A Graph Neural Network Surrogate Model for the Prediction of Turbine Interaction Loss

James Bleeg  
DNV GL, One Linear Park, Avon St, Temple Quay, Bristol BS2 0PS, UK  
james.bleeg@dnvgl.com

Abstract. The current generation of wind farm flow models lacks an option that can efficiently and reliably account for both wake and blockage effects when calculating turbine interaction loss. Traditional wake models are fast but ignore blockage effects. High-fidelity flow models are more complete, but turnaround times can be relatively long. The objective of this study is a model that combines the speed of traditional models with the accuracy of higher-fidelity approaches. To this end, we use a graph neural network (GNN) as a surrogate model for a steady-state Reynolds Averaged Navier-Stokes (RANS) model. Comparisons reveal good agreement between the GNN and RANS results for the atmospheric conditions considered.

1. Introduction

A wind turbine in a wind farm will generally produce less energy than it would operating in isolation. This difference in energy production is called a turbine interaction loss, and the calculation of these losses is a critical part of predicting the energy yield of a planned wind farm. The interaction between an array of turbines and the atmosphere is extremely complex, and thus wind farm flow models invariably include simplifying assumptions to allow for the practical calculation of turbine interaction losses. The most commonly used models assume that a wind turbine can only influence the production of another turbine located downstream, through its wake; as such, turbines in the front row of a wind farm are assumed to produce just as much energy as they each would operating in isolation. This simplification, which we call the wakes-only approach to turbine interaction, greatly increases computational efficiency, but it also decreases fidelity. Modelers regularly make trade-offs between speed and fidelity, but generally try to limit them to areas where the loss in fidelity has little impact on the predicted quantities of interest. The wakes-only approach was generally believed to fall into this benign category; however, recent studies indicate that the approach neglects material blockage-related losses, leading to bias in the overall loss predictions [1][2][3].

A new generation of turbine interaction models may be needed to efficiently and reliably account for the impact of both wakes and blockage. The form that these models should take is not yet quite clear. There are many potential options. The most widely used wind farm flow models use steady-state parabolic solvers, which only allow flow disturbances to propagate downstream. Since blockage effects arise from disturbances travelling upstream, the output from such a model requires a correction, though at present there is little agreement on what that correction should be [4]. Another approach would be to simulate a single turbine in isolation with a more complete solver, one where disturbances can travel both upstream and downstream, and then superpose copies of the calculated flowfield to estimate the flowfield around an array of turbines. Superposition is used in many wakes-only-type
models, but such an approach may not be able to adequately capture wind-farm-scale blockage effects, which likely arise from nonlinear turbine interactions as well as large-scale, inviscid interactions between the wind farm and a stably stratified atmosphere [5][6][7]. The influence of atmospheric stability may also pose a challenge to vortex-based modeling approaches.

The coupling between the wind farm and the atmosphere and between wakes and blockage effects is strong. Thus, separating these elements—a common choice used to simplify models and increase efficiency—can significantly decrease prediction accuracy. There are modeling options that generally avoid separating these effects. Mesoscale weather prediction models allow for the upstream propagation of disturbances, simulate atmospheric stability, and can likely capture some of the important large-scale effects that arise from the interaction between the wind farm and the atmosphere. That said, a reliable calculation of turbine interaction loss requires resolving the flow to turbine scale, something that mesoscale models cannot, at present, do. High-fidelity microscale models, like some Large Eddy Simulation (LES) and Reynolds Averaged Navier-Stokes (RANS) models, are capable of resolving many of the most important physical influences down to turbine scale; however, the time required to run these models limits the contributions they can make to the design of a wind farm.

One way forward could be to use multiple existing model types in careful combination, but this is not the approach we take here. Instead we develop a different type of turbine interaction model. The objective is a prediction capability with the speed of a wakes-only parabolic flow model and (nearly) the accuracy of a high-fidelity microscale model. Such a tool could be used to optimize a wind turbine layout while accounting for both wake and blockage effects.

To this end, we constructed a surrogate of a 3D RANS microscale model. LES likely has an accuracy advantage over RANS when predicting wakes, but when it comes to predicting blockage effects, the advantage likely narrows or even disappears, as blockage effects are probably less subject to the influence of turbulence. RANS is also much faster than LES and can thus produce a larger volume of training data for the surrogate model.

Advances in the theory and use of deep learning over the last dozen years make it an attractive option for the surrogate model. One approach could be to create a surrogate that predicts the flowfield through and around the wind farm. We use a different approach where the surrogate model predicts wind conditions at the turbine locations only. Such an approach requires much less training data, but it does raise other challenges. A typical deep neural network takes in a fixed number of inputs in a particular order, whereas wind farms have a variable number of turbines with no clear order. In part to get around this problem, we turned to a special type of neural network that has been getting more attention in recent years: the graph neural network (GNN).

There are many types of GNNs [8]. For this study, we selected a basic type that has been referred to, auspiciously, as an interaction network [9]. We customized the GNN to apprehend pairwise interactions between turbines in a wind farm— with the goal of achieving combinatorial generalization [10]. In the context of this study, combinatorial generalization means the ability to accurately predict RANS results for a new wind farm configuration (i.e. turbine layout plus wind direction) using elements, or building blocks, derived from experience with RANS results from other configurations.

The GNN and the RANS models are described in more detail in the next section of the paper. Then in the Results section, turbine interaction predictions from the GNN are compared with RANS predictions in order to assess the accuracy of the GNN. We then discuss the findings and draw conclusions.

2. Modeling

This section describes the three primary components of the modeling approach: the RANS model, the surrogate GNN, and the training of the GNN using output from the RANS simulations.

2.1. RANS

The steady-state RANS model was constructed within STAR-CCM+ [11], a general-purpose physics simulation software package best known for computational fluid dynamics. This subsection
summarizes some of the key aspects of this model, which has been customized to simulate wind farm flows. More information about the model and its validation can be found in [1][12][13].

Buoyancy effects are simulated with a shallow Boussinesq formulation, which features an energy equation coupled with a buoyancy term in the momentum equation. There are also buoyancy terms in the turbulence equations. With this formulation, along with appropriate boundary and initial conditions, atmospheric stability/buoyancy effects can be reasonably represented in the RANS results. The coupled solver in STAR-CCM+ helps ensure efficient and robust solutions to the equation system.

Wind turbines are represented with actuator disks specially designed for use in a turbine interaction analysis. A turbine interaction analysis quantifies for each wind farm turbine the difference between its power production in isolation ($P_I$) and its production when the other wind farm turbines are present ($P$). Naturally this difference depends upon how the wind conditions at each wind farm turbine differs from the wind conditions it would experience operating in isolation. Hence, the key output from the RANS model is the set of wind conditions at the actuator disks.

The actuator disks are also controlled based on the wind conditions at the disks, specifically the average axial velocity over the rotor’s swept area ($U_{disk}$). The applied body forces ultimately derive from a manufacturer-provided table of power, thrust, and freestream wind speed. Using a procedure explained in [1] and [14], we add a critical fourth column to the table: $U_{disk}$.

This modeling choice, to have the analysis hinge on $U_{disk}$, is based on the understanding that the thrust and power of an actual wind turbine is a function of the air density and the velocity distribution across the rotor face. It is these conditions that determine the aerodynamic loads, which in turn integrate directly to thrust and torque (and power). These functions for thrust and power are the same whether the turbine is operating in a wind farm or in isolation. Our approach to modeling a wind turbine mimics this reality in that thrust and power are functions of $U_{disk}$, and the functions are the same whether a given actuator disk definition is simulated in a wind farm or simulated in isolation.

The RANS-based turbine interaction analysis involves running a set of wind farm simulations and a set of freestream simulations (i.e. turbine-free simulations), with both sets sharing the same boundary conditions and mesh. From the wind farm simulations, $U_{disk}$ values are extracted, and from these the power ($P$) at each wind farm turbine can be determined, along with the effective turbine wind speed ($U$), which is the wind speed used to look up the power in a traditional power curve. From the freestream simulations and several isolated turbine simulations (four isolated turbines, three wind directions each), we can accurately estimate the power and effective wind speed for any wind farm turbine were it to be simulated in isolation ($P_I$ and $U_I$). From these we then calculate the turbine interaction loss factors (TILF) for power and wind speed, $P/P_I$ and $U/U_I$. Wind speed TILF is the target quantity upon which the surrogate model was trained.

The accuracy of the RANS model with respect to predicting the freestream spatial variation of wind speed over terrain has been demonstrated, favorably, through comparisons with mast measurements at hundreds of wind farms, including multi-site blind tests. Good agreement has also been found between the RANS model and field observations related to wind farm blockage [1]. With respect to wake effects, the standard $k$-ε approach used in this model is known to generate excessive turbulence in the near wake behind a simulated actuator disk [15]. Nevertheless, validation to date, some of which is reported in [1], indicates that the wake-related predictions from this RANS model are reasonable, at least in terms of the impact of the wakes on the energy production of downstream turbines. But of course, the focus of this research is not on how well the RANS model predicts reality, though this is important. It is on how well the surrogate model can predict the RANS predictions.

2.2. Graph Neural Network (GNN)

This section describes the GNN and how it is used. The inputs, to the full process, are turbine coordinates, rotor diameters, a wind direction, turbine power curves, coefficient of thrust curves, and the effective wind speed at each turbine were it to be operating in isolation ($U_I$). The output of the model is the wind speed TILF (i.e. $U/U_I$) at each turbine—from which TILF for power ($P/P_I$) can also be derived.
The following list of five steps summarizes the GNN algorithm, including pre- and post-processing:

1. Pre-process: Build the graph vertices ($v_i$) and the edge inputs ($e_{ij}$) to the GNN
2. Compute the influence of $v_i$ on $v_i$ and store the result in $e_{ij}^{i}, j = 1 \ldots N, j \neq i$
3. Aggregate the influences on $v_i$ and store in $e_{ij}^{i}$
4. Compute the effect of the aggregated influences and store in $v_i^{i}$ (i.e. $U/U_i$ for turbine $i$)
5. Post-process (e.g. calculate $P/P_i$)

Figure 1 describes these steps graphically for an example six-turbine wind farm. The rest of this subsection provides the details behind each step.

Figure 1. Steps in the GNN algorithm: (a) Step 1 – Create graph; (b) Step 2 – Compute influence of other wind farm turbines; (c) Steps 3 and 4 – Aggregate influences and compute effect (i.e. $U/U_i$); (d) Step 5 – Post-process
In step 1, we create the graph, which provides the input to the GNN. Each vertex \( v_i \) represents a wind turbine, and each edge \( e_{ij} \) represents an interaction relationship between a sender turbine \( v_i \) and a receiver turbine \( v_j \). The graph is fully connected, with edges between every possible pair of turbines. \( v_i \) is a vector defining the turbine position \((\tilde{x}_i, \tilde{y}_i)\), rotor diameter \(D_i\), and thrust coefficient \(C_{ti}\). The tilde over the coordinates denotes that the wind farm has been rotated so that the wind direction is always the same (e.g. 180 degrees). Thus while wind direction is an input to the pre-processor, it is not a direct input to the GNN. \( U_f \) is also not directly input into the GNN. Instead, it is used to make an estimate of \( C_t \) at each turbine through a table lookup of the turbine \( C_t \) curve. This rough approach to determining \( C_t \) probably decreases the accuracy of the model, though the magnitude of the related errors are likely mitigated because the effective wind speeds in this study are limited to the plateaus of the \( C_t \) curves. Since \( C_t \) is not particularly sensitive to wind speed when the turbine is operating on the curve plateau, the \( C_t \) inputs used in this study are reasonable estimates.

The direct inputs to the GNN are the edge definitions contained in the \( e_{ij} \) vectors. Each vector contains the normalized differences in \( \tilde{x} \) and \( \tilde{y} \) between the sender and receiver turbines as well as the coefficient of thrust from the sender turbine: \( e_{ij} = \left[ (\tilde{x}_i - \tilde{x}_j)/D_j, (\tilde{y}_i - \tilde{y}_j)/D_j, C_{ti} \right] \).

Steps 2-4 comprise the GNN. The GNN evaluates each receiver turbine independently from the evaluation of the other turbines, and further, in Step 2 each edge is evaluated independently of the other edges. In this step the edges directed to the receiver turbine are fed into the first of two artificial neural networks (ANN) in the GNN. Figure 1b illustrates the Step 2 process. Each of the five edges \( \Delta e_{1j}, \Delta e_{2j}, \ldots, \Delta e_{6j} \) directed to the receiver turbine \( v_i \) is fed into the same ANN function, \( \emptyset_e \). The output from \( \emptyset_e \) is a vector \( \varepsilon_{ij}' \), which represents the influence of \( v_j \) on \( v_i \).

In step 3, the outputs from \( \emptyset_e \), are aggregated to a vector, \( \varepsilon_i \), which represents the combined influence of the other wind farm turbines on the receiver turbine. The aggregation function must be associative and commutative so that the order of the sender turbines does not matter. In this study, we use a simple sum for the aggregation (Figure 1c). In step 4, each \( \varepsilon_i \) is fed into the second ANN \( \emptyset_v \), which calculates the effect on turbine \( i \) \( (v_i') \) due to the combined influence of the other wind farm turbines. \( v_i' \) is a scalar and the output of the model, which in this study is trained to target values of \( U/U_f \).

Both ANNs, \( \emptyset_e \) and \( \emptyset_v \), have five layers and use ReLU activation functions. \( \emptyset_e \) is shared across all edges. \( \emptyset_v \) is shared across all vertices. They are trained together as a single network, the GNN.

2.3. Training

Inputs and outputs from 208 wind farm CFD simulations comprised the validation and training sets for the GNN—of these, 163 were randomly assigned to the training set. The data included 36 wind farms, representing a mix of real and hypothetical turbine layouts on generally flat terrain. The simulated wind speeds corresponded to the plateaus of the turbines’ \( C_t \) curves, and at some of the wind farms, multiple inflow wind directions were simulated. The number of turbines in the wind farms ranged from 10 to 208. There were 11 different turbine types in the data sets. All the RANS simulations were run with the same stability conditions: neutral boundary layer; capping inversion starting at approximately 1000 m above ground level; and uniform stratification in the free atmosphere above. Altogether there were 16,310 simulated turbines—with the same number of \( U/U_f \) values—and over 1.8 million pairwise interactions in the data sets.

There are many instances where the same wind farm appears in both the validation and the training sets, but with different inflow wind directions. This is because the training and validation sets were broken up by wind farm simulation, not by wind farm. The benefit of this approach is that the GNN is exposed to a broader, more diverse, training set. The downside is that the validation statistics, strictly speaking, cannot be said to represent how well the GNN emulates the RANS predictions at a wind farm it has never seen before. Having said that, they may not be far off. The GNN does not directly predict the turbine interaction loss for the whole wind farm. Instead, it predicts turbine interaction loss on an individual turbine basis; the training of the model also takes place on individual turbines. \( U/U_f \) for a given wind farm turbine can be quite sensitive to small changes in wind direction, and this limits
the learning that can be achieved for that turbine based on simulation results for the same layout but a different wind direction. This matter is revisited, quantitatively, in the next section.

3. Results

The GNN model used in this study was trained over 250 epochs, during which the turbine interaction loss prediction errors dropped to acceptably low levels (Figure 2). At the end of the training, the mean squared error on $U/U_1$, the loss function used for the training, was 0.00004 for the training set and 0.00007 for the validation set. If we filter the validation set for wind farms that do not appear in the training set in any form—just four wind farms in all—the errors on $U/U_1$ are similar to that of the full set: 0.00005 mean squared error (MSE) for individual turbines and 0.0029 mean absolute error (MAE) for the wind farms.

![Figure 2](image.png)

**Figure 2.** $U/U_1$ error vs. epoch for (a) training and validation turbines and (b) validation wind farms

We also evaluated the GNN on a small test set. The test set comprised 20 simulations across two wind farms, with 2020 simulated turbines in all. The wind farms were clearly distinct from any of the wind farms in the training and validation sets, though the turbine layouts were well within the size and density ranges of those sets. MAE on wind speed TILF at wind farm level was 0.0028 for the test set. The MAE on power TILF (array efficiency) was 0.0075.

One of the test wind farms was created specifically to explore and share a use scenario for the GNN. The layout, which is on flat terrain, has 110 turbines with nearest neighbors approximately seven rotor diameters apart from each other. Figure 3 depicts the wind farm along with $P/P_1$ predictions from the GNN and RANS models. The average wind direction at hub height in the simulations is 179.4°. The average effective wind speed at the leading turbines is 7.1 m/s.

Because the results correspond to a single unchanging wind direction, there can be sharp variations in $P/P_1$ from turbine to turbine. Figure 4 illustrates why: some turbines downstream of the leading row are much more affected by wakes than others. Despite these sharp variations in the steady-state results, the RANS-predicted pattern of production (Figure 3a) is remarkably well-captured by the GNN (Figure 3b). Where RANS predicts large turbine interaction losses, so does the GNN. Where RANS predicts low turbine interaction losses, even deep within the array, so does the GNN (e.g. on the left, the third row/arc from the back). That said, the GNN regression blunts some of the more extreme predictions. This can be seen more clearly in the scatter plot in Figure 3c, which quantifies the comparison between the two sets of results.

Even across the leading string of turbines, the agreement between the two models is quite good (Figure 5). Both models indicate that most of the turbines will produce significantly less than they each would operating in isolation, with the exception of the turbines at the edges of the string, which are predicted to produce more than they would in isolation. These results are in stark contrast to the predictions that would come out of a wakes-only model. Such a model would predict the same power for each turbine along this leading string with $P/P_1$ equal to 1.0 (i.e. zero turbine interaction loss).
Figure 3: $P/P_I$ at a 110-turbine wind farm as predicted by (a) the GNN, (b) RANS, (c) both. Wind direction equals 179.4°. Inflow wind speed equals 7.1 m/s.
Figure 4. RANS-predicted percent change in wind speed relative to freestream conditions at hub height. The magenta curves enclose areas where the wind speed is more than 1% greater than freestream. Wind direction equals 179.4°. Inflow wind speed equals 7.1 m/s.

Figure 5. $P/P_f$ along the front row of the 110-turbine wind farm as predicted by the GNN and the RANS model. Wind direction equals 179.4°. Inflow wind speed equals 7.1 m/s.

Turbine interaction loss predictions were made with the GNN for 180 wind directions in increments of 2° at the 110-turbine wind farm. RANS predictions were made for 8 wind directions. The two sets of wind farm TILF predictions ($\sum P / \sum P_f$) are in good agreement with each other (Figure 6). The GNN does underpredict the RANS result by an average of -0.86%, but the bias does
not appear to be part of a strong, broader trend. The GNN overpredicts the power by 0.05% at the other wind farm in the test set, and the prediction bias across the full validation set is -0.2%.

Figure 6 also includes results from a typical wakes-only calculation, with an Eddy Viscosity (EV) wake model and a Large Wind Farm (LWF) correction [16]. The energy production predicted by the wakes-only model is notably higher than that of the other two models. There are a number of possible contributing factors to this prediction disparity, and blockage-related bias in the wakes-only model is likely one of them. An analysis of the RANS results, done according to the approach described in [1], suggests that the blockage-related prediction bias associated with the wakes-only assumption is 2.9-4.2%, depending on the wind direction (generally higher from the east and west than the north and south). Correcting for this bias would close a large portion of the gap between the wakes-only results and the other results in Figure 6.

![Figure 6](image.png)

**Figure 6.** Predicted wind farm TILF for power (a.k.a. array efficiency, $\sum P / \sum P_I$) vs. wind direction for the 110-turbine wind farm. The hub-height inflow wind speed is 7.1 m/s.

## 4. Discussion

The objective of this work is a model that can predict the impact of wakes and blockage on energy production and do it with both speed and accuracy. With respect to speed, the GNN takes 1.15 seconds on an Intel Core i9-9980HK to calculate the 180 wind directions in Figure 5 (exclusive of pre-processing). The structure of this GNN is especially well suited for parallel computing. Most notably, the model limits turbine interactions to pairs of turbines and so calculates these interactions independently of each other. Turbine $U/U_I$ predictions are also calculated independently. Thus, with appropriate parallel processing and multi-core machines, quick turnaround times can be achieved.

That said, treating turbine interactions as binary and independent may have drawbacks with respect to accuracy. For example, in Figure 1 the influence of v3 on v6 could potentially affect the influence of v6 on v1, but the structure of this GNN precludes the direct capture of such second-order interactions. There are GNN architectures that could potentially capture n-order interactions, while also offering improved thrust estimates for the sender turbines. These options, and the related trade-offs in speed and accuracy, will be explored in future work.

Notwithstanding these opportunities for improvement, the current implementation of the model demonstrates good agreement with the RANS results for wind farm configurations that are not in the training set. If the model could be expanded to a larger parameter space, including more inputs, like turbulence intensity, it could offer a number of practical benefits. At the least, it could help interpolate
between results from the high-fidelity model—for example, between the RANS results in Figure 6. This could increase the accuracy of the overall analysis and/or reduce the number of high-fidelity simulations needed. The GNN could also potentially make reliable turbine interaction loss predictions at wind farms where no high-fidelity simulations have been run. Such a tool could be used to optimize a turbine layout for energy, a notoriously challenging problem. A GNN offers some advantages that may make the problem more tractable. Speed benefits from parallelization could lessen the burden imposed by the enormous number of simulations needed for a typical layout optimization. In addition, the GNN is differentiable. This would enable the practical use of gradient-based optimization methods, which according to [17], may on average yield more energetic wind farm designs.

In practice, deployment and reliable use of the GNN will likely require establishing range of applicability. Number of turbines and density of the wind farm layout are natural candidates for defining the range of applicability of the current GNN, though these have yet to be tested. The GNN in this study was trained to emulate a RANS model, but it could instead be trained on results from a different type of flow model, like an engineering model or LES, and likely still achieve a comparable level of accuracy relative to that model. In fact, the GNN could, theoretically, be trained using field observations. The primary obstacle to such an approach is that it is not straightforward to extract reliable turbine interaction loss values from field measurements. Power ($P$) and effective wind speed ($U$) can readily be obtained from turbine data; however, $P_I$, the power that each turbine would generate in isolation, is much more difficult to determine. Until reliable estimates can be made for this quantity, the GNN is probably best matched with a high-fidelity flow model.

5. Conclusions
The results of this study demonstrate that a basic, well-trained graph neural network is capable of predicting with good accuracy the turbine interaction loss predictions from a high-fidelity flow model in relatively flat terrain—at least for the specific conditions tested (below-rated wind speeds, a neutral boundary layer, and a relatively narrow range of turbulence intensities). Future work will involve exploring different conditions and expanding the model parameter space to increase the viability of the approach for use in the design and analysis of wind farms. Such a model, if well validated and characterized, would allow the designer/analyst to confidently account for both wakes and blockage when optimizing the wind farm layout and predicting energy yield.

References
[1] Bleeg J, Purcell M, Ruisi R, and Traiger E 2018 Energies 11 1609
[2] Segalini A and Dahlberg J A 2019 Wind Energy 23 120
[3] Ebenhoch R, Muro B, Dahlberg, J A, Hägglund P B, and Segalini A 2016 Wind Energy 20 859
[4] Elkinton M 2019 Blockage session introduction Proc. AWEA Wind Resource & Project Energy Assessment Conf. Seattle
[5] Allaerts D and Meyers J 2017 Bound. Layer Meteorol. 166 269
[6] Allaerts D and Meyers J 2019 J. Fluid Mech. 862 990
[7] Wu K L and Porté-Agel F 2017 Energies 10 2164
[8] Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, Li C, and Sun M 2019 Graph neural networks: a review of methods and applications Preprint arXiv:1812.08434v4
[9] Battaglia P W, Pascanu R, Lai M, Rezende D, and Kavukcuoglu K 2016 Interaction networks for learning about objects, relations and physics Preprint arXiv: 1612.00222v1
[10] Battaglia P W, et al. 2018 Relational inductive biases, deep learning, and graph networks Preprint arXiv:1806.01261v3
[11] Siemens PLM Software 2017 User Guide for STAR-CCM+ Version 12, Siemens PLM Software, Plano, TX USA
[12] Bleeg J, Digraskar D, Horn U, and Corbett J F 2015 Modelling stability at microscale, both within and above the atmospheric boundary layer, substantially improves wind speed predictions Proc. EWEA Conf. Paris
[13] Bleeg J, Digraskar D, Woodcock J, and Corbett J F 2014 *Wind Energy* 18 369
[14] van der Laan M P, Sørensen N, Réthoré P E, Mann J, and Kelly M 2014 *Wind Energy* 18 2223
[15] van der Laan M P, Sørensen N, Réthoré P E, Mann J, Kelly M, Troldborg J, Schepers J G and Machefaux E 2014 *Wind Energy* 18 889
[16] DNV GL 2014 *WindFarmer Theory Manual* Version 5.3, DNV GL, Bristol UK
[17] Baker N F, Stanley A P, Thomas J J, Ning A, and Dykes K 2019 Best practices for wake model and optimization algorithm selection in wind farm layout optimization *Proc. AIAA Scitech* San Diego