Aeroacoustics Conferences
June 14-17, 2022, Southampton, UK
28th AIAA/CEAS Aeroacoustics 2022 Conference

10.2514/6.2022-2883

Fan Wake Prediction Via Machine Learning

Zijie Huang*, Hao Shen*, Kelly Kung*, Luis Carvalho†
Boston University, Boston, MA, 02215, U.S.A.

Austin Thai‡, Berkely Wachtmann§, Tyler Ramsarran§
Boston University, Boston, MA, 02215, U.S.A.

Julian Winkler¶, C. Aaron Reimann‖, Michael Joly**, Kin Gwn Lore††, Jeff Mendoza‡‡
Raytheon Technology Research Center, East Hartford, CT 06108

Sheryl M. Grace§§
Boston University, Boston, MA, 02215, U.S.A.

The interaction noise in the fan stage of a turbofan engine is now a dominant engine noise source especially on approach and landing. The noise is created by the interaction of the fan wake and the exit guide vanes. Prediction of the noise thus requires knowledge of the fan wake. The tonal noise is attributed to the mean fan wake while the broadband noise is produced by the turbulence in the wake. In this research, the possibility of developing machine learning algorithms for use as a surrogate model of the fan wake flow is investigated. The preliminary study utilized 4 rather similar fan geometries at 7 rotor speeds and multiple inlet mass flows to train two different genres of machine learning algorithms. Both methods work well in this study providing motivation for further investigation in which additional fans with larger geometrical differences are added to the training database.

I. Introduction

The noise signature of contemporary turbofan engines contains large contribution from the fan stage in both tonal and broadband components. The noise downstream of the fan stage is mainly created by interaction between the fan wake and the fan exit guide vane (FEGV). The tonal portion of the noise is driven by the mean fan wake, the number of fan blades, and the fan rotation speed. The broadband noise is driven by the turbulence in the fan wake and the fully averaged flow speed into the FEGV.

Good reviews of current methodologies for predicting fan interaction tonal and broadband noise can be found in Refs. [1]&[2]. The low-order, faster, methods focus on the FEGV response to inflow disturbance which is then coupled to an acoustic transmission model. The FEGV inflow conditions are currently determined from experiment or computation. Basic Computational Fluid Dynamics (CFD) solvers that provide solutions to the Reynolds Averaged Navier Stokes (RANS) equations offer a method for obtaining the FEGV inflow [2, 3]. Higher fidelity methods utilizing Large Eddy Simulation (LES) or Very Large Eddy Simulation (VLES) have also been used ([4–6]. The need for such simulations in order to create the input for the FEGV response calculation renders the entire prediction process untenable for use in design. Alternatively, we seek a method to determine the FEGV inflow conditions based on data available from a standard performance design tool such as AxStream [7].

This study, describes efforts to develop such a fan-wake surrogate model using machine learning. Several machine learning algorithms to predict the wake parameters of interest have been have been developed and tested in this research. The wake parameters that require a surrogate model are explained together with the development of the training and

*Graduate Student, Mathematics & Statistics Department.
†Associate Professor, Mathematics & Statistics Department.
‡Graduate Student, Mechanical Engineering Department, AIAA Student Member.
§Undergraduate Student, Mechanical Engineering Department
¶Sr. Principal Engineer, Acoustics, Aerothermal & Intelligent Systems Department, AIAA Senior Member.
‖Sr. Manager, Acoustics, Aerothermal & Intelligent Systems Department, AIAA Senior Member.
**Principal Engineer, Aerodynamics, Aerothermal & Intelligent Systems Department.
††Principal Engineer, Advanced Learning & Analytics, Aerothermal & Intelligent Systems Department.
‡‡Technical Fellow, Acoustics, Aerothermal & Intelligent Systems Department, AIAA Associate Fellow.
§§Associate Professor, Mechanical Engineering Department, AIAA Associate Fellow.

testing data in Section II. The machine learning methods that have been considered are discussed in Section III. The ability to use machine learning as a surrogate model for fan wake parameters is explored in Section IV.

II. Fan wake data

The fan-stage interaction noise prediction methods require input information related to the flow upstream of the FEGV. For tonal noise, the required inflow information includes the average passage form of the mean wake velocity from hub to tip together with the fan speed and number of blades. For broadband noise, the mean flow velocity at locations from hub to tip is again needed, but only the overall average value. In addition, input related to the wake turbulence is required. All low-order formulations rely on knowledge of the turbulence intensity and turbulence length scales. When turbulence isotropy is assumed, turbulence intensity can be related easily to turbulent kinetic energy (TKE). Some methods utilize the passagewise distribution of the turbulence intensity or TKE at locations from hub to tip and other methods rely a single averaged value at each radial position. Most methods are formulated such that the longitudinal turbulence length scale appears in the definition of the turbulence spectrum. Some also require the transverse radial turbulence length scale to either set the turbulence spectrum and/or the radial discretization.

For years, the NASA Source Diagnostic Test (SDT) has served as a benchmark for fan noise research. As part of SDT, flow measurements downstream of the fan were obtained [8, 9]. The SDT used a fan with 22 blades and data from 3 fan speeds were widely distributed. The data have been used as input for the interaction noise models and the predictions have been compared to acoustic measurements made as part of the SDT [10–12]. The data are used here to provide a visual understanding of the fan flow parameters for which a surrogate model is sought.

First, the mean wake is considered. Fig. 1 shows the axial, circumferential and radial mean values from hub to tip at normalized locations measured from the fan tip trailing edge. Results for the approach fan speed which is 67% of the design speed are shown here. At this point, the reader is directed to consider the experimental hotwire data labeled as HW. The normalization is based on fan radius and the two normalized axial positions are then 0.27 and 0.57 denoted as Stations 1 & 2 in the figure. For tonal noise, the average passage profile of the mean flow (often specified in the streamwise, upwash and tangential frame of reference) is important. The average passage values near the midspan radial position are shown in Fig. 2 at both Stations downstream of the fan.

Next the turbulence parameters are presented. At the approach fan speed, hot-wire data were obtained in 3 directions. Therefore, three components of the turbulence intensity downstream of the fan are available. Fig. 3 shows the turbulence intensity in the streamwise, upwash and tangential directions. Combining the turbulence intensities, one can obtain the TKE. For isotropic turbulence the relationship is $u' = \sqrt{(2/3)\overline{\text{TKE}}}$. The TKE is shown in its average passage form near midspan in Fig. 4 as well as the overall average value of TKE from hub-to-tip in Fig. 3g&h.

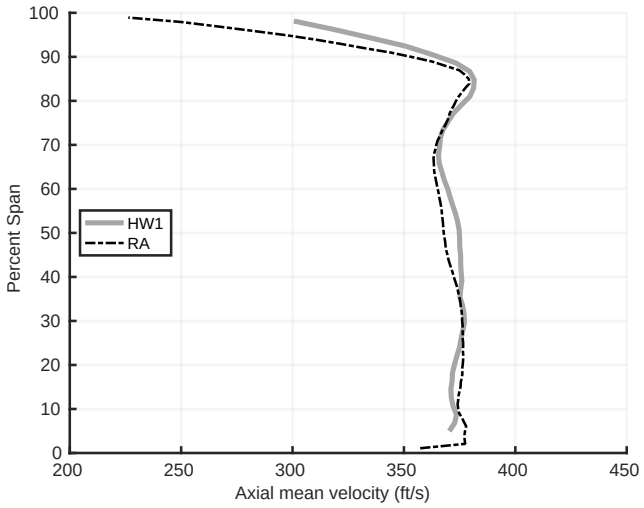
The final turbulence parameter of interest is the length scale. The hotwire data can also be use to compute longitudinal and lateral turbulence length scales. The method relies on the Taylor's hypothesis for frozen turbulence, and utilizes the time signal of the streamwise or upwash velocity via

$$L_s = \overline{U_s} \int_0^\infty \frac{\overline{u'_{s,u}(\tau) u'_{s,u}(\tau + t)}}{u'^2_{s,u}} dt \tag{1}$$

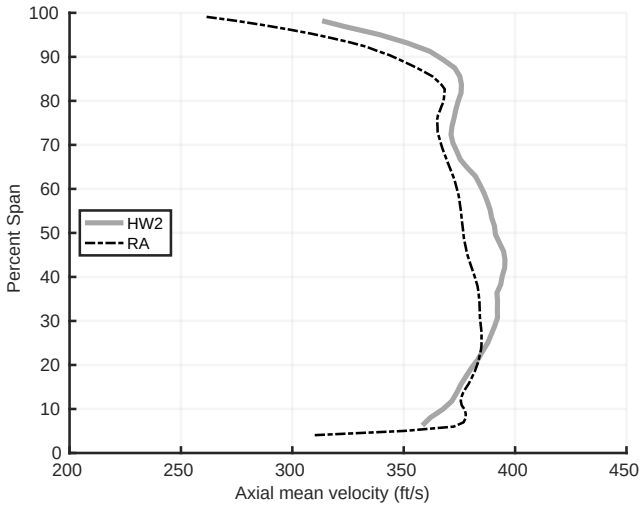
where $\overline{U_s}$ is the overall mean streamwise velocity at the radial location of interest. u'_s is the turbulence intensity in the streamwise direction and u'_u is turbulence intensity in the upwash direction (perpendicular to the streamwise direction). The length scale from hub-to-tip computed in this manner is shown in Fig. 5 as the HW curves.

As experimental data are not usually available, a fast, reliable way to obtain this type of flow information downstream of a fan is sought. Sophisticated, time consuming computations based on large eddy simulation can provide time resolved data equivalent to an experimental hot wire and as such all of the information in the Figs. 1-5 can be identically derived with reasonable accuracy [13]. Faster, RANS based computations offer time averaged mean flow and turbulence model quantities. In particular, two-equation turbulence models assume isotropy and require the computation of the TKE (or k in the model) and a second parameter related to the turbulence dissipation, ϵ or ω ($\omega = 0.09$ TKE ϵ). The turbulence length scale is then defined by a model such as that given by Pope [14] $L_s = 0.43\text{TKE}^{1.5}/\epsilon$.

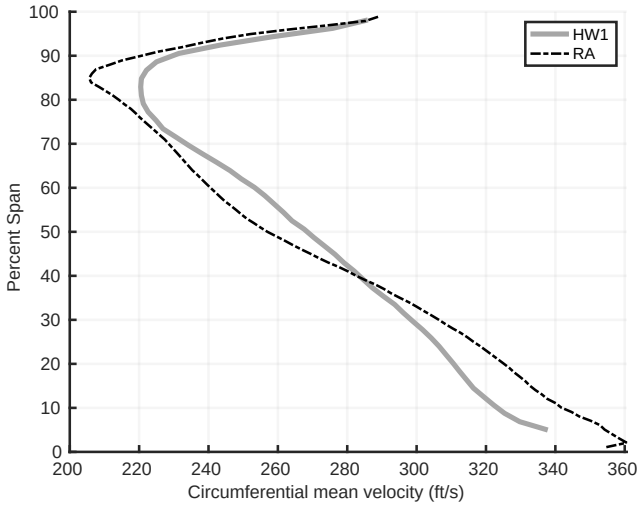
The development of a machine learning surrogate model for these input parameters requires a large database. In this work a database is being developed via computation. The simulations in this study are performed using a multi-block structured in-house CFD code (UTCFD) that solves the compressible unsteady Reynolds-averaged Navier–Stokes equations (URANS) for an ideal gas. The code uses the Lax-Wendroff multiple grid scheme of Ni [15], centered second-order spatial differencing with second-order and fourth-order smoothing. Turbulence is modeled using the



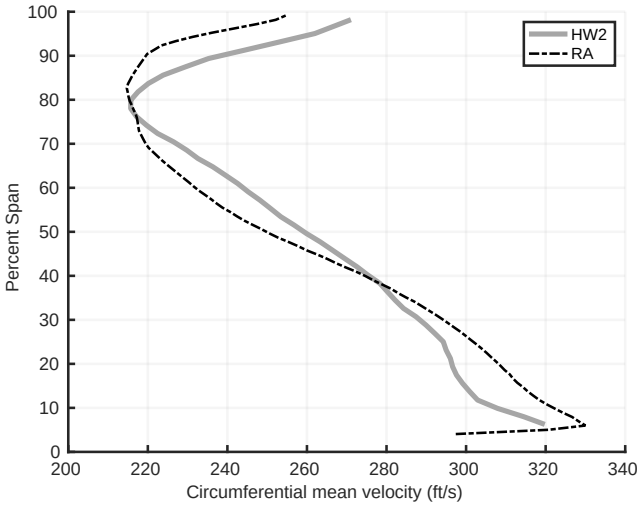
(a) Axial velocity. Station 1.



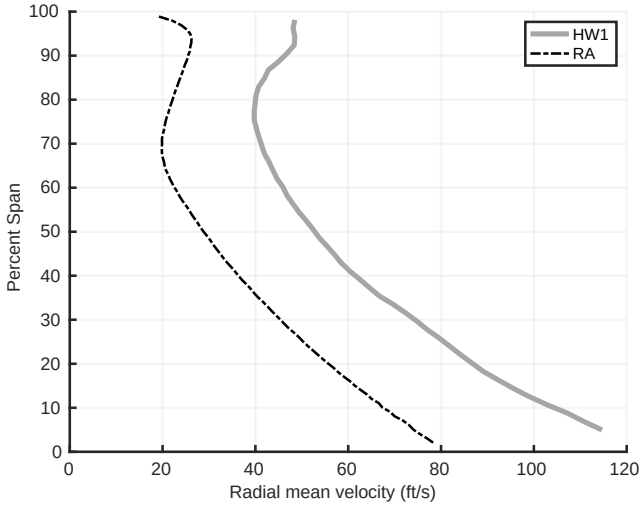
(b) Axial velocity, Station 2.



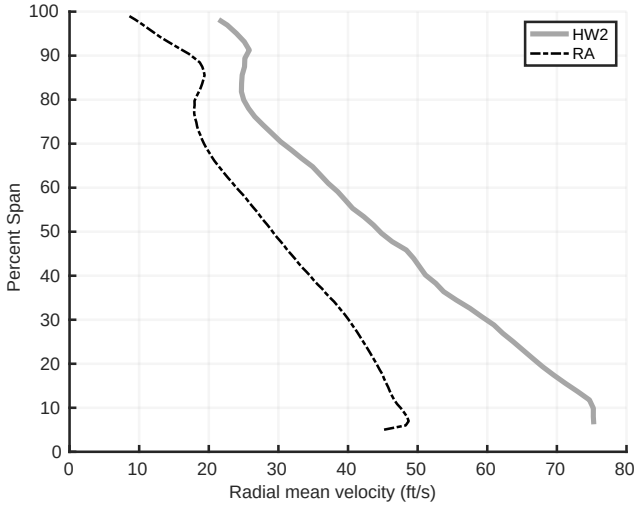
(c) Circumferential velocity, Station 1.



(d) Circumferential velocity, Station 2.

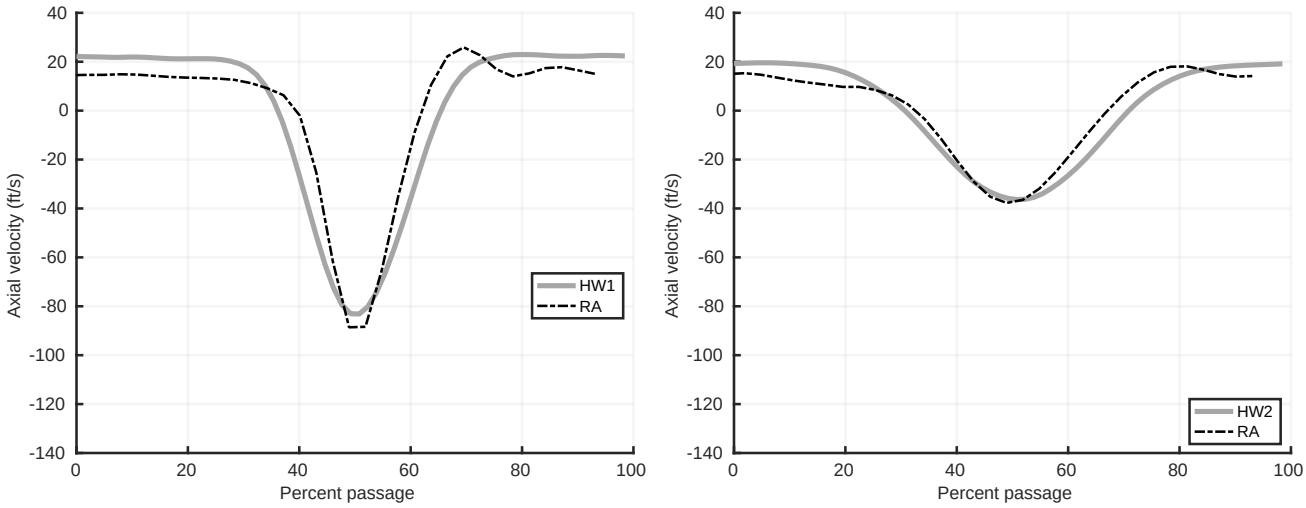


(e) Radial velocity, Station 1.



(f) Radial velocity, Station 2.

Fig. 1 Overall mean velocity values downstream of SDT fan at approach speed. Left: Station 1. Right: Station 2.



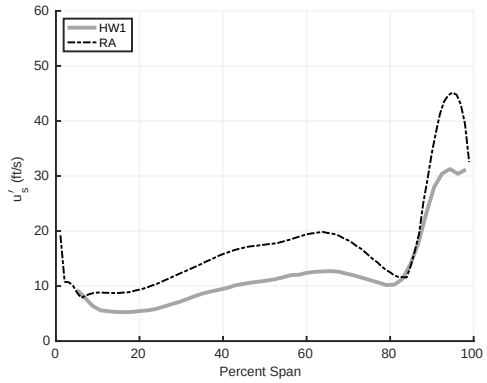
(a) Station 1. (b) Station 2.

Fig. 2 Average passage mean axial velocity at midspan for the approach fan speed. Left: Station 1. Right: Station 2.

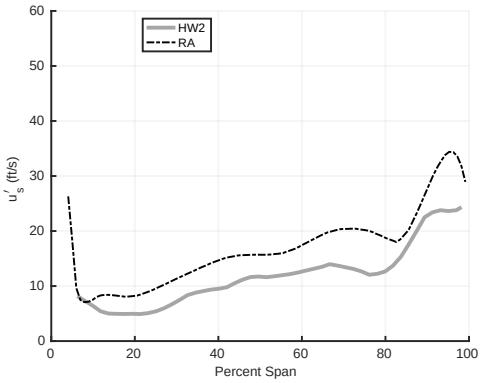
k-ω-model of Wilcox [16]. The solver has been extensively validated for numerous turbomachinery applications, including acoustic studies. For recent application of the present CFD code to turbofan noise predictions see Winkler et al. [17, 18] and Prasad et al. [19].

For the current CFD simulations, the fan alone is simulated as a single sector with periodic boundary conditions, and the governing equations are solved in a steady state fashion in the rotating frame of reference. The inlet and outlet conditions are steady 2D non-reflecting characteristic boundary conditions, based on Giles [20]. Initial total pressure and temperature profiles are specified at the domain inlet, and the desired target mass-flow rate is specified at the domain outlet, together with a radial equilibrium assumption of the flow. All simulations are run until all monitor points reach a converged value. Near choke and the near stall, the simulations show more strongly developed unsteady flow characteristics, which would neccessitate a switch to the full URANS simulation mode. No such cases were included in this work.

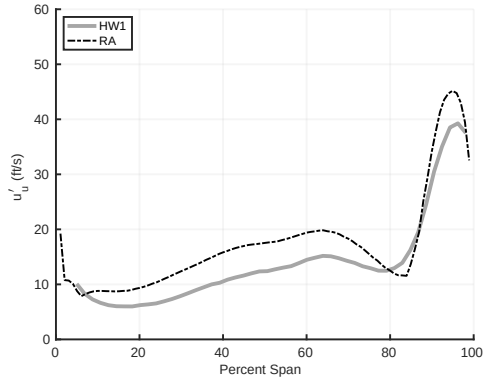
The computational grid consists of 145 radial points and 17 points in the tip gap, where the tip gap size is 0.02". The total grid count for the single rotor passage is approximately 6 million. While the RANS, fan-alone, computations are faster than URANS or LES, the simulations still require roughly 25 hours per case when running on a single core. The wall clock time can be reduced of course by using multiple cores. Additionally, a change in the fan geometry requires the redevelopment of a grid.



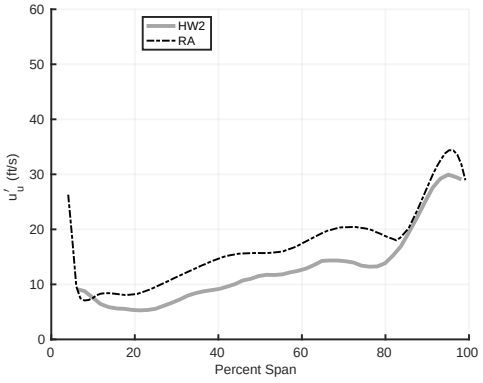
(a) Streamwise RMS, Station 1.



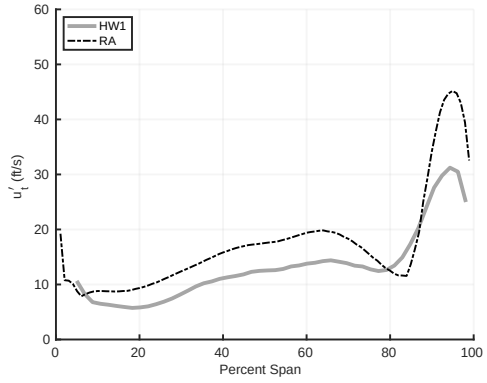
(b) Streamwise RMS, Station 2.



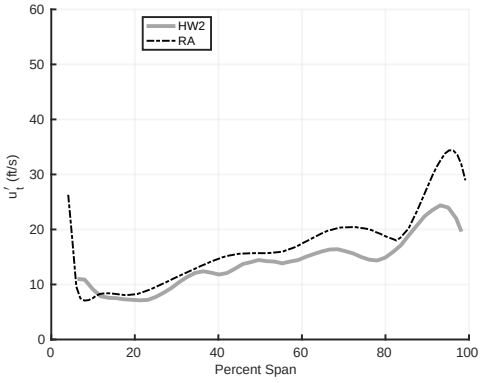
(c) Upwash RMS, Station 1.



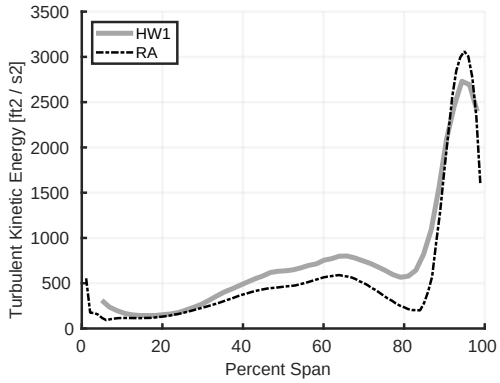
(d) Upwash RMS, Station 2.



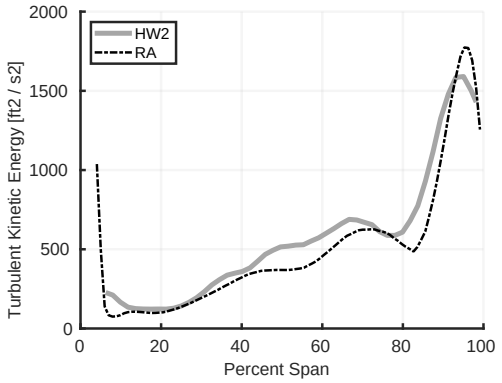
(e) Tangential RMS, Station 1.



(f) Tangential RMS, Station 2.

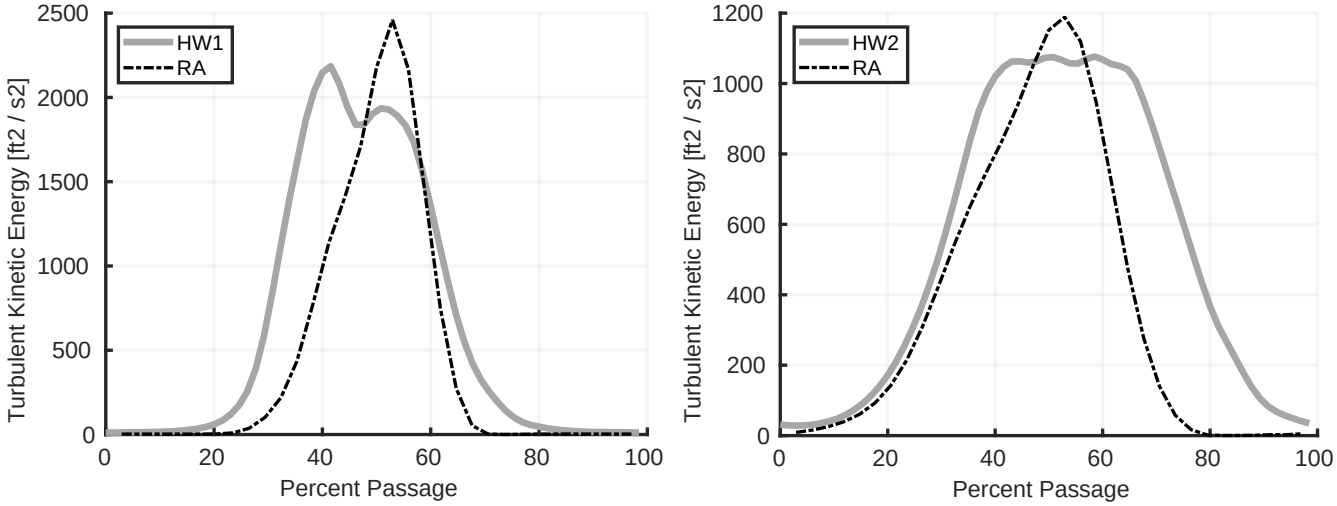


(g) TKE, Station 1.



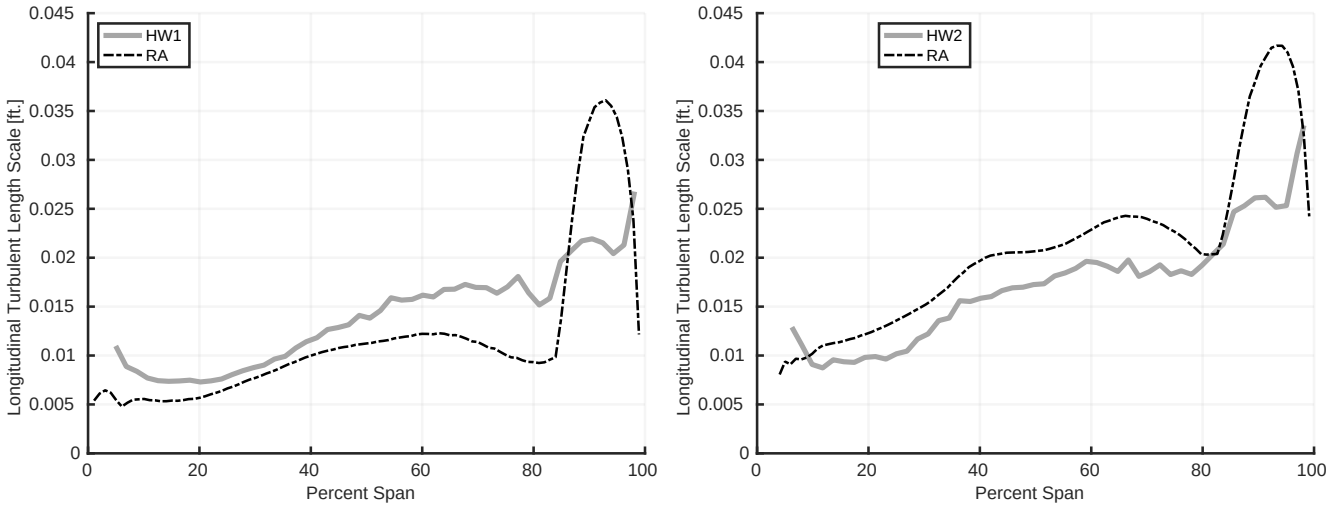
(h) TKE, Station 2.

Fig. 3 Average turbulence intensity in three directions and TKE for fan approach speed. Left: station 1. Right: station 2.



(a) Station 1. (b) Station 2.

Fig. 4 Average passage turbulent kinetic energy at midspan for the approach fan speed. Left: Station 1. Right: Station 2.



(a) Station 1 (b) Station 2

Fig. 5 Longitudinal length scale computed using the stationary method for fan approach speed.

III. Machine Learning Methods

A. Background

In recent years, the application of machine learning (ML) to problems in the general field of fluid dynamics has emerged. A main area of interest has been the use of ML to create new turbulence models, e.g. [21, 22]. However, other applications that focus on learning aspects of aerodynamics are more closely related to the current effort. In particular, some have applied machine learning to determine parameters related to aerodynamic performance such as the velocity field around an airfoil [23, 24] or properties of interest to helicopter rotors such as the surface pressure and aeroelastic response [25, 26]. Also of interest is past ML research that focused on predicting wake behavior downstream of objects. Image recognition was used to recreate the Karman vortex street flow downstream of a cylinder [27] and an airfoil at high angle of attack [28] as well as the flow downstream of a wind turbine [29, 30].

The fan wake in this research is similar to the wind turbine wake, however, here we strive to fully describe the circumferential variation of the mean and turbulence properties in the wake of the fan. In addition, we want the model to be responsive to the fan's characteristics. Existing wind turbine research focuses on studying a single turbine design using flow speed and direction as the independent variables, whereas this work seeks to include the fan's geometric parameters such as chordlength and stagger angle as inputs for the ML algorithm.

B. Data for Training and Testing

CFD RANS simulations, as described in Section II, generated the data for the ML algorithm development in this work. A single case consists of a fan geometry, a fan rotational speed and a mass flow through the fan. The input information related to the case includes: Ω the fan speed, $\dot{m}$ the mass flow into the fan, $(x_{te}, r_{te}, \theta_{te})$ the fan trailing edge, c the chord length, χ the local fan stagger angle, and $\delta_{press}, \delta_{suc}$ the trailing edge boundary layer thickness on the pressure and suction sides respectively, all at specified radial locations.

The output or target data comes from the CFD result downstream of the fan which is interpolated onto a cylindrical grid with 30 evenly spaced points in the axial direction extending the distance of the SDT interstage gap based on the baseline vane case, 30 radial locations, and 100 circumferential locations. In the radial direction, the first and last points lie in the hub and casing boundary layers. The flow at these positions is not wake-like. As such, the CFD data at only the 2nd through 29th radial locations are used in this study currently for both training/validation and testing. This effectively means that we are leaving out 6% at the hub and 6% at the tip. The same radial locations are used to define the input data related to the fan. Additional input information to the ML then is $(x_{new}, r_{new}, \theta_{new})$ the relative distance to the position of the wake data and finally the circumferential location of the output data point of interest $\theta_{percent}$. This gives an input dimension of 13.

Thus far, 268 CFD RANS data sets have been developed for this project. Four geometries related to the NASA Source Diagnostic Test (SDT) fan: CAD, approach hot, cutback hot and takeoff hot geometries; seven fan speeds; 7-10 mass flow rates at each fan speed. The set of fan speeds and mass flows align with tests performed at NASA. They are shown in Fig. 6.

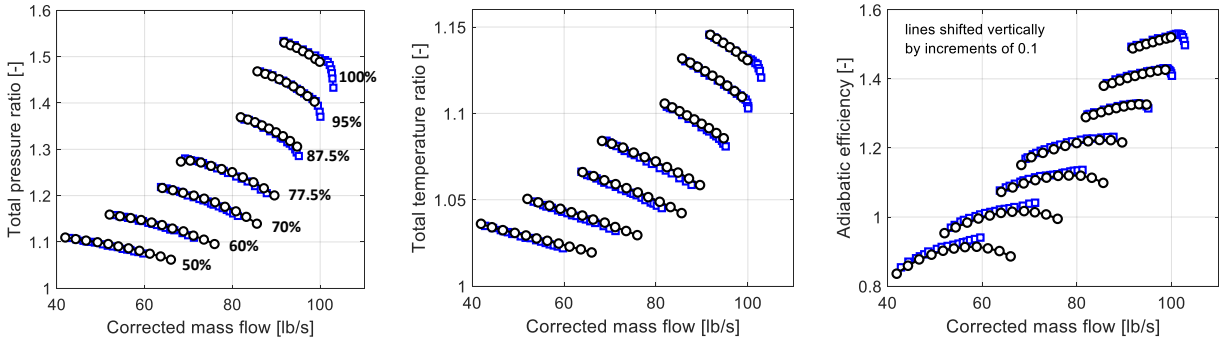


Fig. 6 Fan performance. Varying speed and mass flow. Black: current simulations. Blue: experimental values.

For the single-output methods, we applied two methods for utilizing the CFD data downstream of the fan. The first method randomly selects 80% of the entire database for training and validation leaving 20% for testing. Fig. 7 shows

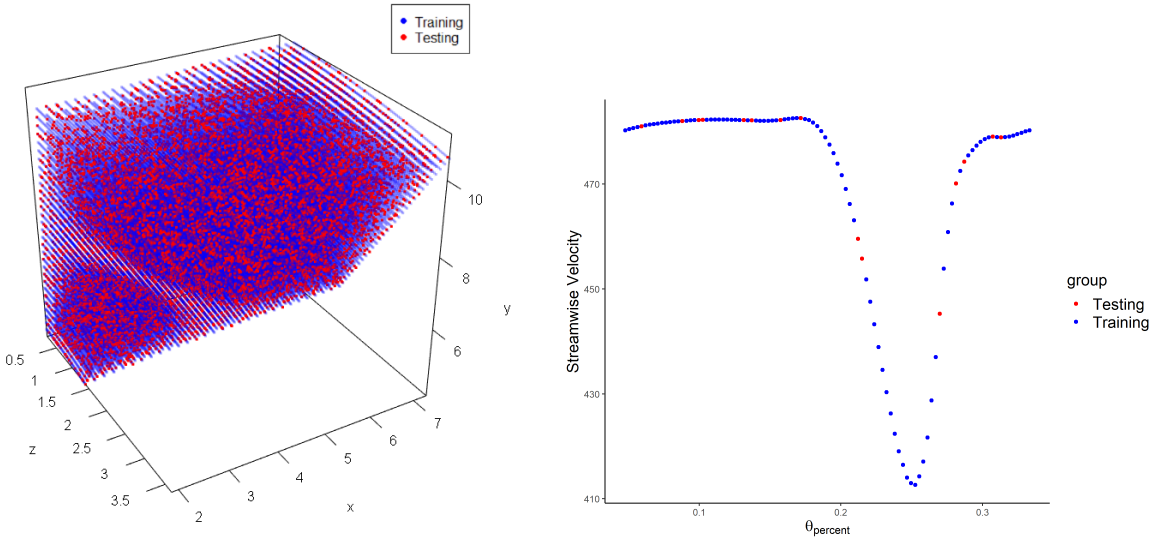


Fig. 7 Left: Random split for one of the CFD RANS data sets. Right: Random split when fixing i and k

an example of the training and testing points utilized from a single case as related to the mean streamwise velocity component. The 30 axial slices along the x-direction are seen, and it is clear that a single passage of data is what is available. The plot on the right shows the data as a function of circumferential location extracted from a single axial and radial position. The data for all of the cases would be represented as such. It is hard to imagine that the ML methods would fail to give the wake profile as shown on the right when 80% of the data (blue) have been used for the training because, as the figure shows, those 80% basically do a good job of defining the wake completely (the 20% (red) aren't really even needed). A more challenging test for the machine learning was then constructed in which a number of axial slices of the data were retained only for testing. For instance, out of the 30 axial slices in every fan run (fan geometry, fan speed, mass flow) the 5th, 10th, 20th, and 30th axial slices were made unavailable for training and validation. These slices would then be used in the testing phase. A final method for utilizing the data was to reserve an entire geometry or an entire fan speed (with all geometries) for testing. As this scenario best epitomizes how we hope to use the ML in the future, it is this method that will be discussed in the results section.

C. ML Accuracy measures

The machine learning methods are measured using two parameters : MSE and R^2. The mean squared error (MSE) measures the average of the squares of the errors. R-squared (R^2) measures the proportion of variation in the target variable that is explained by the predictors in a model. For a dataset with n observations, let Y denote the vector of the observed value of the target variable, and $\hat{Y}$ denote the vector of the fitted value of the target variable. Then, the MSE and the R^2 are calculated as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2}$$

$$R^2 = 1 - \frac{\sum(Y_i - \hat{Y}_i)^2}{\sum(Y_i - \bar{Y})^2} \tag{3}$$

The root mean square error which is the square root of Eq. (2) is often reported as opposed to the MSE value. It is also noted that the R^2 value can take on negative values. If a single predicted value, $\hat{Y}$, is quite incorrect, then the numerator in Eq. (3) will be larger than the denominator and lead to a negative R^2.

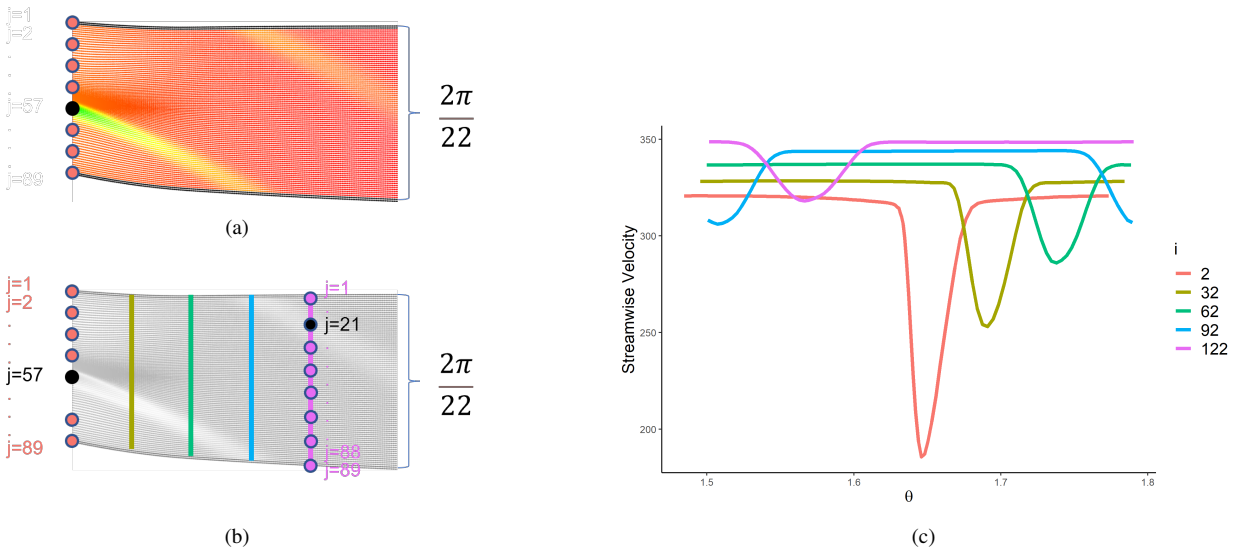


Fig. 8 Left: Streamwise velocity downstream of rotor at a given percent radial location as function of axial distance from rotor and circumferential position. Periodic condition (2 pi/22) seen as wake wraps based on axial distance from the rotor. Right: Streamwise velocity at different axial distances from the rotor, at fixed percent radial position. Colors align with colors on bottom left.

D. Single output

In the single output method, each fan input value to the model at a given radial location is independent and each point in the CFD grid downstream of the fan is independent. It was determined, that the ML could be used to learn the average passage profile of a parameter such as the streamwise velocity or the TKE value only if first, the wakes were "rectified." In the rectification phase the location of the minimum velocity position at a given axial location is learned. This is done by describing the wake centerline position using two functions of axial distance from the fan trailing edge: one for the theta position and one for the radial position. The understanding here though is that the fan trailing edge has been sampled at given set of percent span locations which are the same percent annular values used at locations downstream of the fan. Fig. 8 shows the streamwise velocity at a specific percent radial position in the gap. The circumferential extent of $2\pi/22$ represents the single fan blade passage of the SDT which had 22 fan blades. The rectification of the wakes such that the centerlines align as one moves axially downstream is shown in Fig. 9. The functions that provide the wake center position are learned using a polynomial regression model.

The fitted polynomial regression model for the radius of the lowest velocity points is

$$r_{new} \sim (r_{te} + x_{te}) \times poly(x_{new}, 3) \tag{4}$$

where r_{new} is the difference between the radius of the lowest velocity point and the radius of the trailing edge, r_{te} based on a matched percent radial location, x_{te} is the axial location of the trailing edge, and x_{new} is the difference between the axial location of the lowest velocity point and the axial location of the trailing edge. The term $poly(x_{new}, 3)$ includes three terms: x_{new}, x_{new}^2, and x_{new}^3. The $\times$ symbol in the formula is the indication of interaction (product of variables). For example, $X_1 \times X_2$ means we include X_1, X_2 and $X_1 \times X_2$ in the model.

Similarly, the theta location function is fitted using polynomial regression and is described as

$$th_{new} \sim poly(\Omega, 2) \times \chi \times r_{te} \times x_{te} \times poly(x_{new}, 2) \tag{5}$$

where th_{new} is the difference between the circumferetial position of the lowest velocity point and the circumferential position of the fan trailing edge at the same instant, Ω is the rotor speed, $poly(\Omega, 2)$ includes two terms: Ω and Ω^2 and χ which refers to the fan stagger angle at the radial loction of interest.

Once the wakes are aligned, the second phase of the ML focuses then on learning the wake flow parameters of interest: mean velocity vector, TKE or turbulence intensity and the turbulence length scale. Note that aligning the wakes also aligns the passagewise TKE distribution. Because the turbulence length scale is related to the TKE and the

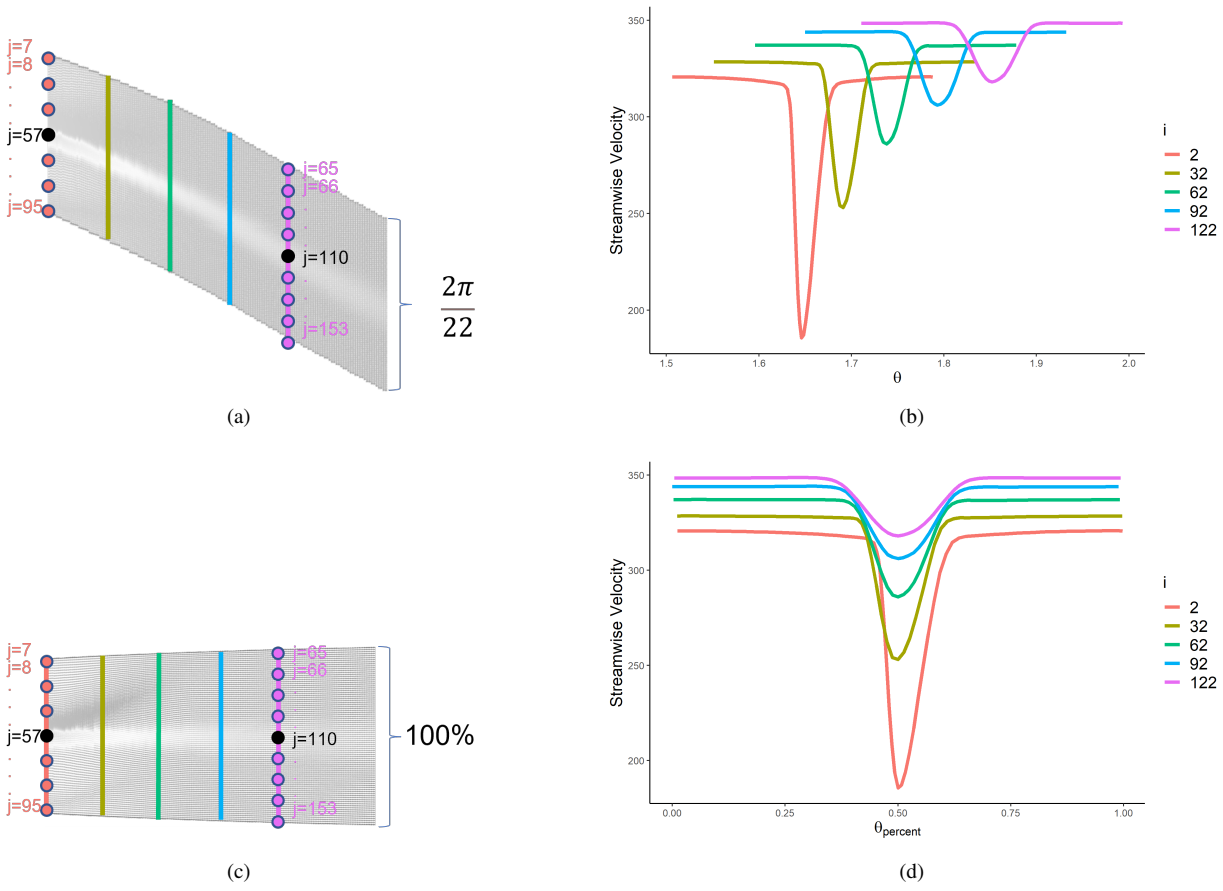


Fig. 9 Top: unrolled wakes. Bottom: centered wakes.(Final paper will have larger legend font.)

turbulent dissipation these are the turbulence parameters considered in this work. Because the overall averages can easily be obtained from the average passage distributions, emphasis was placed initially on learning the passagewise distributions of the parameters of interest. Several ML methods have been considered and their applicability for learning these wake parameters is explored in this research.

1. Spline Methods

Usually, a spline refers to a cubic spline. A cubic spline first contains a basis for a cubic polynomial, namely x, x^2, x^3. Then, a truncated power basis function per knot is added. A truncated power basis function is defined as

$$h(x,\xi) = (x-\xi)_+^3 = \begin{cases} (x-\xi)^3 \text{ if } x > \xi \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

where ξ is the knot. A cubic spline consists of a basis for a cubic polynomial and K truncated power basis functions (K is the number of knots). A cubic natural spline is a cubic spline with one additional constraint: the function is required to be linear at the boundary. The boundary is the region where X is smaller than the smallest knot, or larger than the largest knot. The R-programming language, has a built-in function $ns()$ for fitting a natural spline. It requires the specification of the degree of freedom (dof) which consists of two parts. The first part is the cubic polynomial which has a dof of 3. The second part is the number of knots K. In R, $ns(x, 5)$ is a cubic natural spline with $knots = dof - 3 = 5 - 3 = 2$. The default placement of the knots is in the quantile of the domain (e.g. three knots will be placed at the 25%, 50% and 75% quantile of the domain). However, one can also specify the knot placement. In this application, we chose the best locations based on trial and error using test MSE, adjusted R-squared value, and the residual plot.

The multivariate adaptive regression splines (MARS) model is a non-parametric regression technique and can be seen as an extension of linear models that captures the nonlinear relationships in the data by assessing knots in a piecewise manner. Each data point is viewed as a knot, and different linear regression models are fit using data points less than and greater than the knot value. Knot values that give models with the smallest error are chosen. A set of knots are added successively and are chosen in a similar fashion, and the piecewise linear models now results in a non-linear model overall. This method takes the tensor product of the spline function as the basis function. Once the full set of knots has been identified, we can sequentially remove knots that do not contribute significantly to prediction accuracy. This process is known as "pruning." The advantages of the MARS model are that it can process large amounts of data and deal with high dimensionality. In addition, it automatically includes and excludes terms during the pruning process, essentially performing automated feature selection, and supports variable interaction selection. Therefore, the method provides feedback on the relative importance of the various input parameters. This in and of itself was a very interesting facet of the method and the parameters that were shown to carry the most weight made sense physically for the few fan cases being considered.

The spline methods were tested using the original SDT data set which consisted of 3 different geometries each run at a single fan speed and single mass flow. The mean streamwise velocity was the focus of the initial tests. The methods showed promise with the random selection method of training/testing even for much more aggressive splits of the data (e.g. 40% train, 60% test). However, they did not perform well when entire axial slices of the data were reserved for testing. In addition, these methods are manually tuned and while this was possible for a small set of fan cases, automated methods must be used with large data sets. Therefore the spline methods were abandoned in favor of a gradient boosting decision tree and a deep neural network.

2. XGBoost and DNN methods

An extreme version of gradient boosting decision tree (GBDT) was developed. The version called XGBoost is used in this work. In essence, it is a combination of a large number of decision trees. Three XGBoost models were created: for the mean velocity, the TKE and the dissipation parameter ω. All of the models use the same input feature dimension of 13 as described in Section III.B. When we remove the 5th, 10th, 20th, and 30th axial slices of CFD data, this leaves about 15 million rows of data for training and provides about 4 million rows of data for testing. All three models used gbtree as the booster, regression with squared loss as objective, and the root mean square error as the evaluation metric. However, the parameters for the tree booster in each of the three models are different and are given in Table 1.

The parameter tuning process is done in five steps. For each step, 2-fold cross validation is used to select the best tuning parameters. In the first step, *eta* and, *nround* are tuned. *eta* ranges from some small number to 1, and *nround* ranges from dozens to thousands. The rest of the parameters are set to some default or manually chosen values. The

Table 1 XGBoost models.

Mean streamwise velocity model	
Parameter	Value
Learning rate (eta)	0.3
Maximum depth of tree (max_depth)	12
Minimum sum of instance weight need in a child (min_child_weight)	1
Subsample ratio of the training instances (subsample)	1
Subsample ratio of columns when constructing each tree (colsample_bytree)	1
Minimum loss reduction required to make further partition on a leaf node of the tree (gamma)	0
Number of trees (nround)	20
TKE model:	
Parameter	Value
Learning rate (eta)	0.3
Maximum depth of tree (max_depth)	10
Minimum sum of instance weight need in a child (min_child_weight)	1
Subsample ratio of the training instances (subsample)	0.7
Subsample ratio of columns when constructing each tree (colsample_bytree)	1
Minimum loss reduction required to make further partition on a leaf node of the tree (gamma)	0.5
Number of trees (nround)	500
ω **model:**	
Parameter	Value
Learning rate (eta)	0.15
Maximum depth of tree (max_depth)	10
Minimum sum of instance weight need in a child (min_child_weight)	4
Subsample ratio of the training instances (subsample)	0.8
subsample ratio of columns when constructing each tree (colsample_bytree)	1
Minimum loss reduction required to make further partition on a leaf node of the tree (gamma)	0.05
Number of trees (nround)	1000

idea of this step is to select a decent learning rate. In the second step, *max_depth* and *min_child_weight*, and *nround* are tuned. In this step, the parameters that control the complexity of the tree are tuned. In the third step, *subsample*, *colsample_bytree*, and *nround* are tuned. In this step, the parameters that involve randomness are tuned. In the forth step, *gamma*, and *nround* are tuned. In this step, *gamma* is tuned to avoid over-fitting. Finally, in the fifth step, *eta* and, *nround* are tuned again. In this step, *eta* is tuned again because the previous value is not optimal due to the other non-optimal parameter values. The tuning process is flexible which means it can be changed based on the tuning results. For example, when tuning the mean streamwise velocity model, after the first two steps, the R-squared was almost 1 while the rest of the parameters were at their default values. Then, the tuning process can stop. Also, the tuning process for each step can be done more than one time. It can be first tuned in a wider range, and then in a narrower range in order to save time. Notice that, *nround* is included in all the tuning steps. This is because one can easily observe how the performance of the model changes with *nround* when the other parameters are fixed. For the default or manually chosen values mentioned in the first step, one can use a grid random search to set them which makes this method much more automated than the MARS method.

The XGBoost method worked well when an axial slice of the wake data is reserved for testing. However, when an

Table 2 DNN model parameters for mean streamwise velocity.

Parameter	Value
Number of neurons in input layer	13
Number of hidden layers	2
Number of neurons in hidden layer-1	128
Number of neurons in hidden layer-2	8
Activation function at hidden layer	ReLU
Activation function at output layer	Linear
Optimizer	Adam
Learning rate	0.001
Objective function	Mean squared error
Metrics	Mean absolute error
Number of epochs	200
Mini batch size	512

entire geometry is reserved or a fan speed across all geometries is reserved, XGBoost did not reconstruct the mean streamwise velocity well. It did still perform well for the turbulence values: TKE and ω. This is demonstrated in the results section. This led to the consideration of a Deep Neural Network (DNN).

The DNN is a neural network with multiple hidden layers, also known as Multi-layer perceptron (MLP). It contains three types of layers: input, hidden, and output. For DNN, input feature normalization is necessary in order to make learning quicker. With different ranges of values, gradients may end up taking a longer time. Therefore, the 13 input parameters are scaled using the general process: $X_{scale} = \dfrac{X - \bar{X}}{\sigma_X}$ where the overbar indicates the average and σ is the standard deviation. DNN was only used to model the mean streamwise velocity because this was the case in which XGBoost failed. The model parameters for this case are given in the Table 2.

The tuning process for the DNN is completed in three steps. In the first step, the number of layers is set. In the second step, some trials on the model are run by changing the number of neurons in the hidden layers, the optimizer, the learning rate, and so on. In this step, the best values of MSE and R^2 may not be achieved but a decent or acceptable values are. This means that at first, an $R^2 > .7$ is sought. Then, improvements to the R^2 and accompanying MSE are sought. In the third step, parameters are tuned around the value selected in the second step. All parameters are not tuned simultaneously. at Usually, the optimizer and the activation functions are set first and then the rest of the parameters are tuned. If the result is not acceptable, a change is made to the the number of layers, the optimizer, the activation function, the number of dropout layer, etc. The training stops when the validation loss does not improve for 20 epochs.

E. Multi-output method

The single output methods described thus far assume that each point in the CFD grid downstream of the rotor and each rotor input value at a given radial location are independent. Clearly this is not physical given that the radial slices along the rotor are most definitely connected. As well, there is radial connectedness (as well as circumferential connectedness) in the region downstream of the rotor. This prompted the development of a multi-output machine learning method. Fig. 10 shows the basic difference between the single and multi-output methods.

A Convolution Neural Network (CNN) is a neural network that is commonly applied to analyze images. CNNs are different from multi-layer perceptrons. Instead of using a fully connected layer to learn global patterns, they use convolution layers to learn local patterns in the feature space. They can learn the spatial hierarchies of patterns. The input for a CNN is usually an image of similar dimension to the desired output image. In this application however, the input consists of the numeric scalars tied to the fan trailing edge in the radial direction together with fan speed and mass flow. Therefore, an encoder which would be applied from an input image to a set of input features is not needed. Only the decoder part of the CNN is requred for our model. The input features are used to build the latent space, and then

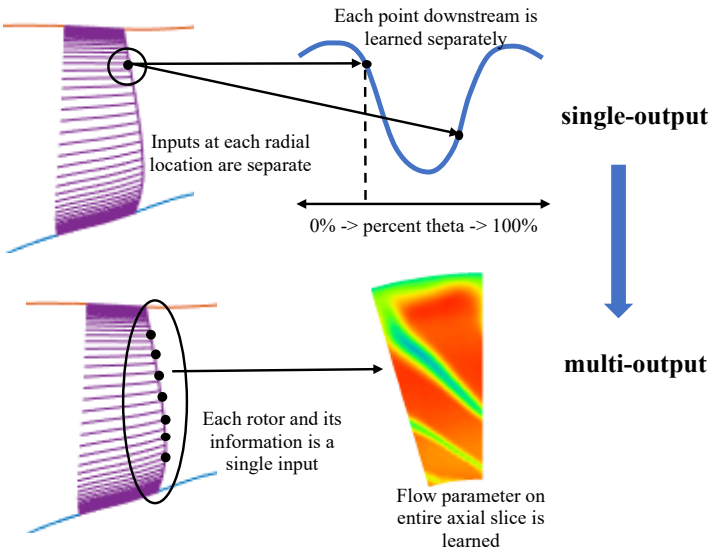


Fig. 10 Single vs multi output machine learning idea. Multi output method does not require rectification of the wakes (centering) before learning the wakes.

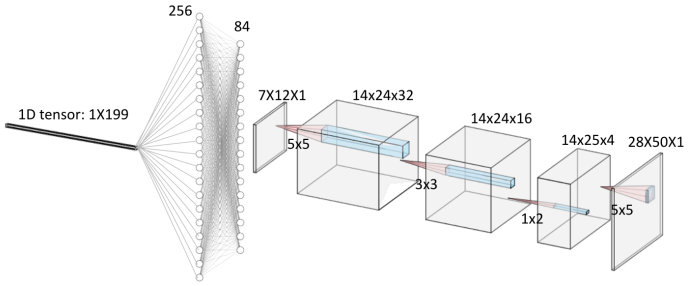


Fig. 11 Decoder part of CNN - focus of current multi-output method.

deconvolution layers are used to reconstruct the output image. Fig. 11 shows a typical CNN decoder architecture. The fan input parameters are grouped into the long 1D tensor, then 2 fully connected layers are created followed by several transposed convolution layers until finally the resulting passagewise picture of the desired parameter is found. For the CNN training, it was determined that the CFD interpolated grids which had at each axial location 100 circumferential and 30 radial points, were unnecessarily large. As such, the CNN is being used with interpolated grids that have just 50 circumferential points and 30 radial points. Again, only the 2nd through 29th radial points are used in this study. Hence the final output from the CNN is shown as being 28x50x1 to correspond to the 28 radial locations and 50 circumferential locations. Like DNN, all inputs are normalized. Again, separate models for the mean streamwise velocity, the TKE and the ω are created. The parameters for the three models are given in Table 3 - 8.

The tuning process was done in three steps. In the first step, the number of deconvolution layers is set by starting with 3 layers and adding layers until acceptable performance is achieved. In the second step, the number of feature maps, learning rate, and mini batch size are tuned together. In the third step, the size of the kernel and parameters of the activation function are tuned together. The idea of using two transposed 2D convolution layers to double the feature map size twice came up by trial and error. The training stops when validation loss does not improve for 20 epochs.

IV. Results

In this section, the ability of ML to predict the passagewise distributions of the mean streamwise velocity, the TKE and the turbulent dissipation parameter (ω) is demonstrated. 268 cases of fan geometry, fan speed and fan mass flow

Table 3 CNN architecture for the mean streamwise velocity.

Structure	Numbers of feature maps	Size of feature map	Size of kernel	Stride
Fully connected layer-1	256	1×1	/	/
Fully connected layer-2	84	1×1	/	/
Transposed 2D convolution layer-1	64	7×12	4×4	1×1
Transposed 2D convolution layer-2	64	7×12	4×4	1×1
Transposed 2D convolution layer-3	64	14×24	4×4	2×2
Transposed 2D convolution layer-4	32	14×24	4×4	1×1
Transposed 2D convolution layer-5	16	14×24	4×4	1×1
Transposed 2D convolution layer-6	32	14×25	1×2	1×1
Transposed 2D convolution layer-7	4	28×50	4×4	2×2
Transposed 2D convolution layer-8	1	28×50	4×4	1×1

Table 4 CNN model parameters for the mean streamwise velocity.

Parameter	Value
Activation function at hidden layer	LReLU(α = 0.05)
Activation function at output layer	Linear
Optimizer	Adam
Learning rate	0.01
Objective function	Mean squared error
Metrics	Mean absolute error
Number of epochs	106
Mini batch size	64

Table 5 CNN architecture for TKE.

Structure	Numbers of feature maps	Size of feature map	Size of kernel	Stride
Fully connected layer-1	256	1×1	/	/
Fully connected layer-2	84	1×1	/	/
Transposed 2D convolution layer-1	32	7×12	6×6	1×1
Transposed 2D convolution layer-2	64	7×12	3×3	1×1
Transposed 2D convolution layer-3	64	14×24	4×4	2×2
Transposed 2D convolution layer-4	16	14×24	5×5	1×1
Transposed 2D convolution layer-5	32	14×24	3×3	1×1
Transposed 2D convolution layer-6	32	14×25	1×2	1×1
Transposed 2D convolution layer-7	2	28×50	3×3	2×2
Transposed 2D convolution layer-8	1	28×50	3×3	1×1

Table 6 CNN model parameters for TKE.

Parameter	Value
Activation function at hidden layer	LReLU(α = 0.05)
Activation function at output layer	Linear
Optimizer	Adam
Learning rate	0.001
Objective function	Mean squared error
Metrics	Mean absolute error
Number of epochs	325
Mini batch size	128

Table 7 CNN architecture for ω.

Structure	Numbers of feature maps	Size of feature map	Size of kernel	Stride
Fully connected layer-1	256	1×1	/	/
Fully connected layer-2	84	1×1	/	/
Transposed 2D convolution layer-1	32	7×12	4×4	1×1
Transposed 2D convolution layer-2	32	7×12	4×4	1×1
Transposed 2D convolution layer-3	16	14×24	4×4	2×2
Transposed 2D convolution layer-4	32	14×24	4×4	1×1
Transposed 2D convolution layer-5	8	14×24	4×4	1×1
Transposed 2D convolution layer-6	32	14×25	1×2	1×1
Transposed 2D convolution layer-7	4	28×50	4×4	2×2
Transposed 2D convolution layer-8	1	28×50	4×4	1×1

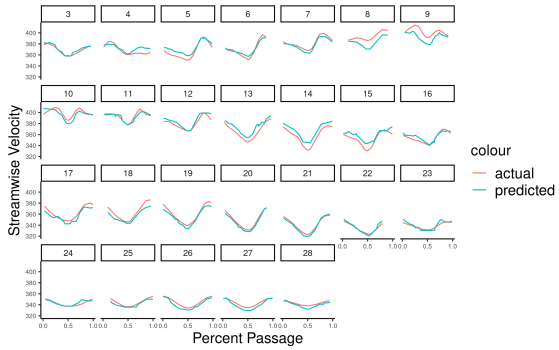
Table 8 CNN model parameters for ω.

Parameter	Value
Activation function at hidden layer	LReLU(α = 0.1)
Activation function at output layer	Linear
Optimizer	Adam
Learning rate	0.001
Objective function	Mean squared error
Metrics	Mean absolute error
Number of epochs	778
Mini batch size	128

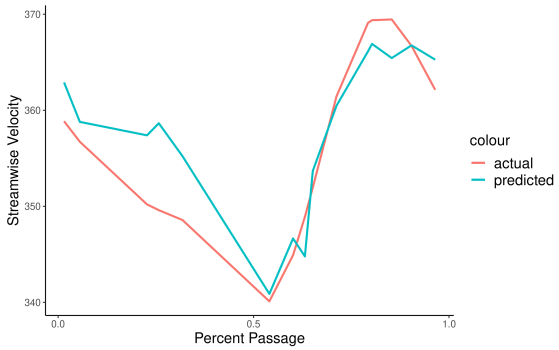
were used for training/testing. However, the four geometries used are not significantly different and the fan wakes associated with the different geometries are very similar. As such, reserving one geometry for testing is not a difficult test for the method and is not discussed here. The outcome in terms of MSE and R-squared values for many of the cases described in this section are summarized in Table 9 at the end of this section.

A. Single Output

The first attempt to train a machine learning model reserved several axial locations for testing. In particular, the 30th axial location was reserved which coincides with the location just upstream of the SDT FEGV where the input are required for the low-order noise calculation. In addition to reserving several axial locations for testing, the training data are further split into 80% for training and 20% for validation. Fig. 12 shows the reconstruction of the streamwise velocity at this axial location provided by the ML algorithm. The streamwise average passage velocity is shown at the 30th axial location. Each numbered panel refers to a radial location with 2 near the hub and 28 near the tip. The case shown on the left for the fan speed 6328.5rpm at the lowest mass flow 42lbm/s gave the lowest performance metrics with a test MSE of 11.51 and an R-squared value of 0.999. The TKE and ω were similarly well reconstructed under these testing conditions. Interestingly, the parameter with most relevance for the mean flow wake profile is the mass flow. However for the TKE and turbulence dissipation parameter, the most relevant parameter is the relative circumferential position of the point. Example relevance plots are shown in Fig. 13. The TKE relates to the boundary layer thickness strongly which makes physical sense. The turbulence dissipation however does not.



(a) Rotor: APPCold, fan speed: 6328.5 rpm, mass flow: 42 lb/s, i: 30



(b) k: 16

Fig. 12 Reconstructions of the streamwise wake at axial location 30 (just upstream of SDT FEGV) using the single output method and random train/test method.

The second test of the machine learning algorithm utilized a different training/test selection method. Full rotor speeds were excluded from the training, one at a time. The prediction of that speed from the ML trained using the other speeds was analyzed based on the MSE and R-squared values. When an entire rotor speed was left out, the XGBoost performed as might be expected when a tree basis is utilized. The predictions had a "nearest neighbor" type of outcome. This meant that there were large differences between the predicted and actual wake profiles as shown in Fig. 14. The wake shapes are rather close, but the average value is quite off. The issue is not as severe for the TKE and ω values because their overall mean values from hub to tip do not change like the streamwise value does. In order to move away from the decision tree type behavior, a Deep Neural Net (DNN) was then considered. The DNN gave streamwise wake predictions that had a better overall mean value, but the wake shape was not well captured as seen in Fig. 15.

For the application to low-order broadband noise prediction, the wake shape is not necessary. Instead, just the overall mean value (from hub to tip) is needed. (Some low-order broadband methods require the TKE shape in the wake but not the method utilized by the authors.) Therefore, the DNN was tested for learning just the overall mean value. Fig. 16 shows the DNN results for the overall mean value of the three parameters vs radial location. The DNN performs very well for determining the overall mean value. The figure shows the worst performing predictions (out of all the rotor speeds and mass flow combinations).

The results show that the single-output ML algorithms can provide a surrogate model for the wake. However, both the DNN and XGBoost/decision tree algorithms must be applied: the DNN to obtain the overall mean value from hub to

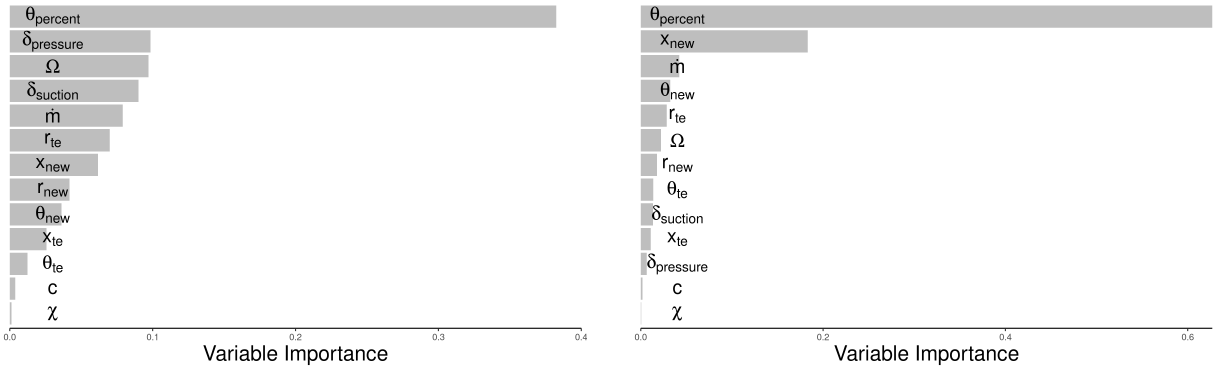


Fig. 13 Left: Variable importance for the TKE. Right: Variable importance for the turbulence dissipation parameter.

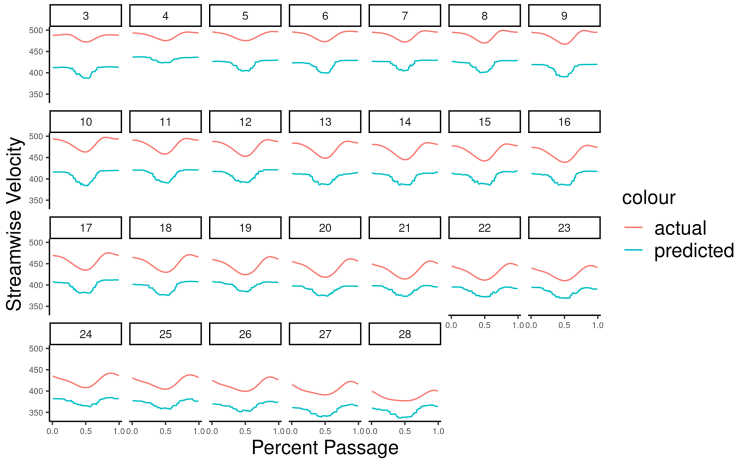
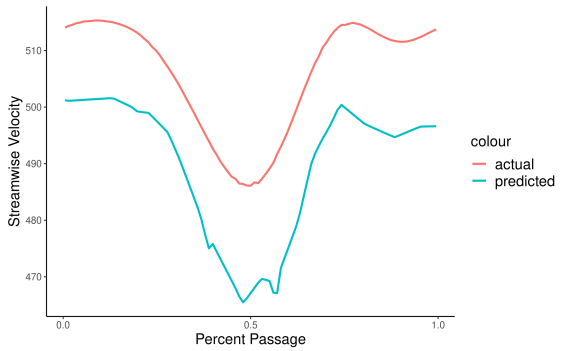


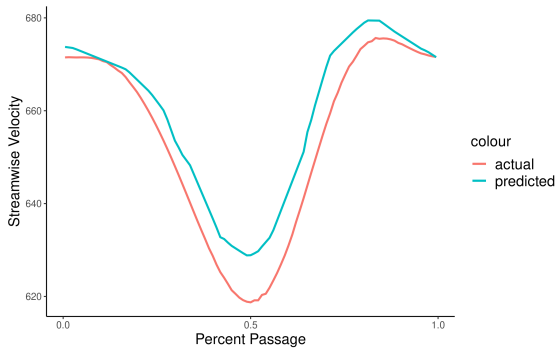
Fig. 14 Reconstructions of the wake using XGBoost when data for a full fan speeds are reserved for testing.

tip and the XGBoost to determine the passage shape.

The method of reserving one, two or three fan speeds for testing to challenge the machine learning algorithm can be unintentionally biased. As such, cross validation should be used. For the present data set with 7 fan speeds, a 7-fold cross validation can be performed. Using this process leads to slightly different ML models than the ones given in Tables 1-2. However, the overall outcome remains the same in that DNN captures the mean value well and XGBoost captures the passagewise shape well.

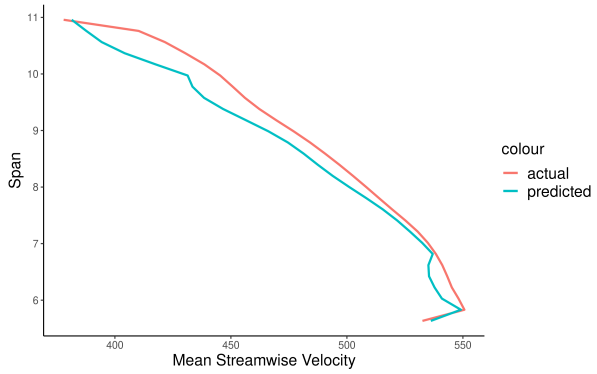


(a) Rotor: TOHot, fan speed: 7594.2 rpm, mass flow: 66.4 lb/s, i: 30, k: 14

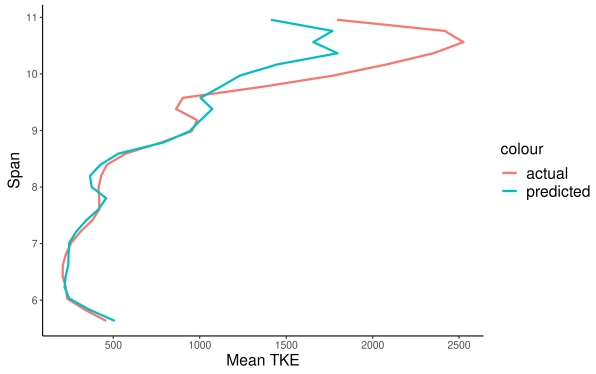


(b) Rotor: CBHot, fan speed: 11074.875 rpm, mass flow: 83.6 lb/s, i: 30, k: 14

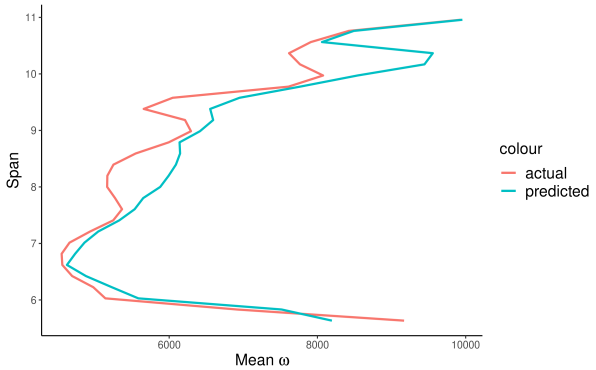
Fig. 15 Reconstructions of the wake using DNN when data for a full fan speed are reserved for testing.



(a) Streamwise Velocity. Rotor: APPHot, fan speed: 7594.2 rpm, mass flow: 66.4 lb/s, i: 30



(b) TKE. Rotor: APPHot, fan speed: 7594.2 rpm, mass flow: 52 lb/s, i: 30

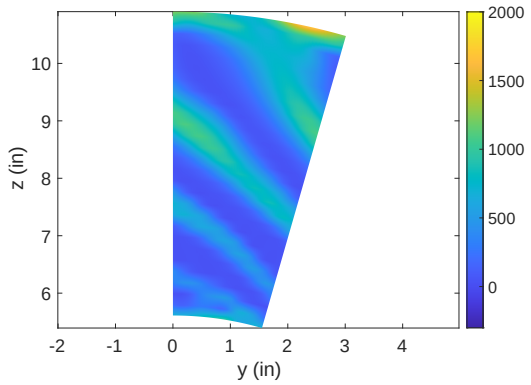


(c) ω. Rotor: APPHot, fan speed: 7594.2 rpm, mass flow: 52 lb/s, i: 30

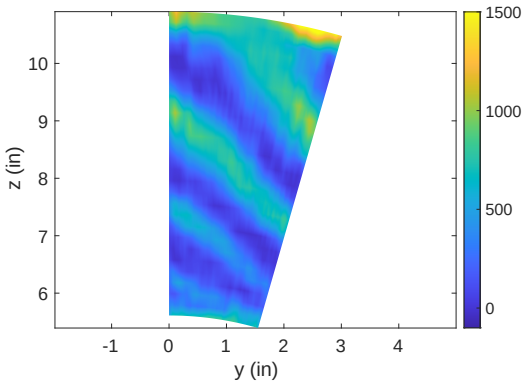
Fig. 16 Reconstructions of the mean parameters using DNN when data for a full fan speeds are reserved for testing. Worst cases are shown.

B. Multi-output

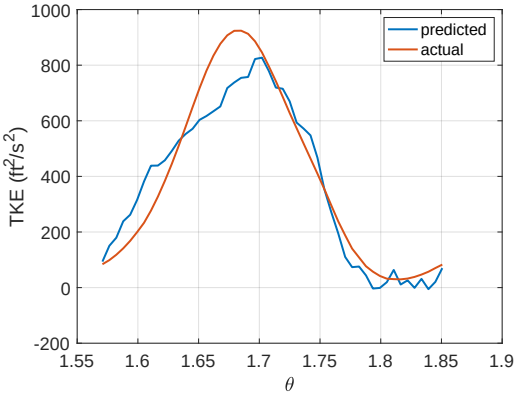
The multi-output ML method captures the connectedness between the input values at radial slices on the fan and reconstructs the flow parameter simultaneously from hub to tip and across the passage at an axial location. The number input parameters therefore decreases to 10 (i.e. Ω, $\dot{m}$, x_{te}, θ_{te}, r_{te}, $\delta_{pressure}$, $\delta_{suction}$, c, χ, & x_{new}) We consider first the TKE and the dissipation parameter ω. Suitable CNNs have been created that can predict an axial slice that has been omitted from the training data. Fig. 17 shows a case with the worse reconstruction. It gives a MSE of 41013 and an R^2 of 0.96. Similar good prediction can be found for ω.



(a) Actual TKE test data



(b) CNN prediction of TKE



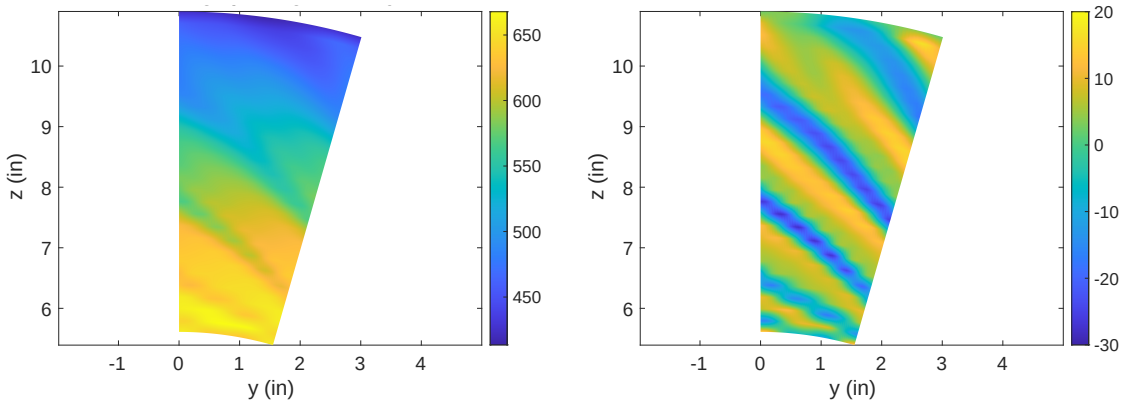
(c) Mid span comparison

Fig. 17 ML results for the TKE using CNN. Rotor: APPCold, fan speed: 8859.9 rpm, mass flow: 73.6 lb/s, i: 30.

When the CNN is applied to the streamwise velocity, an issue arises because of the large change in the mean value from hub to tip. One can see in Figs. 14 and 15 that as the radial locations moves from hub to tip (denoted by the k counter in the figures), the mean value changes greatly. At each k there is a similar wake deficit that is smaller than the overall change in the mean velocity from hub to tip. The ML CNN architectures tested have a difficult time resolving both the wake deficit and the overall change in the mean wake value from hub to tip. Therefore, the data are split into two parts (the mean and the variation across a passage) and two ML algorithms are used as described in Section III.E. Fig. 18 shows the difference between the original image of the wake and the image with the background mean removed. The figure on the right is much more like the TKE variation shown in Fig. 17 and as such the ML can perform a prediction well.

The mean background value and the velocity deficit are then predicted separately using the ML. An example outcome when several axial slices are reserved for testing is shown in Fig. 19. The MSE and R^2 values for these cases is : 70.1 and 0.995, and 14.51 and 0.95.

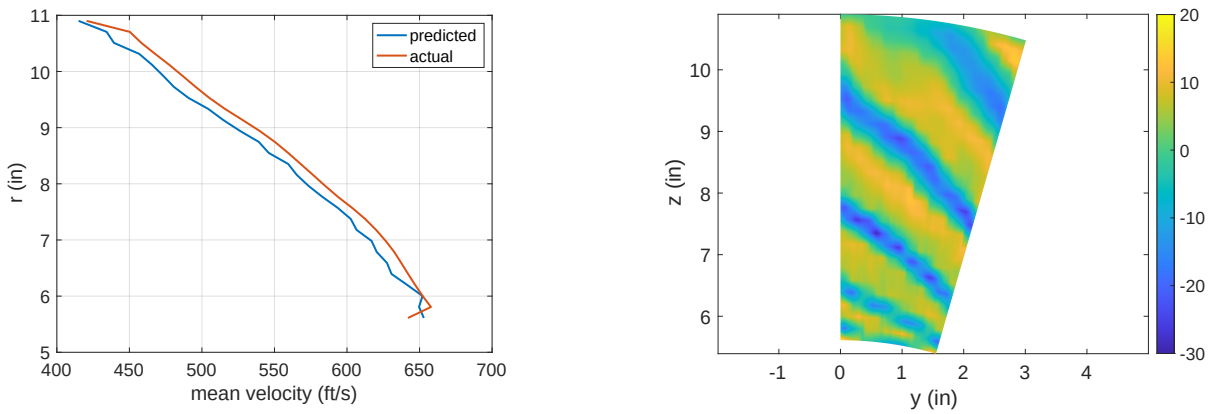
The CNN was also run using the second method for choosing test data. Here a full rotor speed with all associated



(a) Mean streamwise velocity

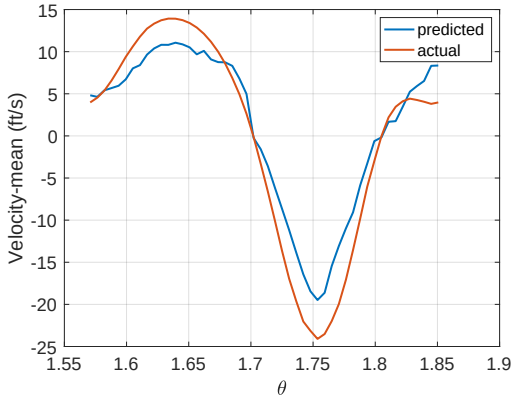
(b) Background mean value removed

Fig. 18 Left mean streamwise velocity at 30th axial slice. Right streamwise velocity with background mean value removed. Rotor: CBHot, fan speed: 7594.2 rpm, mass flow: 76 lb/s, i: 30.



(a) DNN prediction of overal mean.

(b) CNN prediction of wake deficit



(c) Mid span comparison of deficit

Fig. 19 ML results for mean streamwise velocity. Left: Overall mean of streamwise velocity hub to tip from DNN. Right: Velocity deficit (mean removed) from CNN. Rotor: CBHot, fan speed: 7594.2 rpm, mass flow: 76 lb/s, i: 30.

fan geometries and mass flow cases was reserved for testing. Each speed was left out sequentially. The results are shown in the table. As in the case of the single input methods, a more robust cross-validation method of determining the final CNN architecture using the 7-folds available due to the 7 fan speeds will be considered in the future.

Table 9 Performance of the ML models for different test cases.

Model	Target variable	Method for training and testing	MSE	R^2
XGBoost	Mean streamwise vel	Random selection	11.51	0.999
XGBoost	TKE	Random selection	4776.7	0.994
XGBoost	ω	Random selection	1643150	0.986
DNN	Mean streamwise vel	Leave out rotor speed	427.378, 220.699, 38.162, 32.172, 57.962, 55.258, 42.921	0.835, 0.929, 0.990, 0.990, 0.970, 0.949, 0.951
DNN	TKE	Leave out rotor speed	34341.37, 29214.42, 82932.59, 311472.64, 12695.66, 35980.85, 26664.88	0.917, 0.947, 0.847, 0.633, 0.975, 0.951, 0.969
DNN	ω	Leave out rotor speed	2594728, 1112382, 1682813, 3035567, 1953510, 3963403, 89958659	0.957, 0.986, 0.984, 0.974, 0.987, 0.976, 0.628
CNN	Mean streamwise vel	Random selection	299	0.98
CNN	Mean hub to tip streamwise velocity	Random selection	70.1	0.995
CNN	Velocity deficit mean removed	Random selection	14.51	0.95
CNN	TKE	Random selection	41013	0.96
CNN	ω	Random selection	9714263	0.95
CNN	Mean hub to tip streamwise velocity	Leave out rotor speed	186.879, 57.474, 8795.737, 177.146, 49.491, 48587.365, 254.816	0.934, 0.983, -1.106, 0.946, 0.976, -40.427, 0.593
CNN	Velocity deficit mean removed	Leave out rotor speed	20.780, 27.758, 34.469, 33.210, 27.038, 28.324, 47.075	0.868, 0.871, 0.876, 0.895, 0.920, 0.922, 0.872

V. Conclusion

This research shows initial success with using machine learning as a surrogate model for fan wake flow. One consistent outcome from the single and multi output ML methods considered is that the streamwise mean flow average passage behavior is best learned in two steps: the overall mean value from hub to tip and the wake passagewise profile as a variation from the mean. The models were trained on the BU SCC processors with 28 cores. The trainings for the three XGBoost models took about 5 minutes, 1 hour, and 2 hours respectively. The trainings for the CNN models took about 30 minutes to 1 hour. When the training is done, the prediction of a wake parameter for a new set of inputs takes seconds. The CNN uses the blade as a connected surface and treats each axial slice as connected from hub to tip as well as circumferentially. Thus, while this method does not give the user feedback on feature importance, its overall representation of the problem is more physical and will be developed further in the future. In particular, cross validation using multiple folds based on removal of fan speeds will be completed. In addition, the robustness of the method when presented with fan geometries encompassing a much wider design space will be tested.

Acknowledgments

This research was partially funded by the U.S. Federal Aviation Administration Office of Environment and Energy through ASCENT, the FAA Center of Excellence for Alternative Jet Fuels and the Environment, project 75 through FAA Award Number 13-C-AJFE-BU under the supervision of Christopher Dorbian. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the FAA. Other funding has come from RTRC and BU Mechanical Engineering Department, and the BU Master of Science Statistics Program. The simulation and data analysis related to the LBM/VLES of the SDT was supported by the Ohio Aerospace Institute on a grant from the AeroAcoustics Research Constortium. The authors thank Eric Kolaczyk, director of BU's Hariri Institute of Computing and Computational Science and Engineering, and Masanao Yajima director of BU's MS in Statistics Program's consulting for their help in launching this research.

References

[1] Moreau, S., "Turbomachinery Noise Predictions: Present and Future," *Acoustics*, Vol. 1, 2019, pp. 92–116.

[2] Guerin, S., Kissner, C., Seeler, P., Blázquez, R., Carrasco Laraña, P., Laborderie, H., Lewis, D., Chaitanya, P., Polacsek, C., and Thisse, J., "ACAT1 Benchmark of RANS-Informed Analytical Methods for Fan Broadband Noise Prediction: Part II—Influence of the Acoustic Models," *Acoustics*, Vol. 2, 2020, pp. 617–649.

[3] Grace, S., "Further Investigations into a Low-Order Model of Fan Broadband Noise," *AIAA Paper No. AIAA 2015-3283*, 2015, pp. 1–15.

[4] Sanjose, M., Daroukhb, M., Laborderie, J., Moreau, S., and Mann, A., "Tonal noise prediction and validation on the ANCF rotor–stator configuration," Vol. 63, No. 6, 2015, pp. 552–562.

[5] Leonard, T., Sanjose, M., and Moreau, S., "Large Eddy Simulation of a scale-model turbofan for fan noise source diagnostic," *AIAA Paper No. 2016-3000*, 2016, pp. 1–24.

[6] Casalino, D., Hazir, A., and Mann, A., "Turbofan Broadband Noise Prediction using the Lattice Boltzmann Method," *AIAA Journal*, Vol. 56, No. 2, 2018. doi:10.2514/1.J055674.

[7] Axstream, "AxSTREAM Software Suite – Turbomachinery Design," Tech. Rep. Version 3.9.12, SoftInWay, Incorporated, 2021.

[8] Podboy, G. G., Krupar, M. J., Helland, S. M., and Hughes, C., "Steady and Unsteady Flow Field Measurements Within a NASA 22-Inch Fan Model," Tech. Rep. TM-2003-212329, NASA, 2003.

[9] Podboy, G. G., Krupar, M. J., Helland, S. M., and Hughes, C. E., "Steady and unsteady flow field measurements within a NASA 22 inch fan model," *AIAA Paper No. 2002-1033*, 2002, pp. 1–30. doi: 10.2514/6.2002-1033.

[10] Maunus, J., Grace, S. M., and Sondak, D. L., "Effect of Rotor Wake Structure on Fan Interaction Noise," *AIAA Journal*, Vol. 50, No. 4, 2012, pp. 818–831.

[11] Posson, H., Moreau, S., and Roger, M., "Broadband noise prediction of fan outlet guide vane using a cascade response function," *Journal of Sound and Vibration*, Vol. 330, 2011, pp. 6153–6183.

[12] Grace, S., "Fan Broadband Interaction Noise Modeling Using a Low-Order Method," *Journal of Sound and Vibration*, Vol. 346, 2015, pp. 402–423.

[13] Grace, S., Gupta, A., Gonzalez-Martino, I., and Casalino, D., "Statistics and structure of turbulence in a fan/FEGV interstage and their aeroacoustic implications," *AIAA Paper No. AIAA 2018-4186*, 2018, pp. 1–18.

[14] Pope, S. B., *Turbulent Flows*, Cambridge University Press, Cambridge, UK, 2000.

[15] Ni, R. H., "A Multiple-Grid Scheme for Solving the Euler Equations," *AIAA Journal*, Vol. 20, No. 11, 1982, pp. 1565–1571.

[16] Wilcox, D. C., *Turbulence modeling for CFD*, DCW industries, Canada, 1998.

[17] Winkler, J., Reimann, C. A., Reba, R. A., and Gilson, J., "Turbofan Inlet Distortion Noise Prediction with a Hybrid CFD-CAA Approach," *AIAA Paper No. 2014-3102*, 2014.

[18] Winkler, J., Reimann, C. A., Gumke, C. D., Ali, A. A., and Reba, R. A., "Inlet and Aft Tonal Noise Predictions of a Full-Scale Turbofan Engine with Bifurcation and Inlet Distortion," *AIAA Paper No. 2017-3034*, 2017.

[19] Prasad, D., Li, D., and Topal, D., "Dispersion, dissipation and refraction of shock waves in acoustically treated turbofan inlets," *Journal of Sound and Vibration*, Vol. 352, 1982, pp. 46–62.

[20] Giles, M., "Non-reflecting boundary conditions for Euler equation calculations," *AIAA Journal*, Vol. 28, 1988, pp. 2050–2058.

[21] Duraisamy, K., and nad H. Xiao, G. I., "Turbulence Modeling in the Age of Data," *Annual Reviews in Fluid Mechanics*, Vol. 51, 2019, pp. 357–377.

[22] Mohan, A., Daniel, D., Chertkov, M., and Livescu, D., "Compressed Convolutional LSTM: An Efficient Deep Learning framework to Model High Fidelity 3D Turbulence," , 2019.

[23] Bhatnagar, A., Afshar, Y., Pan, S., Duraisamy, K., and Kaushik, S., "Prediction of aerodynamic flow fields using convolutional neural networks," *Computational Mechanics*, Vol. 64, 2019, pp. 1–30.

[24] Zhang, Y., Sung, W. J., and Mavris, D. N., "Application of Convolutional Neural Network to Predict Airfoil Lift Coefficient," *AIAA Paper No. 2018-1903*, 2018, pp. 1–9.

[25] Martinez, D., Sitaraman, J., Brewer, W., Rivera, P., and Jude, D., "Machine Learning Based Aerodynamic Models For Rotor Blades," *VFS Transformative Vertical Flight*, 2020, pp. 1–13.

[26] Chatterjee, T., Essien, A., Ganguli, R., and Friswell, M., "The stochastic aeroelastic response analysis of helicopter rotors using deep and shallow machine learning," *Neural Computing and Applications*, Vol. 33, 2021, pp. 16809–16828.

[27] Fukami, K., Fukagata, K., and Taira, K., "Super-resolution reconstruction of turbulent flows with machine learning," *Journal of Fluid Mechanics*, Vol. 870, 2019, pp. 106–120.

[28] Omata, N., and Shirayama, S., "A novel method of low-dimensional representation for temporal behavior of flow fields using deep autoencoder," *AIP Advances*, Vol. 9, No. 015006, 2019, pp. 1–14.

[29] Renganathan, S. A., Maulik, R., Letizia, S., and Iungo, G. V., "Data-Driven Wind Turbine Wake Modeling via Probabilistic Machine Learning," *Neural Computing and Applications*, 2021, pp. 1–15.

[30] Tia, Z., Denga, X., and Yang, H., "Wake modeling of wind turbines using machine learning," *Applied Energy*, Vol. 257, 2020, pp. 1–17.