Surrogate Modeling a Computational Fluid Dynamics-based Wind Turbine Wake Simulation using Machine Learning

Brett Wilson*, Sarah Wakes†, Michael Mayo*
*Department of Computer Science
University of Waikato
Hamilton, New Zealand
bkw5@students.waikato.ac.nz, michael.mayo@waikato.ac.nz
†Centre for Materials Science and Technology
University of Otago
Dunedin, New Zealand
sarah.wakes@otago.ac.nz

***Abstract*—The wind farm layout optimisation problem involves finding the optimal locations for wind turbines on a wind farm site in order to minimise the so-called "wake effect". The wake effect is the effect of turbulence on wind velocity produced by a turbine's rotating blades. This results in reduction in power production and increased fatigue in downstream turbines inside the wake. This paper uses wind velocity data produced from expensive Computational Fluid Dynamics (CFD) simulations of a rotating wind turbine at various incoming wind speeds to generate ground truth wake data, and explores the ability of machine learning algorithms to create surrogate models for predicting the reduced-velocity wind speeds inside a wake. In an extensive evaluation, we show that (i) given data from a CFD simulation, we can construct a model to interpolate wind velocity inside the wake at any arbitrary 3D point with high levels of accuracy; and (ii) given data from several CFD simulations (the training data) we can also accurately predict wind velocities in the wake of CFD simulations that we have not yet run (i.e. we can extrapolate to simulations where the incoming wind speeds are different to those in the training data). The net effect of these findings are that they pave the way towards the construction of novel and improved wake models for wind turbines, which in turn can be incorporated into existing algorithms for solving wind farm layout optimisation problems more accurately.**

***Index Terms*—surrogate model, wind turbine, wind farm layout optimisation problem, machine learning, computational fluid dynamics**

I. Introduction

The Wind Farm Layout Optimization problem involves finding the optimal positions for wind turbines on a wind farm site. Current Metahueristic based methods make use of a combination of turbine specifications and parameters, mathematical models and empirically produced power production equations to estimate the energy output of a real wind farm [15]. The overarching variable in any optimisation function is wind speed - this is what used to determine the power generated. Therefore, accurate predictions of wind speeds at specific points across the volume of the site are needed. In this paper, Computational Fluid Dynamics (CFD) was used to simulate a full scale rotating wind turbine blade with fluid (air) at various wind speeds flowing past the turbine. The wake effect can be observed and leads to decrease in wind speeds, as expected. Wind speed at specific x, y and z (3D) coordinates were sampled and used as input to common Machine Learning regression algorithms to create different surrogate models. This was needed as each individual CFD experiment takes approximately 8 hours to complete, so it is not feasible to continuously repeat these simulations inside a metaheuristic optimiser.

Several surrogate models that use common Machine Learning regression algorithms were created and compared. We show that these surrogate models can accurately predict the wind velocity at any arbitrary point inside the wake. We further show that the surrogate models can also be used to predict wind velocities in wakes produced by novel wind speeds and still make accurate predictions, potentially avoiding the need to run extremely computation expensive CFD simulations for all the required wind speeds, saving significant time.

II. Background

A. Wind Farm Layout Optimisation problem

When constructing wind farms, each turbine comes at a large financial burden to the investors. Each turbine costs roughly 1-2 million US dollars per MW of nameplate capacity and also carries regular maintenance costs. Therefore, it is clear that optimised placements of each and every turbine is crucial and well worth the research and effort. A key problem that has been identified is the significant negative impact that wake effects of the turbines can have on wind speeds, turbulence and overall efficiency of the wind farm. Generally, inside the wake generated by the wind passing the rotating blades of a turbine the wind is more turbulent and more importantly, has a decreased velocity which leads to less power being generated by any other turbines that are within the wake. Due to this effect being the main reason

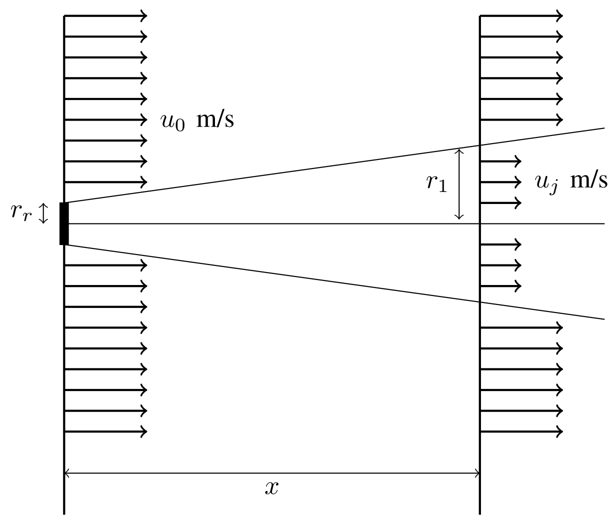

Fig. 1. Depiction of the wake effect (reproduced from [15]).

for decrease in efficiency of wind farms, the wind farm layout optimisation problems is focused on minimising this factor. Current methods, such as the Jensen model, consist of attempting to accurately estimate the wake effects (using mathematical models) and then use power generated as the metric to optimise. Therefore, power generated is maximised, which is the same thing as the wake effect being minimised.

B. Jensen model

The Jensen model is a mathematical model that attempts to estimate the far wake effect of a wind turbine and the overall velocity deficits caused by these far wakes [9] [10]. The model is a reasonably computationally inexpensive and simple approach for assessing wake interactions for two dimensional (i.e. hub height assumed to be equal for all turbines) layouts where the set of wind speeds and wind directions are given as parameters and assumed to be constant. Although it was originally devised in the mid 1980s, the Jensen model is still the most widely used mathematical model in the literature.

Figure 1 demonstrates a possible scenario that the Jensen model will produce. The wind flow is from left to right at speed u_0 and hits a turbine depicted by the black bolded rectangle on left. The exact parameters of the turbine are application specific and must be given as arguments in the algorithm - examples of turbine parameters supported by the Jensen model are the rotor diameter, d, and hub height, z, where each turbine is assumed to have the same values for these variables. As the wind flows past the turbine, the rotating blades cause the velocity of the wind to reduce. At a distance x meters downstream on the x-axis, the wind speed is u_j m/s, which will be less than the original, unaffected and constant wind speed, u_0. Figure 1 also demonstrates how the wake radius increases linearly in size with distance (causing a cone shape to occur). At x meters downstream on the x-axis, the wakes radius, r_1 is larger the radius of the rotor, r_r. The Jensen model can produce velocity deficits at given points inside the wake of a turbine, this information is then used to calculate the (potentially) altered wind speeds at each turbine, i.e. for each turbine in the wind farm, it is possible to calculate the total velocity deficit caused by the wake effects of other surrounding turbines. The predicted wind speeds at each turbine can then be used to estimate the power production of each turbine and therefore the overall power production of the wind farm. The optimization objective then becomes to maximize the total power production of the wind farm.

The main disadvantage of the Jensen model is that the accuracy of the mathematical model is still not clear. [6] compared three different mathematical wake models (including the Jensen model) with power production data from Horns Rev and Lillgrund offshore wind farms. They concluded that the models were robust and accurate, excluding single rows with narrow sectors. However, [2] highlighted physical deficiencies in the ability of the Jensen model to accurately predict the wake velocities. The main problem reported was that the expansion factor was not consistently accurate. [3] did a comprehensive comparison of six different mathematical wake models (including the Jensen model) with measurements from the Vindeby wind farm and showed that the mean absolute error was 15%, with a root-mean-square-error of 0.88 m/s, both of which are high error rates. The Jensen model also relies on many heavy assumptions, one of which is that the wake expands linearly and in a simple manner. The Jensen model is also a 2D model and therefore does not take into account the change in wind direction and wind rotation effects that occur as the wind flows past the rotating turbine blades in its basic form. For a more detailed explanation of the Jensen model, including explanations and examples of the net velocity deficit for each turbine, possible power functions and overall optimization functions, see [15]

C. Computational Fluid Dynamics simulations of wind turbine wakes

Computational Fluid Dynamics (CFD) is a branch of fluid mechanics that uses the Navier-Stokes equations, numerical analysis and data structures to solve fluid flow problems. An example of a fluid flow problem would be air flowing past a rotating wind turbine and the effects on the wind because of this, e.g. the wake effect or turbulence generated, the decrease in wind velocity, the change in wind direction etc. Results can be validated by comparing results with a wind tunnel and even full-scale testing such as flight tests or measurements from a wind turbine farm. Due to ever increasing computational power and the ability to easily parallelise the computation process, even large scale CFD simulations are possible.

However, in the case of simulating a wind farm, due to there being multiple wind turbines in which each turbine is a large object, with rotating blades which are of a complex shape, the run-times of the simulation quickly become infeasible due to the large mesh requirements.. Other possible alternative methods of modelling a wind turbine have been presented such as the actuator turbine model. The actuator turbine model predicts blade forces using empirically produced mathematical equations. The blade forces are predicted using these equations

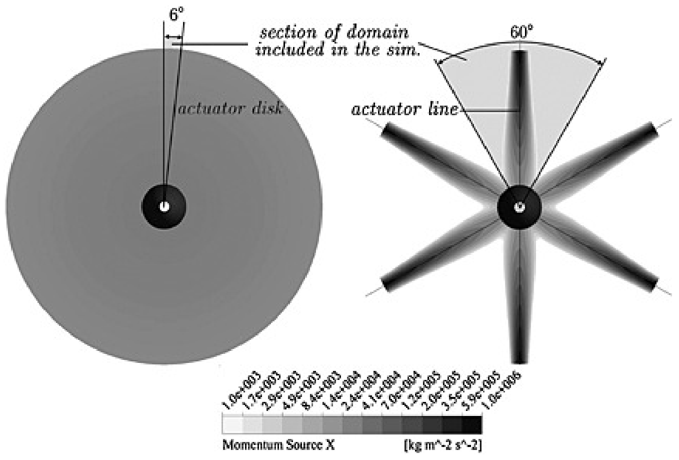

Fig. 2. The geometry of the Actuator disc (left) and Actuator Line (right) are shown [17].

depending on the local fluid (air) velocity at many specific points in each actuator element (run-time and accuracy increase as you calculate more points). These calculated forces are then used as input to a CFD simulation of the turbine field thereby reducing mesh requirements [24].

There are many different implementations of the actuator turbine model, here we will briefly explain two of the more commonly used implementations, as well as the full turbine blade model. Figure 2 shows the main difference between the two methods (being the area in which actuator elements are present) [17].

1) Actuator Disc Model: The Actuator Disc Model simulates the rotating turbines as a thin disc within the flow that imposes forces on the fluid (air) as it flows past the disc [11]. This disc occupies the swept area of the blades. Usually, the rotor swept area will be divided into many more elements relative to surrounding areas of the geometry. This means that calculations and equations will be solved with more accuracy around the disc region compared with regions further away from the disc, which generally will contain less points of calculation, with the intention of saving computation time.

Some implementations of the Actuator Disc Model apply both thrust and tangential forces, while others just apply basic thrust forces. The Actuator Disc Model which only applies basic thrust forces is based on the momentum theory of propellers. The forces are uniformly distributed over the disc region and calculated using a simple equation which takes into account thrust force, density, velocity of the wind at the centre of the rotor disc, area of the rotor swept area (the disc area) and the thrust coefficient. The model does not take into account the rotation in the flow produced by the blades, which is less than ideal. When the Actuator Disc Model is implemented where both thrust and tangential force are taken into account, more realistic results can be calculated [1]. It is implemented by taking into account both the lift and drag of the blades, called the lift and drag coefficients, which can be calculated empirically and are often included in official descriptions of turbine blade models. Chord length is also taken into account, among other variables. For a full description and comparison of the two different implementations of the Actuator Disc Model, see [13].

Similarly to the Jensen model, the Actuator Disc Model does not simulate near wakes generated by turbines very well. This is due to the fact that it does not capture tip vortices - which are created by wind hitting the rotating blades, since blades are purposely not part of the geometry, to decrease complexity and runtime, this factor cannot be simulated very well.

2) Actuator Line Model: The Actuator Line Model simulates the wind turbine blades as a set of blade elements [21]. As mentioned earlier, real data can be used to calculate the lift and drag of each blade element and then applied as body forces to the flow. In other words, a simple geometry can be created that uses mathematical equations to simulate the real blade geometry - the Actuator Line Model does not simulate the full geometry of blades. The geometry of the turbine is created by emulating the nacelle, hub and ground of a specified wind turbine. The blades are defined as a series of points, in which each point is its own actuator element. Once you define one blade element, you can easily use a rotation matrix to create the other blades in the correct position that they should be. During the calculation, you can emulate rotation of the blades by moving the blade elements equally around the correct axis by an appropriate amount depending the expected rotational velocity of the blades, given the wind speed.

Like the Actuator Disc Model, the Actuator Line Model is recorded to have accurate results when results are compared with a wind tunnel set-up. However, it still has it trade-offs: each blade element is only an emulation of a real blade and mathematical equations are still used to estimate the blades effects. Therefore, the only major difference between the Actuator Line Model and the Actuator Disc Model is the construction of the geometry - either a thin disc region is used to emulate rotating blades, or several rotating blade elements are used instead. Readers are referred to both [16] and [7] for a more detailed explanation of the Actuator Line Model, as well as results from an implementation of the model. Both these authors use tabulated airfoil data to calculate the blade variables (lift and drag).

3) Full Blade Model: The obvious alternative to using an actuator turbine model is to use a full-blown wind turbine blade model, that is based from real specifications of a particular wind turbine. In terms of CFD simulations, assuming the model that is created is accurate, this method represents the most realistic way in which to model the turbine blades and therefore should lead to the most accurate results. This requires far more computational power and leads to significantly higher runtime. Therefore, it is not feasible to continuously redo these simulations, which, for instance, would be needed if you are trying to evaluate newly generated wind farm layouts during an iteration of an optimization algorithm. In this paper, we use a proper blade model and then apply Machine Learning algorithms to create surrogate models that can accurately predict wind speeds inside a wake. This means that the simulations do not need to be redone for each new layout, but instead the surrogate model can be used. The upcoming sections describe our methods in more detail.

D. *Surrogate models based on machine learning*

Many real world optimisation problems are very computationally expensive and standard metaheuristic algorithms are generally only effective for problems on a much smaller scale. Real world problems are often high dimensional and fitness functions that are developed can take several hours or even days for just a single evaluation [8]. Therefore, there has been an increase in research aimed at trying to mitigate the runtimes of large scale expensive optimisation problems. The use of surrogate models to assist metaheuristic algorithms, namely, "surrogate assisted metaheuristic algorithms", has received particularly high attention over recent years. Some researchers have developed solutions that involve using surrogate models for fitness approximation, to replace the actual expensive fitness function, either partially or completely [20]. It has been shown that these surrogate assisted metaheuristic methods are able to achieve competitive results while using a far less computational power and have lower runtimes. Machine learning is one such way to develop surrogate models, standard regression algorithms can be used to predict fitness function outcomes or variable values at different stages of the evaluation. For example, [22] created a machine learning framework to predict the heating and cooling load of buildings, i.e. the energy performance of buildings. Eight variables (relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area, glazing area distribution) are used as input (features) to standard linear regression and also the more complex, random forest algorithms. They showed that, with use of the correctly weighted variables and data, it was possible to predict the energy performance of residential buildings with low mean absolute error (MAE). In this paper, we also are able to use surrogate models created by machine learning algorithms to predict wind velocities at specific 3D points with low MAE.

III. Dataset Construction

The blade used in this paper is similar in size and shape to a GE 1.5XLE turbine [12] but not identical. The total length of the blade is 42.3 meters and has a standard cylindrical shape with three key airfoil designs. Airfoil S818[18] defines the root, airfoil S825[19] defines the body and S826[19] defines the tip - the pitch angle at the tip is 4 degrees.

1) Geometry and Meshing: The software used was ANSYS Fluent and Workbench 18.0. ANSYS Fluent uses Computational Fluid Dynamics to simulate fluid flows, such as air/wind. This commercial software is considered the benchmark software for CFD calculations and has been used in both Academia and Industry for many years. Entire planes, helicopters and ship hulls can even be simulated, as well as effects of wind on fast moving racing cars and so on. The geometry for the simulation, which contains any objects to be

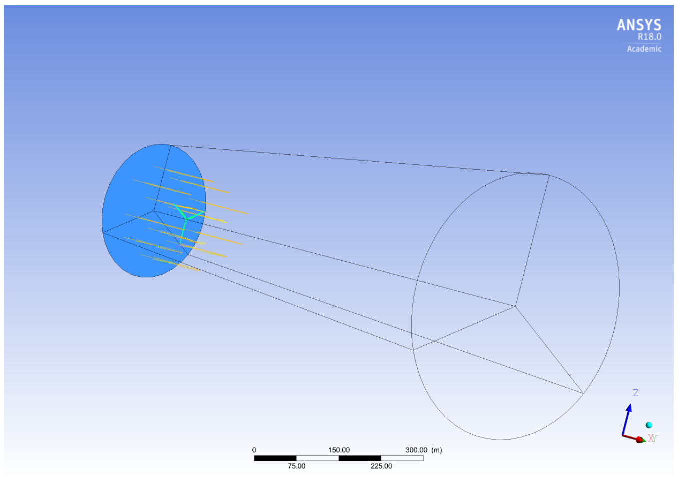

Fig. 3. The geometry of the GE 1.5XLE turbine blade model. The turbine blades are coloured in green, the wind flows from the inlet (blue shaded region) in the positive x direction and is shown as orange lines. Note that only a few lines are shown, the real simulation contains many more points. The cone extends 1000m along the x-axis

used in the simulation and also the domain in which the objects are within, is created in ANSYS DesignModeller. Figure 3 shows the final geometry, the turbine blade was enclosed inside a cone like domain, with the entire geometry extending 1000m, with the blade located at (90, 0, 0). Fluid (air) flows at the starting wind speed from the inlet or bottom of the cone, the centre of the inlet is located at (0, 0, 0).

Once the geometry of the blade has been created or imported, the shape and fineness of the mesh must decided. Meshing is the process of splitting flow domains into smaller sub-domains. Usually, these sub-domains are just simple 2D or 3D shapes, here simple 3D shapes are used. The reason for this splitting is because the partial differential equations that govern fluid flow and heat transfer are not usually possible to implement analytically. To get around this, the governing equations are discretised and solved inside each sub-domain. It is not hard to see that the more of these sub-domains are present, the more accurate the results will be, but also this means solving more equations which leads to longer runtimes. Here, the total mesh element count was 1.75 million, with the mesh around the turbine blade being far finer to increase accuracy of the effect the rotating blade has on the wind - the most important part of the experiment.

2) Experiment settings and parameters: ANSYS Fluent is used to set up the experiment type and parameters and run the experiment. A total of 8 experiments were run, each experiment had differing wind speeds ranging from 5.5 m/s to 17.5 m/s (these values fall within the turbines expected range), see Tables I and VI for a full list of the wind speeds used. SST K-omega was used as the viscous model. To create the rotation of the blade, a frame motion was used. The angular velocity of the blade was set to its optimal value based from the wind speed in any particular experiment, as defined in [14]. For instance, when the starting wind speed was set to

11.5, the rotational velocity, ω, was calculated to be 1.09. For a full list of the wind speeds and rotational velocities see Tables I and VI. To set the starting wind speed, its source and direction, a boundary condition was used. The fluid (air) velocity was set at the inlet to be one of the starting values described earlier and flows in the positive x direction. The blade is defined as a wall in ANSYS fluent - which means fluid cannot flow right through it. Once the starting variables have been assigned the solution methods must also be chosen. These methods determine how the experiment is run and iterated. For instance, it is possible to stop the experiment early if ANSYS determines that it has converged. You can change the convergence threshold settings so that the experiment does not end prematurely. Here, the convergence residuals were all set to 1×10^{-6} - meaning that early stoppage is very unlikely. The experiment was initialized from the inlet (standard) and the number of iterations was set to 2500.

3) Sampling experiment results: Once the experiment had completed, ANSYS CFD-Post was used to process the results. To export the velocities, a streamline was used, which is a standard way to display and export specific variables that are altered throughout the simulation. Here velocity in stationary frame is the chosen variable to be exported, velocity in frame is the absolute velocity of the wind at a given point in the wake. When using a streamline, the amount of sampled points and method of sampling can be chosen. Here, the the amount of sampled points was set to 100,000 (maximum possible in CFD-Post) and sampling method was set to equally spaced. The resulting exported data contained 2.3 million datums, where one line contains the velocity in stn frame, x, y and z coordinates. Any datums that are at a location in front of the blade were removed. The wind speed here will just be the starting speeds since the wind is yet to hit the rotating blade, so these datums are not needed, this area is the area between the turbine and the inlet, as can be seen in Figure 3.

The remaining datums are then reduced by dividing the geometry into 10x10x10 cells and selecting 100 datums per cell, generating roughly 72,800 examples. 100,000 examples are expected, however, not every cell contains 100 datums, Tables III and VII give exact counts. The process was repeated for each separate simulation at different wind speeds. When reading the wind velocities values in Tables III and VII it may seem that the wind speed is not really changed compared to the starting wind speed. Figure 4 gives a more accurate depiction of the significant variance in wind velocities at different points.

IV. Machine Learning Experiments

Fundamentally, what we are trying to do is accurately predict wind velocities inside the wake created by a rotating turbine. We completed advanced CFD simulations which computed velocities inside the wake, but it is not feasible to rerun these simulations on a regular basis inside an optimiser as each one takes roughly 8 hours to complete. Therefore, we now use Machine Learning to create surrogate models that can be used to predict velocities inside the wake effect of this particular turbine with good accuracy.

Wind Velocity	ω	TSR
5.5	0.521	4.189
8.5	0.805	4.189
11.5 (rated)	1.090	4.189
14.5	1.090	3.322
17.5	1.090	2.753

TABLE I
Wind speeds, optimal rotational frequencies (ω) and tip speed ratios (TSRs) for the five CFD simulations that were run. ω and TSR values are based on formulas given in Equations 38 and 39 of [14] for a three-bladed turbine with rotor radius 44.2 meters.

Features	Description
w	Wind velocity outside the wake.
x, y, z	Position relative to turbine hub inside wake in Cartesian coordinates.
r, θ, ϕ	Position relative to turbine hub inside wake in spherical coordinates.
$\frac{1}{x}, x^2, \frac{1}{x^2}, \frac{1}{y}, y^2, \frac{1}{z}, z^2, xy, yz, xz$	Non-linear transformations of the Cartesian coordinates.
w'	Wind velocity inside the wake (the class variable).

TABLE II
Features used to construct the datasets.

Dataset	Num. examples	w'
$D(5.5)$	72,785	5.48±0.118 m/s
$D(8.5)$	72,804	8.47±0.178 m/s
$D(11.5)$	72,794	11.45±0.246 m/s
$D(14.5)$	72,824	14.46±0.251 m/s
$D(17.5)$	72,781	17.46±0.26 m/s
$D(all)$	363,988	11.46±4.24 m/s

TABLE III
Datasets constructed for the experiment from the CFD simulation results. Mean and standard deviation are given for the w' values but the distribution is not normal: see Figure 4

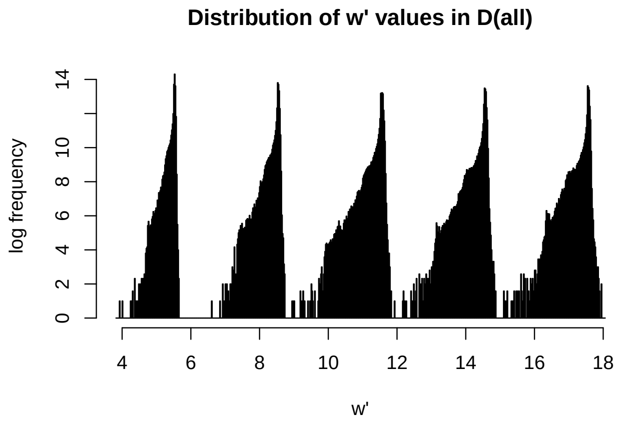

Fig. 4. Log frequency histogram of w' values in $D(all)$, illustrating the skew.

Algorithm	Description
Mean	Simple algorithm predicting the mean wind velocity based on the training data.
LR	Standard linear regression augmented with the M5′ attribute selection and the Akaike criterion for model selection.
M5′	Decision tree with linear regression models at the leaves [23].
RF	Random forest ensemble of 100 decision trees [4].
MLP	Standard multilayer perception trained using BGFS to minimise squared loss with a quadratic regularisation penalty, used in third experiment only.

TABLE IV
MACHINE LEARNING ALGORITHMS USED IN THE EXPERIMENTS. ALL ALGORITHM IMPLEMENTATIONS ARE AVAILABLE IN WEKA 3.8.0 [5] AND ALL PARAMETER SETTINGS ARE DEFAULTS.

$$\begin{aligned} w' = \quad & -1.1412 \\ & -0.001 \times x \\ & -0.0192 \times z \\ & +0.9981 \times w \\ & -5.3146 \times \tfrac{1}{x} \\ & +40.1374 \times \tfrac{1}{x^2} \\ & +0.0001 \times \tfrac{1}{y} \\ & -0.0001 \times z^2 \\ & +0.2117 \times \tfrac{1}{z} \\ & +0.0014 \times r \\ & +0.1921 \times \theta \\ & -0.0061 \times \phi \end{aligned}$$

Fig. 5. Example of a linear model trained on $D(all)$ using linear regression. The model shows that the non-linear transformations of x,y and z are useful, especially $\frac{1}{x^2}$.

A. Initial evaluation of regression algorithms

We ran our data through four different commonly used regression algorithms inside the WEKA framework [5]. As stated earlier, each instance of data has the x, y and z coordinates and also the objective or class to predict, which is velocity in stn frame. In other words, we are trying to predict the velocity at a specific 3D point. The algorithms used were: Mean, LR, M5 and RandomForest (RF), see Table IV for more details. 10x10-fold cross validation experiments were run for each algorithm and each dataset. Results show that more complex models like RF produced the most accurate wind velocity predictions, with the mean absolute error (MAE) being 0.0118 m/s compared with the baseline algorithm, Mean, which had a MAE of 0.7160 m/s. Overall, we show that complex Machine Learning algorithms can give good wind velocity predictions based from CFD simulation results. These initial experiments were promising as we were able to show proof of concept that surrogate models can be used to accurately predict wind velocities when compared to advanced CFD simulations. See Table V and Figure 6 for full results details.

B. Generalisation of the models to novel wind speeds

Given these initial promising results, we ran a further experiment to see if we could predict wind velocities at certain

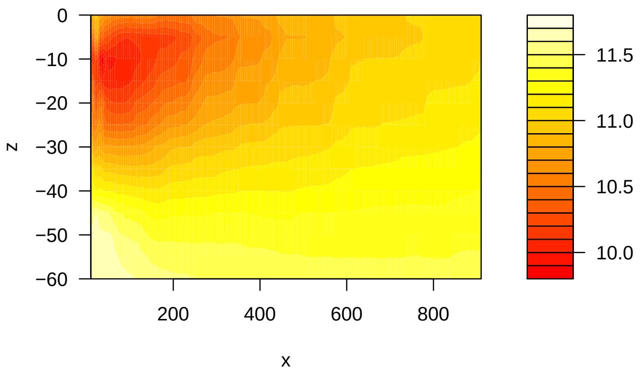

Fig. 6. Predictions made by the random forests classifier after being trained on the $D(11.5)$ dataset. Both scales are in meters. Predictions were made by setting $y = 0$ and dividing the $x \in [10, 910]$ vs. $z \in [-60, 0]$ subplane into a grid with spacing 5m between grid points. A prediction was made at each grid point. This figure shows only one half of the wake; the wake where $z > 0$ is assumed to be a mirror image of $z < 0$.

points with novel wind speeds, i.e. simulations at starting wind speeds that have not been run. The ability to predict velocities at given points accurately even if a CFD simulation has not been run with that starting wind speed is useful, as this means that these CFD experiments do not actually need to be completed if the error is acceptably low. We used the model trained on the $D(All)$ dataset - a dataset which is the concatenation of all of the five initial datasets, to predict wind velocities given starting wind speeds of: 7, 13 and 19 m/s, see Tables VI and VII for wind speeds, rotational velocities, TSRs and number of examples in each dataset. Results show that linear models, in this case LR, generalise the best, with the MAE being 0.1954 m/s compared to the other algorithms which have MAEs of 1.1442 or higher. See Table VIII and Figure 7 for full details.

C. Further generalisation experiments non-linear models

Given that linear models generalise the best, we explored ways in which to further reduce the error rate of these top performing algorithms. A simple way to do this is add non-linear features, this was already done in the original experiments as described in Table II. Figure 5 shows an example of a linear model that was trained on $D(all)$, non-linear transformations of x, y and z had a noteworthy weight, but $\frac{1}{x^2}$ had the most significant impact. Interestingly, this feature is also present heavily in the Jensen model and other mathematical models.

Another, more complex option involves learning a neural network and gradually add hidden nodes. Linear regression is analogous to a neural network with no hidden nodes and a single output with no activation function, so learning a more complex neural network should lead to better results. Here we used a Multilayer Perceptron inside the Weka framework (MLPRegressor) with gradually increasing numbers of hidden

Dataset	Mean	LR	M5′	RF
$D(5.5)$	0.0743±0.0009	0.0466±0.0005	0.0054±0.0004	0.0033±0.0002
$D(8.5)$	0.1139±0.0014	0.0708±0.0009	0.0086±0.0009	0.0050±0.0002
$D(11.5)$	0.1557±0.0020)	0.0963±0.0013	0.0114±0.0010	0.0068±0.0003
$D(14.5)$	0.1629±0.0020	0.0978±0.0012	0.0169±0.0041	0.0080±0.0004
$D(17.5)$	0.1660±0.0021	0.0992±0.0015	0.0160±0.0029	0.0089±0.0005
$D(All)$	3.6233±0.0114	0.0752±0.0005	0.0714±0.0008	0.0385±0.0009
Average	0.7160	0.0810	0.0216	**0.0118**

TABLE V

MEAN ABSOLUTE ERROR (MAE) IN M/S OF THE WIND VELOCITIES PREDICTED BY THE FOUR MACHINE LEARNING ALGORITHMS AFTER A 10×10-FOLD CROSS VALIDATION EXPERIMENT.

Wind Velocity	ω	TSR
7	0.663	4.189
13	1.090	3.705
19	1.090	2.535

TABLE VI

WIND SPEEDS, OPTIMAL ROTATIONAL FREQUENCIES (ω) AND TIP SPEED RATIOS (TSRS) FOR THE THREE ADDITIONAL CFD SIMULATIONS THAT WERE RUN.

Dataset	Num. examples	w'
$D(7)$	69,613	7.132±0.499
$D(13)$	72,221	13.123±0.639
$D(19)$	72,770	18.957±0.269

TABLE VII

ADDITIONAL DATASETS CONSTRUCTED FOR THE SECOND EXPERIMENT.

nodes starting from $n = 2$ and doubling in amount up to $n = 64$. Results show that the MAE is lowest with $n = 32$ at 0.1749 m/s, which is also an improvement over standard LR, see Table IX and Figure 8 for full results.

V. CONCLUSION

The Wind Farm Layout Optimisation problem has been approached from many different angles and involves various interacting variables and assumptions. However, when predicting the efficiency of a wind turbine, wind speed is the overarching variable that is used to estimate the power production of each individual turbine and therefore the wind farm as a whole. Here, we have shown that a combination of wind velocity data produced from CFD simulations and Machine Learning algorithms can be used to accurately predict wind speeds with an acceptably low MAE. Predictably, the more advanced Machine Learning algorithm (RF) performed the best during the initial evaluation of the CFD data. However, linear models generalised the best for novel wind speeds and use of a neural network (MLP) can drive the error rate down even further. This paper only explored wind velocity predictions on a single rotating turbine and therefore the multiple wake effect was not seen or addressed. Future work may try to predict wind speeds when multiple interacting turbines are present. The actuator disc or line method would be more appropriate for simulating multiple turbines due to it using significantly less resources and reduced runtimes. CFD simulations and surrogate models may also be applied to the efficient global wind turbine layout optimization problem. Overall, we have shown that it is feasible to create models

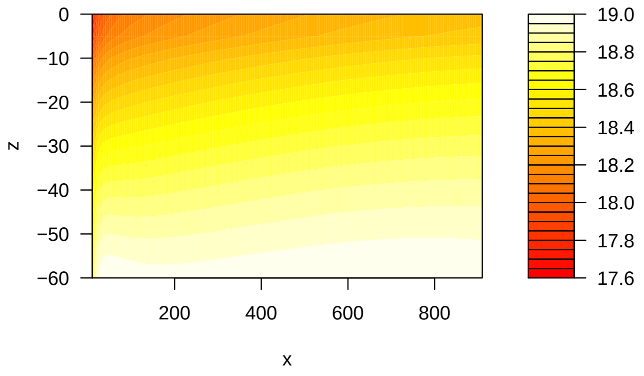

Fig. 7. Predictions made by linear regression for a wind speed of 19 m/s after being trained on the $D(all)$ dataset. Image was produced in the same way as Figure 6.

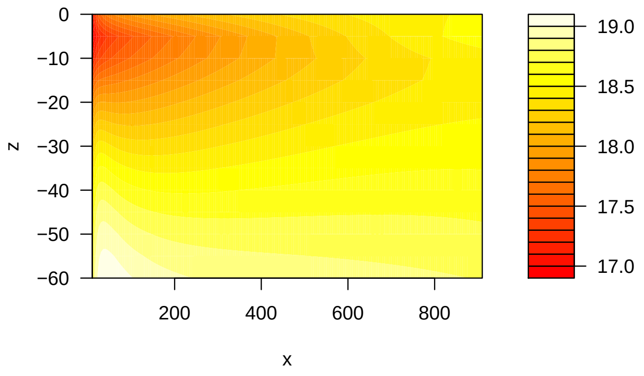

Fig. 8. Predictions made by a multilayer perceptron with 32 hidden nodes for a wind speed of 19 m/s after being trained on the $D(all)$ dataset.

Test Dataset	Mean	LR	M5′	RF
$D(7)$	4.3326	0.2696	0.2698	1.3946
$D(13)$	1.6586	0.2337	1.6633	1.4461
$D(19)$	7.4923	0.0829	1.4995	1.5131
Average	4.4945	**0.1954**	1.1442	1.4513

TABLE VIII
MEAN ABSOLUTE ERROR (MAE) IN M/S OF THE WIND VELOCITIES PREDICTED BY THE FOUR MACHINE LEARNING ALGORITHMS AFTER BEING TRAINED ON $D(All)$ AND TESTED ON THE ADDITIONAL DATASETS.

Test Dataset	$n = 2$	$n = 4$	$n = 8$	$n = 16$	$n = 32$	$n = 64$
$D(7)$	0.2501	0.2428	0.2735	0.2423	0.2394	0.2406
$D(13)$	0.2366	0.2097	0.2356	0.2010	0.1959	0.2002
$D(19)$	0.0850	0.1269	0.2477	0.1041	0.0895	0.0994
Average	0.1906	0.1940	0.2523	0.1825	**0.1749**	0.1801

TABLE IX
MEAN ABSOLUTE ERROR (MAE) IN M/S OF THE WIND VELOCITIES PREDICTED BY AN MULTI-LAYER PERCEPTRON WITH n HIDDEN NODES TRAINED ON $D(All)$ AND TESTED ON THE ADDITIONAL DATASETS.

using Machine Learning algorithms to replace full expensive CFD simulations, but more work is needed.

REFERENCES

[1] I. Ammara, C. Leclerc, C. Masson, et al. A viscous three-dimensional differential/actuator-disk method for the aerodynamic analysis of wind farms. *Transactions-American Society of Mechanical Engineers Journal of Solar Energy Engineering*, 124(4):345–356, 2002.

[2] S. J. Andersen, J. N. Sørensen, S. Ivanell, and R. F. Mikkelsen. Comparison of engineering wake models with CFD simulations. In *Journal of physics: Conference series*, volume 524, page 012161. IOP Publishing, 2014.

[3] R. Barthelmie, G. Larsen, S. Frandsen, L. Folkerts, K. Rados, S. Pryor, B. Lange, and G. Schepers. Comparison of wake model simulations with offshore wind turbine wake profiles measured by sodar. *Journal of atmospheric and oceanic technology*, 23(7):888–901, 2006.

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] E. Frank, M. Hall, and I. Witten. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, 4th edition, 2016.

[6] M. Gaumond, P.-E. Réthoré, A. Bechmann, S. Ott, G. C. Larsen, A. Pena Diaz, and K. Kurt. Benchmarking of wind turbine wake models in large offshore windfarms. *Proceedings of the science of making torque from wind*, pages 9–11, 2012.

[7] S. Ivanell, J. N. Sørensen, R. Mikkelsen, and D. Henningson. Analysis of numerically generated wake structures. *Wind Energy*, 12(1):63–80, 2009.

[8] T. Jansson, L. Nilsson, and M. Redhe. Using surrogate models and response surfaces in structural optimization–with application to crashworthiness design and sheet metal forming. *Structural and Multidisciplinary Optimization*, 25(2):129–140, 2003.

[9] N. Jensen. A note on wind generator interaction. Technical report, Risø DTU National Laboratory for Sustainable Energy, 1983.

[10] I. Katic, J. Høstrup, and N. Jensen. A simple model for cluster efficiency. In *Proc. Europe and Wind Energy Association Conference and Exhibition*, 1986.

[11] R. Mikkelsen. Actuator disc methods applied to wind turbines. *Technical University of Denmark*, 2003.

[12] M. Patel. GE Wind Components: 1.5 XLE ESS Dimensions, 2009.

[13] F. Porté-Agel, H. Lu, and Y.-T. Wu. A large-eddy simulation framework for wind energy applications. In *The fifth international symposium on computational wind engineering*, page 21, 2010.

[14] M. Ragheb and A. M. Ragheb. Wind turbines theory – the Betz equation and optimal rotor tip speed ratio. In R. Carriveau, editor, *Fundamental and Advanced Topics in Wind Power*, chapter 2. InTech, 2011.

[15] M. Samorani. *The Wind Farm Layout Optimization Problem*, pages 21–38. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[16] W. Z. Shen. Numerical modeling of wind turbine wakes. *J. Fluids Eng*, 124(2):393–399, 2002.

[17] M. Shives and C. Crawford. Mesh and load distribution requirements for actuator line cfd simulations. *Wind Energy*, 16(8):1183–1196, 2013.

[18] D. M. Somers. S816, S817, and S818 Airfoils: October 1991–July1992. Technical report, National Renewable Energy Lab., Golden, CO (US), 2004.

[19] D. M. Somers. The S825 and S826 airfoils. *National Renewable Energy Laboratory, Subcontractor Report*, 2005.

[20] C. Sun, J. Ding, J. Zeng, and Y. Jin. A fitness approximation assisted competitive swarm optimizer for large scale expensive optimization problems. *Memetic Computing*, Jul 2016.

[21] N. Troldborg, J. N. Sørensen, and R. Mikkelsen. Actuator line simulation of wake of wind turbine operating in turbulent inflow. In *Journal of physics: conference series*, volume 75, page 012063. IOP Publishing, 2007.

[22] A. Tsanas and A. Xifara. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings*, 49:560–567, 2012.

[23] Y. Wang and I. Witten. Induction of model trees for predicting continuous classes. In *Poster papers of the 9th European Conference on Machine Learning*. Springer, 1997.

[24] P. Wesseling. *Principles of computational fluid dynamics*, volume 29. Springer Science & Business Media, 2009.