alonso, Juan J 2015 A machine learning strategy to assist turbulence model development. In *53rd AIAA aerospace sciences meeting*, p. 1287.

Wang, JX, Wu, JL & Xiao, H 2017 Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. *Physical Review Fluids* **2** (3), 034603.

Wu, JL, Wang, JX, Xiao, H & Ling, J 2017 A priori assessment of prediction confidence for data-driven turbulence modeling. *Flow, Turbulence and Combustion* **99** (1), 25–46.

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