A Bayesian optimization approach for wind farm power maximization

Jinkyoo Park* and Kincho H. Law
Department of Civil and Environmental Engineering, Stanford University,
Stanford, CA, 94305 USA.

ABSTRACT

The objective of this study is to develop a model-free optimization algorithm to improve the total wind farm power production in a cooperative game framework. Conventionally, for a given wind condition, an individual wind turbine maximizes its own power production without taking into consideration the conditions of other wind turbines. Under this greedy control strategy, the wake formed by the upstream wind turbine, due to the reduced wind speed and the increased turbulence intensity inside the wake, would affect and lower the power productions of the downstream wind turbines. To increase the overall wind farm power production, researchers have proposed cooperative wind turbine control approaches to coordinate the actions that mitigate the wake interference among the wind turbines and thus increase the total wind farm power production. This study explores the use of a data-driven optimization approach to identify the optimum coordinated control actions in real time using limited amount of data. Specifically, we propose the Bayesian Ascent (BA) method that combines the strengths of Bayesian optimization and trust region optimization algorithms. Using Gaussian Process regression, BA requires only a few number of data points to model the complex target system. Furthermore, due to the use of trust region constraint on sampling procedure, BA tends to increase the target value and converge toward near the optimum. Simulation studies using analytical functions show that the BA method can achieve an almost monotone increase in a target value with rapid convergence. BA is also implemented and tested in a laboratory setting to maximize the total power using two scaled wind turbine models.

Keywords: Wind farm power production, Cooperative control, Model-free optimization, Bayesian Optimization, Gaussian Process.

1. INTRODUCTION

Modern wind turbines allow adjusting the blade angle, the yaw angle and the generator torque to maximize the power production and to protect the mechanical and electrical components from excessive structural or electrical loads. Not only affecting its own power production, these control actions influence the power productions of the downstream wind turbines by changing the wake characteristics of the wind flow. In spite of wake interference, a wind turbine in a wind farm is operated to maximize its own power production, which can possibly lead to lower efficiency of the total power production of a wind farm.

Realizing that the interactions among the wind turbines can have impact on power production, researchers have proposed cooperative control approaches to maximize the total energy production of a wind farm by manipulating the wake interference pattern. In such approaches, the induction factor and the yaw-offset angle of a wind turbine are used as control inputs to adjust the wake interference pattern. The induction factor, which is determined by the blade pitch angle and the generator torque, is used to determine the power production of a wind turbine and, at the same time, to control the amount of wind speed reduction inside the wake, thereby influencing the energy production of the downstream wind turbines. The joint set of induction factors for wind turbines have been employed to optimize and to increase the total energy production of a wind farm.[1,2,3,4] The yaw-offset angle, defined as the misalignment angle between the wind direction and the rotor, decrease the power production of the wind turbine but can possibly increase the power productions of downstream wind turbines by deflecting the wake trajectory. The joint set of yaw-offset angles has also been used to optimize the total energy.[5,6,7,8]

Most cooperative control approaches find the optimum control actions by optimizing analytical wind farm power functions that mathematically relate the control inputs of wind turbines and the total power production of a wind farm.

*jkpark11@stanford.edu; phone 1 512 917 7751.

However, the analytical wind farm power functions might not reflect the conditions of a wind farm site or a wind turbine model because they are often constructed based on simplified wake model, e.g., Park Wake model.[9,10] To overcome the limitations of simplified wind farm power functions, Gebraad *et al.*[7] has used high-fidelity Computational Fluid Dynamics (CFD) simulation to construct the parametric wind farm power function. The constructed wind farm power function is then used to derive the optimal yaw offset angles of wind turbines. However, a CFD model itself requires the specification of a large number of parameters representing the environmental and wind turbine conditions. To avoid the use of the wind farm power functions, data-driven optimization methods have been proposed. For example, using only measurement data, Marden *et al.*[3] has applied game theoretic search algorithm and Gebraad and Wingerden[4] have applied the maximum power point tracking method to find the optimum coordinated induction factors that maximize the total power production of wind farm. However, these data-driven approaches often require a large number of measurement data to reach an optimum. So far, the feasibility of such approaches has only been verified through numerical simulations.

Recently, efforts have been made to optimize a target system using scarce data by exploiting the expressivity of non-parametric regression model. For example, Bayesian Optimization (BO) iteratively approximates the input and the output relationship of a target system using Gaussian Process (GP) and uses the probabilistic GP model to find the inputs that improve the target values.[11,12] Because BO uses the nonparametric GP regression that does not rely on a specific type of basis function, it can model complex relationship between the input and output of the target system using a smaller number of data points when comparing to search based algorithms. However, BO with the conventional sampling strategies still requires large number of data points over a large input space of the target system to reach the optimal operational conditions. Because sampling target values is associated with cost (time, energy, etc.) or loss (reduction in a target value), it is difficult to deploy conventional sampling strategies of BO to real-time control applications. In this paper, we develop what we call a Bayesian Ascent (BA) method that combines the strengths of the Bayesian Optimization and gradient-free trust region algorithms. BA is built upon the BO framework so that the target function can be efficiently modelled using Gaussian Process with a small number of data points. Furthermore, BA adapts the strategy of regulating the optimization scope, as used in the Trust Region method, into a Bayesian Optimization (BO) framework to select the sampling point that can monotonically increase a target value. Due to the use of trust region constraint on sampling procedure, BA tends to increase the target value rapidly converging toward near the optimum.

This paper is organized as follows: First, to provide the context for BA, the basic principles of Bayesian Optimization are presented. Then, the overall procedure of the BA method and its characteristics are illustrated using simple 2D functions. Finally, the BA method is employed to maximize the total power of two scaled wind turbines in a laboratory setting.

2. BAYESIAN OPTIMIZATION

The Bayesian Ascent (BA) method proposed in this paper is based on the Bayesian Optimization (BO) framework. BO seeks to solve $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} f(\boldsymbol{x})$ using the data on the input $\boldsymbol{x}$ and the corresponding output $y = f(\boldsymbol{x}) + \epsilon$, where ϵ represents the noise added during the measurement process. A function $f(\boldsymbol{x})$, whose analytical expression is usually unknown, can be thought of as the model that represents the target system, which produces the noisy output y given input $\boldsymbol{x}$. BO is composed of two iterative processes, namely the learning and optimization phases, each of which are discussed in the following.

2.1 Learning phase

In the learning phase, using the collected data from the inputs and the outputs, the unknown objective function $f(\boldsymbol{x})$ is modeled as a Gaussian Process (GP). A GP is a collection of random variables (stochastic process), any finite set of which has a joint Gaussian distribution.[13] A GP is completely described by its mean function $m(\cdot)$ and the covariance function $k(\cdot,\cdot)$, i.e., $f(\cdot) = \mathrm{GP}(m(\cdot), k(\cdot,\cdot))$. The mean function $m(\cdot)$, which is usually assumed to be zero, captures the prior trend on the output responses, while the covariance function $k(\cdot,\cdot)$ implicitly describes the underlying structure (i.e., the smoothness) of the function $f(\boldsymbol{x})$.

In GP, the value $f(\boldsymbol{x})$ of the target system given $\boldsymbol{x}$ and the observed outputs $\boldsymbol{y}^{1:n} = \{y^1, \dots, y^n\}$ follow a multivariate Gaussian distribution as[13]:

$$\begin{bmatrix} \boldsymbol{y}^{1:n} \\ f(\boldsymbol{x}) \end{bmatrix} \sim N\left(\boldsymbol{0}, \begin{bmatrix} \mathbf{K} & \boldsymbol{k} \\ \boldsymbol{k}^T & k(\boldsymbol{x},\boldsymbol{x}) \end{bmatrix}\right) \tag{1}$$

where $\boldsymbol{k}^T = \left(k(\boldsymbol{x}^1, \boldsymbol{x}^{n+1}), \ldots, k(\boldsymbol{x}^n, \boldsymbol{x}^{n+1})\right)$ and $\mathbf{K}$ is the covariance matrix (kernel matrix) whose (i,j)th entry is $\mathbf{K}_{ij} = k(\boldsymbol{x}^i, \boldsymbol{x}^j)$. The value of the covariance function $k(\boldsymbol{x}^i, \boldsymbol{x}^j)$ quantifies the similarity between the two input vectors $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$; the more the two vectors differ, the closer the value of the covariance becomes zero, meaning that they are not correlated in terms of their function values. We use a squared exponential covariance function whose evaluation between the two inputs $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$ is expressed as[14]

$$k(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sigma_0^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}^i - \boldsymbol{x}^j)^T \mathrm{diag}(\boldsymbol{\lambda})^{-2}(\boldsymbol{x}^i - \boldsymbol{x}^j)\right) + \sigma_\epsilon^2 \delta_{ij} \tag{2}$$

where σ_0 is the amplitude of the covariance and σ_ϵ^2 is the noise variance, assuming that $y^n = f(\boldsymbol{x}^n) + \epsilon$, where $\epsilon \sim N(0, \sigma_\epsilon^2)$. In addition, $\mathrm{diag}(\boldsymbol{\lambda})$ is a diagonal matrix with entries of $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_i, \ldots \lambda_N)$ along its diagonal, where λ_i is termed the length scale quantifying the relevance of the corresponding component x_i in the input vector $\boldsymbol{x}$ for predicting the outputs.

The hyper-parameters, i.e., σ_0, $\boldsymbol{\lambda}$ and σ_ϵ^2 are determined as ones maximizing the log-likelihood of the training data $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i) | i = 1, \ldots, n\}$ as

$$\begin{aligned}(\sigma_0^*, \boldsymbol{\lambda}^*, \sigma_\epsilon^*) &= \underset{(\sigma_0, \boldsymbol{\lambda}, \sigma_\epsilon)}{\mathrm{argmax}}\, L(\sigma_0, \boldsymbol{\lambda}\,; \boldsymbol{D}^n) \\ &= \underset{(\sigma_0, \boldsymbol{\lambda}, \sigma_\epsilon)}{\mathrm{argmax}} \left(-\frac{1}{2}(\boldsymbol{y}^{1:n})^T \boldsymbol{K}^{-1} \boldsymbol{y}^{1:n} - \frac{1}{2}\log|\boldsymbol{K}|\right)\end{aligned} \tag{3}$$

With the measurement data and the covariance function (hyper-parameters) updated, GP predicts the hidden function value $f(\boldsymbol{x})$ at arbitrary input $\boldsymbol{x}$ in a probabilistic framework. Since a distribution conditional of any subset of data is also Gaussian, the posterior distribution on $f(\boldsymbol{x})$, given the collected data $\boldsymbol{D}^n = \{(\boldsymbol{x}^i, y^i) | i = 1, \ldots, n\}$ and the new input $\boldsymbol{x}$, can be expressed as an 1-D Gaussian distribution with the mean and variance expressed as[13]

$$\mu(\boldsymbol{x}|\boldsymbol{D}^n) = \boldsymbol{k}^T (\mathrm{K})^{-1} \boldsymbol{y}^{1:n} \tag{4}$$

$$\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T (\mathrm{K})^{-1} \boldsymbol{k} \tag{5}$$

2.2 Optimization phase

The GP mean function $\mu(\boldsymbol{x}|\boldsymbol{D}^n)$ and the variance function $\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n)$, both of which representing the unknown target function $f(\boldsymbol{x})$ probabilistically, can be used to select the next input $\boldsymbol{x}^{n+1}$ in order to *learn* more about the target function as well as to *improve* the target value at the same time. If we are to learn about the target system only, one possible strategy would be to select the next input as the one maximizing the variance function $\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n)$, which reduces the uncertainty (variance) around the selected input (exploration). This sampling strategy, called active learning, has been widely used to select sampling points or to determine experimental parameters.[15] On the other hand, if we are to increase the target value, the natural strategy would be to select the next input as the one maximizing the mean function $\mu(\boldsymbol{x}|\boldsymbol{D}^n)$ that reflects the current belief about the target system (exploitation). However, this strategy is too greedy to search for a superior target value than the current one. For the success of BO, we need to balance the selection of the maximum value for optimization and the uncertainty space with respect to the opportunity for the selection.

In general, the next sampling point is being selected as one maximizing the acquisition function that incorporates both the aspects of exploration and the exploitation. For example, Dennis *et al.*[16] proposed a method of selecting the next input as the one that maximizes the Upper Confidence Bound (UCB) acquisition function as

$$\boldsymbol{x}^{n+1} = \arg\max_{\boldsymbol{x}}(\mu(\boldsymbol{x}|\boldsymbol{D}^n) + \rho^n \sigma(\boldsymbol{x}|\boldsymbol{D}^n)) \tag{6}$$

where the parameter ρ^n is selected to balance between the exploration and the exploitation. If ρ^n is small, $\boldsymbol{x}^{n+1}$ is selected as the one that maximizes the mean $\mu(\boldsymbol{z}|\boldsymbol{D}^n)$ (exploiting). On the other hand, if ρ^n is large, $\boldsymbol{x}^{n+1}$ is selected as one that is associated with large variance (exploring). In addition, Mockus *et al.*[17] proposed a method of selecting the next input based on maximizing the expected improvement (EI) acquisition function as

$$\boldsymbol{x}^{n+1} = \arg\max_{\boldsymbol{x}} \mathrm{E}[\max\{0, f(\boldsymbol{x}) - f^{max}\} \,|\boldsymbol{D}^n] \tag{7}$$

where $\max\{0, f^{n+1}(\boldsymbol{x}) - f^{max}\}$ is the improvement toward the maximum power output compared with the maximum power f^{max} observed so far. The expected value of this improvement is analytically derived.[17] The expected improvement function $\mathrm{EI}(\boldsymbol{x})$ has a higher value when either the mean or the variance is large. Therefore, by selecting $\boldsymbol{x}$ that maximizes $\mathrm{EI}(\boldsymbol{x})$, we can obtain either the improved objective function value (exploitation) or the updated objective function with reduced uncertainty (exploration) at that point.

3. BAYESIAN ASCENT METHOD

Bayesian Optimization (BO) can effectively optimize a target function using a limited number of sampled data points from a target system. However, BO with commonly used acquisition functions, such as EI and UCB, tend to select the sequential inputs over a large input space when exploring the target system. Two issues arise when BO is used to control a physical system: (1) the difference between the successive input is often too large, and (2) the inputs are chosen from the region where the uncertainty is too large. In general, control actions, i.e., the input, cannot be changed abruptly in a physical system and the selected action chosen from highly uncertain input space can result in significantly inferior target value. So far, BO has not been widely applied to real-time control applications. In this section, we discuss a modified sampling strategy of BO so that BO can be efficiently applied to real-time control applications. Specifically, we impose a proximity constraint in solving Eq. (7) such that BO could sample target values that are near the best solution observed so far. This strategy is similar to imposing a trust region constraint in mathematical optimization. We also strategically adjust the size of the trust region to expedite the rate of convergence to an optimum.

3.1 Hypercube trust region

Gradient-based mathematical optimization searches for a (local) optimum by iteratively optimizing a model function, which is in general a quadratic function, constructed using the gradient and the Hessian at each iteration. To guarantee that the next solution is selected from the region where the approximated model function is close to the target function, a trust region is defined to impose a proximity constraint such that the next iterate is selected only within the trust region.[18] The size of the trust region can be adjusted depending on the improvement in the target value to guarantee that the algorithm converges to the (local) optimum. Specifically, if the observed increase with respect to the previous function value is larger than a certain threshold of the predicted increase from the approximate model function, the solution will be used as the next iteration, and the trust region is expanded to expedite the convergence rate. Otherwise, the current solution will be rejected, and the trust region will be contracted to find a solution nearer the current solution. Because of this strategic adjustment in the trust region method, a target value increases monotonically, and the solution at each iteration converges gradually toward a (local) optimum.

To take advantages (i.e., the monotone increase in a target value and the gradual convergence to an optimum) of the trust region method, we impose the trust region constraint to the optimization phase of BO. With a trust region constraint imposed, BO then finds the next input close to the input $\boldsymbol{x}^{max}$ that corresponds to the best target value f^{max} observed so far. That is, BO selects the next input from the region where the uncertainty of the model function is not large (i.e., the GP mean function is close to the underlying target function around the region with sampled points). The optimization phase of BO can be posed as a constrained optimization problem described as:

$$\begin{aligned} &\text{maximize} \quad \mathrm{E}[\max\{0, f(\boldsymbol{x}) - f^{max}\} \,|\boldsymbol{D}] \\ &\text{subject to} \quad \boldsymbol{x} \in \boldsymbol{T} := \{\boldsymbol{x} |\, \|x_i - x_i^{max}\| < \tau_i \ \text{ for } \ i = 1, \dots, q\} \end{aligned} \tag{8}$$

where the objective function is the expected improvement, which is given in Eq. (7). The trust region $\boldsymbol{T}$ is defined as a hypercube with its center being $\boldsymbol{x}^{max}$; the ith component τ_i of $\boldsymbol{\tau} = (\tau_1, \dots, \tau_q)$ determines the range where the ith component x_i of $\boldsymbol{x} = (x_1, \dots, x_q)$ to be sampled next. Thus, the vector $\boldsymbol{\tau}$ controls the overall size of the hypercube trust region where the exploration takes place.

We call the BO algorithm with the trust region constraint Bayesian Ascent (BA) method, in that the algorithm follows the ascending direction estimated probabilistically from the sequence of observations. The overall procedure is summarized in Algorithm 1. As in the trust region method, BA adjusts the size of the trust region depending on the improvement in the target value. Let's denote the solution of Eq. (8) as $\boldsymbol{x}^{n+1}$. We check whether $\boldsymbol{x}^{n+1}$ sufficiently improves the target value. If the observed increase $y^{n+1} - y^{n}$ with respect to the previously measured value y^{n} is larger than a certain threshold ratio γ of the average increase $(1/n)(f^{max} - y^{0})$, where y^{0} is the initial measurement, the input $\boldsymbol{x}^{n+1}$ will be updated as the best solution $\boldsymbol{x}^{max}$ observed so far, and the trust region is expanded as $\boldsymbol{\tau}^{n+1} = \beta\boldsymbol{\tau}^{n}$, with $\beta > 1$ to expedite the convergence rate. Otherwise, the trust region will be reset as $\boldsymbol{\tau}^{n+1} = \boldsymbol{\tau}^{0}$, where $\boldsymbol{\tau}^{0}$ is the size of initial trust region. Here, we use the average increase $(1/n)(f^{max} - y^{0})$ as the criterion for evaluating the improvement because it is a more reliable criterion than a point value that is susceptible to randomness.

Algorithm 1 Bayesian Ascent (BA) algorithm

```
Initialize $\boldsymbol{x}^0$ and $\boldsymbol{\tau}^0$(initial trust region size)
**Repeat until convergence,** $n = 1, 2, 3 \ldots$
  1) Optimize the hyper parameters
     $(\sigma_0^*, \boldsymbol{\lambda}^*) = \underset{(\sigma_0, \boldsymbol{\lambda})}{\operatorname{argmax}} L(\sigma_0, \boldsymbol{\lambda}\,; \boldsymbol{D}^n)$
  2) Construct a GP model
     $\mu(\boldsymbol{x}|\boldsymbol{D}^n) = \boldsymbol{k}^T(\mathrm{K})^{-1}\boldsymbol{y}^{1:n}$
     $\sigma^2(\boldsymbol{x}|\boldsymbol{D}^n) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T(\mathrm{K})^{-1}\boldsymbol{k}$
  3) Select the next input by solving
       $\boldsymbol{x}^{n+1} = \arg\max_{\boldsymbol{x} \in \boldsymbol{T}} \mathrm{E}[\max\{0, f(\boldsymbol{x}) - f^{max}\}\,|\boldsymbol{D}^n]$
       $\text{s.t. } \boldsymbol{x} \in \boldsymbol{T} := \{\boldsymbol{x}|\ \|x_i - x_i^{max}\| < \tau_i^n \text{ for } i = 1, \ldots, N\}$
  4) Append the data $\boldsymbol{D}^{n+1} = \{(\boldsymbol{x}^i, y^i)|\ i = 1, \ldots, n+1\}$
  5) Update the exploration rate
    **if** $y^{n+1} - y^n \geq \gamma(1/n)(f^{max} - y^0)$ **then**
      $\boldsymbol{\tau}^{n+1} = \beta\boldsymbol{\tau}^n, (\beta^{success} > 1)$
    **else**
      $\boldsymbol{\tau}^{n+1} = \boldsymbol{\tau}^0$
    **end if**
```

Although it adapts the trust region strategy, the BA method is fundamentally different from the gradient-free trust region method. First, the BA method does not reject $\boldsymbol{x}^{n+1}$ but appends the data pair $(\boldsymbol{x}^{n+1}, y^{n+1})$ to the historical data set $\boldsymbol{D}^{n}$ regardless of whether or not $\boldsymbol{x}^{n+1}$ improves the target value because every single data pair can add information to the GP model, which can help sampling the next input. That is, the BA method uses the entire historical input and output data to construct the surrogate model function to approximate the target function, which is different form the gradient-free trust region method that uses only a subset of local data points to construct a local (quadratic) model function. In addition, the BA method does not contract the size of the trust region $\boldsymbol{\tau}$ but resets it to the initial size $\boldsymbol{\tau}^{0}$ because continuously contracting the trust region can possibly cause the sampling to be staying at an arbitrary point that is not a (local) optimum. When a (local) optimum is trapped by the hypercube, BO cannot find the input that results in a target value larger than the local optimum inside the trust region, leading the BA method to converge to the (local) optimum. Note that this study uses $\gamma = 0.05$, $\beta = 1.1$. In addition, the initial trust region $\boldsymbol{\tau}^{0}$ can be determined considering the ranges of the input component. Using 2~5% of the input range for each input component works well in general.

3.2 Illustrative example

Figure 1 illustrates how the BA method finds the optimum of a 2D quadratic function $f(x_1, x_1) = -(x_1^2 + x_2^2)/50 + 1$. The figures in the upper row show the trajectory of sampled inputs overlaying on the true function represented as contours. In addition, the figures in the bottom row show how the next sampling point is being determined at each iteration. In each figure, the circular dots are the inputs that have been sampled so far and the field square is the best input among the sampled inputs in the current iteration. Based on the sampled inputs and the corresponding target values, the expected improvement function, shown as contour plots in the bottom figures, is constructed. The next sampling point is then determined by maximizing the expected improvement function within the trust region, shown as shown by the open square. The selected next input is marked as field diamond shown in the figures.

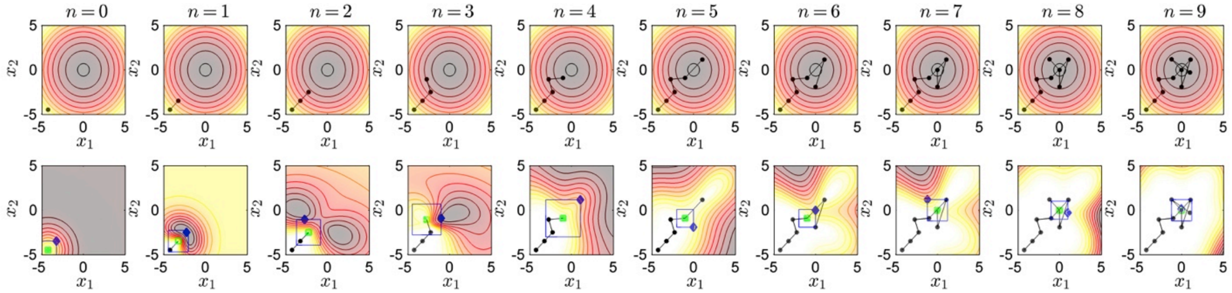


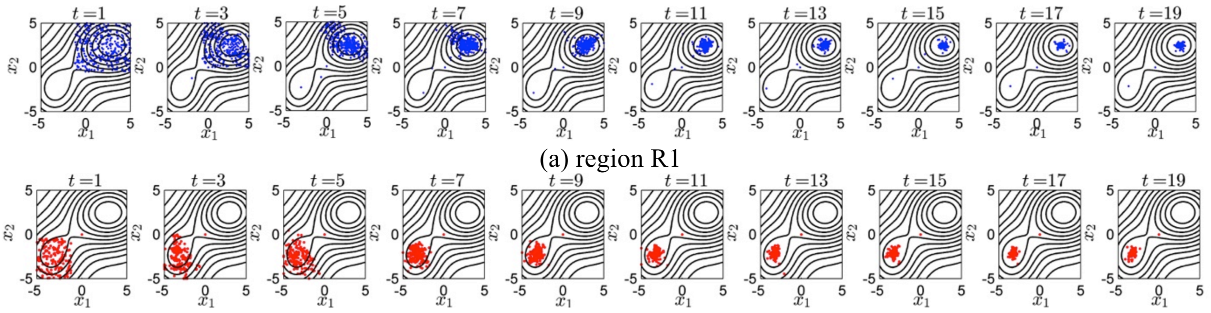
Figure 1. Bayesian Ascent method for finding the optimum of 2D quadratic function

As shown in Figure 1, for the first five iterations ($n = 0, \dots, 4$), the target value increases monotonically and the trust region expands by setting $\boldsymbol{\tau}^{n+1} = \beta\boldsymbol{\tau}^n$ with $\beta = 1.1$. During the $5th$ iteration ($n = 5$), it is observed that the newly sampled value does not increase the target value, i.e., the value does not satisfy the condition of $y^6 - y^5 > \gamma(1/5)(f^{max} - y^0)$. As a result, the size of the trust region resets to $\boldsymbol{\tau}^6 = \boldsymbol{\tau}^0$ and the center $\boldsymbol{x}^{max}$ of the trust region remains unchanged. The same situation occurs at $n = 6$. With the learned information from these two failed steps, the iteration proceeds to sample the input that increases the target value. Due to the use of the rust region constraint, the BA method improves a target value monotonically except during the iterations where BA tries to find the ascending direction ($n = 5$ and 6). In addition, the BA method regulates the step size, i.e., the distance between two successive samples, which can be beneficial in controlling a physical system. This is because, for a physical system, changing the control inputs gradually (progressively) is more preferable than changing them radically.

3.3 Example illustrating convergence

With the trust region constraint, the BA method is able to increase a target value monotonically by gradually changing the input values. As a result, the BA algorithm seeks to find a local optimum, similar to gradient-free trust region optimization algorithms. Here, we investigate the local convergence characteristics of BA using a simple analytical function. Let's consider the function $f(\boldsymbol{x}) = \frac{1}{2}\exp\left(-\frac{(x_1-2)^2}{2} - \frac{(x_1-2)^2}{2}\right) + \frac{1}{2}\exp\left(-\frac{(x_1-2)^2}{2} - \frac{(x_1-2)^2}{2}\right) \in [0, 1]$ which has two local optima. In this example, for each simulation, we randomly sample an initial point from one of the two regions $R_1 = \{\boldsymbol{x} | -5 < x_1 < -1, -5 < x_2 < 0\}$ and $R_2 = \{\boldsymbol{x} | -1 < x_1 < 5, 0 < x_2 < 5\}$, each has its local optimum. We then track the iterations to see how the target value sampled by the BA algorithm approaches to the local optimum for the region where the initial point resides. Furthermore, to test whether the BA method can identify and follow the ascending direction even with noisy data, we use noisy target value $y^n = f(\boldsymbol{x}^n) + \epsilon$, with Gaussian error $\epsilon \sim N(0, \sigma_\epsilon^2)$ with $\sigma_\epsilon = 0.01$ (1%).

Figure 2 summarizes the results depicting how the BA method converges to a local optimum based on an initial point. Figure 2(a) shows the results of 1000 simulations using 1000 initial points randomly sampled from region R1. Starting at an initial point ($t = 1$), the BA algorithm proceeds to sample the target values, shown as dots in each figure, until it reaches the local optimum in the region nearest to the initial point. Each dot in the subfigure ($t = 2$ to $t = 19$) represents the sample during an optimization step. The figure shows that the population of the sampled points converge to the (local) optimal region $\{\boldsymbol{x} | f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*) - \sigma_{error}\}$ within about 15 iterations (i.e., using 15 evaluated function values). Note that even though the BA method does not necessarily locate the exact local optimum because of the noise added to the target value. Figure 2(b) shows the result for the same simulations using the 1000 initial points sampled from region R2. The simulation studies conclude that the BA method can effectively locate a (local) optimum using the strategic sampling strategy based on trust region method. Particularly, the rapid convergence to a (local) optimum can be useful when the BA method is applied to real time control application. That is, given any initial point, the BA method is able to find an input that provides a target value that is superior to that of the initial point.



(a) region R1

(b) region R2

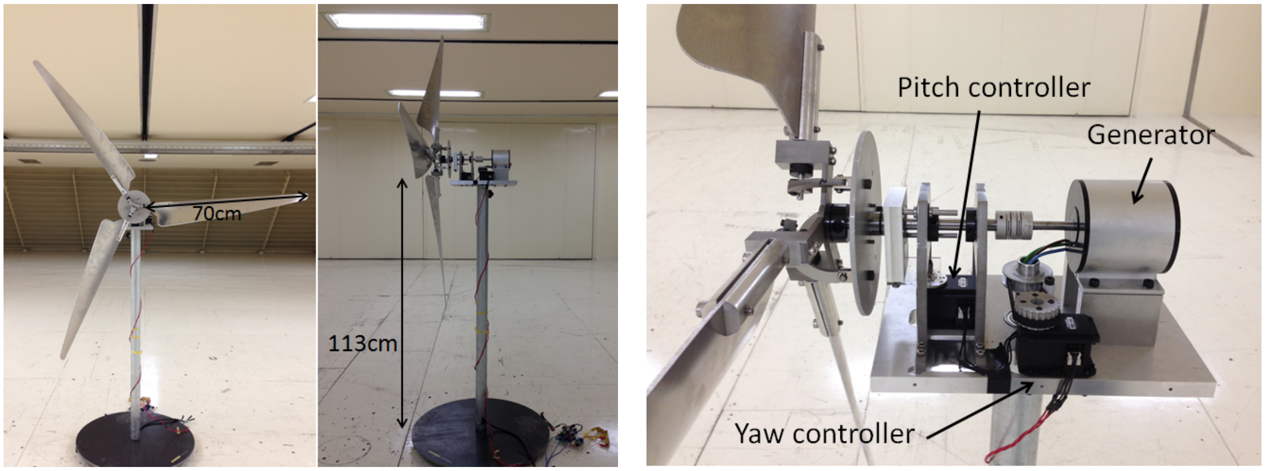
Figure 2. Convergence to a local optimum

4. WIND TURBINE CONTROL

This section discusses an experimental study to optimize wind turbine power output using the BA algorithm. Using the measurement data, the goal of the experiment is to find the optimum yaw and pitch angles of the front (upstream) wind turbine that maximize the total power output from an experimental setting with two wind turbines.

4.1 Scaled wind turbine model

The scaled wind turbine model, shown in Figure 3(a), is made of three aluminum blades with a length of 70 cm. The rotor diameter is 148 cm. The tower is made of a steel tube with a height of 113 cm. The blade pitch angles are collectively controlled by a servomotor (Dynamixel-64T). As shown in Figure 7(b), the rotation of the servomotor is transformed into linear motion, which rotates the blade angles through the mechanical linkage. The yaw angle is also controlled by a servomotor (Dynamixel-64T) through the mechanical pulley system, as shown in Figure 7(b). An AC generator, shown in Figure 7(b), is used to convert the mechanical energy into the electrical energy. The torque of the generator is controlled by changing the resistance (electrical load) connected to the generator. To change the load resistance value, a variable resistance controlled by Pulsed Width Modulation (PWM) signal is used.



(a) Scale wind turbine model

(b) Wind turbine control mechanism

Figure 3. Scaled wind turbine model

Figure 4 shows the circuit board used to execute the control actions for the wind turbine and to measure the power output from the wind turbine. The microcontroller (Arbotix-M), shown in Figure 4, is used for controlling the two servomotors and the variable resistance. The PWM signal generated from the microcontroller determines the value of the resistance, which in turn adjusts the torque of the generator. The AC power produced by the generator is converged into DC power by the rectifier shown in Figure 4. The rectified power is then measured using the voltage and the current sensors, from which the instantaneous power is computed. The power time series is continuously measured by the microcontroller, from which the average power is computed by the microcontroller. The microcontroller transmits the computed average power to the central node (laptop computer) through the Xbee radio module. Then, the BA algorithm, running in the central node, determines the next control inputs by proceeding to the next iteration of the BA algorithm using the transmitted power measurement. The determined control inputs are then wirelessly transmitted back to the microcontroller to change the blade pitch, the yaw, and the load resistance in the wind turbine.

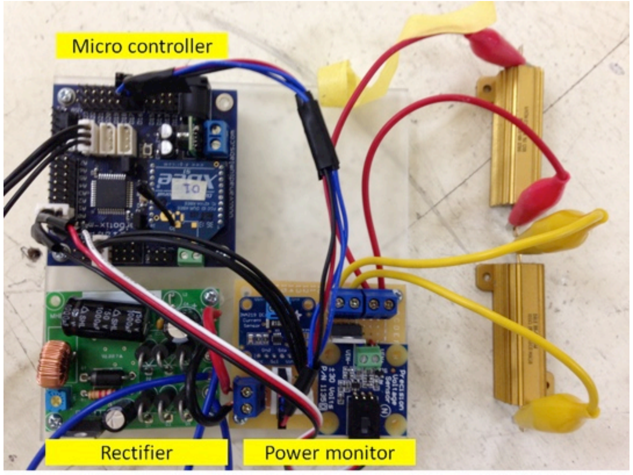
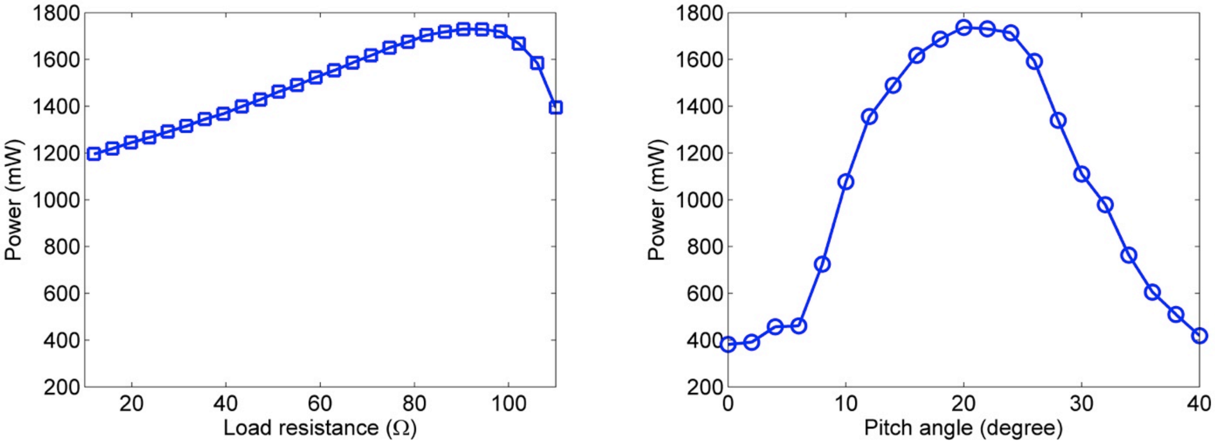


Figure 4. Wind turbine control and power monitoring board

4.2 Optimal control of a single wind turbine

First, the BA method is employed to determine the optimum pitch angle and the load resistance that maximize the power output from a single wind turbine. The wind speed is maintained as 4m/s in the wind tunnel experiment. Because the wind direction is fixed, the yaw angle is not controlled and is fixed at zero degree. Before applying the BA method, a series of tests are conducted to determine the optimum load resistance and the pitch angle that produces the maximum average power, which is computed using 30 sec power time series, from the wind turbine. As shown in Figure 5(a), the load resistance changes the power of the wind turbine. When the load resistance is around 90 Ω, the wind turbine produces the maximum power. Given the fixed optimum load resistance of 90 Ω, Figure 5(b) shows how the power changes with the blade pitch angle. When the blade pitch angle is zero, where the blade face is almost perpendicular to the wind flow, the power output is the lowest. As the blade angle increases, as shown in the figure, the power of the wind turbine keeps increasing until reaching the maximum at the pitch angle of 20 degrees. When the blade pitch angle exceeds the optimum, the power starts to decrease. Since the optimum load resistance is the same for all rotational speeds, we conclude that the blade pitch angle of 20 degree and the load resistance of 90 Ω maximize the power of the wind turbine at the given wind speed (4 m/s).

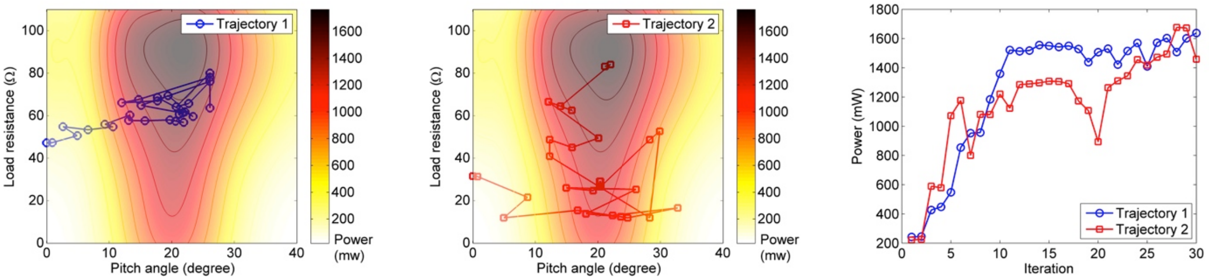


(a) Power variation by load resistance

(b) Power variation by pitch angle

Figure 5. Variations of the wind turbine power due to changes in the pitch angle and the load resistance

Knowing the optimum control actions for the wind turbine, the BA method is applied to see whether it can find the optimum control actions using the measured input and power output data. We first set limits on the two control inputs, blade pitch angle and the load resistance; the pitch angle and the load resistance are limited between 0° and 40° and between 0 Ω and 110 Ω, respectively. Within the limited values, two initial control actions are chosen, from each of which the BA method initiates the iterative control procedure. For each of the two initial control actions, Figure 6(a) and 6(b) show the input trajectories, histories of the sampled inputs by the BA algorithm. The contour plots shown in Figures 6(a) and 6(b) represent the wind turbine power corresponding to a combination of the blade pitch angle and the load resistance. The contour plot is constructed based on Gaussian Process regression using the input and output data measured during the wind tunnel experiments. The contour plot helps to visualize how the control actions determined by the BA algorithm reach the optimum coordinated actions (i.e., the peak of the contour). Figure 6(c) shows the measured wind turbine power corresponding to the two trajectories of control actions shown in Figure 6(a) and 6(b). The result shows that BA can increase the wind turbine power significantly by executing only a small number of trial control actions. Note that the BA control algorithm does not attain the true maximum possibly due to relatively flat surface for the power curve around the optimum point and the variability of the power measurement data.



(b) Control input trajectory 1 (b) Control input trajectory 2 (c) Power improvement

Figure 6. Maximization of the power for the wind turbine by Bayesian Ascent method

4.3 Cooperative wind turbine control

The BA method is now employed to maximize the total power output using two scaled wind turbine models. The objective is to derive the coordinated control actions of wind turbines using the input (control actions of the wind turbines) and output (the wind farm power) data. Figure 7 shows the two wind turbines separated at seven times of the rotor diameter (7D) apart. The control actions of the second (downstream) wind turbines are fixed at the optimum control settings with the yaw angle, the pitch angle and the load resistance of the second wind turbines at 0°, 20° and 90 Ω, respectively. The control actions of the first (upstream) wind turbine, namely the yaw and the blade pitch angles (with the load resistance fixed at 90 Ω), are then controlled to maximize the total power production.

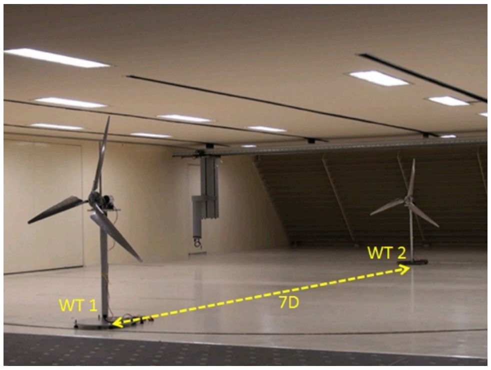
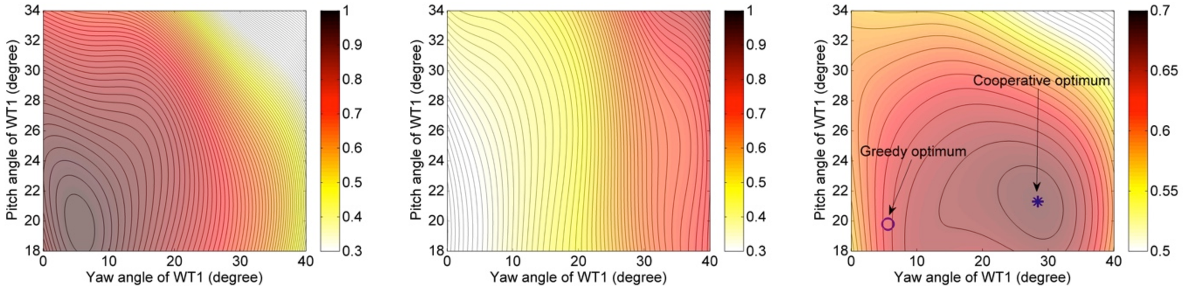


Figure 7. Wind turbine arrangements for cooperative wind turbine control.

Figure 8 shows how the yaw and the pitch angle of the upstream wind turbine affect individual powers generated by the two wind turbines and the total power obtained from the two wind turbines. For easy comparison, the powers are normalized with the maximum power that can be produced without wake interference. For example, the value of 0.7

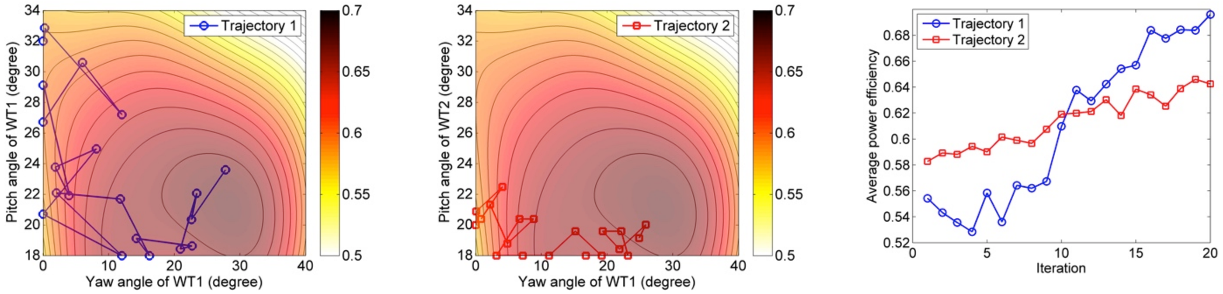
means that the wind turbine produces 70% of the power that is obtainable when there is no wake interference. As shown in Figure 8(a), there clearly exist optimum control actions that maximize the upstream wind turbine (greedy control action). These greedy control actions by the upstream wind turbine, however, significantly lower the power production by (see Figure 8(b)). As the yaw and the blade pitch angles of the upstream wind turbine (WT 1) becomes far from the optimum, as shown in the figure, the power from the upstream wind turbine decreases, whereas the power from the downstream wind turbine (WT 2) increases. The contour plot shown in Figure 8(c) represents the wind farm power efficiency, i.e., the total power production by the two wind turbines normalized by the maximum power obtainable without wake interference. As shown in the figure, the decrease of the power for the upstream wind turbine is compensated by the increase in the power produced by the downstream wind turbine. The maximum efficiency is obtained when the control actions of the upstream wind turbine (WT 1) is set at the yaw angle around 30° and the blade pitch angle around 21°.



(a) Power for WT 1 (b) Power for WT 2 (c) Total power

Figure 8. Variation of powers due to the control actions of WT 1.

The BA method is employed to find the optimum coordinated control actions that maximize the total power produced by the two wind turbines. Figure 9(a) and 9(b) shows, respectively, the two trajectories of the wind turbine control actions executed during the BA iterations started from two separate initial control actions. The trajectories are shown overlaying the contour plots for the total power efficiency for the two wind turbines to illustrate how the BA method reaches the optimum. As shown in the figures, after 20 iterations, the BA method reaches the solutions that are near the optimum. The wind farm power efficiency corresponding to the executed control actions are shown in Figure 9(c). Note that the measured average power has relatively large variance, i.e., noise level, due to the fluctuation in the power time series as well as the wind speed (turbulence), which makes it hard to for the BA method to follow the ascending direction of the power curve. Nevertheless, starting from some initial (control actions) point, the results show that the BA algorithm can increase the total power generated by the two wind turbines even with noisy measurement data.



(a) Control input trajectory 1 (b) Control input trajectory 2 (c) Power improvement

Figure 9. Maximization of the sum of powers from the two wind turbines by Bayesian Ascent method

5. SUMMARY AND DISCUSSION

This paper discusses the development of the Bayesian Ascent (BA) method that can rapidly find an optimum value of a target system using small number of noisy measurements. Using Gaussian Process regression, the BA method can model the complex input and output relationships of the target system using a small number of data points. The constructed GP model function is then exploited to find the next input that can increase the target value with the highest probability. In selecting the next input, the BA algorithm imposes a trust region constraint so that the next input is selected from the region near the best input observed so far. Due to the regularized search step, the BA algorithm is able to increase the target value in a monotonic manner by gradually changing the inputs. The size of trust region is also adjusted at each iteration to expedite the convergence rate. One additional advantage of the BA method is that the sampled inputs are often clustered in a small region rather than a widely spread out region; this feature makes it possible for GP to estimate the hyper-parameter of the Kernel function (i.e., local structure of the target function) reliably.

The BA method is employed, in a laboratory setting, to maximize the power production of a single wind turbine. Result shows that, starting from two different initial random control actions, the BA method gradually changes the control actions to increase the wind turbine power production. In addition, the BA method is applied to maximize the total power production from two wind turbines. In this case, the control actions of the upstream wind turbine are gradually changed while the control actions of the downstream wind turbines are fixed. Result shows that the BA method can identify suboptimal coordinated control actions that reduce the power from the upstream wind turbine but increase the total power production by adjusting wake interaction pattern. Here, the two scaled wind turbines are used to represent a physical system whose governing relationships is difficult to model, and thus the optimal operational conditions for the system are difficult to identify. Furthermore, the experiments illustrated how a probabilistic data-driven optimization can efficiently process the noisy measurement data.

The BA method integrates the strengths of Bayesian Optimization and Trust Region Optimization to improve a target value rapidly with a limited number of the input and output data. At the expense of achieving such advantages, the BA method inherits the weaknesses of these two methods as well. For example, the BA method can become computationally expensive for building the GP regression model because the BA method needs to construct a kernel matrix and to compute its inversion at each iteration, which has computational complexity proportional to the cube of the number of data points. However, this generally does not cause an issue because BA is aiming to optimize a target system using a small number of input and output, say less than a hundred. Furthermore, there exists an efficient method to represent the kernel matrix using a subset of the data points. On the other hand, BA method is likely to find only a local optimum close to the initial point due to the use of the trust region constraint. Noting that there doesn't exist yet an efficient method that can locate the global optimum using only a small number of data, a good strategy is to focus on finding a local optimum as quickly as possible, while minimizing the cost associated with exploration.

ACKNOLWEDGEMENT

The authors would like to thank Prof. Soon-Duck Kwon of Chonbuk National University in Korea, who provides the wind tunnel laboratory (KOCED Wind Tunnel Center) facilities for validating the BA method with the scaled wind turbine models.

REFERENCES

[1] Johnson, K. and Thomas, N., "Wind farm control: addressing the aerodynamic interaction among windTurbines," Proceedings of the American Control Conference, pp. 2104–2109 (2009).

[2] Madjidian, D. and Ranter, A., "A stationary turbine interaction model for control wind farms," Proc. 18th World Congress of the International Federation of Automatic Control, pp. 4921–4926 (2011).

[3] Marden, J., Ruben, S. and Pao, L., "A model-free approach to wind farm control using game theoretic methdos," IEEE Transactions on Control Systems Technology, 21(4), pp. 1207–1214 (2013).

[4] Gebraad, P. M. O. and Wingerden, J. W., "Maximum power-point tracking control for wind farms," Wind Energy, 18(3), pp. 429-447 (2015).

[5] Wagenaar, J., Machielse, L., and Schepers, J., "Controlling wind in ECN's scaled wind farm," Proc. Europe Premier Wind Energy Event, pp.685–694 (2012).

[6] Dahlberg, J. and Medici, D.,"Potential improvement of wind turbine array efficiency by active wake Control," Proc. European Wind Energy Conference, pp.65–84 (2003).

[7] Gebraad, P., Teeuwisse, F., Wingerden, J., Fleming, P., Ruden, S., Marden, J. and Pao, L., "Wind plant power optimization through yaw control using a parametric model for wake effects-a CFD simulation study," Wind Energy, DOI: 10.1002/we.1822 (2014).

[8] Fleming, P., Gebraad, P, Lee, S., Wingerden, J., Johnson, K., Churchfield, M. and Moriarty, P., "Simulation comparison of wake mitigation control strategies for a two-turbine case," Wind Energy, DOI: 10.1002/we.1810 (2014).

[9] Jensen, N. O., "A Note on Wind Generator Interaction," Tech. Rep. Risø-M2411(EN), Risø National Laboratory (1983).

[10] Katic, I., Hojstrup, J. and Jensen, N., "A simple model for cluster efficiency," Proc. of European Wind Energy Association Conference and Exhibition, pp. 407–410 (1986).

[11] Bronchu, E., Cora, M. V. and Freitas, N.,"A tutorial on Bayesian optimization of expensive cost functions, with application to active user modelling and hierarchical reinforcement learning," Tech. Rep. arXiv:1012.2599, Department of Computer Science, University of British Columbia (2010).

[12] Osborne, M., [Bayesian Gaussian process for sequential prediction, optimisation and quadrature], PhD Dissertation, University of Oxford (2010).

[13] Rasmussen, C.E. and Williams, C.K., [Gaussian processes for machine learning], MIT Press, Cambridge, MA (2006).

[14] Neal, R. M., [Bayesian learning for neural networks], Springer-Verlag, New York, NY (1996).

[15] Kapoor, A., Grauman, K., Urtasun, R. and Darrell, T., "Active learning with Gaussian Process for object Categorization," Proceedings of IEEE International Conference on Computer Vision, pp. 1–8 (2007).

[16] Dennis, D. and Susan, J., "A statistical method for global optimization," Multidisciplinary design optimization: state of the art, 315–329 (1997).

[17] Mockus. J., Fretitas, A. and Castelanous, J. A., [Towards global optimization], Elsevier (1978).

[18] Nocedal, J. and Wright, S., [Numerical Optimization], Springer (2000).