Optimization of Sparse Sensor Network in Wind Farms using Reinforcement Learning

Siobhan Powell, Michael Howland, and Jackson Crane
Stanford University, Stanford, CA, 94305

While wind farms produce low-carbon energy, they are generally subject to production intermittency which is challenging to predict and forecast. Forecasting future wind farm power production can be improved through increased data aggregation from the wind farm. In the present study, we utilize reinforcement learning in order to optimally place wind speed measurement sensors within a wind farm in order to improve power production forecasting. Reinforcement learning is able to reduce the forecasting error of the wind farm power prediction by 20% **over expert-selected baseline sensor locations.**

I. Introduction

The IPCC Special Report 15 on global warming [1] found that current rates of emissions will result in a temperature rise from pre-industrial levels of 1.5°C by 2040. The report found that coal-based energy production must drop to single digits from its current state as 40% of global energy production. Wind and solar energy, today at 20%, must compensate for this transition and increase to 67% of electricity generation by 2050 [1].

As more wind farms are deployed and connected to the grid, it becomes more important to be able to accurately forecast their power output. Power forecasting on minute to hour timescales is important for grid stability, spinning reserve planning, demand response programs, and price forecasting. An important component of power forecasting in wind farms is accurate atmospheric data [2]. This data can be collected from sensors dispersed throughout the wind farm installation. These sensors can measure wind speed and direction, temperature, humidity, precipitation, and other relevant real-time atmospheric properties. However, the deployment of large numbers of these sensors is expensive and drives up the cost of wind farm installation. Meteorological (MET) towers are often required to measure the atmospheric properties at the hub height of the wind turbine; this is typically around 100 meters. The sensors on the MET tower alone cost $O(\$1,000)$. These costs discourage wind farm operators from collecting data and improving their forecasting capabilities.

A recent study focused on the optimal sensor placement in a wind farm in order to predict the low order modes of the wind turbine atmospheric boundary layer [3]. The study, which combines low order turbulence models with a Kalman filter, found that the optimal sensor placement to model the velocity field was in the shear layers of the wind turbine wakes. However, this sensor placement will likely not be optimal if the objective is to forecast future power production.

Xiao *et al.* recently reviewed the state-of-the-art in wind power forecasting methods [4]. In general, the forecasts decrease in accuracy as the prediction horizon is increased and none of the methods perform significantly better than predicting no power change from the current time. These errors are likely the result of the lack of data [4]. As such, in the present study, we will find the optimal sparse sensor placement within a wind farm for the purpose of predicting the wind farm power production one-minute in the future.

II. Dataset

The data set utilized consists of high-fidelity large eddy simulations (LES) of a model wind farm within realistic atmospheric boundary layer conditions. The LES methodology is described in [5]. There are 36 wind turbines in a domain of 6.4 km by 3.2 km by 2.4 km. A snapshot of the domain is shown in Figure 1(a). The fluid flow is left to right on an angle of approximately 20° as shown in Figure 4. The velocity data is taken from a time step which is one-minute prior to the time step of corresponding power data. Twenty distinct time steps are used for averaging. In the limiting case with infinite time averaging steps the fluid will travel according to the mean velocity and the optimal sensor location can be approximately computed as $d = \overline{v}\Delta t$ where d is the distance in the direction of flow upstream the sensor should be placed, $\overline{v}$ is the mean speed, and Δt is the time step (one-minute in the present study). The domain is periodic in the horizontal directions and the spacing is constant between all turbines. As such, in an infinite statistical average, all

turbines will have identical power production and resulting flow fields around each turbine The power productions for the 36 turbines during the temporal horizon of the present study are shown in Figure 1(b).

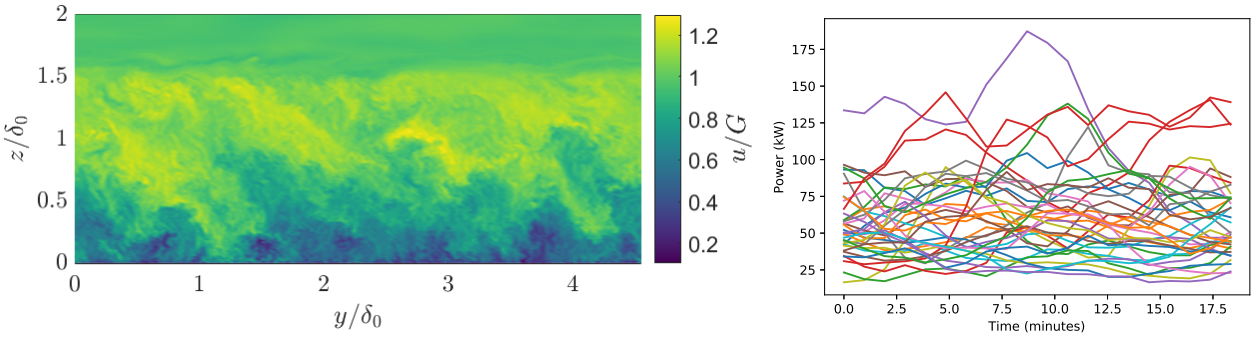


Fig. 1 (a) Easting velocity cross-plane snapshot for the LES simulation used in the present study. The velocity is normalized by the geostrophic velocity of $G = 5$ m/s. The dimensions are normalized by the initial boundary layer height of 700 m. (b) Power production for the 36 wind turbines as a function of time.

III. Online Methods

Due to the large domain size, increasing the number of sensors quickly causes the full state space to become computationally intractable. For example, with six sensors, the size of the full state space is $|S| = (256 \times 128 \times 192)^6 \approx 6 \times 10^{40}$ which brings challenges to model-based and model-free approaches. This naturally suggests utilizing an online method, wherein it is only necessary to consider the state space which is reachable from the current state. This section explores the implementation of two online methods: forward search (FS) and Monte Carlo tree search (MCTS).

Common to both methods were a few choices about the framing of the problem. The number of sensors chosen for the online method was fixed at 6: a balance between having enough sensors to estimate the power well and managing the computational concerns. To make the problem more tractable the action space is setup to only allow one sensor to move one cell in 3 dimensions at a particular time, which makes the dimension of the action space $|A| = N_{sensors} \times 6 = 36$. The reward function chosen for both methods was a least squares linear regression with the speed cubed measured by the sensors multiplied by an inverse sigmoid function based on the distance between the sensor and the turbine of interest $f(d) = 1 - 1/(1 + e^{4-d})$ as an input. This function effectively weights turbines which are closer to the sensor higher than those far away from the sensor. The target of the least squares linear regression is the power output of each turbine 1 minute later. The loss function is defined as: $\mathcal{L} = \sqrt{\sum_{i=1}^{m}(\hat{P}_i - P_i)^2}$ where $\hat{P}_i$ is the predicted power of the ith turbine.

In the preliminary and hyperparameter testing simulations for both FS and MCTS a constrained random initialization of the sensor positions was employed. The only constraint on the sensor placements was in the z-direction: the allowed placement in the z-direction was between the ground ($z = 0$) and two-times the turbine hub-height ($z \approx 0.5$). This manifests from our prior belief that the most valuable information is close to the elevation of the turbines. In the final optimization for sensor positions an informed prior was used for initial sensor positions to ensure reasonable initial spacing between sensors.

A. Forward Search

Forward search (FS) was implemented to provide a baseline performance for a simple online method. FS is a simple online algorithm wherein the loss function is calculated for each possible set of actions to a certain depth, d. The action chosen is the action which maximizes the expected return calculated to the given depth [6]. Because the computational cost scales with $|A|^d$, all searches were performed with $d = 2$ to balance depth of search and computational cost.

The only other hyperparameter in the FS algorithm is the discount factor used. Because in this problem the only reward that ultimately matters is the final loss function, it is expected that a large (1 or possibly greater) discount factor would be most effective. A traverse of four possible discount factors were performed in a 200 second trial of the FS algorithm with the same random initial conditions, the results of which are shown in Fig. 2. It is evident that all discount factors perform relatively uniformly, but that the discount rate of 0.75 performs the best out of the rates tested.

Although FS performs better than MCTS (discussed in the next subsection) in the limited hyperparameter sweeps, it

is believed that due to the limited depth achievable with forward search, the algorithm is susceptible to local minima, and will not produce an globally optimal solution. As such, MCTS was pursued for the final simulation.

B. Monte Carlo Tree Search

The other online method implemented was the Monte Carlo tree search algorithm (MCTS). MCTS is a more sophisticated algorithm than FS which hallucinates scenarios to a certain depth with a given exploration scheme, and then performs a policy rollout for conditions it has not encountered [7]. The rollout policy selected for this work was a random policy. The MCTS performs simulations to a given depth for a certain amount of time, and then, from the results of these simulations, selects the best action, and restarts. In this implementation of MCTS, the Q- and N-matrices (which represent the utility of state-action pairs and the number of times a particular state-action pair has been explored, respectively) were reinitialized after an action was made due to memory limitations.

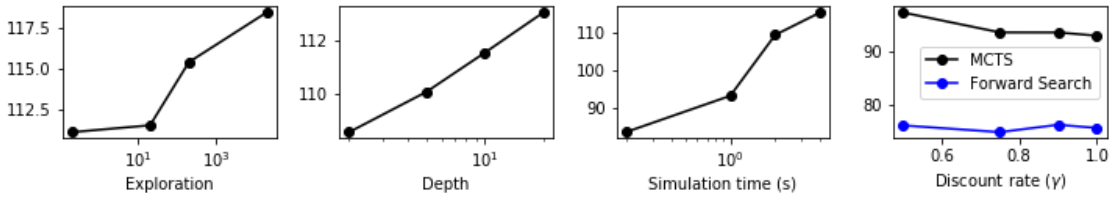
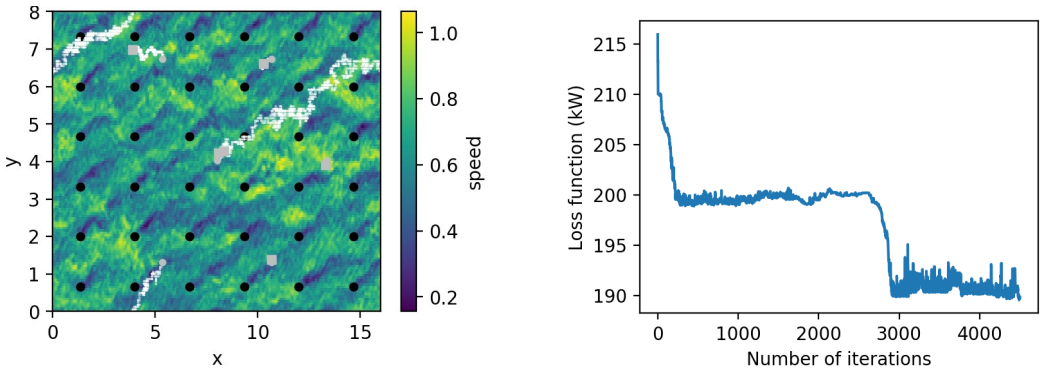


Fig. 2 Results of hyperparameter sweeps for the exploration parameter, depth, and simulation time for MCTS, and discount factor for both MCTS and FS. On the ordinate of all plots is the final loss function value. All simulations were performed for 200 seconds from the same initialization.

Four hyperparameters were considered for the optimization of the MCTS algorithm, (a) the exploration parameter, c, (b) the search depth, d, (c) the pre-decision simulation time, and (d) the discount rate, γ. Four parameter sweeps were performed corresponding to these four hyperparameters, the results of which are shown in Fig. 2. The sweep of the exploration parameter, (a), shows that less exploration, corresponding to a smaller exploration parameter, performed best. From this sweep, the exploration parameter equal to 20 was chosen which performed well, but still allows some exploration in longer simulations. Sweeps (b) and (c) show that low depth and low simulation time (which corresponds to more actions in a fixed total-time run) perform best. It is believed that because the simulation time is so low (200 seconds) the parameters which perform best are those that allow the exploration of the most actions and the most actions to be performed. This doesn't necessarily hold at longer simulation times, and so the depth chosen for the final simulation was 6, and the simulation time 12 seconds. The final hyperparameter examined was the discount factor, which, unlike the FS algorithm, performed best at a high discount rate, as expected. As such, a discount rate of 1 was chosen.



(a) x-y history of positions of sensors within domain

(b) Loss function with number of iterations

Fig. 3 Result of final MCTS simulation. In (a) the black circles represent turbines, the grey circles represent the sensor initial positions, and the grey squares represent the final positions.

A final simulation was performed for 15 hours with the hyperparameters described above. Two time steps were used for the regression (which explains why the loss function is higher than in the hyperparameter sweeps). The results are plotted in Fig. 3, and are not entirely expected. Notable is that the majority of the sensors do not move appreciably. Essentially the sensors move a little bit to be in the optimal position behind the turbine, in the advection direction. This corresponds with the initial drop in the loss function observed after approximately 300 iterations (*cf.* Fig. 3b). The loss function then plateaus for a long time, while one sensor in particular (the one that started in the lower left) searches for a better position. Note that, due to the domains periodic boundary conditions, a sensor can move across the domain boundaries and is placed on the other side. From Fig. 3a it appears that the sensor ends up in an identical position to another sensor. But the figure does not show the z-direction, and, in fact, this sensor has ended close to the top of the boundary (at $z = 3.8$), well above the turbine hub height ($z = 0.25$). The other sensors remain close to hub height. This movement, accompanied by the second drop in the loss function at approximately an iteration count of 3000, suggests that there is further information about the velocity field above the atmospheric boundary layer. This is unexpected from a fluid mechanics perspective. The end result of the simulation, however, still suggests that predicting each turbine power output with only a few sensors is challenging, at least with the regression scheme used. Although the algorithms were able to improve the loss function some, a large residual still remains. This is probably due to the simplified loss regressor, and also because it is not clear that the algorithm was fully converged.

IV. Model-free Methods

A. One Sensor Sub-domains

In general, we do not have a reliable explicit reward model for the prediction of future power based on velocity measurements [2]. As such, model-free approaches are an attractive solution model for the present problem. However, when multiple sensors are used, the size of the state space scales exponentially and quickly becomes infeasible as noted in the previous section. As such, we may utilize expert knowledge of the domain of interest to significantly reduce the state space. Since the domain is periodic and all turbines are statistically identical in an infinite temporal average, we can reduce the problem to a single turbine periodic box. This is sketched in Figure 4(a). One sensor will be used in this approach. One turbine will be predicted, as shown in Figure 4(a). The reward is given by the Pearson correlation coefficient between the cube of the sensor measured speed and the turbine power one-minute after the measurement is made. The correlation is computed as $R(v^3, P) = \left\langle (v^3 - \left\langle v^3 \right\rangle)(P - \langle P \rangle) \right\rangle / \mathrm{STD}(v^3)\mathrm{STD}(P)$ where $\langle \cdot \rangle$ denotes time averaging and $\mathrm{STD}(\cdot)$ denotes the standard deviation. Twenty independent temporal realizations of the flow are used to compute the correlation coefficient.

Within the sub-domain of interest, the model-free Sarsa algorithm is used. The sensor is randomly initialized in the domain and the epsilon greedy ad-hoc exploration strategy is used with $\epsilon = 0.1$ in order to significantly emphasize exploration as a result of local gradients in the fluid flow. The Sarsa algorithm is run for 10^7 steps. This took approximately 30 minutes on a standard workstation. The final sensor location is computed from the resulting Q values. The final sensor location can be seen in Figure 4(b). While the sensor was not significantly constrained in the vertical dimension, the final sensor location was located at hub height. The sensor is not precisely in the advection location but it should be noted that fluid flows do not exactly follow the mean advection and instead generally follow the local advection velocity. Therefore, the advection location shown in Figure 4(b) is not necessarily the statistically optimal solution. It is promising that the final sensor location is near the mean advection dashed curve.

B. Extension to Multiple Sensors

We can run the Sarsa algorithm as implemented in Figure 4 on each of the 25 sub-domains in the data set. The results are shown in Figure 5. We see the optimal sensor location in each box varies significantly across the different sub-domains. The blue circle shows the mean location for the sensor, but as there is no clear cloud around that point we find this is an inconclusive result.

In an effort to better breakdown this distribution we tried clustering the turbines by their power outputs using k-means. The number of clusters, k, is equivalent to the number of sensors used in the domain. Once the turbines are clustered, there will be one sensor assigned to predict the power for the cluster. The results for $k = 3$ clusters are shown in Figure 5. For the $k = 3$ case there does not seem to be a very clear spatial relationship to the clusters nor a connection to their optimal sensor placement results; however, for some number of clusters, this strategy might bring an improvement on the $k = 1$ case.

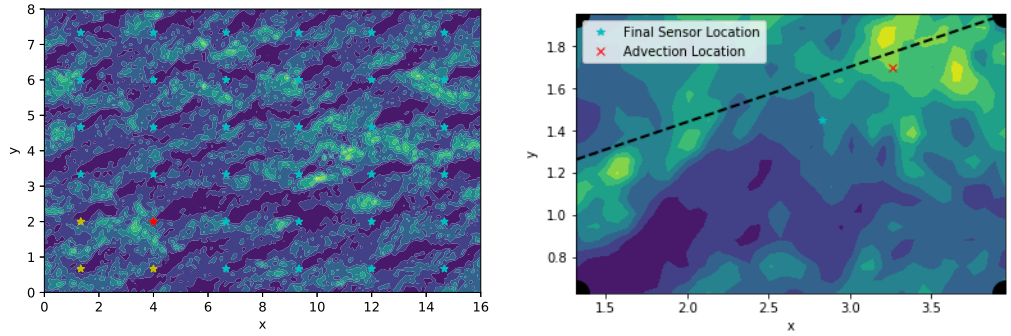


Fig. 4 (a) Domain of interest at the wind turbine hub height. The periodic box is outlined by the orange turbines. The red star denotes the turbine of interest for power prediction. (b) Final sensor location in the periodic sub-domain of interest shown in (a). The black circles indicate the bounding turbines. The turbine of interest is the top right turbine. The dashed line indicates the mean direction of fluid advection. The red line indicates the upstream location of the fluid at the turbine of interest one-minute in the past (i.e. likely best location for the sensor if a true infinite statistical average is performed).

To test this we considered the average optimal sensor placement within the results for each cluster of turbines. Then for each cluster one sensor was placed at that average optimal location in one sub-domain from the cluster, intending that it would be able to make a good estimate of the powers of all the turbines in that group.

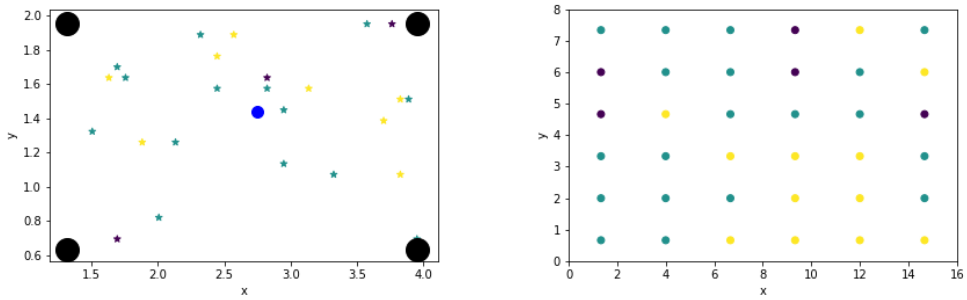


Fig. 5 (a) This plot shows the optimal sensor locations output by the Sarsa algorithm for each of the 25 periodic sub-domains. Figure 4 shows this result for one sample sub-domain. The black circles indicate the bounding turbines and the blue circle indicates the mean sensor location calculated across all the points. (b) The turbines were clustered by comparing their power outputs over 20 time steps. This result shows the three groups output by k-means clustering for $k = 3$. The same colouring is applied to the corresponding turbines in plot (a).

The same loss function from the MCTS simulations is used to judge the accuracy of the prediction for each case. As the number of sensors is increased the number of clusters is also increased. There are two main benefits to this: the location within the sub-domain for each cluster is refined as the average is taken across fewer and more similar turbines, and the effect of sharing one sensor between a whole cluster is improved as the cluster becomes tighter and smaller as well. This result is shown in Figure 6. This is one way of extending the Sarsa sub-domain solution to the full domain, and clearly illustrates the trade-off between the number of sensors implemented and the accuracy of the prediction.

V. Comparison of Methods

The two most important points of comparison for these results are the accuracy of the power prediction and the computational time required. The MCTS simulation was run assuming only 6 sensors for the whole domain, so we can compare that to the 6 turbine (6 cluster) result of the Sarsa algorithm in Figure 6.

The MCTS simulation took approximately 15 hours to achieve the results shown in Figure 3 and reached a final loss function value between the first two time steps of approximately 190 kW. The Sarsa simulation took minutes to run

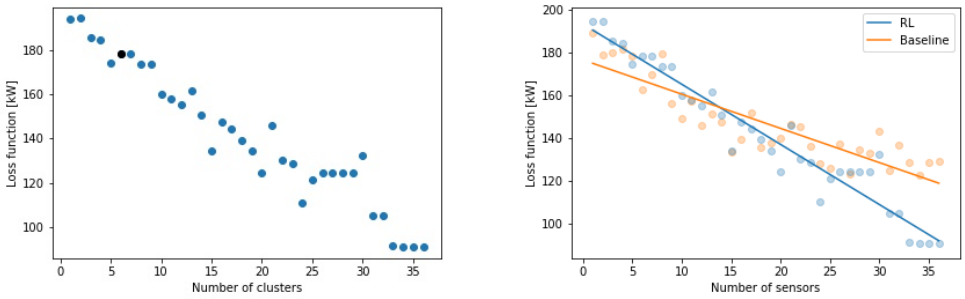


Fig. 6 (a) This figure shows the loss function decreasing as the number of sensors is increased, dictated by the number of clusters of turbines. The black point indicates the result for the case with 6 sensors, where the loss function value is 178.4 kW. (b) This plot compares the result of (a) to the same results generated using an industry standard baseline of putting the sensors at the advection location. Both series are modeled with linear regressions, and we see the cross-over point where the Sarsa method starts to beat the baseline occurs when there are 13 sensors.

for each sub-domain and a total time of approximately 30 minutes to solve all 25 sub-domains. Then clustering and calculating the loss function value with the 6 cluster case took less than 1 minute. It achieved a final loss function value of less than 180 kW. We can also compare to an industry-standard baseline sensor position in the advection location discussed in Section IV.A. We placed sensors at this location in front of 6 turbines randomly chosen from the set of 36. The result ran in less than 10 seconds and gave a loss function value of 182 kW. This value, however, shows large changes in each run due to the random choice of turbines.

We can also compare the case of 36 sensors (one per turbine) in the Sarsa and baseline cases. The baseline method using the advection location takes under 10 seconds to run and consistently achieves a loss function of around 130 kW. The Sarsa method, however, achieves a loss of 90.9 kW as shown in Figure 6. This is a significant improvement on the baseline. We also note that running the MCTS code on 36 sensors would be challenging due to the computational resources and time required.

The loss function is a decreasing function of the number of the sensors for both the reinforcement learning method and the expert baseline. For a small number of sensors, the baseline is superior to the learning, but for a large number of sensors, the reinforcement learning has significantly less prediction error. Figure 6(b) shows linear fits for the baseline and reinforcement learning methods. The cross-over point where reinforcement is beneficial over the baseline occurs at 13 sensors. The cause of this trend and cross-over is likely due to the clustering. For a small number of clusters, the turbines are not adequately grouped and prediction will be subject to the error in the clustering. The expert baseline is less subject to clustering error since it is a statistically robust prediction based on the mean flow advection. As the number of clusters increases the turbines are more effectively clustered and the reinforcement learning method is able to improve over the baseline through the use of the measurement of local flow structures near the turbines it is predicting.

VI. Conclusion

Both the online and model-free methods were able to improve over random sensor placement in the prediction of future wind energy power production in the large-eddy simulation. The model-free method improved significantly on the baseline expert-knowledge sensor locations when applied to the 36 sensor case (one sensor per turbine), reducing the loss function by approximately 20%.

While the model-free method outperformed the online method, the model-free method required a significant amount of expert knowledge in order to be used effectively and in a computationally feasible manner. The expert knowledge manifested as arguments of periodicity and statistical similarity. This expert knowledge is not necessarily available in a real, non-periodic wind farm and therefore the online methodology will have benefits in that setting.

In future work for the online methodology, we will incorporate a more accurate and detailed regression function and perform more hyperparameter sweeping including over γ, the discount factor. Future work for the model-free methods will include more temporal snapshots for time averaging. Future work for both methods will incorporate cooperative planning among the sensors.

Contributions

S.P. and M.H. collaborated to produce the model-free methods section, and J.C. produced the online methods section. In addition, M.H. provided the data-set and provided the background. All students provided equal contributions to the overall work.

References

[1] "Global Warming of 1.5°C: An IPCC special report on the impacts of global warming of 1.5°C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty," , 2018.

[2] Lei, M., Shiyan, L., Chuanwen, J., Hongling, L., and Yan, Z., "A review on the forecasting of wind speed and generated power," *Renewable and Sustainable Energy Reviews*, Vol. 13, No. 4, 2009, pp. 915–920.

[3] Annoni, J., Taylor, T., Bay, C., Johnson, K., Pao, L., Fleming, P., and Dykes, K., "Sparse-Sensor Placement for Wind Farm Control," *Journal of Physics: Conference Series*, Vol. 1037, IOP Publishing, 2018, p. 032019.

[4] Xiao, L., Wang, J., Dong, Y., and Wu, J., "Combined forecasting models for wind energy forecasting: A case study in China," *Renewable and Sustainable Energy Reviews*, Vol. 44, 2015, pp. 271–288.

[5] Howland, M. F., Ghate, A. S., and Lele, S. K., "Influence of the horizontal component of Earth's rotation on wind turbine wakes," *Journal of Physics: Conference Series*, Vol. 1037, IOP Publishing, 2018, p. 072003.

[6] Kochenderfer, M. J., *Decision making under uncertainty: theory and application*, MIT press, 2015.

[7] Kocsis, L., and Szepesvári, C., "Bandit based monte-carlo planning," *European conference on machine learning*, Springer, 2006, pp. 282–293.